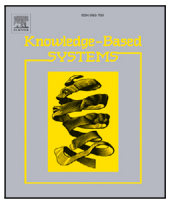
Knowledge-Based Systems 343 (2026) 115993

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys



Metric-privacy-inspired noise calibration in federated learning: Improving convergence and preventing client inference attacks

Judith Sáinz-Pardo Díaz [a,*], Andreas Athanasiou [b,c,1], Kangsoo Jung [c], Catuscia Palamidessi [c], Álvaro López García [a]

[a] Instituto de Física de Cantabria (IFCA), CSIC-UC, Avda. los Castros s/n, 39005 Santander, Spain
[b] Technische Universiteit Delft (TU Delft), Mekelweg 5 2628, Delft, Zuid-Holland, Netherlands
[c] INRIA Saclay, École Polytechnique (LIX), 1 Rue Honoré d'Estienne d'Orves 91120, Palaiseau, Île-de-France, France

ARTICLE INFO

Keywords:
Federated learning
Differential privacy
Metric-privacy
Data privacy
Medical imaging

ABSTRACT

Federated learning (FL) enables the training of a global model across multiple data owners (clients) without sharing raw data. This distributed architecture is orchestrated by a central server that aggregates the local models from the clients. In cases where the server is trusted but not all network nodes, differential privacy (DP) can be used to privatize the aggregated model by adding noise. However, this may affect convergence across the FL rounds. In this work, we build on the notion of metric-privacy as a design principle to calibrate the noise added by the server under a global-DP setting, with the objective of mitigating its impact on the convergence of the aggregated model. We do not enforce metric-privacy as a formal guarantee, but rather use it to guide noise calibration. We compare our approach with vanilla FL and global-DP by analyzing the impact on six aggregation strategies and applying it to a medical imaging use case, simulating different scenarios with homogeneous and non-i.i.d. clients. Finally, we introduce the client inference attack (CIA), where a semi-honest client tries to find whether another client participated in the training and study how it can be mitigated using DP and metric-aware noise calibration. Our experiments show that metric-privacy aware noise calibration strategy improves the accuracy compared to standard DP in all the scenarios analyzed, while achieving a comparable success rate against CIA. These results indicate that metric-privacy inspired noise calibration can deliver a superior utility-privacy trade-off in medical-imaging federated settings.

1. Introduction

The development of artificial intelligence models, such as machine and deep learning (ML/DL) and their application on distributed data is a field of emerging interest, especially with the rise of *federated learning* [1]. This architecture allows to train a global ML/DL model on multiple data owners without sharing their data with each other or with a third party [2].

Federated learning (FL) is considered a privacy preserving machine learning architecture, since the global model is built from local models trained by each data owner (in the following, client), without the need for the data to leave the devices, institutions or data centers that generate and/or store it. However, these approaches are not free from attacks on the privacy of the clients involved and the data used, with the most notable being membership inference attacks [3,4], reconstruction attacks and model inversion attacks [5,6] among others.

Differential privacy (DP) and homomorphic encryption (HE) are commonly included in FL architectures to add an additional layer of privacy. For example, if we want to perform a secure model aggregation, HE can be applied so that the models are aggregated in an encrypted way [7]. Another option is to add noise to each local model before sending it for aggregation (applying DP), in case the aggregator (in the following, the server) is not trusted.

A variant of standard DP is *metric-privacy* (also known as *d-privacy*) [8,9] which can be employed in domains with a notion of distance. Like standard DP, metric-privacy offers a bound on the probability that the same result is obtained from two different datasets. This probability depends on a parameter ϵ (like standard DP) but also on the distance between the two datasets (unlike standard DP). In other words, metric-privacy suggests that noise should be calibrated according to the distance between inputs, offering stronger privacy guarantees when

* Corresponding author.
E-mail addresses: sainzpardo@ifca.es (J. Sáinz-Pardo Díaz), a.athanasiou@tudelft.nl (A. Athanasiou), gangsoo.zeong@inria.fr (K. Jung), catuscia@lix.polytechnique.fr (C. Palamidessi), aloga@ifca.unican.es (Á. López García).
[1] Work performed partly at TU Delft and partly at INRIA.

https://doi.org/10.1016/j.knosys.2026.115993
Received 15 May 2025; Received in revised form 23 March 2026; Accepted 10 April 2026
Available online 16 April 2026

inputs are close and weaker ones when they are far apart. This is especially beneficial in applications where hiding an element within a group of neighbors provides adequate privacy protection. However, metric-privacy has received little attention in FL.

Protecting against an untrusted central server using DP has been widely studied in FL. However most studies focus only on the server using *FedAvg* (Federated Average) as the aggregation function. Therefore, there is a gap in the literature regarding the impact of global-DP in the accuracy of the model when other aggregation functions are used.

Moreover, in this paper we also pose a different problem together with its potential solution. Specifically, we assume that all clients are semi-honest and that the server (i.e. aggregator) is trusted. However, we suppose that one of the clients can act as an attacker to find out if another one is participating in the training or not. We name this a *Client Inference Attack* (CIA). Knowing that a specific client participated in the dataset can lead to further attacks, which can be a privacy concern. For instance, it may strengthen other attacks, such as the membership inference attack.

Our approach, which will be detailed in Section 6, exploits the intuition behind metric-privacy from the server side to calibrate the noise added to the aggregated model, with the objective of improving utility while maintaining protection against this type of attack. Thus, it is important to note that we do not enforce metric-privacy as a formal guarantee, but rather use it as a design principle to guide noise calibration. Accordingly, throughout this work, the term "metric-privacy-inspired noise calibration" (or "metric-aware noise calibration") refers to a heuristic approach that dynamically scales the standard deviation of the Gaussian noise using a distance metric derived from the local model parameters. Hereafter we will refer to this approach as metric-privacy for simplicity of terminology.

For the experimental part of this work, an openly available dataset belonging to the field of medical imaging has been used, simulating different scenarios in terms of client distribution (homogeneous or non-i.i.d.), as well as an additional scenario to test the protection against the proposed client inference attacks.

1.1. Contributions and structure of the work

The main contributions of this work can be summarized as follows:

- First, we apply global-DP (i.e. server-side DP) to test the performance of the model across six different aggregation strategies. This yields interesting results as global-DP affects the performance of the model differently depending on the chosen aggregation function. We experimentally evaluate the performance of metric-privacy, global-DP and, as a baseline, vanilla FL (i.e. without adding any noise at all), across different aggregation functions, showing that metric-privacy offers in all cases a better model accuracy than global-DP. The experiments are conducted using a medical imaging database openly available and simulating different clients and scenarios. To the best of our knowledge this comparison between different aggregation strategies and privacy approaches has not been studied before in scientific papers.
- We study the application of metric-privacy principles in FL assuming a trusted server by introducing a proper distance metric. Different metrics have been evaluated and can be applied adapting the code implemented which is publicly available (see Appendix H). This distance metric is dynamically calculated from the server side in each round only taking into account the local model parameters (i.e. without analyzing the clients' data distributions, which could affect privacy).
  As far as we are aware of, this is the first work that incorporates metric-privacy-inspired noise calibration into an FL architecture, thereby constituting a novel contribution to the state of the art in utility-aware privacy-preserving FL.
- We introduce the concept of *Client Inference Attack* (CIA) where a semi-honest client tries to infer whether or not a certain client is participating in the training. We conduct experiments showing that while both metric-privacy and global-DP provide adequate protection against this type of attack, the former produces a more accurate model. To our knowledge, this is the first time that these types of attacks have been introduced in the literature along with approaches to assess and mitigate them.

This work is structured as follows: first, in Section 2 we report the state of the art and the related work. Section 3 presents different aggregation strategies that can be used in federated learning. Then, Section 4 provides the theoretical basis for using DP in federated learning. In Section 5 we define the client inference attacks. Section 6 explains the methodology followed and how we can apply metric-privacy in FL. Next, in Section 7 we conduct experiments across homogeneous and non-i.i.d clients and two CIA scenarios. Finally, Section 8 draws the conclusions and outlines future work.

2. Related work

The incorporation of DP in data science ecosystems, especially in relation to the training of ML/DL models, is a topic that has been extensively studied in the field of privacy-preserving machine learning [10]. Regarding the training of deep learning models (more specifically deep neural networks), the use of DP during the model training has been explored in multiple works following the differentially private stochastic gradient descent (DP-SGD) approach [11–13], which consists of adding noise during the process of stochastic gradient descent. This is implemented in different Python libraries, like TensorFlow Privacy [14] or Opacus [15].

FL is susceptible to privacy attacks, despite the fact that each client trains their model locally, without revealing their local dataset. Among various inference attacks, the Membership Inference Attack (MIA) [3,16,17] determines whether a particular record is part of a dataset or not. To do so, it is assumed that the adversary (central server) owns a so-called shadow model of the target client, which is similar to the target client's model (i.e. follows the same distribution) but is trained on a different, disjoint, dataset. MIAs are widely employed in the literature for evaluating privacy risks [18].

The Source Inference Attack (SIA) [4] is a more advanced inference attack. An honest-but-curious server in FL attempts to identify exactly which client owns a specific data point used for training. To estimate the source of the data point, the server leverages the prediction losses of local models on particular data points by using a Bayesian approach.

Among reconstruction attacks, arguably the most well-known is Deep Leakage from Gradients (DLG) [5]. DLG usually exploits the shared gradients during the training process in federated learning to reconstruct sensitive training data. This attack shows that even gradient information can leak important details about the original data. In both reconstruction and inference attacks, applying DP is crucial to protect sensitive information of the participating clients.

However, in FL adversaries may also take the form of participants or other external actors with access to the (shared) aggregated model. In the case of an FL architecture where the server that orchestrates the training is not trusted, clients can add DP during training by applying DP-SGD, but they can also do so once the model is trained, adding noise to the model updates [19]. This will prevent the central server from extracting information from the data used in the training by providing local differential privacy (LDP).

This paper proposes an experimental setup applied to two use cases related to the field of medical imaging. It is therefore important to note that the conclusions and results obtained are robust for the simulations performed in these scenarios (distinguishing between homogeneous and non-i.i.d. client cases for greater consistency). This is in line with the state of the art regarding privacy-preserving machine learning

approaches and the growing focus in the healthcare sector, where the handling of sensitive data is a critical issue. Specifically, multiple studies related to FL address its application to cases in the medical field, such as [20], that presents current trends, challenges, and future work concerning FL in medical applications, or [21] that shows a concrete FL application for predicting clinical outcomes in patients with COVID-19. Concerning medical imaging, FL and DP, in [22] the authors apply DP-SGD in a FL architecture for medical image analysis, specifically using histopathology images. In the same line, several works have explored the incorporation of DP to FL architectures, such as [23], where DP is added from each client to the model updates before sending the model to the server, specifically in a personalized federated learning approach.

In this work we assume that the server is trusted. The attacker can either be another client who tries to infer information about other clients (e.g. find out if a certain client is participating in the training). Hence the model that we aim to protect is the aggregated (global) one. Therefore, the server can globally add DP to the aggregated model after receiving the updates from all the clients. This will prevent the extraction of information by the clients in each round. Also it will protect the final resulting model when it is made publicly available. Few works can be found in the literature concerning applying DP from the server side. For example, in [24] local and central DP are compared in a FL setting showing the protection against backdoor attacks, defining central DP (CDP) as the case in which *"the FL aggregation function is perturbed by the server"*. In this work we refer to this approach as *global-DP* (GPD). Concerning GDP in FL, the server clips the local model updates and then add noise to the aggregated model. Specifically, we can consider fixed clipping and adaptive clipping strategies. In the first case, a fixed threshold is predefined, and in the second, this threshold is dynamically adjusted based on the updates received. Returning to LDP in FL, adaptive clipping has also been analyzed in recent studies such as [25,26], while the number of studies using GDP in FL is very limited.

This paper seeks to expand the state of the art in the field by introducing the notion of metric-privacy adapted to the case of its integration from the server side in FL. To our knowledge, this is the first paper that proposes this methodology with the aim of calibrating the noise added in each round.

