Location Leakage in Federated Signal Maps

Evita Bakopoulou†, Mengwei Yang†, Jiang Zhang, Konstantinos Psounis, *Fellow, IEEE*, Athina Markopoulou, *Fellow, IEEE*

Abstract—We consider the problem of predicting cellular network performance (signal maps) from measurements collected by several mobile devices. We formulate the problem within the online federated learning framework: (i) federated learning (FL) enables users to collaboratively train a model, while keeping their training data on their devices; (ii) measurements are collected as users move around over time and are used for local training in an online fashion. We consider an honest-but-curious server, who observes the updates from target users participating in FL and infers their location using a deep leakage from gradients (DLG) type of attack, originally developed to reconstruct training data of DNN image classifiers. We make the key observation that a DLG attack, applied to our setting, infers the average location of a batch of local data, and can thus be used to reconstruct the target users' trajectory at a coarse granularity. We build on this observation to protect location privacy, in our setting, by revisiting and designing mechanisms within the federated learning framework including: tuning the FL parameters for averaging, curating local batches so as to mislead the DLG attacker, and aggregating across multiple users with different trajectories. We evaluate the performance of our algorithms through both analysis and simulation based on real-world mobile datasets, and we show that they achieve a good privacy-utility tradeoff.

Index Terms—Federated learning, deep leakage from gradients (DLG), location privacy, signal maps.

1 Introduction

Mobile crowdsourcing is widely used to collect data from a large number of mobile devices, which are useful on their own and/or used to train models for properties of interest, such as cellular/WiFi coverage, sentiment, occupancy, temperature, COVID-related statistics, etc. Within this broader class of spatiotemporal models trained by mobile crowdsourced data [1], we focus on the representative and important case of cellular signal maps. Cellular operators rely on key performance indicators (a.k.a. KPIs) to understand the performance and coverage of their network, in their effort to provide the best user experience. These KPIs include wireless signal strength measurements, especially Reference Signal Received Power (RSRP), which is going to be the focus of this paper, and other performance metrics (*e.g.,* coverage, throughput, delay) as well as information associated with the measurement (*e.g.,* location, time, frequency band, device type, etc.).

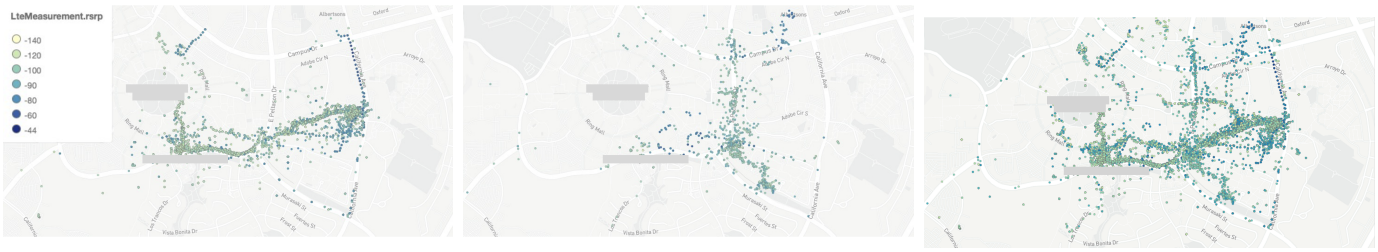
Cellular signal strength maps consist of KPIs in several locations. Traditionally, cellular operators collected such measurements by hiring dedicated vans (a.k.a. wardriving [2]) with special equipment, to drive through, measure and map the performance in a particular area of interest. However, in recent years, they increasingly outsource the collection of signal maps to third parties [3]. Mobile analytics companies (*e.g.,* OpenSignal [4], Tutela [5]) crowdsource measurements directly from end-user devices, via standalone mobile apps or SDKs integrated into popular partnering apps, typically games, utility or streaming apps. The upcoming dense deployment of small cells for 5G at metropolitan scales will only increase the need for accurate and comprehensive signal maps [6], [7]. Because cellular measurements are expensive to obtain, they may not be available for all locations, times and other parameters of interest, thus there is a need for signal maps prediction based on limited available such measurements.

Signal maps prediction is an active research area and includes: propagation models [8], [9], data-driven approaches [10], [11], [12], combinations thereof [13], and increasingly sophisticated ML models for RSRP [3], [14], [15], [16] and throughput [17], [18]. All these prediction techniques consider a centralized setting: mobile devices upload measurements to a server, which then trains a global model to predict cellular performance (typically RSRP) based on at least location, and potentially time and other features.

Fig. 1 depicts an example of a dataset collected on a university campus, which is one of the datasets we use throughout this paper. Fig. 1(a) and (b) show the locations where measurements of signal strength (RSRP) were collected by two different volunteers as they move around the campus. The measurements from all users are uploaded to a server, which then merges them and creates a signal map for the campus (shown in Fig. 1(c)); and/or may train a global model for predicting signal strength based on location and potentially other features. However, this utility comes at the expense of privacy: as evident in Fig. 1, frequently visited locations may reveal users' home, work, or other important locations, as well as their mobility pattern; these may further reveal sensitive information such as their medical conditions, political beliefs, etc [19]. The trajectories of the two example users are also sufficiently different from each other, and can be used to distinguish between them, even if their identifiers are removed from the dataset.

In this paper, we make three contributions (in the problem setup, privacy attack and defense mechanisms), all leveraging the patterns of human mobility underlying our

†*The first two co-authors made equal contributions. E. Bakopoulou was with the University of California, Irvine, when this work was conducted; she is currently with Google, Mountain View, CA. M. Yang and A Markopoulou (corresponding author) are with the University of California, Irvine. Email: {ebakopou, mengwey, athina}@uci.edu. J. Zhang and K. Psounis are with the University of Southern California. E-mail: {jiangzha, kpsounis}@usc.edu.*
This work has been supported by NSF Awards 1956393, 1900654, 1901488, and 1956435.



(a) RSRP measurements of one user. (b) RSRP measurements of another user. (c) RSRP measurements from all users.

Fig. 1: Examples from the `Campus Dataset`. We see the locations where measurements are collected and the color indicates the values of signal strength (RSRP) in those locations. Fig. (a) and (b) depict measurements collected by two different users. Fig. (c) shows the union of all measurements from all users in the `Campus Dataset`.

data. First, we design a lightweight online federated learning framework, specifically for the signal strength prediction *problem*. Second, we introduce a *privacy attack*, specifically for this framework: an honest-but-curious server employs gradient inversion to infer the location of users participating in the federated signal map framework. This DLG-based attack is specifically designed to reconstruct the average location in each round; this is in contrast to state-of-the-art DLG attacks on images or text, which aims at fully reconstructing all training data points. Third, we propose a *defense* approach that selects local batches so that the inferred location is far from the true average location, thus misleading the DLG attacker. Evaluation results show that our defense mechanisms achieve better privacy-performance trade-off compared to state-of-the-art baselines.

First, w.r.t. the *signal maps prediction* based on crowdsourced data: we formulate a simple version that captures the core problem. We train a DNN to predict signal strength (RSRP) based on GPS location (latitude, longitude), while local training data arrive in an online fashion. The problem lends itself naturally to Federated Learning (FL): training data are collected by mobile devices, which want to collaborate without uploading sensitive location information. FL enables mobiles to do that by exchanging model parameters with the server but keeping their training data local [20]. The problem further lends itself to online learning because the training data are collected over time [21], [22] as users move around. We design a lightweight online FL scheme, which trains only on data collected during the current round, and we show that it performs well in this setting.

Second, w.r.t. the *location privacy*: we consider an honest-but-curious server, which implements online FL accurately but attempts to infer the location of users. Since gradient updates are sent from users to the server in every FL round, FL lends itself naturally to inference attacks from gradients. We adapt the DLG attack, originally developed to reconstruct training images and text used for training DNN classifiers [23], [24]. A key observation, that we confirm both empirically and analytically, is that a DLG attacker who observes a single gradient (SGD) update from a target user, can reconstruct the *average* location of points in the batch. Over multiple rounds of FL, this allows the reconstruction of the target(s)' mobility pattern.

Third, on the defense side, we leverage this intuition to design local mechanisms that are inherent to FL (which we refer to as "FL-native") specifically to mislead the DLGs attacker and protect location privacy. In particular, we show that the averaging of gradients inherent in FedAvg provides a moderate level of protection against DLG, while simultaneously improving utility; we systematically evaluate the effect of multiple federate learning parameters (E, B, R, η) on the success of the attack. Furthermore, we design and evaluate two algorithms for local batch selection, `Diverse Batch` and `Farthest Batch`, that a mobile device can apply locally to curate its local batches so that the inferred location is far from the true average location, thus misleading the DLG attacker and protecting location privacy. (3) We also show that the effect of multiple users participating in FL, w.r.t. the success of the DLG attack, depends on the similarity of user trajectories.

Throughout this paper, we use two real-world datasets: (i) our own geographically small but dense `Campus Dataset` [25] we introduced in Fig. 1); and (ii) the larger but sparser publicly available `Radiocells Dataset` [26], especially its subset from the London metropolitan area. We show that we can achieve good location privacy, without compromising prediction performance, through the privacy-enhancing design of the aforementioned FL-native mechanisms (*i.e.,* tuning of averaging, curation of diverse and farthest local batches, and aggregation of mobile users with different trajectories). Add-on privacy-preserving techniques, such as Differential Privacy (DP) [27], [28] or Secure Aggregation (SecAgg) [29], are orthogonal and can be added on top of these FL mechanisms, if stronger privacy guarantees are desired, at the expense of computation or utility. Our evaluation suggests that `Diverse Batch` and `Farthest Batch` alone are sufficient to achieve a great privacy-utility tradeoff in our setting.

The outline of the paper is as follows. Section 2 formulates the federated online signal maps prediction problem and the corresponding DLG attack, and provides key insights. Section 3 describes the evaluation setup, including datasets and privacy metrics. Section 4 presents the evaluation results for DLG attacks without any defense, as well as with our privacy-enhancing techniques, for a range of simulation scenarios. Section 5 discusses related work. Section 6 concludes and outlines future directions. The appendix – uploaded under supplemental materials – provide additional details on datasets, parameter tuning, analysis, and evaluation results.

2 LOCATION DLG ATTACK

In Section 2.1, we model the problem within the online federated learning framework and we define the DLG attack that allows an honest-but-curious server to infer the whereabouts of the target user(s). In Section 2.2, we provide analytical insights that explain the performance of the DLG attack for various user trajectories and tuning of various parameters, and also guide our algorithm design choices.

2.1 Problem Setup

Signal Maps Prediction. Signal maps prediction typically trains a model to predict a key performance indicator (KPI) y_i based on the input feature vector $x_i = [x_{i,1}, x_{i,2}, ...x_{i,m}]^T$, where i denotes the i-th datapoint in the training set. W.l.o.g. we consider the following: y is a metric capturing the signal strength and we focus specifically on Reference Signal Received Power (RSRP), which is the most widely used KPI in LTE. For the features x used for the prediction of y, we focus on the spatial coordinates (longitude, latitude), *i.e.,* $m = 2$.[1] We train a DNN model with weights w, per cell tower, $y_i = F(x_i, w)$; the loss function ℓ is the Mean Square Error, and we report the commonly used Root Mean Squared Error (RMSE).

We consider a general DNN architecture that, unlike prior work [23], is quite general. We tune its hyperparameters (depth, width, type of activation functions, learning rate η) via the Hyperband tuner [31] to maximize utility. Tuning the DNN architecture can be done using small datasets per cell tower, which are collected directly or contributed by users or third parties willing to share/sell their data.

Measurements over time and space. We consider several users, each with a mobile device, who collect several signal strength measurements ($\{x_i, y_i\}_{i=1}^{i=N}$), as they move around throughout the day and occasionally upload some information to the server. Fig.1 shows examples of users moving around on a university campus; Fig. 2 shows a single such user and the locations where measurements were collected for three different days. It is important to note that the measurement data are not static but collected in an online fashion. Users continuously collect such measurements as they move around throughout the day, and they periodically upload them to the server, *e.g.,* every night when the mobile is plugged into charge and connected to WiFi. This is a special case of mobile crowdsourcing (MCS) [1].

Let the time be divided into time intervals or "rounds" indexed by $t = 1, \ldots, R$. All rounds have the same duration T; in the previous example, T was one day, but we also consider other values: 1-3 hours, one day, one week, etc. [2] At the end of each such round, user k processes the set of measurement data D_t^k collected during that round and sends an update to the server. We also refer to D_t^k as the local data that "arrive" at user client k in round t. Collected over $t = 1, \ldots, R$, D_t^k reveals a lot about user k's whereabouts, as evident by the examples of Fig. 1 and Fig. 2. Human mobility is well known to exhibit strong patterns: people spend several hours a day in a few important places (*e.g.,* their home, work, and other important locations), and move between them in continuous trajectories. The locations x collected as part of the signal maps measurements $(x_i, y_i)_{i=1}^{i=N}$ essentially sample the real user's trajectory.

Notation	Description
x	Input features: (lat, lon) coordinates (and potentially more)
y	Prediction label for RSRP
ℓ	MSE loss for RSRP prediction (RMSE is reported as utility)
η	Learning rate
i	i^{th} measurement used for training: (x_i, y_i)
T	Duration of time interval/round over which users process the online data; one interval corresponds to one round in FL and one local batch
t	Index of FL round, $t = 1, 2., ..R$; each round has duration T
D_t^k	Local data arriving to user k in round t
LocalBatch	Subset of all local data D_t^k used as a local batch in FL In FedAvg: LocalBatch =D_t^k; In `Diverse Batch` and `Farthest Batch`: LocalBatch $\subset D_t^k$
w_t	Global model weights at round t
w_t^k	Local model weights at round t from user k
B	Mini-batch size; if $B = \infty$ then mini-batch = LocalBatch
B_t^k	Partition of user k's LocalBatch at round t into mini-batches
E	Number of local epochs
∇w_t^{target}	Gradient of target's model weights at round t
$\mathbb{D}$	Cosine loss used in DLG attack (Algorithm 2)
x_{DLG}	Reconstructed location by DLG attack (Algorithm 2)
$\bar{x}_t$	Centroid: average location of data in a local batch: $\bar{x}_t = \sum_{i=1}^{i=N} x_i$
eps	DBSCAN parameter that controls total clusters for `Diverse Batch`
num	`Farthest Batch` parameter that controls the number of measurements selected in each LocalBatch.

