# Protecting Against Inference Attacks on Co-location Data

Ritesh Ahuja Gabriel Ghinita Nithin Krishna Cyrus Shahabi
University of Southern California
Los Angeles, U.S.A
{riteshah,ghinita,ottiling,shahabi}@usc.edu

***Abstract*—The proliferation of location-centric applications results in massive amounts of individual location data that can benefit domains such as transportation, urban planning, etc. However, sensitive personal data can be derived from location datasets. In particular, *co-location* of users can disclose one's social connections, intimate partners, business associates, etc. We derive a powerful inference attack that makes extensive use of background knowledge in order to expose an individual's co-locations. We also show that existing techniques for location protection, which do not focus specifically on co-locations, distort data excessively, resulting in sanitized datasets with poor utility. We propose three privacy mechanisms that are customized for co-locations, and provide various trade-offs in terms of user privacy and data utility. Our extensive experimental evaluation on a real geo-social network dataset shows that the proposed approaches achieve good data utility and do a good job of protecting against discovery of co-locations, even when confronted with a powerful adversary.**

## I. Introduction

The widespread availability of mobile devices with accurate positioning capabilities (e.g., GPS, Wi-Fi localization) led to the emergence of a wide array of popular location-centric applications, e.g., location-based services, geo-social networks, ride sharing, etc. As a by-product of these applications, large amounts of individual location data are collected by service providers. Sharing such data benefits research on topics such as optimizing traffic, improving transportation efficiency, or studying disease spreading patterns. However, directly sharing location data also introduces serious privacy concerns, as an adversary can use locations to derive sensitive details about an individual's health status, social connections or alternative lifestyle. The last decade witnessed a vast amount of research on the topic of location protection [1], [12], [13], and a commonly-used approach is to employ a *sanitization* process on the collected data prior to sharing it with untrusted entities, as illustrated in Figure 1.

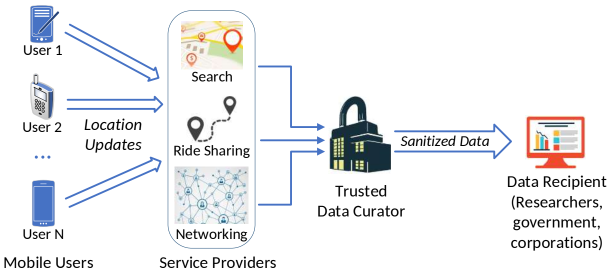


Figure 1: System Model

Our focus is the privacy leakage resulting from the information that two or more persons were situated in close proximity to each other at a certain time. For example, if two people are inside the same restaurant at roughly the same time, it may imply that a social connection between them exists [10]. Based on this observation, several studies successfully derive information about links in a social network from the locations of individual check-in records [6], [10], [19]. More recent studies take it a step further, and show that it is possible to predict *future* individual locations from the inferred social links [8], [27], thereby increasing considerably the amount of privacy breaches.

We define *co-location* as the presence of two users in the same geographical location at approximately the same time. Co-location attacks can be very powerful, as illustrated by recent revelations of the NSA's *PRISM* and *Co-Traveler* programs which make use of co-location data collected from tech companies and cellular networks to track down terrorism suspects [11]. While revealing co-location data in clear to government agencies for national security purposes may be acceptable, it is certainly very dangerous to disclose co-locations by directly releasing data to private companies or to the general public. Effective protection techniques are needed to sanitize co-location information before release.

We emphasize that, our focus is to protect *co-locations* derived from location check-in records, but *not necessarily* the check-ins themselves. The rationale behind this approach is that many users are willing to disclose their check-ins publicly, e.g., through posts on their Facebook or Twitter profiles, in return for various incentives. Furthermore, attempting to protect individual check-ins entirely may result in published datasets that have very poor utility, since the amount of noise added by the sanitization process becomes excessive. As we show in Section V, employing a protection mechanism designed for individual locations such as geo-indistinguishability [1] may render the released data useless.

Existing work has serious limitations when protecting co-locations. The work in [3] evaluates the effectiveness of basic obfuscation mechanisms in protecting against friendship inferences through co-locations [6], [7], [10], [19]. However, the solution *(i)* is limited to applications that discover social ties by extracting relevant features from co-locations, whereas the more general problem of protecting against all types of inferences (reachability [21], social influence [18]) is not addressed; *(ii)* does not account for the background knowledge of the adversary, or the information leaked by the obfuscation mechanism; and *(iii)* provides an inaccurate estimation of co-location privacy of users since it does not account for reverse-engineering the noise introduced by obfuscation. In contrast, our attack model focuses on reconstructing an accurate representation of the original data from the sanitized release by tracing users back to their potential co-locating partners.

Our specific contributions are:

1) We formulate a powerful attack on co-location privacy that makes extensive use of prior information on both check-in history of individual users, as well as past (public) co-locations with other users.
2) We introduce three protection methods that specifically protect co-locations. First, we adapt the Gaussian noise approach [2], [14], [17], [23] to co-location protection. However, we observe that such a simple method suffers from a large variance in the protection levels provided, due to sparseness of co-locating neighbors in real-world datasets. To address this limitation, we introduce an *adaptive* obfuscation technique that adjusts the magnitude of noise based on the distribution of neighbor locations around a user in both spatial and temporal dimensions. This way, we improve both the privacy of users' co-location data and the utility of the data. Finally, we consider a syntactic notion of co-location privacy that groups together multiple co-locations in order to confuse the attacker.
3) We empirically evaluate our approaches on a real geo-social network dataset. Results show that generic protection methods (e.g., geo-indistinguishability) are not suitable for co-location protection, as they introduce excessive distortion, rendering the data useless. The proposed approaches, on the other hand, achieve interesting trade-offs between privacy and data utility.

Sec. II formally defines the studied problem. Sec. III presents the details of the proposed inference attack on co-locations. Protection techniques are introduces in Sec. IV, followed by an extensive experimental evaluation in Sec. V. Sec. VI surveys related work, followed by conclusions and directions for future work in Sec. VII.

## II. Preliminaries

**Problem formulation.** Consider a set of users $U = \{u_1, u_2, ..., u_N\}$ that participate in location-centric applications (e.g., location-based services, geo-social networks, etc). Users report their locations to the application service provider in the form of *check-ins*, consisting of user identifier, location and time triplets $\langle u, l, t\rangle$. We consider discrete locations which are part of a location universe $L = \{l_1, l_2, ..., l_P\}$. The historical record of check-ins of a user $i$ is represented as a vector $C_i = \{c_i^1, c_i^2, ..., c_i^j\}$. The set of check-ins of all the users in the system is denoted as $C = \{C_1 \cup C_2 \cup ... \cup C_N\}$. We use the notations $c.l$, $c.t$ and $c.u$ to denote the location, time and user id of a check-in $c$.

A *co-location* is formally defined as follows:

*Definition 1:* Given a spatial threshold $\Delta_s$ and a temporal threshold $\Delta_t$, the check-ins of users $u$ and $v$ are said to be co-located if they are within $\Delta_s$ spatial distance and $\Delta_t$ temporal distance of each other.

The distance function used to characterize a co-location is orthogonal to this study. In this paper, we use the Euclidean distance for the spatial dimension (i.e., $||c_u^i.l, c_v^j.l|| \leq \Delta_s$) and absolute difference for the temporal dimension (i.e., $|c_u^i.t - c_v^j.t| \leq \Delta_t$). The precise values of $\Delta_t$ and $\Delta_s$ are application-dependent (for example, $\Delta_t$ may range from less than an hour in the case of discovering social ties [7] to days in the the case of contact tracing for disease control [16]). The set of all co-locations is denoted by $CL = \{cl_1, cl_2, cl_3 \ldots cl_M\}$, wherein an element $cl_i$ of form $(c_u^i, c_v^j)$ denotes a co-location between check-ins $c_u^i$ and $c_v^j$.