3. Federated learning aggregation functions

The classic horizontal federated learning architecture involves a server and multiple clients or data owners. On the one hand, the clients have their data stored locally in their own infrastructures and are in charge of performing the model training in a distributed way. On the other hand, the server is in charge of aggregating the individual locally trained models in order to build a global model. Thus, the scheme followed under a FL architecture can be summarized in the following steps: (1) each client receives a model to be trained locally (defined by the server or consensuated by the clients); (2) each client trains the model locally with its data; (3) the clients send the parameters or weights that define the model trained locally (or the model updates); (4) the server receives the models/updates from the clients involved and aggregates them to build a global model; (5) the aggregated global model is distributed back to the clients to be retrained. This process is repeated from step (2) for a pre-defined number of rounds.

One of the most critical steps is the aggregation process carried out from the server side, since it is the moment when the global model is built from all the locally trained models. Thus, selecting an appropriate aggregation strategy in relation to the use case employed is key to achieve an aggregated global model that is robust and accurate. Some classically applied aggregation functions are presented below:

Federated average (FedAvg). The most natural approach is to simply average the parameters. However, it must be taken into account that there will be clients who have trained the model using more data than others, thus being able to obtain a more accurate prediction. Thus, it is important that the average is made in a weighted way taking into account the number of data available for each client. Proposed by B. McMahan et al. in 2017 [2] when introducing the FL architecture (where its successful application was shown for the MNIST, CIFAR10 and Shakespeare datasets), the Federated Averaging strategy (*FedAvg*) is the most widely used aggregation function in the literature for its simplicity and effectiveness.

Let n be the number of clients participating in the FL architecture, n_i the number of data of client i ($\forall i \in \{1,\dots,n\}$) and $w_i^{(r)}$ the weights or model parameters obtained for each client i after training in the round r. The aggregation performed with *FedAvg* consists on the weighted mean of the parameters of each clients, as follows, for round r:

$$w^{(r)} = \frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} w_i^{(r)} n_i \tag{1}$$

Then, $w^{(r)}$ are the aggregated weights resulting from round r that will be transmitted to each client when starting round $r+1$.

Federated average with momentum (FedAvgM). This strategy was implemented in [27] and the proposed *FedAvgM* method arises from the idea of improving stability and efficiency by using the momentum to store gradients from previous rounds, also in order to dampen oscillation in the training. Then, let $w^{(r)}$ the global parameters in round r and $\Delta w^{(r+1)}$ the aggregated difference between the parameters obtained in rounds $r+1$ and r: $\Delta w^{(r+1)} = \frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} n_i (w^{(r)} - w_i^{(r+1)})$, note that $w_i^{(r+1)}$ are the local parameters obtained for client i in round r.

In order to get the aggregated parameters using *FedAvgM* we have to calculate the momentum vector in round r, $v^{(r)}$ (initialized as $v^{(0)} = 0$), in such a way that $v^{(r+1)} = \beta v^{(r)} + \Delta w^{(r+1)}$. Then, we will calculate the weights in round $r+1$ as $w^{(r+1)} = w^{(r)} - v^{(r+1)}$, with β being the momentum. Additionally, the term μ can be added as the server learning rate. It can be introduced as follows: $w^{(r+1)} = w^{(r)} - \mu v^{(r+1)}$.

Thus, the main idea of *FedAvgM* is to incorporate the notion of momentum at the server side during the aggregation process to improve the stability and speed of convergence of the overall model [27]. According to the implementation in the *Flower* library [28], if the value for the momentum (β) is strictly greater than zero, some global initial parameters for the model must be included. As will be explained in Section 7, a validation dataset will be used to get these initial model parameters.

Federated median (FedMedian). The Federated Median strategy (*FedMedian*) was introduced in [29]. The proposed algorithm for aggregating using the median takes into account byzantine machines so that only in the case of normal worker machines each such client computes the local gradient. Thus, let $g_i(w^{(r)})$ be the gradients obtained for client i with the weights aggregated in round r (w_r). The central server will be in charge of getting the median of the local gradients in each round r: $g_{median}(w^{(r)}) = median(g_i(w^{(r)}), i \in \{1,\dots,n\})$. Then, once such aggregated gradient is calculated, the weights are updated for round $r+1$: $w^{(r+1)} = w^{(r)} - \mu g_{median}(w^{(r)})$ [29], being μ the step-size or learning rate.

FedProx. The main objective of the *FedProx* aggregation strategy is to deal with heterogeneous settings in FL schemes [30]. Specifically, it is a generalization and re-parametrization of the *FedAvg* strategy. In order to calculate $w_i^{(r+1)}$ (the weights for client i in round $r+1$) we have to minimize the following objective function (h_i) for each client i, with $F_i(\cdot)$ the global objective function at each client i:

$$h(w, w^{(r)}) = F_i(w) + \frac{\mu}{2} \|w - w^{(r)}\|^2.$$

Thus, $w_i^{(r+1)} = arg\min_w h_i(w)$. Then the server aggregates the values $w_i^{(r+1)}$ $\forall i \in \{1,\dots,n\}$ by taking the mean.

Note than in this approach we need to choose a value for the parameter μ.

Federated optimization (FedOpt) and adaptive federated optimization with yogi (FedYogi). The *FedOpt* strategy introduced in Algorithm 1 from [31] aims to improve *FedAvg* by using gradient-based optimizers with given learning rates that must be customized both from the client and server side. Specifically, let $w^{(r)}$ the global weights at round r and $w_i^{(r)}$ the weights for client i at round r, then we can define $\Delta w_i^{(r)} = w_i^{(r)} - w^{(r)}$ and $\Delta w^{(r)} = \frac{1}{n}\sum_{i=1}^{n} \Delta w_i^{(r)}$. From the server side, we will use an optimizer for calculating $w^{(r+1)}$ given $w^{(r)}$, $\Delta w^{(r)}$ and the learning rate.

Note that this approach allows the use of different adaptive optimizers in the server side, such us *Adam*, *Adagrad* or *Yogi* among others (the we will have the *FedAdam*, *FedAdagrad* and *FedYogi* aggregation strategies respectively). The adaptive methods are used on the server side while SGD is used from the client side [31]. More details are given in the pseudocode of Algorithm 2 from [31].

These strategies require additional parameters, such as the server-side and client-side learning rate, β_1 and β_2 as momentum and second momentum parameters respectively and τ for controlling the degree of adaptability of the algorithm. According to its implementation in Flower, some initial global parameters for the model need to be introduced when using these functions. The use of these global initial parameters provides a consistent starting point for all clients.

4. Differentially private federated learning

Differential privacy (DP) is a privacy preserving technique that aims to provide a formal guarantee about what an analyst (adversary) can learn about an individual in a database. The probability that the adversary observes any event is comparable in cases where certain information from an individual is or is not included in the dataset. In addition, with DP we can ensure that an adversary with unlimited computational capacity and auxiliary information cannot break the established privacy level.

In other words, DP states that an algorithm is differentially private if by viewing its result an adversary cannot know whether a particular individual's data is included in the database or not. This is typically done with the addition of noise (i.e. data obfuscation). The goal is to protect user's privacy while allowing the data to be meaningful for analysis. For this purpose, different mechanisms can be followed to ensure that the added noise does not significantly alter the analysis.

Let us start by introducing the notions of adjacency and randomized algorithm which are crucial for defining DP:

Definition 1. Two databases D and $\overline{D}$ are adjacent (notated by $D \sim D'$) if they differ by exactly one record.

Definition 2. A probabilistic or randomized algorithm for the query f over the database $\mathcal{X}$ is a probabilistic function g from $\mathcal{X}$ to a set of values $\mathcal{Z}$ such as $g : \mathcal{X} \longrightarrow \mathcal{D}(\mathcal{Z})$, with $\mathcal{D}(\mathcal{Z})$ the set of probability distributions in $\mathcal{Z}$.

Now we can define ϵ-*differential privacy*:

Definition 3. A randomized algorithm $\mathcal{M}$, with domain $\mathcal{D}$ and range $\mathcal{R}$, satisfies ϵ-differential privacy if for every pair of adjacent datasets $D, \overline{D} \in \mathcal{D}$, for every $S \subseteq \mathcal{R}$ and $\epsilon > 0$:

$$\mathbb{P}[\mathcal{M}(D) \in S] \leq e^{\epsilon}\mathbb{P}[\mathcal{M}(\overline{D}) \in S]$$

In view of the above definition, the value of ϵ is the *privacy budget*, which allows to control the level of privacy (the amount of privacy loss allowed). We note that the lower the value of ϵ, the higher the privacy, but, in most of the cases, the lower the usefulness of the data for analysis (although this is not always the case [32]).

Approximate DP is a famous relaxation of the previous definition where we allow the above inequality to not hold with a small probability. We can define (ϵ, δ)-*differential privacy* as:

Definition 4. A randomized algorithm $\mathcal{M}$, with domain $\mathcal{D}$ and range $\mathcal{R}$, verifies (ϵ, δ)-differential privacy if for every two pair of adjacent datasets $D, \overline{D} \in \mathcal{D}$ and for every $S \subseteq \mathcal{R}$, $\epsilon > 0$ and $\delta \in [0, 1]$:

$$\mathbb{P}[\mathcal{M}(D) \in S] \leq e^{\epsilon}\mathbb{P}[\mathcal{M}(\overline{D}) \in S] + \delta$$

The parameter δ is the probability of exceeding the privacy budget, i.e. with probability $1 - \delta$ the privacy loss will not be greater than ϵ.

Some of the most commonly used mechanisms to ensure DP are among others the Laplace Mechanism, the Exponential Mechanism and the Gaussian Mechanism.

Let us introduce the Gaussian Mechanism, as it will be used during the experimental settings performed in this work.

First, let us define the notion of sensitivity:

Definition 5. Be $f : \mathcal{D} \longrightarrow \mathbb{R}^k$, the l_2-sensitivity of f is defined as follows:

$$\Delta_2(f) := \max_{\|x-y\|_1} \|f(x) - f(y)\|_2$$

Now we can define the Gaussian Mechanism as given in Definition 6:

Definition 6. Given the function $f : \mathcal{D} \longrightarrow \mathbb{R}^k$, we define the Gaussian Mechanism ($\mathcal{M}_G$) for (ϵ, δ)-differential privacy as follows:

$$\mathcal{M}_G(x, f(\cdot), \epsilon, \delta) := f(x) + (Y_1, \ldots, Y_k).$$

Note that Y_i $(\forall i \in \{1, \ldots, k\})$ follows the distribution $N(0, \sigma^2)$, with $\sigma = \frac{\Delta_2(f)\sqrt{2\log(1.25/\delta)}}{\epsilon}$.

DP can be applied using two approaches: locally and globally. In the local setting we assume that the clients have to obfuscate their data on their own, using local-DP (LDP). On the other hand, in the global setting the clients totally trust a central entity to collect their data and add noise to the output of a query or algorithm, using global-DP (GDP). Note that trusting the curator or central server is a vital assumption. If such assumption cannot be made using local-DP is a better choice.

If we extrapolate these notions to a FL architecture, we can think of two paradigms depending on the side of the network in which the potential attacker is located:

- *Trusted server:* in this case we can add DP from the server side. The goal here is to protect against some clients that can act as attackers, or to prevent information extraction from the final global model when it is published. This is exactly our aim in this work.
  In this line, the idea is to perturb the global aggregated model by adding noise to it once aggregated. In this work, we analyze the impact of adding such noise according to the aggregation strategy applied, assuming that we have a network composed by a trusted honest server.
- *Untrusted server:* in this case the clients will be interested in sending to the server a noisy version of the model. This is because some information could be extracted concerning the data of each client by analyzing the local updates that are sent in each round. In this line the clients can train the model and then add noise before sending to the server for aggregating, or to perform differentially private stochastic gradient descent (DP-SGD) for adding DP during the training process. Recall that we do not consider this scenario in this work.

