iBAT: Detecting Anomalous Taxi Trajectories from GPS Traces

Daqing Zhang[†], **Nan Li**[‡,♮], **Zhi-Hua Zhou**[‡], **Chao Chen**[†], **Lin Sun**[†], **Shijian Li**[♭]
[†] Institut TELECOM SudParis, France
{daqing.zhang, chao.chen, lin.sun}@it-sudparis.eu
[‡] National Key Laboratory for Novel Software Technology, Nanjing University, China
{lin, zhouzh}@lamda.nju.edu.cn
[♮] School of Mathematical Sciences, Soochow University, China
[♭] Department of Computer Science, Zhejiang University, China shijianli@zju.edu.cn

ABSTRACT
GPS-equipped taxis can be viewed as pervasive sensors and the large-scale digital traces produced allow us to reveal many hidden "facts" about the city dynamics and human behaviors. In this paper, we aim to discover anomalous driving patterns from taxi's GPS traces, targeting applications like automatically detecting taxi driving frauds or road network change in modern cites. To achieve the objective, firstly we group all the taxi trajectories crossing the same source-destination cell-pair and represent each taxi trajectory as a sequence of symbols. Secondly, we propose an Isolation-Based Anomalous Trajectory (*iBAT*) detection method and verify with large scale taxi data that *iBAT* achieves remarkable performance (AUC>0.99, over 90% detection rate at false alarm rate of less than 2%). Finally, we demonstrate the potential of iBAT in enabling innovative applications by using it for taxi driving fraud detection and road network change detection.

Author Keywords
Anomalous trajectory detection, GPS trace, isolation-based anomaly detection, taxi

ACM Classification Keywords
H.2.8 Database applications: Data mining.

General Terms
Algorithms

INTRODUCTION
With recent advances in sensing, communication, storage and computing, the digital traces left by people while interacting with cyber-physical spaces are accumulating at an unprecedented rate. The scale and richness of different digital traces provides us with new opportunities to understand society behaviours and community dynamics in different contexts, showing great potential to revolutionize the services in various areas ranging from public safety, urban planning to transportation management [10, 27].

In modern cities, more and more vehicles, such as taxis, have been equipped with GPS devices for localization and navigation. Gathering and analyzing the large-scale GPS traces have provided us a great opportunity to reveal the hidden "facts" about the city dynamics and human behaviors, enabling diverse innovative applications [22, 13, 18, 30, 26, 21, 23, 28, 24]. Recent years have witnessed an increasing interest in trajectory anomaly detection [14, 6, 9], which aims to detect suspicious moving objects automatically. However, while several aspects of abnormality of moving objects have been investigated, there are very few works on discovering anomalous driving patterns by mining GPS traces with practical applications examined. In this paper, we intend to motivate our research on anomalous taxi driving trajectory detection with the following potential applications:

EXAMPLE 1. Many people, mostly tourists, are victims of taxi driving frauds committed by greedy taxi drivers who overcharge passengers by deliberately taking unnecessary detours. To ensure quality taxi services, it is crucial to detect and penalize such frauds. Currently, detecting taxi driving frauds is often done by experienced staff via manually checking the GPS trajectories corresponding to the taxi rides, based on complaints from passengers, but this is costly and not very effective because many frauds are not even noticed by passengers. As the traces of driving frauds often significantly deviate from normal ones, it is possible to automatically detect the anomalous driving trajectories by mining taxi GPS traces and hence taxi driving frauds.

EXAMPLE 2. Urban road networks often change over time in developing cities, it is important to update these changes in the digital map. If this is done manually by digital map providers, it would be expensive and also difficult to capture the changes in time. If GPS-equipped taxis are viewed as moving sensors probing the real-time information about urban road network, then the taxi traces accumulated in a new and different area might indicate a sudden road network

UbiComp'11, September 17–21, 2011, Beijing, China.

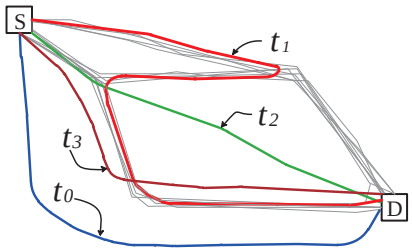


Figure 1. An illustration of taxi trajectories between S and D.

change, *i.e.* a newly-built or blocked road segment nearby. Hence, detecting anomalous taxi driving trajectories can be helpful in identifying road network changes promptly.

Consider the taxi trajectories between two places S (source) and D (destination) as shown in Fig. 1. Assume that the three clusters of trajectories between the $(S; D)$ pair are defined as normal ones, then the four trajectories (t_0, t_1, t_2, t_3) are considered as anomalies since they are "*few*" and "*different*" from the normal ones. Detecting driving anomalies is a non-trivial task because of the following challenging issues.

- First, as shown in Fig. 1, there might be different sets of normal trajectories between each $(S; D)$ pair and these trajectory clusters usually have different densities or distance distributions. If we exploit traditional anomaly detection techniques [14, 6, 9] based on distance or density, it is hard to choose the parameters and identify all anomalous trajectories.

- Second, multiple normal routes between each $(S; D)$ pair also mean different driving distances. If we directly model driving distance for anomalous trajectories detection, it is not able to discover those anomalies whose driving distance is close to that of the normal trajectories (like t_3).

- Third, the road network often changes over time in developing cities: a new (anomalous) route may become normal and an old road segment can be blocked. Hence, it is important to be able to detect an emerging cluster of anomalous trajectories and incorporate these changes in the model.

- Finally, some traditional anomaly detection methods often require that the taxi trajectories be represented as fixed-length feature vectors. However, the real taxi trajectories are variable-length sequences of points, thus traditional methods can not be directly used. If we transform them into fixed-length feature vectors, spatial information can be lost. 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 driving trajectory detection method which addresses the four challenges above. Firstly, we extract valid taxi rides from all the taxi GPS traces, split the city map into grid-cells of equal size, group all the taxi rides crossing the same source-destination cell-pair, and augment and represent each taxi trajectory in each source-destination pair as an ordered sequence of traversed cell symbols. In such a way, the problem of anomalous driving trajectory detection is converted to that of finding anomalous trajectories from all the trajectories with the same source-destination cell pair. Secondly, for all the taxi trajectories between a certain source-destination cell-pair, we define those trajectories that are "*few*" and "*different*" from the normal trajectory clusters as anomalies. Instead of profiling the normal trajectories and detecting the anomalies by employing the similarity or density measure, this paper proposes an Isolation-Based Anomalous Trajectory (*iBAT*) detection method which exploits the property that anomalies are susceptible to a mechanism called *isolation* [20]. Finally, we perform an empirical evaluation of *iBAT* with real-world taxi GPS data and show how the two applications (*i.e.*, taxi driving fraud detection and road network change detection) can be enabled by using *iBAT*. In summary, the main contributions of this paper include:

- We identify a new kind of anomalous trajectory detection problem based on two motivating applications with taxi GPS traces. We further propose a series of techniques to transform the problem of anomalous driving trajectory detection into an easy-to-solve form: finding anomalous trajectories from all the trajectories with the same source-destination cell pair, with each taxi trajectory represented as a sequence of cell symbols.

