2017 IEEE 37th International Conference on Distributed Computing Systems

Speed-based Location Tracking in Usage-based Automotive Insurance

Lu Zhou, Qingrong Chen, Zutian Luo, Haojin Zhu[1], Cailian Chen

SHANGHAI JIAO TONG UNIVERSITY
{zl19920928, chenqingrong, eggstays_lzt, zhu-hj, cailianchen}@sjtu.edu.cn

***Abstract*—Usage-based Insurance (UBI) is regarded as a promising way to offer more accurate insurance premium by profiling driving behaviors. Compared with traditional insurance which considers drivers' history of accidents, traffic violations and etc, UBI focuses on driving data and can give a more reasonable insurance premium based on the current driving behaviors. Insurers use sensors in smartphone or vehicle to collect driving data (e.g. mileage, speed, hark braking) and compute a risk score based on these data to recalculate insurance premium. Many insurance programs, which are advertised as being privacy-preserving, do not directly use the GPS-based tracking, but it is not enough to protect driver's location privacy.**

In real world, many environment factors such as real-time traffic and traffic regulations can influence driving speed. These factors provide the side-channel information about the driving route, which can be exploited to infer the vehicle's trace. Based on the observation, we propose a novel speed based trajectory inference algorithm which can track drivers only with the speed data and original location. We implement the attack on a public dataset in New Jersey. The evaluation results show that the attacker can recover the route with a high successful rate.

Keywords – Vehicular Network, Usage-based Insurance, Location Tracking, Dynamic Time Warping

I. Introduction

The current pricing policy of automotive insurance companies around the world is based on the traditional factors, such as age, location of residence, history of accidents and traffic violations. This means that all customers would pay similar prices, despite potentially large variations in their driving habits. The emerging telematics-based Usage-based Insurance (or pay-how-you-drive programs) is dramatically reshaping the landscape of the global auto insurance market. Examples of such programs in North America and Europe include Progressive's Snapshot, AllState's Drivewise, State Farm's In-Drive, National General Insurance's Low-Mileage Discount, Travelers' Intellidrive, and others.

Usage-based Insurance (UBI) relies on the collection of the drivers' data (speed, mileage, harsh braking/accelerating, cornering and etc) based on different technologies (OBD-II, smartphone, or Hybrid OBD-Smartphone) during a monitoring period, which can reflect the probability of getting involved in a traffic accident. UBI provides a feasible way to differentiate the less risky drivers from the risky ones, which forms the basis for a risk categorization and thus for subsequent discounts or surcharges on the premiums dependent of the driving behavior. UBI market was at the starting point of 4.5 million subscribers in 2013. This number is expected to be around 100 million by 2020 and is projected to grow to approximately 50% of the worlds vehicles by 2030 [1].

Though UBI is regarded as a promising approach to offer more accurate insurance services by profiling driving habits, the data collected in this method can leak users' privacy, especially on the aspect of users' location privacy. Many insurance programs, which are advertised as being privacy-preserving, record the speed rather than directly using the GPS-based tracking. Some of the previous works [2] [3] challenge this best practice adopted by the industry by proposing a tracking algorithm merely based on the collected speed data. Unfortunately, the proposed algorithms suffer from limited tracking performance.

In this study, we propose a novel speed based trajectory inference algorithm based on the following observations: along with the development of various location based services (LBSs), it is easy to automatically retrieve the road speed limit (e.g., Speed Limits of Non-Curve and Curve roads) and real-time traffic from the public available interface provided by the mainstream navigation systems such as Google or Baidu Map. These speed limits and real-time traffic information provide us with important hints for actual driving speed of the target vehicle on a specific road, which can be exploited by an attacker to infer the target vehicle's real trajectory merely based on the speed data. We perform the comprehensive experiments to validate the proposed attack, which shows a significant improvement on the inference performance.

The contributions of this paper are summarized as follows:

- We perform a comprehensive survey on the current UBI system and propose a general risk calculation model based on the existing industry practice and prior works.
- We identify an attack which can infer a driver's trace even if insurance companies only know the speed data and the original location. Attackers can successfully launch the attack by exploiting the public available navigation information, such as the speed limits and real time traffic.
- We perform comprehensive experiments on the real world dataset to validate the discovered attack.