Another variation of DP is *metric-privacy* or *d-privacy* [8,33]. In metric-privacy the datasets are not just adjacent (i.e. they just have one value that is not the same) but their actual difference is calculated using some distance metric. Then, the privacy level can be adapted so as it offer better privacy when the distance is small. In other words, metric-privacy allows the adversary to find a rough estimation of the clients' data, but the actual values remain protected. This is done in exchange of better utility, as less noise is added.

We can define metric-privacy as:

Definition 7. Let D be a domain provided with a metric $d : D^2 \longrightarrow \mathbb{R}_{\geq 0}$. The randomized algorithm $\mathcal{M}$ with range $\mathcal{R}$ and with domain D satisfies ϵ-metric-privacy, if for every two pair of inputs $x, \overline{x} \in D$ with distance $d(x, \overline{x})$, every $y \in \mathcal{R}$ and every $\epsilon > 0$:

$$\mathbb{P}[\mathcal{M}(x) = y] \leq e^{\epsilon \cdot d(x,\overline{x})} \mathbb{P}[\mathcal{M}(\overline{x}) = y]$$

In this work our goal is to evaluate the impact of metric-privacy and classic global-DP according to different aggregation strategies in FL architectures. Note that our goal is to apply metric-privacy-inspired noise calibration in each round of the FL architecture, using the distance between local models to dynamically adjust the noise injected into the aggregated model

Finally, note that we aim to analyze in practice how metric-privacy-inspired noise calibration can improve the convergence of a federated model with respect to classic global-DP (in the following metric-privacy for simplicity) with respect to classic global-DP, while we are protecting the aggregated model from information extraction and ensuring a similar protection regarding client inference attacks. We want to highlight that this work does not aim to perform a fair comparison between global-DP and metric-privacy (as the amount of noise added in each approach is different, as will be explained later), but to introduce this notion for better tuning the amount of noise added for finding a balance between predictive performance and resistance against client inference attacks.

5. Client inference attacks

One of the main hypotheses that we want to assess within this work is that metric-privacy, besides offering a better accuracy than standard global-DP in FL, may also be helpful against inference attacks.

In this section, we introduce the *Client Inference Attack (CIA)*, a novel attack where a semi-honest client (attacker) receives the global aggregated model from an honest (trusted) server and tries to determine if another client belongs to the list of participants. This attack can serve as a first step to strengthen other attacks, such as a Membership Inference Attack [3] or a Source Inference Attack [4].

For instance, consider a model jointly built by hospitals and competing pharmaceutical companies to treat a disease. No party (apart from the central server) knows exactly which hospitals or which companies are working on the model. The central server is honest and releases the aggregated model back to the clients. However, a curious pharmaceutical company P may want to launch a CIA attack and figure out whether or not a certain hospital H is participating in the training. If this CIA shows that H did indeed participated, then P can facilitate further inference attacks on H and the data used (which may be associated with the patients from the hospital).

5.1. Threat model and definition

In order to achieve this attack, we assume that the attacker has sufficient knowledge of the target client's training dataset to create a *shadow training dataset*; a dataset that mimics the distribution of the target client's dataset. This shadow dataset can either be entirely disjoint from the target dataset (but sampled from the same distribution) or it may overlap as a subset, if the attacker has obtained some data points from the target. Note that a similar assumption is essential to other well-studied inference attacks such as the Membership Inference Attack (MIA) [3]. In addition, the clients participating in a FL architecture can also receive information about the process from the server, such as the aggregated metrics calculated in each round, in order to evaluate the convergence of the model.

Let us consider a bit vector s with n bits, with each bit i being set to 1 if user i participated in some round r in the training of the model W, and 0 otherwise. We can define the CIA as follows:

Definition 8 (*Client Inference Attack*)**.** Given a global aggregated model W and the shadow dataset D_x of client x, the Client Inference Attack can be defined as:

$$C(D_x, W) := \mathbb{P}(s_x = 1 | D_x, W)$$

Note that in this work we assume that the number of clients is fixed in all rounds. This makes sense in the case of cross-silo configurations. Thus, the client that is going to perform a CIA, wants to know if the target has participated or not based on the shadow dataset available.

Then, a central assumption is that the attacker has access to a shadow dataset D_x that approximated the data distribution of client x. However, in practical cross-silo medical settings, obtained such representative dataset is challenging, and the attacker may rely on: (1 public datasets that partially reflect the target distribution; (2) historical data from similar sources; (3) small subsets of the target data obtained through other means.

Even when the shadow dataset is not representative, partial signals can still provide useful information for participation inference with reduce confidence. This highlights that the threat model is strongest under strong knowledge assumptions, but remains relevant in realistic limited-knowledge scenarios.

5.2. Attack variants and impact in other attacks

Our analysis focuses on passive CIA, in which the attacker only observes the aggregated model. However, more sophisticated variant are possible and can be considered for future work:

- *Active CIA:* in this case we can suppose that there is also a malicious client that sends specifically crafted updates (e.g. scaled or perturbed gradients) to amplify differences attributable to the target client.
- *Adaptive CIA:* an attacker could combine information from multiple rounds, observing how the global model evolves to improve confidence in client participation.

It is important to note that the CIA outcome can significantly amplify other attacks, such as MIA or SIA. Specifically, knowing which client participated in the training reduces the uncertainty about whose data influenced the model, allowing an attacker to better target a MIA. In addition, confirming the participation of given client helps to identify which client contributed to certain model behaviors. For example, let us assume that we infer with a CIA that the hospital H participated in the training. The, we also observe that the model has a low error rate on a specific piece of data similar to what H usually trains on. This makes it much easier to conclude that such data probably comes from H (successful SIA).

5.3. Defense with differential privacy and the metric-privacy notion

First of all, it is clear that global-DP can be a measure that helps to prevent this type of attack. This is because in CIA the malicious client analyzes the performance of the aggregated model. Adding Differential Privacy to the aggregated model limits the amount of information disclosed by mathematical definition. Thus, adding noise to the aggregated model seeks to mask the individual contributions of the different participating clients. However, it is necessary to calibrate the noise added to maintain a good balance between protection against CIA and the performance of the model.

In this sense, we propose to use the notion of metric-privacy. As we are about to explore, adding noise based on a metric allows to make a finer balance between privacy and model performance, since in case of similar contributions from the clients it will be necessary to add less noise. Even in scenarios with non-i.i.d. clients, metric-privacy can contribute to the proper calibration of privacy by adding only the "necessary amount" of noise. Using such notion of metric-privacy in

this case makes sense as we assume that the participating clients are static throughout the rounds.

We will show in Section 7 an experimental analysis of this approach in an example of medical imaging. The hypothesis which we will be testing is that metric-privacy can contribute to better calibrate the noise compared to global-DP. Such noise calibration can reduce the defense against CIA, but at the same time improve the performance of the same allowing to achieve a better balance between utility and privacy.

6. Proposed methodology

We are now ready to describe our proposed approach. Let us recall our trust model: we assume that there is an honest central server in a cross silo setting with all the initial clients participating during the whole training. However, we suppose that some of the clients involved may act as attackers seeking to extract information from the models and therefore from the data of other clients participating in the network. Specifically, our goal is to protect against Client Inference Attacks.

To do this, we first apply differential privacy with fixed clipping norm to the aggregation process from the server side (global-DP). By doing so we also prevent information leakage from external attackers, when publishing the aggregated model in a model repository. Specifically, in our approach we use the Gaussian Mechanism setting a noise multiplier associated to the privacy level we want to define.

Concerning the selection of the parameters for applying DP, we have to note that the clipping norm is a hyperparameter that must be optimized. More specifically, for the first round we calculate the following value for the clipping norm, with w_i the weights for client i, $\forall i \in \{1, \dots, n\}$:

$$\tilde{c} = \max_{\substack{i,j \in \{1,\dots,n\} \\ i \neq j}} \|w_i - w_j\|_2.$$

We started optimizing the clipping norm using $\tilde{c}$, computed from the initial local models across different data splits and random seeds. After conducting various tests, we finally chose the value $C = 5$ for the experimental simulations, since this value consistently fell within the range of the calculated values.

Next, we propose a novel noise calibration strategy inspired by the notion of metric-privacy to mitigate such attacks. Recall that the metric-privacy notion suggests that privacy protection should be calibrated according to the distance between inputs, and we exploit this intuition by adjusting the noise based on the distance between the local models of the clients. However, the server does not have access to the raw data of the clients. Hence the question arises: *how can the server calculate the distance between the clients, without knowing their data?*

To overcome the hurdle, we propose a metric that depends on the distance between the model updates rather than the actual clients' data. In order to do so, the server calculates the maximum distance for each pair of clients by analyzing the local weights received from each one. In this regard, different metrics have been analyzed and experimental tests have been carried out on a validation set. Finally, in this specific use case, the following metric was chosen, also in line with studies in the field of FL that use a similar metrics (e.g. [34]) to analyze the divergence between clients.

Let $w_i^{(r)}$ be the local weights of the model trained for client i in round r, with $w_i^{(r)}(\ell)$ being the weights of the layer ℓ of the model in round r. Moreover let $\mathcal{L}$ be the set of layers of the trained model and n_c be the number of clients. We define the distance in round r as follows:

$$d^{(r)} = \max_{\substack{i,j \in \{1,\dots,n_c\} \\ i \neq j}} \left(\frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} \|w_i^{(r)}(\ell) - w_j^{(r)}(\ell)\|_F \right).$$

Note that $\|\cdot\|_F$ represents the Frobenius norm.

The hypothesis which we will be testing is that using metric-privacy with $d^{(r)}$ will help to fine-tune the amount of noise needed in each round, ultimately allowing better convergence.

We explained above that we will compare metric-privacy with global-DP applied using the Gaussian mechanism. For a fair comparison we will employ the Gaussian mechanism in metric-privacy as well setting $\bar{\epsilon} = \epsilon \cdot d^{(r)}$ for each round r.

The classic implementation of DP with fixed clipping in FL involves introducing the clipping norm C, the noise multiplier (n_ϵ) and the number of sampled clients n_c. Note that the noise multiplier has the opposite interpretation to the privacy budget, since the lower the value, the higher the utility and the lower the privacy, as less noise is injected.

In this line, we have calculated the metric $d^{(r)}$ dynamically in each round r and use it by dividing the standard deviation of the Gaussian noise as follows:

$$\mathcal{N}(0, \sigma) = \mathcal{N}\left(0, \frac{n_e \cdot C}{n_c \cdot d^{(r)}}\right).$$

Then, the objective is to analyze the impact of adding metric-privacy to the aggregated model form the server side, applying metric-privacy-inspired noise calibration in each FL round. In addition, we want to evaluate the impact of this privacy enhancing technology compared to classic global-DP in terms of a client inference attack. Note that if $d^{(r)} > 1$, the metric-aware calibration injects more noise than standard global-DP with the same noise multiplier, consistent with the metric-privacy intuition that greater divergence between participants requires stronger perturbation. When $d(r) < 1$, less noise is injected, improving utility. In practice, $d(r)$ tends to decrease as training progresses and local models converge, explaining the observed utility gains over rounds.

Finally, in Algorithm 1 we present the pseudocode for including metric-privacy-aware noise from the server side.

Algorithm 1 Metric-Aware Aggregation with Differential Privacy.

INPUT: $\mathcal{R} = \{(c_i, \theta_i)\}_{i=1}^N$; client updates ($\theta_i$) and number of data ($c_i$),
INPUT: $\theta^{(r)}$; global model in round r,
INPUT: clipping norm C: clipping norm,
INPUT: n_ϵ; noise multiplier,
INPUT: $\text{Agg}(\cdot)$; aggregation strategy.
function AGGREGATE_MDP($\mathcal{R}$, $\theta^{(r)}$, C, σ, Agg)
  $d \leftarrow \text{DISTANCEMETRIC}(\mathcal{R})$
  for each client update $\theta_i \in \mathcal{R}$ **do**
    $\Delta_i \leftarrow \theta_i - \theta_t$
    $\Delta_i \leftarrow \text{CLIP}(\Delta_i, C)$
    $\theta_i \leftarrow \theta_t + \Delta_i$
  end for
  $\bar{\theta} \leftarrow \text{AGG}(\{c_i, \theta_i\}_{i=1}^N)$
  $N \leftarrow |\mathcal{R}|$
  $n_{\epsilon,d} \leftarrow n_\epsilon / d$
  $\sigma \leftarrow (n_{\epsilon,d} \cdot C)/N$
  $\theta^{(r+1)} \leftarrow \bar{\theta} + \mathcal{N}(0, \sigma)$
  return $\theta^{(r+1)}$
end function

Note that in the previous algorithm we draw Gaussian noise with mean 0 and standard deviation σ. We use this notation instead of the variance-based one ($\mathcal{N}(0, \sigma^2)$) for better readability. The standard deviation is proportional to the clipping norm and inversely proportional to the number of sampled clients. Thus, the distance metric is used for noise calibration and does not constitute a formal metric-privacy guarantee.

