Understanding Vulnerabilities of Location Privacy Mechanisms against Mobility Prediction Attacks

Zohaib Riaz, Frank Dürr, Kurt Rothermel
Institute for Parallel and Distributed Systems
University of Stuttgart, Stuttgart, Germany
{zohaib.riaz,frank.duerr,kurt.rothermel}@ipvs.uni-stuttgart.de

ABSTRACT

In today's online social networks such as Facebook, users increasingly share their location information as a popular type of personal information. However, since location data can leak privacy-sensitive information about individuals such as the type of places they like to visit, a number of *location obfuscation* mechanisms have been proposed to avoid such disclosure. These mechanisms publish bigger regions containing the actual user location in order to make it imprecise. Thus an attacker may find it hard to precisely locate the user in a privacy-sensitive place such as a hospital.

In this paper, we show that state-of-the-art location obfuscation mechanisms do not provide privacy guarantees against attacks based on *mobility prediction*. In this regard, we design and demonstrate a mobility prediction attack that exploits location history information of users and show its effectiveness on a year-long real-world location dataset. In particular, our results show that such an attack can successfully de-obfuscate up to 50% of sensitive user visits with high precision ($\geq$ 80%), *even when the location history data used for the attack is already obfuscated*. We also analyze the success of our mobility prediction attacks and suggest important design improvements for future location privacy mechanisms.

CCS CONCEPTS

• **Information systems** → *Social networks*; • **Security and privacy** → **Privacy protections**; • **Computing methodologies** → **Adversarial learning**;

KEYWORDS

location privacy, attack algorithms, semantic location information, mobility prediction, hidden Markov models

ACM Reference format:
Zohaib Riaz, Frank Dürr, Kurt Rothermel. 2017. Understanding Vulnerabilities of Location Privacy Mechanisms against Mobility Prediction Attacks. In *Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, Melbourne, VIC, Australia, November 7–10, 2017 (MobiQuitous'17),* 10 pages.
https://doi.org/10.1145/3144457.3144505

MobiQuitous'17, November 7–10, 2017, Melbourne, VIC, Australia

ACM ISBN 978-1-4503-5368-7/17/11…$15.00
https://doi.org/10.1145/3144457.3144505

1 INTRODUCTION

Today's online social networks (OSNs) such as Facebook, Google+, and Twitter provide their users with multi-faceted personal information sharing options. Millions of users avail these options on their mobile devices to share, among other personal information, their location data by either geo-tagging their online content (photos and posts) or, more explicitly, as status updates and location check-ins. While sharing location information certainly brings important benefits, such as keeping close ones informed about whereabouts, it also puts personal privacy at stake by disclosing sensitive information like a user's home location, their revealing visits (e.g., to a doctor), as well as their frequently visited locations [14]. An important reason for such private disclosures is the fact that raw location data, or geo-coordinates, can be easily interpreted into highly meaningful visit information by enriching it with semantic information such the types of locations visited by the users (e.g., hospitals, church, etc.).

An obvious way to maximize user privacy is to *suppress* sharing of visits that convey user presence in privacy-sensitive place types as studied in [8]. However, suppression of location information not only disables any location-based services that the user may require (e.g., nearby point-of-interest searches), it may also be undesired in the OSN scenario where users also share location information in response to requests from their social connections [4]. An alternative privacy mechanism that promises to strike a balance between user privacy and their location sharing needs is *spatial obfuscation* whereby the actual location of the user is made *spatially imprecise* before sharing [3, 6, 13]. For instance, a user may share his location with his friends as a bigger region containing his actual location thereby hiding his exact whereabouts while approximately satisfying the location sharing requirement from his friends.

Unfortunately however, a spatially obfuscated location may be easily *pruned or de-obfuscated* by a privacy attacker possessing additional background information. For example, an attacker possessing a map may know that the user reported obfuscation region, for example, contains only a single underlying venue and that too of type hospital thus threatening user privacy. To prevent such disclosure, advanced *semantic cloaking* mechanisms (SCMs) were introduced to ensure that an obfuscation region, also called a *cloaking region* (CR), is l-diverse, i.e., it subsumes $l \geq 2$ semantically different locations (i.e., $l-1$ cover locations plus the actual location) inside the obfuscation region [18]. For example, with $l = 3$, the CR published for visiting a bar may also contain a nearby restaurant as well as a grocery store. Thus, an attacker is unable to trivially determine the semantics of user visit. For selecting the $l-1$ cover (semantic) locations, existing works [5, 11, 20] propose various algorithms that are designed to be robust against additional background knowledge

of the attackers including map-based venue information, statistical spatial distribution of users, and popular visiting times of various semantic locations.

However, a common drawback of all these approaches is that they ignore the possibility of an attacker having access to users' location history information. Obviously, this possibility is highly likely as attackers take various roles in the location sharing setting (e.g., an acquaintance in OSNs, the OSN provider, or any other location-based service provider in general) and may posses location information previously shared with them by the users. Moreover, it is also realistic to assume that the attacker can semantically label the historic location information by using online venue directories such as Foursquare [1]. Thus with semantically labeled location histories, the attacker can acquire an approximate knowledge of users' mobility which in turn can be used to de-obfuscate their location data, thereby, significantly raising the level of privacy risk for users.

Therefore, in this paper, we study the detrimental effect of location history information on privacy guarantees that are offered by existing SCMs. To this end, we show that, by exploiting users' location histories, attackers can indeed de-obfuscate privacy-sensitive user visits that are protected using existing SCMs. Note that in the wider context of data deanonymization, successful attacks on protected data are generally only possible due to the availability of additional public information (e.g., external datasets). In stark contrast however, the attacks that we design in this paper require no external datasets and are also effective when only the *obfuscated* location history information, which is released gradually over time by existing SCMs, is available. In more detail, we contribute to the state of the art in the following ways:

- We critically analyze the construction of the existing SCMs and discuss their design limitations.
- We design a novel *semantic mobility model* to predict user movements along the location-semantics dimension using Hidden Markov Models (HMMs).
- We handle the non-trivial training of HMMs from location history information by defining algorithms to flexibly take advantage of both, the obfuscated and the non-obfuscated visits, of the user. We also ensure that the training process can accommodate non-periodic location updates in the history information as well as temporal discontinuities.
- We demonstrate, in concrete steps, the success of our attacks on real-world movement traces of 278 users in the form of their location check-in histories.
- Finally, we analyze the success of our attacks and discuss possible improvements for the design of SCMs that are robust against location history information attacks.

By validating the success of our attacks on real-world data, our work highlights the realistic nature of the posed privacy threat and, hence, points to an important gap in semantic location privacy research that needs immediate attention from the research community.

Next, we briefly explain the existing semantic cloaking mechanisms which are the focus of attacks developed in this paper.

2 SEMANTIC CLOAKING MECHANISMS: BACKGROUND AND CRITICAL ANALYSIS

In this section, we discuss the construction of existing semantic cloaking mechanisms (SCMs) [5, 11, 18, 20]. We also point out the detailed limitations of these SCMs that make them vulnerable to attacks.

2.1 Overview

On a superficial level, the basic property of cloaking regions (CRs) generated by all existing SCMs is the same, i.e., they include $l \geq 2$ semantically different locations from the finite set $\mathbb{S}$ of all possible semantic location types such as hospital, restaurant, home, etc. For example, Fig. 1 shows two CRs with l-diversity values of 4 and 2. Generating a CR with incrementally higher values of l equates to monotonic addition of more confusion about the actual user (semantic) location for the attacker, and therefore, increases user privacy. However, this increased privacy comes at the cost of decreased utility of the CR, i.e., its meaningfulness in informing about the users' actual semantic location is proportionally reduced. Therefore, in general, the choice of the value of l is bounded on the upper end by users' minimum utility requirements and on the lower end by their minimum privacy demands.