TABLE 1: Main parameters and notation. See also Fig. 2.

Federated Signal Maps. State-of-the-art mobile crowdsourcing practices [1], [5] rely on the server to collect the raw measurements (in our context locations and associated RSRP measurements) from multiple users, aggregate them into a single map, and maybe train a centralized prediction model, with the associated location privacy risks. In this paper, we apply for the first time FL [20] to the signal maps prediction problem, which allows users to keep their data on their device, while still collaborating to train a global model, by exchanging model parameter updates with the server.

In the federated learning framework [20], [32], the server and the users agree on a DNN architecture and they communicate in rounds ($t = 1, \ldots, R$) to train it. In every round t, the server initializes the global parameters w_t and asks for a local update from a subset (fraction C) of the users. The user k trains a local model on its local data and sends its update for the local model parameters w_t^k to the server. The server averages all received updates, updates the global parameters to w_{t+1} and initiates another round; until convergence. If a single gradient descent step is performed on the local data, the scheme is called Federated SGD (FedSGD). If there are multiple local steps, (*i.e.,* local data are partitioned into mini-batches of size B each, there is one gradient descent step per mini-batch, and multiple passes E epochs) on the local data), the scheme is called Federated Averaging (FedAvg) [20]. FedSGD is FedAvg for $E = 1, B = \infty, C = 1$. $B = \infty$ indicates that the entire local batch is treated as one mini-batch.

Online Federated Learning. Differently from the classic

1. In prior work on centralized signal map prediction [3], [30], we assessed the feature importance on the campus and other datasets, and found location, time, and cell tower id to be the most important, while device type, frequency and outdoor/indoor location had negligible effect on our campus datasets. In this paper, we focus on the most important features, *i.e.,* spatial coordinates, we handle time within the online learning framework, and we train one DNN per cell id.

2. Values of T were chosen to match the time scales of human mobility (on the order of hours, days or weeks) and not the dynamics of the wireless channel (much shorter time scales, *e.g.,* below a second). Selecting T to be one day or one week also allows for enough datapoints in a round (see Fig. 17 in App. A.2). It also reflects common practices in crowdsourcing signal strength (apps collect measurements throughout the day but upload once, at night, when the phone is charging).

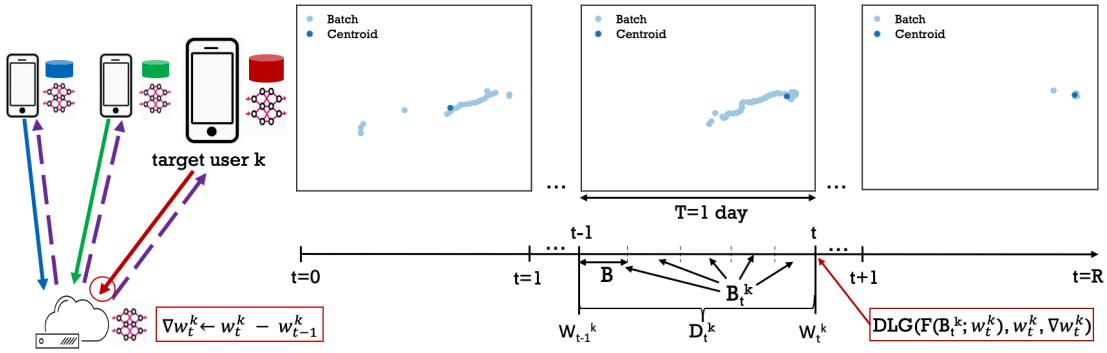


Fig. 2: **Example of the Online Federated Learning Framework with DLG Attack, using data from the campus.** The target user k collects data in an online fashion, and processes them in intervals/rounds of duration T=1 day. The right part of the picture shows real locations (in light blue) the target user visited on different days. During round t, the target collects local data D_t^k, uses it for local training, updates the local model weights w_t^k and shares them with the server. (During local training, the local data D_t^k is further split into a list B_t^k of mini-batches, each of size B.) The server observes the model parameter update w_t^k at time t, computes the gradient ($\nabla w = w_t^k - w_{t-1}^k$) and launches a DLG attack using Algorithm 2. For each day t, it manages to reconstruct the centroid (average location) of the points in D_t^k (shown in dark blue color). During the last day, where the user did not move much, the centroid conveys quite a a lot of information.

FL setting [20], the local data of user k are not available all at once, but arrive in an online fashion as the user collects measurements. We consider that the interval T (for processing online data) coincides with one round t of federated learning, at the end of which, the user processes the local data D_t^k that arrived during the last time interval T; it then updates its local model parameters w_t and sends the update to the server. We introduce a new local pre-processing step in line 16 in Algorithm 1: the user may choose to use all recent local data D_t^k or a subset of it as its LocalBatch. (Unless explicitly noted, we mean LocalBatch = D_t^k, except for Section 4.3 where `Diverse Batch` is introduced to pick LocalBatch $\subset D_t^k$ so as to increase location variance and privacy.) Once LocalBatch is selected, FedAvg can further partition it into a set of mini-batches (B_t^k) of size B. An example is depicted in Fig. 2, where data are collected and processed by user k in rounds of T = one day.

How to update the model parameters based on the stream of local data ($\{D_t^k\}, t = 1, \ldots, R$) is the subject of active research area on online learning [21], [22]. We adopt the following approach. In every round t, user k uses only its latest batch D_t^k for local training and for computing w_t^k. The data collected in previous rounds ($D_1^k, \ldots D_{t-1}^k$) have already been used to compute the previous local (w_{t-1}^k) and global (w_{t-1}) model parameters but is not used for the new local update. This is one of the state-of-the-art approaches in federated learning [33], [34], [35]. Our design choice to discard data from previous rounds, raises a concern about catastrophic forgetting [36], [37]. Our intuition is that this will *not* happen in our datasets because of the predictable and repeated patterns in human mobility data. As users visit the same locations and follow the same trajectories over days and weeks, they contribute similar data over time. This intuition was, indeed, confirmed by the model evaluation. [3]

3. Due to lack of space, quantitative comparison to alternative approaches (*i.e.*, "Cumulative Online FL", which accumulates all training data as they arrive, and "Testing on Past Data", which trains on the current round but tests on the past data) are deferred to Appendix A.3, under supplementary materials.The consistency in performance across all evaluation scenarios confirms that there is no catastrophic forgetting.

Algorithm 1: Online FedAvg with DLG Attack.

Given: K users (indexed by k); B local mini-batch size; E number of local epochs; R number of rounds of duration T each; C fraction of clients; n_t is the total data size from all users at round t, η is the learning rate; the server aims to reconstruct the local data of target user k.

Server executes:
- Initialize w_0
- **for** *each round t = 1,2, ... R* **do**
  - $m \leftarrow max(C \cdot K, 1)$
  - $S_t \leftarrow$ (random subset (C) of users)
  - **for** *each user* $k \in S_t$ *in parallel* **do**
    - $w_t^k \leftarrow UserUpdate(k, w_{t-1}, t, B)$
    - **if** k==*target* **then**
      - $\nabla w_t^{target} \leftarrow w_t^k - w_{t-1}^k$
      - **DLG**$(F(x; w_t^k), w_t^k, \nabla w_t^{target})$
  - $w_t \leftarrow \sum_{k=1}^{K} \frac{n_t^k}{n_t} w_t^k$

UserUpdate(k, w, t, B)**:**
- Local data D_t^k are collected by user k during round t
- Select LocalBatch $\subseteq D_t^k$ to use for training
- $n_t^k = |\text{LocalBatch}|$: training data size of user k at round t
- $B_t^k \leftarrow$ (split LocalBatch into mini-batches of size B)
- **for** *each local epoch i: 1...E* **do**
  - **for** *mini-batch* $b \in B_t^k$ **do**
    - $w \leftarrow w - \eta \nabla \ell(w; b)$
- **return** w *to server*

Therefore, the design choice of discarding past data allows us to train a good signal strength model, while keeping storage and computation light.

Honest-but-Curious Server. We assume an honest-but-curious server who receives and stores model updates from each user, and whose goal is to infer the user's locations. The server may be interested in various location inference goals: *e.g.*, the user trajectory at various spatiotemporal granularities, important locations (*e.g.*, home or work), presence near points-of-interest (*e.g.*, has the user been at the doctor's office?). W.l.o.g., the server targets a user k who participates

Algorithm 2: DLG Attack.

Input: $F(x; w_t)$: DNN model at round t; w_t: model weights, ∇w_t:model gradients, after target trains on a data batch of size B at round t, learning rate η for DLG optimizer; m: max DLG iterations; a: regularization term for cosine DLG loss.
Output: reconstructed training data (x, y) at round t
Initialize $x'_0 \leftarrow \mathcal{N}(0,1)$, $y'_0 \leftarrow \bar{y}$ `// mean RSRP`
for $i \leftarrow 0,1, \ldots m$ **do**
$\nabla w'_i \leftarrow \partial \ell(F(x'_i, w_t), y'_i)/\partial w_t$
$\mathbb{D}_i \leftarrow 1 - \frac{\nabla w \cdot \nabla w'_i}{\|\nabla w\| \|\nabla w'_i\|} + \alpha$ `// cosine loss`
$x'_{i+1} \leftarrow x'_i - \eta \nabla_{x'_i} \mathbb{D}_i$, $y'_{i+1} \leftarrow y'_i - \eta \nabla_{y'_i} \mathbb{D}_i$
return $x_{DLG} \leftarrow x'_{m+1}$

in round t: it compares updates across successive rounds w^k_{t-1} and w^k_t and computes the gradient for round t; see Algorithm 1 and Fig.2. It then uses this gradient to infer user k's location in round t, as described next.

DLG Attack against a Single Update. At Line 11 of Algorithm 1, the server launches a DLG attack to infer the location of user k in round t. The DLG attack is defined in Algorithm 2 and an example is shown in Fig. 3. In each iteration i, the DLG algorithm: 1) randomly initializes a dummy location x'_0 (shown in yellow), 2) obtains the gradient at dummy location, $\nabla w'_i$, a.k.a. dummy gradient, 3) updates the dummy location towards the direction that minimizes the cosine distance between the dummy and true gradient. We choose to minimize cosine, as opposed to euclidean loss to match the direction, not the magnitude, of the true gradient [24].

Implementation details. (1) The attacker reconstructs both the location x (*i.e.,* latitude and longitude coordinates), and the RSRP value y; we cannot use the analytical reconstruction of the label proposed in [38], since we have regression instead of classification. (2) We observe that different location initializations converge to the same point in practice, as shown in Fig. 6. We initialize the prediction label with the mean RSRP from the training data, which is realistic: the attacker can have access to public cellular signal strength measurements or collect a few measurements around each cell tower. (See Appendix A.4 on "Analysis of DLG Label Initialization", for a discussion on different RSRP initializations) (3) We set the maximum number of iterations to $m = 400,000$, and add an early stopping condition: if the reconstructed location does not change for 10 DLG iterations, then we declare convergence.

Key observation O1: DLG on one batch. An example of our DLG attack on one SGD update from user k is depicted in Fig. 3: it reconstructs x_{DLG}, which ends up being close to the average location $\bar{x}$ in batch D^k_t. We experimented with multiple initializations and we found that to be true in practice, independently of initialization; see Fig. 6. Therefore, when applied to a single local update, *Algorithm 2 reconstructs one location x_{DLG}, which is close to the average location $\bar{x}$ of the N points in that batch.* This is in contrast to the original DLG attacks on images [23], [24], which aimed at reconstructing all N points in the batch.

Since local data arrive in an online fashion, the server can launch one DLG attack per round and reconstruct one (the average) location in each round. All reconstructed locations together reveal the user's whereabouts, as discussed next.

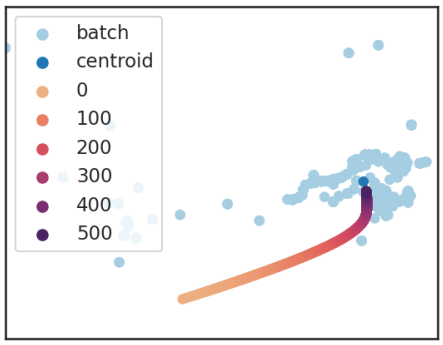


Fig. 3: **Example of DLG attack (Algorithm 2), using a user from the `Campus Dataset`. The target has been in the light blue locations during an interval of T=1 week. The true average location (centroid, $\bar{x}$) of points is shown in dark blue. The DLG attack starts (iteration "0") with a dummy location (shown in yellow) and it gets closer to the centroid with more iterations (darker color indicates progression in iterations), by minimizing the cosine loss $\mathbb{D}$ of the true gradient and the gradient of the reconstructed point. After $m = 500$ iterations, the distance between the reconstructed location x_{DLG} (dark purple) and $\bar{x}$ (dark blue) is only 20 meters.**

Key observation O2: DLG across several rounds. Human mobility is well-known to exhibit strong spatiotemporal patterns, *e.g.,* (i) people move in continuous trajectories and (ii) they frequently visit a few important locations, such as home, work, gym, etc [39], [40]. Because of (i), inferring the average location in successive rounds essentially reveals the trajectory at the granularity of interval T. Because of (ii), there is inherent clustering around these important locations, as exemplified in Fig. 4a. These can reveal sensitive information [19] and help identify the user [41].