- To solve the above mentioned problem, we propose an Isolation-Based Anomalous Trajectory (*iBAT*) detection method which exploits the property that anomalies are susceptible to a mechanism called *isolation*. To our best knowledge, this is the first work applying the isolation mechanism in the trajectory anomaly detection.

- We evaluate *iBAT* with real-world GPS traces collected from 7,600 taxis for one month. It achieves remarkable detection rate with low processing-time, it also outperforms the density-based method as a baseline approach in terms of AUC (*i.e.*, the Area Under the ROC Curve) [4].

- By using two examples (*i.e.*, taxi driving fraud detection and road network change detection), we show how innovative applications can be achieved by using *iBAT*.

RELATED WORK

In this section, we briefly review the related work which can be grouped into three categories. The first category includes the work on analyzing or exploiting GPS traces with research issues other than anomaly detection. For instance, Patterson, Liao, et al. [22, 18] used GPS traces to infer an individual's mode of transportation and daily routine, providing reminders for persons with mild cognitive disabilities when they, for instance, take the wrong bus; Krumm et al. [13, 8] showed it is possible to predict the destination and entire route of a vehicle based on historical GPS traces, and recently reported further results building routable road networks from raw GPS traces [7]. Based on the observation that taxi drivers are experienced in finding the best route to a destination, Ziebart et al. [30] and Yuan et al. [26] designed *PROCAB* and *T-Drive*, respectively, providing driving direction guidance by leveraging taxis' GPS traces. Zheng et

al. [29] gave travel location recommendation to new visitors by mining GPS data left by previous tourists. Liu et al. [21] developed a novel methodology to reveal taxi drivers' operation patterns by analyzing their GPS traces. Zheng et al. [28] mined GPS traces to discover interesting locations and possible activities that can be performed there for recommendations. Phithakkitnukoon et al. [23] derived a model to predict vacant taxis. Qi et al. [24] measured social functions of city regions by analyzing taxi's GPS traces. Li et al. [15] studied the passenger-finding strategies of taxi drivers. While these works target research problems different from ours, they have inspired us to motivate our research with innovative applications using taxi GPS traces.

The second category focuses on anomalous trajectory detection, which is highly related to our work. In the literature, some solutions for trajectory outlier detection have already been reported, each addressing certain aspects of abnormality. For instance, Lee et al. [14] proposed the partition-and-detect framework to detect outlying sub-trajectories, they used both distance and density for anomaly detection. For a similar problem to Lee et al.'s, Ge et al. [9] detected the evolving trajectory outliers by computing the outlying score, based on the evolving direction and density of trajectories. Bu et al. [6] presented an outlier detection framework for monitoring anomalies over continuous trajectory streams. The key idea is to build local clusters upon trajectory streams and detect anomalies by a cluster join mechanism. Li et al. [17], instead, developed a temporal outlier detection approach for vehicle traffic data, which aimed to discover the abnormal traffic change in the road network. In addition, there are also some learning-based approaches reported [16, 25, 19], but they all require training data which is expensive to label. In contrast, we define a different trajectory anomaly problem from previous ones, *i.e.*, given all the taxi trajectories between a certain source-destination cell-pair, our objective is to discover those trajectories that are "*few*" and "*different*" from the normal trajectory clusters.

The third category includes anomaly detection methods that are not designed for trajectory data. This category of work has its deep roots in database, data mining, computer vision, etc. The proposed methods range from supervised approaches [1] to distance-based [12, 3], density-based [5], and model-based methods [11], each having a specific formulation of the problem with different anomaly measures and notions. Among these approaches, one new and fundamentally different anomaly detection method directly inspiring this work is *iForest* [20]. By assuming anomalies are "*few and different*", they found that anomalies are susceptible to a mechanism called "*isolation*", with which *iForest* can detect anomalies without employing any distance or density measure. It has been shown that *iForest* outperforms the state-of-the-art outlier detection techniques like one-class SVM in terms of AUC and processing time, at a low time and space complexity. It also deals with the effects of swamping and masking effectively. However, as *iForest* is designed for anomaly detection problems with fixed-length feature vector as input, it can not be directly applied to handle the trajectory data. In this paper, we first adapt the idea of *iForest* to trajectory data, and improve it by considering the characteristics of the sequence of trajectories so that it can manage more anomalous cases.

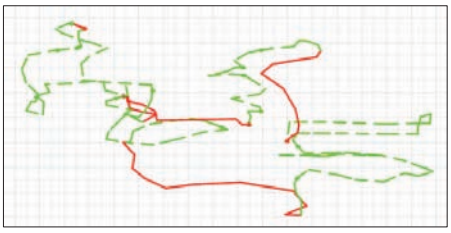


Figure 2. One taxi's GPS trace in two hours, where red (solid) or green (dashed) indicates the taxi is occupied or vacant.

PROBLEM STATEMENT

A taxi's GPS trace consists of a sequence of GPS points (*i.e.*, latitude and longitude) generated by the taxi GPS device, the time stamp, the estimated speed and the operation status (*i.e.*, vacant or occupied) associated with each time stamp. On a two dimensional plane, a taxi's moving trajectory over a time interval can be depicted by connecting these GPS points. For instance, Fig. 2 shows one taxi's moving trajectory in two hours, where red (solid) lines and green (dashed) lines correspond to the taxi's occupied and empty status, respectively. As the taxi driving fraud occurs only when the taxi is occupied by passengers, in this paper we only consider the taxi trajectories corresponding to the taxi rides (red lines, where taxi's operation status is occupied).

DEFINITION 1. A *taxi trajectory* is a sequence of GPS points pertaining to an occupied taxi ride, *i.e.*,

$$t: \ p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_n \ ,$$

where p_i's are GPS points, p_1 and p_n are *source* and *destination* of the trajectory, respectively.