For the generation of a CR, every SCM guarantees privacy according to a specific attack metric, which we represent as a function, $f_{att}(CR)$. For example, $f_{att}(CR)$ can represent the probability with which an attacker can find out the correct semantic label for the actual user location (visited venue) inside the CR. Thus SCMs can ensure user privacy by bounding $f_{att}(CR)$ to under a user-defined privacy threshold θ_{priv}.

2.2 Existing SCMs and their limitations

The initial l-diversity based approach proposed in [18] assumes $f_{att}(CR) = 1/l$. Note that this definition of $f_{att}(CR)$ treats every semantic location equally in terms of the amount of confusion it generates for the attacker. As this may not be true, advanced mechanisms consider additional information about semantic locations. In [11], the authors define $f_{att}(CR)$ in the sense of information gain for the attacker as the distance between the probability distribution of semantic locations inside the CR and their distribution inside the surrounding region, e.g., the full city. Hence, different semantic locations effectively get weighted differently according to their probabilities in the two distributions.

However, the above works assume that an attacker cannot associate together multiple location updates reported by a single user. Moreover, they do not allow personalization in generation of CRs, i.e., the same CRs are generated for every user for the same locations. The more general SCMs proposed in [5, 20] address these issues and are, thus, the focus of our attacks in this paper. We now discuss these general SCMs in more detail.

2.2.1 The General SCMs [5, 20]. These SCMs protect a user-defined set $\mathbb{L} \subset \mathbb{S}$ of *sensitive* semantic locations, e.g., hospitals, churches, etc., while separately defining the privacy threshold θ_{priv} (also called *sensitivity*) for each sensitive location. By only focusing on the sensitive set of locations, these SCMs can potentially avoid high utility loss by selective obfuscation of user trips according to

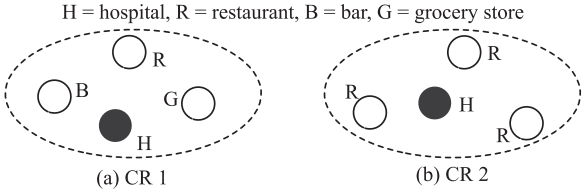

Figure 1: Two CRs with different l-diversity values. (a) a CR with $l = 4$. (b) a CR with $l = 2$ due to multiple occurrences of venues of type *restaurant*.

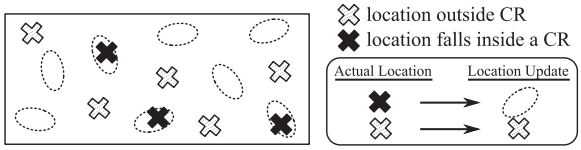

Figure 2: Location update algorithm used by SCM$_{pop}$

the particular user's sensitivities for different locations (personalization).

CR Generation. The general CR generation algorithm for these SCMs ensures that the probability of user presence in a sensitive location $s_i \in \mathbb{L}$ inside the CR, represented by $f_{att}(CR)$, is lower than $\theta^{s_i}_{priv}$. Thus even though the attacker knows the set $\mathbb{L}$, he may be unable to detect when a location $s_i \in \mathbb{L}$ is visited by the user.

To estimate the probability of user presence in different semantic locations, this algorithm exploits the statistical popularities of these locations among the user population. Formally, the attack metric is defined as follows.

$$f_{att}(CR_{s_i}) = \frac{pr_{s_i}}{\sum_{\forall s_j \in CR} pr_{s_j}} \quad (1)$$

where pr_{s_i} is the popularity of the sensitive location s_i that is present within this CR (only one sensitive location per CR). The CR *expands* out from s_i to incrementally include nearby locations until the condition $f_{att}(CR_{s_i}) \leq \theta^{s_i}_{priv}$ is satisfied. By fixing the start of this expansion from the sensitive location s_i, the CR generation algorithm provides an important privacy guarantee, i.e., CRs are generated in an independent manner without any dependence on the user's location inside the CR. In other words, the CR generation algorithm leaks no information about the actual user location.

For future reference, we will call this approach *popularity-based semantic cloaking* or SCM$_{pop}$ in short.

Location Updates. To perform location updates, the fundamental question to answer is *when to generate a CR?* A naive update algorithm may be to generate a CR only when the user visits a sensitive location. However, if an attacker knows the update algorithm as well as the user's set of sensitive locations $\mathbb{L}$, he may easily conclude on observing a CR that the user has visited one of his sensitive locations. To avoid this privacy breach, SCM$_{pop}$ performs location updates in a two-stage process as shown in Fig. 2. In the first stage, all possible CRs are generated in an offline fashion for each occurrence of users' sensitive locations $s_i \in \mathbb{L}$ over the city-map according to their user-defined sensitivities. This map with CRs is available to the SCM$_{pop}$ algorithm to perform real-time location updates (the second stage) in the following way. The algorithm checks whether the current location of the user falls within a CR, i.e., within any venue, *not necessarily the sensitive one*, that is part of a CR. If so, it publishes the particular CR. Otherwise, it publishes the actual location. The location update algorithm additionally avoids attacks based on the last reported location. More precisely, it ensures that all venues inside the currently published CR are reachable from the last reported user location with a reasonable movement velocity [7].

2.2.2 Limitations of SCM$_{pop}$. As pointed out earlier, an important limitation of SCM$_{pop}$ (as well as other SCMs in general) is that it assumes that the attacker does not possess users' location history information. Moreover, since SCM$_{pop}$ only focuses on reducing relative popularity of s_i inside the CR (see Eq. 1), it does not forbid multiple venues of the same semantic type to act as cover locations thus providing no direct control to the users over the l-diversity of the generated CRs apart from the assurance that $l \geq 2$. To illustrate this case with an example, Fig. 1 shows two CRs generated by SCM$_{pop}$ for two differently located *hospitals* (the sensitive location). Due to the different distribution of nearby locations for the two hospitals, the resulting l-diversities of these two CRs differ ($l = 4$ & 2 for CR1 and CR2 respectively) although both the CRs satisfy the privacy requirement $f_{att}(CR) \leq \theta_{priv}$. Thus if an attacker possesses additional background knowledge, he may find it easier to de-obfuscate the 2-diversity CR generated by SCM$_{pop}$. In such cases, the basic l-diversity approach with a fixed value of l, e.g., $l = 4$, will ensure higher privacy for the user than SCM$_{pop}$ by always ensuring a minimum of 4 different semantic locations inside each CR independently of the distribution of nearby locations.

Hence, for our attacks in this paper, we propose an improved version of SCM$_{pop}$ to represent the state-of-the-art. This improved version is similar to SCM$_{pop}$ in all respects apart from its CR generation algorithm where, unlike SCM$_{pop}$, it uses the basic l-diversity principle to guarantee a fixed value of l for all generated CRs. For the rest of the paper, we refer to this improved version as *improved-SCM$_{pop}$* or ψ for short.

3 SYSTEM MODEL AND PROBLEM STATEMENT

In this section, we first introduce the location sharing setting. In doing so, we explain the roles and knowledge (about users' location information) of the entities in this setting. Based on this, we state the attacker model and define the problem of attacking users' privacy using historic location information.

3.1 The System Model

As shown in Fig. 3, the location sharing scenario comprises three entities, namely, the users, their social connections, and the OSN platform.

To share location information, users as well as their social connections (other users such as friends, family, etc.) register with a common OSN platform. To perform location sharing, a user accesses the OSN platform through his location-enabled mobile device. Apart from the OSN app instance, the mobile device runs the SCM ψ algorithm (cf. Section 2.2) to appropriately obfuscate location data before sharing it with the OSN platform.

