D-TOUR : Detour-based point of interest detection in privacy-sensitive trajectories

Maja Schneider[1], Lukas Gehrke[2], Peter Christen[3], Erhard Rahm[4]

Abstract: Data collected through mobile sensors on private and commercial devices can give valuable insights into mobility patterns and facilitate applications such as urban planning or traffic forecasting. At the same time, such data can carry immense privacy risks for the data producers. Stop detection approaches can reveal a person's points of interest (POI) by clustering temporal and spatial features, uncovering private attributes such as home or work addresses. Privacy-preserving mechanisms aim at hiding these POIs, for example via speed smoothing approaches that are able to preserve high data utility. We show experimentally on two real-world data sets that trajectories can contain anomalies that are contained to a certain extent when smoothing a route and are not detected by state of the art stop detection algorithms. We propose a novel attack *D-TOUR* that reveals POIs based on deviations from the optimal route. Our experiments suggest that our proposed attack has similar performance on unprotected data but outperforms the baseline approach, especially when protection is higher and route features become more sparse.

Keywords: Location Privacy; Trajectory data; Inference attack; Points of interest

1 INTRODUCTION

The increasing use of location-based services (LBS) on personal mobile devices or tracking-devices in connected vehicles is generating a wealth of mobility data that is being used by businesses and governments to improve their services and workflows, among other things [BBA15]. In this context, data about movements, such as the trajectories of people and vehicles, are of particular interest. Such data, typically collected using Global Positioning System (GPS) sensors, can provide valuable insight into mobility behavior and enable traffic forecasts and analyses that facilitate, for example, urban planning or cab dispatching [Mo16]. Mobility data drives improved user experience by enhancing mobile apps through personalization or context awareness [LLL16], and offers economic advantages because such data can be monetized and used for targeted advertising.

On the other hand, mobility data are inherently privacy sensitive [Kr07]. For example, private information can be learned by identifying a person's points of interest (POI). Both private POIs, such as home or work addresses, and public POIs, such as stores or museums,

[1] Scads.AI, University of Leipzig, Germany mschneider@informatik.uni-leipzig.de
[2] Scads.AI, University of Leipzig, Germany lg58weky@studserv.uni-leipzig.de
[3] School of Computing, The Australian National University, Canberra, Australia peter.christen@anu.edu.au
[4] Scads.AI, University of Leipzig, Germany rahm@informatik.uni-leipzig.de

are informative for identifying information such as gender, age, education level, or marital status [Zh15b]. Commercially collected movement data, for example in the context of parcel or food deliveries, can also reveal private information about customers, such as the times they are at home. As a recent example, a data leak of a food delivery service could be linked with other data to identify the residence of secret service agents and other politically interesting persons.[5] For corporate customers, POIs can also leak details about business relationships that may be of interest to competitors. Even more can be learned about people's lives if location data can be linked to other sensitive data such as medical records [CRS20].

The privacy risk of trajectory data can be measured, for example by the success probability of attacks on private data, such as tracking or inference attacks, or POI-reconstruction [AS03, GKd11, Kr07, Sh11]. To identify POIs, approaches are generally used that build on temporal and spatial clustering, known as stop detection [HT04]. While this approach relies on the presence of point clusters, a protection against such an attack is to smooth temporal and spatial features [Pr15]. The resulting trajectories hide POIs but preserve to some extent the mobility patterns and thereby ensure data utility, which is a typical challenge for location privacy preserving mechanisms (LPPM) [ABN10].

In this paper we investigate the question of whether such smoothed trajectories still contain enough information for an adversary to uncover private information such as POIs. To this end, we test the attack success of a state of the art stop detection algorithm on smoothed trajectories. We then devise a new attack algorithm *D-TOUR* that builds on analyzing unusual behaviour, such as taking a detour. We argue that trajectory data contains anomalies that can be exploited to identify POIs, and show experimentally on two real-world data sets that these anomalies cannot be sufficiently removed with time- and space-based smoothing approaches. In our experiments, *D-TOUR* is detecting POIs in unprotected trajectories with almost similar performance as the baseline stop detection algorithm. In trajectories protected by the smoothing approach *Promesse* [Pr15], our attack outperforms the baseline, especially when protection is higher and trajectory information becomes more sparse.

