Detecting Unmetered Taxi Rides from Trajectory Data

Qing Liao, *Member, IEEE*, Xibo Zhou, *Member, IEEE*, Ye Ding, *Member, IEEE*, Fengchao Peng, *Member, IEEE*, Qiong Luo, *Member, IEEE*, Lionel M. Ni, *Fellow, IEEE*

Abstract—Taxi fraud has become a serious problem in many large cities, where passengers are overcharged by taxi drivers in various ways. Researchers have developed a number of methods to detect taxi frauds with the assumption that fraudulent trips, among normal trips, are recorded by taximeters. In this paper, different from the previous work, we identify a new type of taxi fraud called unmetered taxi rides, where taxi drivers carry passengers without activating the taximeters. Since these fraudulent rides are not recorded by taximeters, previous detection approaches cannot directly apply to them. Hence, we propose a novel fraud detection system specifically designed for unmetered taxi rides. Our system uses a learning model to detect unmetered trajectory segments that are similar to metered rides, and introduces a heuristic algorithm to construct maximum fraudulent trajectories from the trajectory dataset. We have conducted detailed experiments on real-world datasets, and the results show that the proposed system can detect unmetered taxi rides effectively and efficiently.

Index Terms—Taxi fraud detection, trajectory, anomaly detection.

1 Introduction

In modern cities, taxi service is an important part of the public transportation system, providing convenience for our daily life. However, in recent years, taxi fraud, in terms of overcharging passengers those whom have no idea about what deception the taxi driver used, has become a serious problem in many large cities, which causes strong complaints from passengers and damages the reputation and fame of taxi service.

There are many various forms of taxi frauds, including: 1) detour, where taxi drivers overcharge passengers by deliberately taking unnecessary detours; 2) taximeter tampering, where taxi drivers tamper the taximeters so that they record longer distances than the actual; and 3) passenger denial, where taxi drivers refuse to deliver passengers to their destinations due to various reasons. These fraudulent behaviors usually have evident properties that differ from normal taxi rides and we could make use of these properties. For example, the distance of a detour is usually significantly longer than normal paths between a pair of source and destination [2]; the reported speed of the taxi with a tampered taximeter tends to be higher than its actual speed [3]; and taxi drivers with a higher income are more likely to refuse passengers traveling to unpopular areas because there are more foot passenger distribution and traffic flow in these areas. [4]. Many approaches are proposed to detect these behaviors by utilizing these properties.

In this paper, we identify another type of taxi fraud, namely **unmetered taxi rides**, where taxi drivers carry passengers without activating the taximeters. In this way, taxi drivers can overcharge passengers by setting high prices without being recorded. Unmetered taxi rides are a common and severe problem in large cities with complex traffic situations. For example, 146 taxi drivers were caught for illegally refusing or arbitrarily pricing rides during only three days in Guangzhou, China [6]. Unmetered taxi rides are problematic to the society due to three major reasons. First, they hurt the quality of taxi service, especially for tourists that are unfamiliar with the city transportation, or during rush hours when the supply of taxis is short. Second, they break the consistency of taxi pricing, which often leads to unfair competition between taxis and causes trouble for traffic management. Third, they are difficult to track or regulate by taxi companies, because *no meter record is collected for these unmetered trips*. Therefore, it is necessary to develop a method to detect such type of fraud.

Fig. 1: A taxi driver is trying to overcharge the passengers in Lam Kui Fong, Hong Kong [5].

Unfortunately, existing approaches are often not suitable for detecting unmetered taxi rides. First, existing approaches usually assume that the fraud trips are recorded by taximeters, which is no longer true for unmetered taxi frauds. An alternative method is

- *Qing Liao is with Department of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), China. E-mail: liaoqing@hit.edu.*
- *Xibo Zhou, Fengchao Peng and Qiong Luo are with Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong. E-mail: {xzhouaa, pfengaa, luo}@ust.hk.*
- *Ye Ding is with School of Software Engineering and Cyperspace Science, Dongguan University of Technlogy, China. E-mail: dingye@dgut.edu.cn.*
- *Lionel M. Ni is with University of Macau, Macau. E-mail: ni@umac.mo.*

An earlier version of this work appeared in the Proceedings of IEEE International Conference on Big Data [1], 2017.

to utilize the occupancy information collected from seat sensors on taxis. However, the occupancy information contained in the trajectory dataset is inaccurate, which may be caused by aging or deliberate damage of these sensors. Based on our observation, 12% of the taxis in our dataset have rides with metered records yet the occupancy status showed vacant. Hence, the occupancy information is not reliable and cannot be directly used to detect unmetered taxi frauds. Second, existing approaches assume that fraud trips exhibit anomalous behaviors from normal metered trajectories, which is also not true for unmetered taxi rides. Based on our observation, the driving behavior or dynamics of unmetered fraud rides are more similar to normal metered trips than to vacant taxi trajectories. Hence, it is unsuitable to perform anomaly detection on metered trajectories. Third, existing approaches treat each metered trajectory as a single object, and aim to find anomalous objects. However, an unmetered taxi ride is often a partial segment of an entire unmetered trajectory including when the taxi was vacant. Thus, we aim to detect anomalous trajectory fragments rather than complete trajectory objects. Unfortunately, existing approaches are not suitable for this task.

In this paper, we propose a novel taxi fraud detection system specifically for unmetered taxi rides. The system finds anomalous trajectories that are not recorded by the taximeter but have driving behaviors similar to regular metered trips by utilizing both the taxi trajectory data and the taximeter records. First, we propose a learning model to capture the different features of trajectories when they are metered versus not. Second, we implement a heuristic algorithm to construct anomalous unmetered trajectories from the entire trajectory dataset. The contributions of this paper lie in the following aspects:

- We introduce a new type of taxi fraud called **unmetered taxi rides**. Different from previous taxi frauds, we aim to find anomalous taxi trajectories that are similar to metered trips but not recorded by taximeters.
- We design a novel system for detecting unmetered taxi rides. We propose a learning model to predict the passenger occupancy status of taxis from unmetered trajectories, and implement a heuristic algorithm to construct anomalous unmetered trajectories.
- We evaluate our system on real-world taxi trajectory data. The results show that our system is effective to find unmetered taxi rides.

In the remainder of this paper, we first discuss the related works in Section 2, and then introduce the problem definitions and the system in Section 3. Next, we describe each component of our system in Section 4, 5, and 6, respectively. We evaluate our system in Section 7, and conclude the paper in Section 8.

2 Related Work

Existing approaches for general anomaly detection on trajectory data can be categorized into two groups: spatial anomaly detection and temporal anomaly detection. In spatial anomaly detection, Lee *et al.* [7] proposed a partition-and-detect framework to detect outlying sub-trajectories based on distance and density. Bu *et al.* [8] proposed a window-based approach to detect anomalous sub-trajectories inside a window based on distance. Ge *et al.* [9] proposed a spatial anomaly detection framework for evolving trajectories based on travel direction and density. Chen *et al.* [10] focused on detecting trajectories that deviate significantly from histories. Lv *et al.* [11] proposed a prefix tree-based algorithm to calculate the travel frequency of routes, and implemented a clustering-based approach to find the anomalous trajectories. In temporal anomaly detection, Li *et al.* [12] tried to identify temporal outliers by utilizing historical similarity trends. Zhu *et al.* [13] proposed a novel algorithm that takes travel time into consideration and detects anomalous trajectories based on temporal route popularity. These approaches are not directly applicable to our problem.

On taxi fraud detection, two groups [2] [14] introduced a scenario of taxi fraud, where some taxi drivers deliberately take unnecessary detours in order to overcharge passengers. They transformed the taxi fraud detection problem into finding anomalous trajectories from all the trajectories with the same source-destination pairs, and used the spatial distance as the main feature to design the anomaly detection method. Liu *et al.* [3] introduced another case, where fraud taxi drivers modify the taximeters to a smaller scale so that they record longer distances than the actual ones. They modeled taxi fraud behaviors in a trajectory-free and map-free scenario, constructed a model by the speed information instead of location or distance, and proposed a speed-based clustering method to detect taxi fraud. To the best of our knowledge, we are the first to target at the unmetered taxi ride detection problem.