The OSN platform typically stores user shared location information in order to make it available to other users as well as for

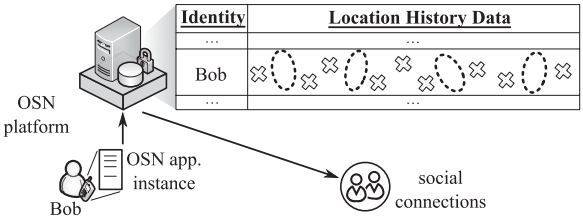

Figure 3: The location sharing scenario in OSNs with the user (Bob), the OSN platform, and the users' social connections.

providing additional features such as historic user time-lines or targeted advertizements, etc. The OSN platform also provides means for users to define their location sharing preferences so as to limit the amount of data shared with various social groups such as family or friends, etc.

The social connections of users access their location information from the OSN platform according to the allowances defined in users' sharing preferences. Thus, social connections may have access to full or partial location history information about the users.

3.2 Attacker Model

We assume that the OSN provider as well as the social connections can be malicious and hence can act as attackers.

We also assume that the attacker has aggregated the past location updates from the target user into a set called the location history H. We consider the two cases separately where H is: (1) *accurate*, i.e., the user had not applied any SCM to his past location updates, or (2) *obfuscated*, where the user already applied SCM ψ to his location updates in the past. Aside from possessing H, we only make those assumptions about the attacker's knowledge that are also made by existing SCM ψ [5, 20]. Consequently, we assume that the attacker knows the algorithmic construction of SCM ψ and the set of sensitive semantic locations $\mathbb{L} \subset \mathbb{S}$ of the target user as well as his real identity.

3.3 Problem Statement

In this paper, our goal is to show that the privacy guarantees offered by state-of-the-art SCMs, represented by the SCM ψ (cf. Section 2), do not hold against mobility prediction attacks based on location history information. Thus using location history, we want to de-obfuscate sensitive user trips that are cloaked (semantically obfuscated) using SCM ψ. We highlight here that *the de-obfuscation of any CR undermines user privacy only when the actual location visited inside the CR is among the sensitive locations $s_i \in \mathbb{L}$ of the user*. Note that in the other case, when the user actually visits a non-sensitive location inside a CR and publishes this CR, the attacker does not learn any privacy-sensitive information even by successful de-obfuscation of the CR. The intuition behind our attacks comes from the fact that, in regular usage, SCM ψ may reveal some of the non-sensitive trips accurately (without obfuscation) when they fall outside any of the users pre-generated CRs. Thus an attacker may learn useful information about the user's visting patterns to non-sensitive locations, which can potentially help him to de-obfuscate CRs and identify sensitive visits of the user.

3.3.1 The generalized attack problem. We represent the ground-truth about the sensitive or non-sensitive nature of user visits inside a given CR as an indicator function $g(CR)$ which takes a value of 1 if the sensitive location is actually visited and 0 otherwise. The attacker's goal is therefore to mimic the ground-truth function $g(CR)$, which is obviously *un*available to him, by making a prediction about the nature of user visits inside a given CR which we represent as another indicator function $g_{attack}(CR)$. The success of such an attack may be computed as the precision of detecting sensitive visits on a set of previously unseen CRs as defined below.

$$Pr(g_{attack}) = \frac{num\,(g_{attack}(CR_i) = g(CR_i) = 1)}{num(g_{attack}(CR_i) = 1)}, CR_i \in \mathbb{CR} \quad (2)$$

In words, the precision $Pr(g_{attack})$ represents the ratio of correctly identified sensitive user visits that occur in the previously unseen set $\mathbb{CR}$ to that of the total predicted sensitive visits. Thus, $Pr(g_{attack})$ can reach a maximum value of 1 when all predicted sensitive visits are also actually sensitive.

3.3.2 The mobility modeling based attack problem. To estimate $g_{attack}(CR)$, the attacker exploits the location history information $H = \{h_{t_0}, h_{t_1}, ..., h_t\}$ comprising observed location updates from the user until the time t. We call H the obfuscated location history when each location update $h_i \in H$ is released by SCM ψ.

Using H, the attacker learns a semantic mobility prediction model of the user denoted as Ω. When needed, we differentiate between the accurate and obfuscated versions of history H and the correspondingly learned model Ω with subscripts act and obf, i.e., $\Omega_{act} \leftarrow f(H_{act})$ and $\Omega_{obf} \leftarrow f(H_{obf})$, respectively.

Based on the mobility model Ω, the attacker predicts user locations inside any newly observed CRs, i.e., after time t. Each CR may be represented as a set $CR = \{S, I\}$. Here $CR.S$ represents a subset of semantic locations from the full set $\mathbb{S}$ (with m different location types) that are included inside the CR whereas $CR.I$ represents any contextual information about the CR (such as the publishing time and its average geo-location) that can be trivially determined. The attacker uses Ω to predict user location as $\hat{s} = f(\Omega, CR)$. This is achieved by determining the probability of visiting each of the locations $s_i \in CR.S$ according to Ω and then choosing $\hat{s}$ as the one with the maximum probability. Thus when $\hat{s}$ belongs to the set of the user's sensitive locations $\mathbb{L}$, the attacker assumes that the ground-truth is sensitive, i.e., $g_{attack}(CR) = 1$, and vice versa. For future reference, we denote the attack precision based on the mobility model Ω as $Pr(g_{attack}|\Omega)$.

3.3.3 SCM effectiveness metrics. Apart from attack precision, we are also interested in quantifying the negative impact Δ_ψ for the attacker of not possessing the actual location history H_{act} but rather only the obfuscated one H_{obf}. Let the function $\epsilon(\Omega) = (1 - Pr(g_{attack}|\Omega))$ represent the overall *loss of precision* for the attacker in general for a given mobility model Ω. Then Δ_ψ can be quantified easily by computing $\epsilon(\Omega_{obf}) - \epsilon(\Omega_{act})$, i.e., the difference between the loss of attack precisions due to the obfuscated and accurate location histories as follows.

$$\Delta_\psi = Pr(g_{attack}|\Omega_{act}) - Pr(g_{attack}|\Omega_{obf}) \quad (3)$$

An ideal attack should try to minimize this difference Δ_ψ down to zero, which will imply that obfuscating location updates using

state-of-the-art SCMs (represented by SCM ψ) does not hinder the attacker in learning as good a mobility model, from the obfuscated location history H_{obf} of the user, as he could have learned from the accurate one, i.e., H_{act}.

4 ATTACK ALGORITHMS

In this section, we present the semantic mobility model, which is built upon the location history information H and represents the attacker's main tool in de-obfuscating users location data. Note that our model is applicable to both types of location histories, i.e., actual and obfuscated, and handles the specifics for the two cases in a single model.

4.1 The semantic mobility model Ω

As introduced in Section 2.1, we assume that user visits can be classified as belonging to one of the semantic locations in the finite set $\mathbb{S}$. Thus user location at any time may be modeled, as popularly done in existing works [8, 9, 19], as a finite-state Markov chain where the state q_t (a semantic location) at time t takes one of the m possible values in $\mathbb{S}$. Thus user movements over a day can be represented as a sequence $q_1, ..., q_T$ in discrete-time by temporally dividing the day into T equal length time-slots. The probabilities $P(q_{t+1} = j | q_t = i) = a_{ij}$ of moving between two states in consecutive time-slots are defined by $m \times m$ non-homogeneous transition matrices A_t which vary with the discrete time $t \in 1, ..., T-1$ to accommodate different movement tendencies during the day. For example, a user may be more inclined to go to *shopping* places in the evening than during morning time. Finally, the probability of beginning in a certain state (at $t = 1$) is specified by a prior distribution π. Thus a users *basic* semantic mobility model can be specified as a tuple of parameters $\Omega = \langle \pi, A_{t \in \{1,...T-1\}} \rangle$.

This basic model however does not account for contextual information that the attacker can acquire with each observed location event $h_t \in H$. For example, for each user visit, the attacker gets to learn the time of visit, its approximate geo-location, as well as other complex information such as the types of locations inside the region. This information can be reciprocally used by the attacker to predict the type of location visited inside a CR. In fact, contextual information forms a fundamental *signal* for our algorithm to learn users' visiting patterns to sensitive locations which are never observed directly (only inside CRs).

