SIFU: Sequential Informed Federated Unlearning for Efficient and Provable Client Unlearning in Federated Optimization

Yann Fraboni[*]
Accenture Labs
INRIA Sophia-Antipolis

Martin Van Waerebeke[*]
INRIA Paris
DIENS - PSL

Richard Vidal
Accenture Labs

Laetitia Kameni
Accenture Labs

Kevin Scaman
INRIA Paris
DIENS - PSL

Marco Lorenzi
INRIA Sophia-Antipolis

Abstract

Machine Unlearning (MU) is an increasingly important topic in machine learning safety, aiming at removing the contribution of a given data point from a training procedure. Federated Unlearning (FU) consists in extending MU to unlearn a given client's contribution from a federated training routine. While several FU methods have been proposed, we currently lack a general approach providing formal unlearning guarantees to the FedAvg routine, while ensuring scalability and generalization beyond the convex assumption on the clients' loss functions. We aim at filling this gap by proposing SIFU (Sequential Informed Federated Unlearning), a new FU method applying to both convex and non-convex optimization regimes. SIFU naturally applies to FedAvg without additional computational cost for the clients and provides formal guarantees on the quality of the unlearning task. We provide a theoretical analysis of the unlearning properties of SIFU, and practically demonstrate its effectiveness as compared to a panel of unlearning methods from the state-of-the-art.

1 Introduction

With the emergence of new data regulations, such as the EU General Data Protection Regulation (GDPR) (Voigt and Von dem Bussche, 2017) and the California Consumer Privacy Act (CCPA) (Harding et al., 2019), the storage and processing of sensitive personal data is often the subject of strict constraints and restrictions. In particular, the "right to be forgotten" states that personal data must be erased upon request from the concerned individuals, with subsequent potential implications on machine learning models trained by using this data. Machine Unlearning (MU) is an emerging discipline that studies methods that aim at removing the contribution of given data instances used to train a machine learning model. Current MU approaches are essentially based on routines that modify the model weights in order to guarantee the "unlearning" of a given data point, i.e. to obtain a model equivalent to a hypothetical one trained without this data point (Cao and Yang, 2015; Bourtoule et al., 2021).

Motivated by data governance and confidentiality concerns, Federated Learning (FL) has gained popularity in the last years to allow data owners to collaboratively learn a model without sharing their respective data. Among the different FL approaches, federated averaging (FedAvg) has emerged as the most popular optimization scheme (McMahan et al., 2017). An optimization round of FedAvg requires data owners to receive the current global model from the server, which is updated by performing a fixed amount of Stochastic Gradient Descent (SGD) steps before sending back the resulting model. The new global model is then created as the weighted average of the client updates. The FL communication design ensures clients that their data is solely used to compute their model update, while established theory guarantees convergence of the model to a stationary point of the clients' joint optimization problem (Wang et al., 2020; Li et al., 2020).

With the current deployments of FL in the real-world,

Proceedings of the 27[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. [*] Equal contribution.

it is of crucial importance to extend MU to Federated Unlearning (FU), for guaranteeing the unlearning of clients wishing to opt-out from a collaborative training routine. This is not straightforward, since most current MU schemes have been proposed in the centralized learning setting, and cannot be seamlessly applied to the federated one. Typical issues include the need for exchanging high-order quantities related to the model parameters, or additional and potentially sensitive client information (Guo et al., 2020; Izzo et al., 2021; Golatkar et al., 2020a,b, 2021). While several Federated Unlearning (FU) methods have been proposed, few are backed by theoretical guarantees on the effectiveness of unlearning (Liu et al., 2022; Jin et al., 2023), or are compatible with typical FL assumptions on data access or availability (Liu et al., 2021).