2.2 Analytical Insights

In this section, we are interested in analyzing how close is the location reconstructed by the DLG attack (x_{DLG}) to the true average location of the batch ($\bar{x}$). The analysis explains analytically our empirical observations, provides insights into the performance of DLG attacks depending on the input data characteristics, and guides our design choice to maximize privacy.

Given a data mini-batch of size B: $\{(x_i, y_i)\}_{i=1}^{i=B}$, the DLG attacker can reconstruct a unique user location x_{DLG}, which satisfies:

$$x_{DLG} = \frac{1}{B} \sum_{i=1}^{i=B} \frac{g_i(w)}{\bar{g}(w)} \cdot x_i, \tag{1}$$

where $g_i(w) = \frac{\partial \ell(F(x_i,w),y_i)}{\partial b^1_h} \in \mathbb{R}$ is an element in the gradient ∇w representing the partial derivative of loss ℓ w.r.t b^1_h, and $\bar{g}(w) = \frac{1}{B} \sum_{i=1}^{i=B} g_i(w)$.

Proof: We are able to prove this lemma under two assumptions on the DNN model architecture: (1) the DNN model starts with a biased fully-connected layer (Assumption 1 in Appendix A.1.1); and (2) the bias vector of its first layer (b^1_h) has not converged (Assumption 2 in Appendix A.1.1). □

Next, we bound the distance between the reconstructed user location by the DLG attacker and the centroid of user locations in a data mini-batch as follows:

Theorem 1. Suppose that a data mini-batch of size B: $\{(x_i, y_i)\}_{i=1}^{i=B}$ is used to update the DNN model $y_i = F(x_i, w)$ during a gradient descent step. Then, the reconstruction error of the DLG attacker, defined as the L_2 distance between the user location reconstructed by DLG attacker x_{DLG} and the centroid of user locations in this mini-batch $\bar{x} = \frac{1}{B}\sum_{i=1}^{i=B} x_i$, can be bounded by the following expression:

$$||x_{DLG} - \bar{x}||_2 = \frac{1}{B|\bar{g}(w)|}||\sum_{i=1}^{i=B}\big(g_i(w) - \bar{g}(w)\big)\cdot\big(x_i - \bar{x}\big)||_2$$
$$\leq \frac{1}{2B|\bar{g}(w)|}\sum_{i=1}^{i=B}\Big(\big(g_i(w) - \bar{g}(w)\big)^2 + ||x_i - \bar{x}||_2^2\Big). \quad (2)$$

Proof: See Appendix A.1.2. □

Theorem 1 says that the reconstruction error of the DLG attacker is equal to the L_2 norm of the sample co-variance matrix between the partial derivative over the bias $g_i(w)$ and the user location x_i, divided by the (absolute value of the) average partial derivative within a mini-batch $\bar{g}(w)$. Moreover, this error can be upper bounded by the sum of the sample variance of $g_i(w)$ and the sample variance of $||x_i||_2$ in the mini-batch. This is a tight bound that can be achieved if, for any i and j, it is: $|x_i - \bar{x}| = |x_j - \bar{x}| = |g_i(w) - \bar{g}(w)| = |g_j(w) - \bar{g}(w)|$.

The above theorem involves the partial derivatives $g_i(w)$ whose values are hard to predict when x_i varies. To bound the error of the DLG attacker without involving such derivatives, we further make the mild assumption that the gradient function is Lipschitz continuous (see Assumption 3 in Appendix A.1.3), and state the following theorem:

Theorem 2. Subject to the Lipschitz continuity assumption about $\nabla\ell(F(x_i, w), y_i)$, the reconstruction error of the DLG attacker can be bounded by:

$$||x_{DLG} - \bar{x}||_2 \leq \frac{L^2}{B|\bar{g}(w)|}\sum_{i=1}^{i=B}\big(\alpha||x_i - \bar{x}||^2 + ||y_i - \bar{y}||^2\big), \quad (3)$$

where $\bar{y} = \frac{1}{B}\sum_{i=1}^{i=B} y_i$ and $\alpha = 1 + \frac{1}{2L^2}$.

Proof: See Appendix A.1.3. □

Theorem 2 says that the reconstruction error can be bounded by the weighted sum of the sample variance of the user data $||x_i||_2$ and the labels y_i in the mini-batch. To achieve the bound of Eq. (2), Assumption 3 should hold in addition to the condition for achieving the bound of Eq. (1). For instance, when the mini-batch size B is equal to 1, the equality in Eq. (1) and Eq. (2) can be achieved.

(I1) Impact of data mini-batch variance. Theorem 1 shows that the variance of user locations affects the upper bound of the DLG attacker's reconstruction error. The smaller the data mini-batch variance is, the smaller the upper bound of the DLG attacker's reconstruction error is. Theorem 1 also shows that the DLG error depends on the variance of the gradients. One may intuitively argue that since the randomness of the gradients comes from the randomness of the data, the larger the data variance the larger the gradients variance too, thus the larger the error. Theorem 2 does not involve gradients. It directly shows how the variance of local user data and associated labels affect the upper bound of the reconstruction error. Motivated by the above discussion, we propose an algorithm, which we refer to as `Diverse Batch` to increase the mini-batch data variance of each user during training, see Sec. 4.3. In theory, an increasing upper bound does not guarantee that the actual reconstruction error will increase. We empirically show this to be the case (see Table 2 in Sec. 4.3).

(I2) Impact of model convergence rate. As shown above, another key component affecting the upper bound of the attacker's reconstruction error is $|\bar{g}(w)|$, which is the average partial derivative over the bias and reflects the convergence of the global model: As the global model converges (*i.e.,* the training loss ℓ converges to zero), $|\bar{g}(w)|$ will also converge to zero, and hence the upper bound of the reconstruction error will diverge to infinity. This is expected since the attacker needs user information from the gradient to reconstruct users' location. Recall that the attacker attempts to reconstruct one user location at each mini-batch and FL round. As the model converges faster, the reconstruction error will diverge faster and thus a smaller fraction of reconstructed user locations will be accurate, those corresponding to early reconstruction attempts, see Fig. 7.

(I3) Impact of Averaging. While under FedSGD the attacker observes the gradient updates after processing each mini-batch, under FedAvg the attacker observes the gradient update at the end of the batch/round, thus this gradient update is the time average of $\frac{\partial\ell(F(x_i,w),y_i)}{\partial b_h^1}$ during each training round. We can apply Theorems 1 and 2 to the FedAvg case by re-defining $g_i(w)$ as the time average of $\frac{\partial\ell((x_i,w),y_i)}{\partial b_h^1}$ during each training round, and the impact of each parameter will be the same, as that discussed above for FedSGD. In Sec. 4.1 and 4.2, we show the impact of FedAvg parameters (B, E) on the DLG attack: as B decreases and/or E increases, the attack is less accurate, due to faster model convergence rate caused by multiple local gradient descent steps (see Fig. 9 and Fig. 10).

(I4) Impact of multiple users. FL involves the participation of multiple users, who will jointly affect the convergence of the global model. Prior work has shown that as the data diversity across multiple users increases, *i.e.,* the dissimilarity or heterogeneity between users increases, the global model may converge slower or even diverge [42], [43]. The global model convergence rate impacts the DLG attacker's reconstruction accuracy. Thus, when the data diversity across multiple users increases, we expect that the global model will converge slower, resulting in a more accurate reconstruction of user locations by the DLG attacker. In Sec.4.6, we empirically show how the similarity of users affects the DLG attacker's performance; see Table 3.

3 Evaluation Setup

We evaluate the success of the DLG attack for different scenarios: we specify the exact configuration and parameter tuning for the online federated learning (Algorithm 1), DLG attack (Algorithm 2), and any defense mechanism in Section 4. In Sec. 3.1, we describe two real-world datasets that we use as input to our simulations. In Sec. 3.2, we define privacy metrics that quantify the privacy loss due to the attack.

3.1 Datasets

`Campus Dataset` [25]. This dataset is collected on a university campus and is the one depicted in Fig. 1. It contains real traces from seven different devices used by student volunteers and faculty members, using two cellular providers over a period of four months. IRB review was not required as the proposed activity was deemed as non-human subjects research by the IRB of our institution. It is a relatively small dataset; it spans an entire university campus, a geographical area of approx. 3 km^2 and 25 cell towers. However, it is very dense: it consists of 169,295 measurements in total. Pseudo-ids are associated with the measurements which facilitate the simulation of user trajectories in FL. The number of measurements per cell tower and user varies, details are deferred to Appendix A.2. For the evaluation in Sec. 4, we pick the cell tower x204 with the largest number of datapoints, and choose the user 0 with the most measurements as the target user. The campus is depicted in Fig. 1c, and the locations of the target (user 0) are depicted in Fig. 1a (all cells) and Fig. 4a (cell x204 only). It is worth noting that we know the frequently visited locations coincide with home (on the right part of the picture) and work (campus buildings on the left part of the picture) locations for the user. This side information becomes useful when we assess the success of the attack. Moreover, the attacker uses the campus boundaries (an area of 3 km^2) as the defined area of the attack; if some reconstructed locations fall outside this area, they are treated as diverged.

`Radiocells Dataset` [26]. We also consider the large-scale real-world `Radiocells Dataset`, which comes from a community project founded in 2009 under the name openmap.org. It contains data from over 700 thousand cell towers around the world and is publicly available in [26]. Raw measurement data are available for in-depth analysis of communication networks or cell coverage, and can be used for our problem of signal maps. The measurements are contributed by real users without logging their user ids. Users log and upload their data into multiple upload files: each containing device information, version of the app, etc. that can be used for distinguishing users, as in [1]. We focus on cellular data from 2017 (8.5 months) in the area of London, UK, from the top cell tower (x455), which had the most measurements (64,302 in total) from approx. 3,500 upload files and a geographical area that spans approx. 5,167 km^2. Each upload file corresponds to a single device, typically containing 16 measurements per $2h$ on average. Since no pseudo-ids are provided that would allow us to link multiple upload files of a user, we use heuristics to create longer user trajectories so that we have enough data points per batch; see details in Sec. 4.6. The trajectories of the synthetic users are depicted in Fig. 16a.

For both datasets, we partition the data into batches D_t^k for each k user, corresponding to different time intervals T in time-increasing order ($t = 1, .., R$), so as to simulate the online collection of data points as it happens. We mainly focus on T=1 week; each batch contains all datapoints in that week, which are used in one FL round. Choosing coarser T results in fewer batches/rounds but it includes more datapoints per batch, which facilitates local training. In the `Campus Dataset`, there are 11 weeks for user 0 (the target user in the `Campus Dataset`) and the average batch size is 3,492 measurements. In `Radiocells Dataset`, most users' batches contain fewer than 50 datapoints on average, as it is a much sparser dataset. The target user in `Radiocells Dataset` (user 3) contains 26 1-week batches. The features in each dataset are standardized by subtracting the mean and scaling to unit variance so that their values span an appropriate range.

3.2 Location Privacy Metrics

We use the RMSE for the signal maps prediction problem as our utility metric, as described at the beginning of Sec. 2.1. We also need metrics that capture how similar or dissimilar the reconstructed locations are from the real ones. Any FL algorithm and any defense mechanism must be evaluated w.r.t. the privacy-utility trade-off they achieve.

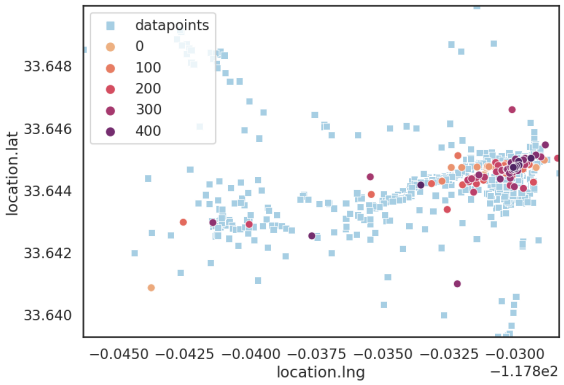
Visualization. Visually comparing the reconstructed (x_{DLG}) to the true locations in the batch (D_t^k) provides intuition. Fig. 4(a) shows the real (shown in light blue) and the reconstructed (shown in color) locations reconstructed per 1h-long batches, for a user in the `Campus Dataset`. One can see that the reconstructed locations match the frequently visited locations of the user. For example, DLG seems indeed to reconstruct locations on the right side of the figure, where we confirmed that graduate student housing is on this campus. This is expected, based on our key observation O2, *i.e.,* the characteristics of human mobility.

Distance from the Centroid. In order to assess how accurate the attack is within a single round t, we use the distance $||x_{DLG} - \bar{x_t}||$, *i.e.,* how far (in meters) is the reconstructed location from the average location of points in that batch B_t^k. Based on the key observation O1, we expect this distance to be small when the attack converges.

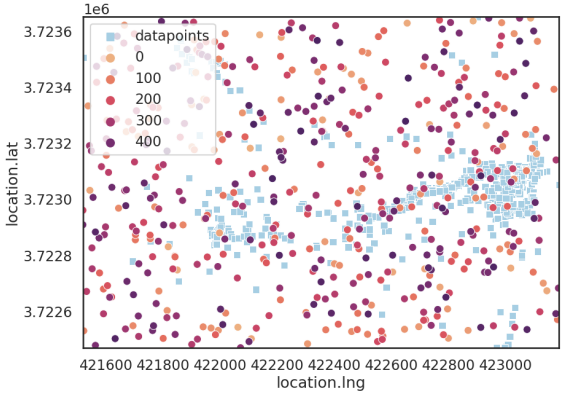
Comparing location distributions. To assess the success of the attack considering all rounds $t = 1, ..., R$, we need a metric that captures how similar or dissimilar are the reconstructed locations from the real ones; *e.g.,* see Fig. 4 for an example of real (light blue) vs. reconstructed (in color) locations over multiple rounds. We considered the KL-divergence and Jensen-Shannon distance, which are well-known metrics for comparing distributions. However, they only capture the difference in probability mass, not the spatial distance between real and inferred frequent locations, which is of interest in our case, as per key observation O2.