To incorporate this information, we replace the basic model with a Hidden Markov Model (HMM) which can additionally accommodate contextual information as state-dependent *emissions* o_t, as shown in Fig. 4. An HMM treats states as hidden information that may be inferred based on emissions, which suits our setting where the attacker is unsure about the user location inside a CR but is able to capture CR related context information. In our design, we deviate from the norm of having uni-variate emissions to accommodate multivariate emissions in order to represent the various types of contextual information, most importantly, the temporal features of user visits such as hour of day, day of week as well as spatial variables such as the associated geographic region's id. Hence $o_t = \{o_t^1, ..., o_t^D\}$ in our model and each emitted variable o_t^k may take (from among) n^k different discrete values.

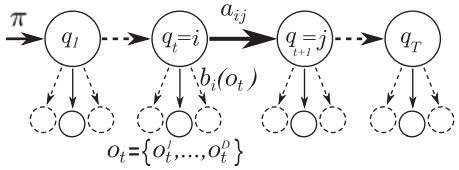

Figure 4: Discrete-time Hidden Markov Model with multivariate observations.

Since the emissions are assumed to depend only on the current state in HMMs, this information is encoded as an additional parameter matrix called emission probabilities B. In the standard case of uni-variate emissions, each row of the $m \times n$ B matrix corresponds to the conditional probability distribution, given a particular hidden state j, over the n values of the emitted variable o_t, i.e, $P(o_t | q_t = j) = b_j(o_t)$. With multivariate emissions, $b_j(o_t)$ represents the joint conditional distribution of all emission variables given state j. To simplify the approximation of this conditional distribution, we adopt the naive Bayes method to assume that the emitted variables are independent of each other given the hidden state. Hence $b_j(o_t)$ can be computed as follows:

$$b_j(o_t) = P(o_t | q_t = j) = \prod_{i=1}^{D} b_j^k(o_t^k) \text{ where } b_j^k \in B^k \quad (4)$$

Thus in our case, we have D instances of matrix B, one for each of the emitted variables in o_t. Hence, overall, our semantic mobility model is parametrized as $\Omega = \langle \pi, A_{t \in \{1,...T-1\}}, B_{k \in \{1,...D\}} \rangle$. Using this model and an emission sequence O, the attacker can compute useful probabilities such as $P(q_t = i | O, \Omega)$ that the user was in a certain state s_i at time t using existing algorithms [12].

4.2 Mapping Location History data to time-slots

The process of learning the parameters of the HMM Ω or its training from the location history H of a user requires data of two kinds, namely, a number of per-day sequences $\mathbb{Q}$ of the hidden states and their corresponding emission sequences $\mathbb{O}$. Any given emission o_t for a particular time-slot is said to be labeled if the corresponding hidden state, i.e., the semantic location, also referred to as the state-label, is available in the location history data. Existing literature offers methods for training HMMs given a fully labeled dataset (supervised learning when both the state-label sequences $\mathbb{Q}$ and the emission sequences $\mathbb{O}$ are known) or an unlabeled dataset (unsupervised learning when only $\mathbb{O}$ is available) [12]. As we will see below, in our case, neither of these cases apply directly as our dataset is labeled in a *mixed* manner, e.g., with parts that are missing labels as well as others that are over-labeled, as discussed below.

For training Ω, we first map the per-day data in H to its respective T time-slots. While HMMs assume that the modeled process (the target user in this case) resides in one of its states during a given time-slot, this assumption is not true for the non-periodically reported location updates in the history data H and thus needs accommodation from our model. As shown in Fig. 5, the mapping of H to time-slots results in four interesting cases[1]. Case 1) *missing data*: a time-slot with no location update. Case 2) *unlabeled data*:

[1]These cases are concluded from our study of a real-world dataset introduced in Section 5

one or more location updates without any available semantic label (emissions are available). Case 3) *(over-)labeled data*: one or more semantically-labeled location updates in a single time-slot. Case 4) *(over-)labeled data with obfuscation*: a time-slot with one or more CRs. Thus our HMM training algorithm generally needs to handle two types of data, *unlabeled* (cases 1 & 2) and *labeled* (cases 3 & 4). We further differentiate between the labeled data based on its obfuscation. We refer to the non-obfuscated labels from case 3 above as the *probabilistic (state-)labels*, i.e., $q_t = \{L, P\}$, where L contains the observed labels (one or more) and P defines the probability of these labels within the current slot. These probabilities may be inferred, for example, by analyzing H for additional information such as the time spent by the user within each state within the particular time-slot. As for the obfuscated labels from case 4 above, we call them the *non-probabilistic labels* as time-spent in all locations within a CR is apparently the same from the perspective of an attacker. Hence for non-probabilistic labels, $q_t.P = \varnothing$, i.e., no probabilistic information is available from H in order for the attacker to differentiate between the states $s_i \in q_t.L$. Also, since non-probabilistic labels represent user visits to CRs, they form the only label type that leaks information about visits to sensitive places. Thus proper handling of non-probabilistic labels is crucial during the training phase for our mobility models in order to ensure effectiveness of subsequent attacks.

We now present our training algorithm which exploits the existing methods of supervised and unsupervised learning of HMM parameters in order to handle the mix labeling in our data.

4.3 Training algorithm for Ω

The backbone of our training algorithm comes from the well known Baum-Welch ($\mathcal{BW}$) algorithm for un-supervised training of HMMs [12] which assumes that state-labels are completely missing, i.e., $\mathbb{Q} = \varnothing$. The intention behind our use of the $\mathcal{BW}$ algorithm is to accommodate any unlabeled time-slots in the location history data as discussed above. Additionally, to incorporate the available labeled data into the training process, i.e., probabilistic state-labels and non-probabilistic state-labels (CRs), we adapt this $\mathcal{BW}$ backbone at two important stages of the underlying algorithm as discussed below in detail.

4.3.1 Stage 1: Defining an initial model. The $\mathcal{BW}$ algorithm only guarantees a locally optimal estimate of the HMM parameters. In order to converge to the globally optimal estimates, a good initial model Ω_{init} is required to begin the training process. To compute Ω_{init}, we exploit the available state label information by switching to the supervised learning method of *maximum likelihood* or $\mathcal{ML}$ parameter estimation (see line 2 of Algorithm 1). Although ML estimation represents the other extreme compared to the $\mathcal{BW}$ estimation by using the labeled data only, it gives us the opportunity to anchor the initial model Ω_{init} around factual data present in H.

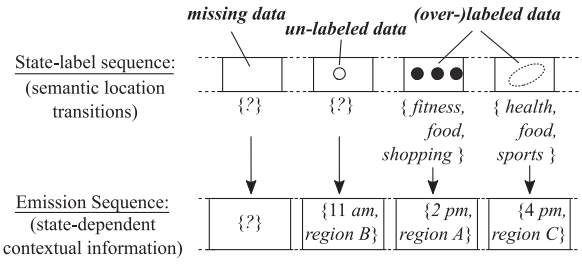

Figure 5: Mapping Location History to discrete time-slots per day.