The rest of this paper is organized as follows. We briefly introduce UBI in Section II. Section III presents the attack. We discuss the experimental evaluation in Section IV. We conclude the paper in Section VI.

[1]Haojin Zhu is the corresponding author

1063-6927/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDCS.2017.278

TABLE I
DATA COLLECTED FROM SOME MAINSTREAM INSURERS

Data / Company	Hard Braking	Mileage	Time of Day	Speed	Acceleration	Turn
StateFarm	√	√	√	√	√	√
Progressive	√	√	√	√		
Allstate	√	√	√	√		
Esurance	√		√	√	√	
ZhongHua Insurance	√	√		√	√	

II. OVERVIEW OF USAGE-BASED INSURANCE

UBI is a telematics-based insurance service [4] where premiums are based on driving behavior of consumers. Different from the traditional insurance in which premiums are a reflection of the drivers' driving history, in UBI, the insurance premium is calculated dynamically not just based on how much you drive but how and when one drives.

Usage-based insurance relies on the following technologies to collect the data related to the driver's driving habits.

- *OBD-II*: Drivers plug a device into the vehicle's diagnostic port. It captures mileage, speed, braking and other vehicle measurements.
- *Smartphone*: In smartphone-based method, the installed application uses sensors to collect the metrics as OBD. A smartphone program like Drivewise can provide rating factors that are accurate enough for insurance premiums and cost $50 - 75\%$ less than an OBD program.
- *Hybrid OBD-Smartphone*: This approach combines smartphone and OBD based approaches.

We have surveyed the major UBI players in US and China markets. Their interested data is summarized in Table I.

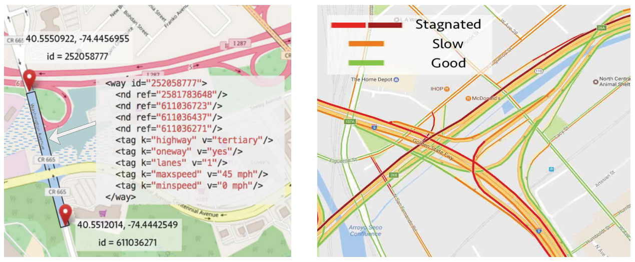
According to Table.I, although the data collected by different companies are slightly different, some factors such as speed (per second), hard braking, mileage, time of day are widely adopted by the insurers for insurance calculation. In practice, drivers do not need to submit the data in real time. According to Allstate, the data can be uploaded at the end of each trip when the smart phone gains access to the internet.

Although some of the UBI are based on the collected GPS information without considering any privacy issue, many other UBI programs are advertised as being privacy-preserving. Some of the insurance companies only record speed information, mileage and hard braking, and claim that they do not intend to collect the user's location. There are some existing researches [2] [3] that point out that it may be possible to infer the driving routines merely with the driving speed. However, the existing researches suffer from a low successfully rate. In the next section, we will present a novel routine inference scheme, which can significantly improve the inference accuracy.

III. THE DISCOVERED NOVEL TRAJECTORY INFERENCE ATTACK BASED ON THE DRIVING SPEED

A. Attack Overview

In this study, the insurer is assumed to be curious-but-honest and he is interested in collecting the driver's location information. However, the insurer does not have the incentive to modify the driver's driving data. Therefore, we use the same assumption proposed in [2] [3]: insurers only know the start location and speed data, and the curious insurers aim to track the target drivers merely based on these data.



```
<way id="252058777">
  <nd ref="2581783648"/>
  <nd ref="611036723"/>
  <nd ref="611036437"/>
  <nd ref="611036271"/>
  <tag k="highway" v="tertiary"/>
  <tag k="oneway" v="yes"/>
  <tag k="lanes" v="1"/>
  <tag k="maxspeed" v="45 mph"/>
  <tag k="minspeed" v="0 mph"/>
</way>
```