2 RELATED WORK

The risk of disclosing a user's POIs and private attributes based on their movement patterns can be measured by the success of an attack on the trajectory, such as inference attacks that infer POI locations, also called stop detection [HT04]. To detect POI locations, most mechanisms work with temporal and spatial features. In a first step, potentially interesting locations are identified as so-called stay-points or stop-points which are then aggregated into stay-regions to infer a POI location. Ashbrook and Starner [AS03] define stay-points as those trajectory points that have a minimum temporal distance from their predecessor, assuming that the large time gap stems from lost GPS-signal in buildings. Stay-points are clustered with

[5] Philipp Bovermann, "Lieferdienst-Datenleck enttarnt russische Geheimdienstmitarbeiter". Online at: https://www.sueddeutsche.de/wirtschaft/datenleak-russland-lieferdienst-bellingcat-1.5561080, 2022

a variant of k-means. However, this approach suffers from false positives due to GPS signal loss in narrow urban areas and fails to identify outdoor locations. Patterson et al. [Pa03] and Liao et al. [Li07] take into account derived features and background knowledge about bus schedules, to learn transportation mode changes that indicate a POI location. This approach relies on such changes and might miss POIs, such as indoor stop locations.

Simple heuristic approaches were mentioned by Gambs et al. [GKd11] to detect POIs, for example choosing the last stop of a trajectory before midnight to identify the home address of a user. The authors reason that POIs can be detected not only by observing spatially correlated events, but also the absence of data in a certain time period which can hint to a POI or private activity. They detected POIs of taxi drivers, for example, by examining when the GPS sensor was not active. Similarly, it is possible to identify muslim cab drivers by comparing their pause timing with praying times.[6]

Several clustering approaches exist for POI location inference that are based on analyzing temporal and spatial features [GKd11, HT04, Ka05, Zh04]. Primault et al. [Pr14] use for their analyses a variation of the Density-Joinable cluster (DJ-Cluster) algorithm [Zh04]. Their adapted algorithm works by first detecting so-called stays, which are areas of a certain diameter in which a person stayed for a certain time period [HT04]. For this the algorithm considers a distance threshold that defines the maximal diameter of a stay area and a time threshold defining the minimum time duration that has to be spent in every stay. In a second step, the stays are grouped into clusters, taking into account a minimum number of stays in a cluster and a merge threshold that defines the maximum distance below which two different clusters are merged into one.

A variety of defenses have been developed against inference attacks. Gambs et al. [GKd11] develop and test the resilience of a clustering mechanism against sanitization measures, such as sampling, a protection algorithm which fuses multiple routes of a user into a single route, and perturbation, which modifies route points by adding spatial Gaussian noise. The authors find perturbation to have the highest defense capabilities, but at the same time a substantial decrease in utility. In general, an LPPM aims at keeping the utility of the data high for the purpose of further analyses.

As clustering algorithms rely on temporal or spatial clusters to work with, an efficient defense against inference attacks based on clustering is to remove points from these clusters, for example by using smoothing approaches. An example for this approach is *Promesse* [Pr15], which keeps distortion in protected trajectories low and utility high. However, the question remains, if it is still possible to infer POI locations based on distinctive characteristics of a trajectory, such as movement patterns, sparsity, or unexpected movement behaviour. In our study we investigate this question and present a novel inference attack that exploits unexpected behaviour in movements to detect POI locations.

[6] L. Franceschi-Bicchierai, “Redditor cracks anonymous data trove to pinpoint muslim cab drivers”. Online at: http://mashable.com/2015/01/28/redditor-muslim-cab-drivers, 2015.

3 FORMALIZATION

To model the investigated problem we rely on the formalization introduced by Shokri et al. [Sh11]. The authors describe the connection between a *mobile user* who produces privacy-sensitive trajectory data while moving, an *adversary* with certain capabilities and knowledge who aims at exploiting the observed data, and an *LPPM* to protect this data from the adversary. We now describe these three concepts.

