A Closer Look: Evaluating Location Privacy Empirically

Liyue Fan
liyue.fan@uncc.edu
Department of Computer Science, UNC Charlotte
Charlotte, North Carolina, USA

Ishan Gote
igote@uncc.edu
UNC Charlotte
Charlotte, North Carolina, USA

ABSTRACT

The breach of users' location privacy can be catastrophic. To provide users with privacy protections, numerous location privacy methods have been developed in the last two decades. While several studies surveyed existing location privacy methods, the lack of comparative, empirical evaluations imposes challenges for adopting location privacy by applications and researchers who may not be privacy experts. This study fills the gap by conducting a comparative evaluation among a range of location privacy methods with real-world datasets. To evaluate utility, we consider different types of measures, e.g., distortion and mobility metrics; to evaluate privacy protection, we design two empirical privacy risk measures via inference and re-identification attacks. Furthermore, we study the computational overheads inflicted by location privacy in CPU time and memory requirement. The results are thoroughly examined in our work and show that it is possible to strike a balance between utility and privacy when sharing location data with untrusted servers.

CCS CONCEPTS

• **Information systems** → **Geographic information systems**; • **Security and privacy** → **Data anonymization and sanitization**.

KEYWORDS

Location Privacy, Comparative Evaluation, Real-World Data

ACM Reference Format:
Liyue Fan and Ishan Gote. 2021. A Closer Look: Evaluating Location Privacy Empirically. In *29th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '21), November 2–5, 2021, Beijing, China.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3474717.3484219

1 INTRODUCTION

Location data is increasingly used for services and social good. For instance, individuals' location data is shared with servers to obtain recommendations [13] and to participate in social distancing monitoring [20]; location data has also been adopted in research studies, e.g., in public policy [15] and mental health research [5, 19, 21]. However, location data is highly sensitive, the disclosure of which may lead to grave consequences. As a result, users who have concerns about sharing real location data with untrusted servers may opt out of the applications or research studies [23].

A plethora of location privacy methods have been developed to hide a user's real location from untrusted servers. A number of excellent surveys, such as [12] and [17], provide a comprehensive overview of those methods. Recently, the authors of [17] analyzed existing location privacy methods in architecture (i.e., trusted third party, non-trusted third party, peer-to-peer, and local) and use case (i.e., online and offline). This study focuses on methods that can be deployed with the *local* architecture for *online* use. Our rationale is two-fold. On one hand, the local architecture ensures that location privacy is enforced on the client device, i.e., not requiring communications with any other party (see Figure 1). It also provides users with a great sense of control over private information (e.g., analogous to the local differential privacy paradigm recently adopted by Google and Apple). On the other hand, the online use case for location privacy allows applications to provide immediate services and data analysis, such as recommendations and monitoring social distancing. It also benefits applications by ensuring data availability at the server side (e.g., in case of disruption).

There has been promising development in the adoption of local, online location privacy. E.g., Android users can choose to share approximate or precise locations with apps; our recent work [8] open-sourced a range of local, online location privacy methods in Java. However, it remains a challenge to *understand the impact of location privacy on data usefulness.* Count-based metrics (e.g., distribution estimation and range queries) that rely on aggregating data from a set of users are examples of well-studied utility metrics for location privacy. It is yet unknown how location privacy may affect applications that rely on features extracted from individuals' longitudinal data, e.g., mental health studies that build on a participant's mobility patterns [5, 19, 21]. Evaluating different types of utility metrics with real-world datasets is thus imperative to help app developers and researchers assess location privacy methods and adopt those that best suit their needs.

Furthermore, it has not been studied *whether location privacy methods effectively mitigate practical privacy risks.* In fact, existing methods operate under various privacy models that are not directly comparable. E.g., geo-indistinguishability [3] is based on differential privacy [7], while Spatial Cloaking [11] aims to hide a user's home location. It is thus beneficial to identify common empirical privacy risks (e.g., re-identification) and to evaluate location privacy methods against those risks. The results will help both users and apps/researchers understand the practical protection offered by location privacy methods, make informed decisions about adopting location privacy, and pave the way to building trust between them.

In this paper, we empirically evaluate location privacy methods regarding utility and privacy, with two real-world trajectory datasets. Our study covers a range of local online methods under

SIGSPATIAL '21, November 2–5, 2021, Beijing, China

ACM ISBN 978-1-4503-8664-7/21/11…$15.00
https://doi.org/10.1145/3474717.3484219

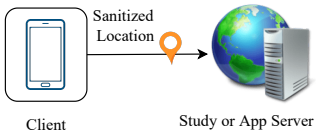


Figure 1: The *Local* Architecture for Location Privacy.

multiple categories, i.e., generalization-based, perturbation-based, and dummy-based. We investigate how parameters of those methods interact with utility and practical privacy protection. To measure utility, We consider a set of metrics, including distortion metrics and mobility metrics; to evaluate privacy protection, we design two empirical privacy risk measures, i.e., via inference attack and re-identification attack. Moreover, we measure the computational overheads inflicted by location privacy methods in CPU time and memory requirement, to examine their performance from a systematic perspective. Our evaluation is the first of its kind and the results show that it is possible to achieve a balance between utility and privacy, e.g., providing accurate or even truthful data at low empirical privacy risk levels.

The rest of the paper is organized as follows: Section 2 defines the utility metrics and privacy risk measures; Section 3 briefly reviews the location privacy methods to evaluate; Section 4 describes the evaluation methodology and discusses the results; Section 5 interprets the outcome, distinguishes application needs, and discusses practical considerations for adopting location privacy; Section 6 concludes the paper and states future research directions.

2 PRELIMINARIES

We focus on the *local* architecture of location privacy in the *online* setting. Specifically: the method must operate locally on the client, i.e., not relying on other users or third-parties (see Figure 1); the method must operate on-the-fly, i.e., reporting a sanitized location upon receiving the real location, without knowing future locations.

2.1 Basic Definitions

We consider that the two-dimensional geospace D is discretized into a set of disjoint grid cells, i.e., $D = \{c_i | i = 1, \cdots, N\}$. We denote the coordinates of c_i as $(lat_i, long_i)$, which represents the center point of the cell. A user's location at time t is approximated by the grid cell he/she is in, i.e., $l_t \in D$. A trajectory of the user is a sequence of locations observed on a particular day: $Tr = \{l_1, l_2, \cdots, l_M\}$, where M is the number of observations per day given a sampling rate, e.g., every 15 minutes or every hour. Note that the amount of data contributed by each user greatly varies in real datasets.

Given a consecutive subsequence of a trajectory $Tr(t_1, t_2) = \{l_{t_1}, ...l_{t_2}\}$ where $1 \leq t_1, t_2 \leq M$, we can identify $N(t_1, t_2)$ *stop places* [5] where the user spends a duration of time. Formally, a stop place is defined as a tuple $Pl = \langle l, T \rangle$, where, l is the location, T is the duration spent at l. The set of stop places during the interval $[t_1, t_2]$ can be denoted as $\left(Pl_1, Pl_2, \ldots Pl_{N(t_1,t_2)}\right)$. For example, let $Tr(t_1, t_1+4) = \{(t_1, c_1), (t_1+1, c_1), (t_1+2, c_2), (t_1+3, c_3), (t_1+4, c_3)\}$. 3 stop places can be identified in $[t_1, t_1 + 4]$: $Pl_1 = \langle c_1, 2 \rangle$, $Pl_2 = \langle c_2, 1 \rangle$, and $Pl_3 = \langle c_3, 2 \rangle$.

A *local*, *online* location privacy method can be viewed as a function that only has access to the user's current location and optionally historical information, and outputs a sanitized location: $LP(l_t[, hist]) = l'_t$, where $l_t, l'_t \in D$. This conceptual model boasts simplicity and applies to most location privacy methods considered in our study. Adjustments will be made and discussed in the next section, for dummy-based methods. We denote the sanitized trajectory as Tr' which is comprised of a sequence of sanitized locations.

2.2 Utility Metrics

We introduce two classes of utility metrics: *distortion* metrics and *mobility* metrics.

2.2.1 Distortion Metrics. Common distortion metrics for location data include the Hamming distance and the Haversine distance. The **Hamming distance** between two locations is defined as:

$$\text{Hamming}(l, l') = \begin{cases} 0 & \text{if } l = l' \\ 1 & \text{else} \end{cases} \tag{1}$$

It is used to measure whether the sanitized location differs from the input. As a user often contributes multiple locations, e.g., in one or more trajectories, we can compute the average Hamming distance among all input locations of the user. The **Haversine distance** measures how far the sanitized location is to the input:

$$\text{Haversine}(l, l') \tag{2}$$
$$= 2r \arcsin\left(\sqrt{\sin^2(\frac{\varphi' - \varphi}{2}) + \cos(\varphi)\cos(\varphi')\sin^2(\frac{\lambda' - \lambda}{2})}\right)$$

where r indicates earth radius in meters, and φ (φ') and λ (λ') are the latitude and longitude of l (l') in radians. Similarly, we can compute the average Haversine distance when a user contributes more than one locations.

