Real Time Anomalous Trajectory Detection and Analysis

Article *in* Mobile Networks and Applications · November 2012

CITATIONS
0

READS
304

1 author:

Chao Chen
Chongqing University
79 PUBLICATIONS **992** CITATIONS

SEE PROFILE

Mobile Netw Appl
DOI 10.1007/s11036-012-0417-8

Real Time Anomalous Trajectory Detection and Analysis

Lin Sun · Daqing Zhang · Chao Chen · Pablo Samuel Castro · Shijian Li · Zonghui Wang

Abstract GPS-equipped taxis can be considered 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 present a novel GPS-based taxi system which can detect ongoing anomalous passenger delivery behaviors leveraging our proposed *iBOAT* method. To achieve real time monitoring, we reduce the response time of *iBOAT* by more than five times with an inverted index mechanism adopted. We evaluate the effectiveness of the system with large scale real life taxi GPS records while serving 200,000 taxis. With this system, we obtain about 0.44 million anomalous trajectories out of 7.35 million taxi delivery trips, which correspond to 7600 taxis' GPS records in one month time in the city of Hangzhou, China. Through further analysis of these anomalous trajectories, we observe that: (1) Over 60 % of the anomalous trajectories are "detours" that travel longer distances and time than normal trajectories; (2) The average trip length of drivers with high-detour tendency is 20 % longer than that of normal drivers; (3) The length of anomalous sub-trajectories is usually less than a third of the entire trip, and they tend to begin in the first two thirds of the journey; (4) Although longer distance results in a greater taxi fare, a higher tendency to take anomalous detours does not result in higher monthly revenue; and (5) Taxis with a higher income usually spend less time finding new passengers and deliver them in faster speed.

Keywords Anomalous trajectory detection · Real time anomaly detection · Anomalous trajectory analysis · GPS equipped taxis · GPS traces

L. Sun · D. Zhang (✉) · C. Chen · P. S. Castro
RST Department, Institut Mines-TELECOM/TELECOM SudParis, Evry 91011, France
e-mail: daqing.zhang@it-sudparis.eu

L. Sun
e-mail: lin.sun.sudparis@gmail.com

C. Chen
e-mail: chao.chen@it-sudparis.eu

P. S. Castro
e-mail: pablosamuelcastro@gmail.com

S. Li · Z. Wang
College of Computer Science, Zhejiang University, Hangzhou, 310027, China

S. Li
e-mail: shijianli@zju.edu.cn

Z. Wang
e-mail: zjuzhwang@zju.edu.cn

1 Introduction

With the wide adoption of various pervasive sensing systems in our daily lives, such as mobile phones, GPS navigators and various types of RFID systems, the digital footprints we leave while interacting with the cyber-physical world have become increasingly rich and abundant. These rich data sources have enabled us to obtain a good understanding of human behaviors and city dynamics [7, 19], and to devise various novel applications. GPS sensors, for instance, have become an increasingly common tool in vehicles for localization and navigation. As pervasive sensors, they are capable of recording the driving traces of the vehicles, and large collections of these digital traces can be used to understand city dynamics [13, 16, 21] and social behaviors

Published online: 08 November 2012

[11, 18, 22]. They have inspired a variety of application-oriented research topics such as context-aware navigation [1, 17], detecting flawed urban planning [21], assisting taxi services [12, 18], amongst others. One of the topics that is receiving increasing attention is the anomaly detection of GPS-based taxi trajectories. This can prove useful for detecting fraudulent driving behaviors, adverse traffic conditions and newly constructed roads.

So far, research on anomalous trajectory detection has focused mainly on the detection method itself. In the literature, there are several methods for trajectory outlier detection, each addressing certain aspects of anomaly [2, 3, 6, 8, 20]. Although all these papers contributed in one way or another to the detection methods of trajectory anomaly, they failed to address the system issues such as how to make the system work in real-time with various real-life considerations. In another words, while they focus on how to detect the anomalous trajectories accurately, they didn't implement and evaluate the anomaly detection system in the real world setting, considering the key system factors such as response time, scalability and working context.

In this paper we would like to present a real time anomaly detection system which, leveraging our previous proposed method *iBOAT*, recognizes the ongoing anomalous passenger delivery behaviors of taxis based on their GPS traces. This system supports a variety of applications, illustrated by the following two examples.

Example 1 By monitoring the anomalous trajectories of all taxis in real-time, the taxi company can keep track of the taxi drivers' behaviors and take effective actions to minimize the fraud driving trips in a city.

Example 2 For passengers who are willing to be informed when the taxi drivers don't follow a normal route, the undertaking detour route with respect to the normal route can be shown in their mobile phones.

With this system, we obtain 0.44 million anomalous trajectories out of 7.35 million trips from 7,600 taxis in Hangzhou, China over a month. From this large collection of anomalous trajectories, we intend to extract common characteristics of anomalous behaviours, uncover the motivations behind fraudulent behaviours and investigate the impact of anomalous behaviors on drivers' revenues. We thus conduct an analysis aiming to answer the following questions.

- What percentage of all trips are anomalous?
- Out of the anomalous trajectories, what percentage of them travel longer distance than necessary?
- What statistical "tendencies" can we discern from the detected anomalous trajectories?
- Do taxi drivers who have a higher tendency to commit fraud have an economical advantage over those who don't?

There are two main contributions in this paper. The first is that we present an efficient GPS-based real time anomalous trajectory detection system. The evaluation results show that it can support a city with 200,000 taxis with a response time within a minute (sampling rate of GPS devices).

The second contribution is the observation resulting from our ensuing analysis, which reveal that:

1. Over 60 % of the anomalous trajectories are "detours" that travel longer distance and take longer time than normal trajectories.
2. Anomalous sub-trajectories are usually less than a third of the total trip length, and tend to begin in the first two thirds of the trip.
3. The average trip length of drivers with high detour tendency is 20 % longer than that of normal drivers.
4. Although longer distance results in a greater taxi fare, a higher tendency to take anomalous detours does not result in higher monthly revenue.
5. Taxis with a higher income usually spend less time finding new passengers and deliver them in faster speed.

The paper is organized as follows. In Section 2 we survey related work. In Section 3 we introduce our system architecture and elaborate the implementation details of each module. In Section 4 we evaluate the system in terms of its response time and applicability. In Section 5 we perform a thorough analysis of the detected anomalous trajectories motivated by the aforementioned questions. In Section 6 we conclude the paper with summary and discussions.

2 Related work

In this section, we survey the related work, which can be categorized into three groups. The first group includes the GPS-based applications and related systems designed for vehicles and, specifically, for taxis. The second group deals with anomalous trajectory detection work and related systems. The third summarizes the work related to the analysis of anomalous trajectories.

The first group is about various GPS-based applications. The past decade has seen a dramatic rise in the use of commercial GPS devices in regular vehicles for tracking and navigation. Researchers have profited

from these pervasive devices and have used them for traffic monitoring, estimating travel time, speed and travel delay [4, 14, 15]. In addition, much work has been dedicated to devising GPS-based applications and systems, specifically targeting taxis services. Liao [10] showed one GPS-based Automatic Vehicle Location and Dispatch System, which effectively supported taxi service management in three taxi companies in Singapore. Some recent work also aimed to improve the performance of taxi drivers, or make it easier for passengers to find vacant taxis. Li et al. [9] studied passenger finding strategies by distinguishing the behaviours of good and bad drivers. Yuan et al. [18] provided a recommender system for taxi drivers and passengers based on the taxi pickup behaviors learned from taxi GPS trajectories. Phithakkitnukoon et al. [12] predicted vacant taxis around the city based on the historical records collected with GPS devices. Ge et al. [5] developed a business intelligence system that could recommend driving route and detect driving fraud.

GPS records of taxis also enable novel applications that go beyond taxi services. Based on the GPS trajectory records of a large number of taxis, Yuan et al. [17] and Brian [1] designed *T-Drive* and *PROCAB*, respectively, to provide smart navigation guidances. Zheng et al. [21] discovered inefficient connectivity among different regions of the city based on the GPS trajectories of taxis traveling in urban areas. Based on the picking-up/dropping-off patterns in different city regions, Qi et al. [13] measured their social functions.

The second group includes the anomaly detection work based on GPS traces which is highly related to this paper. There have been a number of recent papers on detecting anomalous moving vehicles. For example, Bu et al. [2] presented an outlier detection framework which detects anomalies by joining clusters from the local clusters in a trajectory stream. Ge et al. [6] detected the evolving trajectory outliers based on the evolving direction and density information. Lee et al. [8] proposed a partition-and-detect framework to detect outlying sub-trajectory. Noticing the *few* and *different* properties of anomaly trajectories, we recently proposed two methods, namely *iBAT* [20] and *iBOAT* [3], to detect the anomaly of a finished trajectory and an ongoing trajectory, respectively.