(a) Speed Limit From OSM (b) RRT Data From Google

Fig. 1. Real World Data

The basic idea of the proposed attack is based on the following insight: the driving speed is influenced by many environment factors such as the road condition, real-time traffic, and even traffic regulations. From the attacker's perspective, these environment factors as well as the real-time speed provide the side-channel information about the driving routes, which can be exploited to filter out those impossible routes and determine the most likely candidate routes. We summarize these environment information as follows:

- *Advisory Speed Limit for Non-Curve Roads:* In most of the roads (e.g., in city centers and outside schools), the vehicle should follow advisory speed limit v_{max} recommended by government. Even if the vehicle exceeds the speed limit, the exceeding proportion cannot exceed a certain threshold σ. Otherwise, the driver will receive the speeding ticket. In general, advisory speed limit provides upper bound of the speed on a specific road and can be automatically extracted from the map shown in Fig. 1(a). Therefore, the advisory speed limit can be exploited by an attacker to infer the user's trajectory. For further discussion, we give the definition as follow: for a series of speeds $V = [v_1, v_2, \ldots, v_n]$ driving through a road, the percentage of speeding is $\sum_{v_i} \mathbb{I}(v_i)/|V|$, where $\mathbb{I}(v_i) = 1$ if $v_i > v_{max}$, otherwise $\mathbb{I}(v_i) = 0$.
- *Impact of Real-time Road Traffic (RRT):* In practice, the speed of the vehicle is highly influenced by the road traffic, especially during the rush hours. Many maps (e.g., google map, baidu map and etc) offer the API that can help display real-time traffic for the road, which facilitates the tracking attack launched by the attacker. As shown in Fig. 1(b), different colors represent different traffic status, including: "Good" (e.g., driving at v_{max}), "Slow" (e.g., $\frac{2}{3}v_{max}$), and "Stagnated"(e.g., $\frac{1}{3}v_{max}$). Some map systems even offer the theoretical driving time t_{query} for a specific road segment based on the current road traffic.
- *Speed Limit for Driving through a Curve:* When a vehicle approaches a curve of a radius r, the driver should slow down the vehicle by following the speed limit. If the

vehicle speed is higher than the limit of the vehicle speed, the vehicle speed control system provides a warning [5]. Specifically, vehicle's friction must be greater than centrifugal force to ensure the safety:

$$f_r \cdot m \cdot g > m \cdot r \cdot (\frac{v_b}{r})^2 \tag{1}$$

where f_r is the friction coefficient of the road, m, g refer to the weight of the vehicle and the gravitational acceleration respectively. Then we can get the maximum speed $v_b = \sqrt{f_r \cdot g \cdot r}$. Note that, the parameters of the road can be obtained from the map.

Based on the above observations, it is possible for the external attacker to rule out the less likely routes and calculate the most likely candidate routes. Our basic strategy shown in Fig. 2 is that we can build the speed model for different road conditions based on Advisory Speed Limit, Real-time Road Traffic, and Speed Limit for Driving a Curve. By comparing the difference between speed model and the collected speed, we can calculate the probability for each road segment by adopting Dynamic Time Warping (DTW) algorithm. The details of the proposed attack is presented in Section III-C.

B. Problem Formulation

It is assumed that the attacker knows the starting point of the trip and can exploit the public available information (e.g., road information from OpenStreetMap) to launch the attack to track the target driver. Let N be the number of the intersections that reside in the area. A directed graph $G = (V, E)$ can be used to represent all of the intersections. $V = \{1, 2, \cdots\}$ refers to the set of intersections and $e_{xy} \in E$ stands for a road between intersection x and intersection y. The basic goal of the attacker is to find out the route which fits the speed model best.

Then we formalize the problem of inferring users' trace as a Hidden Markov Model (HMM). We define the route of the vehicle as (Q, T):

$$Q = \{q_1, q_2, ...\}, q_i \in E \quad T = \{(t_0, t_1), (t_1, t_2), ...\} \tag{2}$$

where Q is the set of road segments and (t_{i-1}, t_i) are the start time and end time driving on the road q_i.