The $\mathcal{ML}$ estimation works by counting the occurrences of inter-state transitions as well as per-state emissions and normalizing these counts with total occurrences of the relevant states. To accommodate label probabilities, we consider expected counts as shown in Eq. 5 to 7. For example, the transition probability a_{ij} is determined by determining the expected number of transitions from s_i to s_j divided by total number of expected transitions originating from s_i.

$$\pi_i = \frac{\left(\sum_{\forall q_1} P(q_1{=}i)\right) \quad +\omega}{\left(\sum_{k\in[1,m]} \sum_{\forall q_1} P(q_1{=}k)\right)+\omega\cdot m} \tag{5}$$

$$a_{ij} = \frac{\left(\sum_{\forall q_t, q_{t+1}} P(q_t{=}i \rightarrow q_{t+1}{=}j)\right) \quad +\omega}{\left(\sum_{k\in[1,m]} \sum_{\forall q_t, q_{t+1}} P(q_t{=}i \rightarrow q_{t+1}{=}k)\right)+\omega\cdot m} \tag{6}$$

$$b_i(v_j) = \frac{\left(\sum_{\forall q_t} P(q_t{=}i \wedge o_t{=}v_j)\right) \quad +\omega}{\left(\sum_{k\in[1,n]} \sum_{\forall q_t} P(q_t{=}i \wedge o_t{=}v_k)\right)+\omega\cdot n} \tag{7}$$

While computation of π_i is straightforward, the probability $P(q_t = i \rightarrow q_{t+1} = j)$ for computing a_{ij} is determined by multiplying label probabilities of s_i and s_j in consecutive time-slots. Since label probabilities $q_t.P$ at any time t sum to 1, this multiplication ensures that the sum of such probabilistic transitions counts for all combinations of the state-labels between any two given time-slots sums to an overall 1 transition count thus ensuring correctness of the estimate. Similarly for $b_i(v_j)$, the probability $P(q_t = i \wedge o_t = v_j)$ is the label-based probability of s_i given $o_t = v_j$ is observed.

Moreover, to avoid over-fitting Ω_{init} to the labeled data and to incorporate uncertainty represented by the unconsidered un-labeled data, we apply additive Laplace smoothing to these estimates by adding pseudo counts ω to the observed expected counts. For estimation of π_i and a_{ij}, we estimate ω as a function of the proportion of slots with missing data. The intuition is to have more smoothing if large proportions of data was missing. For estimation of $b_i(v_j)$, we conventionally set $\omega = 1$ since o_t is always available for any labeled time-slots q_t.

4.3.2 Stage 2: Iterative Expectation-Maximization. Coming back to the backbone $\mathcal{BW}$-based training, the next step is to iteratively improve the $\mathcal{ML}$-based initial model Ω_{init} by training in an unsupervised manner while only using the emission sequences $\mathbb{O}$. We first briefly describe the standard steps of this iterative procedure and then describe how we incorporate the available state-label information during the unsupervised training.

In each iteration of the $\mathcal{BW}$ training as detailed in [12], a new model Ω_{new} is determined based on the existing model Ω (initialized to Ω_{init}) in two steps as shown in Algorithm 1. In the first step called *expectation* (see line 8), the main goal is to determine the conditional probabilities $\gamma_t(i) = P(q_t = i|O_k, \Omega)$ of each state at each time given the corresponding emission sequence $O_k \in \mathbb{O}$ and the current model Ω. To achieve this in an efficient way over all possible paths of the hidden states for the observed emission sequence O_k, the well-known forward-backward ($\mathcal{FB}$) algorithm is used. This algorithm expresses $\gamma_t(i)$ in terms of two auxiliary

variables, called the *forward* $\alpha_t(i)$ and the *backward* $\beta_t(i)$ variables, as stated below.

$$\gamma_t(i) = \frac{\overbrace{P(q_t = i|o_1, ..., o_{t-1}, \Omega)}^{\alpha_t(i)} \overbrace{P(o_{t+1}, ..., o_T|q_t = i, \Omega)}^{\beta_t(i)}}{\sum_{j\in[1,m]} P(q_t = j|o_1, ..., o_{t-1}, \Omega)P(o_{t+1}, ..., o_T|q_t = j, \Omega)} \tag{8}$$

Moreover, the $\mathcal{FB}$ algorithm recursively computes $\alpha_t(i)$ and $\beta_t(i)$ using dynamic programming principles (see [12] for details).

$$\alpha_t(i) = \sum_{j\in[1,m]} \alpha_{t-1}(j)a_{ji}b_i(o_t) \tag{9}$$

$$\beta_t(i) = \sum_{j\in[1,m]} a_{ij}b_j(o_{t+1})\beta_{t+1}(j) \tag{10}$$

The second step of the $\mathcal{BW}$ algorithm called *maximization* (see line 10 of Algorithm 1) computes the $\mathcal{ML}$ estimates for the parameters of the new model Ω_{new} based on $\alpha_t(i)$ and $\beta_t(i)$ in a similar fashion to Eq. 5, 6, and 7. This iterative process continues until the parameters of the new and current models are almost similar (convergence).

In order to add state-label information during the iterations of the $\mathcal{BW}$ algorithm, we note that if the forward variable $\alpha_t(i)$ is zero for any given state s_i, it implies $\gamma_t(i) = 0$ from Eq. 8 thus ensuring that state s_i does not occur at time t. One of the factors that may result in $\alpha_t(i) = 0$ is a zero value of the emission probability $b_i(o_t)$ (see Eq. 9) which means that the emission o_t could not have been generated by state s_i. Thus emission probabilities $b_i(o_t)$ can be exploited to control the set of states that are allowed to occur at time t. We modify $b_i(o_t)$ as a way of incorporating state-label knowledge into the HMM by pre-processing them to reflect the set of possible states in lines 6 of Algorithm 1.

Thus in our approach, we define a function χ that generates the modified emission probabilities $\widehat{b}_i(o_t) \in \widehat{B}$ as follows.

$$\widehat{b}_i(o_t) = \begin{cases} 1 & \text{if } \{q_t, o_t\} = \{\varnothing, \varnothing\}, \\ b_i(o_t) & \text{if } q_t = \varnothing, \\ \delta \cdot P(i) + (1-\delta) \cdot b_i(o_t) & \text{if } s_i \in q_t.L \wedge q_t.P \neq \varnothing, \\ \delta + (1-\delta) \cdot b_i(o_t) & \text{if } s_i \in q_t.L \wedge q_t.P = \varnothing, \\ 0 & \text{if } s_i \notin q_t.L \end{cases} \tag{11}$$

Algorithm 1 Modified Baum-Welch Algorithm

1: Given $\{\mathbb{Q}, \mathbb{O}\}$
2: $\Omega_{new} = \Omega_{init} = \mathcal{ML}(\mathbb{Q}, \mathbb{O})$
3: **repeat**
4: $\quad\Omega = \Omega_{new}$
5: $\quad$**for all** $\{Q_k, O_k\}$ $\{\mathbb{Q}, \mathbb{O}\}$ **do**
6: $\quad\quad\widehat{B} = \forall_{(o_t\in O_k, q_t\in Q_k)} \quad \widehat{b}_i(o_t) \leftarrow \chi(o_t, q_t, \Omega.B)$
7: $\quad\quad$**Expectation:**
8: $\quad\quad\quad\forall_{t,i}(\alpha_t(i), \beta_t(i)) \leftarrow \mathcal{FB}(O_k, \Omega.\{\pi, A\}, \widehat{B})$
9: $\quad\quad$**Maximization:**
10: $\quad\quad\quad\Omega_{new}.\{\pi, A, B\} = \mathcal{ML}(\alpha_t(i), \beta_t(i))$
11: $\quad$**end for**
12: **until** $\Omega_{new}.\{\pi, A, B\} \neq \Omega.\{\pi, A, B\}$

We now provide an in-order explanation of the different cases handled by $\widehat{b}_i(o_t)$ in Eq. 11. Case 1: *Missing data where neither a state-label nor an emission is available.* Setting $\widehat{b}_i(o_t) = 1$ ensures that, by being equal, emission probabilities do not prioritize the occurrence of any hidden state as also discussed in [21]. Case 2: *Unlabeled data* which is the normal case in standard usage of HMMs. Thus the emission o_t is used to compute $\widehat{b}_i(o_t)$ as the actual emission probabilities $b_i(o_t)$ (see Eq. 4). Case 3: *Probabilistic state-labels.* Here, we use a weighting factor δ to control the contribution of $b_i(o_t)$ and $P(i)$ to the total emission probability $\widehat{b}_i(o_t)$. With $\delta = 1$, we assume that label-probabilities are highly reliable and, thus, emission probabilities $b_i(o_t)$ are ignored and vice versa. Case 4: *Non-probabilistic state labels* (e.g., for states within a CR), i.e., $q_t.P = \varnothing$. Thus a minimal emission probability $\delta > 0$ (e.g., $\delta > 0.5$) is artificially reserved for all states in $q_t.L$ due to our lack of knowledge about the actual state and slight prioritization between states is done only based on $b_i(o_t)$. Note that this case occurs only when the attacker uses the obfuscated location history H_{obf}. In Section 4.3.3, we will discuss how we can adapt this case to influence the learning of sensitive visiting patterns, i.e., to $s_i \in \mathbb{L}$. Case 5: This case ensures that if a state s_i is not part of the given label $q_t.L$, its probability is zero as discussed earlier, i.e., $\gamma_t(i) = 0$.