The last group is about the analysis of anomalous trajectories. Large number of trajectories have been used to demonstrate the spatiotemporal variation of taxi services and the relationship between pick-up and drop-off locations in [16]. The analysis that is closely related to our work can be found in [5, 11]. In [5] Ge et al. detected driving frauds and provided some analysis about fraudulent taxi drivers based on the GPS traces of 500 taxis, observing that fraudulent drivers earn more on average than non-fraudulent drivers. In [11], Liu et al. found that for trips under 3 km, top performance drivers take more direct route than ordinary drivers. And in contrast, for trips above 3 km, top drivers take more indirect routes than ordinary drivers. And top drivers are more skilled at finding the fastest way than ordinary drivers, perhaps because they have more knowledge about the road network and the traffic conditions of the city.

In this paper, we present a real time anomaly detection system which recognizes the anomalous passenger delivery behaviors. Different from the previous work, we focus on the design and implementation of the mechanisms in the anomalous trajectory detection system in order to achieve real-time response, even with 200,000 taxis operating in the system. We further provide thorough analysis of the large collection of anomalous trajectories detected and gain deep insights about the characteristics, motivations and impact of the anomalous behaviors.

3 System architecture

In this section we present the design and implementation of the anomalous trajectory detection system.

3.1 Basic idea of the anomaly detection system

As shown in Fig. 1, the anomalous trajectory detection system works with an incoming stream of GPS packets from a fleet of taxis (message ①). Each packet contains the taxi's ID, GPS position (longitude and latitude), passenger status (occupied or vacant), speed, orientation and timestamp. In Table 1 we display some examples of GPS packets. When receiving these raw packets, the server pre-processes them, constructs the on-going passenger delivery trajectories, and detects which trajectories are anomalous based on *iBOAT* method. After detecting the anomalous trajectories, the server can either inform the passengers (③ in Fig. 1) or show the anomaly map and analysis to the system administrators (④ in Fig. 1).

The anomaly detection task in our system is performed based on our previously proposed method *iBOAT*. In order to detect if a testing trajectory t is anomalous or not, *iBOAT* first gathers a set of trajectories T that are with the same source and destination (abbreviated as $\langle S, D \rangle$) pair and obtained in the similar context (similar traffic, weather condition). Because the GPS position of a taxi can be any value in a continuous two-dimensional plane, it is difficult to work with the

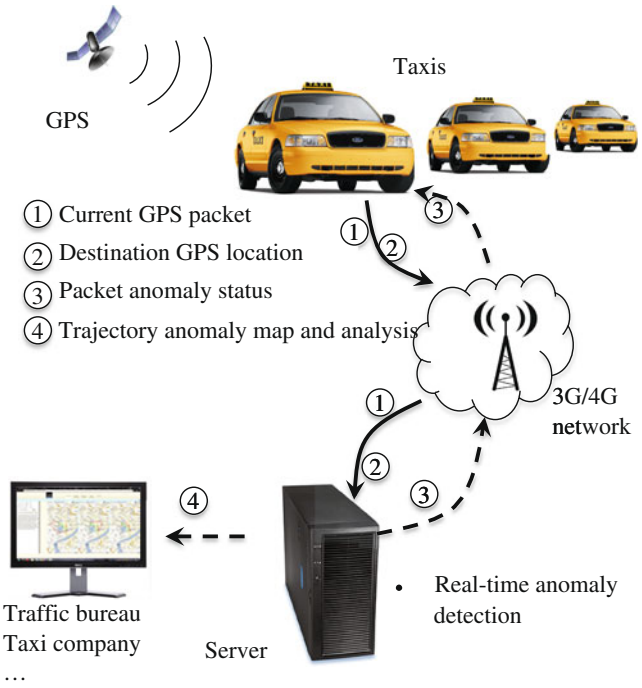


Fig. 1 Taxi anomaly detection work flow

GPS coordinates directly. A common approach is to construct a finite abstract representation of this two-dimensional plane, and work on this representation instead. It implies that all incoming GPS points must be mapped to one of the possible states of the abstract representation. Our anomaly detection system uses grid partition, in which the entire city map is split into equal-sized grids and each GPS point is mapped to the grid it "falls" in. Then a trajectory t is mapped and augmented into a sequence of adjacent grids starting from the source grid.

Given a sub-trajectory t^* which tracks the longest normal grid sequence of a testing trajectory t. *iBOAT* maintains those trajectories from T that contain the same grid sequence of t^* (denoted as T^*). If T^* is rare compared to T (judged by a predefined threshold on $|T^*|/|T|$), it means there are very few trajectories in T that follow the current traveling route, and it reports an anomaly. Then the next grid of t is assigned to t^* and T^* is initialized as T. If T^* is big compared to T, then the next grid in t is added to t^* to extend the sub-trajectory. This process is repeated until the whole trajectory is tested and we can get all the anomaly records in t. Meanwhile, *iBOAT* defines an anomaly score as the traveling length of the anomalous segments.

As the anomalous trajectory detection system aims at serving a large number of taxis simultaneously in *real time*, the primary issue we have to address is the response time of the system. In a city like Hangzhou, China (where our dataset is taken from), there are 8,500 taxis operating in the city. Thus, our system should be able to perform anomaly detection on all 8,500 occupied taxis *simultaneously*, at a rate of at least once per minute (taxi GPS sampling time).

In the process of anomalous trajectory detection, another critical issue is to consider the effect of context such as time and weather during the preparation of historic trajectory set. Since our system is based on historical "traces", it is important to consider only those historical trajectories that occurred under similar context. It should be noted that previous anomaly detection mechanisms have only considered physical location as a relevant context for accumulating historical trajectories. We believe that additional contextual information will improve the overall accuracy of the system.

3.2 Overall system design

The overall system architecture is shown in Fig. 2, it is comprised of three modules: data pre-processing, dataset preparation and anomaly detection. Data pre-preprocessing is in charge of receiving, transforming and filtering the incoming GPS packets to construct valid and symbolic trajectories for each taxi. These trajectories are then fed into the Anomaly Detection Engine, which performs on-line anomaly detection. In the dataset preparation module, on one hand, the individual packets are grouped into trajectories and augmented with contextual information; on the other hand, the relevant historic trajectories with the same source, destination and context are provided to the Anomaly Detection Engine for anomaly detection.

The general workflow of the system is as follows. The input of the system are the GPS packets received from taxis via a telecommunication network, such as 3G, which are aggregated through the GPS Packet Aggregator. Then they are fed into the Trajectory Manager which filters out erroneous GPS points and keeps track of all the incoming occupied trips. For each received

Table 1 Examples of GPS packets

TaxiID	Longitude	Latitude	Speed	Orientation	Occupation	Year	Month	Day	Hour	Min	Sec
9970	120.157762	30.259317	3.7	280	Vacant	2011	11	12	13	2	20
11200	120.258423	30.365834	27	120	Busy	2011	11	12	13	2	50
7869	123.340354	30.289732	18.3	340	Vacant	2011	11	12	13	3	05

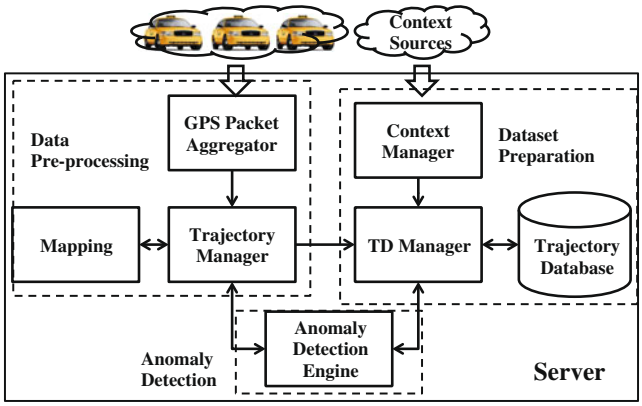


Fig. 2 Overall system architecture

packet of an ongoing trajectory, Trajectory Manager calls the Mapping component to convert the current GPS point into grid location. The Mapping component returns the ID of the mapped grid to the Trajectory Manager. Once a trajectory is completed (i.e. the status of the taxi has gone from *occupied* to *vacant*), the full trajectory is augmented by the Mapping component, and then updated to the Trajectory Database through the TD Manager, under the category of trajectories with same source, destination and context.

Meanwhile, for each received packet, Trajectory Manager checks whether it falls in the same grid as the previous packet. If yes, the anomalous status of this packet is the same as the previous one. Otherwise, it calls the Anomaly Detection Engine to test if the grid ID mapped from the new GPS packet represents an anomalous cell with *iBOAT* method. Trajectory Manager also associates the anomalous status to each packet. For each ongoing GPS trajectory, TD Manager obtains the current context from the Context Manager, selects the trajectories with the same source, destination and context from the Trajectory Database, and returns this set of trajectories to the Anomaly Detection Engine for anomaly detection.

3.3 System components

In this section, we elaborate the implementation details of each system component in Fig. 2.