Formally, an HMM is characterized by the following:

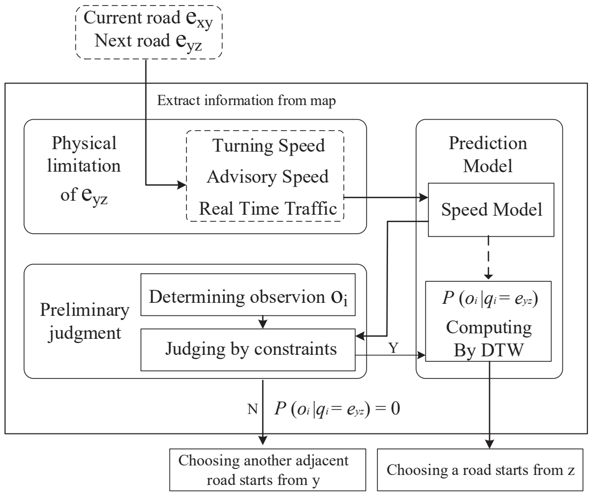
- The state transition probability distribution $A = \{a_{e_{xy} \to e_{mn}} | \forall e_{xy}, e_{mn} \in E\}$, where

$$a_{e_{xy} \to e_{mn}} = p(q_{i+1} = e_{mn} | q_i = e_{xy}) \tag{3}$$

  $a_{e_{xy} \to e_{mn}}$ means the probability of directly moving from road e_{xy} to road e_{mn}. We set $a_{e_{xy} \to e_{mn}} = 0$ if $y \neq m$ and set other probabilities to be uniformly distributed over all possible transitions.
- The observation symbol probability distribution B. In our model, observation is a series of speed values and corresponding timestamps. We denote the observation as $O = \{o_1, o_2, \ldots\}$. Then B can be denoted as

$$B = \{b_{e_{xy}}(o_k) = p(o_k | q_t = e_{xy})\} \tag{4}$$

  where $b_{e_{xy}}(o_k)$ is the probability of generating the given speed values o_k while the vehicle drives through the road segment e_{xy}. In this work, this probability can be calculated by comparing the distance between the collected speed values and built speed model by running the DTW algorithm.
- We define $\pi = \{\pi_{e_{xy}}\}$ as the initial state distribution, where $\pi_{e_{xy}}$ is the probability that the vehicle initially goes through the road segment e_{xy}. Since the starting location (at a intersection) is known, we set obtain $\pi_{e_{on}}$ to $1/k$. Here, e_{on} refers to the neighboring road segments of original location and k refers to the total number of neighboring segments. Others is set to 0.



Current road e_{xy}
Next road e_{yz}
Extract information from map
Physical limitation of e_{yz}
Turning Speed
Advisory Speed
Real Time Traffic
Prediction Model
Speed Model
Preliminary judgment
Determining observion o_i
Judging by constraints
Y
$P(o_i|q_i = e_{yz})$ Computing By DTW
N
$P(o_i|q_i = e_{yz}) = 0$
Choosing another adjacent road starts from y
Choosing a road starts from z

Fig. 2. Overview of Our Attack

We can define our problem as a classical HMM problem. Given an observation O, and $\lambda = (\pi, A, B)$ which indicates the parameter set of the model, our goal is choosing a state sequence Q which is optimal to translate this observation. According to [6], this problem is equivalent to maximizing $P(Q, O|\lambda) = P(O|Q, \lambda) \cdot P(Q|\lambda)$. We assume the driver has the equal probability to go through these routes. Therefore, this problem can be interpreted as the problem of finding an optimal route Q such that $P(O|Q, \lambda)$ is maximized. We denote $P(O|Q, \lambda)$ as $P(O|Q)$ for ease of presentation.