3 Preliminaries

3.1 Problem Definition

Definition 1 (Tracing Record). A tracing record r of a taxi is denoted as a tuple $r = \langle id, t, p, o \rangle$, where $r(id)$ is the taxi id and it is used to identify a unique taxi objection,$r(t)$ is the record time, $r(p)$ is the location point of the taxi at $r(t)$, and $r(o)$ is the occupancy status of the taxi at $r(t)$.

A location point is represented by its latitude and longitude. The occupancy status is a boolean value, recorded as 1 if the taxi is **occupied** by a passenger and 0 if the taxi is **vacant**.

Definition 2 (Taximeter Record). A taximeter record m of a metered trip is denoted as a tuple $m = \langle id, st, et \rangle$, where $m(id)$ is the taxi id, $m(st)$ is the start time of the trip where the taxi driver has activated the meter counter in his taxicab, and $m(et)$ is the end time of the metered trip and it is recorded when passger confirms that he has arrived destination.

Given a tracing record r and a set of taximeter records M, we say r is **metered** if $\exists m \in M$ where $r(id) = m(id)$ and $m(st) \leq r(t) \leq m(et)$, and **unmetered** otherwise.Then,We can use this data to detect whether a taxi driver takes a passger into some time-consuming detour.

Definition 3 (Trajectory). A trajectory l of a taxi with id tid is a sequence of tracing records denoted as $l = (r_1, r_2, \cdots, r_n)$, where $r_i(id) = tid$ for $i = 1, \cdots, n$. We denote $r_i \in l$ for $i = 1, \cdots, n$ and $|l| = n$.

For simplicity, we denote the taxi id of trajectory l as $l(id)$, where $l(id) = r_i(id)\ \forall r_i \in l$. Given the trajectory l, a trajectory $l' = (r_s, r_{s+1}, \cdots, r_d)$ of consecutive tracing records where $1 \leq s < d \leq |l|$ is called a sub-trajectory of l, denoted as $l' \subseteq l$.

Definition 4 (Unmetered Trajectory). Given a trajectory l, we say l is an *unmetered trajectory* if r is unmetered $\forall r \in l$.

Definition 5 (Fraud Trajectory). Given an unmetered trajectory l, we say l is a *fraud trajectory* if r is occupied $\forall r \in l$.

In order to detect fraud trajectories, it is necessary to determine whether a tracing record r is both unmetered and occupied. However, as mentioned in Section 1, the occupancy status contained in real-world trajectory datasets is usually imprecise. Hence, we need to predict the occupancy for each tracing record before detecting fraud trajectories.

Definition 6 (Occupancy Prediction Problem). Given a set of unmetered trajectories L, for each tracing record $r \in l$ where $l \in L$, predict the value of $r(o)$.

After detecting the occupied tracing records, our next problem is to find the maximum fraud trajectories from the unmetered trajectory set.

Definition 7 (Maximum Fraud Trajectory Problem). Given a set of unmetered trajectories L, for each trajectory $l \in L$, find a *maximum fraud trajectory* l' where:

1) $l' \subseteq l$,
2) l' is a fraud trajectory, and
3) $|l'| \geq |l''|, \forall l'' \subseteq l$ where l'' is a fraud trajectory.

In this paper, we will solve both Problem 6 and 7 in the following sections.

3.2 Overview

Figure 2 shows the process of our taxi fraud detection system, which is divided into three phases:

- *Pre-processing*: Raw trajectories are segmented into metered and unmetered trajectories based on taximeter records. The metered trajectories are filtered from the trajectory set, and the unmetered trajectories are then map-matched into connected paths constrained to the road network.
- *Feature Extraction*: The map-matched unmetered trajectories are first fragmented into unit segments with constant length. Then, a vector of features are calculated for each unit segment, including spatial-temporal features (such as average travel speed and arc-chord ratio) and statistical features (such as transition frequency between roads and cheating occurrence of each taxi ID).
- *Anomaly Detection*: Given the feature vector set extracted from fragmented unmetered unit segments, we implement an integrated predictor to detect the occupied instances. After detecting the unit segments that are occupied but not metered, a trajectory construction algorithm is performed to heuristically form the longest fraud trajectories.

4 Pre-processing

4.1 Trajectory Filtering

Given the set of trajectories and the taximeter records, our goal is to find the trajectories that are occupied and unmetered. Since we assume the taximeter records are accurate, the first task is to obtain all the unmetered trajectories from the trajectory set. Due to the difference of data formats for these two dataset, we implement a matching method to segment the trajectory dataset into metered and unmetered trajectories. The process is as follows:

First, we sort all the taximeter records by taxi ids. For each taxi id, we sort its taximeter records by start time. Then, for each taxi id, we can obtain all the metered and unmetered time periods. More specifically, for a sorted taximeter records of taxi id tid denoted as $(tid, st_1, et_1), (tid, st_2, et_2), \cdots, (tid, st_n, et_n)$, the metered time periods are $(st_1, et_1), (st_2, et_2), \cdots, (st_n, et_n)$, whereas the unmetered time periods are (et_1, st_2), $(et_2, st_3), \cdots, (et_{n-1}, st_n)$. Next, we obtain all the trajectories for each taxi id, and locate each tracing record of the trajectories into the matching time period. More specifically, given a taxi id tid with its metered and unmetered time periods obtained, and a trajectory l where $l(id) = tid$, the matching time period for $r \in l$ is $(st_i, et_i), st_i \leq r(t) \leq et_i$ or $(et_i, st_{i+1}), et_i \leq r(t) \leq st_{i+1}$. Finally, for each taxi id and each of its unmetered time period, we fetch all the tracing records located in it, and construct an unmetered trajectory by sorting these records by time.

4.2 Map-matching

In practice, the location information of raw trajectories are usually imprecise due to the measurement noise and sampling errors. In order to extract spatial features from these trajectories, we perform a map-matching task by aligning the location points to the road networks. More specifically, given a trajectory l and road network $G = (V, E)$, where V is a set of road intersections, and E is a set of road segments, a map-matching task finds a path $P = (e_1, e_2, \cdots, e_n)$, where e_i and e_{i+1} are connected for $i = 1, 2, \cdots, n-1$, such that each location point $r_i(p)$ of $r_i \in l$ is matched to exactly on road segment $e_j \in E$. The sequence of match points is denoted as $(p'_1, p'_2, \cdots, p'_n)$.

The definitions and the output format of the map-matching task are as follows.

Definition 8 (Road Segment). A road segment e is a directed polyline between two road intersections v_i and v_j, and there is no other road intersection on e. We denote $v_i \in e$ and $v_j \in e$.

Definition 9 (Road Network). A road network is a weighted directed graph $G = (V, E)$, where V is a set of road intersections (or vertices), and E is a set of road segments (or edges). The weight of a road segment is represented by its properties.

Definition 10 (Path). A path $P = (e_1, e_2, \cdots, e_n)$ is a sequence of road segments where e_i and e_{i+1} are connected for $i = 1, 2, \cdots, n-1$. Two road segments e_i and e_j are connected if there exists some intersection v such that $v \in e_i$ and $v \in e_j$.

Definition 11 (Match Point). Given a tracing record r and a road segment e, we say $r(p)$ is **matched** to e if r was sampled when the taxi was moving on e, and p' is the match point of $r(p)$ on e.

Definition 12 (Map-Matching). Given a trajectory l and a road network G, a map-matching task finds a path P, such that each location point $r_i(p)$ of $r_i \in l$ is matched to exactly on road segment $e_j \in E$. The resulting sequence of match points is denoted as $(p'_1, p'_2, \cdots, p'_n)$.