As mentioned in introduction, location data reported by a set of users is often used for estimating distributions and answering range queries. Specifically, at a given timestamp, the server may aggregate data from all users and count users at each location c_i or within a 2-D query window r_k. We denote the real number of users at location c_i as cnt_{c_i}, and the number based on reported locations as $\hat{cnt}_{c_i}$ (and for query r_k, cnt_{r_k} and $\hat{cnt}_{r_k}$, respectively). **MAE** (mean absolute error) measures the absolute difference between cnt and $\hat{cnt}$, averaged among all locations/range queries:

$$\text{MAE}_{freq} = \frac{1}{|D|} \sum_{c_i \in D} |\hat{cnt}_{c_i} - cnt_{c_i}| \tag{3}$$

$$\text{MAE}_{range} = \frac{1}{|Q|} \sum_{range_k \in Q} |\hat{cnt}_{r_k} - cnt_{r_k}| \tag{4}$$

where Q represents the set of range queries to be answered.

2.2.2 Mobility Metrics. Our work is the first to consider the usefulness of location privacy in the context of behavioral studies. For example, several studies extracted mobility features from participants' GPS traces to understand their mental health states [5, 19]. We adopt the mobility metrics proposed in [5], which are defined for a single user in a time period, e.g., $[t_1, t_2]$. The metrics are computed based on the user's stop places, which can be extracted for the real trajectory and the sanitized trajectory. Note that stop places identified using the sanitized Tr' may differ the real stop places. Seven mobility metrics are evaluated in our study: Tot Dist,

Max Dist, Std Dev Displacement, Max Dis Home, Rad Gyration. # Diff Places, and # Significant Places. The detailed definitions are reported in Appendix A.1.

For a specific user and time interval $[t_1, t_2]$, we measure the *absolute relative error* between the metrics computed using the real trajectory Tr and those computed using the sanitized trajectory Tr'. It is also important to examine the impact of the *window size*, i.e., $t_2 - t_1 + 1$, as a larger window may amplify the effect of distortion introduced by location privacy.

2.3 Empirical Privacy Measures

Existing location privacy methods differ in the underlying privacy models, which may not be directly comparable. For example, it is difficult to compare the privacy protection provided by Laplace [3], which is based on the differential privacy model, to that of Spatial Cloaking [11], which aims to hide a user's home location. To conduct a comparative evaluation, we design two types of attacks which quantify the empirical privacy risks in sharing location data. For simplicity of notation, we consider that every user contributes one trajectory (i.e., one day's data) to the dataset, although the following attacks can be extended to incorporate the user's data generated across multiple days.

Notation and assumptions. Let Tr_j denote the trajectory of user u_j and Tr'_j denote the sanitized trajectory by a location privacy method. The sanitized dataset $\mathcal{D}'$ contains the sanitized trajectories from all users, i.e., $\mathcal{D}' = \bigcup_j Tr'_j$. Let $S_j = \text{set}(Tr_j)$ be the set of distinct locations in Tr_j and $S'_j = \text{set}(Tr'_j)$ for Tr'_j, respectively. Let u_{tar} denote the target user of an attack. We assume an adversary (e.g., the server or researcher) who has *partial knowledge* about the target's location history, e.g., some locations in S_{tar}. Practically, the adversary may not know the exact time when the target visits a location or the sequence of the visits. Its goal of the inference attack is to predict the previously unknown location of the target user, while the goal of the re-identification attack is to identify the target's trajectory in the sanitized dataset with only a small number of known locations. In both attacks, the adversary has access to $\mathcal{D}'$ and can perform set() on the sanitized trajectories.

2.3.1 Inference Attack. In this attack, the adversary knows all but one locations in S_{tar} and tries to infer the unknown location by querying the sanitized dataset $\mathcal{D}'$. Specifically, the adversary creates a query $q \in \{S | S \subset S_{tar}, |S| = |S_{tar}| - 1\}$ and retrieves all sanitized trajectories Tr'_js such that the distinct location set is a superset of the query, i.e., $q \subset \text{set}(Tr'_j)$. Then the adversary identifies the most visited location(s) L by the retrieved trajectories that are not in q. The attack with q is successful if L contains the unknown location, i.e., $S_{tar} \setminus q \subset L$. We consider a target user is successfully attacked as long as one query $q \in \{S | S \subset S_{tar}, |S| = |S_{tar}| - 1\}$ is successful. The inference attack captures whether the sanitized dataset preserves *correlations* among locations visited in a trajectory, i.e., between locations in q and $S_{tar} \setminus q$. Intuitively, users who visit a small number of locations that are also visited by other users are prone to this attack.

2.3.2 Re-identification Attack. In this attack, the adversary knows a small subset of locations in S_{tar} and tries to re-identify the trajectory that belongs to u_{tar} in the sanitized dataset $\mathcal{D}'$ [1]. Specifically, the adversary creates a query $q^k \in \{S | S \subset S_{tar}, |S| = k\}$ and retrieves all sanitized trajectories Tr'_js such that $q^k \subset \text{set}(Tr'_j)$. If only one trajectory is retrieved and the trajectory belongs to u_{tar}, we consider the attack with q^k successful. If all q^ks are successful, we can conclude that u_{tar} can be re-identified with knowing any k locations. In our evaluation, we report the smallest k for each user, which indicates the difficulty (i.e., the minimum knowledge required) of re-identifying the user. The re-identification attack captures the *uniqueness* of users in the sanitized dataset. Unlike the inference attack, users who visit a small number of locations that are also visited by other users are less likely to be re-identified.

Note that the distortion inflicted by location privacy on the sanitized data has an effect on the success of both attacks, since they require matching the sanitized trajectories to queries constructed with real data. An exception is for non-randomized location privacy methods, where the adversary can construct queries with sanitized locations, i.e., by applying the non-randomized methods to the query. We will explore the "improved" version of re-identification in the experiment section. We also conduct both attacks on real data, i.e., by querying the real dataset $\mathcal{D} = \bigcup_j Tr_j$, to provide a reference of risks without enhancing location privacy.

3 LOCATION PRIVACY METHODS

In this section, we briefly describe the location privacy methods implemented and evaluated in this study. We group the methods according to the categorization in [17].

3.1 Generalization-based Methods

Generalization entails reporting coarser information instead of the exact location. Such methods often report location data with reduced precision, resulting a trade-off between privacy and utility.

Rounding [11, 14]. Also known as *truncation* in [14], this method snaps the real location coordinates, i.e., *lat* and *long*, to a fixed square grid where the spacing s (in meters) is specified by the user. Note that the grid used in the method may differ from the discretized space D. Specifically, the rounded coordinates lat' and $long'$ can be computed using the following formulae [14]:

$$lat' = s_{lat} \left\lfloor \frac{lat}{s_{lat}} \right\rceil, \quad long' = s_{long} \left\lfloor \frac{long}{s_{long}} \right\rceil \tag{5}$$

where s_{lat} and s_{long} are the spacing s translated into degrees of latitude and longitude. They can be derived using standard approximations, e.g., with WGS 84. The parameter s controls the level of privacy: the larger s, the coarser the grid, and more input locations will produce the same output, thus more private.

Spatial Cloaking [11]. Although "cloaking" often refers to hiding one user among other users in the region, this method in [11] applies to a single user's data and protects a specific sensitive location, e.g., home of the user. By deleting location data near the sensitive location, this method hides the sensitive location inside a cloaked

[1] Note that the re-identification risk is different from membership disclosure, as the adversary (server or researcher) in our problem setting knows the target's participation in the data.

region, which is centered at a randomly selected point. Illustrated in Figure 2, three input variables are supplied to the method, i.e., the location to protect, the radius of the small circle r, and the radius of the large circle R. The user's location data, if fallen inside the large circle, is deleted. Parameters r and R indicate the level of privacy: the larger r and R, more uncertainty to locate the user's home and more data deleted. In our experiment, we fix the r value while varying R, to study the effect of data deletion on utility and privacy.

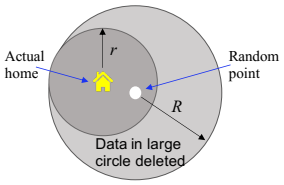


Figure 2: Spatial cloaking [11]. A point is randomly selected within r of the home location; a circle with radius $R > r$ is formed, and location data inside the circle will not be reported.

3.2 Perturbation-based Methods

Perturbation-based methods modify the location data, and most methods add noise to the real location. There is also a trade-off between privacy and utility; if the data is highly distorted, it would protect privacy but offer little usefulness.

Noise [11]. Adapting from [11], a random 2-D Gaussian noise can be drawn and added to the input's latitude and longitude. The direction of the noise vector is chosen uniformly at random from $[0, 2\pi)$ and the magnitude of the noise vector is drawn from a Gaussian distribution $N(0, var)$. A negative magnitude reverses the direction of the vector. $\sqrt{var}$ is the standard deviation in meters that can be specified by the user, e.g., 50 meters as in [11]. It can be seen that higher var values would offer higher levels of privacy as the output location is further away from the real location.