By extracting valid taxi rides from all the taxi GPS traces based on operation status, we can obtain a large collection of taxi trajectories. If we split the city map into equal sized grid-cells and group all the taxi trajectories crossing the same source destination cell-pair, then we can have many taxi trajectories between two cells (*e.g.*, S and D) as shown in Fig. 1. Assume that the three clusters of trajectories (the majority of trajectories with similar routes) correspond to normal taxi rides, the rest of the scattered trajectories are considered to be anomalous. The anomalous trajectories can be long detours made by greedy taxi drivers like $t_0$and t_1 in Fig. 1, they can also be short-cuts or new routes taken by experienced drivers like t_2 and t_3. Our goal is to find these anomalous trajectories, based on which new applications can be enabled. Formally, the problem is defined as follows.

PROBLEM. Given a set of trajectories $T = \{t_1, t_2, \ldots, t_n\}$ between two locations S and D, find those in T that are significantly different from the majority.

In this paper, by exploiting the "*few and different*" properties of anomalous trajectories, we propose an isolation-based anomalous trajectory detection method.

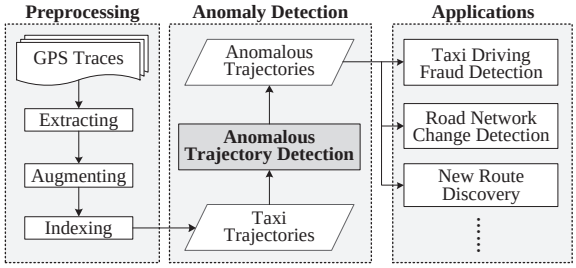


Figure 3. Overview of our approach.

OUR PROPOSED APPROACH

A Three-Step Procedure

As shown in Fig. 3, our approach to detecting anomalous taxi trajectories consists of three main steps. In the first step, after extracting taxi trajectories from GPS traces, we split the city map into grid-cells of equal size, then we group all taxi trajectories crossing the same source-destination cell-pair, augment and represent each trajectory as a sequence of traversed cells.In the second step, we present our *iBAT* method to discover anomalous trajectories for a specific source destination cell-pair. Specifically, we exploit the "*few and different*" properties of anomalous trajectories, and apply the isolation mechanism to detect anomalous trajectories. Finally, we perform further analysis on detected anomalous trajectories, and demonstrate their effectiveness in enabling the two innovative applications, *i.e.*, taxi driving fraud detection and road network change detection. We describe each step of our approach below.

Preprocessing GPS traces

Extracting and Augmenting Taxi Trajectories

Given a large collection of GPS traces, our first task is to extract taxi ride trajectories by partitioning GPS traces according to the taxi operation status. As we split the city map into grid-cells of equal size, then each taxi trajectory is mapped to the cell grid and become a sequence of traversed cells.

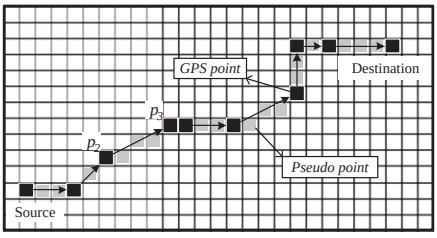


Figure 4. An illustrative example of augmenting trajectory.

In practice, GPS devices usually report data at a low frequency, for example, about one record per minute in our real-world dataset. This results in a non-detailed representation of taxi trajectories, because one taxi may traverse multiple consecutive cells without GPS points recorded. As shown in Fig. 4, due to the low-sampling-rate problem, a taxi trajectory often consists of a series of GPS points (shown as black cells) which are not adjacent to each other (like p_2 and p_3). In such a way, if two taxis take the same route, their trajectories can be different. In order to ensure that same taxi trajectories are represented equally in the system, we need to argument the trajectory. In this paper, we take a simple method to augment taxi trajectories. That is, along the line defined by two GPS cells, we insert pseudo cells between them. For example, in Fig. 4, three pseudo cells (shown as gray cells) are inserted between p_2 and p_3 . Eventually, such an augmenting process will allow us to obtain a cascaded cell sequence for the representation of each trajectory.

Finding Trajectories between a Source-Destination Cell-Pair

In practice, the extracted taxi trajectories have various sources and destinations. Given a source-destination cell-pair, we need to find all the related taxi trajectories. For a given period of time and grid cell size, we may face the problem that there are not sufficient taxi trajectories between certain source-destination pairs to form "normal" trajectory clusters. To alleviate this problem, we not only count the set of taxi trajectories with exactly the same source-destination pair we also add all those taxi trajectories which pass through the source-destination cell-pair. For example, given two taxi trajectories

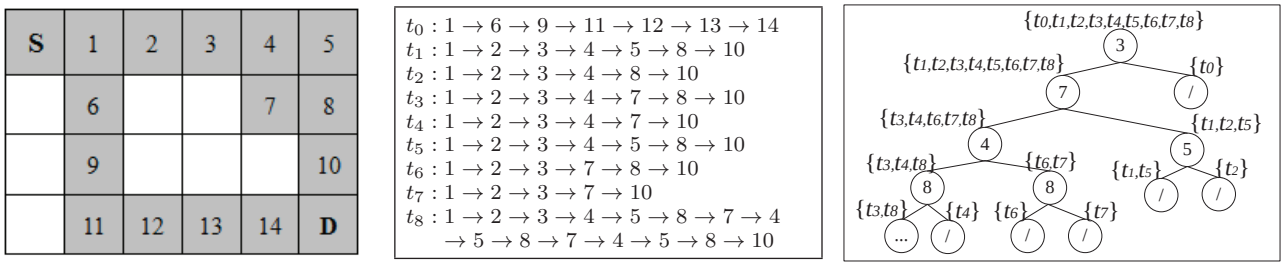
$$t_1 : p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_5,$$
$$t_2 : p_3 \rightarrow p_1 \rightarrow p_4 \rightarrow p_5 \rightarrow p_2,$$

both of them will be counted if we want to find trajectories with p_1 and p_5 as source and destination (for t_2, only the segment $p_1 \rightarrow p_4 \rightarrow p_5$ is included). To achieve this, we employ the inverted index mechanism [31], which is popularly used in information retrieval, to index all the taxi trajectories, As an analogy, the taxi trajectory and cell correspond to the document and word [31], respectively. As an illustrative example, given t_1 and t_2 as above, we have the inverted index as

$$p_1 : \{(t_1, 1), (t_2, 2)\}, \quad p_2 : \{(t_1, 2), (t_2, 5)\},$$
$$p_3 : \{(t_1, 3), (t_2, 1)\}, \quad p_4 : \{(t_2, 3)\},$$
$$p_5 : \{(t_1, 4), (t_2, 4)\},$$

where, for instance, $p_1 : \{(t_1, 1), (t_2, 2)\}$ means that p_1 appears in t_1 and t_2 at the 1st and 2nd place, respectively. Then, if we want to get trajectories whose source and destination are p_1 and p_5, both t_1 and t_2 will be sorted out because both p_1 and p_5 appear in t_1 and t_2 in the correct order (*i.e.*, $1 < 4$ and $2 < 4$); but if p_1 and p_3 are the concerned source and destination, only t_1 will be returned because the order of p_1 and p_3 is incorrect in t_2 although they both appear in in t_1 and t_2.