6.1. Privacy accounting and theoretical guarantees

With the aim of analyzing the theoretical privacy guarantees offered by the two approaches compared in this work (global-DP and metric-privacy), in this section we review formal bound that clarify the protection ensured by each mechanism under the federated learning architecture considered in the experimental setup.

- **Global-DP**: in this case, we have implemented differentially private versions of different FL aggregation strategies following the

work done in [35]. Regarding privacy accounting, in such work the authors used the *Moments Accountant* method [11], which allows to calculate an upper bound for the privacy budget taking into account the number of FL rounds. Specifically, it allows the contributions from several rounds of training to be combined, resulting in a more accurate overall privacy level than the standard composition.

In our approach, for calculating such bound the sensitivity is bounded by the clipping norm fixed by the user and the noise is added using the Gaussian Mechanism with a fixed noise multiplier.

It is important to note that the privacy bound remains valid for all aggregation functions used (*FedAvg*, *FedAvgM*, *FedMedian*, *FedProx*, *FedOpt* and *FedYogi*), because noise is added to the global aggregated model, regardless of how local updates are combined.

- **Metric-privacy**: the compositionality of metric-privacy have been analyzed in recent studies [36]. However, our approach does not implement an specific mechanism for metric-privacy, instead, we exploit the notion of metric-privacy for noise calibration, and we implement it by simply modifying the noise multiplier for the Gaussian Mechanism used in the global-DP setup. Specifically, as in the case of global-DP, in this second approach we add Gaussian noise to the global aggregated model after each round. The difference is that the noise multiplier becomes distance-dependent, i.e. is scaled according to the distance between the local weights of the clients. This means that the privacy level changes dynamically in each round depending on the similarity of the local models, which allow us to calibrate the noise added. Then, to obtain the privacy bound after all rounds, the same composition mechanisms can be used as in the case of global-DP, simply taking into account that now the value of the noise multiplier is given by $\overline{\epsilon} = \epsilon \cdot d^{(r)}$ for each round r.

  It is important to note that this means that in this approach we are using the notion of metric-privacy as a dynamic noise variant of global-DP, rather than applying a formal metric-privacy mechanism. Therefore, the privacy bounds that can be calculated using the moments account method capture the global-DP guarantees induced by the Gaussian mechanism and should not be interpreted as bounds in the sense of metric-privacy.

In this work, instead of reporting formal (ϵ, δ) values, we adopt an empirical approach to privacy evaluation. Specifically, we assess the protection offered by each mechanism directly through the CIA experiments defined in Section 5, comparing global-DP and metric-privacy aware calibration under identical conditions in terms of noise multiplier and clipping, and evaluating their resistance to the same adversarial attack. This attack-based evaluation is intentional and even more informative in practical setups than a formal bound: it measures what an adversary can actually infer from the aggregated model rather than providing a worst-case theoretical guarantee. Practitioners requiring formal DP certification should increase the noise multiplier and apply standard Moments Accountant composition. In order to guide the selection of the appropriate noise multiplier, Appendix B provides results across a range of noise multipliers.

7. Experimental results

7.1. Data under study

In this work, for the experimental simulation of the proposed methodology we have taken the data from the Alzheimer dataset available in [37]. Our goal is to carry out this comparative study with this open dataset that can be meaningful in a practical scenario. The dataset contains brain MRI images and is especially useful for the testing and investigation of applied healthcare AI models. Specifically, the goal with this dataset is to predict, based on the MRI images, the level of Alzheimer's disease of the patient. For this purpose there are 4 categories: mild demented (0), moderate demented (1), non-demented (2) and very mild demented (3). The initial train dataset is composed of 5120 images and in the test dataset 1280 images are available. The distribution of the classes in each of these two datasets is shown in Table 1.

Table 1
Original train and test set distribution.

	Total	*Class 0*	*Class 1*	*Class 2*	*Class 3*
Train	5120	724	49	2566	1781
Test	1280	172	15	634	459

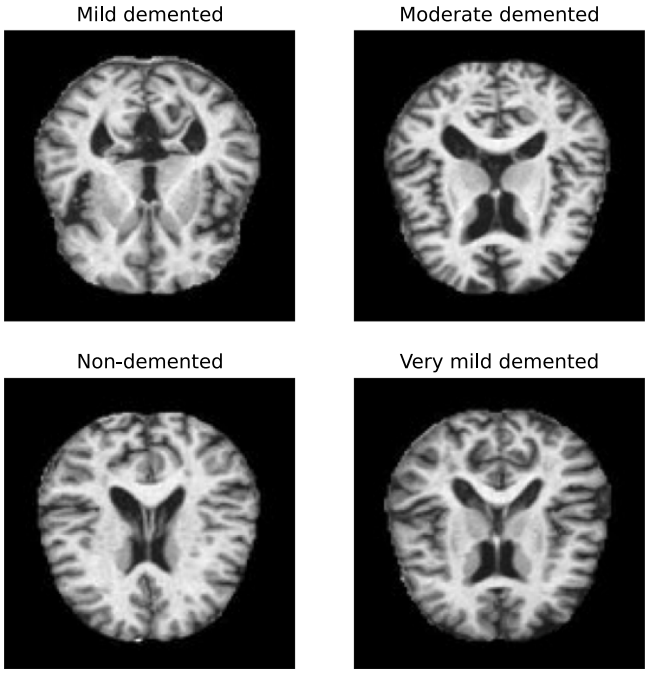


Fig. 1. Example of the original images from the train set by classes.

In addition, Fig. 1 shows an example of each class of images available in the train dataset.

To perform the FL scheme we will simulate 4 clients from the train dataset, while leaving the initial test set for analyzing the performance of the final models and for validation. Specifically, the 50% of the test set will be taken as validation from the server side to train some initial parameters for those aggregation functions that require them. The remaining 50% will be considered the test set for analyzing the resulting models.

In order to analyze the impact of adding global-DP and metric-privacy according to the aggregation function used in FL, we propose two experimental settings for the same initial centralized dataset: homogeneous clients and non-i.i.d clients. Once the train data has been distributed to the different clients, in both cases we leave 20% for test in each client and 80% for train. This split is done in a stratified way for maintaining the statistical significance associated with the representation of each class.

7.1.1. Homogeneous clients

In this first approach we perform a division of the initial training set into four disjunct clients such that the distribution of the number of images in each class is homogeneous. For this, we perform a stratified division of the training set by assigning to each client the same number of data in a balanced way (almost the same number of each class for each client split). Thus, Table 2 shows for each client the distribution of the four classes present in the original database.

Table 2
Homogeneous clients distribution.

	Total	*Class 0*	*Class 1*	*Class 2*	*Class 3*
Client 1	1280	181	12	641	446
Client 2	1280	181	12	642	445
Client 3	1280	181	12	642	445
Client 4	1280	181	13	641	445

Table 3
Non-i.i.d clients distribution.

	Total	*Class 0*	*Class 1*	*Class 2*	*Class 3*
Client 1	1792	280	16	881	615
Client 2	768	107	13	368	280
Client 3	2048	257	17	1054	720
Client 4	512	80	3	263	166

7.1.2. Non-i.i.d clients

In this second setting, the aim is to simulate a scenario where we work with clients who possess different datasets. For this purpose, the initial training set has been divided among four disjoint clients, with each assigned a different number of data points and an unequal distribution of classes represented in the dataset. Specifically, Table 3 shows the number of data for each client and the number of data belonging to each predicting class in each of them.

7.2. Model analyzed

Different models and hyperparameters have been analyzed during the experimental procedure detailed in this work in order to select the one implemented. Finally, the implemented model for performing the different tests (which was fine-tuned taking into account a validation split of the dataset) is a convolutional neural network (CNN) composed by the following layers:

- *Conv2D layer*. 32 neurons. Activation *relu*. Input shape: *(128, 128, 1)*.
- *MaxPooling2D((2,2))*.
- *Conv2D layer*. 64 neurons. Activation *relu*.
- *MaxPooling2D((2,2))*.
- *Conv2D layer*. 128 neurons. Activation *relu*.
- *Flatten layer*.
- *Dense layer*. 64 neurons. Activation *relu*.
- *Dropout layer*. Rate 0.1.
- *Dense layer*. 32 neurons. Activation *relu*.
- *Dropout layer*. Rate 0.1.
- *Output: Dense layer*. 4 neurons. Activation *softmax*.

The model is compiled using the *Adam* optimizer, the *sparse categorical cross-entropy* as loss function and the *accuracy* as metric. The batch size used for the training in 32 and we train it during 5 epochs and 20 rounds of the Federated Learning architecture.

Concerning the aggregation strategies used, Table 4 presents the specific parameters selected for each of them, taking into account the recommendations given in the implementation documentation and after validating different values.

In addition, is important to note that some of the aggregation functions used (*FedAvgM*, *FedOpt*, *FedYogi*) require an initial model. This initial model was obtained by training for 20 epochs with batch size 32 on the validation set, which was taken as 50% of the test split of the initial database, in a stratified way.

Note that in order to follow a real use case, we have performed a hyperparameter tuning using a validation dataset. Then, we have run the model to simulate a real use case with a fixed seed for the libraries with stochastic behavior, in order to make a fairer comparison between the three approaches (non-DP, global-DP and metric-privacy).

Table 4
Values used in the experimental setup for the specific parameters of each aggregation strategy analyzed.

Strategy	*Specific parameters*
FedAvg	*None*
FedAvgM	$\beta = 0.5,\ \mu = 0.1$
FedMedian	*None*
FedProx	$\mu = 0.5$
FedOpt	$\beta_1 = 0,\ \beta_2 = 0,\ \tau = 1e-9$
FedYogi	$\beta_1 = 0.9,\ \beta_2 = 0.99,\ \tau = 1e-3$

Table 5
Comparison on homogeneous clients of the mean and the standard deviation of the aggregated accuracy in the last five rounds of the FL scheme. Aggregation performed without using DP and with global-DP and metric-privacy with noise multiplier of 0.01.

	Mean accuracy ± std (last 5 rounds)		
Strategy	*Vanilla FL*	*Global-DP*	*Metric-privacy*
FedAvg	0.929 ± 0.008	0.881 ± 0.013	0.905 ± 0.007
FedAvgM	0.878 ± 0.013	0.759 ± 0.024	0.799 ± 0.022
FedMedian	0.920 ± 0.005	0.885 ± 0.013	0.905 ± 0.018
FedProx	0.931 ± 0.006	0.879 ± 0.008	0.907 ± 0.022
FedOpt	0.956 ± 0.004	0.918 ± 0.014	0.948 ± 0.005
FedYogi	0.950 ± 0.006	0.906 ± 0.008	0.911 ± 0.012

However, in order to analyze the robustness of the results, we have tested the performance by running the model 5 times with each scenario and each aggregation function (e.g. for the case of i.i.d. clients we have conducted 90 experiments). These results are shown for the test set as a function of the accuracy in Table A.14 in Appendix A. In view of the standard deviation obtained in the 5 runs of the model for such scenario, which is statistically significant, the results of one run of the model are shown to better fit a real scenario (see for instance [22] or [38]).