We use the **Earth Movers Distance (EMD)** [44], [45] to capture the distance between the 2D-distributions of reconstructed and real locations. It has been previously used in location privacy as a measure of t-closeness [46], l-diversity [47]. It is defined as: $EMD(P, Q) = \inf_{\gamma \in \Pi(P,Q)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|]$, where $\Pi(P, Q)$ is the set of all joint distributions whose marginals are P (true locations) and Q (reconstructed locations). EMD takes into account the spatial correlations and returns the minimum cost required to convert one probability distribution to another by moving probability mass.[4] We use the Euclidean distance when

4. A classic interpretation of EMD is to view the two probability distributions as two ways to pile up an amount of dirt ("earth") over a region and EMD as the minimum cost required to turn one pile into the other. Cost is defined as the amount of dirt moved x the distance it was moved.



(a) Reconstructed locations by the DLG attack, considering one attack per 1-hour batches and FedSGD. The distance between the real and inferred location distributions is small: EMD=5.3.

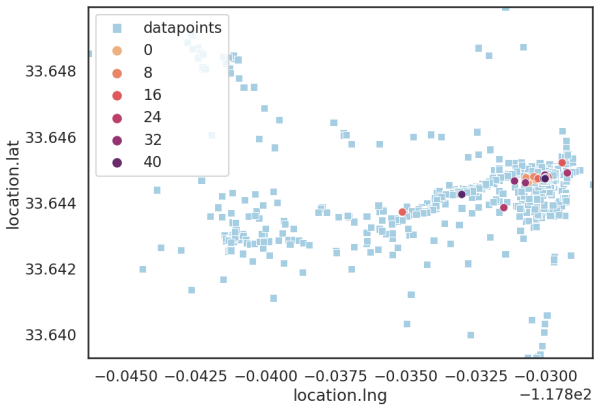


(b) Locations generated uniformly at random (1 per hour) within the defined campus boundaries. The distance between the real and uniform random location distributions is high: EMD = 21.33 for 5 realizations; it provides a baseline for comparison.

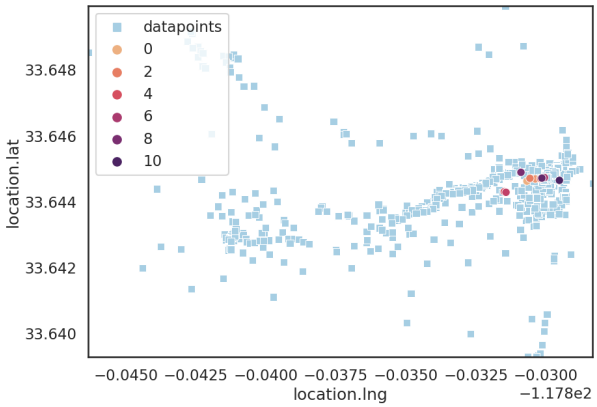
Fig. 4: We consider a target user and its real locations on campus, 489 in total (with $T = 1$ hour), depicted in light blue. The over-sampled area on the right corresponds to home location of that user. The other area on the left, corresponds to their work on campus. The DLG attacker processing updates from 1h rounds can successfully reconstruct the important locations of the user: the difference between the distribution of real and the inferred locations is EMD=5.2. To put that in context, if one would randomly guess the same number of locations, the EMD would be 21.33.

calculating EMD on GPS coordinates in UTM. We compute EMD using Monte Carlo approximations for $N = 1000$ projections; which is more computationally efficient than the exact EMD calculation and it is suitable for 2D distributions.

The range of EMD values depends on the dataset and spatial area. $EMD = 0$ would mean that the distributions of real and reconstructed locations are identical. Low EMD values indicate successful location reconstruction by the DLG attack thus high privacy loss. To get more intuition, let's revisit Fig. 4. In Fig. 4(a), the DLG attacker reconstructed locations around the frequently visited locations (close to home) and achieved $EMD = 5.3$. In Fig. 4(b), we show the same number of locations chosen uniformly at random, which leads to $EMD = 21.33$; this provides an upper bound in privacy (random guesses by the attackers) in this scenario.



(a) $T = 24$-hour rounds: EMD=6.4.



(b) $T = 1$-week rounds: EMD=7.6.

Fig. 5: The reconstructed locations in the case of strongest attack for various $T = 24$-h vs. 1-week. The light blue square points are the ground truth points and the circle points are the reconstructed points for each round; the darker color represents the later rounds. RMSE is 4.93, 4.91, 5.16 for 1w, 24h, 1h respectively. Reconstruction with 1-hour rounds (Fig.4a) reveals fine-grained user trajectories. The coarser rounds (24-h, 1-week) still reveal the frequent locations of the target, *e.g.*, their home/work locations.

%Attack Divergence. In our simulations, we observed that: (i) if the DLG attack converges, it converges to $\bar{x}$ regardless of the initialization; (ii) however, the DLG attack did not always converge, depending on the location variance of the batch, the tuning of parameters of the DLG optimizer and FedAvg. Examples of attack divergence are shown in Fig. 7a. In Sec. 4, we define a rectangular geographical area of interest for the attacker *e.g.*, the entire 3 km^2 campus in `Campus Dataset`. If some reconstructed locations are outside the boundaries, we declare them "diverged" and (i) we discard them when computing the privacy (EMD) metric, but also (ii) we report the fraction of those attacks that diverged. In practice, if an attack diverges outside the area of interest, the attacker can relaunch the DLG attack with a different initialization hoping until it reaches convergence to $\bar{x}$. This, however, is costly for the attacker, therefore the % of attacks that diverged is another metric of success or failure of the DLG attack.

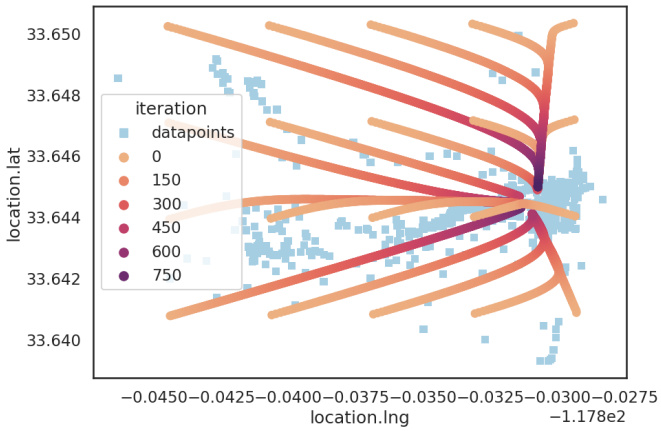


Fig. 6: **FedSGD for one Round.** DLG converges visually and in terms of cosine loss ($\mathbb{D} < -0.9988$) to the average location $\bar{x}$ regardless of the initialization point.

4 Evaluation Results

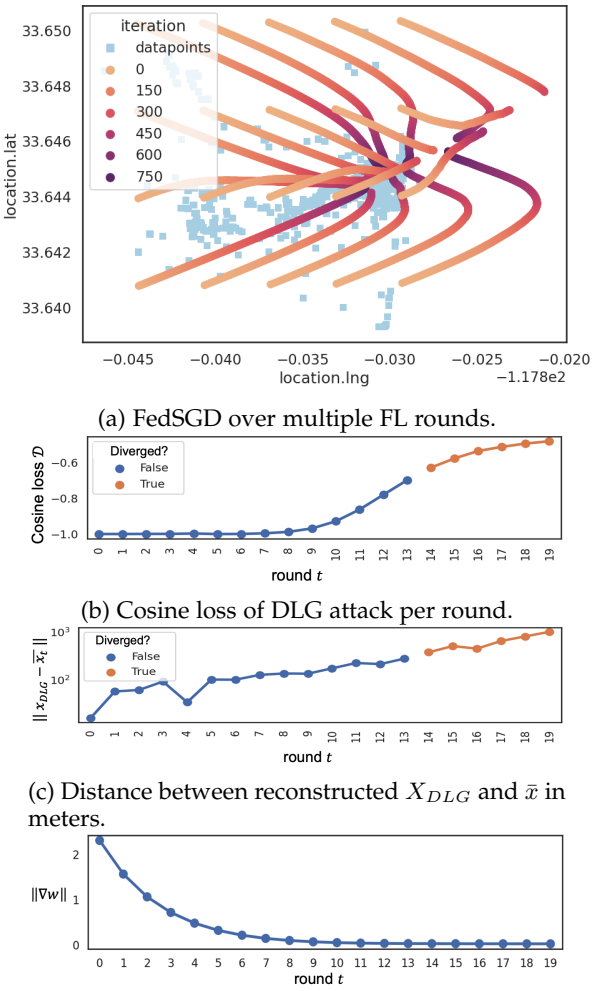
Next, we evaluate the DLG attack in a range of scenarios. In Section 4.1, we consider FedSGD, which is the most favorable scenario for DLG – the strongest attack. In Section 4.2, we show that the averaging inherent in FedAvg provides a moderate level of protection against DLG, which also improves utility. In Section 4.3, we propose a simple defense that users can apply locally: `Diverse Batch` curates local batches with high variance. In Section 4.6, we show the effect of multiple users on the success of the DLG attack. We use a default $\eta = 0.001$ (or $\eta = 10^{-5}$ in case of mini-batches), details on tuning η are deferred to Appendix A.4. W.l.o.g., we focus on a particular target, whose locations are to be reconstructed by the DLG attacker and we report the privacy (EMD, % diverged attacks)-utility tradeoff (RMSE).

4.1 Location Leakage in FedSGD

Strongest Attack. FedSGD is a special case of Algorithm 1 with $B = \infty$, $E = 1$: in each round t the target user performs a single SGD step on their data D_t^k and sends the local model parameters w_t^k to server, which in turn computes the gradient (line 10 in Algorithm 1). ∇w_t^k corresponds to the true gradient obtained on data D_t^k and it is the best scenario for the attacker.

Impact of time interval T. Fig. 4a shows the true vs. the reconstructed locations via DLG for intervals with T=1h. Fig. 8 shows the results for the same data but divided into intervals with duration longer than 1h, *i.e.*, 24h and 1 week. The shorter the interval, the better the reconstruction of locations: we can confirm that visually and quantitatively via EMD, while the utility (RMSE) is not affected significantly. This is in agreement with Insight I1, since smaller T leads to smaller batch variance in the target's trajectory. However, the most visited locations by the user (*i.e.*, the home and work) are successfully reconstructed for all T; this is due to the mobility pattern of the target, who repeats his homework trajectory over time.

Impact of DLG Initialization. First, we consider a single FL round and we evaluate the effect of DLG initialization. For example, consider LocalBatch to consist of the measurements of the target from week $t = 7$ (D_t^0) in `Campus Dataset`, and perform one local SGD step to train the local model. The global model is initialized to the same random weights before local training. The attacker splits the geographical area into a grid of 350 meters and uses the center of each grid cell as a candidate initial dummy point in Algorithm 2. Fig. 6 shows the results for 20 random initialization. We observe that, for all initializations, the attack converges (cosine loss $<$ -0.9988) to the average location $\bar{x}$ of the local data, regardless of the distance between the initialization and the centroid of the data. This is explained by Insight I2 in Sec. 2.2; the gradient provides information for the attack, since the model has not converged.



(a) FedSGD over multiple FL rounds.

(b) Cosine loss of DLG attack per round.

(c) Distance between reconstructed X_{DLG} and $\bar{x}$ in meters.

(d) Norm of gradient of flattened per-layer weight matrices between two consecutive rounds.

Fig. 7: **FedSGD over Multiple FL Rounds.** FL is performed over multiple rounds. (a) The local data are considered the same from one round to the next, and the same as the batch used in Fig. 6. The attacker uses the same 20 random initialization as in the single round case. The effect of multiple ML rounds is that (b) the gradients are converging to zero, which results in (c) higher cosine loss for the DLG attacker and eventually (d) the reconstructed points X_{DLG} end up further away from the centroid $\bar{x}$.

Impact of Number of FL Rounds. Second, we repeat the same experiment, but with the goal of evaluating the effect of multiple FL rounds, everything else staying the same. To that end, we consider that the LocalBatch data is

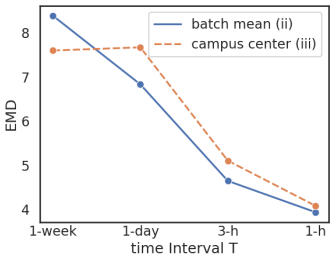
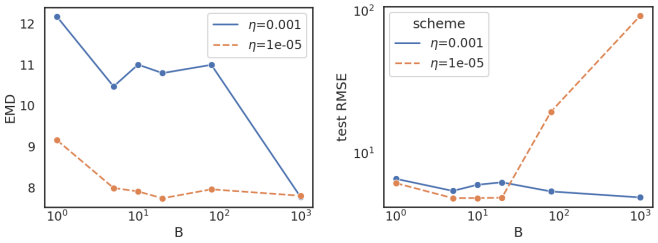
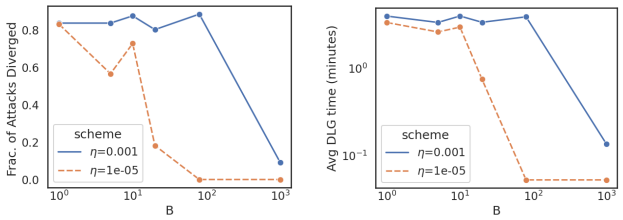


Fig. 8: **Effect of Interval Duration:** the shorter the interval T, the less privacy (low EMD), due to lower batch variance. **Effect of Initialization Strategies:** (ii) mean-init of the batch with Gaussian noise, (iii) initialize in the campus center, then use previous batch's x_{DLG} to initialize next dummy point. Both are strong attacks regardless of the initialization strategy: all reconstructed points converge and result in similar privacy (EMD).