Note that while the modified emission probabilities $\widehat{b}_i(o_t)$ can fully control the non-occurrence of states at t, i.e., $\gamma_t(i) = 0$ if $\widehat{b}_i(o_t) = 0$, they cannot enforce this equivalence relationship for non-zero values, i.e., for $\widehat{b}_i(o_t) > 0$ which occurs when $q_t.P(i) > 0$. This is because the value of $\gamma_t(i)$ (see Eq. 8) also depends on other factors apart from $\widehat{b}_i(o_t)$. Hence for non-zero $\gamma_t(i)$, our method represents a best-effort solution to weight state-probabilities according to observed label probabilities.

Apart from the above adaptations, we modify the maximization step of the $\mathcal{BW}$ algorithm to compute independent emission matrices B_k for each of the D emitted variables. Overall, our adaptations result in a guided learning of the parameters by means of modified emission probabilities $\widehat{B_k}$ that are biased to reflect the partially known state-labels. Note that if a single state-label is known for every time-slot, our modified algorithm naturally converges to the case of fully supervised learning with ML estimates.

4.3.3 Learning sensitive patterns from CRs in H_{obf}. Note that for the case of non-probabilistic labels representing CRs in Eq. 11 (case 4), the attacker has no knowledge about the actually visited location. Lets assume that the attacker observes the set $\mathbb{CR}_{train}$ of CRs within H_{obf} during the training phase where a fraction λ of these represent actually sensitive visits. If $\lambda \approx 0$, this implies that sensitive trips are rare and therefore, in general, are hard to learn about as well as to predict as they do not occur in the set $\mathbb{CR}_{train}$ very often. Thus our policy of assigning a minimum probability of, for instance, $\delta = 0.5$ (see Eq. 11 case 4) to each location $s_i \in CR.S$ including the sensitive location *incorrectly* prioritizes learning of sensitive visits even though they occur rarely in this case. Thus such a trained mobility model Ω may predict much more sensitive visits than the actual ones when $\lambda \approx 0$ (or $\delta > \lambda$ in general) resulting in a lower attack precision.

On the contrary, if $\lambda \approx 1$, this implies that almost all trips represented by the set $\mathbb{CR}_{train}$ are actually sensitive. Thus by assigning

a value of δ in the range $0 < \delta < \lambda$ in this case, say $\delta = 0.5$, also to the non-sensitive locations inside the CRs in $\mathbb{CR}_{train}$, our algorithm can unfairly reduce the chances of learning user visiting patterns to the sensitive location (since it is frequently visited in this case), thus possibly resulting in the lower fractions of total sensitive trips (in the prediction phase) that can be detected, i.e., lower recall.

To maximize the success of the attack in terms of precision and recall while not knowing λ, we propose to set $\delta = 1$ for the sensitive locations only and $\delta = 0$ for the non-sensitive locations inside $\mathbb{CR}_{train}$ for all users. By doing so, we assume that every CR in $\mathbb{CR}_{train}$ represents a sensitive visit. The impact of this assumption when the actual value of $\lambda \approx 1$ is that the attack correctly assumes that all CRs represent sensitive visits. Thus, we may learn the best mobility model Ω equivalent to the model learned from the actual location history H_{act}. In the other case when $\lambda \approx 0$ actually, our assumption can likely result in false detections of sensitive visits. However, these false detections also occur when we assign an equal value of δ for all $s_i \in CR.S$ as discussed above. We will further substantiate our choice of permanently setting $\delta = 1$ only for the sensitive locations (and only during the training phase) in Section 5 when we evaluate our attack models.

4.4 Predicting locations inside CRs

After obtaining a trained HMM Ω for a user, we predict his location inside a previously unseen CR using recent observations that occur on the same day. Thus we assume that the obfuscated state-label sequence Q_{new} and its corresponding emission sequence O_{new} for the particular day are available. Note that a single day may contain multiple CRs.

To predict the user location inside the CR_t, we compute the state-label probabilities $\gamma_t(i)$ based on $\alpha_t(i)$ and $\beta_t(i)$ as determined by the $\mathcal{FB}$ algorithm. Note again that we use modified emission probabilities $\widehat{B_k}$ (see Eq. 11) in $\mathcal{FB}$ to take advantage of available state-label information in Q. Hence our method of modifying the emission probabilities $\widehat{B}$ also benefits us in the prediction phase by incorporating the available state-label information. Furthermore, for determining the modified emission probabilities for CRs (non-probabilistic state-labels, cf. case 4 of Eq. 11), we weight the actual emission probabilities higher by setting $\delta \approx 0$ since our model Ω has already learned state-dependent emission behavior during the training phase. We can thus predict the user's semantic location inside CR_t as $\widehat{s_t} = \text{argmax}_i\ \gamma_t(i)\ \forall\ i \in CR_t.S$.

5 EVALUATIONS

In this section, we evaluate the strength of our attacks against existing SCMs using real-world data.

5.1 The Dataset and its Pre-processing

To test our attacks, we needed long-term real-world data of user location histories enriched with semantic location information about visited as well as neighboring real-world venues. To this end, we crawled Twitter's public feed [2] from November 2015 to November 2016 for geo-tagged tweets. For each of these tweets, we acquire the semantic information about the venues from Foursquare [1]. Overall, our dataset contains 1,081,280 tweets from 80,377 users. We cleaned this data to rule out users with low check-in frequency (less than 2 per day), unavailable semantic information (from Foursquare), or low number of recorded days (less than 60). We further cluster nearby check-ins in short time-intervals (e.g., within 5 minutes) into single visits or stay-points (that form over-labeled time-slots in Fig. 5). Finally, we filter users with their stay-point frequency per day (>= 0.5) leaving us with 278 users with a mean recording period of 246 days ($\approx$ 8 months) and an overall total of 284,472 useful check-ins/tweets.

Overall, we generalize user check-ins into 17 high level semantic classes including *food, shopping, health,* etc., using Foursquare's category hierarchy [1]. To better understand the vulnerabilities of the SCM ψ against our attacks, we conduct different experiments by choosing a single sensitive location for each user that is visited in the following manner, i.e., (1) once a month, (2) once a week, (3) or on a daily basis. After selecting the sensitive locations of the users for a given experiment, we equally divide the user location data into a *training* part, i.e., the location history H for learning our attack models, and a *testing* part denoted R for validating our models. Moreover, we implement and apply SCM ψ (cf. Section 2.2) on H and R with various values of the obfuscation level $l \in [2, 8]$, i.e., the semantic diversity, to obtain the obfuscated versions of training and test data, i.e., H^l_{obf} and R^l_{obf}. Next, we map the training data ($\{H_{act}, H^l_{obf}\}$) and test data ($\{R^l_{obf}\}$) to $T = 12$ time-slots per-day (cf. Section 4.2). In our case, the multivariate emissions o_t representing contextual information of user trips comprise of three variables: (1) time-slot of day, (2) a binary variable indicating type of day, i.e., a weekday or a weekend, and (3) geo-location-id which represents the spatial location of a user among his finite number of popular regions of presence determined by applying k-means clustering to his location history H. After the training phase (cf. Section 4.3), we obtain mobility models of two types, i.e., Ω_{act} from H_{act} and Ω^l_{obf} from H^l_{obf}. Next we report the concrete results of our attacks.