We model co-location data as an undirected graph $G = (C, CL)$, referred to as *co-location network*, where a node $c_u^i \in C$ represents a user's check-in and an edge $cl = (c_u^i, c_v^j) \in CL$ indicates a co-location between $c_u^i, c_v^j \in C$. Before publishing a dataset of locations, the co-location network must be protected using a *co-location privacy mechanism*. We study the case when the co-location of two users is independent of other check-ins of those users. Following the distinction made by [22] between sporadic and continuous location reports, we consider the former, and assume no temporal correlations among a moving user's locations. This is reasonable in cases where check-ins of the same user have a sparse distribution over time, and thus can be considered independent. Other models exist in literature which consider that a spatio-temporal correlation exists between consecutive check-ins [24] (such as the high likelihood of traveling to home after work). These correlations can be exploited by an adversary to stage more powerful attacks [23], but they are outside our scope.

**Co-location privacy mechanisms.** The data publisher implements a privacy mechanism that obfuscates co-locations. Given $G = (C, CL)$, the mechanism uses a mapping function $\mathcal{F}$ to construct another graph $G' = (C', CL')$. Given a co-location $cl = (c_u^i, c_v^j)$, the purpose of a privacy mechanism is to apply distortion to the locations and timestamps of the constituent check-ins $c \in \{c_u^i, c_v^j\}$.

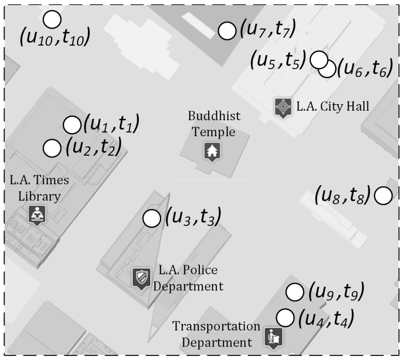


Figure 2: Running Example of a Co-location Network

Table I: Summary of Notations

| Notation | Definition |
|---|---|
| $C, CL$ | Set of check-ins $C$ and induced co-locations $CL$ |
| $G = (C, CL)$ | Co-location graph |
| $\Delta_s, \Delta_t$ | Spatial and Temporal distance thresholds |
| $G, G', RG$ | Actual, obfuscated and reconstructed graph |
| $ST_{dist}$ | Spatio-temporal distance |
| $MAX_S$ | Normalizing factor for Spatial distance |
| $MAX_T$ | Normalizing factor for Temporal distance |
| $QL$ | Quality Loss metric |
| $\|\|c_u^i.l, c_v^j.l\|\|$ | Spatial distance (i.e. $l_2$ norm) |
| $\|c_u^i.t, c_v^j.t\|$ | Temporal distance (i.e. $l_1$ norm) |
| $c.l$, $c.t$, $c.u$ | Location, timestamp and user id of a check-in $c$ |
| $IP, IR$ | Precision and recall of co-location inference |
| $\sigma_s, \sigma_t$ | Standard deviations of the spatial and temporal Gaussian distributions |
| $\epsilon$ | Privacy parameter of Geo-Indistinguishability |
| $k$ | Privacy parameter of Adaptive Perturbation |
| $b$ | Privacy parameter of Co-location Masking |

Figure 2 illustrates a simple co-location privacy mechanism. Two check-ins are co-located if they have the same location (e.g., library) and differ by no more than one time unit (i.e., $t_1 - t_2 \leq \delta_t$, but $t_4, t_9 > \delta_t$). Accordingly, there are two co-locations: $cl_1 = (u_1, u_2)$ and $cl_2 = (u_5, u_6)$. One approach is to move $u_3$'s check-in such that it appears to have co-located with $u_1$ and $u_2$. This would make the true co-location $cl_1$ appear indistinguishable from the now co-located pairs $(u_1, u_3)$ and $(u_2, u_3)$. Another approach is to add noise to the attributes of a check-in: e.g., moving check-in of user $u_2$ to the temple would essentially break the co-location $cl_1$ by violating the $\Delta_s$ and $\Delta_t$ constraints. Table I summarizes the notations used in the paper.

**Evaluation Metrics** Privacy-preserving mechanism inherently create distortion in the data to achieve protection. Hence, a trade-off emerges between protection strength and data utility after sanitization. Next, we define metrics to evaluate the privacy and utility of the considered approaches.

*Privacy Metric.* The adversary has access to the sanitized co-location graph $G' = (C', CL')$, and performs an inference attack to derive a *reconstructed graph* $RG = (RC, RCL)$. We quantify a user's co-location privacy as the adversary's *precision* (the fraction of correct co-location instances over all instances inferred), and *recall* (the fraction of correct co-locations retrieved by the adversary over the total count of co-locations in the original data):

$$\text{Inference Precision: } IP = \frac{|CL \cap RCL|}{|RCL|}$$

$$\text{Inference Recall: } IR = \frac{|CL \cap RCL|}{|CL|}$$

*Utility Metric.* We define *quality loss (QL)* as follows[1]:

*Definition 2:* For an obfuscated check-in $c_u^i$ of user $u$, *quality loss QL* is defined as the linear combination of spatial distance between the real check-in location and the reported one, and the temporal difference between the real timestamp and the reported one[2].

$$QL_u^i = \gamma \cdot \frac{||c_u^i.l, c_u^i.l'||}{MAX_S} + (1 - \gamma) \cdot \frac{|c_u^i.t, c_u^i.t'|}{MAX_T} \qquad (1)$$

where $\gamma$ is a weighing parameter which trades-off the magnitude of spatial distortion for temporal distortion. The value of $\gamma$ compensates for the difference in sensitivity of the spatial and temporal dimensions. For example, given a fixed value of quality loss, if $\gamma = 0.33$, then for the same absolute value of check-in spatial distortion, the temporal distortion applied is double compared to the case when $\gamma = 0.5$. Factors $MAX_S$ and $MAX_T$ normalize the spatial and temporal distances in the range [0,1]. These normalizing factors are typically set to the maximum spatial or temporal distortion that may be introduced to a co-location.

## III. Co-location Inference Attack

We consider a powerful adversary with access to both the sanitized co-location graph, as well as a significant amount of background knowledge. We assume the adversary has prior information about user mobility patterns (i.e., frequently visited locations), as well as co-location patterns (i.e., frequent co-locating partners). The adversary may have access to historical check-in data, and other public information about locations visited by each user (e.g., home, workplace) and about the identity of the users they meet.

The adversary's prior knowledge for *each* user consists of *(i)* a probability distribution of the user's location over the entire location universe $L$; and *(ii)* a probability distribution over all potential co-locating partners in set $U$. The adversary also knows the privacy-protection mechanism employed by the data publisher, and she can take advantage of the information leaked by the privacy mechanism to reduce her uncertainty about a user's true position and his co-locating partner. The adversary's goal is to conjointly exploit the mobility and co-location patterns between pairs of users, in order to localize them at certain points in space and time.

Let $O_u^{(l_o, t')}$ denote the *observed* check-in of user $u$ at location $l_o$ and time $t'$ (i.e., the check-in present in the sanitized data, to which the adversary has access). Likewise, let $A_u^{(l_a, t)}$ denote the *actual* check-in (i.e., before sanitization) of user $u$ at $l_a$ and time $t$. The privacy mechanism maps an actual check-in to an observed one as follows:

$$\mathcal{F}(l_a, t)(l_o, t') = \Pr\{O_u^{(l_o, t')} | A_u^{(l_a, t)}\} \qquad (2)$$

The obfuscation function $\mathcal{F}$ depends on the mechanism implementation and its parameters. Given the details of the mechanism, and the set of sanitized locations and co-locations, the adversary tries to reconstruct the actual set of co-locations by assigning each check-in of a user to its most probable co-located position and partner.

More precisely, the adversary's goal is to extract information from both the observed position of $u$'s check-in $c$,

[1] For a more accurate assessment of utility, in our experimental evaluation (Section V) we also consider the accuracy of answers for actual queries, in addition to the QL metric.

[2] The distances semantics depend on usage: in a road network, one may use Manhattan distance (i.e., $l_1$ norm).

and the observed positions of all other check-ins that could have co-located with $c$. The goal is to conjointly capture a user's check-in behavior and co-location patterns, with the intuition that a user is more likely to co-locate with users with whom he has a social connection (e.g., friendship, work relationship). Given an observed check-in position $(l_o, t_o)$ of user $u_1$, the adversary builds the posterior distribution $\Pr\{A_{u1}^{(l_j,t_j)}|O_{u1}^{(l_o,t_o)}, O_{u2}^{(l_p,t_p)}, ..., O_{uN}^{(l_q,t_q)}\}$ over positions of all other check-ins with whom user $u_1$ might have co-located. Each tuple $(l_j, t_j)$ takes discrete values representing the spatio-temporal positions of the check-ins of other users in the data. The posterior distribution gives the probability distribution over the check-in's actual co-location positions. The adversary assigns the check-in to a location sampled from the posterior. The set of reconstructed co-locations $RCL$ is returned as the set of all recovered co-locations.