As the number of taxi trajectories between two cells is a function of time given a fixed cell size, apparently another effective way to get more taxi trajectories is to use more historical data (with longer period of time). As the focus of the paper is on the anomalous trajectory detection method, here we just simply assume that we have sufficient taxi trajectories to form "normal" routes between our interested source and destination cell-pair. Moreover, we can further alleviate this problem by using more historical data. It is worth noting that all above operations allow online updates, thus newly generated taxi trajectories can be easily incorporated.



(a) traversed cells (b) trajectories (c) isolation tree for trajectory

Figure 5. An illustrative example of isolation tree for trajectories partition. (a) cells that are traversed by trajectories (gray, 1 to 14); (b) nine trajectories from S to D, where t_0 and t_8 are significantly different from others; (c) an isolation tree, where the randomly selected cell is shown at each node, the anomalous trajectory t_0 has much shorter path length than other trajectories.

$$t_0 : \{t_1, \ldots, t_8\} \xrightarrow{12} \emptyset$$
$$t_8 : \{t_0, \ldots, t_7\} \xrightarrow{5} \{t_1, t_5\} \xrightarrow{7} \emptyset$$
$$t_3 : \{t_0, t_1, t_2, t_4, \ldots, t_8\} \xrightarrow{3} \{t_1, t_2, t_4, \ldots, t_8\} \xrightarrow{2} \{t_1, t_2, t_4, \ldots, t_8\} \xrightarrow{7} \{t_4, t_6, t_7, t_8\} \xrightarrow{8} \{t_6, t_8\}$$

Figure 6. An example of eliminating trajectories by randomly selecting cells from the concerned trajectory, where at each step only the ones that passed the selected cell (superscript of arrow) are kept.

Isolation-Based Anomalous Trajectory Detection

Now, we present our proposed *iBAT* (*i.e.*, Isolation-Based Anomalous Trajectory detection) method.

Characterizing Anomalous Trajectories

Before trying to detect anomalous trajectories, we need to characterize anomalous trajectories precisely. This is not straightforward in practice, because anomalous trajectories can be varied and normal ones can fall into several different clusters (as illustrated in Fig. 1).

Instead of using distance or density measure, we exploit the following two intrinsic properties of anomalous trajectories:

1. anomalous trajectories are *few* in number ;
2. they are *different* from the majority, in particular, they pass *different* locations, or pass similar locations in *different* orders.

Since anomalous trajectories are "*few and different*", normal ones are "*many and similar*". Hence, it is not difficult to find that separating a normal trajectory from the rest requires more effort, since there are "*many*" "*similar*" ones; while anomalous trajectories are easier to be separated from the majority of the trajectories, *i.e.*, they are susceptible to *isolation*. This constitutes the basis of our proposed method.

Adapting iForest for Trajectory Data

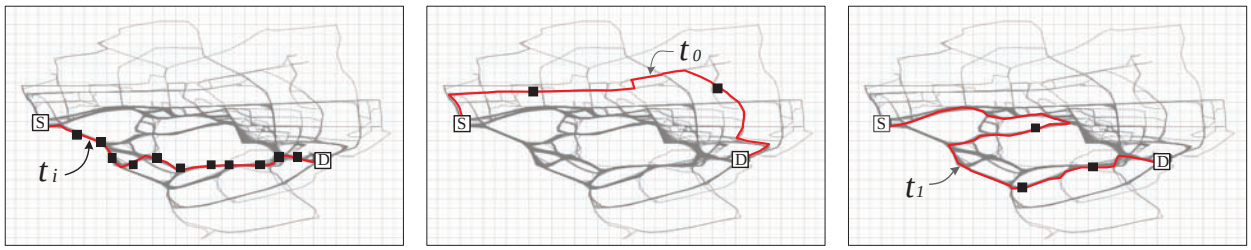
Isolation Forest (*iForest*) [20] is a novel anomaly detection method, which adopts a fundamentally different approach to take advantage of anomalies' intrinsic properties of being "few and different". It applies a data-induced random tree to partition all the instances until all of them are isolated (*iTree*). This random partitioning produces noticeable shorter paths for anomalies and long paths for normal instances. When a forest of *iTree*s collectively produce shorter path lengths for some particular points, then these instances are highly likely to be anomalies. By exploiting sub-sampling, *iForest* achieves state-of-the-art performance with low linear time-complexity and small memory-requirement. Unfortunately, *iForest* is designed in the traditional anomaly detection framework where instances are fixed-width vectors, thus it cannot be directly used for anomalous trajectory detection.

Noting that anomalous trajectories often contain points (cells) that are not always contained by other trajectories, we adapt *iForest* to handle our trajectory data based on this property. Specifically, we use a randomly selected cell to recursively divide the data in each node of the *iTree* until the node has only one trajectory or all trajectories at the node are the same. The trajectories that have short path lengths in *iTree* are suspicious to be anomalous. For example, Fig. 5 (b) shows a set of nine trajectories, among which t_0 is significantly different from others; Fig. 5 (a) shows all the 14 cells (gray) that traversed by at least one trajectory; Fig. 5 (c) shows an adapted *iTree*, where a cell is randomly selected at each node for partition. If the trajectory contains this cell, it falls into its left child node, else right child node. For instance, the cell 3 is selected at root node, then t_0 falls to its right child node since it does not pass the cell 3. Finally, we can obtain an *iTree* in Fig. 5 (c), where the path length of the anomalous trajectory t_0 is 1, much smaller than other trajectories. Consequently, after such an adaption, *iForest* can be used for anomalous trajectory detection.

This adapted *iForest* can detect anomalous trajectories that traverse different cells from the majority; but it fails if anomalous trajectories detour in cells that are always contained by other trajectories. For example, the trajectory t_8 is definitively an anomalous trajectory since it detours a lot in the top right area of Fig. 5 (a); but if we build *iTree* as Fig. 5 (c), the path length of t_8 will be not smaller than any other trajectory, which suggests it is a normal trajectory. Obviously, this results is not correct, and a better method is needed.

iBAT: Applying Isolation in a Lazy Learning Manner

Now, we present the *iBAT* method which improves the adapted *iForest* via lazy learning [2]. Lazy learners do not train a model until presented with a test sample, they usually achieve



(a) Isolating normal trajectory t_i (b) Isolating anomalous trajectory t_0 (c) Isolating anomalous trajectory t_1

Figure 7. Anomalous trajectories are easier to be isolated. In each sub-figure, the test trajectory is highlighted in solid red, while others are in dashed gray. Given a set of 593 trajectories from S to D, (a) a normal trajectory t_i requires 11 randomly selected cells (solid black cells) to be isolated; (b) the anomalous trajectory t_0 requires only 2 randomly selected cells to be isolated. (c) the anomalous trajectory t_1 requires only 3 cells.