To address these shortcomings, in this work we introduce **Sequential Informed Federated Unlearning (SIFU)**, a novel efficient FU approach to remove clients' contributions from the federated model with quantifiable unlearning guarantees. SIFU is compatible with FedAvg-based training and requires minimal additional computations from the server *and none from the clients*. Specifically, at every round of FL optimization, the server quantifies the norm of each client's contribution to the global model. Upon receiving an unlearning request from a client, the server retrieves the iteration at which the client's contribution exceeds a pre-defined unlearning budget from the FL training history, and initializes the unlearning procedure from the associated intermediate global model. Unlearning guarantees are provided by introducing a novel randomized mechanism to perturb the selected intermediate model with client-specific noise. We develop a theory demonstrating the unlearning capabilities of SIFU in both convex and non-convex FL optimization settings. We first introduce IFU (Informed FU) to account for only one unlearning request, and then generalize it to SIFU, which can handle an arbitrary number of sequential requests, as is done in Neel et al. (2021); Gupta et al. (2021).

This manuscript is structured as follows. In Section 2, we provide formal definitions for MU, FL, and FU, and introduce the state-of-the art of FU. In Section 3, we introduce sufficient conditions for IFU to unlearn a client from the FL routine (Theorem 2). In Section 4, we extend IFU to the sequential unlearning setting with Sequential IFU (SIFU). Finally, in Section 5, we experimentally demonstrate on different tasks and datasets that SIFU leads to more efficient unlearning procedures as compared to basic re-training and state-of-the-art FU approaches.

2 Background and Related Work

Sections 2.1 and 2.2 respectively introduce the basic concepts of MU and FL. The state-of-the-art on FU is discussed in 2.3.

2.1 Machine Unlearning

Let us consider a dataset $\mathcal{D}$ composed of two disjoint datasets: $\mathcal{D}_f$, the cohort of data samples on which unlearning must be applied after FL training, and $\mathcal{D}_k$, the remaining data samples. Hence, we have $\mathcal{D} = \mathcal{D}_f \sqcup \mathcal{D}_k$. We also consider $\mathcal{M}(\mathcal{D})$, the ML model parameters resulting from training with optimization scheme $\mathcal{M}$ on dataset $\mathcal{D}$. We introduce in this section the different unlearning baselines and methods typically used to unlearn $\mathcal{D}_f$ from the trained model $\mathcal{M}(\mathcal{D})$.

MU through retraining. Within this setting, a new training is performed from scratch with only $\mathcal{D}_k$ as training data. Retraining from scratch is a typical MU baseline as it provides unlearning by construction, albeit with a generally high computational cost.

MU through fine-tuning. Fine-tuning on the remaining data $\mathcal{D}_k$ has been proposed as a practical approach to unlearn the specificities of $\mathcal{D}_f$. This is a common MU baseline (**?**), with however no unlearning guarantees (Appendix A).

MU through model scrubbing. Another unlearning approach consists in applying a "scrubbing" transformation h to the model $\mathcal{M}(\mathcal{D})$ such that the resulting model is as close as possible to the one that would be trained with only $\mathcal{D}_k$, i.e. $h(\mathcal{M}(\mathcal{D})) \approx \mathcal{M}(\mathcal{D}_k)$ (Ginart et al., 2019). Existing work mostly relies on the quadratic approximation of the loss function to define the scrubbing method h as

$$h_{\mathcal{D}_k}(\boldsymbol{\theta}) = \boldsymbol{\theta} - H^{-1}_{\mathcal{D}_k}(\boldsymbol{\theta}) \nabla f_{\mathcal{D}_k}(\boldsymbol{\theta}), \tag{1}$$

where $H_{\mathcal{D}_x}(\boldsymbol{\theta})$ is the Hessian of the loss function evaluated on the remaining data points $\mathcal{D}_k$. With equation (1), h reduces to performing a Newton step, and has been derived in previous MU works under different theoretical assumptions that can be generalized by considering a quadratic approximation of the loss function (Guo et al., 2020; Izzo et al., 2021; Golatkar et al., 2020a,b, 2021; Mahadevan and Mathioudakis, 2021). The main drawback behind the use of the scrubbing function (1) is the estimation of the Hessian, which can be both intractable for large models and prone to information leakage. Finally, the scrubbing function (1) is often coupled with Gaussian noise perturbation on the resulting weights, to compensate the quadratic approximation of the loss function, or the approximation of the Hessian (Golatkar et al., 2020a,b, 2021).