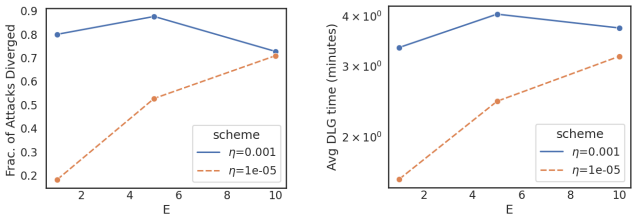


(a) EMD vs. mini-batch size B. (b) RMSE vs. mini-batch size B.



(c) $B \leq 20$ increases divergence. (d) $B \leq 20$ increases DLG time.

Fig. 9: **Impact of mini-batch size B in FedAvg.** [T=1 week, $E = 1$.] Reducing mini-batch size B introduces more averaging of the gradients which increases EMD (and privacy) and makes the attack more expensive due to divergence. The default η seems to be less sensitive to B in terms of RMSE. Here B=1000 corresponds to $B = \infty$, thus FedSGD, which leads to reduced privacy but also to higher RMSE for the lower η.



(a) $E \geq 5$ increases divergence. (b) $E \geq 5$ increases DLG time.

Fig. 10: **Impact of local epochs E on 1-week rounds.** We set $B = 20$ and increase local epochs, which increases the EMD and attack divergence, while utility is preserved.

the same for all rounds $D_t^0 = D_7^0, t = 0, ..., 20$ (and the same as in the previous experiment: D_t^0); this removes the effect of local data changing over time in an online fashion. We also use the exact same 20 random initializations, as above. The global model is now updated in each round and iterates with the target as in Algorithm 1. The norm of the flattened per-layer weight matrices approaches zero after round 9 (see Fig. 7d) and at this point the DLG attack starts diverging; the reconstructed point is further away from the mean of the data as shown in Fig. 7c and the final cosine loss $\mathbb{D}$ starts increasing (Fig. 7b). In the worst case, when the attack diverges, the reconstructed point is 1 km away from the centroid location of the batch. This can be explained by Insight I2 in Sec. 2.2: after several rounds, the model starts converging and the gradients decrease; the average gradient goes to 0, the bound in Theorem 1 goes to ∞, and x_{DLG} can go far from $\bar{x}$. Thus, even in the worse scenario of the FedSGD, without any add-on averaging or defense mechanisms, there is some protection against the DLG attack, after the initial FL rounds when the global model converges.

DLG Initialization Strategies. There are different strategies for initializing the dummy points in each batch. (i) The attacker could pick randomly within the geographical area of interest, as we did in Fig. 6, or the middle of the campus. (ii) The attacker could use a rough estimate of $\bar{x}$ plus Gaussian noise. (iii) The attacker could leverage the reconstructed location from a previous round and use it to initialize the dummy point in the next round, in order to leverage the continuity of user mobility and make an educated guess especially in the finer time intervals. Fig. 8 compares strategies (ii) and (iii): both are strong attacks, with all points converging within the area of interest, and resulting in similar EMD. If an attack diverges, (i) would be better than (ii) or (iii), to keep the dummy point within the defined boundaries. For the rest of the paper, we use by default strategy (ii) for faster simulation.

4.2 Location Leakage in FedAvg

FedAvg is the general Algorithm 1, of which FedSGD is a special case for $B = \infty$, $E = 1$. FedAvg [20] has many parameters that control the computation, communication and storage at the users and server. The learning rate η is tuned for each dataset, see Appendix A.4; the number of FL rounds R is discussed previously for FedSGD; the fraction C of users in round is related to global averaging in Section 4.6. Our focus here is to evaluate the effect of local averaging through the use of B-sized mini-batches and E number of epochs, as a way to defend against DLG attacks. Intuitively, the more local SGD steps (smaller B and high E), the more averaging over local gradients, and the less successful the DLG attack by the server based on the observed $w_t^k - w_{t-1}^k$. Interestingly, more averaging improves both convergence and utility [20]. Throughout this section, we focus on $T = 1$ week which gives 11 intervals.

Impact of Mini-batch Size B in FedAvg. Fig. 9 shows the utility (RMSE) and privacy (EMD, Fraction of Attacks Diverged and Avg DLG time) metrics when the target splits its local data into mini-batches of size B, and performs one SGD step per mini-batch. In terms of the convergence speed and accuracy, prior works such as [43], [48] have provided theoretical rules for the bound of mini-batch Size B and

proven that larger B can lead to faster convergence speed. At one extreme $B = \infty$, the entire LocalBatch is treated as one mini-batch, and this becomes FedSGD. At the other extreme, $B = 1$, there is one SGD step per local data point, which maximizes privacy. Smaller B values decrease RMSE, especially for lower η, but for the default $\eta = 0.001$ the RMSE is not significantly affected. In addition to EMD, we show in Fig. 9c the fraction of diverged DLG attacks. As B decreases, the fraction of diverged attacks increases (which makes the attack less accurate) due to increased gradient descent steps. It also makes the attack more expensive (in terms of execution time and DLG iterations). We choose $B = 20$ (3rd marker in Fig. 9) and the lower η =1e-05 in order to get some privacy protection (EMD increases slightly) and to maximize utility.

Impact of Local Epochs E. Another parameter of FedAvg that affects the number of SGD steps is the number of epochs E, *i.e.,* the number of local passes on the dataset, during local training. We set $B = 20$, based on the previous experiment and we evaluate the impact of local epochs for two learning rates η. Fig. 10 shows that increasing E increases privacy both in terms of EMD and divergence. It also improved utility (not shown): can reduce RMSE from 6.25 to 4.75dbm.

Putting it together. We choose $E = 5$ and $B = 20$, which together provide improved privacy and utility. In summary, averaging in FedAvg provides some moderate protection against DLG attacks, which is also consistent with Insight I3. However, even with these parameters, the frequently visited locations (*i.e.,* home and work) of the target can still be revealed (*e.g.,* see Fig. 11a), which motivated us to design the following algorithm for further improvement.

4.3 FedAvg with Diverse Batch

Intuition. So far, we have considered that every user, including the target, processes *all* the local data *in the order they arrive* during round t, *i.e.,* in line 16 of Algorithm 1 it is LocalBatch = D_t^{target}. The local averaging in FedAvg prevents the server from obtaining the real gradient, thus providing some protection. We can do better by exploiting the key observation (O1) supported analytically by insight (I1): the variance of locations in a batch affects how far the reconstructed x_{DLG} is from the batch centroid $\bar{x}$. If the target preprocesses the data to pick a subset LocalBatch $\subseteq D_t^{target}$, so that the selected locations have high variance, then we can force the DLG attack to have high $|x_{DLG} - \bar{x}|$ and possibly even diverge.

`Diverse Batch` Algorithm. There are many ways to achieve the aforementioned goal. We designed `Diverse Batch` to maximize variance of locations in LocalBatch, using DBSCAN clustering. In each FL round t, the target does the following at line (16) of Algorithm 1:

1) The data D_t^{target} that arrived during that round t are considered candidates to include in LocalBatch.
2) Apply DBSCAN on those points and identify the clusters.
3) Pick the center point from each cluster and include it in LocalBatch; this intuitively increases variance.
4) If more datapoints are needed, remove the selected points from D_t^{target} and repeat steps 1, 2 recursively on the remaining data, until the desired LocalBatch size is reached.

We refer to this selection of LocalBatch as `Diverse Batch` and it is applied in line (16) of Algorithm 1. After that, FedAvg continues as usually, potentially using mini-batches and multiple epochs. Appendix A.4 shows more details.

Data Minimization. In our implementation of `Diverse Batch`, the target applies step 3 above exactly once, and skips step 4. As a result, `Diverse Batch` uses significantly fewer points, and thus has a data minimization effect, in addition to high variance. For example, with $eps = 0.05$km, `Diverse Batch` achieves EMD=15.23 using 1% of the location compared to a random sampling of 1% of the points that leads to EMD=7.6; see Fig. 11(b) vs. 11(c), as well as Table 2.

Tuning `Diverse Batch`. First, we tune the learning rate $\eta = 0.001$ via a grid search; see Appendix A.4 for details. An important parameter of DBSCAN is eps, which controls the maximum distance between points in a cluster: higher eps leads to fewer clusters, *i.e.,* less datapoints chosen in LocalBatch, but higher variance in the coordinates of points in LocalBatch. Table 2 shows the performance of `Diverse Batch` for various eps values. We make the following observations. First, as expected, increasing eps decreases the number of training points and results in smaller batches, due to fewer clusters. Second, w/o averaging, the RMSE increases with eps; with $B = 20, E = 5$, RMSE is not affected by eps. On the other hand, EMD increases for larger eps in both cases, but with averaging EMD is higher overall. The percentage of diverged attacks increases with eps, which makes the attack more expensive and less accurate. Third, we consider two baselines: (i) the same number of points as DBSCAN in each round, but randomly selected; (ii) FedAvg with $B = 20$, $E = 5$ but LocalBatch =D_t^{target}: the EMD remains low regardless of eps since the batch variance is not controlled.

Performance of FedAvg with `Diverse Batch`. Fig. 11 shows that adding `Diverse Batch` to FedAvg ($B = 20, E = 5$) significantly improves privacy from $EMD = 9.7$ to $EMD = 15.23$. Random sampling of the same random of points per batch performs worse ($EMD = 6.6$), because it preserves the spatial correlation in trajectories. This indicates that the main benefit of `Diverse Batch` is controlling the batch location variance, rather than data minimization.

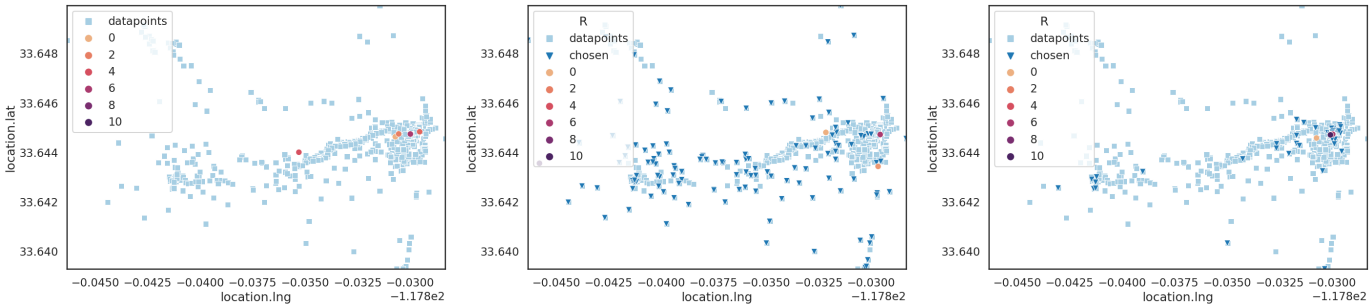
In Fig. 12, we compare all methods (in increasing privacy: FedSGD, FedAvg with mini-batches and epochs, and `Diverse Batch`) w.r.t. both privacy (EMD, divergence, distance from centroid) and utility (RMSE). (a) EMD starts at 7.5 in FedSGD, increases in FedAvg, and almost doubles (15) in `Diverse Batch`. (b) In FedSGD attack, there is almost no divergence, while with `Diverse Batch` more than 60% of the attacks diverge. Divergence makes the DLG attack more expensive, since the attacker needs to relaunch the attack with other initializations. (c) We also report the distance between the reconstructed location and the centroid of the batch $|x_{DLG} - \bar{x}|$: in FedSGD, this is less than 30 meters, while with `Diverse Batch` it increases to more than 350 meters.

4.4 FedAvg with Farthest Batch

Intuition. There are many ways to select the local batch so

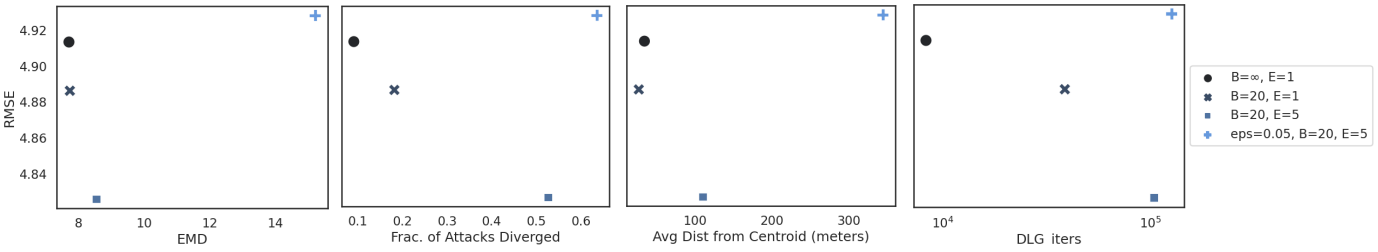
eps (km)	**Avg B** size	**% chosen** points	**RMSE FedSGD/FedAvg**	**EMD FedSGD/FedAvg**	**% diverged FedSGD/FedAvg**	**Avg Dist (m) FedSGD/FedAvg**	**Random** RMSE/EMD
0.0001	239	6.84	5.24/4.88	9.61/10.59	18/82	139/265	4.82/7.5
0.001	180	5.2	5.34/4.86	9.72/10.84	18/90	134/245	4.83/7.9
0.005	98	2.8	5.78/4.83	10.72/14.15	9/57	170/290	4.82/7.9
0.05	16	0.45	8.78/4.93	14.524/15.23	13/64	331/345	4.96/7.6

TABLE 2: **Performance of `Diverse Batch`.** *Parameters:* T =1-week; DBSCAN is run once in each round; $\eta = 0.001$, dropout=0.05. When two numbers are reported (X/Y) they correspond to FedSGD and FedAvg ($B = 20, E = 5$), respectively. For each value of the main parameter *eps* of DBSCAN we report the following metrics. Since `Diverse Batch` picks LocalBatch of different size in every round, we report the *average batch size*. Since it picks a subset of all data LocalBatch $\subset D_t^k$, we report of the *% of datapoints chosen*. The utility (*RMSE*) is not significantly affected by *eps*, but is improved by FedAvg, as expected. For privacy, we report the *EMD* between reconstructed and real locations, the *% of diverged attacks*, and the *average distance* $|x_{DLG} - \bar{x}|$ in meters. In general, as *eps* increases, privacy increase. In the last column, as a baseline for comparison, we report the utility and privacy if the same number of points as in column 3 are picked uniformly at random: the EMD is approx. half.