VHC [16]. The framework proposed in [16] constructs a context-aware space partitioning structure and maps the partitions into 1-D space using the Hilbert space-filling curve, called Various-size-grid Hilbert Curve (VHC).To sanitize a user's location, the method finds the partition that contains the location, adds a *uniformly distributed* random noise drawn from $[-\sigma, \sigma]$ to the partition's 1-D value, and reports a randomly selected 2-D point in the 1-D perturbed partition. The various-size grid is constructed recursively as a quad-tree such that all partitions have homogeneous road densities. In our evaluation, we construct the structure with road network data retrieved from OpenStreetMap[2] and the recursive partitioning stops at 500 nodes. The perturbation of 1-D partitions introduces uncertainty. And the parameter σ indicates the level of privacy: higher σ, higher uncertainty and perturbation inflicted in the output location. Note that space partitioning is done offline and the structure can be shared with the client; the perturbation of each location is performed online with computational complexity $O(\log n)$ where n is the number of partitions.

Laplace [3]. The notion of *geo-indistinguishability* is closely related to *differential privacy* [7], which is the state-of-the-art privacy paradigm for statistical databases. In [3] the authors proposed a sampling based approach to report a perturbed location, where the sampling distribution provides indistinguishability guarantees for the real location. Specifically, given the real location x_0 and privacy parameter ϵ, the following distribution for sampling an output location x is shown to satisfy ϵ-*geo-indistinguishability* [3]:

$$D_\epsilon(x_0)(x) = \frac{\epsilon^2}{2\pi} e^{-\epsilon d(x_0, x)} \tag{6}$$

where $d(\cdot, \cdot)$ denotes the Euclidean distance. The distribution in Equation 6 is called *planar Laplacian* centered at x_0. Our implementation [8] is based on the browser extension tool[3] developed by the authors of [3]. The parameter ϵ controls the privacy level: smaller ϵ value, higher indistinguishability and stronger privacy.

3.3 Dummy-based Methods

Another group of methods generate fake locations, called dummies, in order to hide the real location among them. The challenge is to generate realistic fake data, ideally indistinguishable from real data. There is a trade-off between privacy and efficiency, as multiple locations are generated for each input.

SpotME [18]. This method allows a user to claim to be at multiple locations simultaneously, thus providing privacy protection for the user's real location. Specifically, the user's device creates a position map for the 2-D grid D. When a new location is available, for each cell the method claims to be in that cell with probability p or report the truth (whether the real location is in that cell or not) with probability $(1 - p)$. Thus the reported position map may contain multiple cells that the device claims to be in. In our evaluation, we modify the output of SpotME to be comparable to other methods: among all cells the device claims to be in, we randomly select one cell as the output location. The value of p controls the level of privacy for this method: higher p, more dummies generated; and the output is more likely a dummy.

Moving in the neighborhood (MN) [10]. So far, we have introduced several methods that do not consider historical information, e.g., previously reported locations. The drawback is that when looking at the output trajectory, an adversary may identify non-realistic output locations, filtering out dummies as in [18]. The MN method was proposed to generate dummies that cannot be distinguished from real data. The user specifies n, and at each timestamp the method reports the real location along with $n - 1$ dummies. At the first timestamp, $n - 1$ dummies are randomly generated over the map. The location of each previously reported dummy is memorized, and new dummies are generated around the memory. Specifically, given a dummy's coordinates $(lat, long)$ at time t, a new dummy is generated for time $t + 1$ by sampling uniform at random $lat' \in [lat - m, lat + m]$ and $long' \in [long - m, long + m]$. The parameter m controls the maximum distance between the new dummy and the previously reported dummy. In our evaluation, we vary m to study its impact on utility and privacy. To be comparable to other methods, we randomly select one out of n trajectories as the sanitized trajectory for each user.

[2]https://www.openstreetmap.org

[3]https://github.com/chatziko/location-guard

Table 1: Dataset Summary

Dataset	#Users	Frequency	Resolution	Avg. # Traj's	Avg. # Loc's
GeoLife[25]	182	1 to 5 seconds	182×182	54	15640
RioBuses[6]	14149	every minute	170×170	9	2661

Table 2: Default Parameter Settings

Privacy Method	Parameter	Privacy Method	Parameter
Laplace	$\epsilon = 0.02$	MN	n=5, $m = 10^{-5}$
SpotME	$p = 10^{-5}$	VHC	$\sigma = 50$
Rounding	$s = 200$	Spatial Cloaking	$r = 500$, $R = 1000$
Noise	$var = 5000$		

4 EXPERIMENTS

4.1 Experiment Methodology

Datasets. We adopt two real-world datasets in our evaluation: GeoLife [25] and RioBuses [6]. The GeoLife dataset includes GPS data of 182 users in Beijing with a total duration of more than 48000 hours. The trajectories were recorded with a variety of sampling frequencies, and the majority of data was logged every 1 to 5 seconds. The Rio Buses dataset contains trajectory of 14149 buses in October 2010 from the city of Rio de Janeiro. The real-time GPS data is updated every minute. For the purpose of the evaluation, we consider data from 200 randomly selected buses (also referred to as users). Note that users may contribute different numbers of locations and trajectories to the datasets recorded on different dates. We summarize the characteristics of two datasets in Table 1.

Pre-processing. For both datasets, we subsample each raw trajectory every 5 minutes and impute missing values with the last known location. We discretize the map range of each dataset into roughly $300m \times 300m$ grid cells and the spatial resolution for each dataset is reported in Table 1. We infer the *home* location of a user in GeoLife as the location in which the user was found most often at 02:00, 06:00 and 20:30 during weekdays. In the RioBuses dataset, we assign the home location of every user to the Central bus station in Brazil. For every user in each dataset, we compute and store the 10 most visited locations as the user's *significant* locations.

Settings All location privacy methods are implemented in Java and are publicly available [8]. The default parameter values are reported in Table 2. To evaluate the mobility metrics, we randomly select a 30-minute window (i.e., 6 consecutive 5-minute intervals) for each user. For each mobility metric, we report the relative error between the value computed on the sanitized trajectory and that of the real trajectory. The experiments were conducted on a Linux workstation with a 3.50 GHz Intel processor and 16 GB Java heap space. Results were averaged among users in each dataset.

4.2 Varying Privacy Parameters

We first study the utility and privacy trade-off for every location privacy method. Specifically, we evaluate the impact on utility by varying the privacy parameter for each location privacy method. As the window for evaluating mobility metrics is randomly selected for each user and several location privacy methods adopt randomized mechanisms, we also report the average relative error across all mobility metrics for the overall trend. In addition to mobility metrics, we report the average Hamming distance and Haversine distance (in meters) between locations in real and sanitized trajectories. Intuitively, 1 in Hamming distance indicates the location has been altered by the privacy method, and unaltered otherwise. The Haversine distance indicates how far away the output location is from the input. The results for GeoLife are reported in Tables 3,4,5,6,7,8,9. Results for RioBuse can be found in Appendix A.2. As our observations are consistent in both datasets, our discussions focus on the results of GeoLife.

Table 3: Laplace Utility Experiment - GeoLife

Utility/Params	ϵ					
	0.001	**0.01**	**0.02**	**0.04**	**0.05**	**0.1**
Hamming	0.74	0.41	0.20	0.03	0.01	0.00
Haversine (in m)	1494.96	121.57	46.52	7.34	2.83	0.02
Tot Dist (in %)	99.18	91.16	73.06	18.14	6.43	0.00
Max Dist (in %)	98.25	89.98	72.27	17.88	6.15	0.00
Std Dev Displacement (in %)	98.74	85.26	58.36	12.85	4.47	0.00
Max Dist Home (in %)	69.94	26.41	16.68	2.97	0.00	0.00
Rad Gyration (in %)	98.02	89.84	72.40	17.92	6.18	0.00
# Diff Places (in %)	96.87	90.66	72.94	18.16	6.15	0.00
# Significant Places (in %)	14.97	22.75	12.57	3.59	1.20	0.00
Avg Mobility Error (in %)	82.28	70.87	54.04	13.07	4.37	0.00

Table 4: SpotME Utility Experiment - GeoLife

Utility/Params	p					
	10^{-7}	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
Hamming	0.00	0.02	0.08	0.15	0.51	0.71
Haversine (in m)	28.79	280.14	1301.12	2482.79	8560.14	11870.35
Tot Dist (in %)	1.68	11.17	38.55	64.80	98.88	100.00
Max Dist (in %)	1.68	11.17	38.55	64.52	98.76	100.00
Std Dev Displacement (in %)	1.12	9.50	27.93	51.40	94.10	99.44
Max Dist Home (in %)	0.00	10.67	28.95	60.08	94.65	93.96
Rad Gyration (in %)	1.68	11.17	38.55	64.45	98.77	100.00
# Diff Places (in %)	1.68	10.34	36.78	60.99	97.18	95.47
# Significant Places (in %)	0.00	0.00	0.00	0.00	2.10	9.68
Avg Mobility Error (in %)	1.12	9.15	29.90	52.32	83.49	85.51