In this work, we use the Hidden Markov Model (HMM)-based map-matching algorithm [15]. The algorithm builds a hidden Markov model for each map-matching task, where a trajectory represents an observation sequence, each location point is an observation, and each candidate road segment is a hidden state. The algorithm defines the emission and transition probabilities based on distance information, and uses Viterbi algorithm to compute the best path, which gives an inference of the correct

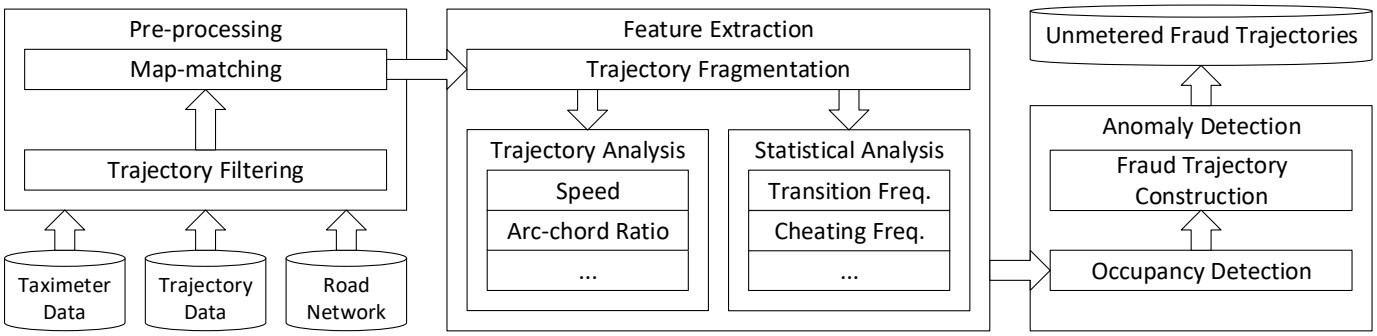


Fig. 2: The work-flow of taxi fraud detection.

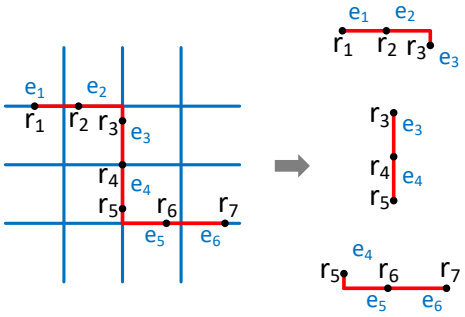


Fig. 3: An example of trajectory fragmentation (w=2).

road segment for each location point. Each two consecutive match points are connected by the shortest path between them along the road network, which guarantees the connectivity of the entire path.

5 Feature Extraction

5.1 Trajectory Fragmentation

Given the set of unmetered trajectories, before we start to extract features, it is necessary to fragment these trajectories into small pieces. There are two reasons: 1) an unmetered trajectory contains all the tracing records between two continuous metered trips of a taxi. Due to the diversity of occupancy rates of taxi drivers, the lengths of unmetered trajectories are significantly varied; and 2) each unmetered trajectory might be mixed with occupied and vacant tracing records, it is inappropriate to treat it as a single instance. In this work, we fragment the unmetered trajectories into unit segments with a constant length, and extract the corresponding map-matched path for each unit segment, as demonstrated in Figure 3. In this example, trajectory $(r_1, r_2, \cdots, r_7)$ is fragmented into three fragments of constant length 2: (r_1, r_2, r_3), (r_4, r_5, r_6), and (r_7, r_8, r_9). Formally, given an unmetered trajectory l and a constant w, l is fragmented into $\lceil (|l|-1)/w \rceil$ unit segments, represented as:

$$(r_1, \cdots, r_{w+1}), (r_{w+1}, \cdots, r_{2w+1}), \cdots \tag{1}$$

Each unit segment is denoted as $u = (r_i, \cdots, r_j)$, we say $u \subset l$ if u is fragmented from l. Meanwhile, suppose the map-matched path of l is $P = (e_1, e_2, \cdots, e_n)$, then the map-matched path of $u = (r_i, \cdots, r_j)$ is $P' = (e_s, e_{s+1}, \cdots, e_t)$ where r_i is matched to e_s and r_j is matched to e_t. We say $u = (r_i, \cdots, r_j)$ is occupied if:

$$\frac{\sum_{k=i}^{j} r_k(o)}{j-i+1} \geq 0.5 \tag{2}$$

or vacant otherwise.

In the following steps of our system, the feature extraction, occupancy detection and fraud trajectory construction are based on unit segments rather than tracing records.

5.2 Feature Analysis

After fragmenting the unmetered trajectories into unit segments, we extract features that have impact on occupancy. In general, there are two types of features extracted by our system, namely spatial-temporal features and statistical features. The spatial-temporal features reflect the individual pattern of each trajectory, including average speed, tortuosity, arc-chord and time; and the statistical features reflect the statistical information of the trajectories, including road transition frequency and taxi cheating frequency. We will introduce each feature in detail:

5.2.1 Average Speed

Given a unit segment $u = (r_i, \cdots, r_j) \subset l$, the average speed v_{avg} of u is:

$$v_{avg} = \frac{\text{rdist}(p'_i, p'_j)}{r_j(t) - r_i(t)} \tag{3}$$

where p'_i is the match point of r_i, p'_j is the match point of r_j, and $\text{rdist}(p'_i, p'_j)$ is the driving distance along the map-matched path P' of u from p'_i to p'_j.

5.2.2 Arc-chord Ratio

Given a unit segment $u = (r_i, \cdots, r_j) \subset l$, the arc-chord ratio τ_a of u is:

$$\tau_a = \frac{\text{rdist}(p'_i, p'_j)}{\text{cdist}(p'_i, p'_j)} \tag{4}$$

where p'_i is the match point of r_i, p'_j is the match point of r_j, $\text{rdist}(p'_i, p'_j)$ is the driving distance along the map-matched path P' of u from p'_i to p'_j, and $\text{cdist}(p'_i, p'_j)$ is the great circle distance between p'_i and p'_j.

5.2.3 Curvature

As described in section 4.2, the map-matched path is a sequence of road segments, thus the path from p'_i to p'_j is can be represented as a polyline, say $(d_1, d_2, \cdots, d_m)$ where each d_i represents a line segment. The curvature τ_c of $(d_1, d_2, \cdots, d_m)$ is:

$$\tau_c = \frac{\sum_{k=1}^{m-1}(\frac{\angle(d_k, d_{k+1}) \times \pi}{180} \times \frac{1}{\text{cdist}(d_k)})^2}{\text{cdist}(p'_i, p'_j)} \tag{5}$$

where $\angle(d_k, d_{k+1})$ is the intersection angle of d_k and d_{k+1} and $\text{cdist}(d_k)$ is the length of d_k.

5.2.4 Time

We first split the time in one day into a number of slots μ_t with a constant period, and then flatten each spatial-temporal feature into a vector based on the corresponding time slot. More specifically, given a unit segment $u = (r_i, \cdots, r_j)$ and a constant time period Δ_t, the index h of the corresponding time slot for u is:

$$h = \frac{\max(r_i(t), r_j(t))}{\Delta_t} \tag{6}$$

For each spatial-temporal feature f extracted from u, it is flattened as $f_{flat} = \langle \underbrace{0, \cdots, 0}_{h}, f, \underbrace{0, \cdots, 0}_{\mu_t - h - 1} \rangle$, where num_slot is the total number of time slots.