7.3. Results

In this section we show the results obtained with the three approaches: vanilla FL (baseline, no DP), global-DP and metric-privacy from the server side with six different aggregation strategies. In addition, we show the aggregated accuracy obtained for the clients test set and the accuracy, precision and F1-score obtained with each strategy for the initial test set. In addition, in Appendix C we review the AUC and the ROC curves obtained.

Finally, we compare the three approaches regarding the prevention of the Client Inference Attack.

7.3.1. Homogeneous clients

The aggregated results in the last five rounds of the FL architecture (mean and standard deviation), using the CNN model presented in the previous section are shown in Table 5, comparing the mean accuracy and standard deviation in the last 5 rounds across the six aggregation strategies.

Table 5 shows that with metric-privacy we always achieve better results compared to those with global-DP, as well as a better convergence over the course of the rounds (see Appendix D). The evolution of the accuracy of the aggregated model in clients test set (aggregated) is shown for each round of the FL schema in Fig. D.4 from Appendix D for the cases of the global aggregated model without DP, with global-DP and with metric-privacy.

Both in the case of global-DP and metric-privacy we introduce the same value for the noise multiplier (0.01), however, note that with metric-privacy we are dividing the noise multiplier by the distance $d^{(r)}$ in each round r. If $d^{(r)} > 1$ we will be adding more noise (we could expect the model to get worse in its performance), while if $d^{(r)} < 1$ we would be adding less noise, which will result in greater similarity

Table 6
Accuracy, F1-score and precision obtained in the test set with each approach. Homogeneous clients.

	Vanilla FL			*Global-DP*			*Metric-privacy*		
Strategy	*Accuracy*	*F1-Score*	*Precision*	*Accuracy*	*F1-Score*	*Precision*	*Accuracy*	*F1-Score*	*Precision*
FedAvg	0.909	0.903	0.950	0.884	0.862	0.918	0.894	0.843	0.943
FedAvgM	0.884	0.768	0.932	0.762	0.554	0.607	0.823	0.608	0.631
FedMedian	0.932	0.899	0.946	0.875	0.827	0.913	0.895	0.843	0.943
FedProx	0.909	0.848	0.873	0.877	0.853	0.914	0.903	0.877	0.944
FedOpt	0.950	0.936	0.970	0.918	0.877	0.911	0.930	0.917	0.950
FedYogi	0.933	0.914	0.957	0.908	0.841	0.884	0.917	0.888	0.941

Table 7
Comparison on non-i.i.d clients of the mean and the standard deviation of the aggregated accuracy in the last five rounds of the FL scheme. Aggregation performed without using DP and with global-DP and metric-privacy with noise multiplier of 0.01.

	Mean accuracy ± std (last 5 rounds)		
Strategy	*Vanilla FL*	*Global-DP*	*Metric-privacy*
FedAvg	0.933 ± 0.002	0.878 ± 0.012	0.927 ± 0.008
FedAvgM	0.909 ± 0.007	0.790 ± 0.018	0.840 ± 0.016
FedMedian	0.926 ± 0.009	0.838 ± 0.013	0.905 ± 0.007
FedProx	0.937 ± 0.014	0.878 ± 0.016	0.943 ± 0.003
FedOpt	0.964 ± 0.007	0.925 ± 0.009	0.945 ± 0.006
FedYogi	0.945 ± 0.005	0.912 ± 0.009	0.919 ± 0.010

with the non-DP model. Then, metric-privacy help us to find a proper value for the standard deviation of the Gaussian noise, while we are still protecting privacy and client inference attacks, as will be shown in Section 7.4.

In Table 6 we show the results obtained with each approach and each aggregation function for the global model obtained after 20 rounds for the test set in terms of the accuracy, F1-score and precision. In view of Table 6 we can highlight that the performance with metric-privacy is better than the one with global-DP with all the six strategies analyzed in terms of the accuracy and the precision, and in 5 out of 6 in terms of the F1-score. In addition, note that for *FedProx* strategy, the F1-score and the precision is better with metric-privacy that without DP, this may be due to a better generalization capability in terms of the less-represented classes and a better mitigation of potential overfitting.

In addition, for completeness of the analysis, Appendix C shows the value for the AUC and the ROC curve obtained in each case with each strategy and each of the three proposed approaches (Fig. C.2).

7.3.2. Non-i.i.d clients

In this second setup we conduct the experiments on the non-i.i.d. clients case. It should be noted that the distribution of each class in each data owner is statistically significant with the initial distribution, as it would not make sense for a hospital to have the minority class as the majority class, which would not fit the usual distribution of the pathology.

The non-i.i.d. nature is determined by the amount of data in each data owner, with one being more representative than the others as a function of the number of samples, as shown in Table 3. Then, we have trained the FL architecture with these four clients and the six aggregation strategies explained above. Note that some of these strategies will be more appropriate than others to deal with the non-i.i.d nature of the simulated data owners.

Table 7 shows the aggregated results obtained for each client's test set in terms of the mean accuracy in the last 5 rounds of the FL scheme as well as the standard deviation in these rounds.

From Table 7 we can see that the results with metric-privacy are again better than those obtained with classic global-DP in the 6 cases analyzed depending on the aggregation function. Moreover, as we have already seen in the case of homogeneous clients, we can observe that in the case of *FedProx* strategy, the mean accuracy with metric-privacy is even better than without adding DP. Although again this may be due to a reduction of overfitting for better adjustment of the underrepresented class, we can highlight by observing the standard deviation, that in the best case without DP we would have an accuracy of 0.951, while in the best case with metric-privacy it would be 0.946. Therefore, this improvement is due to fluctuations in the overall aggregated accuracy in the last 5 rounds, as can be seen in Fig. D.5 of Appendix D.

In addition, Table 8 shows the results obtained with the six aggregation strategies and the three paradigms (no DP or vanilla FL, global-DP and metric-privacy) on the test dataset. Specifically, this table shows the model performance in this test set as a function of the accuracy, F1-score and precision obtained with the aggregated model obtained after 20 rounds of FL training. In this case, again the trend is maintained: metric-privacy improves the global-DP results, in general. Metric-privacy is better in all cases with respect to accuracy and precision and in 5 out of 6 with respect to F1-score (only slightly worse with *FedAvg*). Furthermore, with *FedProx* we see the anomaly already commented (i.e. the precision and F1-score obtained with metric-privacy is better than without DP).

Finally, for further analysis, Fig. D.5 in Appendix D shows the evolution of the aggregated accuracy in the training set of the different clients in this scenario of non-i.i.d. clients with the three proposed privacy approaches. In addition Fig. C.3 in Appendix C shows the value for the AUC and the ROC curve obtained in each case for the test set.

7.4. Impact of a client inference attack

In the last two sections we have seen that by including metric-privacy in the FL architecture, from the server side, we can improve the accuracy of the model, as we are better tuning the amount of noise added. However, we are interested in analyzing whether adding metric-privacy can contribute to reduce the risk of a client inference attack (following the definition proposed in Section 5) in a similar way as with global-DP. For the experimental setup, let us consider two kind of attacks:

7.4.1. First round single shot CIA

As already stated, our assumption is that a client can act as an attacker to infer whether certain client is participating in the training, and thus learn information about it. For example, in this medical imaging case, an attacker might want to infer whether a certain hospital is participating in the training, in order to learn about the hospital's data distribution and thus sensitive information about the hospital's patients and their pathologies.

In this sense, we can intuitively expect that the attack will be more effective in the first round. Although an attacker would use additional information to try to extract additional insights throughout the different rounds, it will be in the first round (when the model has not yet converged) when it will be possible to infer more knowledge. As a reminder, we assume that the clients participating in the federated training are the same throughout all rounds.

Thus, we are going to simulate a new scenario. In this case, for the sake of simplicity, we will take 3 clients, one of which will be the attacker who wants to know whether or not another client is participating in the training. In addition, this client on which the attack will be performed (target) will have an anomalous data distribution compared to the distribution on the other two clients.

Table 8
Accuracy, F1-score and precision obtained in the test set with each approach. Non-i.i.d clients.

	Vanilla FL			*Global-DP*			*Metric-privacy*		
Strategy	*Accuracy*	*F1-Score*	*Precision*	*Accuracy*	*F1-Score*	*Precision*	*Accuracy*	*F1-Score*	*Precision*
FedAvg	0.919	0.930	0.956	0.867	0.787	0.886	0.925	0.914	0.931
FedAvgM	0.909	0.862	0.944	0.784	0.573	0.594	0.837	0.621	0.640
FedMedian	0.933	0.917	0.961	0.859	0.626	0.641	0.917	0.926	0.940
FedProx	0.920	0.910	0.950	0.886	0.871	0.904	0.903	0.890	0.909
FedOpt	0.936	0.927	0.957	0.922	0.913	0.960	0.916	0.885	0.889
FedYogi	0.927	0.870	0.950	0.900	0.845	0.836	0.895	0.847	0.865

Table 9
Clients distribution for simulating a client inference attack.

	Total	*Class 0*	*Class 1*	*Class 2*	*Class 3*
Client 1	1747	120	9	1122	496
Client 2	1591	180	11	894	506
Client 3	1882	524	29	550	779

Table 10
Loss obtained for the aggregated test set and for the shadow dataset of client 3 with *FedAvg* strategy.

Client	*Vanilla FL*	*Global-DP*	*Metric-privacy*
Aggregated	1.032	3.603	1.135
Target	1.182	4.848	1.506
Difference (%)	12.719	25.679	24.631

In Table 9 the distribution of the clients simulated for this experiment is shown, with client 1 being the attacker and client 3 being the target. Note that in this case, in order to get a better fit to the training data of each client, we are going to train the model for 20 epochs. Then, at the end of the first round, client 1 (attacker) knows its local model as well as the global aggregated one and some aggregated metrics sent from the server.

For our approach concerning CIA from client 1 to client 3 as target, we assume that the former has a shadow dataset from client 3, which will be a random split of 10% of client 3 training data. In addition, as the attacker participates in the training he/she will receive from the server the aggregated metrics (accuracy and loss) obtained in each round. Then, we can analyze the loss obtained with the aggregated model for the shadow train dataset of client 3. In Table 10 we show the results for the cases where this client participates in the training performing the aggregation with the *FedAvg* approach (as it is the most used one) and the three privacy approaches.

Note that in Table 10 we take into account the loss instead of other metrics such as the accuracy, as for this multi-class classification problem it is more representative, as we are dealing with clients with quite non-i.i.d classes distributions. In view of this example, we can note that for the *FedAvg* strategy we get quite similar values for the relative difference between the aggregated result in each client's test sets and the loss for the target client shadow dataset with global-DP and with metric-privacy. As we could expect, we get a slightly greater relative difference with global-DP than with metric-privacy, but in both cases we are protecting from the CIA attacks in a similar way (specifically more than in the case without DP, vanilla FL, with a 12.719% of difference). In addition, the difference between global-DP and metric-privacy approaches is lower than 1.1% .

In Table 11 we also show the results obtained for this CIA simulation using the *FedOpt* strategy, as this is the one that provides better results in general for two the previous scenarios analyzed. We can observe that in this case the convergence of the model is much better due to the aggregation strategy used, since it has an auxiliary initial model trained with the validation dataset. Also, in this case it is worth noting that we obtain a greater difference with metric-privacy than with global-DP, thus providing a greater resistance to the CIA in this particular example.

Table 11
Loss obtained for the aggregated test set and for the shadow dataset of client 3 with *FedYogi* strategy.

Client	*Vanilla FL*	*Global-DP*	*Metric-privacy*
Aggregated	0.595	0.743	0.756
Target	0.599	0.807	0.853
Difference (%)	0.598	7.936	11.320

Table 12
Comparison of the loss obtained for the test set in the first round. Aggregation performed without using DP, with global-DP with noise multiplier of 0.01 and with metric-privacy with 0.01 as noise multiplier. CIA scenario.

	Test loss		
Strategy	*Vanilla FL*	*Global-DP*	*Metric-privacy*
FedAvg	1.010	3.514	1.106
FedAvgM	0.993	1.048	1.041
FedMedian	1.046	2.071	1.387
FedProx	1.010	1.744	1.149
FedOpt	0.645	0.845	0.823
FedYogi	0.870	1.036	0.902

At the same time, as is shown in Table 12, the results for the loss in the test set are also better with metric-privacy than with global-DP.