Mobile user: We consider a user $m \in \mathcal{M}$ moving in a *region* $\mathcal{R}$ during an observed *time period* $\mathcal{T}$. The region consists of distinct locations (or points) $r \in \mathcal{R}$. The time period has a certain resolution, such as minutes, and is defined as a list of timestamps $[t^{start}, t^{start+1}, \ldots, t^{end}]$ that mark, for example, the start of such an interval. An *event* $e \in \mathcal{E}$ captures a single measurement of the user's location r at time $t \in \mathcal{T}$, and is described by a tuple $e = \langle m, r, t \rangle$. Successive events generated by one user form a *trajectory* (or route) $L = [e_1, e_2, \ldots, e_n]$. Such a trajectory can be generated, for example, when a moving person is tracked every few minutes on their cell phone while using a navigation app.

Adversary: We focus on offline privacy, which means that an adversary only looks at the full data set of trajectories. We do not consider online privacy, which investigates the privacy risk that exists at the time of data collection and transmission. The trajectories observed by the adversary may have been previously manipulated by a privacy preserving mechanism.

We assume the adversary knows the mobile user's transport mode (such as walking, car, subway) or is able to deduce it. Indeed, this information may not be coherent in one trajectory, for example when a user travels partly by foot and subway. The adversary also has knowledge of the optimal (e. g. fastest) route between arbitrary locations in region R, taking into account the known mode of transportation. To achieve this, the adversary queries navigation services that calculate the fastest road connection between two locations.

Location Privacy Preserving Mechanisms (LPPM): We consider *Promesse* [Pr15] as the privacy preserving mechanism, which smoothes temporal and spatial features in a trajectory in order to avoid the formation of point clusters of temporally and geographically close events. This algorithm was shown to provide a good trade-off between protection against stop detection attacks and preservation of data utility.

First, the algorithm regularly samples locations along the shape that is formed when successive points of the original route are connected. Successive sampled points have a distance of at least α, the so-called smoothing distance. Then, the first and last point of the route are removed to avoid leaking potentially private information, because trajectory endpoints are likely meaningful [GKd11]. Finally, the timestamps of the sampled points are set to be equally distributed between the original start and end timestamps.

4 DETOUR-BASED POI DETECTION

State of the art approaches in stop detection rely on the presence of stay-points, which are multiple points that are geographically or temporally clustered [HT04]. These clusters result from extended stays by a user in a particular area, which in turn may indicate a potentially privacy-sensitive POI. An intuitive approach to protecting POIs is to remove points from these clusters in order to hide stay-points and thus prevent inference attacks. At the same time, this should be done in a way that does not interfere too much with mobility patterns, so that the benefits are maintained and the data can still be used for mobility analyses.

However, an adversary can use additional knowledge about typical user movements to reveal anomalies in the trajectories that become visible when comparing privacy-sensitive trajectories and known patterns. Following this approach we propose a novel attack to identify POIs in trajectories, named *D-TOUR*, that investigates whether a trajectory contains unnecessary detours. This idea is based on the reasoning that in order to reach a POI, it is usually necessary to leave the path that would be ideal to attain the final destination of the route. The attack is outlined in Algorithm 1 and consists of four steps, described below. Figure 1 illustrates the functionality of the attack on an unprotected and a protected route.

(1) Map-matching The adversary considers a single trajectory and first maps the locations of each of its events to the road network (lines 3 to 5). This so-called map-matching is a typical pre-processing routine of raw trajectories, because data collected with GPS can be inaccurate and off roads [Zh15a]. This step is optional and not shown in Figure 1.

(2) Subset selection In a second step (lines 7 to 14), the attacker successively selects several events from the trajectory, starting with the first event and maintaining a certain selection distance δ between successive selected events. The parameter δ needs to be adjusted depending on the tracking interval (the time passed between consecutive events) of the data. If δ is too short, detours will not be detected, whereas if it is too long, parts of a route will be wrongly interpreted as a detour. The adversary adds the last route point to the selection in case it was not selected. In Figure 1 the selected points are highlighted by black circles.

(3) Distance to optimal route calculation In the third step (lines 16 to 26), the adversary aims to find deviations between the user's route and an optimal route. For this, the adversary first connects successive points from the selection (line 18) using the function $OPT(e_i, e_j)$ that returns the way points of the optimal route between the locations of the events e_i and e_j. For example, the optimal route can be interpreted as the fastest road connection and queried by public routing services[7], taking into account the user's known means of transport. In Figure 1 the fastest route is shown as a gray line.