While the posterior distribution described above gives a good estimate of the check-in's actual position, computing such a posterior is intractable in real-world datasets [23]. Thus, we slightly simplify the posterior to improve efficiency. Assume that when user $u$ is restored from a position $l_o, t'$ to $l_a, t$, he forms a co-location with user $v$ at $l_a, t$. The attacker estimates the posterior distribution $\Pr\{A_u^{(l_a,t)}|O_u^{(l_o,t')}, O_v^{(l_a,t)}\}$ of the user $u$'s actual location given $u$'s and his potential co-locating partner $v$'s observed check-in locations. The rest of the attack remains unchanged. The procedure is summarized in Algorithm 1.

**Algorithm 1** Adversary's Reconstruction Strategy

```
procedure BUILD POSTERIOR DISTRIBUTION($C$)
    $S \leftarrow \{\}$
    for $c_i(u, l_o, t')$ in $C'$ AND not in $S$ do
        for $c_j(v, l_a, t)$ in $C'$ AND not in $S$ do
            if $c_i.u \neq c_j.v$ then
                Compute $\Pr\{A_u^{(l_a,t)}|O_u^{(l_o,t')}, O_v^{(l_a,t)}\}$
        Assign $c_i$ to location sampled from posterior.
        Add both check-ins to set $S$
    return $RCL$
```

Next, we show how to compute the posterior distribution $\Pr\{A_u^{(l_a,t)}|O_u^{(l_o,t')}, O_v^{(l_a,t)}\}$. We simplify the equation using Bayes Theorem as follows:

$$\Pr\{A_u^{(l_a,t)}|O_u^{(l_o,t')}, O_v^{(l_a,t)}\} = \frac{\Pr\{A_u^{(l_a,t)}, O_v^{(l_a,t)}|O_u^{(l_o,t')}\}}{\Pr\{O_v^{(l_a,t)}|O_u^{(l_o,t')}\}} \tag{3}$$

To simplify presentation, we rewrite the equations without temporal notations; the reader may interpret that location notations $l_x$ in the superscript have a discrete value of time associated with it, as shown in the above equation. The newly obtained equation is:

$$\Pr\{A_u^{l_a}|O_u^{l_o}, O_v^{l_a}\} = \frac{\Pr\{A_u^{l_a}, O_v^{l_a}|O_u^{l_o}\}}{\Pr\{O_v^{l_a}|O_u^{l_o}\}} \tag{4}$$

$$= \frac{\Pr\{O_v^{l_a}|A_u^{l_a}, O_u^{l_o}\} \cdot \Pr\{A_u^{l_a}|O_u^{l_o}\}}{\Pr\{O_v^{l_a}|O_u^{l_o}\}} \tag{5}$$

Conditioned on event $A_u^{l_a}$ (actual location of $u$), observed events $O_v^{l_a}$ and $O_u^{l_o}$ are independent of each other. This happens because the considered sanitization mechanisms process check-ins individually. Hence, factor $\Pr\{O_v^{l_a}|A_u^{l_a}, O_u^{l_o}\}$ can be simplified to

$$\frac{\Pr\{O_v^{l_a} \cdot O_u^{l_o}|A_u^{l_a}\}}{\Pr\{O_u^{l_o}|A_u^{l_a}\}} = \frac{\Pr\{O_v^{l_a}|A_u^{l_a}\} \cdot \Pr\{O_u^{l_o}|A_u^{l_a}\}}{\Pr\{O_u^{l_o}|A_u^{l_a}\}} \tag{6}$$

Using Eq. (6), we can rewrite our objective Eq. (4) as:

$$\underbrace{\Pr\{A_u^{l_a}|O_u^{l_o}, O_v^{l_a}\}}_{\text{Objective}} = \frac{\overbrace{\Pr\{O_v^{l_a}|A_u^{l_a}\}}^{\text{III:}} \cdot \overbrace{\Pr\{A_u^{l_a}|O_u^{l_o}\}}^{\text{I:}}}{\underbrace{\Pr\{O_v^{l_a}|O_u^{l_o}\}}_{\text{II:}}} \tag{7}$$

Next, we illustrate how to compute each component highlighted in Eq. (7).

**I:** $\Pr\{A_u^{(l_a)}|O_u^{(l_o)}\}$ can be estimated using Bayes' theorem as:

$$\Pr\{A_u^{(l_a)}|O_u^{(l_o)}\} = \frac{\Pr\{O_u^{(l_o)}|A_u^{(l_a)}\} \cdot \Pr\{A_u^{(l_a)}\}}{\sum_{l_i \in L} \Pr\{O_u^{(l_o)}|A_u^{(l_i)}\} \cdot \Pr\{A_u^{(l_i)}\}} \tag{8}$$

where $\Pr\{A_u^{(l_i)}\}$ denotes the probability that a user $u$ checks-in at location $l_i$. This value is estimated using the historical aggregates of $u$'s most visited locations.

**II:** The denominator captures the total probability of the observed event. We can describe it using the law of total probability as $\Pr\{O_v^{l_a}|O_u^{l_o}\} = \sum_{j \in L} \Pr\{O_v^{l_a}|A_u^{l_j}\} \cdot \Pr\{A_u^{l_j}|O_u^{l_o}\}$.

**III:** Factor $\Pr\{O_v^{l_a}|A_u^{l_a}\}$ captures the influence of the location-embedded social structure. We rewrite it as:

$$\Pr\{O_v^{l_a}|A_u^{l_a}\} = \sum_{j \in L} \Pr\{O_v^{l_a}|A_v^{l_j}\} \cdot \Pr\{A_v^{l_j}|A_u^{l_a}\} \tag{9}$$

where $\Pr\{A_v^{l_j}|A_u^{l_a}\}$ captures the effects of the social structure, i.e., the check-in behavior of other users ($v \in U$) with respect to user $u$. This network effect may vary in different application settings, such as the inference of social links, in the measurement of social influence or in the contact tracing for disease control. Our framework enables adoption of ad-hoc models for scenarios wherein the check-in behavior may vary from the common social network setting [7], [10].

In the social network setting, we model the network effect using a simple probabilistic model adopted from [6]. The model captures the check-in behavior of pairs of users. Assume a pair of users choose to visit a place together with probability $\beta_{u,v}$, and individually (i.e., without co-locating) with probability $1 - \beta_{u,v}$; in either case the choice of location(s) is made from the discrete set $L$. Then, with probability $\beta_{u,v}$ they visit the same location, and with probability $1 - \beta_{u,v}$ their visited locations are different. Here, $\beta_{u,v}$ is simply the probability of user $u$ co-locating with user $v$, and can be set using the adversary's background knowledge for user $u$, specifically, $u$'s probability distribution of co-locating with other users $v \in U$. Accordingly,

$$\Pr\{A_v^{l_j} | A_u^{l_a}\} = \begin{cases} \beta_{u,v} + (1 - \beta_{u,v}) \cdot \Pr\{A_v^{l_j}\}, \text{ if } l_j = l_a \\ (1 - \beta_{u,v}) \cdot \Pr\{A_v^{l_j}\}, \text{ otherwise} \end{cases} \tag{10}$$

The probabilistic model fit here is orthogonal to the attack framework. One can plug-in a sophisticated model that gives an accurate value for $\Pr\{A_v^{l_j} | A_u^{l_a}\}$. Using Eq. (10) to simplify Eq. (9), we obtain

$$\Pr\{O_v^{l_a} | A_u^{l_a}\} = (1 - \beta_{u,v}) + \beta_{u,v} \cdot \Pr\{O_v^{l_a} | A_v^{l_a}\} \tag{11}$$

Furthermore, using Eq. (11), we get

$$\Pr\{O_v^{l_a} | O_u^{l_o}\} = \sum_{j \in L} \left((1 - \beta_{u,v}) + \beta_{u,v} \cdot \Pr\{O_v^{l_a} | A_v^{l_j}\}\right) \cdot \Pr\{A_u^{l_j} | O_u^{l_o}\} \tag{12}$$

Finally, plugging in the values from Eqs. (12) and (11) into Eq. (7), we get our objective $\Pr\{A_u^{l_a} | O_u^{l_o}, O_v^{l_a}\}$ as

$$\frac{\left((1 - \beta_{u,v}) + \beta_{u,v} \cdot \Pr\{O_v^{l_a} | A_v^{l_a}\}\right) \cdot \Pr\{A_u^{l_a} | O_u^{l_o}\}}{(1 - \beta_{u,v}) + \sum_{j \in L} \beta_{u,v} \cdot \Pr\{O_v^{l_a} | A_v^{l_j}\} \cdot \Pr\{A_u^{l_j} | O_u^{l_o}\}} \tag{13}$$

Factor $\Pr\{A_u^{l_j} | O_u^{l_o}\}$ can be estimated via Eq. (8) and factor $\Pr\{O_v^{l_a} | A_v^{l_j}\}$ depends solely on the sanitization mechanism according to Eq. (2). The combination of these efficiently computable factors enables a tractable algorithm (Algorithm 1) on real datasets.