Table 5: Noise Utility Experiment - GeoLife

Utility/Params	var					
	1000	**3000**	**5000**	**10000**	**15000**	**40000**
Hamming	0.00	0.10	0.25	0.49	0.62	0.84
Haversine (in m)	0.33	21.93	54.27	112.40	148.46	253.04
Tot Dist (in %)	2.23	37.91	75.50	98.35	94.89	95.82
Max Dist (in %)	2.23	36.92	73.07	96.13	92.66	94.18
Std Dev Displacement (in %)	1.12	30.66	64.73	95.25	93.11	96.33
Max Dist Home (in %)	0.00	3.48	12.61	29.87	25.65	32.86
Rad Gyration (in %)	2.23	36.61	72.28	95.13	92.35	93.46
# Diff Places (in %)	2.23	38.01	74.30	97.67	94.88	95.59
# Significant Places (in %)	0.60	5.69	17.07	34.53	31.34	38.47
Avg Mobility Error (in %)	5.42	27.04	55.65	78.13	74.98	78.10

Table 6: Rounding Utility Experiment - GeoLife

Utility/Params	s				
	200	**300**	**400**	**500**	**1000**
Hamming	0.50	0.79	0.89	0.97	0.98
Haversine (in m)	115.99	191.67	254.00	365.82	728.39
Tot Dist (in %)	1.24	0.98	2.54	2.40	3.88
Max Dist (in %)	1.27	0.94	2.53	2.18	3.73
Std Dev Displacement (in %)	1.78	0.91	1.70	2.25	2.73
Max Dist Home (in %)	5.50	19.00	22.93	26.15	35.87
Rad Gyration (in %)	1.19	1.10	2.56	2.50	3.85
# Diff Places (in %)	0.47	0.14	1.16	1.09	1.74
# Significant Places (in %)	17.07	33.83	38.22	36.53	39.52
Avg Mobility Error (in %)	4.07	8.13	10.24	10.44	13.05

For every location privacy method, as the privacy level increases, we observe a decrease in utility as expected. Taking Laplace as an example (Table 3), when ϵ takes smaller values (i.e., stronger privacy), higher Hamming and Haversine distances as well as larger errors for mobility metrics are reported. On the other hand, with $\epsilon = 0.1$, we observe no distortion in the output locations and perfect mobility metrics. Similar trends are observed in SpotME and Noise (see Tables 4 and 5).

Large Haversine distances are reported based on the output of SpotME, Spatial Cloaking, and MN (see Tables 4,7,and 8). The reason is SpotME and MN may output locations randomly selected

Table 7: Spatial Cloaking Utility Experiment - GeoLife

Utility/ Params	R					
	1000	**1500**	**2500**	**5000**	**7500**	**10000**
Hamming	0.42	0.46	0.56	0.71	0.78	0.83
Haversine (in m)	9559.75	12534.77	17843.88	29974.84	34314.19	39228.85
Tot Dist (in %)	4.95	4.00	4.00	18.08	17.33	24.89
Max Dist (in %)	4.74	4.00	4.00	18.02	17.33	24.89
Std Dev Displacement (in %)	4.13	2.67	4.00	17.33	16.00	22.94
Max Dist Home (in %)	23.29	36.25	43.62	48.18	59.30	63.97
Rad Gyration (in %)	4.80	4.00	4.00	17.95	17.33	24.72
# Diff Places (in %)	4.00	2.67	4.00	14.11	15.51	23.18
# Significant Places (in %)	17.12	21.92	24.66	32.88	35.62	36.99
Avg Mobility Error (in %)	9.00	10.79	12.61	23.79	25.49	31.65

Table 8: MN Utility Experiment - GeoLife

Utility/Params	m				
	10^{-6}	10^{-5}	10^{-4}	**0.001**	**0.01**
Hamming	0.79	0.78	0.79	0.79	0.82
Haversine (in m)	12508.64	14770.50	12131.68	13105.17	13466.07
Tot Dist (in %)	7.26	13.39	19.21	66.17	77.88
Max Dist (in %)	7.26	13.27	19.36	66.26	77.65
Std Dev Displacement (in %)	3.91	11.00	11.73	49.67	79.98
Max Dist Home (in %)	66.23	54.86	71.20	72.62	73.71
Rad Gyration (in %)	7.26	13.27	19.35	66.19	77.71
# Diff Places (in %)	4.54	9.58	17.89	64.32	78.55
# Significant Places (in %)	40.72	37.85	35.33	33.53	35.33
Avg Mobility Error (in %)	19.60	21.89	27.72	59.82	71.54

Table 9: VHC Utility Experiment - GeoLife

Utility/Params	σ					
	10	**50**	**100**	**300**	**500**	**1000**
Hamming	0.79	0.79	0.80	0.86	0.91	0.96
Haversine (in m)	225.07	235.35	257.87	345.15	438.29	635.58
Tot Dist (in %)	9.42	18.72	34.36	68.90	89.08	97.74
Max Dist (in %)	8.92	18.30	34.15	67.45	87.09	96.25
Std Dev Displacement (in %)	7.08	17.89	25.72	56.82	79.12	96.20
Max Dist Home (in %)	19.24	20.81	18.45	26.63	33.15	39.07
Rad Gyration (in %)	9.10	18.07	33.80	67.58	86.65	96.37
# Diff Places (in %)	8.27	15.51	29.61	66.96	83.54	95.11
# Significant Places (in %)	38.12	36.53	38.02	36.83	38.02	35.93
Avg Mobility Error (in %)	14.31	20.83	30.59	55.88	70.95	79.52

over the entire map. As for Spatial Cloaking, locations near the user's home are deleted according to the parameter R. We impute the deleted locations in a trajectory using the closest remaining location. For some users and some R values, the entire trajectory or multiple trajectories may be deleted. When a trajectory is deleted, we consider that Hamming distance 1 and maximum Haversine distance per dataset are incurred for every timestamp, hence the high Haversine distances[4].

The mobility metric errors indicate how well the output trajectories reflect the real users' mobility patterns. Lower distortions (i.e., Hamming and Haversine distances) often lead to lower mobility errors, as seen in Tables 3, 4, 5, and 7. Interesting observations are made for Rounding, MN, and VHC (see Tables 6,8,and 9). In Rounding, we see the Hamming distance steadily increases to 0.98 as s grows to 1000 but the Haversine distance grows moderately and the average mobility error reaches only 13.05% with $s = 1000$. The reason is that Rounding may change an input location to a nearby cell without significantly disturbing the mobility patterns, hence low Haversine distances and low mobility errors. For the MN method, as the offset increases, dummies in a trajectory may be further away from each other, which differ from the real users' mobile behaviors and result in higher mobility errors (71.54% when offset= 0.01). However, the Hamming distance and the Haversine distance stay at high values (e.g., 0.78-0.82 for Hamming) as the offset varies. This shows that the dummy trajectories reported by the method with random starting locations largely differ from the real trajectories. As for VHC, we also see the Hamming distance increases with σ to higher values (e.g., 0.96 when $\sigma = 1000$) while the Haversine distance grows moderately, similar to Rounding. As VHC partitions the geospace according to road networks, multiple grid cells in D may be mapped to the same VHC partition and the output location is likely to be different from the input, even when perturbations are low (i.e., smaller σ values).

[4]For RioBuses dataset, Spatial Cloaking does not delete as many locations as for GeoLife as buses travel to more locations outside the "home" area (i.e., central bus station).

Table 10: Avg Mobility Error (in %) as Varying Window - GeoLife

Privacy Method	Window size				
	6	**12**	**24**	**48**	**96**
Laplace	50.33	67.29	72.75	69.91	65.24
SpotME	52.54	72.40	81.65	83.83	83.85
Noise	60.88	69.45	71.79	69.48	65.17
Rounding	4.49	4.12	4.53	5.51	6.66
MN	22.83	23.21	27.82	32.78	45.17
VHC	18.92	22.03	21.63	25.61	25.20
Spatial Cloaking	11.70	12.46	11.52	15.74	16.14

Table 11: Avg Mobility Error (in %) as Varying Window - RioBuses

Privacy Method	Window size				
	6	**12**	**24**	**48**	**96**
Laplace	58.16	60.54	61.17	58.69	55.83
SpotME	45.98	62.68	77.01	80.42	80.35
Noise	59.79	64.00	60.50	59.31	53.80
Rounding	5.79	6.19	8.79	8.34	9.26
MN	23.00	27.12	32.84	40.03	44.76
VHC	11.23	13.60	14.75	15.89	17.20
Spatial Cloaking	1.85	1.85	2.26	2.46	2.71

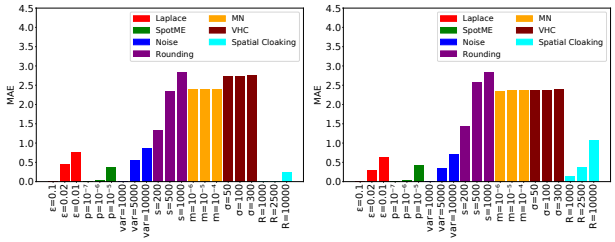
4.3 Varying Window Size

The mobility metrics can be measured for time windows of various lengths. A shorter time window reflects a user's short-term behavior and vice versa. For instance, the authors of [5] considered 1 and 14 days. As datasets adopted in our work present a large variation of available data per user, we will vary the number of 5-minute intervals in the following experiment. In Tables 10 and 11, window size of 6 equates to a 30-minute time window and window size of 96 equates to an 8-hr time window.