5.2.5 Road transition frequency

Given a map-matched path $P = (e_1, e_2, \cdots, e_n)$, we say e_{i+1} is a **follower** of e_i in P. Given a set of map-matched path P_{set} and a road segment e, we can obtain a set of followers E_f of e where $\forall e_f \in E_f$, e_f is a follower of e in at least one path in P_{set}. For each follower $e_f \in E_f$, the road transition frequency from e to e_f is $Fr(e, e_f) = \mu_{e_f}/\mu_{E_f}$, where μ_{e_f} is the number of times when e_f is a follower of e in a path $P \in P_{set}$, and μ_{E_f} is the number of times when e has a follower in a path $P \in P_{set}$. Given a unit segment $u = (r_i, \cdots, r_j)$ along with its map-matched path $P' = (e_s, e_{s+1}, \cdots, e_t)$, the road transition frequency of u is:

$$Fr(u) = \frac{\sum_{k=s}^{t-1} Fr(e_k, e_{k+1})}{t - s + 1} \tag{7}$$

In order to distinguish the road transition frequency for occupied and vacant unit segments, we count the road transition frequency on P_{set} of occupied and vacant unit segments, respectively. The road transition frequency for occupied and vacant unit segments are denoted as $Fr_o(u)$ and $Fr_v(u)$, receptively.

5.2.6 Taxi cheating frequency

Given a taxi id tid and a set of its unmetered unit segments, the taxi cheating frequency of tid is:

$$Fr(tid) = \frac{\mu_o}{\mu_v} \tag{8}$$

where μ_o is the number of occupied unit segments, and μ_v is the number of vacant unit segments.

After extracting these features from the fragmented unmetered unit segments, we perform several post-processing tasks to calibrate the feature table. Since the spatial-temporal features (except time) are directly calculated from the dataset, we normalize them to 0-1 range. Note that these features are flattened into vectors based on time, therefore the normalization is implemented within each column of each feature matrix. In our experiment, we find that taxi cheating frequency is not quite effective as an independent feature, but achieves better performance as a weight combined with spatial-temporal features. Hence, we transform each spatial-temporal feature f in terms of $f_{comb} = \langle \underbrace{0, \cdots, 0}_{h}, f \times Fr(tid), \underbrace{0, \cdots, 0}_{\mu_t - h - 1} \rangle$.

Meanwhile, we calculate road transition frequency in order to capture the contiguity of the consecutive tracing records along trajectories. Therefore, this feature is used in maximum fraud trajectory construction, which will be described in Section 6.2.

6 Anomaly Detection

6.1 Occupancy Detection

In this paper, we develop an integrated predictor by utilizing the features introduced in Section 5.2 to detect the occupied segments from the set of fragmented unmetered unit segments. The structure of our model is shown in Figure 4. First, we build a stochastic gradient descent (SGD) model on each transformed spatial-temporal feature to make a preliminary prediction of occupancy. Then, we integrate the normalized likelihoods of the prediction results from each SGD model. Formally, given an instance u and a threshold λ, the occupancy of u is:

In order to detect the occupied segments from the fragmented unmetered unit segments using our model, there are two issues that need to be solved.

First, as introduced in section 1, the occupancy information contained in taxi trajectory dataset is not precise, therefore we need to find a way to obtain the ground truth. In this work, we use the information of metered trajectories to select reliable occupancy labels. As described in section 4.1, the trajectory dataset is segmented into metered and unmetered trajectories. Given a set of metered trajectories L and a taxi id tid, the reliability on occupancy of tid is defined as:

$$R(tid) = \frac{\mu_{occupy}}{\mu_{all}} \tag{9}$$

where μ_{occupy} is the number of tracing records $r_i \in L$ where $r_i(id) = tid$ and $r_i(o) = 1$, and μ_{all} is the number of tracing records $r_i \in L$ where $r_i(id) = tid$. In this work, we filter out the taxis with reliability less than 90%.

Second, based on our observation, the percentage of occupied unmetered unit segments out of the entire unmetered unit segments is less than 5%. With such a skewed label distribution, it is easy for a classification model to ignore the occupied labels and predict all the instances as vacant. In order to solve this problem, we implement a *randomized training process* as shown below. We first divide the training set into occupied and vacant feature sets, and extract statistical features on the entire occupied and vacant sets, respectively. Then, we evenly partition the occupied and vacant set into n_p groups, respectively. Since there are only 5% occupied segments, the size of each vacant group is 19 times as large as the size of each occupied group. In order to balance the sampling size of the training set for occupied and vacant instances, we further split each vacant group into 19 piles with equal sizes. Thus, the entire training set is partitioned into n_p occupied and $19 \times n_p$ vacant piles. Next, in the training phase, each time we randomly pick one occupied and one vacant pile to form a temporal training set, build the classification model, and use it to predict labels of the rest of occupied and vacant groups. Since there are $19 \times n_p \times n_p$ combinations to form the temporal training

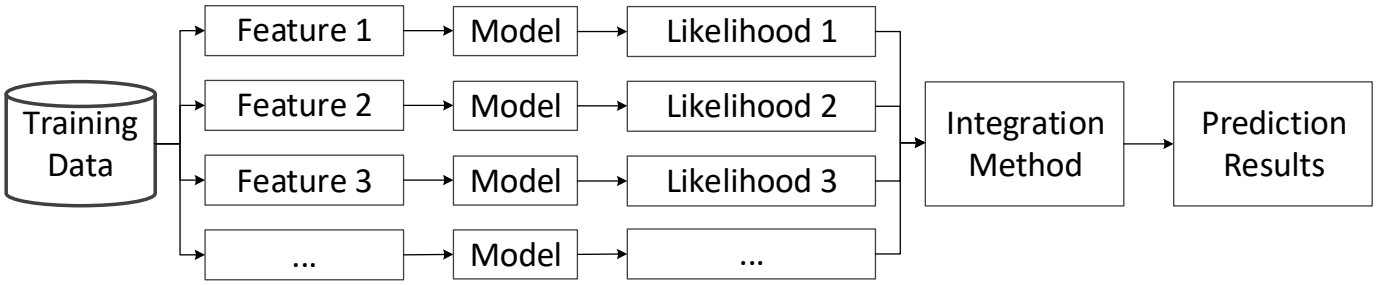


Fig. 4: The structure of our integrated prediction model.

set, it is time-consuming to validate all the combinations. To deal with this, we stop the cross-validation as long as all the occupied set has been used at least once. Finally, we use the trained model to predict the occupancy labels of the test set.

6.2 Maximum Fraud Trajectory Construction

Since the occupied unit segments detected using our model described in section 6.1 are discrete, it is necessary to implement a post-processing method to construct complete occupied unmetered sub-trajectories. In this work, we implement a heuristic method to construct longest occupied unmetered sub-trajectories. Given a sequence of unit segments U fragmented from trajectory l, a set of unit segments $U_t \subset U$ where each $u \in U_t$ is detected to be occupied, and a constant γ, our sub-trajectory construction algorithm is shown in Algorithm 2.

The algorithm starts from each detected occupied unit segment $u \in U_t$, and heuristically search its previous $(\S 5)$ and following $(\S 10)$ unit segment in order, respectively. For each target unit segment, if it is initially detected as vacant, we utilize the road transition frequency to infer the suspicion of it being occupied $(\S 6, 11)$. More specifically, given a unit segment u, the suspicion of u being occupied is:

Algorithm 1 Maximum Fraud Trajectory Construction.

Require: The sequence of unit segments $U = (u_1, \cdots, u_n)$; the set of occupied unit segments $U_t = \{u_{i_1}, u_{i_2}, \cdots, u_{i_k}\}$ where $1 \leq i_1 < i_2 < \cdots < i_k \leq n$; constant ϵ and γ
Ensure: The set of occupied trajectories L_t

$L_t = \emptyset$; $U_o = [1$ if $u_i \in U_t$ else 0 for $u_i \in U]$
for $j = 1; j \leq k; j++$ **do**
  $i_s = i_{j-1} + 1$ if $j > 1$ else 1
  $i_t = i_{j+1} - 1$ if $j < n$ else n
  for $q = i_j - 1; q \geq i_s; q--$ **do**
    if $U_o[q] = 0$ and $S(u_q) \geq \epsilon$ **then**
      $U_o[q] = 1$
    else
      break
    end if
  end for
  for $q = i_j + 1; q \leq i_t; q++$ **do**
    if $U_o[q] = 0$ and $S(u_q) \geq \epsilon$ **then**
      $U_o[q] = 1$
    else
      break
    end if
  end for
end for
$l = \emptyset$
for $j = 1; j \leq n; j++$ **do**
  if $U_o[q] = 1$ **then**
    $l = l + \{u_j\}$
  else
    if $|l| \geq \gamma$ **then**
      $L_t = L_t + \{l\}+$
      $l = \emptyset$
    end if
  end if
end for
if $|l| \geq \gamma$ **then**
  $L_t = L_t + \{l\}$
end if
return L_t

7 Evaluation

7.1 Experiment Setup

7.1.1 Experiment Environment

The experiments are conducted on a server with Intel Core i5-4590 CPU and 16 GB RAM. The operating system is Ubuntu 14.04, and the code is written in Python 2.7.6.