MU through noise perturbation. This unlearning method consists in randomly perturbating the trained model $\mathcal{M}(\mathcal{D})$ to unlearn specificities from data samples in $\mathcal{D}_f$ (Neel et al., 2021; Gupta et al., 2021; **?**). The noise is set such that the guarantees of Definition 1 are satisfied, where (ϵ, δ) are parameters quantifying the unlearning guarantees.

Definition 1. *Let f_m be a randomized mechanism taking model parameters as an input. (ϵ, δ)-unlearning trough f_m of a data point $\{x_m, y_m\}$ from a model $\mathcal{M}(D)$ is achieved if, for any subset $\mathcal{S}$ of the model parameters space and $D_{-m} \coloneqq D \setminus \{x_m, y_m\}$, we have*

$$\mathbb{P}(f_m(\mathcal{M}(D)) \in \mathcal{S}) \leq e^{\epsilon}\mathbb{P}(f_m(\mathcal{M}(D_{-m})) \in \mathcal{S}) + \delta \quad (2)$$

$$\text{and } \mathbb{P}(f_m(\mathcal{M}(D_{-m})) \in \mathcal{S}) \leq e^{\epsilon}\mathbb{P}(f_m(\mathcal{M}(D)) \in \mathcal{S}) + \delta. \quad (3)$$

(Guo et al., 2020) shows the relationship between Definition 1 and the randomized mechanism in Differential Privacy (Dwork and Roth, 2014; Chen et al., 2020).

2.2 Federated Optimization and FedAvg

In FL, we consider a learning setup with M clients, and define $I = \{1, ..., M\}$ as the set of indices of the participating clients. Each client owns a dataset D_i composed of $|D_i| = n_i$ data samples. We consider a loss $f(\boldsymbol{x}_{i,l}, \boldsymbol{y}_{i,l}, \boldsymbol{\theta})$ assessed on each data sample $(\boldsymbol{x}_{i,l}, \boldsymbol{y}_{i,l}) \in D_i$, and define a client's loss function as $f_i(\boldsymbol{\theta}) \coloneqq 1/n_i \sum_{l=1}^{n_i} f(\boldsymbol{x}_{i,l}, \boldsymbol{y}_{i,l}, \boldsymbol{\theta})$. We define for the joint dataset $D_I \coloneqq \cup_{i \in I} D_i$ the federated loss function

$$f_I(\boldsymbol{\theta}) \coloneqq \frac{1}{|D_I|} \sum_{i \in I} |D_i| f_i(\boldsymbol{\theta}). \quad (4)$$

FedAvg (McMahan et al., 2017) optimizes the loss (4) with theoretical guarantees for FL convergence to a stationary point (Wang et al., 2020; Li et al., 2020). Following Algorithm 1, at each step n, the server sends the current global model parameters $\boldsymbol{\theta}^n$ to the clients. Each client updates the model by minimizing its local cost function $f_i(\boldsymbol{\theta})$ with K SGD steps initialized with $\boldsymbol{\theta}^n$. Subsequently each client returns the updated local parameters $\boldsymbol{\theta}_i^{n+1}$ to the server. The global model parameters $\boldsymbol{\theta}^{n+1}$ at iteration step $n+1$ are then estimated by aggregating the clients' contributions, i.e.

$$\boldsymbol{\theta}^{n+1} = \boldsymbol{\theta}^n + \omega(I, \boldsymbol{\theta}^n), \quad (5)$$

where $\omega(I, \boldsymbol{\theta}^n) = \frac{1}{|D_I|} \sum_{i \in I} |D_i| \left[\boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^n\right]$.