3.3.1 GPS Packet Aggregator

To effectively gather data, we adopt the Representational State Transfer (REST) architecture, which represents each resource (in our case taxis) as an URL. The clients of our system use HTTP POST operation to upload the GPS packet information into our system with this URL. After receiving the data, the GPS Packet Aggregator extracts the taxiID from the URL and other parameters from the POST message body to form a complete record. In addition to the GPS records shown in Table 1, when a new passenger is picked up, the destination specified by the passenger will also be included in the POST message. This destination, along with the passenger pickup location (source), will be included in the data records sent out and used by the TD Manager when selecting relevant trajectories.

3.3.2 Mapping

As we mentioned above, in order to feasibly manage the GPS points, we construct a *finite* abstract representation, or decomposition, of the original two-dimensional plane. The Mapping component receives a GPS point and maps it into one of the states of the chosen decomposition. From then on, all computations are performed only on the mapped GPS points. Additionally, the Mapping component receives completed trajectories from the Trajectory Manager and augments them before sending them back to the Trajectory Manager. The purpose of augmentation is to ensure that there are no gaps between successive mapped GPS points in a trajectory. The process of augmentation is described in more detail in [3, 20].

There are a number of ways to decompose the city, such as using a digital map of the road network and partitioning it into different functional regions. For our current implementation, we decompose the city into a set of grids, which is a simple and popular decomposition method. The basic idea is to split the city area into a matrix of grid cells, and each GPS point maps to the grid cell where it "landed".

3.3.3 Trajectory Manager

The Trajectory Manager is in charge of two tasks: data filtering and trajectory extraction. When a new GPS packet arrives, the system uses the distance and time elapsed since the last GPS packet to estimate the speed at the current point. If the speed is unreasonably high given the speed limits of the taxi's location, the point is labelled as Type A noise. Similarly, if the time elapsed since the last point is unreasonably high (4 min), we label the current point as a Type B noise.

For each taxi, we maintain a queue representing the current occupied trajectory. The queue will obviously be empty if the taxi is currently vacant. When a passenger is picked up (i.e. the status goes from vacant to occupied), the Trajectory Manager initializes the queue with the first point, which is set as the trajectory's source. And the trajectory's destination is set to what

was given by the taxi passenger. As a new point arrives, it is added to the end of the queue. If it is landed in the same grid as the one right before it in the queue, its anomalous status is the same as the previous record. Otherwise, it is sent to the Anomaly Detection Engine to verify whether it is anomalous.

If the current point is detected as a Type A noise, it is simply ignored and the system waits for a new point to arrive (so the queue remains intact). If the current point is a Type B noise, or if too much time has elapsed since the last valid point, then the current trajectory is aborted: the queue is emptied and an abort message is forwarded to the Anomaly Detection Engine.

When a trajectory is completed (i.e. status switches to vacant and current position is the pre-specified destination), then the Trajectory Manager sends it to the Mapping component for augmentation. This augmented trajectory is then fed to the TD manager for insertion into the Trajectory Database.

3.3.4 Context Manager

The Context Manager is in charge of providing the current contextual information to the TD Manager for labeling new trajectories that will be added to the Trajectory Database or for selecting which trajectories to provide to the Anomaly Detection Engine. There are many different types of context that can be used, such as the date, time, weather, traffic conditions, etc. However, if too many context variables are provided, there will be few trajectories satisfying any configuration of the context variables. As we will discuss below, this reduces the number of trajectories that can be compared against. In our implementation we use eight different day-time combinations, which will be specified in Section 4. Each context configuration is assigned with a unique ID number. When queried, the Context Manager returns the ID number corresponding to the current context setting.

3.3.5 Trajectory Database and TD Manager

The Trajectory Database stores and manages all of the historical trajectories, which have all been labeled with contextual information provided by the Context Manager. Records in this database are indexed by trajectory ID, and the contents of each record are the grid cells of the trajectory, in the correct order and its context label. Additionally, we make use of the Inverted Index Mechanism [23] for fast retrieval of relevant trajectories. For this mechanism, we maintain a second database whose records are the grid cells and 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.

The TD Manager component is in charge of adding new trajectories to the Trajectory Database and extracting the related trajectories for the Anomaly Detection Engine. When a completed trajectory is received from the Trajectory Manager, the TD Manager gives it a unique ID, labels it with the context information received from the Context Manager and stores it in the Trajectory Database.

When the Anomaly Detection Engine requests the set of relevant trajectories from the TD Manager, it specifies the source and destination for the trajectory currently being examined. Previously we split the entire city into 500m $\times$ 500m large grids and choose the trajectories that have the same source and destination grids in *iBAT* [20], and *iBOAT* [3]. To get as many trajectories for the sparse $\langle S, D\rangle$ pairs as possible, we also incorporate those trajectories that pass through $\langle S, D\rangle$. The detail of this process can be found in [20].

3.3.6 Anomaly Detection Engine

In this section we introduce our implementation of *iBOAT* in Anomaly Detection Engine. Upon receiving a trajectory set T from the TD Manager, the Anomaly Detection Engine can proceed to test the anomalousness of the sub-trajectory t^*. Actually any anomaly detection algorithm can be used for the Anomaly Detection Engine, but for our implementation we use *iBOAT* [3].

Based on the introduced in Section 3.1, when testing a sub-trajectory t^*, *iBOAT* searches all trajectories in T^* and maintains those ones that contain t^*. This is a quite time consuming process as it needs to check from the start to the end of each trajectory in T^*. Actually it is a perfect case for using Inverted Index Mechanism to speed up. For example, for searching whether a grid sequence $\langle g_1, g_2\rangle$ is in a trajectory t, instead of searching it from the start to the end of t, we can easily decide it by checking whether both $pos(t, g_1)$ ($pos(t, g)$ is the position of g in t) and $pos(t, g_2)$ exist (i.e., g_1 and g_2 exist in T) and $pos(t, g_1) < pos(t, g_2)$ (i.e., g_1 is in front of g_2 in T).

Given a trajectory set T, we first transform it into an Inverted Index Dataset IID. Let (t, p) denotes an inverted index, which means the pth position in t. Each item $g_i : \{(t_1, pos_1), (t_2, pos_2), \ldots\}$ in IID contains all the inverted indices of g_i in the trajectories belonging to T. We maintain a working indexing set I, initialized to $I = \{(t, 1)|t \in T\}$. The indexing set I maintains all trajectories that are *consistent* with the sub-trajectory

t^* currently being examined. An element (t, pos) in I means that trajectory t is consistent with on-going sub-trajectory t^* *up to* position *pos*. Thus, at the beginning, all trajectories from I begin at position 1.

The process is outlined in Algorithm 1. The indexing set I is initialized in line 1. Lines 2–18 iterate as long as new grid cells for the current trajectory are received from the Trajectory Manager. In line 3, we fetch the inverted index set G of g in IID. We iterate through the pairs in I in lines 4–11, verifying whether each trajectory is still consistent with the new grid cell g: If g is in t (t is occurred in G) and its location is after *pos* ($pt > pos$), then we change the active position of t in I to be the index of the occurrence of g in t that right *follows* the current active position in t, *pos* (lines 5–7) ; Otherwise, we remove (t, pos) from I (lines 8–9). Once all the trajectories in I have been checked, we determine whether the resulting size of I is above the allowed threshold (line 12): if so, g is labeled as normal (line 16); if not, g is labelled as anomalous and we reset I (lines 12–14). The process then proceeds with the next received grid cell. Once the trajectory is completed, we can compute an anomalous score using the distance of the anomalous sub-trajectories, as outlined in [3].

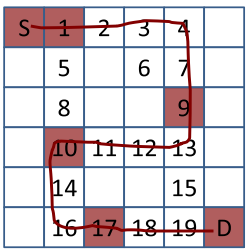
Algorithm 1 $iBOAT$ using inverted index mechanism

Input: IID – inverted index dataset generated from trajectory set T fed in by TD Manager
θ – anomaly threshold

Process:

1: $I \leftarrow \{(t, 1) | t \in T\}$
2: **while** new grid g received from Trajectory Manager **do**
3: $G \leftarrow IID(g)$
4: **for** all $(t, pos) \in I$ **do**
5: **if** any (t, pt) exists in G that $pt > pos$ **then**
6: $newPos \leftarrow \arg\min_i \{i \geq pos | (t, i) \in IID(g)\}$
7: update (t, pos) in I to $(t, newPos)$
8: **else**
9: delete (t, pos) from I
10: **end if**
11: **end for**
12: **if** $|I| < \theta$ **then**
13: Label g as anomalous
14: $I \leftarrow \{(t, 1) | t \in T\}$
15: **else**
16: Label g as normal
17: **end if**
18: **end while**
19: Report the anomalous score of completed trajectory

To illustrate this process, we go through a smiple example. Consider the grid cells in Fig. 3(a), with historical trajectories listed in Fig. 3(b), and with an incoming