## IV. Co-location Privacy Mechanisms

In this section, we present several approaches specifically designed to protect co-location privacy. Note that, one could use an existing technique for general-purpose location protection, and in the process also hide co-location information. For instance, $\epsilon$-geo-indistinguishability [1] is a powerful model that provides provable privacy guarantees for individual location protection. The Planar Laplace (PL) mechanism achieves $\epsilon$-geo-indistinguishability by perturbing every check-in location through addition of noise randomly drawn from a bidimensional Laplace distribution. However, as we show empirically in Section V, the amount of distortion enforced renders the data useless in terms of quality. As discussed in Section I, we consider a problem setting where users are comfortable with making their check-ins public, as long as no inference can be made about their co-locations. Protection can be achieved with less quality loss by sanitizing only co-location data, rather than all locations.

We introduce three customized *co-location* privacy mechanisms that specifically target co-located pairs of users. There are several challenges in sanitizing the co-location network $G = (C, CL)$. First, we must consider the presence of connected components in the network[3]. Figure 3 illustrates an example: connected components are formed as a direct consequence of the co-location generation algorithm, and represent a single check-in being simultaneously co-located with more than one other check-in. To protect co-location information, a sanitization mechanism needs to consider more than just individual pairs of co-locations. The second challenge is a consequence of the fact that a co-location protection mechanism is more successful if it also considers perturbation along the temporal dimension. Since a co-location is determined by both spatial and temporal constraints ($\Delta_s$ and $\Delta_t$), perturbing both attributes provides more flexibility. However, processing becomes more complex, as we need to adopt a distance function that combines the spatial and temporal dimensions. Specifically, we consider a linear function akin to the quality loss function in Eq. (1). Given two check-ins $c_u^i$ and $c_v^j$, the spatio-temporal *(ST)* distance between them is defined as:

$$ST_{dist}(c_u^i, c_v^j) = \gamma \cdot \frac{||c_u^i.l, c_v^j.l||}{MAX_S} + (1 - \gamma) \cdot \frac{|c_u^i.t, c_v^j.t|}{MAX_T} \tag{14}$$

where $\gamma$ is the weighting parameter, and $MAX_S$, $MAX_T$ are normalization constants explained in Section II.

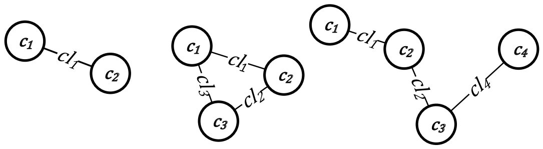


Figure 3: Connected Components in Co-location Network

### A. Gaussian Perturbation (GP)

The Gaussian perturbation mechanism distorts the location $l$ and timestamp $t$ of a co-location $cl$ to $l'$ and $t'$, as follows: for every check-in $c \in cl$, $cl \in CL$, we generate a spatial noise vector and a temporal scalar noise magnitude, each derived from the Gaussian (i.e., normal) distribution. The two-dimensional spatial noise vector is generated with a random uniform direction in the interval $[0, 2\pi)$ and a Gaussian-distributed magnitude drawn from $(\mathcal{N}, \sigma_s^2)$. A negative magnitude reverses the direction of the noise vector. The scalar temporal noise is drawn from $(\mathcal{N}, \sigma_t^2)$. Next, the location of check-in $c.l$ is transformed along the spatial noise vector and the timestamp $c.t$ is distorted by the magnitude of temporal scalar noise. Finally, the check-in is mapped to the closest location to $l'$ from the universe $L$. The amount of protection achieved can be controlled through the values of $\sigma_s$ and $\sigma_t$ (higher values correspond to more protection, but also result in higher utility loss).

The GP scheme is similar in concept to obfuscation techniques previously proposed for location privacy [2], [14], [23]. While some existing approaches for location protection employ the Laplace mechanism (e.g., [1], [26]), it has been shown in previous work [17] that Gaussian perturbation is more robust against adversaries with background knowledge. Since we consider a powerful adversary in our attack model, we adopt Gaussian perturbation for the proposed GP co-location protection technique.

[3] A connected component of an undirected graph is a subgraph in which any two vertices are connected to each other by paths, and which is connected to no additional vertices in the supergraph.

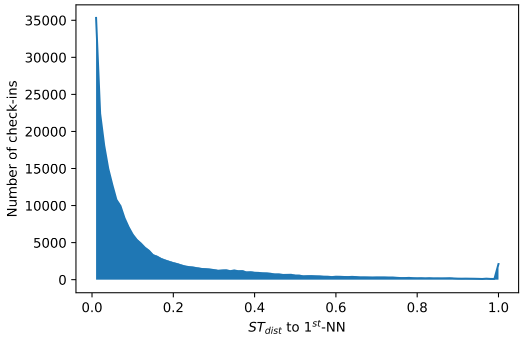


Figure 4: Histogram of normalized $ST_{dist}$ to $1^{st}$-NN

When protecting individual locations, the objective is to create uncertainty in the exact coordinates of a user's check-in, in order to hide his location from an adversary. In contrast, when protecting co-location privacy, the exact coordinates of a user's check-in are less important, and the focus is on hiding the existence of a co-location with another user. However, existing mechanisms that use Gaussian noise, being designed for individual location protection, make no effort to create distortion in the co-locating partners of a user. This has a negative effect on co-location protection, which is amplified in the case of skewed distributions of the spatio-temporal distances to the nearest neighbors of a victim's check-in (a histogram with normalized such distances in the Gowalla dataset is presented in Figure 4). Intuitively, if a co-located check-in is not displaced far enough to appear (falsely) co-located with another user, then there is too little noise added, and protection is not adequate. Consequently, for any fixed magnitude of noise, the Gaussian mechanism provides strong privacy in dense regions, and only limited protection in sparse regions. To address this shortcoming, we present next a mechanism which adapts the magnitude of noise to the distribution of nearby check-ins in the neighborhood of the considered check-in.

### *B. Adaptive Perturbation (AP)*

The adaptive co-location perturbation mechanism introduces noise dependent on the distribution of the nearest spatio-temporal neighbors for that co-location. Algorithm 2 presents the pseudocode for the AP mechanism. Given as input a positive integer $k$, for each check-in $c \in (c_i, c_j)$, the adaptive mechanism finds the set of its $k$ spatio-temporal nearest neighbors, and moves check-in $c_i$ to be co-located with another check-in chosen with uniform probability distribution over the set of positions of the $k$ selected neighbors (and including $c$'s current position). In other words, each check-in in a co-location is either left co-located with its current partner or displaced to appear co-located with one of its $k$ nearest neighbors.

**Algorithm 2** Adaptive Co-location Perturbation

```
procedure ADAPTIVE-PERTURB($G = (C, CL)$,$k$)
    for $cl \in CL$ do
        for $c \in cl(c_i, c_j)$ do
            $k\text{-}NN \leftarrow c \cup$ getNearestNeighbors($c.l, c.t, k$)
            $x \leftarrow$ random integer in $\{0, 1, 2..k\}$
            $c.l \leftarrow k\text{-}NN[x].l$
            $c.t \leftarrow k\text{-}NN[x].t$
```

Given an observed co-location $cl'$ the adversary can deduce that a check-in $c \in cl$ could have been either *(i)* left unperturbed, or *(ii)* moved to its current position from one of its reverse $k$ nearest neighbor(RkNN) locations[4].