(a) FedAvg: B=20, E=5: EMD=9.7, RMSE=4.83. (b) Diverse Batch (selects dark blue points): EMD=15.23, RMSE=4.93. (c) Random baseline to Diverse Batch: EMD=7.6.

Fig. 11: **FedAvg with `Diverse Batch`.** Light blue shows the real locations of the target in `Campus Dataset` for $T = 1$ week. Dark blue shows the points chosen by `Diverse Batch` with $eps = 0.05$km. The colors show the reconstructed locations by DLG.



(a) EMD. (b) Diverged Attacks. (c) Distance from centroid. (d) Avg DLG iterations.

Fig. 12: **Privacy-utility trade-offs for all approaches.** Setup: `Campus Dataset`, 1-week intervals. Algorithms: FedSGD ($B = \infty, E = 1$), FedAvg ($B = 20, E = 5$), and `Diverse Batch`. Privacy metrics: EMD, divergence, distance from centroid. Utility: RMSE. The DLG attack is strongest in FedSGD. FedAvg improves both privacy and utility. FedAvg with `Diverse Batch` improves privacy (doubles EMD, increases divergence above 60%, and distance from 50 to 350m), without significantly hurting RMSE, and while using $< 1\%$ of the data.

as to mislead the attackers, *i.e.*, achieve high $|x_{DLG} - \bar{x}|$. We already presented `Diverse Batch` to maximize location variance in LocalBatch, by picking a subset of points from different DBSCAN clusters. An alternative way is `Farthest Batch` that we present here: we pick the subset of num points to include in LocalBatch $\subseteq D_t^{target}$, from the DBSCAN cluster that is farthest away (*i.e.*, has the highest distance from the batch's true centroid $\bar{x}$). According to our key observation (O1), the DLG attack will infer x_{DLG} close to the average location of the selected LocalBatch, which is, by construction, far away from the true average location $\bar{x}$ in the entire batch D_t^{target} in round t.

`Farthest Batch`. In each FL round t, the target does the following at line (16) of Algorithm 1:

1) Consider data D_t^{target} that arrived during that round as candidates to include in LocalBatch.
2) Apply DBSCAN on those points and identify the clusters, as shown at the line (10) of Algorithm 3.
3) Sort the clusters in decreasing distance between the centroid of that cluster and the true centroid of entire LocalBatch. Then, pick num data points, from the farthest to the closest clusters, and include them in LocalBatch, as shown at the line (11) of Algorithm 3.

Tuning `Farthest Batch`. For learning rate η, we adopt

Algorithm 3: `Farthest Batch`

Given: K users (indexed by k); B local mini-batch size; n_t is the total data size from all users at round t, η is the learning rate; the server aims to reconstruct the local data of target user k. w received from server, eps the DBSCAN parameter, num the number of points to select.

UserUpdate(k, w, t, B, eps, num**):**

Local data D_t^k are collected by user k during round t

Select LocalBatch $\subseteq D_t^k$ to use for training

C_t^k=$DBSCAN(D_t^k, eps)$

n_t^k=$Sort\_and\_select(C_t^k, num)$

$B_t^k \leftarrow$ (split LocalBatch into mini-batches of size B)

for *each local epoch i: 1...E* **do**

  for *mini-batch* $b \in B_t^k$ **do**

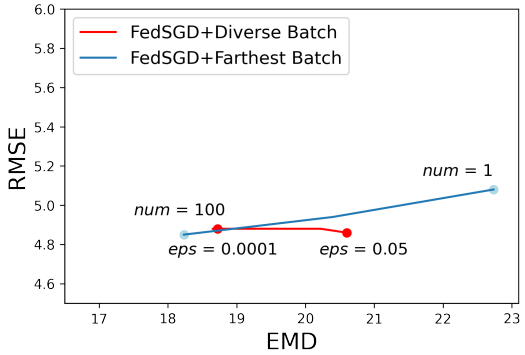
    $w \leftarrow w - \eta \nabla \ell(w; b)$

return w *to server*

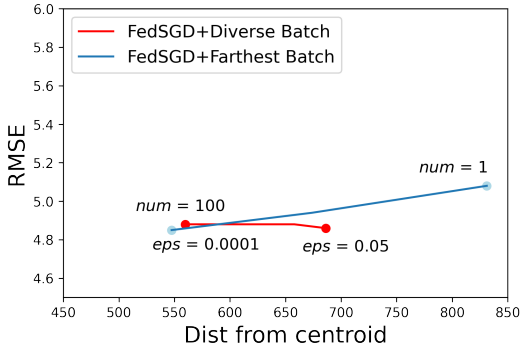
the same values $\eta = 0.001$ as in `Diverse Batch`. Regarding the DBSCAN parameter eps, we select $eps = 0.05$, which shows the best performance in Table 2. An important parameter of `Farthest Batch` is num, which controls the number of measurements selected in each LocalBatch. When $num = 1$, it means we only pick one measurement from the farthest cluster and include it into LocalBatch. Also, by choosing $num = 1$, it will provide the best privacy protection, and the worst utility. By increasing num, there are more measurements selected from farthest to closest clusters, which improves utility and degrades privacy.

Performance of FedAvg with `Farthest Batch`. Fig. 13 and 14 show that, when compared to `Diverse Batch`, `Farthest Batch` can enhance the privacy from $EMD = 20.147$ to $EMD = 22.91$ and from $Dist = 675.9$ to $Dist = 844.35$. in FedAvg. Although utility degrades when privacy increases, the loss of utility is still in a reasonable and acceptable range. This motivates us to use `Farthest Batch` for better privacy protection in this setting.

Discussion: Local Batch Selection. In vanilla FL, local batches are typically randomly selected. To the best of our knowledge, we are the first to notice that there can be many ways to pick a subset LocalBatch of all local measurements, collected and processed in round t, D_t^{target}, so as to construct $\bar{x}$ that is far from the true centroid, thus misleading the DLG attacker as per key observation O1 – which is specific to our setting. Which local batch selection algorithm performs better depending on the characteristics of the mobility patterns (e.g. how many important locations/clusters there are and how fast they change), as well as the time scale T over which new data arrive and are processed in a batch. In this paper, we proposed two intuitive such algorithms for local batch selection (`Diverse Batch` and `Farthest Batch`) and we showed that they achieve good privacy protection, without significant degradation in prediction, with `Farthest Batch` outperforming `Diverse Batch` in our evaluation. This good privacy-utility tradeoff was achieved while operating strictly within the "native" FL framework, and *without* orthogonal add-ons, such as DP that is discussed next.



(a) RMSE vs. EMD.



(b) RMSE vs. Distance from centroid.

Fig. 13: Comparison between `Diverse Batch` and `Farthest Batch` for 1-day interval using FedSGD.

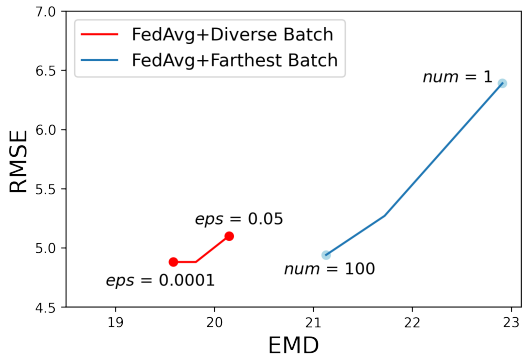
4.5 Baselines: DP, GeoInd, GAN Obfuscation

So far, we enhanced privacy through native FL mechanisms, *i.e.,* tuning the parameters and local batch selection. In this section, we explore how much benefit we get by adding state-of-the-art (but orthogonal) defenses, such as Differential Privacy (DP), Geo-Indistinguishability (GeoInd) [49] and Gan Obfuscation (Gan) [50], with both FedSGD and FedAvg and we compare it to `Farthest Batch` alone. We use $RMSE$ as the utility metric; EMD and $Distance\ from\ centroid$ as privacy metrics separately.

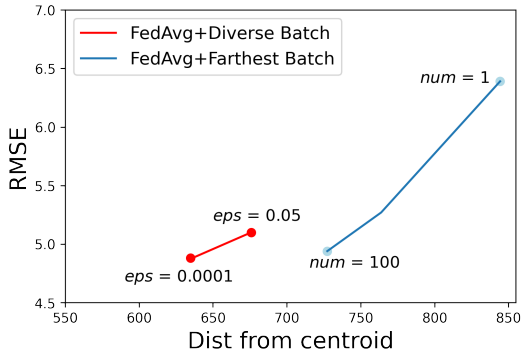
DP. In the FL setting, differentially private (DP) noise is typically added to the gradients of each LocalBatch, as in [51], [52]. Before transferring gradients from each user to the server, the gradients will first be clipped and then DP noise will be added. Clipping can bound the maximum influence of each user and we are using the fixed clipping method with parameter $\mathbb{C}$ in our experiments. We use (ϵ, δ)-DP bound for the Gaussian mechanism with noise N(0, σ^2), where $\sigma = \sqrt{2 \log \frac{1.25}{\delta}} \cdot \frac{\mathbb{C}}{\epsilon}$, $\delta = \frac{1}{\|LocalBatch\|}$. We choose $\mathbb{C}$ = 1 and δ =1e-5.

GeoInd. The mechanism of Geo-Indistinguishability (GeoInd) [49] is a variant of DP and is designed to provide strong privacy guarantees, specifically for location-based applications, by adding spatially controlled local noise to the user's location data. We use Geo_ϵ to represent the noise parameter of GeoInd in our evaluations.

GAN. Generative adversarial networks (GANs) have been applied before to crowdsourced signal maps, stored on a server, to protect the privacy of users [50]. Here, we apply

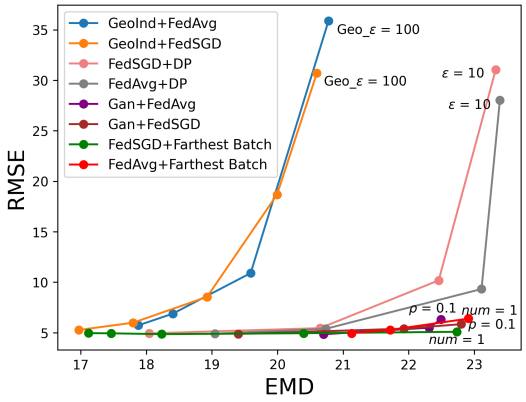


(a) RMSE vs. EMD.

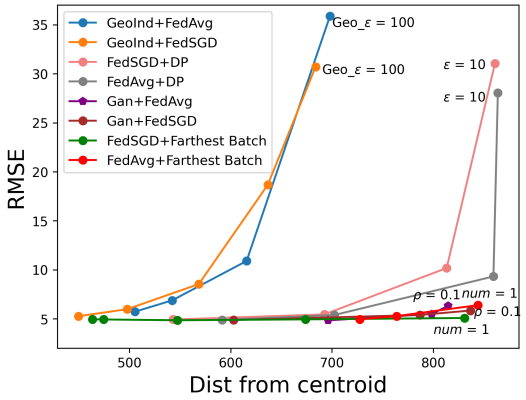


(b) RMSE vs. Distance from centroid.

Fig. 14: Comparison between `Diverse Batch` and `Farthest Batch` for 1-day interval using FedAvg.



(a) RMSE vs. EMD.



(b) RMSE vs. Distance from centroid.

Fig. 15: Comparison between Baselines (DP, GeoInd, Gan) and `Farthest Batch`, for rounds of 1-day, in terms of the privacy-utility tradeoff they achieve. Privacy is captured by two metrics: EMD and Dist from centroid. Model performance is captured by the prediction error (RMSE).

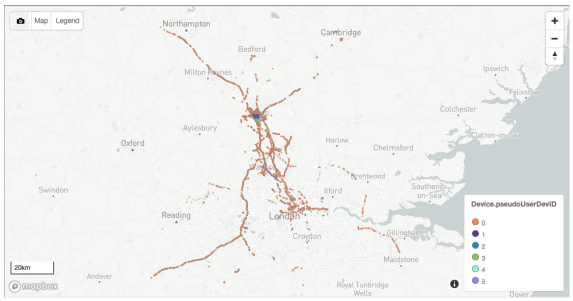
Gan Obfuscation to obfuscate users' private data before the data leaves the mobile device. The goal is to increase privacy such that it is difficult to recover sensitive features from the obfuscated data (*e.g.,* user ids and user whereabouts), while still allowing network providers to obtain accurate signal maps to improve their network services. We use ρ to represent the obfuscation level of Gan.