As the window size increases, we expect increasing average mobility errors as the effect of location privacy accumulates. Exceptions are observed in Laplace and Noise, where the mobility errors first increase and decrease as the window size continues to grow. We hypothesize that this phenomenon is a result of zero-mean noise distributions adopted by both methods. As the window becomes larger, the noise aggregated over a larger set of locations may cancel out. Between two datasets, the location privacy methods behave similarly when increasing the window size. Since the datasets differ in spatial distributions and buses and human users exhibit different mobility behaviors, the same location privacy method may yield different levels of mobility errors between datasets. For example, Spatial Cloaking inflicts smaller mobility errors in RioBuses (Table 11) due to less data deleted.

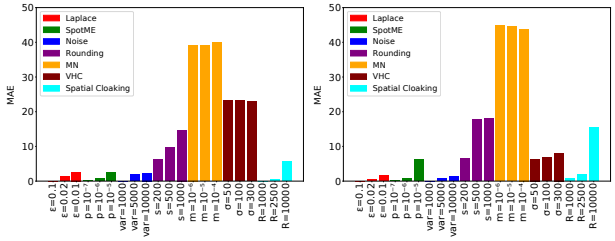
4.4 Frequency Estimation and Range Queries

Frequency estimation and range queries can enable a range of real-time applications, e.g., traffic monitoring and social distancing monitoring. As they are more useful when a large set of users are present, we generate synthetic users for this experiment with 182×182 grid and two types of distributions: *uniform* and *simulated*. For



(a) *Uniform* (b) *Simulated*

Figure 3: Accuracy of Frequency Estimation (best in color and zoomed).



(a) *Uniform* (b) *Simulated*

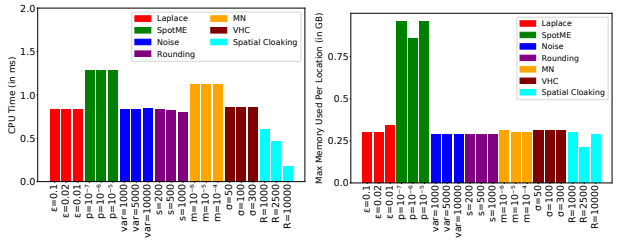
Figure 4: Accuracy of Range Queries (best in color and zoomed).

uniform, each synthetic user's location (home as well[5]) is sampled from the grid uniformly at random; for *simulated*, each synthetic user's location is sampled according to the location popularity among GeoLife users and the synthetic user's home is drawn from the home locations of GeoLife users. We generate 50000 synthetic users for this experiment.

Figure 3 shows the accuracy results for frequency estimation for *uniform* and *simulated* distributions. In both cases, we observe the MAE of location privacy methods correlates with the Hamming distance measure reported previously. Methods and parameter values that yield high Hamming distances also yield high MAE for frequency estimation. VHC performs relatively better in the simulated case, as the location popularity is related to road network connectivity, which serves as the basis for VHC's space partitioning. Spatial Cloaking inflicts higher MAE error in the simulated case, as those synthetic users exhibit real-world behaviors, i.e., more likely to visit locations near home thus more data deleted by the method.

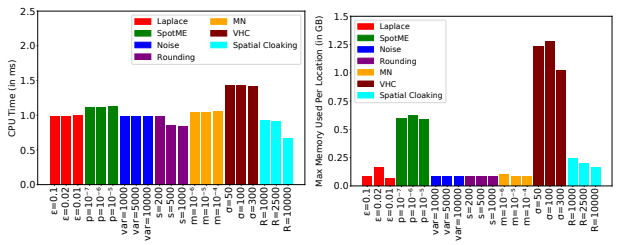
Figure 4 shows the accuracy results for range queries. We randomly generate $1km \times 1km$ query windows in the range of the synthetic dataset and report the average MAE among 100 queries. As each query covers a larger geographic area than a grid cell, the MAE of range queries is higher than that of frequency estimation. When comparing location privacy methods, we observe that Rounding and VHC perform better than MN for answering range queries. Recall that MN produces dummy trajectories around randomly selected starting locations in the map, whereas Rounding and VHC may report sanitized locations near the real locations. In the simulated case, VHC shows much reduced MAE errors, and Spatial Cloaking inflicts higher MAE errors, which are consistent with our observations in frequency estimation. In summary, Laplace, SpotME, Noise, and Spatial Cloaking, in weaker privacy settings, are most accurate for frequency estimation and range queries.

[5]Note that a home location is needed for each synthetic user which is protected by Spatial Cloaking.



(a) CPU Time (b) Peak Memory

Figure 5: CPU Time and Peak Memory - GeoLife (best in color and zoomed).



(a) CPU Time (b) Peak Memory

Figure 6: CPU Time and Peak Memory - RioBuses (best in color and zoomed).

4.5 CPU and Memory Evaluation

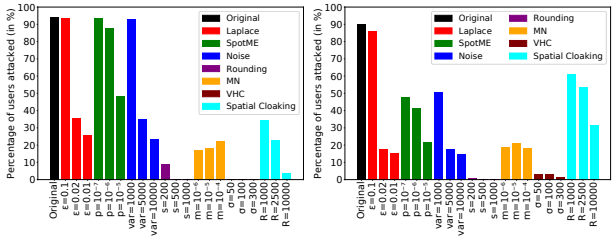
The computational overhead is an important consideration for the adoption of location privacy, as the location privacy method runs on the client. It is essential to evaluate the overheads with real-world datasets which reflects realistic mobility behaviors. To this end, we measure the average CPU time (in ms) to sanitize one location and the peak memory consumption (in GB) by each privacy method with ThreadMXBean in Java. We run each location privacy method with several parameter settings to observe the impact of the parameters. Our results are reported in Figures 5, 6 for two datasets respectively.

In CPU time measurements, all methods are very efficient, i.e., taking less than 2 ms on average to sanitize one location. We see that SpotME, MN, and VHC are higher than other methods, due to additional computations: SpotME iterates over every grid cell to generate dummies; MN produces dummy locations for every dummy trajectory; and VHC accesses the indexing structure for space partitioning. The change in the parameter values does not seem to significantly affect the CPU time for location privacy.

The memory requirements of location privacy methods are also reasonable, i.e., peak memory usage under 1.5 GB for both datasets. SpotME requires more memory than other methods, as a location map is generated for each time interval. We also observe higher CPU and Memory requirements for the VHC method in RioBuses, because a larger number of partitions are present in this dataset, e.g., 8236 partitions in RioBuses vs. 3157 for GeoLife.

4.6 Inference Attack Evaluation

The inference attack exploits the quality of the sanitized dataset. It assumes that the adversary knows all but one locations a user has visited. Note that duplicate visits are not considered in the attack model, i.e., the adversary's knowledge is formulated as a set. Our results show that with real data, 94.41% users in GeoLife and 90% users in RioBuses are attacked (see "Original" bars in Figure 7). We



(a) GeoLife (b) RioBuses

Figure 7: Inference Attack Results (best in color and zoomed).

run the attacks on the sanitized trajectories of each method and report the results in Figure 7.

We see that the success of the inference attack correlates with the Hamming distance measure. Location privacy methods and parameters that yield lower Hamming distances are more prone to inference attacks. Take Laplace for example: when $\epsilon = 0.1$, the inference attacks can be carried out on the sanitized data almost as successfully as they can be on real data. As ϵ is reduced (i.e., stronger privacy), the attack success decreases with it. Due to the differences in spatial distribution between two datasets, SpotME and Noise have different effects on the inference attacks, which are consistent with their Hamming distance results. As users exhibit different mobility behaviors in two datasets, Spatial Cloaking is less effective against inference attacks for RioBuses, as multiple buses run the same route. For the MN method, the inference success is around 20% for both datasets as the method maintains $n = 5$ trajectories for each input trajectory, i.e., creating 4 dummy trajectories. The chance of picking the real trajectory is 20%. Methods that inflict high Hamming distances, including Rounding and VHC, are not vulnerable to inference attacks, e.g., 0% success for VHC in GeoLife.

4.7 Re-identification Attack Evaluation

Re-identification attacks aim to uniquely identify users in the sanitized datasets. In our evaluation, we measure the minimum number (denoted by k) of any real locations visited by a user the adversary needs to know, in order to uniquely identify the user. The larger k is, the more difficult the re-identification attack is, as the adversary must obtain more prior knowledge in order to successfully launch an attack. If a user cannot be uniquely identified, e.g., due to visiting only locations also visited by others, we report such users as "NRI" in our experiments. Figures 8 and 9 depict the re-identification results for all location privacy methods in GeoLife and RioBuses, respectively. For each k value, the y-axis indicates the percentage of users that can be re-identified if the adversary knows any k real locations visited by the user.

In Figures 8 and 9, the blue bars indicate re-identification attack results launched on the real trajectories. In GeoLife, 68.42% users can be re-identified with ≤ 4 locations; in RioBuses, 67% users can be re-identified with ≤ 5 locations. 18.13% users in GeoLife cannot be uniquely re-identified, and 25% users in RioBuses cannot be uniquely re-identified. Note that for Spatial Cloaking, the attack success rate is calculated for those GeoLife users if their home locations can be inferred; 63.9% users can be re-identified with ≤ 4 locations, and 20.83% users cannot be uniquely re-identified.