In what follows, we consider FedAvg as reference FL framework, due to the wide adoption of this scheme in the literature, and the lower sensitivity to information leakage as opposed to FedSGD (Geng et al., 2023).

Algorithm 1 FedAvg(I, N)

1: **for** n from 0 to $N-1$ **do**
2: The server sends $\boldsymbol{\theta}^n$ to every client in I.
3: Clients perform K SGDs to compute $\boldsymbol{\theta}_i^{n+1}$.
4: The server creates $\boldsymbol{\theta}^{n+1}$, equation (5).
5: **end for**
6: **return** the trained global model $\boldsymbol{\theta}^N$

2.3 Federated Unlearning

We first note that while MU through retraining and fine-tuning naturally generalize to FU, this not the case for most MU methods, because of typical data access restrictions of FL. While a variety of FU methods have been recently proposed, only a few of them offer theoretical proofs of their unlearning capabilities. These include FU methods tailored to clustering tasks (Pan et al., 2023), linear regressions (Li et al., 2021) and class-unlearning (Wang et al., 2022). Concerning the problem of FU compatible with the FedAvg routine, some recent works develop theories for model scrubbing (Liu et al., 2022; Jin et al., 2023). Nevertheless, the working assumptions of these methods may be too restrictive as they require the client-wise computation and availability of the model's Hessian, or are based on the existence of independent data available to the server. Other limitations of recent FU methods concern the need for accessing the data to be unlearned after the unlearning request (Halimi et al., 2022). FedEraser (Liu et al., 2021) is a recent method compatible with FedAvg based on adaptive retraining. While this approach has been shown to outperform the retraining baseline, it is not backed by theoretical guarantees. The ensemble of working hypothesis of these methods is summarized in Table 1. As can be observed, there is a lack of a theoretically-proven approaches working with FedAvg, especially in the non-convex setting. With these limitations in mind, in what follows we introduce our contribution.

3 Unlearning a single client with IFU

In this section, we develop our theory for the scenario where a model is trained with FedAvg on the set of clients I, after which a client c requests unlearning of its data. In Section 3.1, we define the sensitivity of the global model with respect to a client's contribution, with an associated bound for both convex and non-convex regimes. Using Theorem 1, we introduce the perturbation procedure in Section 3.2 to unlearn a client c from the model trained with FedAvg. Finally, using Theorem 2, we introduce Informed Federated Unlearning (IFU) (Algorithm 2).

Table 1: Summary of FU methods from state-of-the-art, with related working assumptions.

	FedAvg compatible	Theoretical guarantees		FU does NOT require data from	
		convex	non-convex	the server	the clients to unlearn
Liu et al. (2022)		✓		✓	
Wang et al. (2023)				✓	✓
Gong et al. (2022)				✓	✓
Wu et al. (2022)	✓				✓
Halimi et al. (2022)	✓			✓	
Liu et al. (2021)	✓			✓	✓
Jin et al. (2023)	✓	✓			
SIFU (ours)	✓	✓	✓	✓	✓

3.1 Bounding the Model Sensitivity

In what follows, we define the joint dataset for a subset of client $I_x \subset I$ as $D_{I_x} := \cup_{i \in I_x} D_i$. Additionally, for any given client c, we define $I_{-c} := I \setminus \{c\}$.