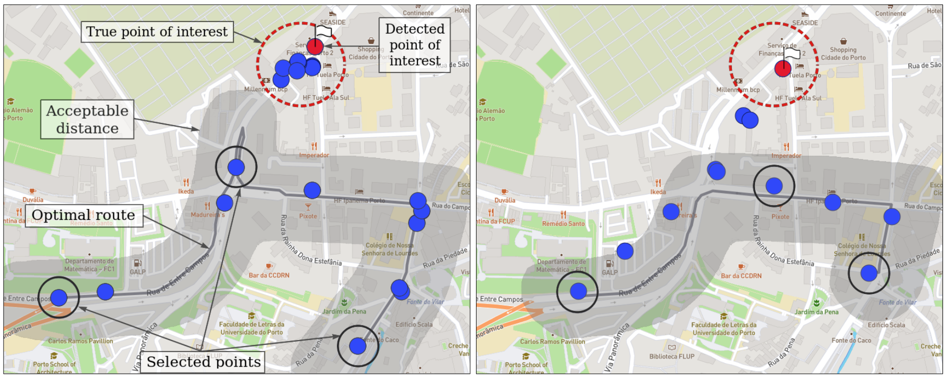
Then, for each point of the user's trajectory, the adversary aims to calculate the shortest distance to the optimal route. A routing service's response usually contains way points that

[7] In our experiments, calculation of the fastest route is done with openrouteservice.org.

Algorithm 1 D-TOUR: Detour-based POI detection

```
Require: route $L = [e_1, e_2, \ldots, e_n]$, selection distance $\delta \geq 0$, sampling distance $\sigma \geq 0$, acceptable
    distance $\gamma \geq 0$
function D-TOUR($L, \delta, \sigma, \gamma$)
                                                                        // Step 1
    for $i = 1, 2, \ldots, n$ do                     ▷ Map each route point to road network.
        $e_i \leftarrow MAP(e_i)$
    end for
                                                                        // Step 2
    $c \leftarrow 1$                                 ▷ Take a selection $C$ of point indices from the route,
    $C \leftarrow [1]$                                  starting by first, keeping a minimal distance $\delta$
    for $i = 2, 3, \ldots, n$ do                        between consecutive events.
        if $DIST(e_c, e_i) \geq \delta \vee i = n$ then
            $c \leftarrow i$
            Add $i$ to $C$
        end if
    end for
                                                                        // Step 3
    $X \leftarrow []$                                ▷ The exceedances of the acceptable distance.
    for $i = 2, 3, \ldots, |C|$ do                   ▷ Loop over selection.
        $O \leftarrow OPT(e_{C[i-1]}, e_{C[i]})$     ▷ Calculate optimal route $O$ between consecutive events.
        $S \leftarrow SAMPLE(O, \sigma)$             ▷ Take regular samples $S$ with distance $\sigma$ from $O$.
        $L^c = [e_{C[i-1]+1}, e_{C[i-1]+2}, \ldots, e_{C[i]}]$   ▷ The route section between selected points.
        for $e$ in $L^c$ do                          ▷ Loop over route section.
            $d = \min_{s \in S} DIST(e, s)$          ▷ Calculate distance $d$ to nearest sample.
            $x \leftarrow \max(d - \gamma, 0)$       ▷ Calculate by how much $d$ exceeds
            Add $x$ to $X$                              the acceptable distance $\gamma$.
        end for
    end for
                                                                        // Step 4
    $I \leftarrow []$                                ▷ The point indices of a POI candidate region.
    $P \leftarrow []$                                ▷ The identified POIs.
    $b \leftarrow X[1] > 0$                          ▷ $b$ is True, if a POI candidate region is detected.
    for $i = 1, 2, \ldots, n$ do                     ▷ Loop over route.
        if $X[i] > 0$ then                           ▷ If a route point exceeds the acceptable distance,
            Add $i$ to $I$                              consider it a POI candidate.
            $b \leftarrow True$
        end if
        if $b \wedge (X[i] = 0 \vee i = n)$ then     ▷ If a closed region of connected POI candidates
            $p = \arg\max_{j \in I} X[j]$               is detected, the candidate $p$ with the greatest
            Add $p$ to $P$                              distance exceeding is considered a POI.
            $b \leftarrow False$
            $I \leftarrow []$
        end if
    end for

    return $P$                                       ▷ Return the identified POIs.
end function
```