(a) Traversed cells

$t_1 : S \rightarrow 1 \rightarrow 5 \rightarrow 8 \rightarrow 10 \rightarrow 14 \rightarrow 17 \rightarrow 18 \rightarrow 19 \rightarrow D$
$t_2 : S \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 7 \rightarrow 9 \rightarrow 13 \rightarrow 15 \rightarrow 19 \rightarrow D$
$t_3 : S \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 7 \rightarrow 9 \rightarrow 13 \rightarrow 15 \rightarrow 19 \rightarrow D$
$t_4 : S \rightarrow 1 \rightarrow 2 \rightarrow 6 \rightarrow 9 \rightarrow 13 \rightarrow 15 \rightarrow 19 \rightarrow D$
$t_5 : S \rightarrow 1 \rightarrow 5 \rightarrow 8 \rightarrow 10 \rightarrow 14 \rightarrow 17 \rightarrow 18 \rightarrow 19 \rightarrow D$
$t_6 : S \rightarrow 5 \rightarrow 8 \rightarrow 10 \rightarrow 14 \rightarrow 17 \rightarrow 18 \rightarrow 19 \rightarrow D$
$t_7 : S \rightarrow 1 \rightarrow 5 \rightarrow 8 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow 13 \rightarrow 15 \rightarrow 19 \rightarrow D$
$t_8 : S \rightarrow 1 \rightarrow 5 \rightarrow 8 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow 15 \rightarrow 19 \rightarrow D$
$t_9 : S \rightarrow 1 \rightarrow 5 \rightarrow 8 \rightarrow 11 \rightarrow 12 \rightarrow 13 \rightarrow 15 \rightarrow 19 \rightarrow D$

(b) Trajectory set

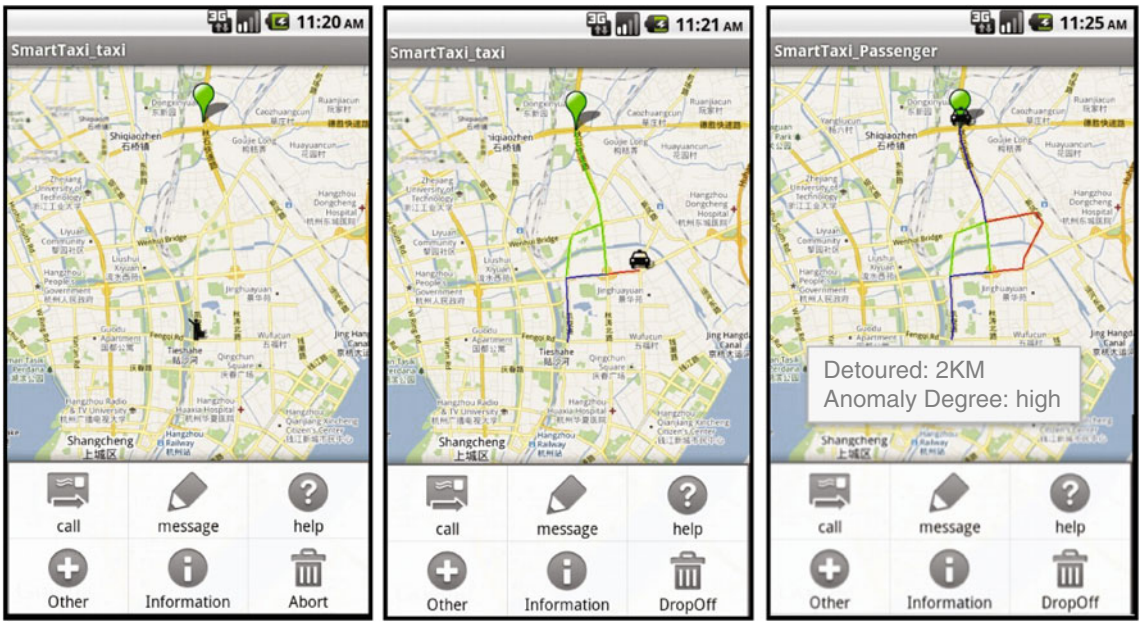
$(grid)\mathbf{S}$: $\{(t_1,1), (t_2,1), (t_3,1), (t_4,1), (t_5,1), (t_6,1), (t_7,1), (t_8,1), (t_9,1)\}$; **1**:$\{(t_1,2), (t_2,2), (t_3,2) (t_4,2), (t_5,2), (t_7,2), (t_8,2), (t_9,2))\}$; **2**:$\{(t_2,3), (t_3,3), (t_4,3)\}$; **3**:$\{(t_2,4), (t_3,4)\}$; **4**:$\{(t_2,5)\}$; **5**:$\{(t_1,3), (t_5,3), (t_6,2), (t_7,3), (t_8,3), (t_9,3)\}$; **6**:$\{(t_4,4)\}$; **7**:$\{(t_2,6), (t_3,5)\}$; **8**:$\{(t_1,4), (t_5,4), (t_6,3), (t_7,4), (t_8,4), (t_9,4)\}$; **9**:$\{(t_2,7), (t_3,6), (t_4,5)\}$; **10**:$\{(t_1,5), (t_5,5), (t_6,4), (t_7,5), (t_8,5)\}$; **11**:$\{(t_7,6), (t_8,6), (t_9,5)\}$; **12**:$\{(t_7,7), (t_8,7), (t_9,6)\}$; **13**:$\{(t_2,8), (t_3,7), (t_4,6), (t_7,8), (t_8,8), (t_9,7)\}$; **14**:$\{(t_1,6), (t_5,6), (t_6,5)\}$; **15**:$\{(t_2,9), (t_3,8), (t_4,7), (t_7,9), (t_8,8), (t_9,8)\}$; **17**:$\{(t_1,7), (t_5,7), (t_6,6)\}$; **18**:$\{(t_1,8), (t_5,8), (t_6,7)\}$; **19**:$\{(t_1,9), (t_2,10), (t_3,9), (t_4,8), (t_5,9), (t_6,8), (t_7,10), (t_8,9), (t_9,9)\}$; **D**: $\{(t_1,10), (t_2,11), (t_3,10), (t_4,9), (t_5,10), (t_6,9), (t_7,11), (t_8,10), (t_9,10)\}$;

(c) Inverted Index Dataset (*IID*)

Grid cell	Contents of I
S	$\{(t_1,1),(t_2,1),(t_3,1),(t_4,1),(t_5,1),(t_6,1),(t_7,1),(t_8,1),(t_9,1)\}$
1	$\{(t_1,2),(t_2,2),(t_3,2),(t_4,2),(t_5,2),(t_7,2),(t_8,2),(t_9,2)\}$
9	$\{(t_2,7),(t_3,6),(t_4,5)\}$
10	$\emptyset$
17	$\{(t_1,7),(t_5,7),(t_6,6)\}$
D	$\{(t_1,10),(t_5,10),(t_6,9)\}$

(d) Evolution of indexing set I

Fig. 3 An example of $iBOAT$ with inverted indexing mechanism. (**a**) Example trajectory (*red line*) with mapped grid cells (*red squares*), *blue lines* are the grid decomposition of the map; (**b**) Trajectory set from S to D; (**c**) The corresponding Inverted Index Dataset; (**d**) Evolution of the indexing set I as the incoming trajectory progresses



(a) (b) (c)

Fig. 4 Mobile phone user interfaces for passengers: (**a**) Pressing the screen for over 3 sec to select the destination (*green bubble*) when pick-up; (**b**) Anomalous alert during the passenger delivery process; (**c**) Trajectory summary when trip is finished

trajectory (red line); the red cells indicate the position where a GPS packet is received. Figure 3(c) shows the corresponding IID. We depict how the indexing set I changes as the trajectory progresses in (d). All nine trajectories support the first cell S; this number drops to 8 in cell 1, and we should notice that the active position in all trajectories increases to 2; upon landing in cell 9 only three trajectories remain, and there are no supporting trajectories at 10 (this means that there are no historical trajectories that include the path $S \rightarrow 1 \rightarrow 9 \rightarrow 10$); at this point the indexing set I is reset to its original state, and on landing on cell 17, there are only three trajectories supporting it; then the destination is finally reached. For simplicity we say the anomalous threshold is set at zero. Thus, in this example, only one point was labelled as anomalous: grid cell 10.

We can see this process is quite efficient as it can directly pinpoint those trajectories that contain the testing sub-trajectory. We will evaluate its efficiency in Section 4.1.

Fig. 5 Taxi anomaly map—user interface for administrators in desk-top PC

3.4 User interface design

In the proposed anomaly detection system, we design two types of user interfaces for users. Figure 4 shows the user interfaces designed for taxi passengers. At the start of the trip, the passenger is requested to select the destination (Fig. 4(a)). During the passenger delivery process, the delivery route is marked blue when it's normal and marked red when anomaly occurs (Fig. 4(b)). When the trip is completed, the undertaking route and normal routes are presented to the passenger (Fig. 4(c)).