Similar to inference attacks, the success of re-identification correlates to the quality of the dataset. For methods and parameter values that lead to high Hamming distances, the re-identification attacks are less successful. Almost all users become non re-identifiable after sanitization in Figures 8f and 9d. When increasing privacy levels of the location privacy methods, we observe the attacks become more difficult. For example, in Figure 8a, for a given k, the percentage of re-identified users decreases as ϵ decreases; more users are non re-identifiable as ϵ decreases. For MN, we observe the percentage of non re-identifiable users is around 80% as we vary the offset for both datasets. This is consistent with other empirical results with the same parameter setting ($n = 5$). For Spatial Cloaking, the re-identification attack is more successful on RioBuses dataset than on GeoLife (see Figure 8g vs Figure 9g). This is also consistent with our previous observations: users in RioBuses visit more locations outside home, and more real data is preserved after sanitization.

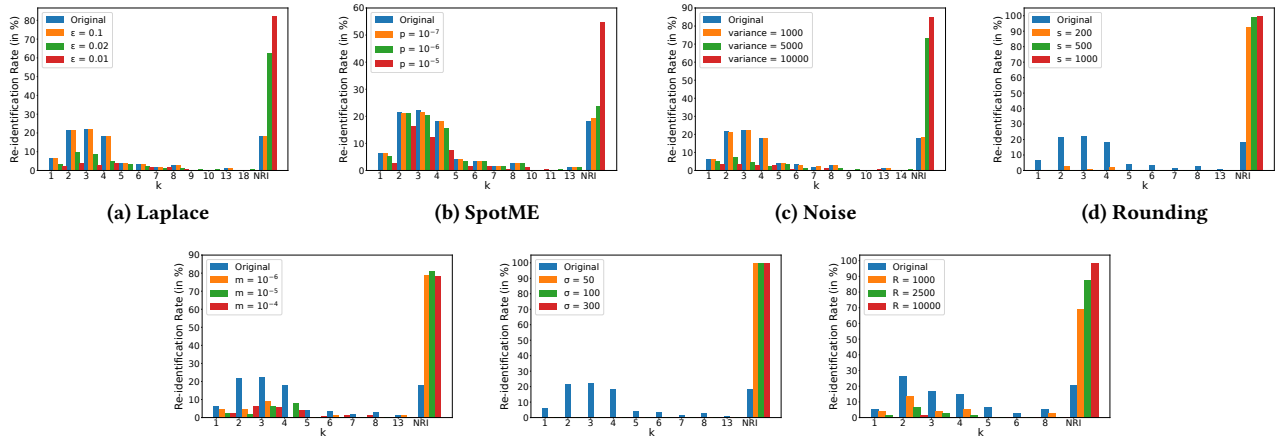
4.7.1 Improved Re-identification Attack for Rounding. From results in Figures 8d and 9d, it may seem that Rounding offers high empirical privacy protection; as the method largely alters the input locations, the adversary's prior knowledge about the target individual cannot be matched with the sanitized data. However, it may be overlooked by some that Rounding is a deterministic approach (and the *only* one among all methods considered). Therefore, the adversary can apply the same Rounding method to known locations of the target and obtain the exact sanitized locations, which can be used for launching the re-identification attack. Based on this observation, we conduct the improved re-identification attack experiment for Rounding and report the results in Figure 10.

In short, Rounding is less effective against the improved attack. The percentage of non re-identifiable users after Rounding is much lower, and a significant amount of users can be re-identified at small k values, compared to in the basic attack (see Figures 8d and 9d). Nonetheless, as Rounding generalizes users' locations, the attack becomes harder as s increases. Compared to "Original", increasing s lowers the percentages of users who can be re-identified with a small number of locations (see $k = 1, 2$ in Figure 10a and $k = 1, 2, 3$ in Figure 10b); higher percentages of users can be re-identified if the adversary knows more (see $k = 5, 6$ in Figure 10a and $k = 6, 7, 8, 9$ in Figure 10b).

5 DISCUSSIONS

Below we provide our interpretation of the evaluation, discuss location privacy in the context of application needs, and point out considerations for practical adoption.

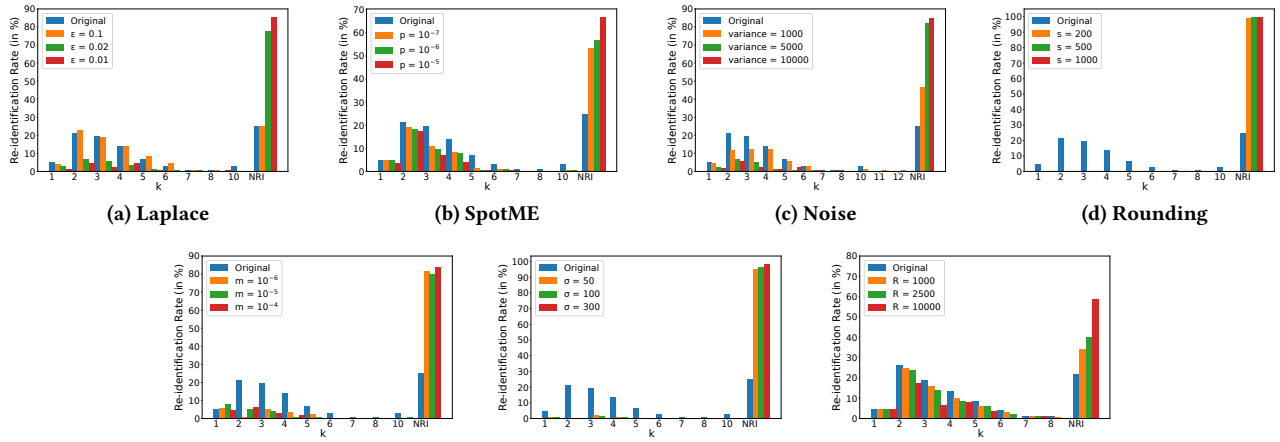
Interpreting the results. From the *computational* perspective, all methods considered in the study are efficient in CPU time and memory requirements. From the *utility* perspective, we see that lower distortions (e.g., Hamming and Haversine distances) lead to more accurate frequency estimation, range queries, and mobility metrics. But the condition is not necessary: accurate frequency estimation, range queries, and/or mobility metrics can be achieved despite higher distortions. Take Spatial Cloaking ($R = 1000$) for example, it inflicts a 42% Hamming distance, similar to that of Laplace ($\epsilon = 0.01$), but provides much more accurate mobility metrics (see Tables 3 and 7); it is also the most accurate in frequency estimation and range queries (see Figures 3 and 4). From the *privacy* perspective, we see that higher distortions lead to lower attack success rates, despite the difference in underlying privacy models



(a) Laplace (b) SpotME (c) Noise (d) Rounding

(e) MN (f) VHC (g) Spatial Cloaking

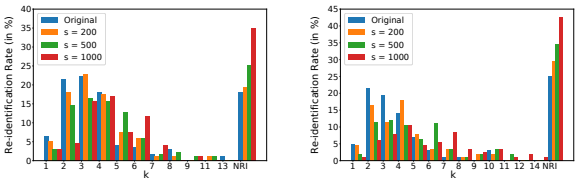
Figure 8: Re-identification Attack Results - GeoLife (best in color and zoomed).



(a) Laplace (b) SpotME (c) Noise (d) Rounding

(e) MN (f) VHC (g) Spatial Cloaking

Figure 9: Re-identification Attack Results - RioBuses (best in color and zoomed).



(a) GeoLife (b) RioBuses

Figure 10: Improved Re-identification Attack for Rounding (best in color and zoomed).

(e.g., Laplace vs. Spatial Cloaking). A special case has been identified for deterministic methods, e.g., Rounding, where an adversary can utilize knowledge about the method to launch an improved attack.

How to choose. We recognize that the priorities of location-based applications may vary greatly. Below we discuss a few use cases for location privacy. For applications that require provable privacy guarantees, Laplace can be considered as it is based on differential privacy [7]; it is also light-weight in computation, and tunable to balance utility and empirical privacy risks. When obtaining truthful data is essential (e.g., social distancing monitoring), applications may consider Spatial Cloaking, which deletes data deemed sensitive by the user and can achieve high utility in mobility metrics, frequency estimation, and range queries; furthermore, it offers comparable empirical privacy risks (i.e., in inference and re-identification attacks) to those of Laplace. For lowering inference and re-identification risks, VHC is the best among all methods; it also answers range queries with accuracy for realistic data distributions. For applications that can tolerate some communication and computation overheads, MN can be considered; the server receives real data along with the dummies, but the dummy trajectories are difficult to filter and the results show that randomly picking one sanitized trajectory incurs low inference and re-identification risks.