7.1.2 Real World Dataset

In this paper, we use the dataset collected from a large city in China. The dataset contains 154 million taxi tracking records and 4 million taximeter records of 15,231 taxis for 26 days [16]. Based on our observation, 69% of the taxi tracking records are unmetered, and 5% of the unmetered taxi tracking records are occupied. The road network used for map-matching consists of 25,613 intersections and 36,451 road segments. As described in Section 6, we use the trajectories of taxis with reliability greater than 90% as ground truth.

7.1.3 Evaluation Metrics

In this paper, we use *precision* to evaluate the effectiveness of our occupancy detection model, which is denoted as:

$$precision = \frac{\mu_{tp}}{\mu_p} \tag{10}$$

where μ_{tp} is the number of occupied unit segments that are correctly detected, and μ_p is the total number of unit segments detected as occupied.

Meanwhile, we use *recall* to evaluate the effectiveness of our maximum fraud trajectory construction algorithm. More specifically, given a fraudulent trajectory $l' \subset l$ and the corresponding maximum fraud trajectory l'' constructed by our algorithm, the *coverage ratio* of l'' is defined as:

$$recall = \frac{|l' \bigcap l''|}{|l'|} \tag{11}$$

where $|l' \bigcap l''|$ is the number of unit segments contained in the intersection sub-trajectory of l' and l''.

7.2 Experiment Results

In our experiments, we use the first thirteen days of taxi tracking records and taximeter records as training set, and the rest as test set. We partition the training set using $n_p = 10$. In the occupancy detection phase, the features we use includes average speed, arc-chord, curvature, and taxi cheating frequency. As explained in Section 1, existing approaches are not suitable for detecting unmetered taxi rides, therefore the prediction models we use for comparison includes logistic regression (LR), stochastic gradient descent (SGD), decision tree (DT), and multi-layer perceptron (MLP). The default values of each parameter described in Section 5 and 6 are: $w = 1$, $\Delta_t = 30$ min, $\lambda = 0.5$, and $\gamma = 0.5$.

In order to show the effectiveness of our integrated predictor compared to the baseline prediction models under different granularity of time interval for feature flattening, we conduct 6 groups of experiments using our integrated predictor and each baseline prediction model by changing Δ_t from 30 minutes to 3 hours.Many common methods which we conduct to solve fraudulent driving problem such as decision tree,logistic regression,stochastic gradient descent,multi-layer perceptron both underperform the method proposed in this paper.As describe in the above figure,when time interval is 30 minute,our methods are better obviously than other common algorithms,particularly the integrated method and Curvature metric method archieve best affect. The results are shown in Figure 5(a). It is clear that our spatiotemporal data-based predictor always achieves a much higher positive predictive value than the baseline prediction models under different Δ_t, which also shows the consistency of our spatiotemporal data-based predictor. In particular, the positive predictive value of our integrated predictor is up to 85% when Δ_t is 30 minutes, whereas some of the baseline prediction models result in less than 30%, which is not acceptable. Moreover, the positive predictive value of occupancy detection decreases when the time Δ_t enlarges for all the prediction models. This means that a smaller granularity of time interval for feature flattening results in a better performance, which shows the importance of time as a feature introduced in Section 5.2.

In fact,in order to validate our precision in this experiment based on the formula which was described as the number correctly detected as occupy devieded by all of the occupy trajectory,we need a lots of and enough trajectory to proof whatever trajectory is our detector can effciently detect it as a correct result.Here,we propose two excellent method to detected our judge in trajectory.The first judging methods is based on the Arc-chord.which is described as a amount of the real Euclidean distance devided by straight distance between the map-matching startpoint and map-matching endpoint.according to the statement above,if a taxi driver begin his business or he drives his taxi on a highway where there are less people call a taxi,then there are cunstomers with high possible in taxi and the state of the taxi is occupy.When this situation occur,we use the Arc-chord method to detect whether it is a cheating trajectory or detour. obviously,a normal drive behaviour will pass on shortcut path as more as possible,thus this trajectory includes shortcut path will has a similar distance with straight distance between two point and it is worth believe that the tracjectory is a tour with a passenger.Whereas if a taxi driver drives his car aimlessly like a chicken with its head cut off,then its Euclidean distance will bigger than the straight distance.Based on this approach,we can clearly devide all trajectories into occupy or unoccupy.The The Arc-chord detect algorithm is described as follow:

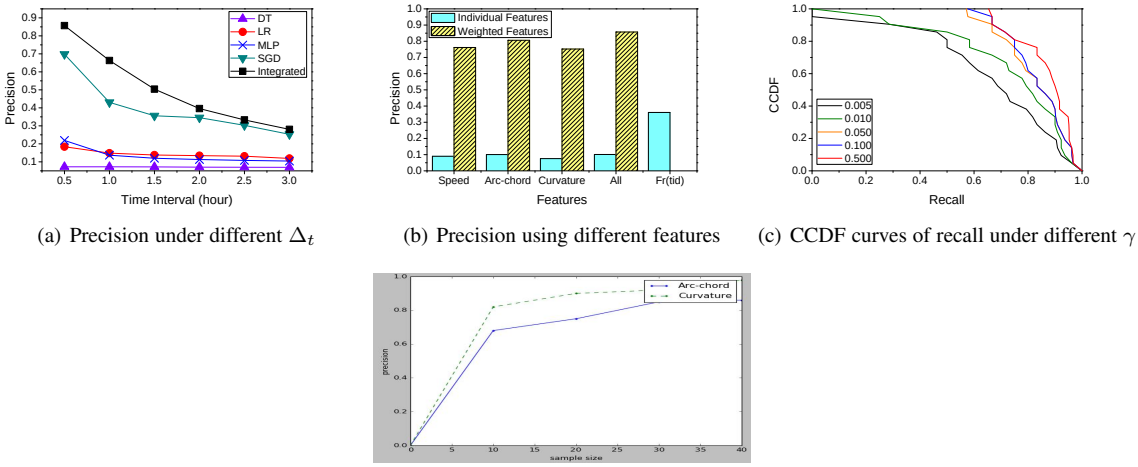
Algorithm 2 Arc-chord detect algorithm

Require: The sequence of unit segments $U = (u_1, \cdots, u_n)$; the set of occupied unit segments $U_t = \{u_{i_1}, u_{i_2}, \cdots, u_{i_k}\}$ where $1 \leq i_1 < i_2 < \cdots < i_k \leq n$;

for $j = 1; j \leq U_{t(}Length); j++$ **do**
  if $Current_{(}ID) != Pre_{(}ID)$ **then**
    if $Pre_{(}ID) != None$ **then**
      if $index(U_{tj}) = 0$ **then**
        $S_point = S_point + U_{tj}$
      else
        $Pre_X = Current_X; Pre_Y = Current_Y$
      end if
    end if
  else
    $S_n = Distance(Pre_X, Pre_Y, U_{tj})$
    $W_{(}dis) = W_{(}dis) + S_n$
    if $index(U_{tj}) = -1$ **then**
      $E_point = E_point + U_{tj}$
      $S_{(}dis) = Distance(U_{tj})$
    end if
  end if
end for
$with$
return $\frac{W_{(}DIS)}{S_{(}DIS)}$

In the above algorithm,PRE_LINE is a value where is used to store previous trajectory ID,Thus we can calculate a tracjectory according to its and previous ID continous data when we know it is a new trajectory.Because we need to calculate the straight distance between startpoint and endpoint.So when get the first point or latest point of a new trajectory then we store it,and $W_{(}DIS)$ is a value where save the cumulative Euclidean distance.Given the data above,it will return a Arc-chord value.