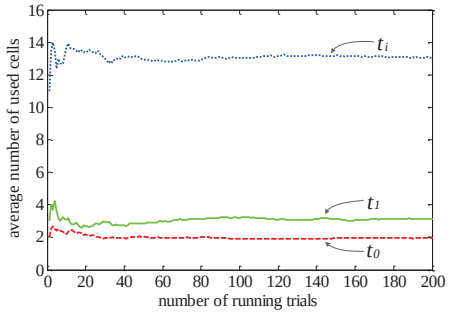


Figure 8. Average number of cells used for isolation, where as in Fig. 7, t_i is a normal trajectory, t_0 and t_1 are anomalous.

competitive performance in practical problems, and can trivially adapt to evolving training data.

In particular, rather than building *iTree* based on all the trajectories at hand, *iBAT* focuses on the test trajectory, and tries to separate it from the rest trajectories by randomly selecting cells solely from the test trajectory. For example, let $T = \{t_0, \ldots, t_8\}$ where t_i's are defined in Fig. 5 (b). If t_0 is a test trajectory, we randomly select one cell from t_0 and remove the trajectories that do not pass the selected cell from the set $T \backslash \{t_0\}$, this process is repeated until no trajectory is left or all the trajectories left contain all the cells t_0 has. As shown in Fig. 6, t_0 is separated from other trajectories after one test; t_8 is isolated after two tests; while t_3 is not separated even after selecting four cells. This indicates that t_0 and t_8 are anomalous, while t_3 is normal. Thus, we can see that the proposed method properly addresses the problem of detecting anomalous trajectories that detour in cells that are always contained by other trajectories (like t_8). In addition, for the loop detours, like t_8 that loops at 4, 5, 8, 7 and $t_9 = 1 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 8 \rightarrow 10$ that loops at 2, 3, it is easy to detect them simply by counting the number of each passed cell.

In Fig. 7, we show a real-world example of isolating trajectories, where the isolation is performed by randomly selecting cells from the test trajectory. It can be observed that a normal trajectory t_i requires 11 randomly selected cells to be isolated, while anomalous trajectory t_0 requires only two cells and another anomalous trajectory t_1 requires three cells to be isolated. In particular, t_0 requires less cells to be isolated because it contains cells that are not always contained by other trajectories, t_1 is easy to be isolated because it passes cells in a quite different order although these cells are always contained by other trajectories. Subsequently, we denote the number of cells used for isolating the trajectory t as $n(t)$.

Since each cell is randomly selected in each individual isolation process, we estimate the expected number of used cells by averaging the number of used cells over multiple isolation processes. Fig. 8 shows that the average number of used cells for t_i, t_0 and t_1 when the number of isolation processes increases. After repeating the random isolation process for 200 times, we can see that the average number of cells needed to isolate t_i converges to 13.11, while those of t_0 and t_1 are both less than 3. This shows that the average number of cells used for isolating anomalous trajectories is less than the average used for that of isolating normal ones.

Following [20], we define the *anomaly score* of a trajectory t as a normalization of the average number of used cells, *i.e.*,

$$s(t, N) = 2^{-\frac{E(n(t))}{c(N)}}, \tag{1}$$

where $E(n(t))$ is average number of cells used for isolating t, N is the number of trajectories from which we separate t, and $c(N)$ is the average of $n(t)$ given N (it equals to the average path length of unsuccessful searches in a binary search tree). In particular,

$$c(N) = 2H(N-1) - 2(N-1)/N,$$

where $H(i)$ is the harmonic number that can be estimated as $ln(i) + 0.57721566$ (Euler's constant). Obviously, when $E(n(t)) \rightarrow 0$, $s(t, N) \rightarrow 1$, meaning that t is definitely an anomalous trajectory; when $E(n(t)) > c(N)$, $s(t, N) < 0.5$, meaning that t can safely be categorized as a normal trajectory.

In practice, given a large amount of of taxi trajectories, we do not need to isolate our trajectory t from the rest, and the isolating process is able to work well with a small sub-sample of all trajectories. Specifically, given one trajectory, our *iBAT* method repeats the isolation process on different sub-samples of all trajectories and computes the anomaly score according to Eq. 1. The pseudo code of the *iBAT* method is summarized in Algorithm 1. Besides the test trajectory t and the set of trajectories T, it has two parameters, *i.e.*, number of running trials m and sub-sample size ψ. Unless otherwise stated, we use $m = 100$ and $\psi = 256$ in our experiments. An analysis on these two parameters can

Algorithm 1 The *iBAT* method

Input: t – test trajectory
T – set of trajectories to be separated from
m – number of running trials
ψ – sub-sample size
Process:
let n be a vector of m zeros (n_i's are zeros)
for $i = 1$ to m **do**
$T' \leftarrow$ randomly sample ψ trajectories from T
repeat
$n_i \leftarrow n_i + 1$
randomly choose a cell p from t
$T' \leftarrow$ select the trajectories that include p from T'
until T' is empty
end for
compute $s(t, \psi)$ according to Equation 1
Output: anomaly score $s(t, \psi)$

be found in the experiments which shows that the performance is nearly optimal at this setting and insensitive to a wide range of values. Furthermore, each isolation process is independent, thus they can be performed in parallel, making *iBAT* suitable for parallel processing.

Discussion

By applying the isolation mechanism in a lazy learning manner, our proposed *iBAT* method solves the problem of anomalous trajectory detection with the following characteristics:

- By exploiting anomalous trajectories' intrinsic properties of being "*few and different*", it is able to detect different kinds of anomalous trajectories.
- By applying the isolation mechanism, it provides a simple but effective way for detecting anomalous trajectories, no distance or density measure is needed.
- By working in a lazy learning manner, it naturally incorporates newly-generated taxi trajectories, thus can detect an emerging cluster of anomalous trajectories and hence the road network change promptly.
- By utilizing sub-sampling, it has the capacity to handle very large-scale set of trajectories, whilst keeping high performance and low processing-time..