Practical considerations. There is a lot to consider when deploying location privacy methods in the field. Here we discuss two points. The first is the possibility of *advanced adversaries*, who have access to more data and/or computational resources. For example,

SpotME [18] discussed the possibility of localization, where an adversary can observe the reported location maps over time and filter out dummy locations. As another example, with access to a randomized location privacy method, an adversary may repeatedly run the method, learn the output probabilities $\Pr(l'|l)$ empirically, and launch inference and re-identification attacks with the most likely outputs of the known locations. Moreover, given the learned probabilities, the adversary may be able to estimate the most likely real location by observing a sanitized location. The privacy risks in the presence of such advanced adversaries may differ from the results of this study. The second is that *additional computation* may be needed when location privacy methods are used in practice. For instance, the privacy budget ϵ for Laplace can be optimized for utility; this optimization process is usually separate from the location sanitization process. Another example is that the grids (and the space partitioning structure for VHC) need to be updated when users move out of the current map; a server-side procedure is needed to detect those events confidently using sanitized locations.

6 CONCLUSION

We have presented a comparative evaluation for a range of local, online location privacy methods with real-world location datasets. The usefulness of location privacy is demonstrated with distortion measures, frequency estimation and range queries, as well as computing mobility metrics. To evaluate the practical privacy protection, we design and conduct two types of attacks. We discuss results obtained, highlight different application needs, and point out considerations for adopting location privacy.

Our study opens up several directions for future work. Firstly, it would be beneficial to showcase the usefulness of location privacy in research studies, e.g., predicting mental health states using sanitized location data. Due to the fine granularity and high sensitivity, data that contains both trajectories and mental health labels is not widely available. It is desirable for future research to collect location data and survey data from human subjects while protecting their privacy [9]. Secondly, it is beneficial to extend the evaluation to recent methods that require more computation initially (e.g., [4]) or during sanitization (e.g., [1, 22, 24]). As the datasets exhibit large domains, e.g., 182×182 grid cells, methods that demonstrate efficiency improvements, e.g.,[2], will be considered. Last but not least, new privacy challenges may arise for applications with large user sets. For instance, when trajectories from different users overlap, inference risks may be higher due to stronger correlation between locations. We consider improving the current location privacy methods in the context of new privacy challenges.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their suggestions. This work has been supported in part by NSF CNS-1951430 and UNC Charlotte. The opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

REFERENCES

[1] Berker Agir, Thanasis G. Papaioannou, Rammohan Narendula, Karl Aberer, and Jean-Pierre Hubaux. 2014. User-side adaptive protection of location privacy in participatory sensing. *GeoInformatica* 18, 1 (01 Jan 2014), 165–191.

[2] Ritesh Ahuja, Gabriel Ghinita, and Cyrus Shahabi. 2019. A Utility-Preserving and Scalable Technique for Protecting Location Data with Geo-Indistinguishability. In *Advances in Database Technology - 22nd International Conference on Extending Database Technology, EDBT 2019, Lisbon, Portugal, March 26-29, 2019.* 217–228.

[3] Miguel E. Andrés, Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geo-indistinguishability: Differential Privacy for Location-based Systems. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security (CCS '13).* ACM, New York, NY, USA, 901–914.

[4] Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2014. Optimal Geo-Indistinguishable Mechanisms for Location Privacy. In *CCS - 21st ACM Conference on Computer and Communications Security (CCS '14).* ACM, Scottsdale, Arizona, United States, 251–262.

[5] Luca Canzian and Mirco Musolesi. 2015. Trajectories of depression: unobtrusive monitoring of depressive states by means of smartphone mobility traces analysis. In *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing.* 1293–1304.

[6] Daniel S Dias, Luís Henrique MK Costa, and Marcelo Dias de Amorim. 2016. Capacity analysis of a city-wide V2V network. In *2016 7th International Conference on the Network of the Future (NOF).* IEEE, 1–3.

[7] Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation.* Springer, 1–19.

[8] Liyue Fan and Sriram Yechan Gunja. 2020. Geopriv4j: an open source repository for practical location privacy. In *Proceedings of the Sixth International ACM SIGMOD Workshop on Managing and Mining Enriched Geo-Spatial Data.* 1–6.

[9] Matt Hanson, Lisa E Baranik, Rachel W Smith, and Liyue Fan. 2021. Development of the refugee job search stressor scale. In *Academy of Management (AoM) Annual Meeting.*

[10] H. Kido, Y. Yanagisawa, and T. Satoh. 2005. An anonymous communication technique using dummies for location-based services. In *ICPS '05. Proceedings. International Conference on Pervasive Services, 2005.* 88–97.

[11] John Krumm. 2007. Inference Attacks on Location Tracks. In *Pervasive Computing*, Anthony LaMarca, Marc Langheinrich, and Khai N. Truong (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 127–143.

[12] John Krumm. 2009. A survey of computational location privacy. *Personal and Ubiquitous Computing* 13, 6 (2009), 391–399.

[13] Justin J Levandoski, Mohamed Sarwat, Ahmed Eldawy, and Mohamed F Mokbel. 2012. Lars: A location-aware recommender system. In *2012 IEEE 28th international conference on data engineering.* IEEE, 450–461.

[14] Kristopher Micinski, Philip Phelps, and Jeffrey S Foster. 2013. An empirical study of location truncation on android. In *IEEE Computer Society Security and Privacy Workshops Mobile Security Technologies (MoST '13).*

[15] Stephanie K Pell and Christopher Soghoian. 2012. Can You See Me Now: Toward Reasonable Standards for Law Enforcement Access to Location Data that Congress Could Enact. *Berkeley Tech. LJ* 27 (2012), 117.

[16] A. Pingley, W. Yu, N. Zhang, X. Fu, and W. Zhao. 2009. CAP: A Context-Aware Privacy Protection System for Location-Based Services. In *2009 29th IEEE International Conference on Distributed Computing Systems.* 49–57.

[17] Vincent Primault, Antoine Boutet, Sonia Ben Mokhtar, and Lionel Brunie. 2019. The Long Road to Computational Location Privacy: A Survey. *IEEE Communications Surveys Tutorials* 21, 3 (2019), 2772–2793. https://doi.org/10.1109/COMST.2018.2873950

[18] D. Quercia, I. Leontiadis, L. McNamara, C. Mascolo, and J. Crowcroft. 2011. SpotME If You Can: Randomized Responses for Location Obfuscation on Mobile Phones. In *2011 31st International Conference on Distributed Computing Systems.* 363–372.

[19] Sohrab Saeb, Mi Zhang, Christopher J Karr, Stephen M Schueller, Marya E Corden, Konrad P Kording, and David C Mohr. 2015. Mobile phone sensor correlates of depressive symptom severity in daily-life behavior: an exploratory study. *Journal of medical Internet research* 17, 7 (2015), e175.

[20] SafeGraph. 2021. "Social Distancing Metrics," Accessed June 7, 2021. https://docs.safegraph.com/v4.0/docs/social-distancing-metrics

[21] Akane Sano, Sara Taylor, Andrew W McHill, Andrew JK Phillips, Laura K Barger, Elizabeth Klerman, and Rosalind Picard. 2018. Identifying Objective Physiological Markers and Modifiable Behaviors for Self-Reported Stress and Mental Health Status Using Wearable Sensors and Mobile Phones: Observational Study. *J Med Internet Res* 20, 6 (08 Jun 2018), e210.

[22] Yonghui Xiao, Li Xiong, Si Zhang, and Yang Cao. 2017. LocLok: Location Cloaking with Differential Privacy via Hidden Markov Model. *Proc. VLDB Endow.* 10, 12 (Aug. 2017), 1901–1904.

[23] Heng Xu and Sumeet Gupta. 2009. The effects of privacy concerns and personal innovativeness on potential and experienced customers' adoption of location-based services. *Electronic Markets* 19, 2-3 (2009), 137–149.

[24] Lei Yu, Ling Liu, and Calton Pu. 2017. Dynamic Differential Location Privacy with Personalized Error Bounds.. In *NDSS.*

[25] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. 2009. Mining interesting locations and travel sequences from GPS trajectories. In *Proceedings of the 18th international conference on World wide web.* 791–800.

A APPENDIX: ADDITIONAL MATERIAL

A.1 Mobility Metrics

Below we introduce a set of mobility metrics defined for a single user in the time period $[t_1, t_2]$ as in [5]. The metrics are computed based on the user's stop places: $\left(Pl_1, Pl_2, \ldots Pl_{N(t_1,t_2)}\right)$ and for every i, $Pl_i = \langle l_i, T_i \rangle$.

The **total distance covered** during the time interval $[t_1, t_2]$ is defined as:

$$\text{Tot Dist} = \sum_{i=1}^{N(t_1,t_2)-1} d\left(l_i, l_{i+1}\right), \tag{7}$$

where d is a distance measure between any 2 location pairs and we adopt the Haversine distance in this work.