(a) Unprotected route

(b) Protected route using *Promesse* (α = 200 meters)

Fig. 1: (a) An example of a *D-TOUR* attack on a privacy-sensitive route. Based on a selection of points from the route, *D-TOUR* calculates the optimal route and considers points that deviate from it by more than a certain acceptable distance as POI candidates. Contiguous regions of POI candidates are interpreted as a detour. The furthest point in such a region is identified as a POI. (b) The protection with *Promesse* does not conceal the detour and *D-TOUR* can still identify a point within a certain circular area around the true POI location.

can be far apart. To find a point's closest neighbour on the optimal route it is necessary to sample regularly spaced points from the route in short intervals. For this the function $SAMPLE(L, \sigma)$ moves along the route L on a straight line between consecutive points (line 19) and takes samples from this path in regular intervals of length σ by measuring the location. Each sample forms an event without time and is added to a list. Small values of σ favor a more accurate distance calculation but have greater computational demand.

Using the function $DIST(e_i, e_j)$, the adversary then calculates for each route point e_i the Euclidean distance to each of the sampled points e_j from the optimal route and selects the shortest distance in each case (line 22). At the same time the adversary sets a threshold γ as an acceptable distance between a point and the optimal route and keeps track of by how much each of the calculated distances exceeds this value (lines 23 and 24). The acceptable distance γ is outlined as a gray area around the optimal route in Figure 1.

(4) POI detection In the final step, the adversary considers each route point that exceeds the acceptable distance γ as a POI candidate (lines 32 to 35). Consecutive POI candidates then form a candidate region and indicate a detour. In Figure 1 these candidates are those points, that lie outside of the gray area. For each of these candidate regions, the point with the greatest distance to the optimal route is considered a POI (lines 36 to 41). Figure 1 shows a true POI as a circular area around its actual location. A POI detected by an attack is marked with a flag and considered a true positive if it lies within the circular area.

Tab. 1: Data sets used in experimental evaluation.

Data set	Location	Tracking interval	Timespan	No. routes
Cabspotting	San Francisco (USA)	60 sec	17 May - 10 Jun 2008	12,000
Porto Taxi	Porto (Portugal)	15 sec	1 July - 2 Aug 2013	25,800

5 Experiments

In this section we describe an evaluation of our proposed *D-TOUR* method on two real-world data sets, and discuss our findings.

Data sets: We run experiments on two data sets as shown in Table 1. *Cabspotting* [PSG09] is a collection of taxi routes and contains the occupancy of a cab which we use to derive the pick-up and drop-off points of taxi customers, which serve as the POI ground truth. The *Porto Taxi* [Mo13] data set only includes trips made by occupied cabs and does therefore not contain continuous routes. Trip endpoints correspond to the customer pick-up and drop-off locations and are regarded as POIs. We clean the Porto Taxi data set from trips that take less than a minute or that contain presumably incorrect measurements that indicate speeds of more than 100 kilometers per hour.

We then match trips with the same taxi identifier and day and join them to form a continuous route, either by a direct link, the fastest route, or a random walk. To ensure a realistic matching we compare, among other things, the gap duration between trip endpoints to the duration of their shortest road connection. If the information is not coherent, we create a random walk that satisfies the duration constraints on the endpoints. A random walk is searched for a maximum of 500 attempts. If no valid route is found, a new route is created from the trip. A new route is also created if the gap is longer than 60 minutes.

Experimental setup: We first evaluate our proposed *D-TOUR* algorithm on the raw non-private trajectories of each data set. In a second step we then protect the data with *Promesse* [Pr15] to smooth temporal and spatial features of the routes and hide POIs. We argue that a trajectory protected in such a way still contains anomalies that indicate POIs. Therefore we evaluate POI detection on the protected routes with our detour-based detection approach[8]. We run experiments with data protected by smoothing distances α of 200, 300 and 400 meters.