5.2 Attack Precision and SCM Effectiveness

We first evaluate the proportion of users as well as that of the total sensitive visits for all users that are successfully attacked by our mobility models at a certain precision $Pr(g_{attack})$ (cf. Eq. 2). The results are shown in Fig. 6 for differently frequented sensitive locations, i.e, on a monthly, weekly, or a daily basis. For sensitive locations that are visited daily, as an example, our attacks affect 50% of the users (see the left column, bottom figure) and 60% of overall cloaked visits (right column, bottom figure) at an attack precision of 80% when the obfuscation level $l = 2$. In general, increasing l to higher values results in lower numbers of both users and sensitive visits that are affected at a given precision. However, even at $l = 8$, our attacks still affect 20% of users as well as 20% sensitive visits with 80% attack precision. Note also that, overall, the obfuscated models (dotted lines) and the corresponding accurate models (solid lines) (for the same values of l-diversity) affect approximately equal proportions of users and sensitive trips. These trends, with some degradation that are focused mostly on the lower ranges of the attack precision, are generally the same for the weekly and monthly visited sensitive locations (top two rows of the figure). This gradual degradation is intuitively explained by the fact that our attack

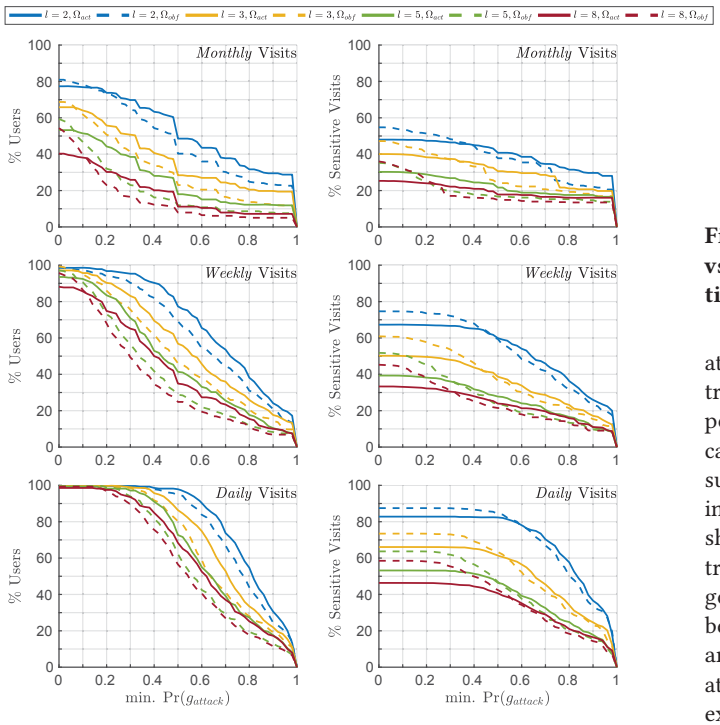

Figure 6: Attack precision vs. percentages of affected users (left column) and their sensitive trips (right column).

models learn less about these sensitive locations due to their less occurrence (in the weekly and monthly visited cases respectively).

In Fig. 7 on the left, we plot the effectiveness Δ_ψ of SCM ψ as defined in Section 3.3.3 (cf. Eq. 3) against values of l-diversity while combining attack precision results for the experiments with differently frequented sensitive locations. In general, the median value of approximately zero as well as the small inter-quartile range of Δ_ψ (up to ≈ 0.2) indicates that the precision loss for the attacker due to the obfuscated location history H^l_{obf} (and in comparison to H_{act}) is not very significant even with higher values of l such as 8. Thus, obfuscation of location history with SCM ψ does not create much of a hindrance for the attacker in terms of the attack precision he can achieve for a given user.

To understand the success of our attacks, on the right hand side in Fig. 7, we plot the distribution of attack precision achieved for the users in our dataset against λ, *which represents the fraction of visits of these users that were sensitive among all of their published CRs.* Note that this plot considers the worst case obfuscation level, i.e., $l = 8$, and graphs the distribution of precisions for Ω_{act} and Ω_{obf} side by side for comparison (left and right respectively). Clearly, higher values of λ proportionally increase attack precision for both mobility models. Moreover, the difference between the distribution of attack precision based on Ω_{act} and Ω_{obf} is not very significant. Note that this high linear coupling between attack precision and λ (Pearson's correlation of 0.9) as well as the almost equivalent attack performance of Ω_{act} and Ω_{obf} helps validate our choice of setting the minimum probability $\delta = 1$ for the users' sensitive locations during the training of our algorithms as discussed in Section 4.3.3.

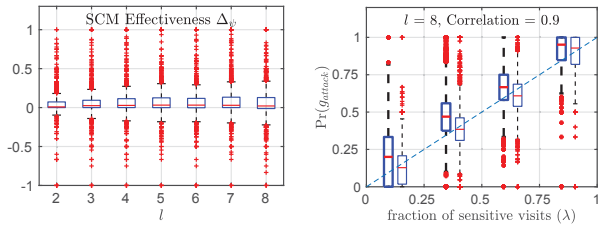

Figure 7: Left) Effectiveness of SCM ψ denoted Δ_ψ (cf. Eq. 3) vs. l-diversity. Right) Attack precision vs. fraction of sensitive visits.

5.2.1 Predicting high Precision using Ω. Since computation of attack precision $Pr(g_{attack})$ requires the knowledge of the ground truth about the observed CRs (cf. Eq. 2), an attacker who only possesses the obfuscated location history H_{obf} of the target user cannot compute $Pr(g_{attack})$ and thus may be unaware of his attack success. Due to space limitations, we briefly mention here that our initial experiments using off-the-shelf machine learning algorithms showed that such an attacker could exploit some features of the trained mobility model Ω_{obf} to estimate his attack precision with good accuracy. For example when classifying whether a user can be attacked with a precision of higher than 80%, we could train an SVM classifier (Gaussian kernel, 10-fold cross validation) to attain a prediction accuracy of 77% (with a recall of 24%). Our initial experiments indicated that this accuracy and recall can be further improved by putting together different machine learning models into a multi-stage classifier. However, a detailed analysis in this regard is subject to future work. The features of Ω_{obf} used in our machine-learning experiments were: (1) proportion of CRs that are predicted to represent sensitive visits, (2) the mean probability assigned to sensitive locations in case of their prediction and (3) their non-prediction. (4) the proportion of time-slots that include CRs.

6 POSSIBLE DESIGN IMPROVEMENTS FOR FUTURE SCMS

The above evaluations highlight that attack precision, irrespective of the available location history type, i.e., H_{act} or H_{obf}, is highly correlated with the parameter λ (proportion of sensitive visits inside CRs) as shown on the right in Fig. 7. This figure also highlights that existing algorithms do not put any upper bound on the value of λ as part of their design. This design limitation can be easily understood based on the fact that the location update algorithm of existing SCMs (cf. Section 2.2.1) chooses to publish a CR based on user visits inside the CR and not based on their visit frequencies to their sensitive place inside the CR.