Then, as analyzed in the previous sections, in this use case we reach a better balance between protection and model performance with metric-privacy, as in all cases analyzed in Sections 7.3.1 and 7.3.2 we obtain a significantly better model performance than with global-DP. In addition, in view of the above we get a similar protection from CIA attacks with *FedAvg* (slightly worse than with global-DP), and better for the case of *FedOpt* for this new scenario. For completeness of the analysis, Appendix E shows the results obtained for the other four aggregation functions analyzed.

Finally, let us see the results for the loss obtained in the test set with these three clients trained after the first round (see Table 12):

In view of Table 12, we can see that in all cases the loss obtained with metric-privacy is lower than with global-DP, while the protection against CIA is similar, and in some cases even higher (e.g. with *FedProx* and *FedAvgM*, see Appendix E). We have to note that the model is not yet converging, as only one round of the scheme has been carried out. However, it is important to analyze the results after this round because it is the one from which an attacker could extract more information. Again in this simple approach the best results are obtained with *FedOpt* as aggregation strategy.

We emphasize that the 10% shadow dataset in this single-round experiment represents a strong adversarial assumption, chosen to provide an upper bound on attack effectiveness. In practice, attacker knowledge would typically be more limited, which would further reduce CIA success rates. In line with threat models in related inference attack literature, in which "noisy real data" are considered [3], we will further explore in the following such approach when considering a multi-round CIA.

7.4.2. Multi-round CIA

In this second experimental setup for evaluating the resistance to CIA, we assume that the attacker observes the aggregated model during

Table 13
AUC and 95% bootstrap confidence interval obtained for the scores of the cases IN and OUT. Aggregated accuracy and F1-score obtained in the test sets (IN and OUT), mean and standard deviation in the last 5 rounds. Aggregation performed with *FedAvg*.

	Results				
Approach	*AUC (CI 95%)*	*Acc (IN)*	*Acc (OUT)*	*F1-Score (IN)*	*F1-Score (OUT)*
Vanilla FL	0.890 (0.767, 0.977)	0.948 ± 0.007	0.894 ± 0.007	0.930 ± 0.006	0.729 ± 0.024
Global-DP	0.397 (0.224, 0.585)	0.637 ± 0.009	0.680 ± 0.026	0.426 ± 0.018	0.400 ± 0.008
Metric-privacy	0.493 (0.296, 0.689)	0.794 ± 0.023	0.746 ± 0.029	0.695 ± 0.035	0.480 ± 0.022

multiple communication rounds of the FL training. The underlying idea is that the influence of a participating client becomes more noticeable throughout the training, compared to a single-round observation.

As in the previous experiment, let us assume that a CIA is performed from client 1 to client 3 as target, considering that the former has a shadow dataset from client 3. However, in this case, in order to simulate a more feasible scenario, we will build this shadow dataset considering a random split containing 10% of client 3 training data perturbed adding gaussian noise with a deviation of 20% of the maximum pixel value. We consider 5 local epochs per client and 20 communication rounds, as in the first experimental setups.

We analyze two scenarios: (i) *IN*, where client 3 participates in the training ; (ii) *OUT*, client 3 does not participates in the training. In contrast to the single-round CIA, here the attacker exploits information across rounds. Then, at each round r, the attacker evaluated the received global model $W^{(r)}$ on the shadow dataset D_x and computed a real-valued attack score: $s_r = -\ell(W^{(r)}, D_x)$, where $\ell(\cdot)$ denotes the loss function.

Each communication round yields a scalar score, and this procedure is repeated for all the rounds under both the IN and OUT settings, producing two sets of score: $\{s_i^{IN}\}_{i=1}^{20}$ and $\{s_i^{OUT}\}_{i=1}^{20}$.

To quantify the success of the attack we calculate the Area Under the ROC Curve (AUC) using the scores calculated across the 20 rounds for both IN and OUT. The AUC measures the probability that a randomly chosen score from the IN case is ranked higher than a randomly chosen score from OUT, thus providing a threshold-independent metric.

In Table 13 we report the AUC obtained when comparing the scores for the cases of IN and OUT during 20 rounds in the three scenarios, vanilla FL, GDP and metric-privacy notion and with *FedAvg* as aggregation strategy. We additionally report the 95% bootstrap confidence interval for the AUC, as well as the mean and standard deviation of the aggregated accuracy and F1-score on the clients' test sets, computed over the last 5 rounds.

In view of Table 13, vanilla FL yields and AUC of 0.890 (CI 95%: 0.767–0.977), indicating that without any privacy mechanism applied, an attacker can certainly distinguish client participation across rounds. Both privacy mechanisms allow to substantially reduce this risk: global-DP yields an AUC of 0.397 (CI 95%: 0.224–0.585) and metric-privacy aware calibration yields an AUC of 0.493 (CI 95%: 0.296–0.689). In both cases where privacy constraints are added, the confidence intervals for both methods overlap with the baseline for the random classifier (AUC = 0.5), confirming that, under this assumptions, an attacker cannot effectively infer the client's participation. The AUC reduction relative to vanilla FL is 0.493 points for global-DP and 0.397 points for metric-privacy, representing a substantial and comparable level of protection against the multi-round CIA.

Regard model utility, the different between the two privacy aware approaches is notable. In the IN scenario, metric-privacy inspired calibration achieves an accuracy of 0.794 ± 0.023 and a F1-score of 0.695 ± 0.035, while this values are drastically reduced in the global-DP approach, with an accuracy of 0.637 ± 0.009 (more than 15% less on average), and a F1-score of 0.426 ± 0.018. A similar pattern is observed in the OUT scenario, where metric-privacy inspired calibration achieves an accuracy of 0.746 ± 0.029 and a F1-score of 0.480 ± 0.022, compared to 0.680 ± 0.026 and 0.400 ± 0.008 for global-DP respective. In both scenarios, the vanilla FL approach remains the most accurate one, as would be expected in the absence of noise injection.

Taken together, these results demonstrate that metric-privacy aware noise calibration achieves a CIA protection level comparable to that of global-DP while offering substantially better model utility across all metrics and in both scenarios.

8. Conclusions and future work

In this paper we have analyzed the impact of including the notion of metric-privacy in an FL architecture for noise calibration. Specifically, we have focused on a cross-silo scenario with a trusted server and a fixed set of semi-honest clients, which may also act as adversaries attempting to extract information from the system. Furthermore, we introduce a new type of attack, the client inference attack (CIA), in which a client in the network aims to infer whether a specific participant is involved in the FL process. As we argue, CIA can serve as a basis for more harmful attacks such as reconstruction, source or membership inference. It is important to note that this is the first work that we are aware of that employs metric-privacy-inspired noise calibration in a federated architecture to improve the privacy/utility trade-off. At the same time it is the first one that defines the CIA and presents an example of application and prevention with global-DP and metric-privacy.

To mitigate this threat while improving privacy-utility trade-off, we propose to use the notion of metric-privacy as a design principle to calibrate the injected noise within a global-DP framework, dynamically adjusting the noise according to the distance between local models. Such distance has been calculated as the maximum distance between the models of each pair of clients, and in this work we propose to use the maximum pairwise divergence between local models. This provides round-wise privacy guarantees, tuning the amount of noise added (allowing for better model performance) and enabling lower noise addition when clients remain close in the model space while still offering protection against CIA.

We evaluate this approach in a medical imaging use case under three settings and using 6 FL aggregation functions: homogeneous clients, non-i.i.d clients and a third case with a client with unbalanced data (target), in order to test the resistance to CIA following two approaches (first round and multi-round CIA). Our results show that the metric-privacy-based approach consistently provides the higher model performance than global-DP in all the cases analyzed, while maintaining a comparable level of protection against CIA. In the third scenario, for the case of first round single shot CIA, the metric-privacy inspired even outperforms global-DP in two of the six aggregation methods concerning CIA resistance.

8.1. Limitations

Despite these promising findings. the present work has some limitations that open lines for future work:

- *Single dataset*: This work has been applied to an openly available dataset of medical images. While this allowed us to analyze the privacy-utility trade-off in detail. it limits the generalization of the results obtained. In order to mitigate this limitation, we have

performed different experiments simulating both homogeneous and non-i.i.d clients.
- *Single model architecture*: We analyzed different architectures but in this study, for simplicity of the analysis, we show the results obtained with a single neural network architecture.
- *Distance metric selection*: We studied the impact of different distance metrics, but after testing in a validation set, we finally select the divergence one as stated in Section 6. For distinct use cases, other distance metrics for noise calibration should be explored, specifically for the cases in which we deal with cross-device scenarios with multiple clients.
- *Scope of CIA analysis*: The CIA scenario explored in this work represents a particular adversarial model. However, other variants, including adaptive attackers, multiple colluding clients or clients with partial auxiliary information, can be explored.

8.2. Future work

Future work will focus on addressing the limitations identified above. In particular, expanding the evaluation to additional datasets from various medical and non-medical fields, as well as to larger federations with a greater number of clients, is the main priority for future work. Specifically, issues such as how the calculation of the distance varies depending on the number of clients and the behavior of metric-sensitive calibration in cross-device FL environments still need to be addressed.

We also aim to study the impact of other distance metrics, including adaptive metric-selection strategies, which may provide stringer robustness against CIA. Regarding the threat model, we intent to extend the CIA scenario to more complex adversarial settings, in order to better capture the real-world implications and to design strengthened mechanisms, as well as expanding the simulations performed for the multi-round setting.

In addition, while Appendix B provides an empirical analysis across different noise levels, a formal Pareto frontier construction across a wider range of noise multipliers and client configurations would provide a more systematic characterization of the trade-off between utility and privacy offered by both global-DP and metric-privacy aware noise calibration.

Finally, as we have already integrated this methodology in state of the art frameworks (e.g. *flower)*, we plan to maintain such integration allowing reproducibility and enabling researchers to build upon this work.

CRediT authorship contribution statement

Judith Sáinz-Pardo Díaz: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Andreas Athanasiou:** Writing – original draft, Methodology, Investigation, Conceptualization. **Kangsoo Jung:** Writing – original draft, Methodology, Investigation, Conceptualization. **Catuscia Palamidessi:** Writing – original draft, Supervision, Methodology, Conceptualization. **Álvaro López García:** Writing – original draft, Supervision, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table A.14
Mean and standard deviation of the accuracy obtained for the test set in 5 different runs of the model. Homogeneous clients,.

	Mean accuracy ± std (5 runs)		
Strategy	*Vanilla FL*	*Global-DP*	*Metric-privacy*
FedAvg	0.930 ± 0.011	0.881 ± 0.013	0.913 ± 0.019
FedAvgM	0.880 ± 0.004	0.781 ± 0.021	0.812 ± 0.028
FedMedian	0.931 ± 0.000	0.867 ± 0.013	0.897 ± 0.004
FedProx	0.923 ± 0.011	0.880 ± 0.014	0.919 ± 0.011
FedOpt	0.947 ± 0.005	0.912 ± 0.010	0.924 ± 0.010
FedYogi	0.939 ± 0.008	0.899 ± 0.007	0.895 ± 0.013

Acknowledgments

Judith Sáinz-Pardo Díaz and Álvaro López García would like to thank the funding through the project AI4EOSC "Artificial Intelligence for the European Open Science Cloud" that has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement number 101058593 and from the SIESTA project "Secure Interactive Environments for Sensitive daTa Analytics", funded by the European Union (Horizon Europe) under grant agreement number 101131957.

The work of Andreas Athanasiou is supported by the project RECITALS, funded by the EU (grant agreement 101168490) and the project CRYPTECS, funded by the ANR (project number ANR-20-CYAL-0006) and by the BMBF (project number 16KIS1439). The work of Kangsoo Jung is supported by the project ELSA, funded by the Horizon Europe Framework (with project number 101070617). The work of Catuscia Palamidessi is supported by the project HYPATIA, funded by the ERC (grant agreement number 835294).