Applications Based on Anomalous Trajectory Detection

With the anomalous trajectories detected and the characteristics of *iBAT*, different applications can be developed. For instance, if the travel distance of an anomalous trajectory is close to or shorter than that of the normal trajectories, then the route can be recommended to mission critical drivers in emergency cases; if the route is detected as a new path with many similar trajectories accumulating, then it can be used to update the road network change in the digital map. If the travel distance is much longer than normal ones, it is suspicious to be a driving fraud; further evidences can be collected to see if the case is caused by a traffic jam, or unfamiliarity with the area, or intentional fraud. Two potential applications will be further elaborated in next section.

EMPIRICAL EVALUATION

In this section, we provide an empirical evaluation of our proposed approach, and demonstrate its potential to enable practical applications.

Experimental Setup

In the experiments, we use a real-world taxi GPS dataset, which is collected from more than 7600 taxis served in a large city in China (Hangzhou) for one year (actually, the data in March 2010 has been used and found sufficient for this experiments). In this dataset, each taxi is equipped with a GPS device with a sampling-rate of about one record per minute. For each record, there are four fields: *latitude*, *longitude*, *passenger status* and *timestamp*. For the sake of simplicity in computation and visualization, we restrict our interest within the metropolitan area of Hangzhou with longitude [120.0°E, 120.5°E] and latitude [30.15°N, 30.40°N], the invalid records and those beyond this area are discarded. We discretize the area into a 100×200 grid cells, each corresponding to a $250m \times 250m$ square.

To provide a quantitative evaluation, we pick up five source-destination cell-pair[1], and ask three volunteers to *manually* label whether the trajectories are anomalous or not, in particular, if one volunteer thinks a taxi trajectory is anomalous, it is labeled to be anomalous. The datasets are summarized in Table 1, and visualized in Fig. 9.

Table 1. Datasets used in our experiments.

	# Trajectories	# Anomalies	Ratio
T-1	1418	41	2.89%
T-2	895	58	6.48%
T-3	593	43	7.25%
T-4	669	33	4.93%
T-5	649	36	5.55%

Table 2. The contingency table.

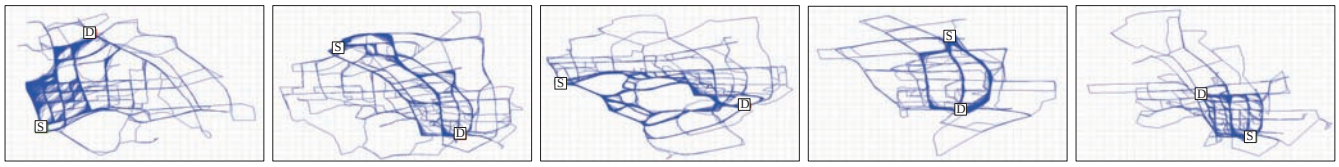
		Real Value	
		Anomalous	*Normal*
Detection Result	*Anomalous*	TP	FP
	Normal	FN	TN

For comparison purposes, we use the density-based method as a baseline, whose basic idea is to rank trajectories according to its density. Specifically, the density of a trajectory is the averaged density of all its cells, and the density of a cell is the number of trajectories that pass through it.

The evaluation criteria we use is AUC (Area Under ROC Curve) [4]. In practice, detection rate (dr, *i.e.*, the fraction of anomalous trajectories that are successfully detected) and false alarm rate (flr, *i.e.*, the fraction of normal ones that are predicted to be anomalous) are two important measures to evaluate the performance of an anomaly detection method. Based on Table 2, they are defined as

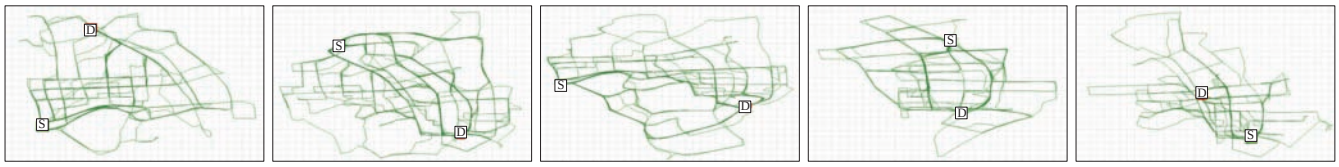
$$dr = \frac{TP}{TP + FN} \quad \text{and} \quad flr = \frac{FP}{FP + TN}.$$

[1]The source and destination are $500m \times 500m$ grid-cells.



(a) all trajectories of *T-1, T-2, T-3, T-4, T-5* (left to right)

(b) manually labeled anomalous trajectories on *T-1, T-2, T-3, T-4, T-5* (left to right)



(c) anomalous trajectories detected by iBAT on *T-1, T-2, T-3, T-4, T-5* (left to right)

Figure 9. Visualization of taxi trajectories, where S and D are source and destination. (top row: all trajectories; center row: manually labeled anomalous trajectories; bottom: anomalous trajectories detected by iBAT, which is of the same number of manually labeled anomalies)

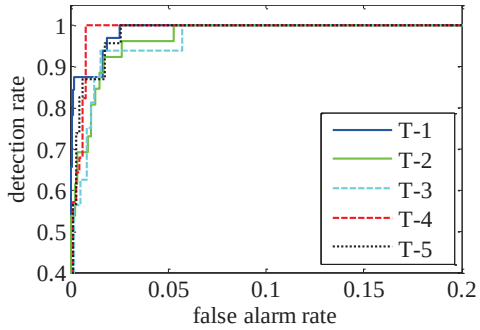


Figure 10. The ROC curves of *iBAT*. For better illustration, the ranges of false alarm rate and detection rate are set to [0, 0.2] and [0.4, 1].

Obviously, a good anomaly detection method should have both high detection rate and low false alarm rate. The ROC curve shows the detection rate (y-axis) against the false alarm rate (x-axis), and the AUC value is defined as the area under the ROC curve. As a statistical explanation, the AUC value is equal to the probability that a randomly chosen anomalous trajectory is ranked higher than a randomly chosen normal one. Obviously, if the AUC value is close to 1, the anomaly detection method is of high quality.

The experiments are run in Matlab on an Intel Xeon W3500 PC with 4GB RAM running Windows 7.

Experimental Results

We first provide a visualization of the detection results in Fig. 9, where manually labeled anomalous trajectories and the anomalous trajectories detected by *iBAT* (the same number of manually labeled anomalies) are depicted. We can see that the visualization of detected results are very similar to that of manually labeled data, which indicates that *iBAT* is effective in detecting anomalous trajectories.

Table 3. The AUC value of *iBAT* and density-based method.