On observing a co-located user pair $cl$, the adversary is uncertain if the check-ins in $cl$ were actually co-located together, or instead they co-located with either one of their $k$ potential partners at the reverse $k$ nearest neighboring (RkNN) locations. However, the reverse nearest neighbor query is not symmetric (i.e., if $B$ is an RNN of $A$, then $A$ is not necessarily an RNN of $B$ [25]). Consequently, if an observed co-located user pair $cl$ in $G'$ has no reverse nearest neighbors, then it becomes clear that $cl$ was also present in the original graph $G$. This provides the adversary with additional insight in reconstructing the true co-location. The other issue arises due to the presence of co-located groups of users (connected components as discussed before), wherein if the group is large compared to the value of $k$, then even after the mechanism is applied, the majority of the observed co-locations are highly likely to have been left unperturbed. For example, if a group of ten users are perturbed over $k = 1$ other locations, then in expectation, a majority of observed co-locations over the total two locations can be determined with high certainty. To alleviate these shortcomings, we introduce next the co-location masking mechanism.

### *C. Co-location Masking (CM)*

The main idea behind co-location masking is to group together multiple nearby co-locations, and thus bound the attacker's probability of identifying which of the co-locations in the group are real and which ones are false. Formally, we define co-location masking as follows:

*Definition 3:* Given a co-location network $G = (C, CL)$, a co-location $cl \in CL$ is said to be $b$-masked if it is spatio-temporally indistinguishable from $b - 1$ other co-locations.

Co-location masking guarantees that the re-identification probability is at most $1/b$. Consider the running example in Figure 2, where check-in $c_1 = (u_1, t_1)$ is co-located with $c_2 = (u_2, t_2)$ at the library. For simplicity, we consider only the spatial dimension. For this example, a possible co-location grouping that guarantees 3-masking is the minimum bounding circle that encloses another nearby check-in, say $c_3 = (u_3, t_3)$. This implies that the original true co-location $cl_1 = (c_1, c_2)$ is indistinguishable from $cl_2 = (c_2, c_3)$ and $cl_3 = (c_1, t_3)$. Thus, any co-location observed in the masked data is expected

[4] The reverse $k$ nearest neighbors(RkNN) of a check-in $c$ are all other check-ins $c' \in C'$ that have $c$ as one of their $k$ nearest neighbors. Specifically, $RkNN(c) = \{c' \in C' | ST_{dist}(c, c') \leq ST_{dist}(c, c'(k))$ , where $c'(k)$ is the $k$-th farthest NN of $c\}$.

to be true with probability at most $1/b$. In practice, the data publisher selects a value of $b$ commensurate with a tolerable re-identification probability (i.e., a threshold risk). Higher values of $b$ result in a lower probability of re-identification, but also introduce more distortion to the data, increasing quality loss.

The algorithm that implements $b$-masking creates co-location groups by agglomerating neighboring check-ins, thus creating new co-location observations indistinguishable from each other. Algorithm 3 presents the pseudocode of co-location masking. Sets $S$ is initialized to be empty, and used to store the connected graph components. The algorithm first computes all connected components $S'$ in the co-location network $G$ and stores them in set $S$[5]. For each connected component $S' \in S$, it computes the center of the Minimum Bounding Region (MBR), and then applies spatial and temporal transformation to each check-in $c \in S'$ in order to move it to the center. This transformation step converts each connected component into a clique situated at the center of the component. This step may create new co-locations amongst the members of the component that were previously further away than $\delta_s$ and $\delta_t$ distance (e.g., see third row of Figure 5(a)). Next, given the value of $b$, the size $|S'|$ of the connected component $S' = (V, E)$, and the number of edges $|E|$, the algorithm computes the minimum number $h$ of check-ins required to ensure $b$-masking within component $S'$. $h$ (in line 8) is computed by solving the following quadratic equation:

$$\underbrace{(b-1)\cdot|E|}_{\substack{\textbf{I} \triangleq \text{ total edges} \\ \text{required at } S'}} = \underbrace{{}^{|V|}C_2 - |E|}_{\substack{\textbf{II} \triangleq \text{ new edges} \\ \text{within set } V}} + \overbrace{{}^{h}C_2}^{\substack{\textbf{III} \triangleq \text{ edges in} \\ \text{clique of } h \text{ check-ins}}} + \underbrace{|V|\cdot h}_{\substack{\textbf{IV} \triangleq \text{ edges b/w} \\ \text{sets } V \text{ and } h}} \tag{15}$$

The variable $h$ is the final size of set $H$, which stores the $h$ nearest neighbors to the center of the component. Given $h$, the algorithm incrementally obtains the next nearest neighbor to the center of MBR($S'$) and adds it to set $H$ until $|H| = h$ check-ins of unique users that are not already in component $S'$ are found (lines 9-12).

To illustrate the algorithm, consider the component consisting of the three users in Figure 5(a). Assume $b = 2$. There are a total of three edges in-between the three users, hence $|V| = 3$ and $|E| = 3$. The privacy objective is to bound the adversary's probability of identifying a co-location with confidence greater than $1/b$. In our example, we must ensure that the three true edges are indistinguishable within a set of six observed ones. The value of $h$ is calculated to be 1 according to Eq. 15 as follows: new edges needed to create necessary masking is **I** = 3, the new edges formed within set $V$ after transformation to the center of $MBR(S')$ **II** = 0, the edges that would form within the $h$ check-ins of set $H$ when brought to the center of $MBR(S')$ **III** = 0, and the new co-locations formed as a result of moving the check-ins in $H$ to $MBR(S')$ **IV** = 3. Similarly, for the connected component with four users and three edges (i.e., $|V| = 4$, $|E| = 3$), **I** = 3, **II** = 3, **III** = 0, and **IV** = 0; implying that no additional edges beyond those created by the transformation to the center of MBR($S'$) are needed.

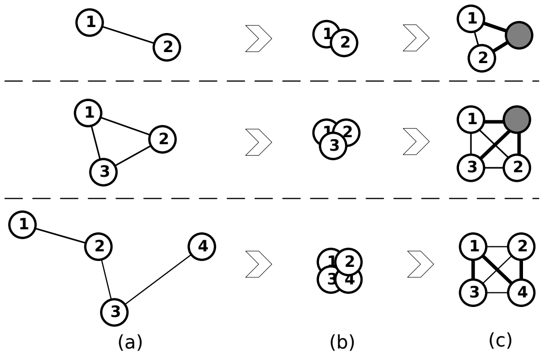


Figure 5: (a) Connected components in $G$ (b) components transformed to center of MBR (c) $b$-masking components

**Algorithm 3** Co-location Masking

```
1: procedure b-MASKING($G = (C, CL)$,$b$)
2:     $S \leftarrow$ getConnectedComponents($G$)
3:     for $S' = (V, E)$ in $S$ do
4:         $(l, t) \leftarrow$ center of MBR($S'$)
5:         for check-in $c$ in $S'$ do
6:             $c.l \leftarrow l, \quad c.t \leftarrow t$
7:         $H \leftarrow \emptyset$
8:         $h \leftarrow \left\lceil 0.5 \cdot (1 - 2 \cdot |V| + \sqrt{8 \cdot b \cdot |E| + 1}) \right\rceil$
9:         $c \leftarrow$ getNextNearestNeighbor($l$,$t$)
10:        while $c.u_id \notin V$ AND $h < t$ do
11:            $H \leftarrow c$
12:            $c \leftarrow$ getNextNearestNeighbor($l$,$t$)
13:        for check-in $c$ in $H$ do
14:            $c.l \leftarrow l, c.t \leftarrow t$
```

## V. Experiments

We use a subset of the Gowalla geo-social network check-in data collected by the SNAP project [5]. We select check-ins within the US, resulting in $3,669,249$ check-ins from $54,551$ unique users at $673,774$ locations between Feb 2009 and Oct 2010.We hash each check-in (represented by user ID and coordinates) to a regular $15,000 \times 15,000$ grid, which produces a resolution of 100 $m^2$ at each cell. To efficiently compute nearest-neighbor queries (used by the AP and masking methods), we adopt the conceptual partitioning method [15]. We also use a hash table on the user ID, to facilitate fast retrieval of co-locating partners. The spatial co-location threshold between check-ins is measured according to Euclidean distance. All algorithms were implemented in C++ and executed on an Intel Core 2 Duo 2.33GHz CPU with 32GB RAM running Ubuntu Linux 16.