Another method which could efficiently infer a trajectory occupy or unoccupy is the Curvature method.As we know.In so many situation ,Whatever the taxi driver will go through various path,this path is less possible a geometric straight line,thus there must be some corner in the path,although they are different from each other.we can imagine that if a trajectory make so many corners,then this trajectory must be not a short distance. In fact,corner is the degree according to the current travel direction and next travel direction.For example,if there is a trajectory like the 6,the imaginary line is the straight line on destination's direction and it;s real trajectory make several corners.Before making a turn ,its travel direction is similar with the dotted line direction,this



(a) Precision under different Δ_t (b) Precision using different features (c) CCDF curves of recall under different γ

(d) method precision

Fig. 5: Effectiveness of integrated predictor and construction algorithm.

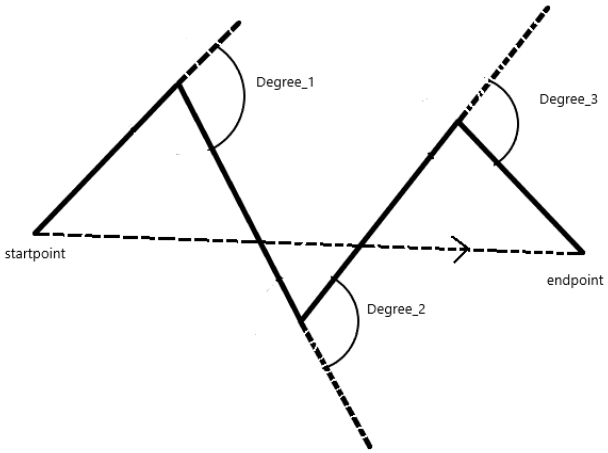


Fig. 6: A example of application of Curvature

direction will make a corner because of the subsequent various direction,just like $Degree_1$ in the 6 . And our Curvature is to add up all of these corners and average them up.By the way,if the average value is big enough,so it is more possible to detect it as a abnormal trajectory.Using this method,we can also distinctly devide all trajectories into occupy or unoccupy.The construct of Curvature algorithm is described as follow:

Because this algorithm is calculating data based on the direction instead of longitude value and lattitude value,thus first of all,we should calculate the direction of each straight line in whole trajectory.In consideration of a entire trajectory is compose of a serial of specific point and every near point-pair will constitute a straight line,so we use the directions of these line.$\{NEW_(Direction)\}$record each direction ,speak again that two near direction-pair generate a degree.Similarly, when we deal with a point ,we will store the current the longitude value and lattitude value into the specific variable Pre_X and Pre_Y.After each calculation of each corner degree,add each corner degree into the sum value of degree.Finally,we can figure out what the

Algorithm 3 Curvature detect algorithm

for $j = 1;j \leq U_t(Length);j++$ **do**
  if $Current_(ID)! = Pre_(ID)$ **then**
    $\{Direction_(SUM)\} = \{Direction_(SUM)\} + 1$
    if $U_t(j-1) not \in LINE_(end)$ **then**
      $LINE_(end) = LINE_(end) + \{Pre_x, Pre_y\}$
      $LINE_(straight) = LINE_(straight) + \{LINE_(start), LINE_(end)\}$
      $degree = Degree(LINE_(end), LINE_(straight))$
      $Degree_SUM = Degree_SUM + \{degree\}$
      for $j = 1;j \leq Degree_SUM;j++$ **do**
        $degree = Degree(LINE_(head), LINE_(straight))$
        $Degree_SUM = Degree_SUM + \{degree\}$
      end for
      $LINE_(head) = LINE_(head) + \{Pre_x, Pre_y\}$
    end if
  else
    $Direction_(SUM) = Direction_(SUM) + \{NEW_(Direction)\}$
    $Pre_X = CURRENT_X, Pre_Y = CURRENT_Y$
  end if
end for
return $\frac{Degree_SUM)}{Lengthof\{Direction_(SUM)\}}$

average is.If this average value is bigger than a threshold,then it is worth possible to believe that it is a abnormal driving behavior.

7.3 Synthetic trajectory

In consideration of we need a lots of trajectory to proof our detour detect algorithm could distinctly differentiate occupy or unoccupy,this section we introduce how to generate synthetic trajectory.According to we had propose above.we hope that some trajectories look like a normal driving behavior just like there is a passgener in it.and other trajectories look like some aimless path or it is in a state of looking for business.So based on these thought,it is feasible to generate normal synthetic trajectory by

using shortest path method.The shortest path method is often used in graph theory.As we widely know,an entire graph includes point sets and edge sets.When we want the shortest path from a specific startpoint to another specific endpoint ,this algorithm will traverse all weights of each edge and calculate a path with a smallest sum value of weight.In this problem situation,we use the distance of every uint road to be weight in our graph simulated from real world road data.

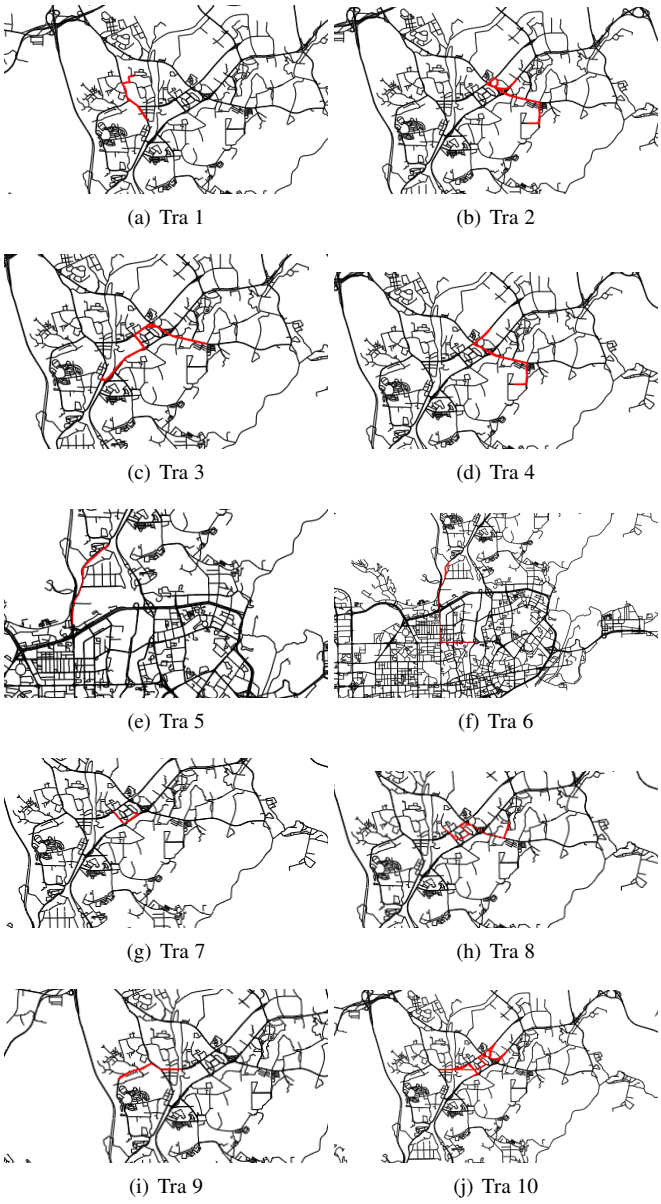
Compared to the shortest path method,random path generation will be more difficult.,it takes some existing paths .Up to now,we already have forty four data of cheating trajectory.,these trajctories include some normal path which looks like a shortcut path, and some path looks like abnormal path. It is important to noted that all of these real trajectories compose of the roads and intersections,among those roads,each road is made up of several points with coordinate,these points with coordinate make up a straight road which can be regard as a straight line.Thus those intersections is a point used to connect two road,although these roads have different directions. Therefore in every road,there must be a head point is same as the intersection or a end point is same as the intersection.In this special problem,it is feasible to regard each roads as those vertexs in the graph and regard intersections as edge used to join points.So,in the first step,we need to make up a logic graph based on road data and intersection data. Concrete algorithm is as follow:

Algorithm 4 Graph Construct algorithm

$\{V\} = \emptyset$;
$\{E\} = \emptyset$
for $j = 1; j \leq tracjetory_(Length); j++$ **do**
  if $j_(point) not \in \{Road_(point)\}$ **then**
    $\{E\} = \{E\} + j_(point)$
  else
    $\{V\} = \{V\} + j_(point)$
  end if
end for
for $j = 1; j \leq V_(Length); j++$ **do**
  for $i = 1; i \leq E_(Length); i++$ **do**
    if $j_(head) = i$ or $j_(end) = i$ **then**
      $\{E_i\} = \{E_i\} + V_j$
    end if
  end for
end for
$G = Graph(\{V\}, \{E\})$
return G

After constructing a graph,we can generate synthetic trajectory.First step.pick any number of points with any probability that is less than the number of the intersection .Second step,Traverse the graph $\{G\}$,find out those road which is connected by these intersection.By the way,when we generate the synthetic path based on random path algorithm.Becuase it may extract out some intersections which compose a approximate straight line,so some of the synthetic path will similar with the shortest path.Here are some generated synthetic trajectory:

We have generated a hundred of these paths and have figure out their Arc-chord value and Curvature value.According to statistical findings,those abnormal trajectory attribute a bigger Arc-chord value .In fact,if a synthetic trajectory is a taxi driving anywhere that we can easily utilize our eyes to distinguish them,its trajectory must be attributed a very large Arc-chord value and a very



(a) Tra 1 (b) Tra 2 (c) Tra 3 (d) Tra 4 (e) Tra 5 (f) Tra 6 (g) Tra 7 (h) Tra 8 (i) Tra 9 (j) Tra 10

Fig. 7: synthetic trajectory in normal path and abnormal path

large Curvature value.For example,in the above picture,the figure 7(e),obviously this taxi follow the best route to get passengers to their destination quickly,no detour path in it ,and its Arc-chord values 7.5,Curvature values 46.whereas,in figure 7(j).this taxi have been traveled almost all the area or roads in this area.its driving distance must be not small because it is spinning around the streets.After the calculation by our algorithm,no matter Arc-chord value or Curvature value,both values a enormous value,Arc-chord values 67,Curvature values 97.

In conclusion,by generating normal trajectory and abnormal trajectory which we can readily identify with our eyes.on this premise,we use proposed alogrithm above to validate them and find that our method,no matter Arc-chord method or Curvature method,both make a decent result and these results turned out to be very close to what we had originally envisioned.In this way, it can be proved that these two methods can infer all the paths as occupy or unoccupy with high accuracy,So this is further proof

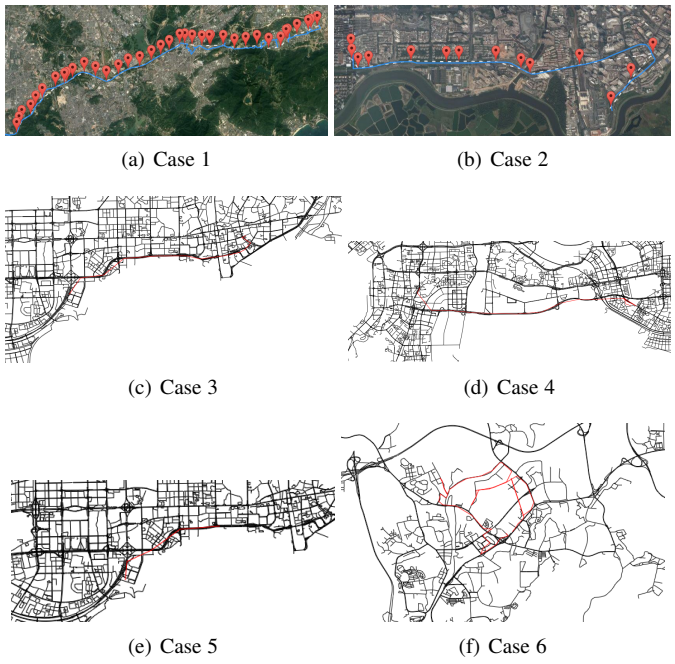
each cheating trajectory is high possibility to be concluded that there is passenger in car but the taxi driver obtains the extra money by dishonest tricks.

Next, we further analyze the performance of our selected features for occupancy detection. We conduct experiments on our integrated predictor by utilizing: 1) each spatial-temporal feature individually, compared with all features together; and 2) each spatial-temporal feature multiplied by taxi cheating frequency as a weight (referred to as transformed features), compared with taxi cheating frequency as an individual feature.Each individual spatial-temporal feature has different metric strategy.As for measure index of Arc-chord,it is used to calculate the ratio of taxi's driving distance on Euclidean distance between two positive local points.This measuring method means that if a trajectory's ratio is big enough,then we could suspect this taxi driver on this trajectory may be take his customer on a detour,because if a taxi has its business,its driving distance should be as similar as possible to the Euclidean distance.As for measure index of Curvature,we could make a conclusion that it is worth to believe a trajectory is cheating detour which exists so many transition and corner between two different unit road segment.i All the features are flattened by time. The results are shown in Figure 5(b). Consistent with the analysis in Section 5.2, the performance of spatial-temporal features without considering taxi cheating frequency is not satisfying. Meanwhile, it is not quite effective to use taxi cheating frequency as an independent feature either. However, the positive predictive values of occupancy detection rise significantly if we treat taxi cheating frequency as a weight by multiplying it with each spatial-temporal feature. Finally, our integrated predictor achieves better performance by taking all the features into modeling, compared to only considering each individual one.

In order to evaluate the effectiveness of our trajectory construction algorithm as introduced in Section 6.2, we conduct 5 groups of experiments by changing γ from 0.005 to 0.5. For the fraudulent trajectory set constructed by each group of experiments, we use the complementary cumulative probability function (CCDF) curve to evaluate its coverage ratio distribution. The results are shown in Figure 5(c). In general, the CCDF curve of the coverage ratio distribution leans towards 100% when γ increases. After γ reaches 0.5, the trajectory construction algorithm achieves the best performance, and the CCDF curve stays unchanged. Moreover, the results show that 81% of the trajectories constructed by our algorithm achieve over 75% coverage ratio, which indicates the effectiveness of our algorithm.