We introduce the *model sensitivity* with respect to client c after n aggregation rounds of FedAvg as

$$\alpha(n, c) := \|\text{FedAvg}(I, n) - \text{FedAvg}(I_{-c}, n)\|_2 , \quad (6)$$

where $\text{FedAvg}(I, n)$ it the global model obtained by applying Algorithm 1 for n iterations over the data of clients in I. While the model sensitivity is an ideal measure of the impact of a given client on the federated optimization result at step n, the computation of this quantity for each client is not feasible in a typical FL routine. We therefore introduce a proxy for this quantity, to keep track at every FL round of each client's contribution to the aggregation (5):

$$\Delta_c(I, \boldsymbol{\theta}) := \|\omega(I, \boldsymbol{\theta}) - \omega(I_{-c}, \boldsymbol{\theta})\| \quad (7)$$

In Theorem 1, we establish a bound for the *model sensitivity*, relating this quantity to the history of updates provided by the clients across FL rounds.

Theorem 1. *For smooth client's local loss functions (i.e. with Lipschitz-continuous gradients), we have*

$$\alpha(n, c) \leq \Psi(n, c), \quad (8)$$

with the bounded sensitivity Ψ defined as:

$$\Psi(n, c) = \sum_{s=0}^{n-1} B(f_I, \eta)^{\gamma_{s,n}} \cdot \Delta_c(I, \boldsymbol{\theta}^s), \quad (9)$$

where η is the learning rate, $\gamma_{s,n} = (n - s - 1)K$, and $B(f_I, \eta) < 1$, $B(f_I, \eta) = 1$ or $B(f_I, \eta) > 1$ if the clients' loss functions are smooth and, respectively, strongly convex, convex, or non-convex. The exact formula for $B(f_I, \eta)$ is given in Appendix B, equations (26) to (28).

Proof. We prove Theorem 1 in Appendix B. □

3.2 From model sensitivity to certified unlearning

In this section, we introduce a randomized mechanism to provide guarantees for the unlearning of a given client c, where the magnitude of the perturbation process (Dwork and Roth, 2014) is defined based on the sensitivity of Theorem 1. In practice, we define a Gaussian noise mechanism to perturb each parameter of the global model $\boldsymbol{\theta}^n$ such that we achieve (ϵ, δ)-unlearning of client c, according to Definition 1. We give in Theorem 2 sufficient conditions for the noise perturbation to satisfy Definition 1.

Theorem 2. *Under smoothness assumptions, applying to the global model $\boldsymbol{\theta}^n$ a Gaussian noise $N(0, \sigma(n, c)^2 \boldsymbol{I}_{\boldsymbol{\theta}})$, with $\boldsymbol{I}_{\boldsymbol{\theta}}$ the identity matrix and*

$$\sigma(n, c) := \left[2\left(\ln(1.25) - \ln(\delta)\right)\right]^{1/2} \epsilon^{-1} \Psi(n, c), \quad (10)$$

achieves (ϵ, δ)-unlearning of client c according to Definition 1.

Proof. While a formal proof is given in Appendix C, this statement follows directly from Theorem 1 coupled with Theorem A.1 of (Dwork and Roth, 2014). □

We note that, according to Theorem 2, (ϵ, δ)-unlearning a client from a given global model requires to prescribe a client-specific standard deviation for the noise, proportional to the bounded sensitivity. This is not an issue, as the bounded sensitivity is a scalar quantity that can be easily computed and stored from the clients' contribution.

In what follows, the unlearning procedure will be defined with respect to the sensitivity threshold Ψ^* related to the unlearning budget (ϵ, δ) and standard deviation σ:

$$\Psi^* := \left[2\left(\ln(1.25) - \ln(\delta)\right)\right]^{-1/2} \epsilon\sigma. \quad (11)$$

3.3 Informed Federated Unlearning (IFU)

Algorithm 2 Informed Federated Unlearning (IFU)

LEARNING WITH FEDAVG

FEDAVG(I, N) initialized on initial model $\boldsymbol{\theta}^0$.
for n from 0 to $N-1$, and i from 1 to c **do**
  Compute $\Psi(n, i)$, equation (9).
end for

UNLEARNING

Require: c, ϵ, δ, σ, amount of retraining steps $\tilde{N}$.
1: Get Ψ^* with equation (11).
2: Get $T = \arg\max_n (\Psi(n, c) \leq \Psi^*)$.
3: The new global model is $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta}^T + N(0, \sigma^2 \boldsymbol{I}_{\boldsymbol{\theta}})$.
4: Run FEDAVG$(I_{-c}, \tilde{N})$ initialized on $\tilde{\boldsymbol{\theta}}$.