Similar to other work studying co-locations [7], [19], we set $\Delta_s$ to 25 meters and $\Delta_t$ to 20 minutes. We set $\gamma = 0.5$ to give equal weighting to both spatial and temporal aspects

[5] It is straightforward to compute the connected components of a graph in linear time using either breadth-first search or depth-first search.

of the $ST_{dist}$ and the Quality Loss (QL) metric. For the Gowalla dataset, the $99^{th}$ percentile of co-located check-ins has a spatio-temporal nearest neighbor within 5 km and 12 hours, so we set $MAX_s$ and $MAX_t$ accordingly.

**Adversarial Knowledge.** Before running the attack on an observed check-in $c$ of user $u$, we compute $u$'s prior as a vector of probabilities over his frequent locations and co-locating partners. We build his prior knowledge using 70% of data, and use the remainder to evaluate our defense mechanisms. However, we omit all users with fewer than 10 total check-ins from this computation, so as to not build highly specific user-priors (as commonly done in literature [4]).

**Baseline Approach.** To the best of our knowledge, our work is the first approach to directly target co-location protection. We use as a baseline for comparison the $\epsilon$-geo-indistinguishability (*GeoInd*) technique [1] which focuses on protecting individual locations. A mechanism satisfies GeoInd if for all observations $l_o \subseteq L$:

$$P\{l_o|l_a\} \leq e^{\epsilon d(l'_a,l_a)} P\{l_o|l'_a\} \qquad \forall\, l_o, l_a, l'_a \in L \qquad (16)$$

In other words, an adversary cannot distinguish whether the user location is $l_a$ or $l'_a$ by a factor larger than $e^{\epsilon d(l_a,l'_a)}$. The planar Laplace (PL) mechanism achieves GeoInd by adding to each location noise drawn from a two-dimensional Laplace distribution centered at the real location $l_a$. Note that, GeoInd does *not* protect time values. Since PL works for locations on the continuous plane, we employ *remapping*, where the reported location is the element from universe $L$ which is closest to the noisy coordinates.

### A. Privacy Evaluation

**Gaussian Perturbation.** Figure 6(a) shows inference precision results when varying the the magnitude of injected Gaussian noise (parameters $\sigma_s$ and $\sigma_t$). We increment $\sigma_s$ and $\sigma_t$ as a multiple of the co-location constraint as $\sigma_s = n_o\Delta_s$ and $\sigma_t = n_o\Delta_t$, where $n_o$ ranges from 3 to 7. As expected, as noise increases the attack precision decreases. However, the variance plot in Figure 6(c) reveals that co-locations protection is rather inconsistent (bottom and top of the box correspond to $25^{th}$ and $75^{th}$ percentiles). Precision of over 50% is achieved for sparser areas of the data (the upper end of the whisker), signifying that users in such areas are not well protected.

To quantify the effect of density, we measure variance of precision as a function of the distance between the check-in location and its nearest neighbors. We group user pairs with similar $ST_{dist}$ to their nearest neighbor into buckets, and measure average precision per bucket. Figure 6(b) shows that attack precision increases sharply in sparser areas.

**Adaptive Perturbation.** The AP mechanism addresses the limitation of GP which fails to provide protection in sparse areas. Each check-in in a co-location is either left co-located with its current partner, or displaced to appear co-located with one of its $k$ nearest neighbors. Figure 7(a) shows the precision of the attack with varying $k$ (higher $k$ results in a stronger protection). As expected, as protection increases, the inference precision decreases. However, as opposed to GP, the amount of protection is more consistent when check-in location density varies, as shown in Figure 7(c). This result is confirmed by the plot in Figure 7(b) that shows a much smoother precision variation as the sparsity increases. In the dense regions (first 10 percentiles), we observe a sharp decrease in the adversary's precision, due to the fact that many co-locations exist in close proximity to each other, and it is more difficult for the adversary to distinguish among users located in close mutual proximity.

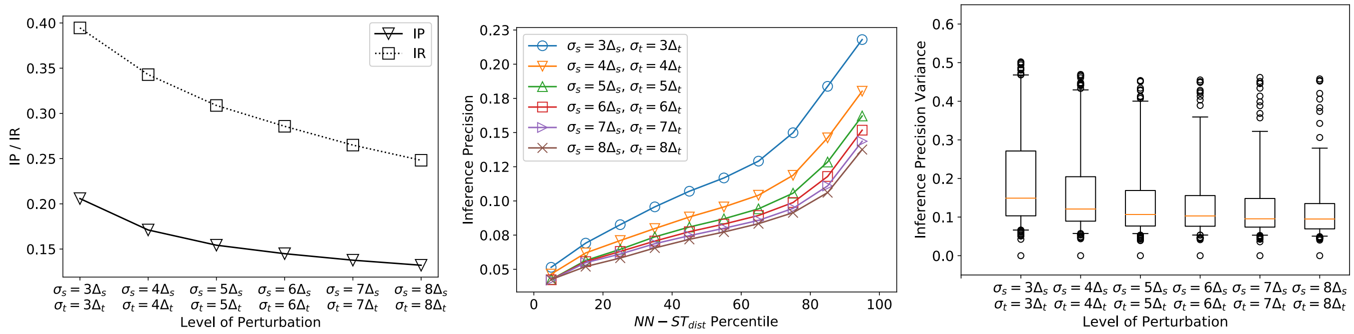
**Co-location masking** Figure 8(a) shows the attack precision when increasing the amount of protection. Recall that $b$-masking bounds the attack precision to $1/b$. For $b = 2$ we find that the observed inference precision is far less than $1/2$, which is a positive aspect. Consider a co-located pair of check-ins $cl = (c_1, c_2)$ (see first row in Figure 5(a) for an illustration). When $b = 2$ we need only one other co-location to protect $cl$. Suppose the procedure displaces $cl$'s nearest neighbor, say $c_3$, thereby forming new co-locations at $cl$. While this satisfies 2-masking, it inadvertently also 3-masks $cl$, since the latter is now indistinguishable from the newly co-located pairs $(c_1, c_3)$ and $(c_2, c_3)$.

Co-location masking achieves comparable protection with AP. However, recall from Section IV-B that AP does not perform well in some configurations due to the asymmetry of the reverse nearest neighbors. While this phenomenon may not be visible on the considered dataset, due to the fact that results are aggregated over a large number of check-ins, co-location masking still has an advantage over AP when the attacker narrows down the area of interest to a few users.

**$\epsilon$-geo-indistinguishability.** Finally, we quantify attack precision when protecting data with the GeoInd baseline (Figure 9). We vary the privacy parameter $\epsilon$ from 4 to 0.5 (higher values correspond to less protection and less noise). We can observe that the precision of the attack is indeed lower with GeoInd. However, the variance box plot in Figure 9(c) reveals that the privacy protection is fairly inconsistent, and often the adversary's precision reaches close to 50%. Figure 9(b) illustrates that GeoInd is also vulnerable when the check-in location density is low. In addition, as we show next, the noise introduced by GeoInd is so large that it renders the published data useless.

### B. Data Utility Evaluation

In Figure 10 we present a scatter plot of the inference attack precision versus the quality loss (QL) metric (Eq. (1)). Due to its excessive noise, the QL obtained by GeoInd is more than one order of magnitude worse than the other techniques. Among the three proposed mechanisms that directly target co-location protection, we find that GP outperforms both AP and co-location masking. However, recall that the protection achieved by GP is weaker, especially in sparse areas. AP and masking have similar performance, with AP slightly better in terms of quality loss, exhibiting an interesting protection-quality trade-off (recall that, masking provides better protection in the case of targeted attacks).