	T-1	*T-2*	*T-3*	*T-4*	*T-5*
iBAT	0.9972	0.9936	0.9923	0.9970	0.9958
Density	0.9448	0.9491	0.9435	0.9712	0.9386

Fig. 10 depicts the ROC curves of *iBAT* on each dataset. We can find that *iBAT* is able to achieve high detection rate whilst keeping low false alarm rate. For all datasets, over 90% of anomalous trajectories can be detected at the false alarm rate of 2%; especially on *T-4*, 100% detection rate is achieved at the false alarm rate of less than 1%.

Table 3 compares the AUC value of *iBAT* with that of the density-based method. We can see that *iBAT* achieves quite high AUC values (>0.99 on all datasets) and the density-based method achieves lower AUC values (<0.95 on 4 datasets, 0.97 on *T-4*), suggesting that *iBAT* makes a better ranking than the density-based method. This is not difficult to understand, because the density-based method can only detect trajectories that pass through low-density cells and it ranks trajectories that detour on high-density cells lower, but our proposed method can detect both these two types of anomalous trajectories. In fact, this is also the reason why the density-based method achieves relatively higher AUC value on *T-4* than on other datasets, because there are less anomalous trajectories that detour on high-density cells.

In addition, we study *iBAT*'s performance and time efficiency in relation to the number of running trails m and sub-sampling size ψ. Specifically, we run experiments on the largest dataset *T-1*, and record the AUC value and processing time for $m \in [1, 200]$ and $\psi \in \{2, 4, 8, 16, \ldots, 1024\}$. Fig. 11 shows how the AUC value and processing time change with respect to m, where ψ is set to 256. It can be seen that the AUC value

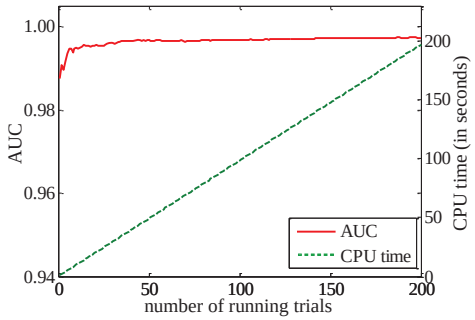


Figure 11. The AUC value (left y-axis, red solid) converges at a small m (x-axis), and processing time (right y-axis, green dashed) increases linearly with m.

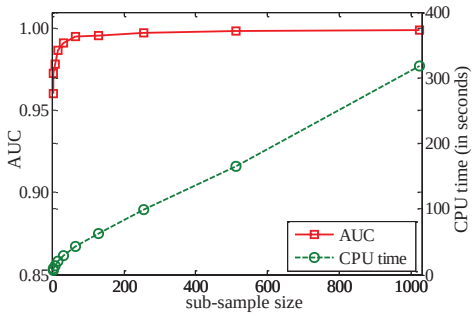
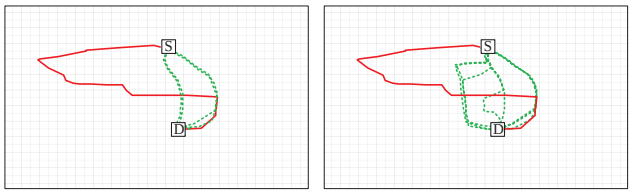


Figure 12. A small sub-sampling size provides both high AUC (left y-axis, red solid) and low processing time (right y-axis, green dashed).



(a) (b)

Figure 13. Avoiding excuses for taxi driving fraud detection. (a) the anomalous trajectory (red solid) is compared with previous trips of the same driver (green dashed); (b) the anomalous trajectory is compared with trajectories in 5 minutes (green dashed).

converges at a small m, and processing time increases linearly with m. Also, *iBAT* is quite efficient, for instance, when $m = 100$ the overall precessing time is about 100 seconds, about 0.07 second per trajectory. Of course, this can be further improved by parallel computing since every trial of *iBAT* is independent. In Fig. 12, we show how the performance changes with respect to ψ, where m is set to 100. It can be seen that both high AUC and low processing time can be obtained at a small sub-sample size.

Taxi Driving Fraud

As Fig. 9 shows, many detected anomalous trajectories are long-distance detours, which may correspond to taxi driving frauds. Therefore, detecting anomalous taxi trajectories can help building taxi driving fraud detection systems.

For detecting taxi driving frauds, a practical challenge is that some taxi drivers, responsible for anomalous taxi trajectories, may be truly unfamiliar with this area and some cunning drivers may use this as an excuse. Moreover, for some suspected frauds, someone may argue that they took a different route due to unexpected car accidents or heavy traffic. In this case, more evidence is needed to verify the fraud. Here, since we have all taxi trajectories between source and destination, if an anomalous trajectory is detected, we can get all previous trajectories of the corresponding driver to check whether he used to travel frequently in this area; meanwhile, we can also find all the trajectories that took place around the same time to check the traffic related excuses. For example, given an anomalous trajectory (red solid) in Fig. 13 (a), we compare it with the driver's previous trips (green dashed) between the same source and destination in Fig. 13 (b) and can see that this driver often operates in this area. Meanwhile, in Fig. 13 (b) we compare it with all the trips (green dashed) that happened around that time, and can see many other drivers did not detour. Based on these, we can deny possible excuses and confirm the fraud.

(a) Considered trajectories (b) Similar trajectories with t_0

Figure 14. Trajectories considered for road network change detection. (a) three trajectories considered: two anomalous trajectories t_0 and t_1 (red solid and blue solid), one normal trajectory t_i (green dashed); (b) 160 trajectories (dark red dashed) that are similar to t_0.

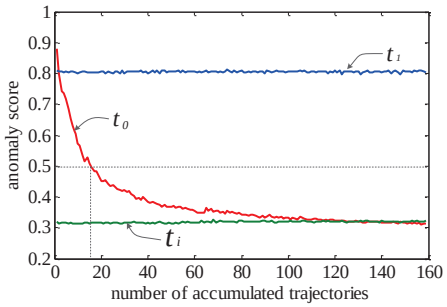


Figure 15. The anomaly score of t_0 decrease rapidly when similar trajectories are accumulating, but that of t_1 and t_1 keeps almost the same.

Road Network Change

As mentioned before, if more and more similar anomalous trajectories are accumulating, it may be an indication that a new road has been built or an old road is blocked in the area. We explain this via an example. As shown in Fig. 14 (a), we consider two anomalous trajectories t_0 and t_1 and one normal trajectory t_i. In addition, Fig. 14 (b) depicts a set of 160 trajectories that are similar to t_0. Then, to see the impact of accumulating trajectories, we add these 160 trajectories into the original trajectory set one by one, and study how the anomaly score of t_0, t_1 and t_i changes. The result is shown in Fig. 15, where we can see that the anomaly score of t_1 and t_i remain basically unchanged, while that of t_0 is reduced from about 0.9 to 0.3, which means that t_0 has definitively become a normal trajectory from an anomalous

one. In particular, after adding 18 similar trajectories, the anomaly score is reduced from 0.9 to 0.5. Hence, in real-world applications, if an anomalous trajectory becomes normal, like t_0, it corresponds to a change in the road network.