C. Attack Algorithm

For a specific route which has m road segments, we split the observation into m sub-observations. Considering any two sub-observations o_i and o_j which are a series of discrete speed values, o_i and o_j are assumed to be mutually independent according to [7]. So based on the property of output independence assumption, we can split one route into many road segments and compute them iteratively:

$$\begin{aligned} p(O|Q) &= p(o_1, ..., o_m | q_1, \ldots, q_m) \\ &= \prod p(o_i | q_1, \ldots, q_m) \\ &= \prod p(o_i | q_i) \\ &= \prod b_{e_{xy}}(o_i) \end{aligned} \tag{5}$$

We can use the forward algorithm to calculate the probability of the state sequence in a specific HMM, and then find the most possible sequence. However the number of all possible routes in a large area may be too large for computation. In general, the complexity of this problem is $O(N^M)$, where N is the number of all roads and M is the possible number of road segments which vehicle traveled. So it is difficult to perform an exhaustive search in our problem. Fortunately, for the considered problem, it is possible to reduce the complexity by the following steps shown in Fig. 2:

- *Speed Model Checking*: In Section III-D, we introduce a speed model. At each iteration i, we will check whether o_i satisfies the constraints defined by the speed model of this road. If it does not match, $p(o_i|q_i)$ is set to 0 and this road segment is not considered in the following search.
- *Probability Calculation*: If a road segment passes the speed model checking, it means that it satisfies the different speed limits and constraints. We can further calculate $p(o_i|q_i)$ by DTW, which represents the probability of the vehicle traversing a given segment q_i.

Based on the above observations, we introduce a pruning-based depth first search (DFS) algorithm to address the target problem. We introduce a pruning heuristic DFS so that candidate routes can be computed efficiently. The basic idea is to remove the routes containing some road segments which are not reachable (or the probability $p(o_i|q_i) = 0$) during the algorithm running time. Through setting reasonable constraints and speed model, we can remove most routes and generate the candidate routes from H. Then we sort the candidate routes by the $P(O|Q)$ to get the top routes. Algorithm 1 describes this logic as pseudocode.

```
Algorithm 1: Pruning-based DFS(node,timestamp)
  Input: original location $x$, currentTimestamp $t_0$
  Output: candidate road segments $H$
  $I_1 \leftarrow$ query_nextnodelist($x$) by state transition matrix A;
  for every node $y \in I_1$ do
      $d_i \leftarrow$ calculate_distance($e_{xy}$);
      $m_i \leftarrow$ generate_speed_model($e_{xy}$);
      $t_{start} \leftarrow$ currentTimestamp;
      $t_{end} \leftarrow$ determine_endtime($t_{start}, d_i$);
      $o_i \leftarrow$ observation($t_{start}, t_{end}$);
      if $v_{turn} > v_b \ || \sum_{v_i} \mathbb{I}(v_i)/|V| > \sigma \ ||$
        $(t_{end} - t_{start}) \notin [\eta_1 \cdot t_{query}, \eta_2 \cdot t_{query}]$ then
          $p(o_i|q_i) = 0$;
          choose another node from $I_1$;
      end
      else
          $d_i = DTW(o_i, m_i)$;
          $d_s = DTW(m_i - m_{threshold}, m_i)$;
          $d_0 = DTW(0, m_i)$;
          get $p(o_i|q_i)$ by Equation 6
          $H \leftarrow$ store($x, y, t_{start}, t_{end}, d_i, p(o_i|q_i)$);
          Pruning-based DFS $(y, t_{end})$
      end
      return $H$
  end
```

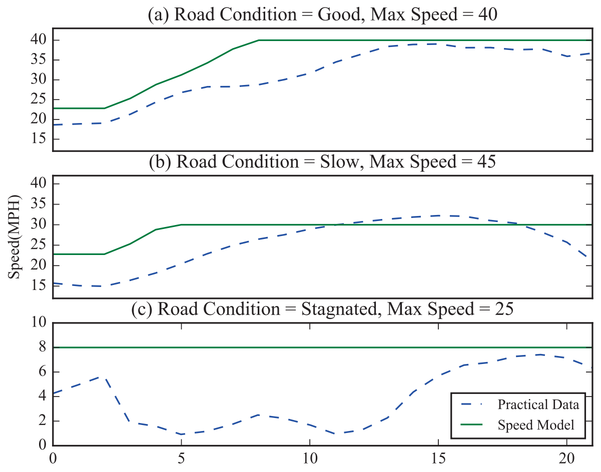


Fig. 3. Speed Model VS Practical Data

D. Building The Speed Model

The goal of speed model is to filter out the impossible routes, which fail to satisfy the speed limits introduced in section III-A. Speed model describes a theoretical maximum speed of a road segment at a specific time and it is defined by a series of continuous speed values. We use a method proposed in [2] and extend it by adding real time traffic to fit different situations. The basic model comes from the value of maximum speed for each road, which can be collected from OpenStreetMap (OSM) and Wikipedia. If a turning event occurs, we improve the speed model by adding turning speed limit which is calculated by the law of cosines according to the previous road segment.