In both steps we compare the performance of *D-TOUR* to a variation of the DJ-Cluster algorithm as described by Primault et al. [Pr14], which is a commonly used stop detection baseline [HT04, Li08]. In this algorithm, first, stays are extracted which are then aggregated into POIs. The algorithm takes as parameters a time threshold Δt which corresponds to the minimum time that has to be spent in a stay, and a distance threshold Δl which represents the maximum diameter of the stay area.

[8] For implementation see: https://github.com/majaschneider/detour-detection.

Evaluation metrics: We measure the performance of a stop detection algorithm based on how many of a route's POIs are correctly detected and whether detected POIs indeed denote a true stop. Following the approach of Primault et al. [Pr14] we define a detected stop and a true stop to be equal when their geographical distance is smaller than a parameter β. With this definition we can count the number of correctly detected true stops (true positives), as well as the number of stops that have not been detected (false negatives), and the number of stops that have been detected but are in fact not a true stop (false positives). In our experiments we consider $\beta = 200$ meters.

We calculate *Recall*, *Precision* and *F-Score* to evaluate the performance of stop detection. Recall denotes the ratio of correctly classified stops to all true stops. If it is low, a high number of stops have not been detected. Precision, on the other hand, is the ratio of correctly detected stops to the total number of stops that have (correctly or incorrectly) been identified as true stops. A low precision value indicates therefore a high false positive rate. Both metrics are combined in the F-Score, which is defined as their harmonic mean.

Experimental results: As opposed to Primault et al. [Pr15] we do not assume fixed parameters for stop detection but tune them on a subset of 2000 routes from each data set to obtain the best settings for both DJ-Cluster and *D-TOUR*. For DJ-Cluster we test time thresholds Δt of 1, 3, 5, 10 and 15 minutes, and distance thresholds Δl of 200, 300, 400, 500 and 1000 meters. The best parameters showed to be $\Delta t = 1$ minute and $\Delta l = 200$ meters for the Cabspotting data set which is amongst the four best settings for the Porto Taxi data set as well (only 3% lower F-Score). Therefore we choose these settings for both data sets.

The success of *D-TOUR* depends heavily on how well the spatial distance σ captures the outline of the route and avoids selected points to be located close to a POI. Therefore we also test combined results of multiple spatial distances and consider only those POIs that have been correctly identified in most settings. In our experiments we tested all such combinations of spatial distances σ of 100, 110, 310, 320, 410, 420, 610, 620 and 1000 meters together with acceptable distances γ of 20, 30, 50, 70, 100 and 150 meters, both with and without map-matching. However, the best parameters (for both data sets) were a single spatial distance $\sigma = 620$ meters and an acceptable distance $\gamma = 20$ meters using no map-matching.

The results of our experiments are presented in Table 2. Both attacks are capable of capturing POIs in unprotected data. The overall performance is low with a maximum F-Score of 0.45, indicating that POI detection is a difficult task. Both attacks have a comparable F-Score in the Cabspotting dataset, while DJ-Cluster in the Porto-Taxi dataset performs 6% better.

As expected, with increasing protection (i. e. larger smoothing parameter α) both attacks are less successful with regards to F-Score. The performance of DJ-Cluster drops for the lowest protection of $\alpha = 200$ meters by up to 22%. *D-TOUR* also shows a performance drop but is able to achieve an F-Score which is, compared to the baseline, 13% better for Cabspotting and 3% better for Porto Taxi data. While in general all performance measures decrease with higher privacy protection, an exception is recall for *D-TOUR* which is improving

Tab. 2: Experimental results.

Algorithm	Protection (*Promesse*)	Cabspotting			Porto Taxi		
		Recall	Precision	F-Score	Recall	Precision	F-Score
DJ-Cluster	Unprotected	0.43	0.47	0.45	0.53	0.34	0.41
	$\alpha = 200m$	0.20	0.34	0.25	0.42	0.13	0.19
	$\alpha = 300m$	0	0	0	0	0	0
	$\alpha = 400m$	0	0	0	0	0	0
D-TOUR	Unprotected	0.51	0.39	0.44	0.38	0.32	0.35
	$\alpha = 200m$	0.67	0.27	0.38	0.43	0.15	0.22
	$\alpha = 300m$	0.52	0.27	0.35	0.15	0.10	0.12
	$\alpha = 400m$	0	0	0	0	0	0

significantly for the lowest protection, rising by 15% (Cabspotting) and 5% (Porto Taxi). The *D-TOUR* algorithm correctly detects more POIs but also makes more mistakes, leading to an overall reduced F-Score. With $\alpha = 300$ meters DJ-Cluster is not able to detect any POIs anymore, whereas *D-TOUR* still achieves an F-Score of up to 0.35. Only with $\alpha = 400$ meters *D-TOUR* is also not able to capture any POIs anymore.