CONCLUSION AND FUTURE WORK

In this paper, we have investigated the problem of detecting anomalous driving trajectories from taxi's GPS trace, which is motivated by the fact that anomalous trajectories can reveal many hidden "facts" about the city dynamics and human behaviors and thus can be used to enable innovative applications. To solve the problem, we first grouped taxi trajectories crossing the same source-destination cell-pair and represented each taxi trajectory as a sequence of symbols. We then proposed the *iBAT* method to detect anomalous trajectories. Specifically, instead of profiling the normal trajectories or utilizing the distance or density measure, *iBAT* exploits anomalous trajectories' intrinsic properties of being "*few and different*", and applied the isolation mechanism to detect anomalous trajectories. We validate our approach in real-world taxi GPS data, and show that it achieves remarkable performance (AUC$>$0.99, detecting over 90% of anomalous trajectories at the false alarm rate of less than 2%). Furthermore, we show the potential of anomalous trajectories in enabling innovative applications, by two examples: taxi driving fraud detection and prompt road network change detection.

In the future, we plan to broaden this work in several directions. First, we will attempt to exploit other information enclosed in GPS traces such as driving speed for anomalous trajectories detection. Second, we would like to develop practical taxi driving fraud detection and road network change detection systems. Third, we are interested in detecting anomalous trajectories when the trajectory is ongoing.

ACKNOWLEDGEMENTS

The authors want to thank anonymous reviewers for helpful comments and Pablo S. Castro for comments and inputs. This research was supported by the Institut TELECOM "Futur et ruptures" Program, the Paris-Region SYSTEM@TIC Smart City Program, NSFC (61021062, 60803109), JiangsuSF (BK2008018), 973 Program (2010CB327903), 863 Program (2009AA011900). This work was done when Nan Li was visiting Institut TELECOM SudParis, France.

REFERENCES

1. N. Abe, B. Zadrozny, and J. Langford. Outlier detection by active learning. In *Proc. KDD 2006*, pages 504–509, 2006.
2. D. Aha. *Lazy Learning*. Springer, 1997.
3. F. Angiulli and F. Fassetti. Dolphin: An efficient algorithm for mining distance-based outliers in very large datasets. *ACM-TKDD*, 3(1), 2009.
4. A. P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145 – 1159, 1997.
5. M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *Proc. SIGMOD 2000*, pages 93–104, 2000.
6. Y. Bu, L. Chen, A. W.-C. Fu, and D. Liu. Efficient anomaly monitoring over moving object trajectory streams. In *Proc. KDD 2009*, pages 159–168, 2009.
7. L. Cao and J. Krumm. From GPS traces to a routable road map. In *Proc. GIS 2009*, pages 3–12, 2009.
8. J. Froehlich and J. Krumm. Route prediction from trip observations. In *Proc. SAE 2008*, pages 1031–1038, 2008.
9. 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. CIKM 2010*, pages 1733–1736, 2010.
10. M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453(5):779–782, 2008.
11. Z. He, X. Xu, and S. Deng. Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9-10):1641–1650, 2003.
12. E. M. Knorr, R. T. Ng, and V. Tucakov. Distance-based outliers: Algorithms and applications. *VLDB Journal*, 8(3-4):237–253, 2000.
13. J. Krumm and E. Horvitz. Predestination: Inferring destinations from partial trajectories. In *Proc. Ubicomp 2006*, pages 243–260, 2006.
14. J.-G. Lee, J. Han, and X. Li. Trajectory outlier detection: A partition-and-detect framework. In *Proc. ICDE 2008*, pages 140–149, 2008.
15. 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 *Workshops of PerCom 2011*, pages 63–68, 2011.
16. X. Li, J. Han, S. Kim, and H. Gonzalez. ROAM: Rule- and motif-based anomaly detection in massive moving object data sets. In *Proc. SDM 2007*, pages 273–284, 2007.
17. X. Li, Z. Li, J. Han, and J.-G. Lee. Temporal outlier detection in vehicle traffic data. In *Proc. ICDE 2009*, pages 1319–1322, 2009.
18. L. Liao, D. J. Patterson, D. Fox, and H. Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5-6):311–331, 2007.
19. Z. Liao, Y. Yu, and B. Chen. Anomaly detection in GPS data based on visual analytics. In *Proc. VAST 2010*, pages 51–58, 2010.
20. F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *Proc. ICDM 2008*, pages 413–422, 2008.
21. L. Liu, C. Andrisa, and C. Rattia. Uncovering cabdrivers' behavior patterns from their digital traces. *Computers, Environment and Urban Systems*, 34(6):541 – 548, 2010.
22. D. J. Patterson, L. Liao, D. Fox, and H. A. Kautz. Inferring high-level behavior from low-level sensors. In *Proc. Ubicomp 2003*, pages 73–89, 2003.
23. S. Phithakkitnukoon, M. Veloso, C. Bento, A. Biderman, and C. Ratti. Taxi-aware map: Identifying and predicting vacant taxis in the city. In *Proc. AmI 2010*, pages 86–95, 2010.
24. G. Qi, X. Li, S. Li, G. Pan, and D. Zhang. Measuring social functions of city regions from large-scale taxi behaviors. In *Proc. PerCom 2011*, pages 21–25, 2010.
25. R. R. Sillito and R. B. Fisher. Semi-supervised learning for anomalous trajectory detection. In *Proc. BMVC 2008*, pages 1035–1044, 2008.
26. 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. GIS 2010*, pages 99–108, 2010.
27. D. Zhang, B. Guo, and Z. Yu. Social and community intelligence. *IEEE-Computer*, PrePrint, 2011.
28. V. W. Zheng, B. Cao, Y. Zheng, X. Xie, and Q. Yang. Collaborative filtering meets mobile recommendation: A user-centered approach. In *Proc. AAAI 2010*, 2010.
29. Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma. Mining interesting locations and travel sequences from GPS trajectories. In *Proc. WWW 2009*, pages 791–800, 2009.
30. 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. Ubicomp 2008*, pages 322–331, 2008.
31. J. Zobel and A. Moffat. Inverted files for text search engines. *ACM Computing Surveys*, 38(2), 2006.