Appendix A. Analysis of the performance of the trained model in different runs

In this section we show the mean accuracy obtained for the three approaches (vanilla FL, global-DP and metric-privacy) with the different aggregation strategies for the case of homogeneous clients. The results shown in Table A.14 correspond with the mean and standard deviation of five runs of the models evaluated in the test dataset (90 models are evaluated in total, five runs for each strategy and privacy approach). The idea is to evaluate if the experiments conducted are statistically consistent.

Note that the interest of running the experiments multiple times is from the point of view of the two privacy approaches (that add a component of randomness, as well as other strategies that use an initial model). In all the cases a seed is fixed in order to start with the same model in all the clients (we see that with *FedMedian* without DP the std is 0). In view of the standard deviation obtained in these 90 experiments (5 for each approach and aggregation function), for the other scenarios (non-i.i.d and client inference attacks), we show the results obtained for a fixed seed, for a better analysis of the results and for a better fit to a real use case where the model is run once the hyperparameter optimization has been completed and then it is evaluated on a new test dataset. We can note than in mean all the results with metric-privacy are better than with global-DP for the test set except for *FedYogi*. However, if we take the upper bound taken into account the std, metric-privacy even performs slightly better than global-DP also in this case.

The standard deviations observed across the five runs are consistently small (e.g., ≤ 0.028 across all strategies and approaches), confirming that the single-run results reported in the main tables are stable and representative of the multi-run distribution. This low variance supports the validity of presenting results for a fixed seed after hyperparameter optimization, which is standard practice in applied studies in the medical domain.

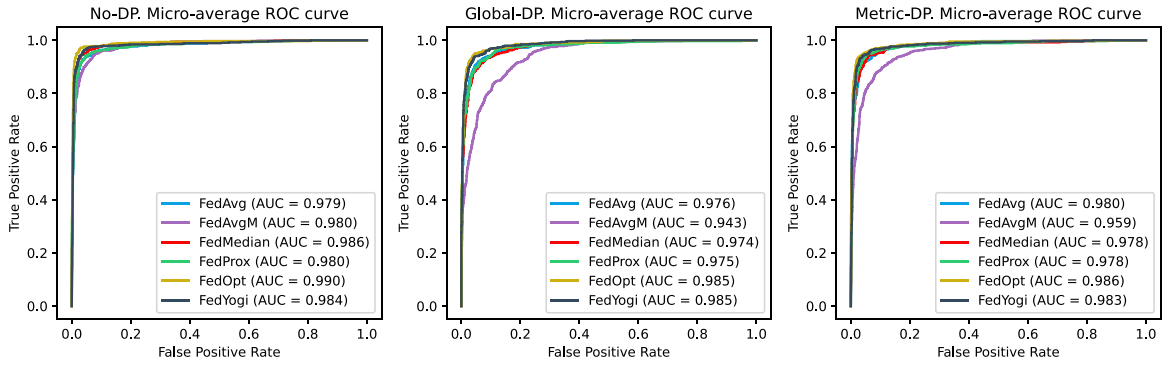


Fig. C.2. ROC curves and AUC obtained in the client test set with each strategy and DP approach. Homogeneous clients.

Table B.15
Comparison on homogeneous clients of the mean and the standard deviation of the aggregated accuracy in the last five rounds of the FL scheme. Aggregation performed with global-DP and metric-privacy with noise multiplier of 0.003.

	Mean accuracy ± std (last 5 rounds)	
Strategy	*Global-DP*	*Metric-privacy*
FedAvg	0.956 ± 0.007	0.968 ± 0.001
FedAvgM	0.851 ± 0.01	0.841 ± 0.011
FedMedian	0.959 ± 0.005	0.963 ± 0.004
FedProx	0.964 ± 0.003	0.973 ± 0.004
FedOpt	0.949 ± 0.006	0.961 ± 0.004
FedYogi	0.927 ± 0.004	0.942 ± 0.004

Appendix B. Selection of the noise multiplier

The noise multiplier was selected through cross-validation with multiple values. After several trials, we found that the value 0.01 yielded the most consistent results. In summary, other noise multipliers we evaluated were 0.005, 0.003, 0.001, 0.05, and 0.1, among others. As a benchmark, we take the case of i.i.d. data. For reference, Table shows the results obtained with 0.003 as noise multiplier. We can note here that the results are in some cases even better than in the case of vanilla FL, so we are adding an insubstantial amount of noise that will not prevent attacks either. In any case, this table shows that the results obtained with the metric-privacy inspired approach improve on those obtained with GDP for all aggregation functions except *FedAvgM*, which is also the one with the worst convergence and performance (see Table B.15).

In view of the above results, considering smaller values (e.g., 0.001) does not seem reasonable, as the amount of noise added would be too low and would not prevent attacks. Furthermore, regarding values greater than 0.01 (adding more noise), we have verified with the *FedAvg* aggregation strategy that, for example with 0.05 and 0.1, the model fails to converge, obtaining for the GDP case an average of 0.453 ± 0.062 and 0.408 ± 0.074 respectively for the last five rounds. Thus, in view of all the tests carried out, we decided to take the noise multiplier value of 0.01 as a reference for the experiments conducted in this work.

Appendix C. ROC curves and AUC in the test set

Fig. C.2 shows the ROC curves and the AUC obtained with each aggregation strategy for the case of homogeneous clients and with the three approaches (no-DP or vanilla FL, global-DP, metric-privacy).

In view of Fig. C.2, we can see that the best results for the scenario of homogeneous clients are reached without DP in 5 out of 6 cases (with *FedAvg* the results are slightly better with metric-privacy, as explained in Section 7.3.1). In addition, the ones obtained with metric-privacy are always better than those with global-DP. The former is also fulfilled for the case of non-i.i.d. clients as is shown in Fig. C.3, but in this case the best results are always obtained without DP. Note than in order to get the ROC curves and to calculate the AUC in this case of multi-class classification, the micro-averaged one-vs-rest ROC-AUC score has been used, comparing each class against the other three in the labels.

Appendix D. Evolution of the aggregated accuracy in the clients' test set

In Fig. D.4 the aggregated accuracy for each client's test set in each round of the FL scheme is shown for the case of homogeneous clients. Here we can note that the models trained with the notion of metric-privacy reach a convergence more similar and softer to the model aggregated without DP that the one aggregated applying global-DP (which shows more fluctuations over the course of the rounds). It should be noted that in the three cases the trends of the rounds without DP, with global-DP and with metric-privacy (in the figure referred as MDP) are similar, with *FedOpt* being the most accurate, followed by *FedYogi* (which is more clearly observed with global-DP), and ending in all cases with *FedMedian* and *FedAvgM* as the least reliable.

Fig. D.5 shows the results obtained for the aggregated accuracy in each round for the case of non-i.i.d clients. Again in this case the curves are smoother when using the notion of metric-privacy than with global-DP, due to the amount of noise added. In the three cases the same trend is maintained, with *FedOpt* being the best performing strategy, while *FedAvgM* is the least accurate.

Appendix E. Client inference attack for different aggregation strategies

In Section 7.4.1 we have shown the performance obtained for the scenario simulated concerning the first round single shot CIA performed using both *FedAvg* and *FedOpt*. For completeness of the analysis, the results for the other four aggregation functions are shown in this appendix in Tables E.16–E.19:

We can note that for the case of *FedAvgM*, the approach under the metric-privacy notion provides a greater difference that global-DP, while it also provides better results in terms of the loss. This may be due for a reduction in the overfitting to the target client. On the other hand, for the other three aggregation functions, a better result in terms of the distance is obtained with global-DP, but there are still better with metric-privacy that without DP. This shows that metric-privacy helps to prevent client inference attacks while allowing a better convergence of the model than classic global-DP.

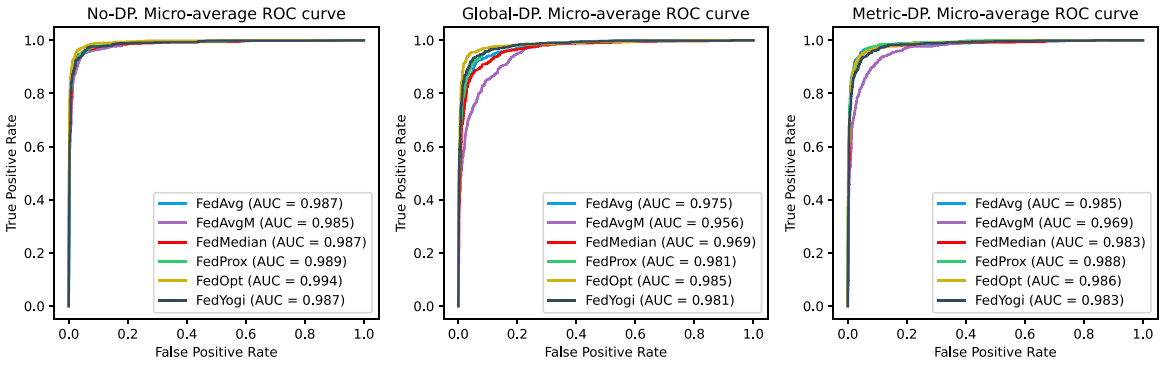


Fig. C.3. ROC curves and AUC obtained in the client test set with each strategy and DP approach. Non-i.i.d clients.

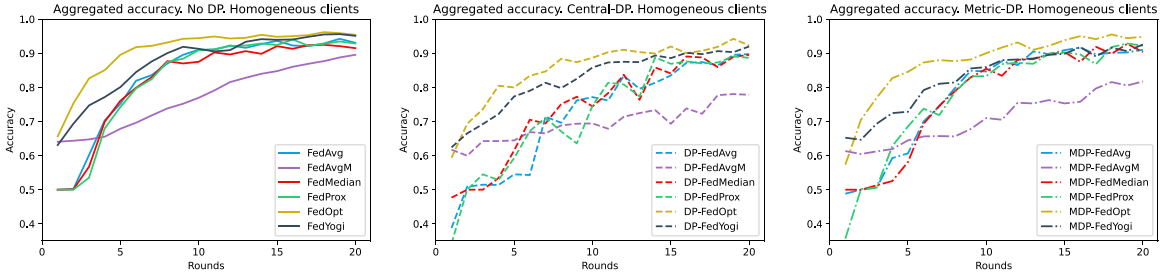


Fig. D.4. Evolution of the aggregated accuracy in each round of the FL architecture. Homogeneous clients.

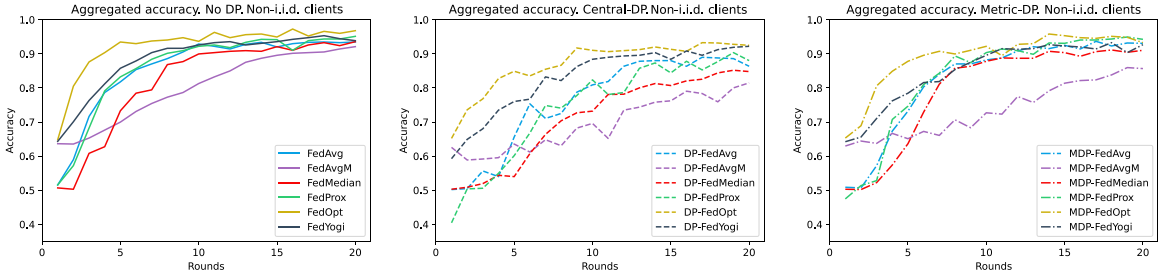


Fig. D.5. Evolution of the aggregated accuracy in each round of the FL architecture. Non-i.i.d clients.

Table E.16
Loss obtained for the aggregated test set and for the shadow dataset of client 3 with *FedAvgM* strategy.

Client	*Vanilla FL*	*Global-DP*	*Metric-privacy*
Aggregated	0.897	0.969	0.945
Target	1.118	1.115	1.211
Difference (%)	19.782	13.105	21.965

Table E.17
Loss obtained for the aggregated test set and for the shadow dataset of client 3 with *FedMedian* strategy.