Figure 5 shows one of the three snapshots of the user interface for taxi company administrators, where the location map of the ongoing anomalous trips can be displayed. Further analysis of historic anomalous trips can be visualized according to administrator's needs. For example, the taxis can be ranked based on the number of detours, frequency of detours, or distance of detours and etc.

4 System evaluation

In this section, we evaluate our system with a large collection of taxis' GPS traces from Hangzhou, China. We first give a brief introduction of the taxi data in Hangzhou and the context information we use during the testing. Whereafter we show the improvement of *iBOAT* with the inverted index mechanism over the original implementation. Then we evaluate the system response time to show it's effectiveness in dealing with large number of taxis. In the end, as the anomaly detection method requires a dataset with sufficient number of trajectories under the same $\langle S, D\rangle$ pair, we also evaluate this assumption in real life scenarios. We define the system coverage as how much percentage of trajectories in real life that fulfill such a requirement and evaluate it in different situations.

Currently there are about 8,400 taxis operating in Hangzhou, China, out of which around 7,600 have been deployed with a GPS sensing device. In March 2010 we collected about 441 million packets from these taxis. In this section we intend to verify whether our proposed system can handle anomaly detection for such a large fleet of taxis.

For the context setup, we divide a week into working and non-working days, and separate each day into four different time slots: night (0:00 $\sim$ 6:59), morning (7:00 $\sim$ 11:59), afternoon (12:00 $\sim$ 16:59) and evening (17:00 $\sim$ 23:59). By combining the type of day and time slots, we get eight different combinations which we encode as WN (**W**orking day **N**ight), WM (**W**orking day **M**orning), WA (**W**orking day **A**fternoon), WE (**W**orking day **E**vening), NWN (**N**on-**W**orking day **N**ight), NWM (**N**on-**W**orking day **M**orning), NWA (**N**on-**W**orking day **A**fternoon), and NWE (**N**on-**W**orking day **E**vening). Since the weather is almost the same for the whole month, we simply ignore its influences in the evaluation.

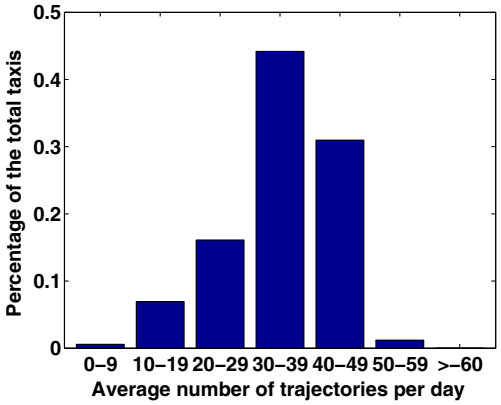


Fig. 6 Distribution of taxis over average number of daily trips

We calculate the average daily delivery trips for each taxi and show the distribution over these averages in Fig. 6. It shows that most of the taxis have about 30$\sim$50 trips per day, but there are a few excellent taxis that make over 50 trips per day. We display the average number of daily trips in each time slot in Table 2. Although there are more trips in working days, there are more trips in non-working nights. This behaviour is in accordance with what is expected from our experience.

4.1 Improvements of *iBOAT* with inverted index machanism

We evaluate the performance of Algorithm 1 comparing with the original implementation in [3] under different historical trajectory set size. We use around 10,000 trajectories from 300 $\langle S, D\rangle$ pairs with various set size. For each trajectory, we perform the anomaly detection with both algorithms under the same computing environment and measure the computing time. Then we calculate the ratio between time needed in the original algorithm and time needed by Algorithm 1, and plot the ratio against the size of trajectory set in Fig. 7.

It can be easily seen that Algorithm 1 is much faster than the original implementation. (The average im-

Table 2 Average number of daily trips

Day type	Night	Morning	Afternoon	Evening
Working	27k	58k	63k	87k
Non-working	34k	54k	61k	95k

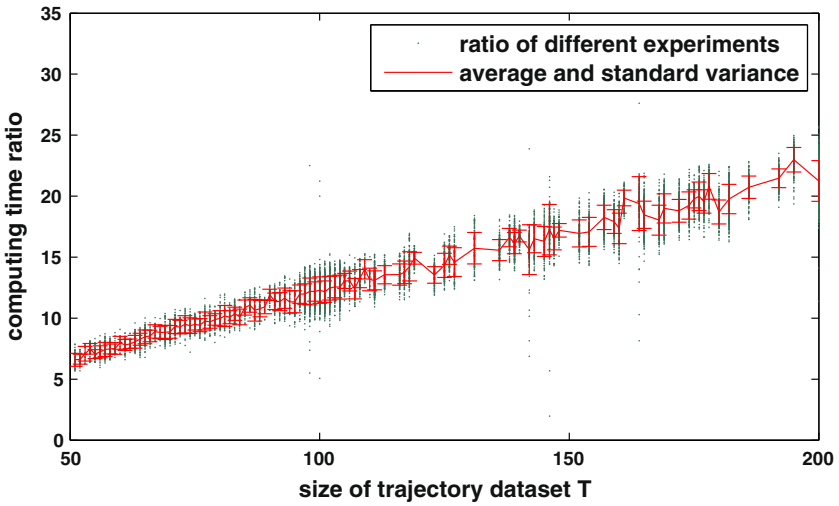


Fig. 7 Computing time ratio (original/Algorithm 1) versus the size of trajectory set T

provements are over five times faster.) The red line depicts the trend of the mean ratio for different trajectory set size. It shows that the ratio is linearly increasing when the trajectory set size becomes large, meaning that the bigger the trajectory set is, the more advantage Algorithm 1 achieves.

4.2 System response time

We would like to verify that our system is able to handle data coming from a large fleet of taxis and monitors the anomalousness of on-going trajectories in a timely manner. Specifically, since the sampling rate of GPS devices is approximately once per minute, we would like to ensure that our system can respond to all taxis within one minute.

We construct our database using data from March 2010, and use the data from April 2010 to simulate a deployed system. Since each GPS packet includes a timestamp, we can "replay" each of these packets in the correct order and at the correct time, effectively replicating the reception of these packets in the real life scenario. We ran the test on a PC with Intel Xeon CPU (2.8 GHz) and 12 G memory.

For the dataset of 7,600 taxis, our system was able to respond within 2 sec, which is well below one minute. We envision this type of system can be deployed in larger cities, such as Beijing (around 70,000 taxis) and Mexico city (around 250,000 taxis), so it is necessary to verify the scalability of the proposed system. We "cloned" the taxi records in our database in order to create a large fleet of taxis. In Fig. 8 we display the response time of the system during the busiest hour in our dataset (8 pm). We can see that the response

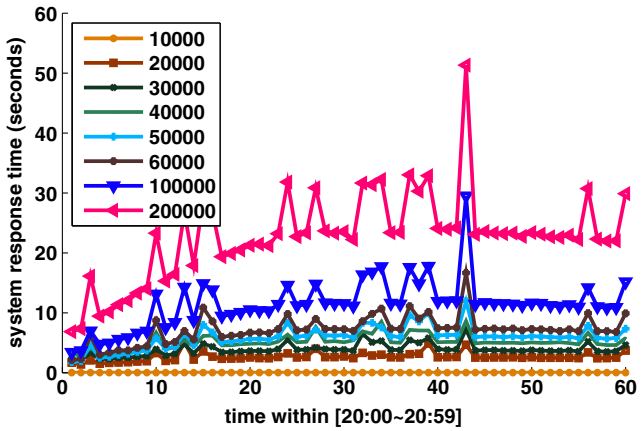


Fig. 8 Response time versus number of taxis in the busiest hour

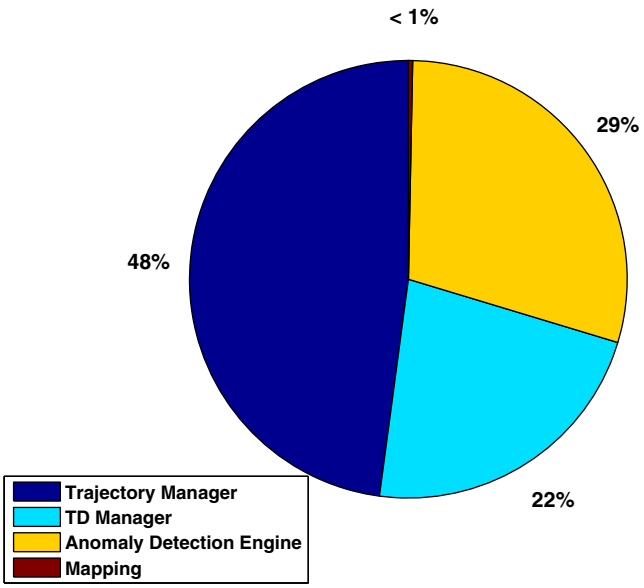


Fig. 9 Decomposition of processing time

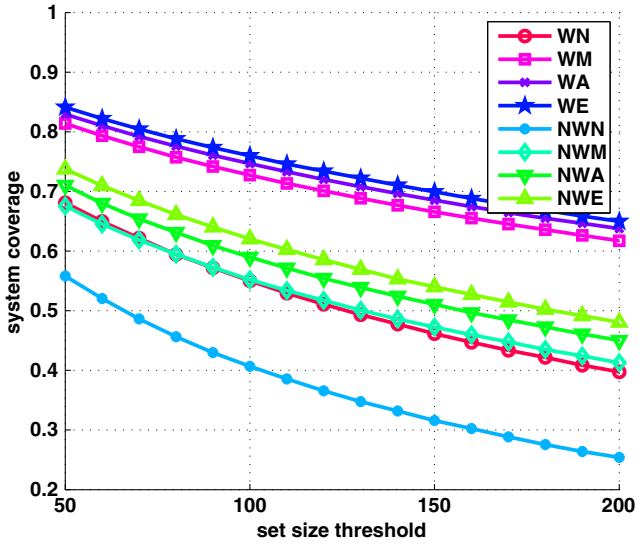


Fig. 10 System coverage versus sufficient trajectory set size thresholds (using four weeks historical data)