Hence, a fundamental design improvement for future SCMs could be to directly consider λ as part of their attack metric $f_{att}(CR)$. This may imply that the algorithm should have some historical knowledge about users' visit frequencies to their sensitive places. We also note that binding the value of λ may be difficult when visit frequencies to sensitive places are too high. This is because, in this case, an even higher total number of visits inside CRs is required to lower the value of λ which may not be practical given the users' mobility preferences. Therefore, it may be reasonable for new SCMs to decide in an online fashion about when to suppress, or even perturb (by adding noise), some of the sensitive visits of

users to lower the requirement on the total number of visits inside the CRs that must be made in order to meet the desired value of λ.

7 RELATED WORK

We now discuss the various works that are related to our contributions in this paper.

Location privacy mechanisms. Apart from the SCMs discussed in Section 2, we now mention a few recently proposed location privacy mechanisms that *partly* avoid location history based attacks. For instance, the mechanisms proposed in [16, 17] model spatial movements of users as Markov chains to generate perturbed (noise added) output locations instead of obfuscation regions. However, the attacker and the privacy models assumed in these mechanisms lack the consideration of semantic information. Thus these works are uninteresting for our attacks as they do not protect semantic location information. Another privacy mechanism proposed in [8] avoids the inference of sensitive user states by means of using *suppression*, and not cloaking, as the privacy algorithm. Thus while it avoids leakage of any contextual information about the sensitive user locations, it also is inapplicable to various scenarios requiring publication of approximate location information, as offered by CRs, and, thus, is also not of interest for our attacks.

Potential attacks on Semantic Location privacy. Works such as [10] propose methods to assign semantic location labels to user visits using machine-learning based classification algorithms. Another relevant work proposes an HMM based semantic mobility model for location-recommendations in [19]. In general, these works rely on accurate as well as fully-labeled data from a population of users for training their models. Hence, these potential attack models are not applicable in our case where training is done on accurate or obfuscated user data with missing labels (cf. Section 4.1) and is focused on a user's own location history (rather than from user-population). In [14], the authors show that typical visiting frequencies of users to various semantic locations are significantly different from each other and thus treat them as a privacy-sensitive attributes in their privacy algorithm. Taking their cue, we assume that visiting frequency information can possibly augment our attacks by narrowing down the candidate set of visited locations inside a CR based on user's visit frequency to the CR.

HMM training with partially-labeled data. Previous works such as [15] propose to accommodate the partial state-label information by modifying the maximization step in the Baum-Welch algorithm. Since there is no maximization step in the prediction phase (after training), these approaches are only useful for those tasks where only the training data is partially-labeled. In our case, we modify the emission probabilities to account for state-label information during the training phase as well as the later prediction phase. Our method additionally handles missing as well as over-labeled states unlike existing works.

8 CONCLUSION

In this paper, we have concretely exposed the vulnerability of existing semantic location privacy mechanisms against attacks based on mobility prediction. To attack the already obfuscated/protected location data of users, we have developed a semantic mobility prediction model that incorporates users' movement patterns (represented by their location history information) as Hidden Markov Models. We have shown that this mobility prediction model can be trained from the actual as well as obfuscated location history data and is effective in de-obfuscating privacy-sensitive user visits. Moreover, we have analyzed the success of our attacks and suggested guidelines for robust design of future location privacy mechanisms.

ACKNOWLEDGMENTS

This work is a part of project *PriLoc* (Privacy-aware Location Management) of the University of Stuttgart funded by German Research Foundation (DFG) grant RO 1086/15-2.

REFERENCES

[1] *foursquare for Developers.* from https://developer.foursquare.com/.
[2] *REST APIs - Twitter Developers.* from https://dev.twitter.com/rest/public.
[3] C. A. Ardagna, M. Cremonini, E. Damiani, S. De Capitani di Vimercati, and P. Samarati. 2007. *Location Privacy Protection Through Obfuscation-Based Techniques.* Springer Berlin Heidelberg, Berlin, Heidelberg, 47–60. https://doi.org/10.1007/978-3-540-73538-0_4
[4] Sunny Consolvo, Ian E Smith, Tara Matthews, Anthony LaMarca, Jason Tabert, and Pauline Powledge. 2005. Location disclosure to social relations: why, when, & what people want to share. In *Proceedings of the SIGCHI conference on Human factors in computing systems.* ACM, 81–90.
[5] Maria L. Damiani, Elisa Bertino, and Claudio Silvestri. 2010. The PROBE Framework for the Personalized Cloaking of Private Locations. *Trans. Data Privacy* 3, 2 (2010), 123–148.
[6] Matt Duckham and Lars Kulik. 2005. A Formal Model of Obfuscation and Negotiation for Location Privacy. In *Pervasive Comp.* LNCS, Vol. 3468. Springer Berlin Heidelberg, 152–170.
[7] Gabriel Ghinita, Maria L. Damiani, Claudio Silvestri, and Elisa Bertino. 2009. Preventing Velocity-based Linkage Attacks in Location-aware Applications. In *Proc. of the 17th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems (GIS).* ACM, New York, NY, USA, 246–255.
[8] Michaela Götz, Suman Nath, and Johannes Gehrke. 2012. MaskIt: Privately Releasing User Context Streams for Personalized Mobile Applications. In *Proc. of SIGMOD.* ACM, 289–300.
[9] Dimitrios Katsaros and Yannis Manolopoulos. 2009. Prediction in wireless networks by Markov chains. *IEEE Wireless Communications* 16, 2 (2009).
[10] John Krumm and Dany Rouhana. 2013. Placer: Semantic Place Labels from Diary Data. In *Proc. of UbiComp.* ACM, 163–172.
[11] Byoungyoung Lee, Jinoh Oh, Hwanjo Yu, and Jong Kim. 2011. Protecting Location Privacy Using Location Semantics. In *Proc. of KDD.* ACM, 1289–1297.
[12] Lawrence R. Rabiner. 1990. Readings in Speech Recognition. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, Chapter : A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, 267–296.
[13] Zohaib Riaz, Frank Dürr, and Kurt Rothermel. 2015. Optimized location update protocols for secure and efficient position sharing. In *Proc. of Int. Conf. on Networked Systems (NetSys).* 1–8.
[14] Zohaib Riaz, Frank Dürr, and Kurt Rothermel. 2016. On the Privacy of Frequently Visited User Locations. In *Proc. of Int. Conf. on Mobile Data Management (MDM).* IEEE, 282–291.
[15] Tobias Scheffer, Christian Decomain, and Stefan Wrobel. 2001. Active hidden markov models for information extraction. In *Int. Symposium on Intelligent Data Analysis.* Springer, 309–318.
[16] George Theodorakopoulos, Reza Shokri, Carmela Troncoso, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. 2014. Prolonging the Hide-and-Seek Game: Optimal Trajectory Privacy for Location-Based Services. In *Proc. of WPES.* ACM, 73–82.
[17] Yonghui Xiao and Li Xiong. 2015. Protecting Locations with Differential Privacy Under Temporal Correlations. In *Proc. of the 22nd ACM SIGSAC Conf. on Computer and Communications Security (CCS).* ACM, New York, NY, USA, 1298–1309.
[18] Mingqiang Xue, Panos Kalnis, and HungKeng Pung. 2009. Location Diversity: Enhanced Privacy Protection in Location Based Services. In *Location and Context Awareness.* LNCS, Vol. 5561. Springer Berlin Heidelberg, 70–87.
[19] Jihang Ye, Zhe Zhu, and Hong Cheng. 2013. What's your next move: User activity prediction in location-based social networks. In *Proc. of SIAM Int. Conf. on Data Mining.* SIAM, 171–179.
[20] Emre Yigitoglu, Maria L. Damiani, Osman Abul, and Claudio Silvestri. 2012. Privacy-Preserving Sharing of Sensitive Semantic Locations under Road-Network Constraints. In *Proc. of MDM.* 186–195.
[21] Shun-Zheng Yu and Hisashi Kobayashi. 2003. A Hidden semi-Markov Model with Missing Data and Multiple Observation Sequences for Mobility Tracking. *Signal Process.* 83, 2 (Feb. 2003), 235–250.