Discussion: POIs found with *D-TOUR* could be further evaluated using additional knowledge. For example, an adversary can try to label POIs using public map data and estimate the probability of correctness using contextual knowledge (such as the source of the data). Also, different types of attacks could be combined to increase the confidence in the results.

However, a privacy preserving mechanism must guarantee that POIs are sufficiently protected, which is why even a low recall is a potential threat to privacy. Further addition of noise (e. g. higher smoothing value α) can help mitigate this issue but is ultimately limited as it deteriorates utility. Instead, a more targeted elimination of anomalies can be beneficial to ensure privacy is protected and utility stays high.

In this context, *Promesse* introduces a spatial error. This can not only lead to a decrease of utility but can be a reason for the performance drop of *D-TOUR* when protected data is attacked. The attempted mitigation with map-matching did not provide an improvement in our experiments, which indicates that the spatial error introduced by *Promesse* is too large.

Visual inspection of the data shows that the street layouts in the investigated cities differ, which can also have an impact on the spatial error in protected data. Even though the tracking interval of San Francisco cabs is larger, the grid layout of the streets in this city keeps the spatial error low. We hypothesize that this is the reason for the better performance of *D-TOUR* in the Cabspotting data set compared to the Porto Taxi data set.

In addition, the fact that start and endpoints of Porto Taxi routes are considered POIs as well contributes to the worse performance of *D-TOUR* on this data set. Heuristics could be used as additional treatment, such as the begin-end-heuristic [GKd11], which simply picks the first and last trajectory point as POIs. This is a low-cost approach based on the idea that start and endpoints of routes are usually meaningful.

According to the authors of *Promesse* [Pr15], the smoothing parameter α is key to hiding POIs that would be otherwise extracted with a parameter $\Delta l \leq \alpha$ in a DJ-Cluster attack. Our experimental results are consistent with this reasoning as POI detection performance of DJ-Cluster drops significantly with $\alpha = 200$ meters and $\Delta l = 200$ meters, and even further with $\alpha = 300$ meters. *D-TOUR*, on the other hand, is still able to detect POIs in this setting, which shows that it is not sufficient to evaluate privacy requirements based only on DJ-Cluster.

6 CONCLUSION AND FUTURE WORK

We showed experimentally on two real-world data sets that POIs in trajectories of mobile users can be regarded as an anomaly from typical movement behaviour, which can be detected by searching for detours in a route. We proposed a novel attack, *D-TOUR*, that compares a sensitive trajectory to the fastest route between a selection of its points and chooses the furthest point of such a deviation from the optimal route as a POI. Compared to DJ-Cluster, a state of the art stop detection algorithm, our attack achieves similar performance on unprotected data but outperforms the baseline when privacy protection was increased.

As future work we will focus on investigating the influence of the actual transport mode (such as walking, car, or subway) on the attack success. We also plan to investigate more diverse data sets containing different data producers, such as logistics companies or tourists that show different moving behaviour and approach other types of POIs. We furthermore plan to investigate whether results can be improved with a revised map-matching approach, that matches points preferentially with larger roads to reduce the spatial error.

Acknowledgements This work is partially funded by the German Federal Ministry of Education and Research under grant BMBF 01IS18026B at ScaDS.AI Dresden/Leipzig.

Bibliography

[ABN10] Abul, Osman; Bonchi, Francesco; Nanni, Mirco: Anonymization of moving objects databases by clustering and perturbation. Inf. Syst., 35(8):884–910, 2010.

[AS03] Ashbrook, Daniel; Starner, Thad: Using GPS to learn significant locations and predict movement across multiple users. Pers. Ubiquitous Comput., 7(5):275–286, 2003.