Using the bounded sensitivity (9) and Theorem 2, we introduce Informed Federated Unlearning (IFU) to unlearn the contribution of client $c \in I$ from a FL training procedure based on FEDAVG. Algorithm 2 provides the implementation of IFU on top of FEDAVG. We note that during the FL training, IFU requires the server to compute the bounded sensitivity metric $\Psi(n, i)$ from each client's contribution $\boldsymbol{\theta}_i^{n+1}$ and current global model $\boldsymbol{\theta}^n$. These quantities are tracked throughout FL iterations and are used to identify the optimal unlearning strategy after request from a client c. We note that since the IFU procedure of Algorithm 2 relies on FEDAVG, the convergence guarantees are the same as in (Wang et al., 2020; Li et al., 2020).

To unlearn client c, the server identifies the unlearning index T associated to the history of bounded sensitivity metrics, i.e. the most recent global model index such that a perturbation of size σ satisfies Theorem 2:

$$T \coloneqq \max\{n \ : \ \Psi(n, c) \leq \Psi^*\}\,. \tag{12}$$

The new global model is obtained after perturbation $\tilde{\boldsymbol{\theta}} \coloneqq \boldsymbol{\theta}^T + \nu$, where $\nu \sim N(0, \sigma^2 \boldsymbol{I}_{\boldsymbol{\theta}})$. Our unlearning criterion 1 is therefore satisfied for $\tilde{\boldsymbol{\theta}}$ (Theorem 2), and the server can perform $\tilde{N}$ new optimization rounds with FEDAVG initialized with $\tilde{\boldsymbol{\theta}}$. Thanks to the contribution of the remaining clients in $\tilde{\boldsymbol{\theta}}$, the retraining with IFU is generally faster than retraining with a random initial model.

Since $\Psi(n, i)$ increases with n, the server can stop computing the bounded sensitivity (9) for client i whenever the condition $\Psi(n_i, i) > \Psi^*$ is verified after n_i optimization rounds. At this point, the model $\boldsymbol{\theta}^{n_i - 1}$ will be selected for the unlearning request of client i, as the models at subsequent iterations do not comply with the desired unlearning budget Ψ^*. Thus, Eq 12 does not need the entire history of the global models to be performed. Only one version of the global model must be kept for each client potentially wishing to be unlearned in latter stages. We also note that computing the bounded sensitivity (9) only requires to compute the norm of sums of vectors already computed, which can be done while the clients perform their local updates. Hence, there is no added time required to compute the bounded sensitivity.

Algorithm 3 Sequential IFU (SIFU)

LEARNING WITH FEDAVG

1: FEDAVG(I, N) initialized on initial model $\boldsymbol{\theta}_0^0$.
2: Compute $\Psi_0(n, i)$, equation (15).

UNLEARNING THE SERIES OF REQUESTS $\{W_u\}$

Require: $\{W_u\}_{u=1}^R$, ϵ, δ, σ, and $\{N_u\}_{u=1}^U$
1: Get Ψ^* with equation (11).
2: **for** u from 1 to U **do**
3: $I_u = I_{u-1} \setminus W_u$.
4: Compute (ζ_u, T_u) with $H(u)$, eq. (17) and (18).
5: The new global model is $\boldsymbol{\theta}_u^0 = \boldsymbol{\theta}_{\zeta_u}^{T_u} + N(0, \sigma^2 \boldsymbol{I}_{\boldsymbol{\theta}})$.
6: Perform FEDAVG(I_u, N_u) initialized on $\boldsymbol{\theta}_u^0$.
7: Update $H(u+1)$ with ζ_u, T_u, and $H(u)$, eq. (19).
8: Compute $\Psi_u(n, I_u)$, eq. (16).
9: **end for**