Adding the real-time traffic information: It is obvious that the vehicle cannot reach its maximum speed limit of a specific road during the rush hour. Therefore, the real-time traffic is critical for improving the tracking precision. In practice, the real-time traffic has been provided by several navigation engines (e.g., google map or baidu map), which will be integrated into our speed model shown in Fig. 3.

E. Calculating The Probability of Possible Route by DTW

DTW algorithm [8] is an algorithm that can compute an optimal match between two given sequences which may have different lengths (e.g. time series) with certain restrictions. We adopt DTW algorithm to calculate the probability of the vehicle being in a specific route. In particular, given a specific road, the corresponding speed model and the observed speed are m_i and o_i, respectively. Then we compute the Euclidean distance d_i between the speed model and observation o_i, and the Euclidean distance d_0 between the speed model and zero. The latter indicates the maximum distance between the speed model and the possible vehicle speed. Further, we define the Euclidean distance d_s between the speed model and the speed model minus a threshold, which means we have the same possibility if driving speed within the scope. Based on the

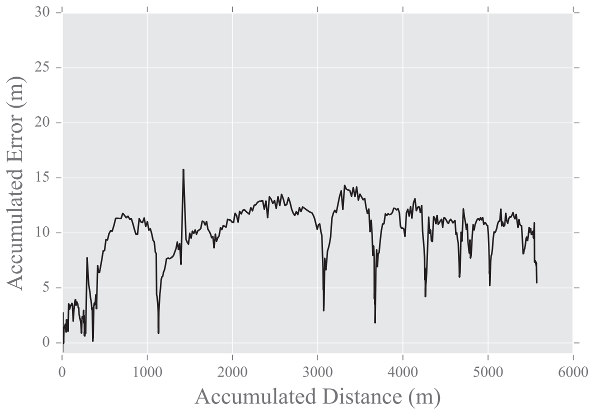


Fig. 4. Error between calculated distance and actual distance

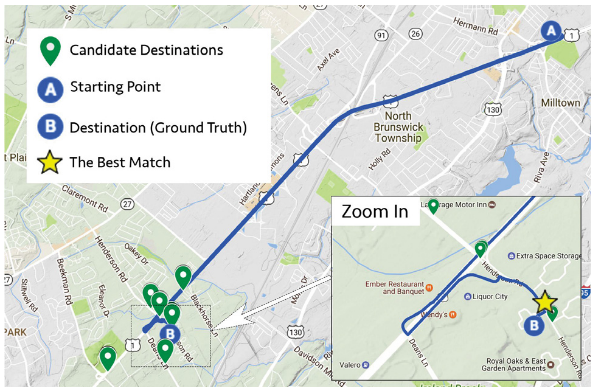


Fig. 5. Inferred Result of a Trip

above parameters, we can calculate the probability of a specific route being the driving route of the target vehicle as follows:

$$p(o_i|q_i) = \begin{cases} 1, & d_i \leq d_s \\ 1 - \frac{(d_i - d_s)}{(d_0 - d_s)}, & d_i > d_s \end{cases} \quad (6)$$

Based on the calculated probability, we can maintain a list of possible locations. Then we sort the candidate routes obtained from H and select the top routes.

F. Choosing Corresponding o_i with Road Segment q_i

We have already known the time t_{start} when vehicle enters into the road segment q_i, so to get the corresponding o_i, we should determine the end time t_{end} leaving q_i. Because we only have pairs of speed per second and timestamp, we use a method to approximately compute the driving distance per second: we assume the movement of every second is a uniformly accelerated (retarded) rectilinear motion. Fig.4 shows the error between the calculated distance and actual distance of a 5.7 km trip, which is very small compared with the whole distance.