Client	*Vanilla FL*	*Global-DP*	*Metric-privacy*
Aggregated	1.068	2.079	1.393
Target	1.188	2.911	1.880
Difference (%)	10.129	28.587	25.888

Table E.18
Loss obtained for the aggregated test set and for the shadow dataset of client 3 with *FedProx* strategy.

Client	*Vanilla FL*	*Global-DP*	*Metric-privacy*
Aggregated	1.032	1.846	1.163
Target	1.182	2.681	1.548
Difference (%)	12.719	31.173	24.820

Table E.19
Loss obtained for the aggregated test set and for the shadow dataset of client 3 with *FedYogi* strategy.

Client	*Vanilla FL*	*Global-DP*	*Metric-privacy*
Aggregated	0.795	0.949	0.815
Target	0.895	1.170	0.957
Difference (%)	11.150	18.861	14.888

Appendix F. Experiments reproducibility

For carrying out the experiments conducted in this work, the following Python libraries and versions with their corresponding dependencies have been used (only the main ones are detailed): flower (v1.13.0), TensorFlow (v2.14.0), keras (v2.14.0), scikit-learn (v1.5.2), scipy (v1.14.1), tqdm (v4.66.5), opencv-python (v4.10.0.84), numpy (v1.26.0), pandas (v2.2.3), pillow (v11.0.0) and matplotlib (v3.9.2).

Appendix G. Including server side metric-privacy in flower

To add the notion of metric-privacy inspired noise calibration, we have modified the code of the class *DifferentialPrivacyServerSideFixedClipping* from the Flower framework in such a way that a new function (named *distance_metric*) is added for dynamically calculate in each round the maximum distance between the different pairs of models as explained in Section 6. This function receives the results containing a list with the tuple *(ClientProxy, FitRes)*.

Specifically, within the function *aggregate_fit*, a call to *distance_metric*) has been included to calculate the distance once the clipping is performed, and then this distance is entered by dividing the noise multiplier in the call to the *add_gaussian_noise_to_params* and *compute_stdv* functions for both adding the noise and calculating the standard deviation of the mechanism applied.

These modifications have been done in the file *dp_fixed_clipping.py* that is located under the following structure (note that this has been implemented for *flower* version v.1.13.0, in newer versions, this structure has been refactored):

```
flower
├── ...
├── datasets
├── dev
├── doc
├── e2e
├── examples
├── glossary
├── src
│   ├── cc/flwr
│   ├── docker
│   ├── kotlin
│   ├── proto/flwr/proto
│   ├── py
│   │   ├── flwr
│   │   │   ├── cli
│   │   │   ├── client
│   │   │   ├── common
│   │   │   ├── proto
│   │   │   ├── server
│   │   │   │   ├── compat
│   │   │   │   ├── driver
│   │   │   │   ├── serverapp
│   │   │   │   ├── strategy
│   │   │   │   │   ├── ...
│   │   │   │   │   ├── dp_fixed_clipping.py
│   │   │   │   │   └── ...
│   │   │   │   └── ...
│   │   │   └── ...
│   │   └── flwr_tool
│   └── swift/flwr
└── ...
```

Appendix H. Code availability and usage

The code implemented is publicly available in https://github.com/AI4EOSC/flower/tree/metric_privacy. This code allows to use the *flower* framework with the option of metric-privacy with different aggregation strategies. As it has been added as a new class, it could be adapted to be integrated in other FL frameworks. In addition, in https://github.com/ai4os/ai4os-federated-server we provide a GitHub repository the code for deploying a *flower* FL server allowing to integrate metric-privacy. The user can customize the distance metric to be applied by modifying the function *distance_metric* taking into account the use case under study.

Data availability

The data used in this study are openly available and can be obtained from [37].

References

[1] L. Li, Y. Fan, M. Tse, K.-Y. Lin, A review of applications in federated learning, Comput. Ind. Eng. 149 (2020) 106854.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, PMLR, 2017, pp. 1273–1282.

[3] R. Shokri, M. Stronati, C. Song, V. Shmatikov, Membership inference attacks against machine learning models, in: 2017 IEEE Symposium on Security and Privacy, SP, IEEE Computer Society, Los Alamitos, CA, USA, 2017, pp. 3–18, http://dx.doi.org/10.1109/SP.2017.41, URL https://doi.ieeecomputersociety.org/10.1109/SP.2017.41.

[4] H. Hu, Z. Salcic, L. Sun, G. Dobbie, X. Zhang, Source inference attacks in federated learning, in: ICDM, IEEE, 2021.

[5] L. Zhu, Z. Liu, S. Han, Deep leakage from gradients, Adv. Neural Inf. Process. Syst. 32 (2019).

[6] C. Chen, L. Lyu, H. Yu, G. Chen, Practical attribute reconstruction attack against federated learning, IEEE Trans. Big Data (2022).

[7] J. Ma, S.-A. Naas, S. Sigg, X. Lyu, Privacy-preserving federated learning based on multi-key homomorphic encryption, Int. J. Intell. Syst. 37 (9) (2022) 5880–5901.

[8] K. Chatzikokolakis, M.E. Andrés, N.E. Bordenabe, C. Palamidessi, Broadening the scope of differential privacy using metrics, in: E. De Cristofaro, M. Wright (Eds.), Privacy Enhancing Technologies, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 82–102.

[9] M.S. Alvim, K. Chatzikokolakis, C. Palamidessi, A. Pazii, Metric-based local differential privacy for statistical applications, in: 31st Computer Security Foundations Symposium, CSF 2018, IEEE Computer Society, 2018, pp. 262–267.

[10] K. Pan, Y.-S. Ong, M. Gong, H. Li, A.K. Qin, Y. Gao, Differential privacy in deep learning: A literature survey, Neurocomputing (2024) 127663.

[11] M. Abadi, A. Chu, I. Goodfellow, H.B. McMahan, I. Mironov, K. Talwar, L. Zhang, Deep learning with differential privacy, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016, pp. 308–318.

[12] T. Ha, T.K. Dang, T.T. Dang, T.A. Truong, M.T. Nguyen, Differential privacy in deep learning: an overview, in: 2019 International Conference on Advanced Computing and Applications, ACOMP, IEEE, 2019, pp. 97–102.

[13] Y. Xie, P. Li, C. Wu, Q. Wu, Differential privacy stochastic gradient descent with adaptive privacy budget allocation, in: 2021 IEEE International Conference on Consumer Electronics and Computer Engineering, ICCECE, IEEE, 2021, pp. 227–231.

[14] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015, URL https://www.tensorflow.org/. Software available from tensorflow.org.

[15] Opacus PyTorch library. Available from opacus.ai https://opacus.ai.

[16] M. Nasr, R. Shokri, A. Houmansadr, Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning, in: 2019 IEEE Symposium on Security and Privacy, SP, IEEE, 2019, pp. 739–753.

[17] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, F. Tramer, Membership inference attacks from first principles, in: 2022 IEEE Symposium on Security and Privacy, SP, IEEE, 2022, pp. 1897–1914.

[18] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P.S. Yu, X. Zhang, Membership inference attacks on machine learning: A survey, ACM Comput. Surv. 54 (11s) (2022) http://dx.doi.org/10.1145/3523273.

[19] A. Pustozerova, J. Baumbach, R. Mayer, Differentially private federated learning: Privacy and utility analysis of output perturbation and DP-SGD, in: 2023 IEEE International Conference on Big Data, BigData, 2023, pp. 5549–5558, http://dx.doi.org/10.1109/BigData59044.2023.10386466.

[20] A. Rauniyar, D.H. Hagos, D. Jha, J.E. Håkegård, U. Bagci, D.B. Rawat, V. Vlassov, Federated learning for medical applications: A taxonomy, current trends, challenges, and future research directions, IEEE Internet Things J. 11 (5) (2024) 7374–7398, http://dx.doi.org/10.1109/JIOT.2023.3329061.

[21] I. Dayan, H.R. Roth, A. Zhong, A. Harouni, A. Gentili, A.Z. Abidin, A. Liu, A.B. Costa, B.J. Wood, C.-S. Tsai, et al., Federated learning for predicting clinical outcomes in patients with COVID-19, Nature Med. 27 (10) (2021) 1735–1743.

[22] M. Adnan, S. Kalra, J.C. Cresswell, G.W. Taylor, H.R. Tizhoosh, Federated learning and differential privacy for medical image analysis, Sci. Rep. 12 (1) (2022) 1953.

[23] R. Hu, Y. Guo, H. Li, Q. Pei, Y. Gong, Personalized federated learning with differential privacy, IEEE Internet Things J. 7 (10) (2020) 9530–9539, http://dx.doi.org/10.1109/JIOT.2020.2991416.

[24] M. Naseri, J. Hayes, E. De Cristofaro, Local and central differential privacy for robustness and privacy in federated learning, 2020, arXiv preprint arXiv:2009.03561.

[25] M.A. Hidayat, Y. Nakamura, B. Dawton, Y. Arakawa, AGC-DP: Differential privacy with adaptive Gaussian clipping for federated learning, in: 2023 24th IEEE International Conference on Mobile Data Management, MDM, 2023, pp. 199–208, http://dx.doi.org/10.1109/MDM58254.2023.00042.

[26] Z. He, L. Wang, Z. Cai, Clustered federated learning with adaptive local differential privacy on heterogeneous IoT data, IEEE Internet Things J. 11 (1) (2024) 137–146, http://dx.doi.org/10.1109/JIOT.2023.3299947.

[27] T.-M.H. Hsu, H. Qi, M. Brown, Measuring the effects of non-identical data distribution for federated visual classification, 2019, arXiv preprint arXiv:1909.06335.

[28] D.J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, H.L. Kwing, T. Parcollet, P.P.d. Gusmão, N.D. Lane, Flower: A friendly federated learning research framework, 2020, arXiv preprint arXiv:2007.14390.

[29] D. Yin, Y. Chen, R. Kannan, P. Bartlett, Byzantine-robust distributed learning: Towards optimal statistical rates, in: International Conference on Machine Learning, Pmlr, 2018, pp. 5650–5659.

[30] T. Li, A.K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, Federated optimization in heterogeneous networks, Proc. Mach. Learn. Syst. 2 (2020) 429–450.

[31] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, H.B. McMahan, Adaptive federated optimization, 2020, arXiv preprint arXiv:2003.00295.

[32] M.S. Alvim, N. Fernandes, A. McIver, C. Morgan, G.H. Nunes, A novel analysis of utility in privacy pipelines, using kronecker products and quantitative information flow, in: Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS '23, Association for Computing Machinery, New York, NY, USA, 2023, pp. 1718–1731, http://dx.doi.org/10.1145/3576915.3623081.

[33] F. Galli, S. Biswas, K. Jung, T. Cucinotta, C. Palamidessi, Group privacy for personalized federated learning, in: Proceedings of the 9th International Conference on Information Systems Security and Privacy-ICISSP, SCITEPRESS-Science and Technology Publications, 2023, pp. 252–263.

[34] W. Zhang, X. Wang, P. Zhou, W. Wu, X. Zhang, Client selection for federated learning with non-IID data in mobile edge computing, IEEE Access 9 (2021) 24462–24474, http://dx.doi.org/10.1109/ACCESS.2021.3056919.

[35] H.B. McMahan, D. Ramage, K. Talwar, L. Zhang, Learning differentially private recurrent language models, 2017, arXiv preprint arXiv:1710.06963.

[36] P. Guerra-Balboa, À. Miranda-Pascual, J. Parra-Arnau, T. Strufe, Composition in differential privacy for general granularity notions, in: 2024 IEEE 37th Computer Security Foundations Symposium, CSF, IEEE, 2024, pp. 680–696.

[37] F.G. Salieh, Alzheimer MRI Dataset, Hugging Face, 2023, URL https://huggingface.co/datasets/Falah/Alzheimer_MRI.

[38] M. Hatami, F. Yaghmaee, R. Ebrahimpour, Investigating the potential of reinforcement learning and deep learning in improving alzheimer's disease classification, Neurocomputing 597 (2024) 128119, http://dx.doi.org/10.1016/j.neucom.2024.128119, URL https://www.sciencedirect.com/science/article/pii/S0925231224008907.