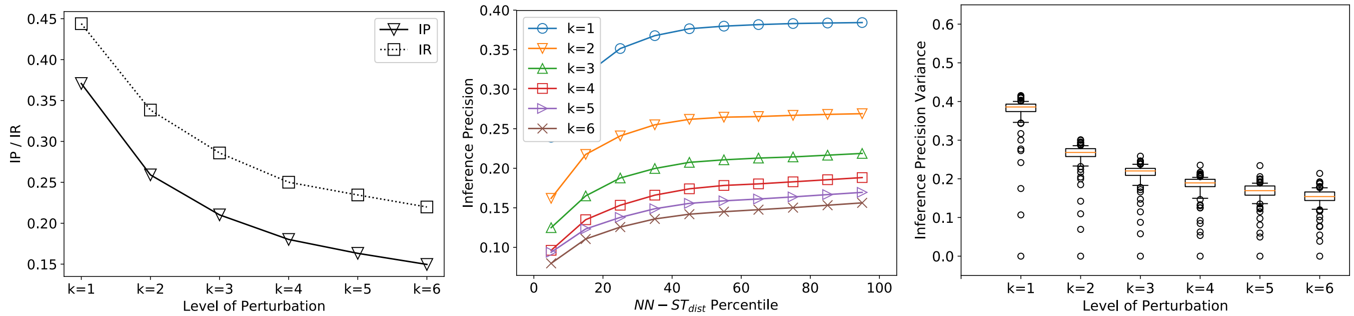


(a) IP/IR vs $\delta_s, \delta_t$

(b) IP w.r.t. $ST_{dist}$ to NN

(c) Precision Variance

Figure 6: Experiments on *Gaussian Perturbation*

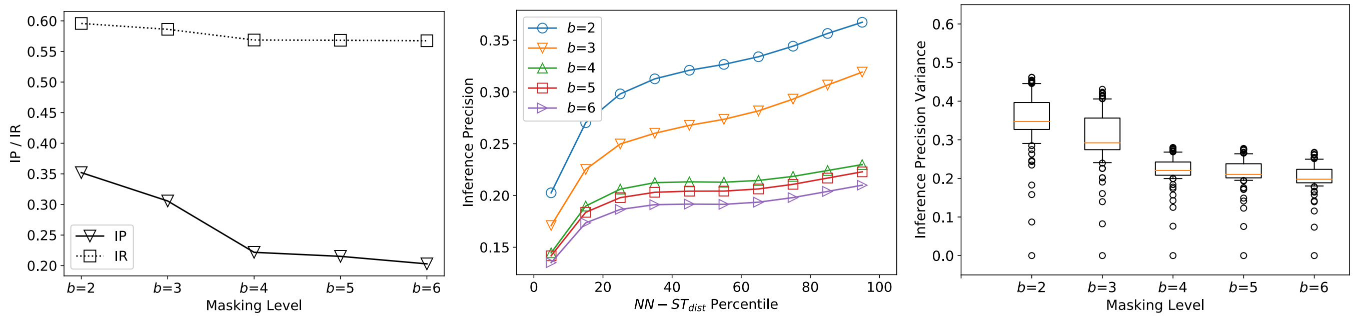


(a) IP/IR vs $k$

(b) IP w.r.t. $ST_{dist}$ to NN

(c) Precision Variance

Figure 7: Experiments on *Adaptive Perturbation*

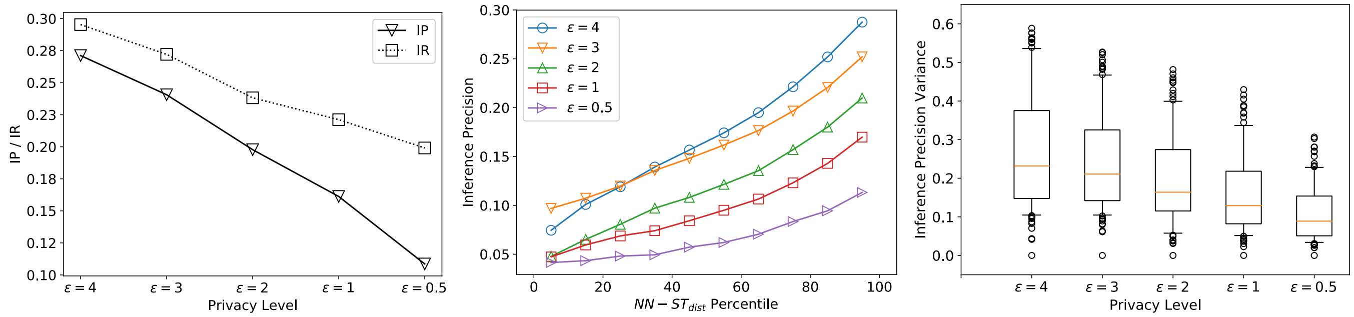


(a) IP/IR vs $b$

(b) IP w.r.t. $ST_{dist}$ to NN

(c) Precision Variance

Figure 8: Experiments on *Co-Location Masking*



(a) IP/IR vs $\epsilon$

(b) IP w.r.t. $ST_{dist}$ to NN

(c) Precision Variance

Figure 9: Experiments on $\epsilon$-geo-indistinguishability

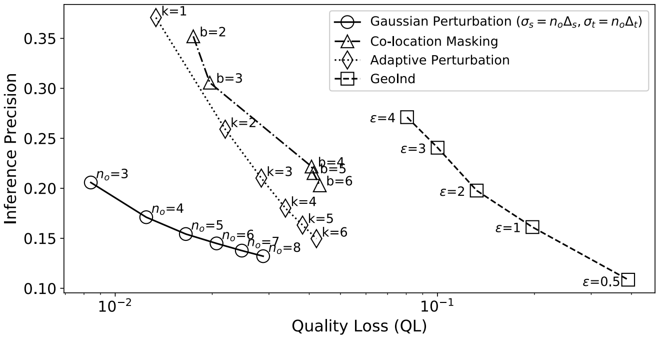


Figure 10: Privacy vs Quality Loss (QL)

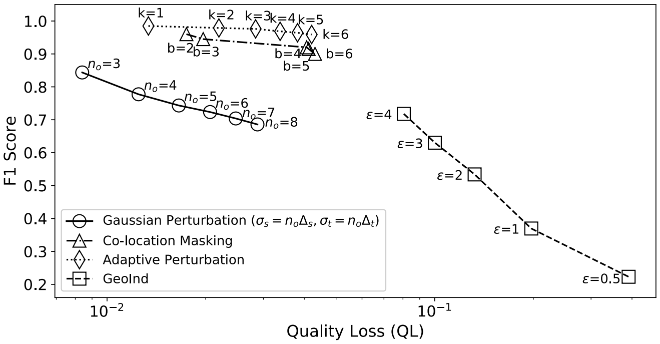


Figure 11: F1 score of Range Search vs Quality Loss

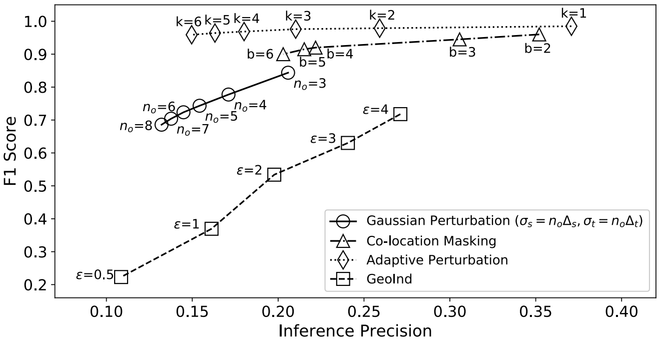


Figure 12: F1 score of Range Search vs Inference Precision

Next, we evaluate utility with respect to actual queries on the sanitized data. Since we expect the data to be used for geospatial processing tasks, we measure the accuracy of spatio-temporal *range* queries (we average results over one thousand queries and ten random seeds). We quantify query accuracy using the F1 score (i.e., the harmonic mean of precision and recall), defined with respect to the data points (i.e., check-ins) that fall within the query range on the actual, and respectively sanitized dataset.

Figure 11 presents a scatter plot of F1 score versus the QL metric. Similar to the previous experiment, we note that the utility loss of GeoInd is excessive, and that the proposed approaches achieve good privacy-utility trade-offs. In addition, the results confirm the validity of the QL metric chosen: note that, the precision of actual queries on the sanitized data correlates well with QL. Finally, in Figure 12 we plot the F1 score of query answering versus attack precision. Again, we observe that GeoInd performs poorly in terms of utility. For the higher values of $\varepsilon$, when the utility is slightly better, there is no gain in terms of attack precision compared to the proposed approaches. We can conclude that conventional GeoInd is not a suitable approach for co-location protection: when protection is high, data become useless, whereas when utility improves (even though it stays considerably below that of the proposed approaches), the protection is no better than that of our techniques.