According to the calculated distance per second and road distance, we can get the end time t_{end}. Then we can get corresponding speed values in time interval $[t_{start}, t_{end}]$, which is the corresponding o_i with the segment q_i.

IV. Implementation and Evaluation

In this section, we firstly implement our attack algorithm on a real world dataset and analyze the environmental factor which may affect the results.

A. Implementation of attack algorithm

We use a public dataset [3] in New Jersey, which contains the timestamped speed data and the ground truth of GPS data. We fetched the street information from OSM, including nodes, ways and relations. In addition, we can get the value of maximum speed for each road from OSM and Wikipedia.

As the real-time traffic is taken into consideration in our algorithm, we implement this part by calling Google Maps API. For Google Map only provides traffic conditions at present or in the future, we could not fetch it corresponding to the collection time of the dataset. However, the real-time traffic has a close relationship with historical data and often fluctuate within a small scope, so it is rational to request traffic conditions in the same time slots of the day as the traces.

In our experiments, we set $\sigma = 0.2$, $\eta_1 = 0.5$ and $\eta_2 = 2$. We define the route as *the best match* if the inferred route completely matches the real route.

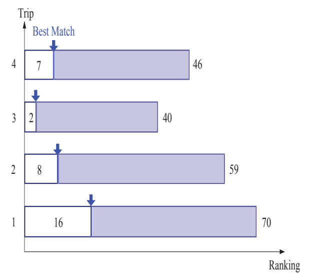
B. Experiment Results

We launch our attack with the data from 120 trips. Each trip varies in the range of $(7km, 21km)$ and average length is about $12km$. Fig. 5 shows the inferred candidate routes of a trip, and the best match is a inferred route which completely matches the real route. Based on our algorithm, we can figure out a few candidate routes (destinations of those routes are marked in green) from massive routes. We can see many of the candidate routes end as being around the destinations. The reason for this is that when the trip is nearly over, most users will slow down and at the same time speed limitations will not function properly.

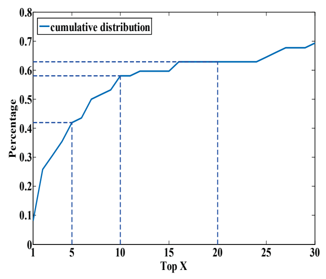
After getting the candidate routes, we will sort them by the probability of each route computed by DTW. This probability describes the degree of similarity between a candidate route and real route. We select four trips whose routes can be successfully recovered by our algorithm and show the ranking of the best match in the candidate routes. It can be seen from the sorting results in Fig. 6(a), the best match's ranking is always on the top of all candidates regardless of the number of candidates, which means DTW performs well in selecting the best match from the whole candidate routes.

Then we show the probability of the best match's ranking among the candidate routes in Fig 6(b). There is a probability of 70% in the top 30 candidate routes, which means the attacker has 70% probability to obtain the real route if he chooses the top 30 candidate routes of a trip. By using the side-channel information, it is easy to get the real route from the top 30 routes.

To illustrate the function of the real time traffic, we conduct a set of comparative experiments. Table II shows real time



(a) The Ranking of Best Match



(b) Best Match's ranking of all trips

Fig. 6. Evaluation Result of Our Attack

TABLE II
EFFECT OF RRT

Without RRT		With RRT	
rank/candidates	recursion times	rank/candidates	recursion times
4/23	719	3/12	664
3/24	545	2/22	501
16/70	460	8/31	309
2/40	315	1/30	154
5/16	566	4/11	452
9/33	2732	5/15	2195
3/32	358	3/22	220
4/9	507	4/6	286
26/136	877	20/106	675

traffic has a significant effect in reducing the recursive times and increasing the inferring accuracy. The more candidate routes, the greater the effect.