Comparison. By applying DP, GeoInd and Gan locally, each user can protect their private training data from DLG attacks. Figure 15 compares the local batch selection approach (using the best of our two algorithms `Farthest Batch`) against DP, GeoInd and Gan for 1-day intervals, in terms of the privacy-utility tradeoff they achieve. We observe that as ϵ, Geo_ϵ and ρ decrease, the privacy (in terms of EMD and $Distance\ from\ centroid$) improves, while the model performance captured by RMSE) deteriorates. We also observe that for the same privacy level, our `Farthest Batch` can provide more utility compared to differential privacy, GeoInd and Gan. Especially, when EMD is higher than 21 and $Distance\ from\ centroid$ is larger than 680, the utility loss in `Farthest Batch` is much smaller. Although Gan could provide similar utility as `Farthest Batch` for the same privacy level, Gan requires more computation resources since two additional neural networks need to be trained (i.e. a generator and an adversary) and it also requires hundreds of training rounds for these two neural networks to converge. Compared with Gan, our local batch selection approach is more efficient and lightweight. In summary, optimizing the batch selection to mislead the DLG attack (via `Farthest Batch`) achieves a better privacy-utility tradeoff compared to all three baselines. Please see Appendix B for additional evaluation results between `Diverse Batch`, `Farthest Batch`, DP, GeoInd and Gan.

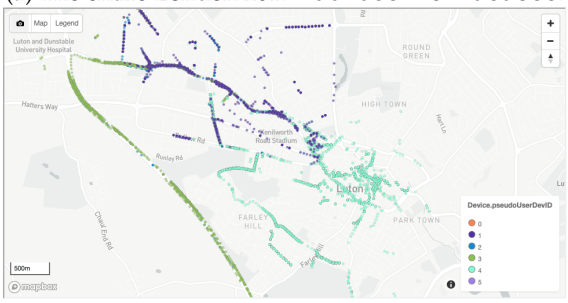
Discussion. The focus of this paper has been the privacy-enhancing design of FL-native local mechanisms for privacy, such as the tuning of local parameters and the local batch selection (*e.g.,* `Diverse Batch` and `Farthest Batch`.) These native mechanisms are necessary ingredients of FL itself and orthogonal to add-ons such as DP, GeoInd, Gan or SecAgg. The comparisons to DP, GeoInd and Gan are provided as a baseline for comparison against state-of-the-art (local) defense mechanisms. In the future, these add-ons (DP, GeoInd and Gan) can co-exist with and be combined with our local batch selection approach (`Diverse Batch` and `Farthest Batch`) further improve performance.

4.6 Multiple Users

Throughout this paper, we focus on the $target$ user exchanging local (w_t^{target}) and global (w_t) model parameter updates with the server, for $t = 1, \ldots, R$. When there are more users participating in FL, they contribute their updates and there is also global averaging of the gradients across users to produce w_t (line 12 in Algorithm 1). The data used for



(a) The entire London 2017 `Radiocells Dataset`.



(b) Zooming into the Luton area.

Fig. 16: The London 2017 `Radiocells Dataset`, partitioned into "synthetic users" 0-5, for cell x455. Background ("user 0", shown in orange,) contains the vast majority (1059 our of 1351) of upload files across the entire London region which are contributed by a large number of real users. Synthetic users 1-5 are also shown. Target is user 3 is shown in green.

updates across rounds may be different, depending on how similar the users' trajectories are. As discussed in (I4) in Sec. 2.2, when the diversity of data (across rounds) increases, convergence slows down and the gradient magnitude $|\bar{g}|$ remains large for more rounds, which makes the DLG attack more successful. Here, we evaluate the performance of all previous methods (FedSGD, FedAvg, FedAvg with `Diverse Batch`) under DLG attack, still from the perspective of a single target user, but considering the presence of multiple users updating the global model.

Creating Multiple Synthetic Users. We use the London 2017 part of the `Radiocells Dataset`, described in Sec. 3.1. Each upload file contains a sequence of measurements from a single device, but without specifying user pseudo-ids. In order to create realistic trajectories for the simulation of FL,we use heuristics to concatenate upload files, and we refer to the result as "synthetic" users.[5] The entire dataset (partitioned into synthetic users 0-5) is depicted in Fig.16. We confirmed visually and via pair-wise similarity (via EMD) that: users 1-4 are highly similar to each other, and dissimilar to users 0 and 5. We pick user 3, with the most measurements, as the target.

5. We start by grouping upload files with identical cell and device information (*i.e.,* device manufacturer, device model, software id and version, etc.). The device information was also used in [1] to create "users", but we also consider the cell id because we train a DNN model per cell. In addition, we merge files with the same device information as follows: if the start and end locations of two upload files are within a certain distance Z, then we assign them to the same "synthetic user". The intuition is that users typically repeat their trajectory across days, *i.e.,* a user starts/ends logging data from/to the same places, such as home or work. By setting $Z = 1$ mile, we obtain 933 potential synthetic users. We select five of them (users 1-5), each with a sufficiently high number of measurements per week and spanning several weeks. We merge *all* remaining upload files of the dataset (1059 out of 1351) into one "background" user 0.

Scheme	User(s)	RMSE	EMD	% diverged
FedSGD	user 3	6.1	17.08	65
FedAvg	user 3	5.43	24.13	91
FedAvg, $eps = 0.005$	user 3	5.42	25.2	97
FedAvg, $eps = 0.01$	user 3	5.44	26.9	97
FedAvg	user 3, 0	5.43	22.51	90
FedAvg, $eps = 0.01$	user 3, 0	5.41	23.01	90
FedAvg	user 3, 1	5.47	29.16	95
FedAvg	user 3, 2	5.43	26.15	95
FedAvg	user 3, 4	5.47	29.30	95
FedSGD	user 3, 5	5.82	18.02	59
FedAvg	user 3, 5	5.42	23.02	92
FedAvg, $eps = 0.01$	user 3, 5	5.41	31.33	96

TABLE 3: The effect of multiple users on the DLG attack. Parameters: `Radiocells Dataset`, $\eta = 0.001$; results are averaged over multiple runs. We compare FedSGD, FedAvg (with $E = 5, B = 10$) and FedAvg with `Diverse Batch` (with eps in km). Users 1,2,3,4 are similar to each other and dissimilar from users 0 (background) and 5; user 3 is the target.

Impact of Additional Users. We simulate DLG on FL with the target and additional synthetic users participating. We report the results in Table 3. First, we consider the target (3) alone: the DLG attack achieves: EMD 17.08, under FedSGD; 24.13, under FedAvg; up to 26.9, for FedAvg with `Diverse Batch` and $eps = 0.01$; the RMSE is roughly the same across all FedAvg and better than FedSGD.

Next, we consider that the background user 0 joins and updates the global model after locally training on its data. This is a realistic scenario of DLG on target 3, where the effect of most other real users in London is captured by this massive "background" user, who updates the global model following the timestamps indicated in the individual upload files. This results in a slight decrease of EMD for FedAvg from 24.13 to 22.51, while attack divergence remains practically the same.

Finally, we consider the other synthetic users (1,2,4,5), joining the target (3) one at a time, and updating the global model via FedAvg. When the most dissimilar user (5) participates, privacy loss is maximum (EMD is 18.02 with FedSGD and 23.02 with FedAvg). When similar users (1,2,4) participate, privacy loss is smaller (EMD $\in [26.1, 29.3]$). This was expected from Insight I4 in Sec.2.2: dissimilar users lead to slower convergence and more successful DLG attack.

Adding `Diverse Batch` to FedAvg offers significant protection to the target: for $eps = 0.005$, one third of the runs resulted in 100% divergence. Increasing eps to 0.01 resulted in two-thirds of runs with 100% divergence, and the remaining had EMD=26.9 and RMSE=5.44. In the multi-users case, it is sufficient that the target applies `Diverse Batch` locally, to get privacy protection, *e.g.,* EMD=31.33 and 96% divergence in the case of (3,5). This is amplified when users are dissimilar, like users (3,5), rather than the case of background (3,0).

5 Related Work

Signal Maps Prediction Framework. There has been significant interest in signal maps prediction based on a lim-

ited number of spatiotemporal cellular measurements [30]. These include propagation models [8], [9] as well as data-driven approaches ZipWeave [10], SpecSense [11], BCCS [12] and combinations thereof [13]. Increasingly sophisticated machine learning models are being developed to capture various spatial, temporal and other characteristics of signal strength [3], [14], [15] and throughput [17], [18]. The problem has been considered so far only in a centralized, not distributed setting. To the best of our knowledge, this paper is the first to consider signal maps prediction (i) in the FL framework (ii) considering online learning in the case of streaming data and (iii) a DLG inference attack on location.

Online federated learning is an emerging area [33], [34], [35]. To the best of our knowledge, existing work such as ASO-Fed [33], Fedvision [34], Fleet [35], in the online setting does not consider privacy leakage and focuses on convergence and device heterogeneity. RoF [53] and [54] focus on achieving efficient federated learning in industrial IoT but do not consider privacy leakage. Different communities [55] consider the case of online location data, although not in a FL way and with different goals (*e.g.,* location prediction, where the utility lies in the location itself).

Location Privacy. Numerous works evaluated location privacy or trajectory data, *e.g.,* [39], [40], [41], [56], [57], [58], Federated RFF KDE [59], where the utility of the dataset lies in the location itself. In this work, we focus on location privacy in mobile crowdsourcing systems (MCS), similarly to [1], [56]. As [1] pointed out, an important difference is that the utility in MCS does not lie in the location itself, but in the measurement associated with that location. In our case, the measurement is RSRP, and location is only a feature in an ML model. However, location is *the* primary feature needed to predict signal strength: additional features other than location and time (such as frequency, device information, environment *etc.*) bring only incremental benefit [3].

Reconstruction attacks and defenses based on gradients. It has been shown that observing gradients DLG [23], iDLG [38] gradients (as in FedSGD) or model parameters [24], ROG [60] (as in FedAvg) can enable reconstruction of the local training data. There has been a significant amount of prior work in this area, [23], [24], [38], [60], [61], [62], [63], [64], [65] to mention just a few representatives. Since gradient/model updates are the core of federated learning, FL is inherently vulnerable to such inference attacks based on observing and inverting the gradients.

Such attacks have been mostly applied to reconstruct image or text training data, with a few exceptions such as FedPacket [61] that inferred users' browsing history. "Deep leakage from gradients" (DLG) [23], reconstructed training data (images and text) and their corresponding labels, from observing a single gradient during training of DNN image classifiers, without the need for additional models (*e.g.,* GANs [63]) or side information. DLG [23] discussed potential defenses, including tuning training parameters such as mini-batch size, which we incorporate in our setting. We consider an attack that builds on "inverting the gradients" in [24]: using a cosine-based instead of the Euclidean distance, and also evaluating against FedAvg and the impact of averaged gradients due to local epochs. In our setting, the attack reconstructs only one (the average) location from a gradient update computed on N locations, as opposed to all N data points in a batch.

The work in [66] points out that information about a user's dataset can still leak through the aggregated model at the server, and provides a first analysis of the formal privacy guarantees for federated learning with secure aggregation. The work in [58] designs a scalable algorithm combining distributed differential privacy and secure aggregation to privately generate location heatmaps over decentralized data; however, the communication and computation costs brought by secure aggregation should be considered for mobile users. In contrast, in our work (1) we only utilize FL-"native" mechanisms to achieve location privacy, without added DP or SecAggr; and (2) we consider an online FL setting, where users collect data in an online fashion, and process them in intervals of duration T.

In Section 4.5, we considered two more privacy-enhancing techniques, beyond just DP. Geo-Indistinguishability (GeoInd) is a state-of-the-art, privacy-preserving method, inspired by DP and modified specifically to protect location data [49]. The GAN obfuscation technique we adopted from [50], applied generative adversarial networks to signal map datasets, stored at the server. We adapted the above techniques as well as DP, to be applied locally at the mobile.

Our prior work. In our prior work, we developed tools for collecting crowdsourced mobile data and uploading to a server [25]; the `Campus Dataset` collected therein is one we use for evaluation here as well. [3], we formulated a framework for centralized signal maps prediction using random forests. In [30], we extended the framework to provide several knobs to operators and allow them to optimize prediction when training on data of unequal important and/or for different tasks. In [50], we proposed a centralized GAN obfuscation technique to provide privacy for such tasks.

6 Conclusion

Summary. In this paper, we make three contributions. First, we design a lightweight online federated learning framework, specifically for the signal strength prediction *problem* in a crowdsourced setting. Second, we introduce a *privacy attack*, specifically for this framework: an honest-but-curious server employs gradient inversion to infer the location of users participating in the federated signal map framework. This DLG attack is specifically designed to reconstruct the average location in each round; this is in contrast to state-of-the-art DLG attacks on images or text, which aim at fully reconstructing all training data points. Third, we propose a *defense* approach that selects local batches so that the inferred location is far from the true average location, thus misleading the DLG attacker. Evaluation results show that our defense mechanisms achieve better privacy-performance trade-off compared to state-of-the-art baselines. Ultimately, the success of the attack depends not only on the FL algorithm and defenses used, but also on the characteristics of the underlying user trajectory data.

Future Directions. First, in terms of applications: signal maps prediction is a representative special case of predicting properties of interest based on crowdsourced spatiotemporal data. The same methodology to protect location privacy against DLG attacks can be applied to other learning tasks

that rely on such crowdsourced measurements. Second, in terms of methodology, there are several directions for extension. Due to the characteristics of human trajectories, there are more dependencies and opportunities to explore in the design of DLG attacks and defenses. Another direction is exploiting similarities and differences in users' trajectories, to further optimize aggregation schemes. Finally, this paper focused on designing and optimizing local FL-"native" privacy-preserving algorithms that can provide inherent privacy protection; these can be combined with other state-of-the-art but orthogonal defenses, such as DP or Gan.