[BBA15] Ben Ayed, Abdelkarim; Ben Halima, Mohamed; Alimi, Adel M.: Big data analytics for logistics and transportation. IEEE ICALT, pp. 311–316, 2015.

[CRS20] Christen, Peter; Ranbaduge, Thilina; Schnell, Rainer: Linking Sensitive Data. Springer, Heidelberg, 2020.

[GKd11] Gambs, Sèbastien; Killijian, Marc Olivier; del Prado Cortez, Miguel Núñez: Show me how you move and I will tell you who you are. Trans. Data Priv., 4(2):103–126, 2011.

[HT04] Hariharan, Ramaswamy; Toyama, Kentaro: Project lachesis: Parsing and modeling location histories. In: Proc. GISci. pp. 106–124, 2004.

[Ka05] Kang, Jong Hee; Welbourne, William; Stewart, Benjamin; Borriello, Gaetano: Extracting places from traces of locations. ACM SIGMOBILE Mob. Comput. Commun. Rev., 9(3):58–68, 2005.

[Kr07] Krumm, John: Inference attacks on location tracks. In: Proc. 5th Int. Conf. Pervasive Comput. pp. 127–143, 2007.

[Li07] Liao, Lin; Patterson, Donald J.; Fox, Dieter; Kautz, Henry: Learning and inferring transportation routines. Artificial Intelligence, 171(5-6):311–331, 2007.

[Li08] Li, Quannan; Zheng, Yu; Xie, Xing; Chen, Yukun; Liu, Wenyu; Ma, Wei Ying: Mining user similarity based on location history. In: Proc. ACM SIGSPATIAL GIS. pp. 298–307, 2008.

[LLL16] Liu, Xin; Liu, Yong; Li, Xiaoli: Exploring the Context of Locations for Personalized Location Recommendations. In: IJCAI. pp. 1188–1194, 2016.

[Mo13] Moreira-Matias, Luis; Gama, Joao; Ferreira, Michel; Mendes-Moreira, Joao; Damas, Luis: Predicting taxi-passenger demand using streaming data. IEEE Trans. Intell. Transp. Syst., 14(3):1393–1402, 2013.

[Mo16] Moreira-Matias, Luís; Gama, João; Ferreira, Michel; Mendes-Moreira, João; Damas, Luis: Time-evolving O-D matrix estimation using high-speed GPS data streams. Expert Syst. Appl., 44:275–288, 2016.

[Pa03] Patterson, Donald J; Liao, Lin; Fox, Dieter; Kautz, Henry: Inferring High-Level Behavior from Low-Level Sensors. In: Proc. UbiComp. pp. 73–89, 2003.

[Pr14] Primault, Vincent; Mokhtar, Sonia Ben; Lauradoux, Cedric; Brunie, Lionel: Differentially Private Location Privacy in Practice. In: Proc. MoST. 2014.

[Pr15] Primault, Vincent; Ben Mokhtar, Sonia; Lauradoux, Cédric; Brunie, Lionel: Time distortion anonymization for the publication of mobility data with high utility. IEEE Trustcom/BigDataSE/ISPA, 1:539–546, 2015.

[PSG09] Piórkowski, Michal; Sarafijanovic-Djukic, Natasa; Grossglauser, Matthias: A parsimonious model of mobile partitioned networks with clustering. In: Proc. of COMSNETS. 2009.

[Sh11] Shokri, Reza; Theodorakopoulos, George; Le Boudec, Jean Yves; Hubaux, Jean Pierre: Quantifying location privacy. In: Proc. IEEE Symp. Security Privacy. pp. 247–262, 2011.

[Zh04] Zhou, Changqing; Frankowski, Dan; Ludford, Pamela; Shekhar, Shashi; Terveen, Loren: Discovering personal gazetteers: An interactive clustering approach. In: Proc. ACM SIGSPATIAL GIS. pp. 266–273, 2004.

[Zh15a] Zheng, Yu: Trajectory data mining: An overview. ACM Trans. Intell. Syst. Technol., 6(3):1–41, 2015.

[Zh15b] Zhong, Yuan; Yuan, Nicholas Jing; Zhong, Wen; Zhang, Fuzheng; Xie, Xing: You are where you go: Inferring demographic attributes from location check-ins. In: Proc. WSDM. pp. 295–304, 2015.