time grows no more than linearly with increasing data sizes, remaining well below a minute even for a fleet of 200,000 taxis. This suggests that even for a city with around 200,000 taxis, the response time should be below one minute by deploying the system in an office PC.

In Fig. 9 we display how the processing time is split amongst the different components. We can see that around half of the time is consumed by the Trajectory Manager, mainly due to the data filtering aspect. It is somewhat surprising that the Anomaly Detection Engine takes less than a third of the total time; this is by virtue of the use of the inverted index mechanism, which dramatically decreases the computation time of *iBOAT*. The TD Manager has also used less than a third of the total computation time, partly because we adopt a caching mechanism for trajectory preparation.

4.3 System coverage with *iBOAT*

The detection of anomalous points is based on the number of *support* from historical trajectories. If there are very few trajectories between a particular source and destination area, the method will not be able to check incoming trajectories between said pair of points. Thus, given a fixed set of trajectories, the threshold we select for determining the sufficiency of trajectories directly determines how many source-destination pairs can be tested by *iBOAT* and how much percentage of trajectories can be tested by our system (i.e., system coverage) in real life. We choose the grid size 500m × 500m, which is the same as in [20]. In Fig. 10 we plot the percentage of trajectories covered by our system as the threshold varies with all the trajectories in March 2010. We can see it drops when the threshold increases. As there are 23 working days in that month while only eight non-working days, the coverage in each time slot of day is bigger in working days than in non-working days. As the criteria of judging whether the trajectories are anomalous is decided by whether it is "few" and

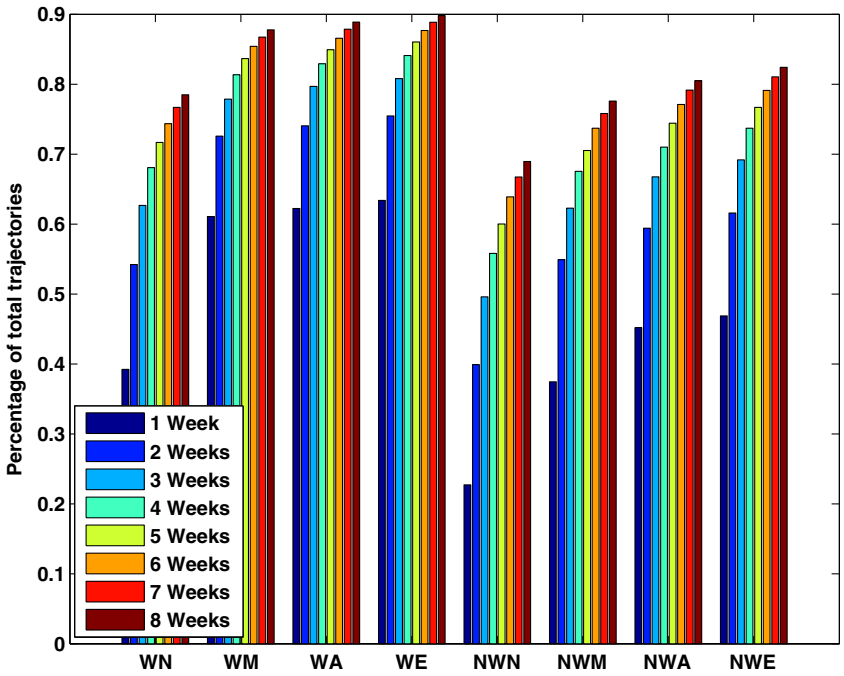


Fig. 11 System coverage versus number of weeks of historical data (threshold = 50)

"different" from the majority, in other words, as long as the "majority" of trajectories are normal routes, the anomaly detection method will work. We visualize many source and destination paris, and find that 50 is a quite safe threshold. It indicates that at least 55 % of the trajectories will be covered during the night time of non-working day, and over 80 % during the morning, afternoon and evening time of working days.

One way to increase the system coverage is to increase the time period of historical dataset, which means that we consider more amount of historical information. In Fig. 11 we vary the number of weeks used to build the historical database and display the percentage of trajectories covered when the threshold is fixed at 50. We can see that with more weeks the system coverage increases but the increment per week is decreasing. It's because, with longer historical data, the uncovered pairs are those with fewer trajectories in one week and thus the increment is slow.

Another way is to increase the size of source and destination area, which will include more trajectories eventually. The detected anomalies are the delivery routes that are not complying with the normal routes between the $\langle S, D\rangle$ areas. With bigger areas, the system gathers more trajectories between two areas and then some anomalies may become normal and can't be detected in our system. But still, the detected anomalies are correct as they are "few" and "different" from the normal routes. We evaluate the system coverage with respect to different area size (grid edge length) and show the result in Fig. 12. We can see with one month taxi data and threshold 50, in working days, the coverage increases to over 95 % when we choose 1km × 1km grids, while in non-working days, it achieves 90 % with 1.5km × 1.5km grids.

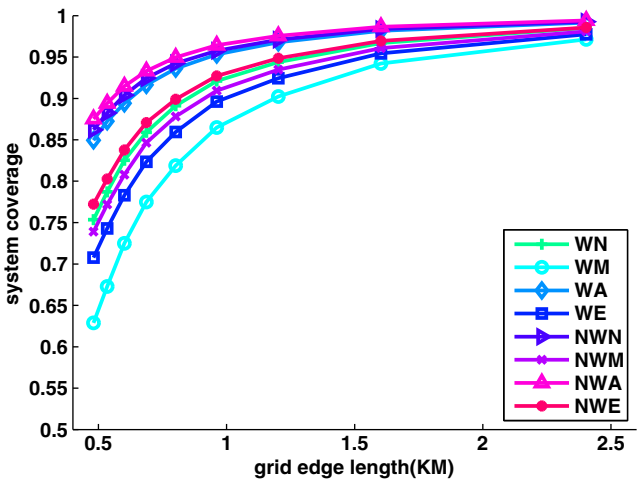


Fig. 12 System coverage versus source/destination grid size (threshold = 50, one month data)

5 Anomalous trajectory analysis

By replaying the GPS records of 7600 taxis from Hangzhou in March 2010 (around 441 million packets), we detected about 0.44 million anomalous trajectories out of 7.35 million trips. This provides us an opportunity to perform a statistical analysis of the anomalous trajectories, in the hope of uncovering common characteristics and motivations of the anomalous behaviours. Motivated by the questions put forth in the introduction, we perform the analyses described below.

5.1 Anomalous driving behavior analysis

In this section we aim to discover what are the common characteristics in the anomalous driving behaviors. Although we cannot know the exact motives behind anomalous behaviours, these analyses provide clues for these motives, and can potentially increase the detection rate of future anomaly detection methods, as they provide the most pertinent conditions that exist when anomalous behaviours occur.

The first aspect we consider is how many anomalous sub-trajectories occur in one trip. We plot these results for the different segments in Fig. 13; we can see that for all time slots the grand majority of anomalous trips have only one anomalous sub-trajectory.

It's also important to consider at what point during the trip the anomalous behaviour begins and ends. We split the trips into thirds and examine where each anomalous sub-trajectory begins and ends. These results are displayed in Table 3, and we can see that anomalous sub-trajectories usually don't start and begin in the same third, and most begin in the first or second third. Although this does not clarify the motivations behind the anomalous behaviours, it does suggest that the anomalous behaviours are occurring as a result of a conscious decision, and not by "accident": if drivers had inadvertedly left their intended route, they would generally return to it immediately. In fact, out of all anomalous trajectories, 27 % of them remain in an anomalous state until they reach the destination, further reinforcing the belief that these anomalous behaviours are not occurring by "accident".