## VI. Related Work

**Attacks on co-location data.** Individuals with strong social ties tend to often be located in close proximity to each other. The work in [10] was one of the first to compare data from cell phone operators with self-reported survey data, and infer user social connections based on co-location and calling patterns. The work in [6] develops a formal probabilistic model to investigate the extent to which social ties can be inferred from co-locations. The authors found that even a small number of co-locations can result in a high empirical likelihood of a social connection. Later in [7], discovering social ties is formulated as a classification problem: a large number of features (e.g., spatial and temporal ranges of co-locations, location diversity, location specificity and other structural properties) are extracted to train a friendship predictor. The work in [19] proposed an entropy-based model (EBM) to estimate the social strength between all pairs of users in a location dataset. The authors identify key features that are strong indicators of friendship between users, such as diversity of non-coincidental co-occurrences [19], popularity of common locations [7] and distinctive personal factors [10].

The work in [3] takes a different approach: the authors propose a feature learning technique to automatically summarize users' mobility features. They model human mobility as a random-walk process with the probability of a user transitioning to a location being directly dependent on the proportion of his check-ins at that location. While protecting against inference attacks that uncover social ties is an important problem, by focusing on co-locations directly we cover the more general problem of protecting against all types of attacks on co-location data, beyond just those specific to location-centric social networks. Note that the discovery of social ties is found to rely on a special set of features, and it is possible that privacy-preserving mechanisms that specifically target these features may perform better than a generic method.

**Location protection mechanisms.** Protection of *individual* locations has been extensively researched in the last decade. Early attempts to protect location of mobile users considered a simple reduction in reporting accuracy. *Location cloaking* obscures locations by reporting a coarser region. Spatial $k$-anonymity (SKA) [12], [13] hides the actual location of a user among a set of $k-1$ other users, attempting to bound an adversary's chance of learning exact user identity and location to $1/k$. Numerous approaches fall in this category, but as shown in [9], $k$-anonymity has serious limitations in the presence of adversaries with background knowledge.

Another stream of work adds noise to the reported location data. The techniques in [2] and [23] reduce precision by dropping the low-order bits of the coordinates. The work in [14] empirically evaluates the effect of adding spatial Gaussian noise to check-in locations of users in order to protect their identity from being discovered through a simple Web search. More recently, the formal model of *geo-indistinguishability* [1] was introduced, which adapts the popular differential privacy model for relational tables to location datasets. From an attacker's perspective, the released location of a user reveals no more information about his whereabouts than the knowledge of his presence within a specified discretized radius $r$ around his actual coordinates. However, this model protects all locations, without specific focus on co-locations. As a result, excessive noise is added, and the sanitized data loses its utility, as we showed in Section V. Our attack model is related to the work in [22], [23]. They consider an adversary with prior information about user mobility patterns. While their approach is similar in concept to ours, we customize the attack to target discovery of co-locations. Moreover, our attack uses both space and time information (in [20], [22] only locations are used).

## VII. Conclusions

In this paper, we conducted the first systematic investigation of the co-location privacy problem. First, we introduced a powerful attack that uses a large amount of background knowledge customized for inferring co-locations. Next, we proposed three privacy mechanisms that protect co-location information, and achieve interesting trade-offs between privacy and utility. Our extensive experimental evaluation shows that state-of-the-art generic methods that do not focus specifically on co-locations introduce excessive noise in the data, significantly lowering the utility of released datasets. In addition, when data are skewed, the protection provided against a powerful adversary is poor. Among the proposed protection mechanisms, we found that the adaptive perturbation scheme is well-suited for maximizing sanitized data utility.

In future work, we plan to study a formal model for co-location protection that can provide provable privacy guarantees. For instance, one promising direction is to devise adaptations of differential privacy that are specifically tailored for protecting co-location data, rather than individual locations (as current existing state-of-the art methods do).

**Acknowledgements.** This research has been supported in part by NSF grant CNS-1461963, Caltrans-65A0674, LA Metro contract LA-Safe-PS36665000, the USC Integrated Media Systems Center, the Annenberg Foundation, and unrestricted cash gifts from Google and Oracle. The opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any of the sponsors.

## References

[1] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *ACM CCS*, 2013.

[2] C. A. Ardagna, M. Cremonini, S. D. C. di Vimercati, and P. Samarati. An obfuscation-based approach for protecting location privacy. *IEEE Trans. on Dependable and Secure Computing*, 8(1):13–27, 2011.

[3] M. Backes, M. Humbert, J. Pang, and Y. Zhang. walk2friends: Inferring social links from mobility profiles. In *ACM CCS*, 2017.

[4] K. Chatzikokolakis, E. Elsalamouny, and C. Palamidessi. Efficient utility improvement for location privacy. *Proceedings on Privacy Enhancing Technologies*, 2017(4):308–328, 2017.

[5] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *SIGKDD*, 2011.

[6] D. J. Crandall, L. Backstrom, D. Cosley, S. Suri, D. Huttenlocher, and J. Kleinberg. Inferring social ties from geographic coincidences. *PNAS*, 2010.

[7] J. Cranshaw, E. Toch, J. Hong, A. Kittur, and N. Sadeh. Bridging the gap between physical location and online social networks. In *ACM UbiComp*, 2010.

[8] C. A. Davis Jr, G. L. Pappa, D. R. R. de Oliveira, and F. de L Arcanjo. Inferring the location of twitter messages based on user relationships. *Transactions in GIS*, 15(6):735–751, 2011.

[9] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, 2006.

[10] N. Eagle, A. S. Pentland, and D. Lazer. Inferring friendship network structure by using mobile phone data. *PNAS*, 2009.

[11] G. Greenwald and E. MacAskill. Nsa prism program taps in to user data of apple, google and others. *The Guardian*, 7(6):1–43, 2013.

[12] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *ACM MobiSys*, 2003.

[13] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. *IEEE TKDE*, 2007.

[14] J. Krumm. Inference attacks on location tracks. *Pervasive computing*, pages 127–143, 2007.

[15] K. Mouratidis, D. Papadias, and M. Hadjieleftheriou. Conceptual partitioning: An efficient method for continuous nearest neighbor monitoring. In *SIGMOD*, 2005.

[16] W. H. Organization et al. *Contact tracing during an outbreak of Ebola virus disease*. World Health Organization, 2014.

[17] S. Oya, C. Troncoso, and F. Pérez-González. Is geo-indistinguishability what you are looking for? In *ACM WPES*, 2017.

[18] H. Pham and C. Shahabi. Spatial influence-measuring followship in the real world. In *ICDE*, 2016.

[19] H. Pham, C. Shahabi, and Y. Liu. Ebm: an entropy-based model to infer social strength from spatiotemporal data. In *SIGMOD*, 2013.

[20] A. Pyrgelis, C. Troncoso, and E. De Cristofaro. What does the crowd say about you? evaluating aggregation-based location privacy. *Proc. of Privacy Enhancing Technologies*, 2017(4):156–176, 2017.

[21] H. Shirani-Mehr, F. Banaei-Kashani, and C. Shahabi. Efficient reachability query evaluation in large spatiotemporal contact datasets. *PVLDB*, 2012.

[22] R. Shokri, G. Theodorakopoulos, G. Danezis, J.-P. Hubaux, and J.-Y. Le Boudec. Quantifying location privacy: the case of sporadic location exposure. In *Privacy Enhancing Technologies*, 2011.

[23] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux. Quantifying location privacy. In *IEEE S&P*, 2011.

[24] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[25] Y. Tao, D. Papadias, and X. Lian. Reverse knn search in arbitrary dimensionality. In *PVLDB*, 2004.

[26] Y. Xiao and L. Xiong. Protecting locations with differential privacy under temporal correlations. In *ACM CCS*, 2015.

[27] D. Xu, P. Cui, W. Zhu, and S. Yang. Find you from your friends: Graph-based residence location prediction for users in social media. In *IEEE ICME*, 2014.