7.4 Case Study

In this paper,we manually generated a group of virtual simulation trajectory for inspecting our screening method in cheating trajectory,some of the routes are the same as they normally are, and that's exactly what customers want,and the rest of these trajectory are so unusual that they don't even look like the path a taxi would take to pick up a passenger,but It's like wandering around a certain area looking for customers, thinking the driver hasn't received any business yet.In the experiment,we use generated trajectory to verify the accuracy of the path proportion method and deflection Angle method proposed above in judging whether the trajectory is detour.Considering that we used the integration method mentioned in this article to compare the accuracy with other baselines,beacuse our critical criterion for precision is based on a taxi record of occupy of not.Thus, there must be a discriminator can efficientivelly detect a trajectory is occupy or not.It is



(a) Case 1 (b) Case 2

(c) Case 3 (d) Case 4

(e) Case 5 (f) Case 6

Fig. 8: Examples illustrating the effectiveness of our taxi fraud detection system.

reasonable to assume that if our discriminator can keenly identify the normal and abnormal driving behavior of a taxi,then we can easily label a trajectory of real data set with occupy or not.,and our integration method can correspondingly leverage the presicion.we conduct several case studies to exhibit the effectiveness of our taxi fraud detection system. Figure 8 illustrates several identical examples of the anomalous trajectories detected by our system. In Figure 8(a), the taxi is driving from the high-speed railway station to a suburban residential area. In Figure 8(b), the taxi is driving from the port of entry to an urban office area. Based on our investigations, these trajectories are highly suspicious to be unmetered taxi rides because of three reasons. First, these departure areas are important traffic terminals. For most of the time, there are crowds of people waiting in line to hire taxis, and regular taxis also have to wait in line to pick up passengers. Due to the inconvenience caused by congestion, it is not likely for a vacant taxi to travel into these areas. Second, there are huge demands of taxis in these departure areas, especially for those citizens traveling between urban and suburban areas every day.Since the profit of picking up passengers in these areas are quite high, it is not likely for a taxi to leave for urban or suburban areas without carrying passengers. Third, these trajectories mainly travel through expressways or main roads. It is efficient and fast to choose these routes for an occupied taxi to deliver passengers, but not effective for a vacant taxi to hunt for passengers. Hence, it is not likely for a vacant taxi to travel along these trajectories. In conclusion, these examples demonstrate that our system is effective in finding unmetered taxi rides with convincing evidences.Let's look at a few more examples where it can illustrated by proportion and deflection. In Figure 8(c),this taxi driver driver from a residential area to a business area ,both the path proportion method and the deflection Angle method have high confidence that this trajectory is a normal passenger trajectory,in fact,if the taxi driver have no customer,he would not driver into a high speed way.In the

same way,in Figure 8(d).It is very possible to conclude that there is a college student leave from a university for take a exercise because the destination of trajectory is a gymnasium in other city partition.It is less possible for a taxi driver to drive a distance for searching business,By the contrary,in Figure 8(f), this is an absolutely abnormal path.first,the distance between its starting point and destination is not very long,but its entire driving distance is 80 times bigger than its straight distance.Second,There are so many corner in this trajectory and if it is a normal path its direction will be almost similiar with the direction between starting point and destination.Therefore,we have a high confidence coefficient to detect it as an unoccupy trajectory.

8 Conclusion

In this paper, we study a new type of taxi fraud called **unmetered taxi fraud**, where taxi drivers carry passengers without activating the taximeters. Existing approaches on taxi fraud detection tries to find anomalous trajectories from occupied ones recorded by taximeters or seat sensors. However, these approaches are not suitable for our problem because: 1) unmetered fraudulent trips are not recorded by taximeters, and 2) the occupancy information collected from seat sensors on taxis is not precise. Therefore, we propose a novel taxi fraud detection system specifically for unmetered taxi frauds. The basic idea of our approach is to find anomalous trajectories that are not recorded by the taximeter but have driving behaviors similar to regular occupied trips. We first propose a learning model to capture the different driving behaviors of taxis when they are occupied or vacant, and then implement a heuristic algorithm to construct unmetered fraudulent trajectories by utilizing trajectory dataset and taximeter records. We conduct intensive experiments on real trajectory data. The results show that our proposed system achieves a satisfactory performance. In this paper, we haven't considered the characteristics of the road networks and the influence of POIs, which will be studied in our future work.

Acknowledgments

This work is supported in part by the National Key Basic Research and Development Program of China (973) Grant 2014CB340304, Macao FDCT Grant 149/2016/A and the University of Macau Grant SRG2015-00050-FST.

References

[1] X. Zhou, Y. Ding, F. Peng, Q. Luo, and L. M. Ni, "Detecting unmetered taxi rides from trajectory data," in *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 2017, pp. 530–535.

[2] D. Zhang, N. Li, Z.-H. Zhou, C. Chen, L. Sun, and S. Li, "ibat: detecting anomalous taxi trajectories from gps traces," in *Proceedings of the 13th international conference on Ubiquitous computing*. ACM, 2011, pp. 99–108.

[3] S. Liu, L. M. Ni, and R. Krishnan, "Fraud detection from taxis' driving behaviors," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 1, pp. 464–472, 2014.

[4] S. Zhang and Z. Wang, "Inferring passenger denial behavior of taxi drivers from large-scale taxi traces," *PloS one*, vol. 11, no. 11, p. e0165597, 2016.

[5] Increasing the taxi fare is a dead end. [Online]. Available: https://hk.lifestyle.appledaily.com/nextplus/magazine/article/20180919/2_624589_0/

[6] 146 taxi drivers being caught by police in guangzhou for illegally refusing or arbitrarily pricing passengers. [Online]. Available: http://www.gzcankao.com/news/wx/detail?newsi=39885

[7] J.-G. Lee, J. Han, and X. Li, "Trajectory outlier detection: A partition-and-detect framework," in *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*. IEEE, 2008, pp. 140–149.

[8] Y. Bu, L. Chen, A. W.-C. Fu, and D. Liu, "Efficient anomaly monitoring over moving object trajectory streams," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 159–168.

[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 *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, 2010, pp. 1733–1736.

[10] C. Chen, D. Zhang, P. S. Castro, N. Li, L. Sun, and S. Li, "Real-time detection of anomalous taxi trajectories from gps traces," in *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Springer, 2011, pp. 63–74.

[11] Z. Lv, J. Xu, P. Zhao, G. Liu, L. Zhao, and X. Zhou, "Outlier trajectory detection: A trajectory analytics based approach," in *International Conference on Database Systems for Advanced Applications*. Springer, 2017, pp. 231–246.

[12] X. Li, Z. Li, J. Han, and J.-G. Lee, "Temporal outlier detection in vehicle traffic data," in *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*. IEEE, 2009, pp. 1319–1322.

[13] J. Zhu, W. Jiang, A. Liu, G. Liu, and L. Zhao, "Time-dependent popular routes based trajectory outlier detection," in *International Conference on Web Information Systems Engineering*. Springer, 2015, pp. 16–30.

[14] Y. Ge, H. Xiong, C. Liu, and Z.-H. Zhou, "A taxi driving fraud detection system," in *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, 2011, pp. 181–190.

[15] X. Zhou, Y. Ding, H. Tan, Q. Luo, and L. M. Ni, "Himm: An hmm-based interactive map-matching system," *Lecture Notes in Computer Science*, p. 3, 2017.

[16] Y. Ding, S. Liu, J. Pu, and L. M. Ni, "Hunts: A trajectory recommendation system for effective and efficient hunting of taxi passengers," in *Mobile Data Management (MDM), 2013 IEEE 14th International Conference on*, vol. 1. IEEE, 2013, pp. 107–116.

Qing Liao received her Ph.D. degree in computer science and engineering in 2016 supervised by Prof. Qian Zhang from the Department of Computer Science and Engineering of the Hong Kong University of Science and Technology. She is currently an assistant professor with School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), China. Her research interests include big data analytics and artificial intelligence.

Xibo Zhou Xibo Zhou recieved his Ph.D. degree in computer science and engineering in 2018 supervised by Prof. Lionel M. Ni and Prof. Qiong Luo from the Department of Computer Science and Engineering of the Hong Kong University of Science and Technology. His research interests include spatial-temporal data mining and distributed systems.

Ye Ding received his Ph.D. degree in computer science and engineering in 2014 supervised by Prof. Lionel M. Ni from the Department of Computer Science and Engineering of The Hong Kong University of Science and Technology. He is currently an associate professor in School of Software Engineering and Cyperspace Science at Dongguan University of Technology. His research interests are spatial-temporal data analytics, big data, and machine learning. He is a member of IEEE since 2013.

Fengchao Peng ...

Qiong Luo ...

Lionel M. Ni is Chair Professor in the Department of Computer and Information Science and Vice Rector for Academic Affairs at the University of Macau. A fellow of IEEE and Hong Kong Academy of Engineering Science, Dr. Ni has chaired over 30 professional conferences and has received eight awards for authoring outstanding papers. He serves on the editorial boards of Communications of the ACM, IEEE Transactions on Big Data and ACM Transactions on Sensor Networks.