This is further supported by Fig. 14, where we display the areas where most of the anomalous trips began. We can see that many of the places are bus stations, where tourists would generally arrive. It is not surprising that they are responsible for a large fraction of the

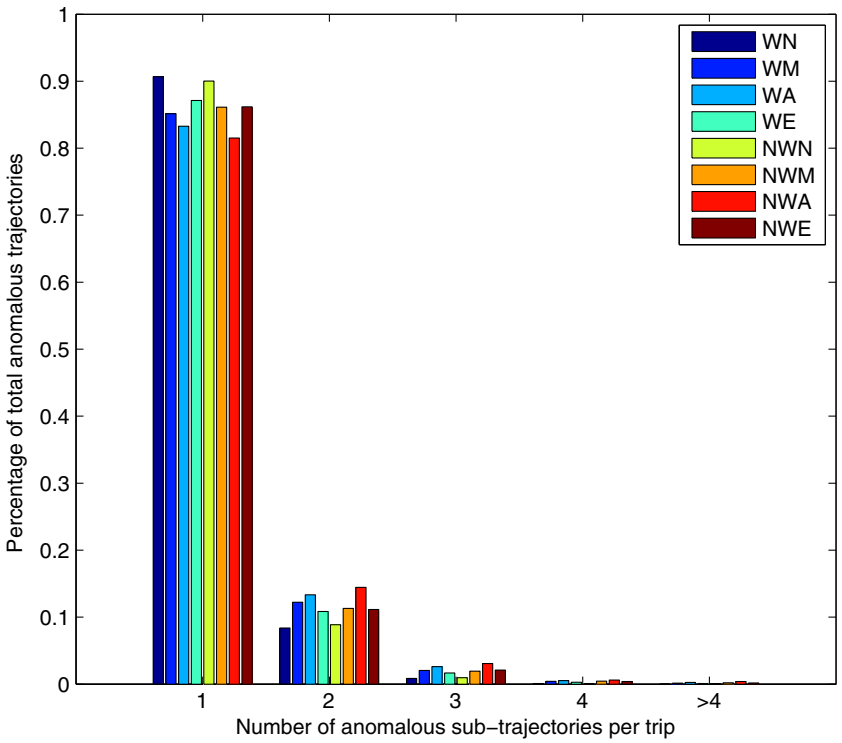


Fig. 13 Number of anomalous sub-trajectories per trip

anomalous trajectories. This further confirms our previous claim that anomalous behaviours are conscious decisions.

In Fig. 15 we display, for varying distances between the source and destination, what proportion of them were anomalous. We can clearly see that the proportion of anomalous trips grows with the trip length, indicating that there is a higher probability of an anomalous trip when the distance between the source and destination is larger. This is consistent with what one would expect from intuition, as it is easier to take detours in longer trips.

We now examine, out of all the anomalous trips, what proportion of the trips are anomalous (i.e. how long are the anomalous sub-trajectories with respect to the whole trip). We display this in Fig. 16 which demonstrates that even though longer distances have a higher potential for anomalous behaviours, over half of the anomalous trips have anomalous sub-sections that are less than 30 % of the trip length.

Table 3 Starting and ending positions of anomalous sub-trajectories

Start	End			Total
	1st third	2nd third	3rd third	
1st third	10 %	19 %	16 %	46 %
2nd third	N/A	12 %	24 %	36 %
3rd third	N/A	N/A	18 %	18 %
Total	10 %	31 %	58 %	

In most research revolving around detecting anomalous taxi driving behaviours, one major interest is 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 a source and a destination; we can safely say that an anomalous trip is *detouring* if the trip distance is longer than this maximal distance. For a source-destination pair, we denote by $maxD$ and $minD$ the maximal and minimal lengths amongst 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 are fraudulent. We could try to determine $maxT$ and $minT$ for the traveling time taken between two points, but due to varying traffic conditions, these values have a high variability. Because of this, for each source-destination pair, we compute the mean time amongst the normal trajectories, μ_T, as well as the standard deviation σ_T. We then define our boundaries as $maxT = \mu_T + \sigma_T$ and $minT = \mu_T - \sigma_T$. In Table 4 we display the distribution of the anomalous trips with respet to these classifications. We can see that over 60 % of all anomalous trajectories are taking long detour and longer time, clearly suggesting that fraud is one of the main motivating factors behind anomalous behaviours.

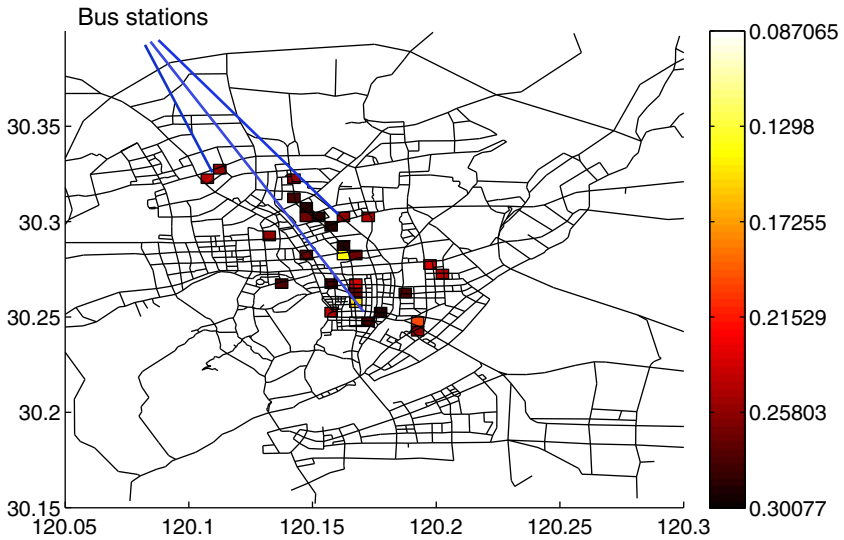


Fig. 14 Areas where most of the anomalous trips began

5.2 Fraudulent behaviors versus revenue

In the last section we observed that most of the anomalous behaviours are due to fraudulent behaviour, and it is safe to say that drivers engage in fraudulent activities for higher revenue, since revenue is based strictly on the distance traveled. It has been previously argued that drivers who take more detours have a higher income [5], in this section we perform a more thorough analysis which demonstrates that the answer is perhaps not as simple as expected.

For each taxi, we compute what we call the *detour ratio* as the total number of anomalous trips divided by the total number of trips. Thus, a higher detour ratio indicates a higher tendency to commit fraud. To estimate the taxi fare for each trip, we use the fare structure of taxi service in Hangzhou to convert the travel distance into income, without considering the waiting time fare compensation (first 3 km, 10 RMB fixed; 2 RMB per kilometer for additional 7 km; over 10 km, 3 RMB per kilometer). In Fig. 17 we plot the monthly revenue versus the detour ratio (each point is a distinct taxi), as well as the distribution over revenue. We can see that the grand majority of taxis are not prone to detour, and those that have a higher tendency to commit fraud (higher detour ratio) usually have a revenue which is only around average. The correlation between the taxi revenue and the detour ratio is 3.57 %, which is quite small. It indicates that fraudulent behaviour does not necessarily result in higher income, despite popular belief and what was argued in [5].

Given that we have revealed that fraudulent behaviours does not necessarily lead to higher income, it is worthwhile trying to discover what differentiates drivers with the highest revenue from drivers with a

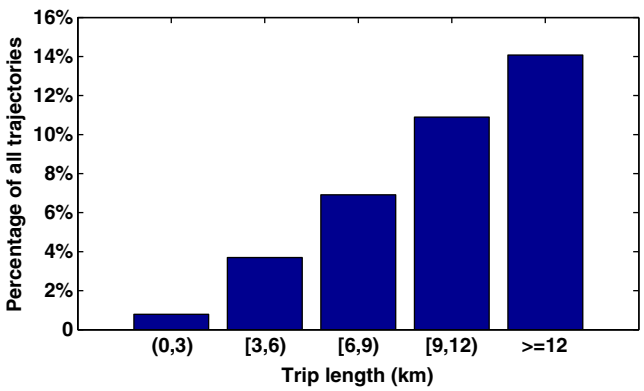


Fig. 15 The anomalous percentage of all trajectories versus distance between source and destination

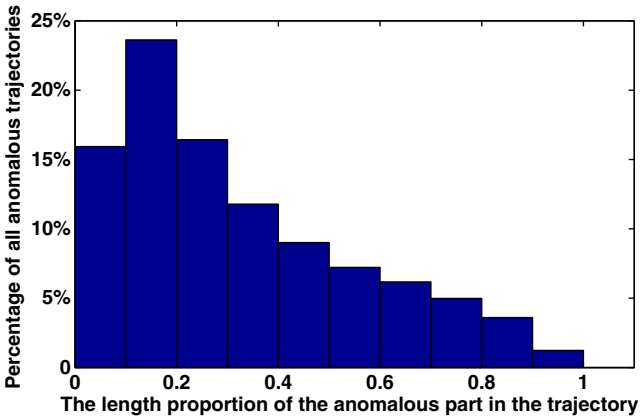


Fig. 16 Percentage of all anomalous trajectories versus Anomalous length proportion of trajectories

Table 4 Distribution of anomalous trajectories with respect to traveling distance and time

Trip length	Travel time		
	$[0, \min T)$	$[\min T, \max T]$	$(\max T, \infty)$
$[0, \min D)$	0.0013	0.0137	0.0117
$[\min D, \max D]$	0.0062	0.1063	0.0881
$(\max D, \infty)$	0.0045	0.1522	0.6162

Table 5 Comparisons of three taxi groups

	Trips (#)	Time (m)	Speed (km/hr)	Ave. rev. (RMB)
High inc.	50.9	15	25.64	26.8k
Average inc.	37.9	30.25	21.95	20k
High detour	32.3	29.64	21.95	18.6k

higher tendency to commit fraud. We compute all the following statistics for the 50 taxis with highest revenue, for the 50 taxis with the highest detour ratio, and for 50 taxis with average revenue. We include this last set of taxis as a baseline to ensure that the differences found between top revenue and top fraudulent drivers are not simply the differences between top and average revenue drivers.