The **maximum distance** assesses the maximum span of area covered by a user during the time interval $[t_1, t_2]$. Formally it is defined as:

$$\text{Max Dist} = \max_{i,j \in \{1,\ldots,N(t_1,t_2)\}} d\left(l_i, l_j\right) \tag{8}$$

The **standard deviation of the displacements** quantifies how the user moves from one place to the next. Displacement refers to the distance between places. Let $D_{dis} = \frac{\sum_{i=1}^{N(t_1,t_2)-1} d(l_i,l_{i+1})}{N(t_1,t_2)-1}$ be the average displacement. This metric is formally defined as:

$$\text{Std Dev Displacement} = \sqrt{\frac{1}{N\left(t_1,t_2\right)-1} \sum_{i=1}^{N(t_1,t_2)-1} \left(d\left(l_i, l_{i+1}\right) - D_{\text{dis}}\right)^2} \tag{9}$$

The **maximum distance from home** reports the furthest a user has been away from home during $[t_1, t_2]$. Let l_H be the home location of the user. Then,

$$\text{Max Dis Home} = \max_{i \in \{1,\ldots,N(t_1,t_2)\}} \left\{d\left(l_i, l_H\right)\right\} \tag{10}$$

The **radius of gyration** quantifies the deviation from the centroid of places visited by a user during a given time interval. Let T_i be the time spent at i-th place, $\tilde{T} = \sum_{i=1}^{N(t_1,t_2)} T_i$ be the total time spent and C_{cen} be the centroid of all places visited in $[t_1, t_2]$.

$$\text{Rad Gyration} = \sqrt{\frac{1}{\tilde{T}} \sum_{i=1}^{N(t_1,t_2)} T_i \cdot d\left(C_i, C_{\text{cen}}\right)^2} \tag{11}$$

The **number of different places** visited by a user during $[t_1, t_2]$ [6] is defined as:

$$\text{\# Diff Places} = \sum_{i=1}^{N(t_1,t_2)} \max\left\{1 - \sum_{j>i} \mathbb{1}_{i,j}, 0\right\} \tag{12}$$

where $\mathbb{1}_{i,j}$ is an indicator function which is equal to 1 if $l_i = l_j$, and 0 otherwise.

The **number of different significant places** quantifies how many significant places are visited during the time interval $[t_1, t_2]$. Assume the user's profile contains 10 significant places, and the associated locations are $l_{s_1}, \cdots, l_{s_{10}}$. This metric is formally defined as:

$$\text{\# Significant Places} = \sum_{j=1}^{10} \min\left\{\sum_{i=1}^{N(t_1,t_2)} \mathbb{1}_{i,s_j}, 1\right\} \tag{13}$$

A.2 Utility Experiments with RioBuses

Tables 12,13,14,15,16,17,18 present the Hamming distance, Haversine distance (in meters), and relative errors for mobility metrics obtained with the RioBuses dataset.

Table 12: Laplace Utility Experiment - RioBuses

Utility/Params	ϵ					
	0.001	**0.01**	**0.02**	**0.04**	**0.05**	**0.1**
Hamming	0.73	0.60	0.32	0.10	0.05	0.00
Haversine (in m)	1435.56	147.28	56.62	13.86	6.26	0.35
Tot Dist (in %)	96.40	86.94	78.64	44.86	23.12	1.00
Max Dist (in %)	87.83	85.27	77.45	44.12	23.04	1.00
Std Dev Displacement (in %)	94.56	88.20	71.41	34.48	18.05	0.52
Max Dist Home (in %)	20.46	5.26	2.33	0.79	0.15	0.02
Rad Gyration (in %)	87.84	85.66	77.54	44.66	23.22	1.00
# Diff Places (in %)	88.80	91.03	82.64	46.73	23.64	1.00
# Significant Places (in %)	13.15	21.52	21.19	7.28	5.30	0.00
Avg Mobility Error (in %)	69.86	66.27	58.74	31.85	16.64	0.65

Table 13: SpotME Utility Experiment - RioBuses

Utility/Params	p					
	10^{-7}	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
Hamming	0.15	0.16	0.20	0.26	0.55	0.72
Haversine (in m)	68.86	324.02	1405.59	2692.85	9506.33	13545.86
Tot Dist (in %)	1.82	10.74	34.19	58.82	97.51	100.00
Max Dist (in %)	1.80	10.75	33.68	58.94	97.52	100.00
Std Dev Displacement (in %)	2.06	7.50	25.94	48.29	94.56	99.36
Max Dist Home (in %)	2.27	8.23	28.11	49.47	93.43	98.21
Rad Gyration (in %)	1.81	10.74	33.74	58.89	97.52	100.00
# Diff Places (in %)	1.50	9.39	27.71	54.17	91.15	95.02
# Significant Places (in %)	9.27	9.60	11.92	7.95	17.44	21.30
Avg Mobility Error (in %)	2.93	9.57	27.90	48.07	84.16	87.70

Table 14: Noise Utility Experiment - RioBuses

Utility/Params	var					
	1000	**3000**	**5000**	**10000**	**15000**	**40000**
Hamming	0.05	0.27	0.42	0.62	0.72	0.88
Haversine (in m)	6.36	35.06	60.72	111.48	147.40	252.09
Tot Dist (in %)	19.71	69.11	81.10	87.18	91.17	90.59
Max Dist (in %)	19.56	68.71	78.95	84.68	87.96	88.05
Std Dev Displacement (in %)	16.74	62.19	77.52	90.42	94.34	92.94
Max Dist Home (in %)	0.44	1.44	2.57	3.19	5.55	8.09
Rad Gyration (in %)	19.68	68.84	79.02	84.82	87.84	87.62
# Diff Places (in %)	21.02	72.42	83.85	91.90	94.46	93.62
# Significant Places (in %)	6.29	20.09	26.16	23.18	35.43	35.21
Avg Mobility Error (in %)	14.78	51.83	61.31	66.48	70.96	70.88

Table 15: Rounding Utility Experiment - RioBuses

Utility/Params	s				
	200	**300**	**400**	**500**	**1000**
Hamming	0.85	0.86	0.97	0.99	0.99
Haversine (in m)	132.86	159.42	281.63	377.93	700.31
Tot Dist (in %)	0.28	0.81	1.10	2.90	1.71
Max Dist (in %)	0.25	0.77	0.88	2.61	1.63
Std Dev Displacement (in %)	0.64	1.81	5.27	1.72	5.21
Max Dist Home (in %)	2.20	1.90	5.98	8.83	10.63
Rad Gyration (in %)	0.26	0.97	1.12	2.60	1.43
# Diff Places (in %)	0.00	0.60	0.70	0.72	1.11
# Significant Places (in %)	32.89	43.49	47.68	47.02	54.97
Avg Mobility Error (in %)	5.22	7.19	8.96	9.49	10.96

[6] slightly modified from [5], by specifying $j > i$ in Equation 12

Table 16: Spatial Cloaking Utility Experiment - RioBuses

Utility/ Params	R					
	1000	**1500**	**2500**	**5000**	**7500**	**10000**
Hamming	0.04	0.06	0.08	0.12	0.22	0.34
Haversine (in m)	103.00	409.03	639.91	1305.57	3897.03	9573.24
Tot Dist (in %)	1.30	1.88	3.05	5.71	8.43	15.78
Max Dist (in %)	1.28	1.84	3.13	5.51	8.17	15.74
Std Dev Displacement (in %)	0.98	1.10	2.57	5.76	8.88	14.92
Max Dist Home (in %)	2.50	5.28	7.95	11.36	14.75	25.56
Rad Gyration (in %)	1.27	1.63	2.92	5.22	8.19	15.67
# Diff Places (in %)	0.71	1.26	1.72	3.63	6.78	13.63
# Significant Places (in %)	1.67	0.56	6.81	4.44	5.93	19.17
Avg Mobility Error (in %)	1.39	1.93	4.02	5.95	8.73	17.21

Table 17: MN Utility Experiment - RioBuses

Utility/Params	m				
	10^{-6}	10^{-5}	10^{-4}	**0.001**	**0.01**
Hamming	0.81	0.80	0.80	0.81	0.80
Haversine (in m)	16289.69	15257.02	16320.35	16958.12	16305.32
Tot Dist (in %)	12.00	16.99	26.65	79.63	72.74
Max Dist (in %)	12.00	16.99	26.25	80.35	74.49
Std Dev Displacement (in %)	11.00	12.50	20.65	68.72	73.61
Max Dist Home (in %)	70.71	57.35	72.81	72.19	68.15
Rad Gyration (in %)	12.00	16.99	26.17	80.35	74.34
# Diff Places (in %)	8.55	12.22	20.94	71.55	73.47
# Significant Places (in %)	41.06	44.07	40.40	39.07	40.40
Avg Mobility Error (in %)	23.90	25.30	33.41	70.27	68.17

Table 18: VHC Utility Experiment - RioBuses

Utility/Params	σ					
	10	**50**	**100**	**300**	**500**	**1000**
Hamming	0.71	0.72	0.74	0.83	0.89	0.94
Haversine (in m)	190.93	201.06	220.54	334.14	443.09	602.06
Tot Dist (in %)	2.10	11.50	28.05	75.81	84.31	89.74
Max Dist (in %)	2.13	11.20	26.78	74.53	82.68	87.04
Std Dev Displacement (in %)	0.89	11.43	26.01	61.32	74.38	92.87
Max Dist Home (in %)	2.50	3.38	4.07	8.30	7.92	10.03
Rad Gyration (in %)	2.08	11.44	26.96	74.27	82.62	86.67
# Diff Places (in %)	0.85	10.37	28.49	74.47	85.21	91.59
# Significant Places (in %)	39.40	37.75	36.75	39.51	40.62	45.53
Avg Mobility Error (in %)	7.14	13.87	25.30	58.32	65.39	71.92