C. Analysis

In this section, we analyze the reason why candidate routes are numerous for some trips. We summarize the factors which may affect the inferring accuracy as follows: 1) Driving speed is very slow at the end of a trip. Under this situation, we cannot remove the incorrect path in time, which will cause too many possible routes to compute. 2) In some special areas, such as a residential area, road segments with respect to the speed limit and road conditions are very similar, so the real route is indistinguishable from other routes. 3) We cannot get the real time traffic at the time when the data was collected. So this factor will have slight effect on the inferring process. In practice, attacker has ability to run another server to collect the information at the same time when user drives, which can get more accurate results.

V. COUNTERMEASURES

One of the most straightforward Countermeasures is to collect these data and compute the risk score by a third party which is believed to be reliable. Such that the insurer can only obtain the aggregated result, which can assure the security of the individual's data. But it is difficult to find the trusted third party which can be recognized by both the user and the insurer, and the existence of the third party will incur other security problems.

A more practical countermeasure is sending all these data to the insurer after adding some noise. based on the formula proposed in [9], the insurer only needs to get the sum of the speed data which is greater than the dangerous speed data defined by the insurer. To ensure the computability of the data, we can add the special noise to these data which is greater than the dangerous speed data with the promise that the sum of these noise is zero, which is inspired by [10], [11]. Then we using a flag bit to indicate whether the speed data is greater than the dangerous speed, 0 for "no" and 1 for "yes". After sending to the insurer, insurer can compute the risk score without leaking individual's data. By hiding speed data into the noise, insurers cannot find the relationship between the speed and environment factors such as real-time traffic and even traffic regulations. They cannot identify the real route from massive routes, so our method can successfully thwart the location tracking attack.

VI. CONCLUSION

In this paper, we find that attackers can track drivers only by the speed data and the original location. By using the physical limitations of a road, attack can identify the possible routes from the massive routes. We demonstrate the correctness of our discovered attack and the efficiency of the proposed framework by the detailed experiments. Our further research will includes other security issues in UBI.

ACKNOWLEDGMENT

This work is supported by National Science Foundation of China (No. U1405251, 71671114, 61672350).

REFERENCES

[1] D. I. Tselentis, G. Yannis, and E. I. Vlahogianni, "Innovative motor insurance schemes: a review of current practices and emerging challenges," *Accident Analysis & Prevention*, vol. 98, pp. 139–148, 2017.
[2] R. Dewri, P. Annadata, W. Eltarjaman, and R. Thurimella, "Inferring trip destinations from driving habits data," in *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*. ACM, 2013, pp. 267–272.
[3] X. Gao, B. Firner, S. Sugrim, V. Kaiser-Pendergrast, Y. Yang, and J. Lindqvist, "Elastic pathing: Your speed is enough to track you," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2014, pp. 975–986.
[4] S. Husnjak, D. Peraković, I. Forenbacher, and M. Mumdziev, "Telematics system in usage based motor insurance," *Procedia Engineering*, vol. 100, pp. 816–825, 2015.
[5] S. C. Harris, "Non real time traffic system for a navigator," May 5 2015, uS Patent 9,026,358.
[6] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.
[7] Y. Michalevsky, A. Schulman, G. A. Veerapandian, D. Boneh, and G. Nakibly, "Powerspy: Location tracking using mobile device power analysis," in *24th USENIX Security Symposium (USENIX Security 15)*, 2015, pp. 785–800.
[8] M. Müller, "Dynamic time warping," *Information retrieval for music and motion*, pp. 69–84, 2007.
[9] L. Boquete, J. M. Rodríguez-Ascariz, R. Barea, J. Cantos, J. M. Miguel-Jiménez, and S. Ortega, "Data acquisition, analysis and transmission platform for a pay-as-you-drive system," *Sensors*, vol. 10, no. 6, pp. 5395–5408, 2010.
[10] Y. Li, L. Zhou, H. Zhu, and L. Sun, "Privacy-preserving location proof for securing large-scale database-driven cognitive radio networks," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 563–571, 2016.
[11] S. Li, H. Zhu, Z. Gao, X. Guan, K. Xing, and X. Shen, "Location privacy preservation in collaborative spectrum sensing," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 729–737.