In Table 5 we display the average number of passenger delivery trips for each category, the average amount of time taken to find a new passenger, the average speed while serving a passenger, and the average raw revenue per month. We can see that on average, taxis that have a higher tendency to detour serve far fewer taxis per day than top revenue drivers, and even fewer than average income drivers, although the amount of time taken to find a new passenger and the speed during these trips is about the same as average taxis. This means that taxis that have a higher tendency to detour have performance that is almost indistinguishable from average income drivers, *except* for the average number of trips served. This is most likely a consequence of these drivers taking longer trips. Indeed, the average distance travelled by drivers with a higher detour ratio is about 20 % than the distance travelled by the other taxis.

Now if we go back to explain the reason why [5] derives a different conclusion from ours, the most probable reason for their differing result is that their study is based on the taxi GPS records from a city where the taxi demand is not as high as in Hangzhou, thus taking long detour often means having more income for each trip and as a whole. While in HangZhou, as Taxis are so demanding, thus how to keep the taxi occupied most of the time and deliver the passengers in shortest possible time is more critical for getting an overall high revenue.

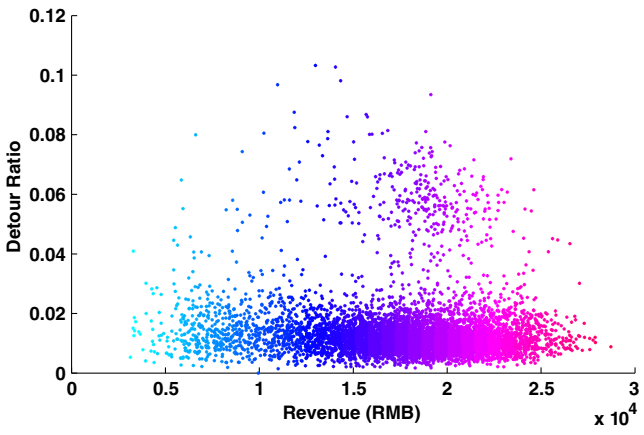


Fig. 17 Revenues of taxis versus detour ratio

6 Summary & discussion

In this paper we have presented a real-time anomaly detection system that can serve the needs of very large taxi fleets. Our system is comprised of three main parts: the data pre-processing, the dataset preparation and the anomaly detection engine. Our implementation makes use of techniques such as caching and the inverted index mechanism to speed up the global computation. Overall, we found that our system is able to respond to all requests within a minute, even when simulating very large taxi fleets (up to 200,000 taxis). We also investigated the system coverage issue and proposed several ways to alleviate the problem. Overall, our system provides extensive coverage in reasonable conditions (three to four weeks historical data).

We implemented our system on a real life dataset from Hangzhou, comprised of 7,600 taxis, emitting GPS signals at a rate of once per minute. We collected the data over a month's time, which resulted in 7.35 million trips, out of which 0.44 were detected as anomalous. Given this large set of data, we conducted a thorough analysis of the anomalous trajectories to uncover the motivation behind the anomalous behaviours, as well as to determine whether there is an economical incentive to commit fraud. From the analysis we conducted we can conclude that not only are anomalous behaviours usually a result of a conscious decision, but most of the anomalous events are fraudulent. Nevertheless, despite the desire to increase revenue, a higher inclination to fraud does not correspond with a higher income. In order to increase one's income, one must learn efficient ways to find new passengers and deliver them to their destination as quickly as possible.

References

1. Brian ZD, Maas AL, Dey AK, Bagnell JA (2008) Navigate like a cabbie: probabilistic reasoning from observed context-aware behavior. In: Proceedings of the 10th international conference on ubiquitous computing. Seoul, Korea, pp 322–331
2. Bu Y, Chen L, Fu AW, Liu D (2009) Efficient anomaly monitoring over moving object trajectory streams. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, pp 159–168
3. Chen C, Zhang D, Castro PS, Li N, Sun L, Li S (2011) Real-time detection of anomalous taxi trajectories from incoming GPS traces. In: Proceedings of the 8th international ICST conference on mobile and ubiquitous systems
4. Faghri A, Hamad K (2002) Application of GPS in traffic management systems. GPS Solutions 5:52–60
5. Ge Y, Xiong H, Liu C, Zhou Z-H (2011) A taxi driving fraud detection system. In: Proceedings of the IEEE international conference on data mining, pp 181–190
6. Ge Y, Xiong H, Zhou Z, Ozdemir H, Yu J, Lee K (2010) Top-eye: top-*k* evolving trajectory outlier detection. In: Proceedings of the 19th ACM international conference on information and knowledge management, pp 1733–1736
7. Lazer D, Pentland A, Adamic L, Aral S, Barabasi A-L, Brewer D, Christakis N, Contractor N, Fowler J, Gutmann M, Jebara T, King G, Macy M, Roy D, Van Alstyne M (2009) Computational social science. Science 323(5915):721–723
8. Lee J-G, Han J, Li X (2008) Trajectory outlier detection: a partition-and-detect framework. In: Proceedings of IEEE international conference on data engineering, pp 140–149
9. Li B, Zhang D, Sun L, Chen C, Li S, Qi G, Yang Q (2011) Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset. In: IEEE international conference on pervasive computing and communications workshops (PERCOM Workshops), pp 63–68
10. Liao Z (2003) Real-time taxi dispatching using global positioning systems. Commun ACM 46:81–83
11. Liu L, Andris C, Ratti C (2010) Uncovering cabdrivers' behavior patterns from their digital traces. Comput Environ Urban Syst 34:541–548
12. Phithakkitnukoon S, Veloso M, Bento C, Biderman A, Ratti C (2010) Taxi-aware map: identifying and predicting vacant taxis in the city. In: Ambient intelligence: first international joint conference. Malaga, Spain, pp 86–95
13. Qi G, Li X, Li S, Pan G, Wang Z, Zhang D (2011) Measuring social functions of city regions from large-scale taxi behaviors. In: The 9th IEEE international conference on pervasive computing and communications, WIP. Seattle, USA, pp 384–388
14. Quiroga CA, Bullock D (1998) Travel time studies with global positioning and geographic information systems: an integrated methodology. Transp Res, Part C Emerg Technol 6:101–127
15. Quiroga C, Bullock D (1999) Measuring control delay at a signalized intersections. J Transp Eng 125:271–280
16. Veloso M, Phithakkitnukoon S, Bento C (2011) Urban mobility study using taxi traces. In: Proceedings of the 2011 international workshop on trajectory data mining and analysis. New York, USA, pp 23–30
17. Yuan J, Zheng Y, Zhang C, Xie W, Xie X, Huang Y (2010) T-drive: driving directions based on taxi trajectories. In: Proceedings of the 18th ACM international conference on advances in geographic information systems. San Jose, CA, pp 99–108
18. Yuan J, Zheng Y, Zhang L, Xie X, Sun G (2011) Where to find my next passenger? In: Proceedings of the 13th ACM international conference on ubiquitous computing. Beijing, China, pp 109–118
19. Zhang D, Guo B, Yu Z (2011) The emergence of social and community intelligence. IEEE Computer 44:21–28
20. Zhang D, Li N, Zhou Z-H, Chen C, Sun L, Li S (2011) iBAT: detecting anomalous taxi trajectories from GPS traces. In: Proceedings of the 13th ACM international conference on ubiquitous computing, pp 99–108
21. Zheng Y, Liu Y, Yuan J, Xie X (2011) Urban computing with taxicabs. In: Proceedings of the 13th ACM international conference on ubiquitous computing, pp 89–98
22. Zheng Y, Zhang L, Xie X, Ma W-Y (2009) Mining interesting locations and travel sequences from GPS trajectories. In: Proceedings of the 18th international conference on World Wide Web, pp 791–800
23. Zobel J, Moffat A (2006) Inverted files for text search engines. ACM Comput Surv 38:1–56