REFERENCES

[1] S. Boukoros, M. Humbert, S. Katzenbeisser, and C. Troncoso, "On (the lack of) location privacy in crowdsourcing applications," in *USENIX Security Symposium*, 2019.

[2] J. Yang, A. Varshavsky, H. Liu, Y. Chen, and M. Gruteser, "Accuracy characterization of cell tower localization," in *Proc. of the ACM UbiComp '10*, 2010, pp. 223–226.

[3] E. Alimpertis, A. Markopoulou, C. Butts, and K. Psounis, "City-wide signal strength maps: Prediction with random forests," in *WWW*, 2019.

[4] Open Signal Inc., "Mobile Analytics and Insights," Jun. 2011.

[5] "Tutela," https://www.tutela.com.

[6] J. Garcia-Reinoso *et al.*, "The 5g eve multi-site experimental architecture and experimentation workflow," in *IEEE 5G World Forum*, 2019.

[7] A. Imran, A. Zoha, and A. Abu-Dayya, "Challenges in 5G: How to empower SON with big data for enabling 5G," *IEEE network*, 2014.

[8] Z. Yun and M. F. Iskander, "Ray tracing for radio propagation modeling: Principles and applications," *IEEE Access*, 2015.

[9] Y. J. Bultitude and T. Rautiainen, "IST-4-027756 WINNER II D1. 1.2 V1. 2 WINNER II Channel Models," *EBITG, TUI, UOULU, CU/CRC, NOKIA, Tech. Rep.*, 2007.

[10] M. R. Fida, A. Lutu, M. K. Marina, and O. Alay, "ZipWeave: Towards efficient and reliable measurement based mobile coverage maps," in *IEEE INFOCOM '17*, 2017.

[11] A. Chakraborty, M. S. Rahman, H. Gupta, and S. R. Das, "Specsense: Crowdsensing for efficient querying of spectrum occupancy," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[12] S. He and K. G. Shin, "Steering crowdsourced signal map construction via bayesian compressive sensing," in *IEEE INFOCOM*, 2018.

[13] C. Phillips, M. Ton, D. Sicker, and D. Grunwald, "Practical radio environment mapping with geostatistics," *Proc. of the IEEE DYSPAN '12*, pp. 422–433, Oct. 2012.

[14] A. Ray, S. Deb, and P. Monogioudis, "Localization of lte measurement records with missing information," in *IEEE INFOCOM*, 2016.

[15] R. Enami, D. Rajan, and J. Camp, "RAIK: Regional analysis with geodata and crowdsourcing to infer key performance indicators," in *Proc. of the IEEE WCNC*, Apr. 2018.

[16] E. Krijestorac, S. S. Hanna, and D. Cabric, "Spatial signal strength prediction using 3d maps and deep learning," *IEEE International Conference on Communications*, 2021.

[17] J. Wang, J. Tang, Z. Xu, Y. Wang, G. Xue, X. Zhang, and D. Yang, "Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach," in *IEEE INFOCOM '17*, May 2017, pp. 1–9.

[18] C. Zhang and P. Patras, "Long-term mobile traffic forecasting using deep spatio-temporal neural networks," in *ACM MobiHoc*, 2018.

[19] "Your Apps Know Where You Were Last Night, and They're Not Keeping It Secret," https://nyti.ms/3a5VCbp.

[20] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *International Conference on Artificial Intelligence and Statistics*, 2017. [Online]. Available: http://arxiv.org/abs/1602.05629

[21] L. Bottou and Y. LeCun, "Large scale online learning," *Advances in neural information processing systems*, 2004.

[22] M. Li, T. Zhang, Y. Chen, and A. J. Smola, "Efficient mini-batch training for stochastic optimization," in *ACM SIGKDD*, 2014.

[23] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *NeurIPS*, 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/60a6c4002cc7b29142def8871531281a-Paper.pdf

[24] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients - how easy is it to break privacy in federated learning?" in *NeurIPS*, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/c4ede56bbd98819ae6112b20ac6bf145-Paper.pdf

[25] E. Alimpertis and A. Markopoulou, "A system for crowdsourcing passive mobile network measurements," *14th USENIX NSDI*, vol. 17, 2017.

[26] "Radiocells Dataset," https://radiocells.org.

[27] C. Dwork, "Differential privacy," *Encyclopedia of Cryptography and Security*, pp. 338–340, 2011.

[28] M. Naseri, J. Hayes, and E. De Cristofaro, "Toward robustness and privacy in federated learning: Experimenting with local and central differential privacy," *arXiv preprint arXiv:2009.03561*, 2020.

[29] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[30] E. Alimpertis, A. Markopoulou, C. T. Butts, E. Bakopoulou, and K. Psounis, "A unified prediction framework for signal maps: Not all measurements are created equal," *IEEE Transactions on Mobile Computing*, 2022.

[31] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *JMLR*, 2018. [Online]. Available: http://jmlr.org/papers/v18/16-558.html

[32] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[33] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-iid data," in *IEEE Big Data*, 2020.

[34] Y. Liu, A. Huang, Y. Luo, Y. Huang, H.and Liu, Y. Chen, L. Feng, T. Chen, H. Yu, and Q. Yang, "Fedvision: An online visual object detection platform powered by federated learning," in *AAAI Conference on Artificial Intelligence*, 2020.

[35] G. Damaskinos, R. Guerraoui, A.-M. Kermarrec, V. Nitu, R. Patra, and F. Taïani, "Fleet: Online federated learning via staleness awareness and performance prediction," in *International Middleware Conference*, 2020.

[36] French, Robert M, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[37] Kemker, Ronald and McClure, Marc and Abitino, Angelina and Hayes, Tyler and Kanan, Christopher, "Measuring catastrophic forgetting in neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[38] B. Zhao, K. R. Mopuri, and H. Bilen, "idlg: Improved deep leakage from gradients," *arXiv preprint:2001.02610*, 2020.

[39] R. Isaacman, S.and Becker, R. Cáceres, S. Kobourov, M. Martonosi, J. Rowland, and A. Varshavsky, "Identifying important places in people's lives from cellular network data," in *International Conference on Pervasive Computing*, 2011.

[40] R. Becker, R. Cáceres, K. Hanson, S. Isaacman, J. M. Loh, M. Martonosi, J. Rowland, S. Urbanek, A. Varshavsky, and C. Volinsky, "Human mobility characterization from cellular network data," *Communications of the ACM*, 2013.

[41] Y.-A. De Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the crowd: The privacy bounds of human mobility," *Scientific reports*, vol. 3, p. 1376, 2013.

[42] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems*, 2020.

[43] Haddadpour, F. and Mahdavi, M., "On the convergence of local descent methods in federated learning," *arXiv preprint arXiv:1910.14425*, 2019.

[44] N. Bonneel, J. Rabin, G. Peyré, and H. Pfister, "Sliced and radon wasserstein barycenters of measures," *Journal of Mathematical Imaging and Vision*, vol. 51, no. 1, pp. 22–45, 2015.

[45] R. Flamary, N. Courty, A. Gramfort, M. Z. Alaya, A. Boisbunon *et al.*, "Pot: Python optimal transport," *JMLR*, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-451.html

[46] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *IEEE 23rd International Conference on Data Engineering*, 2007.
[47] M. Ma, "Enhancing privacy using location semantics in location based services," in *IEEE ICBDA*, 2018.
[48] Yin, Dong and Pananjady, Ashwin and Lam, Max and Papailiopoulos, Dimitris and Ramchandran, Kannan and Bartlett, Peter, "Gradient diversity: a key ingredient for scalable distributed learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018, pp. 1998–2007.
[49] Andrés, Miguel E and Bordenabe, Nicolás E and Chatzikokolakis, Konstantinos and Palamidessi, Catuscia, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, pp. 901–914.
[50] Zhang, Jiang and Clark, Lillian and Clark, Matthew and Psounis, Konstantinos and Kairouz, Peter, "Privacy-utility trades in crowdsourced signal map obfuscation," *Computer Networks*, vol. 215, p. 109187, 2022.
[51] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.
[52] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "Ldp-fed: Federated learning with local differential privacy," in *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, 2020, pp. 61–66.
[53] Zhang, Weiting and Yang, Dong and Wu, Wen and Peng, Haixia and Zhang, Ning and Zhang, Hongke and Shen, Xuemin, "Optimizing federated learning in distributed industrial IoT: A multi-agent approach," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3688–3703, 2021.
[54] Zhang, Weiting and Yang, Dong and Peng, Haixia and Wu, Wen and Quan, Wei and Zhang, Hongke and Shen, Xuemin, "Deep reinforcement learning based resource management for DNN inference in industrial IoT," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 8, pp. 7605–7618, 2021.
[55] A. K. Laha and S. Putatunda, "Real time location prediction with taxi-gps data streams," *Transportation Research Part C: Emerging Technologies*, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X18305400
[56] A. Pyrgelis, C. Troncoso, and E. D. Cristofaro, "Knock knock, who's there? membership inference on aggregate location data," in *NDSS*, 2018.
[57] M. Diao, Y. Zhu, J. Ferreira, and C. Ratti, "Inferring individual daily activities from mobile phone traces: A boston example," *Environment and Planning B: Planning and Design*, 2015.
[58] E. Bagdasaryan, P. Kairouz, S. Mellem, A. Gascón, K. Bonawitz, D. Estrin, and M. Gruteser, "Towards sparse federated analytics: Location heatmaps under distributed differential privacy with secure aggregation," *arXiv preprint arXiv:2111.02356*, 2021.
[59] Z. Zong, M. Yang, J. Ley, C. T. Butts, and A. Markopoulou, "Privacy by projection: Federated population density estimation by projecting on random features," *Proceedings on Privacy Enhancing Technologies*, vol. 1, pp. 309–324, 2023.
[60] K. Yue, R. Jin, C.-W. Wong, D. Baron, and H. Dai, "Gradient obfuscation gives a false sense of security in federated learning," *arXiv preprint arXiv:2206.04055*, 2022.
[61] E. Bakopoulou, B. Tillman, and A. Markopoulou, "Fedpacket: A federated learning approach to mobile packet classification," *IEEE Transactions on Mobile Computing*, 2021.
[62] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *IEEE Symposium on Security and Privacy (SP)*, 2019.
[63] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *ACM SIGSAC*, 2017.
[64] T. Chu, A. Garcia-Recuero, C. Iordanou, G. Smaragdakis, and N. Laoutaris, "Securing federated sensitive topic classification against poisoning attacks," *arXiv preprint arXiv:2201.13086*, 2022.
[65] K. Bonawitz, P. Kairouz, B. Mcmahan, and D. Ramage, "Federated learning and privacy," *Communications of the ACM*, vol. 65, no. 4, pp. 90–97, 2022.
[66] A. R. Elkordy, J. Zhang, Y. H. Ezzeldin, K. Psounis, and S. Avestimehr, "How much privacy does federated learning with secure aggregation guarantee?" *arXiv preprint arXiv:2208.02304*, 2022.
[67] C. Liu, M. Salzmann, T. Lin, R. Tomioka, and S. Süsstrunk, "On the loss landscape of adversarial training: Identifying challenges and how to overcome them," *arXiv preprint arXiv:2006.08403*, 2020.
[68] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *JMLR*, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

Evita Bakopoulou received her B.Sc. and M.Sc. degrees in Computer Science from Athens University of Economics and Business, Greece, in 2014 and 2016, respectively. She received a Ph.D. in Networked Systems from UC Irvine in 2021. She was a Summer Intern with Bell Labs (2017), Oath/Verizon Digital Media Services (2018) Google (2020), and joined Google as a full time Privacy Engineer in December 2021. Her research interests are primarily in the area of machine learning and privacy.

Mengwei Yang is currently a PhD student in the Electrical Engineering program at the University of California, Irvine, advised by Prof. Athina Markopoulou. Mengwei's research interests are in the areas of Federated Learning, Privacy-preserving machine learning and data privacy.

Jiang Zhang is currently a PhD student in the Ming Hsieh Department of Electrical and Computer Engineering at the University of Southern California. His research interests include machine learning and data privacy with applications to mobile networks.

Konstantinos Psounis (Fellow, IEEE) received the BS degree from the Department of Electrical and Computer Engineering, National Technical University of Athens, Greece, in 1997, and the M.S. and Ph.D. degrees in Electrical Engineering from Stanford University, Stanford, CA, USA, in 1999 and 2002, respectively. He is currently a Professor of Electrical and Computer Engineering and of Computer Science with the University of Southern California. He works on modeling, performance analysis, algorithm design, and system implementation for efficient and privacy-preserving networked, distributed systems. He is a Distinguished Member of the ACM.

Athina Markopoulou (S'98-M'02-SM'13-F'21) received the Diploma degree in Electrical and Computer Engineering from the National Technical University of Athens, Greece, in 1996, and the M.S. and Ph.D. degrees in Electrical Engineering from Stanford University in 1998 and 2003, respectively. She joined the faculty of EECS Department at UC Irvine in 2006, where she is currently a Professor and Chair. She has held short-term positions at SprintLabs, Arista Networks, and IT University of Copenhagen. She has received the NSF CAREER award in 2008, the HSSoE Faculty Midcareer Research Award in 2014, the OCEC Educator Award in 2017, and a UCI Chancellor's Fellowship in 2019. She has served as an Associate Editor for IEEE/ACM Trans. on Networking and for ACM Computer Communications Review, as the General Co-Chair for ACM CoNEXT 2016, as TPC Co-Chair for ACM SIGMETRICS 2020 and NetCod 2012. She is also a Distinguished Member of the ACM. Her research interests include network measurement, privacy, mobile and IoT, social networks.