If the clients wish to employ gradient masking techniques to avoid revealing their full updates, each client can compute $\Delta_c(I, \boldsymbol{\theta})$ for themselves. Indeed, Eq. 7 can be re-written as:

$$\Delta_c(I, \boldsymbol{\theta}^n) = \frac{|D_c|}{|D_I| - |D_c|} \left\| \boldsymbol{\theta}^{n+1} - \boldsymbol{\theta}_c^{n+1} \right\| \tag{13}$$

4 Sequential FU with SIFU

In this section, we extend IFU to the sequential unlearning setting with Sequential IFU (SIFU). With Algorithm 3, SIFU is designed to satisfy a series of U unlearning requests expressed by set of indices W_u, corresponding to clients to unlearn at request $1 \leq u \leq U$. SIFU generalizes IFU for which $U = 1$ and $W_1 = \{c\}$. We provide an illustration of SIFU in Figure 1.

The notations introduced thus far need to be generalized to account for series of unlearning requests $W_1, W_2, \ldots, W_U$. Global models are now referenced by their coordinates (u, n), i.e. $\boldsymbol{\theta}_u^n$ represents the model after u unlearning requests followed by a retraining made of n aggregation steps. Hence, $\boldsymbol{\theta}_u^0$ is the initialization of the model when starting to unlearn the clients in W_u. Additionally, we define N_u as the number of server aggregations on the remaining clients required to reach the desired performance threshold (i.e. perform successful retraining). Therefore, by construction, $\boldsymbol{\theta}_u^{N_u}$ is the model obtained after using SIFU to process the sequence of unlearning requests $\{W_s\}_{s=1}^u$. Finally, we define I_u as the

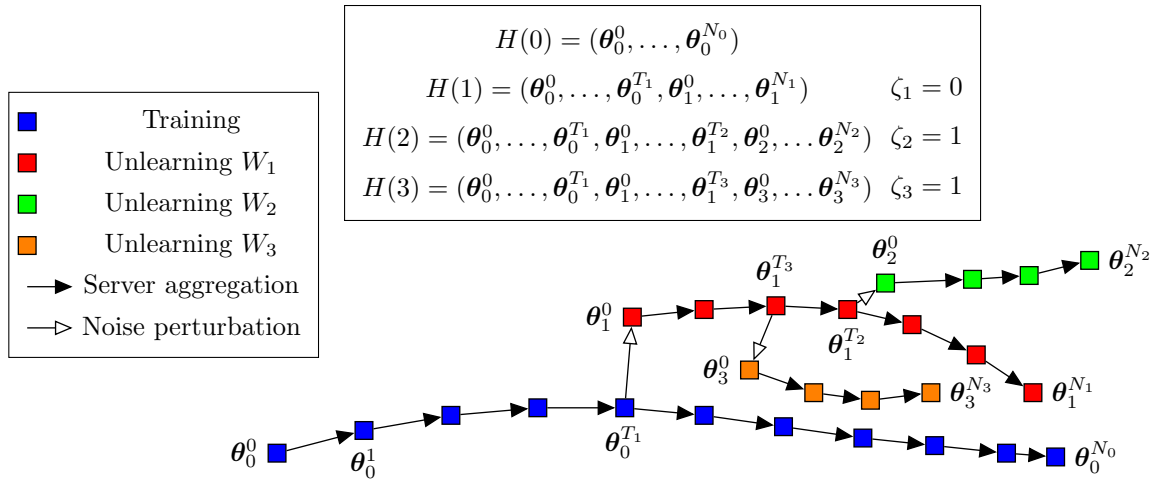

Figure 1: Illustration of SIFU (Algorithm 3) when the server receives $U = 3$ unlearning requests, through the evolution of the global model parameters $\boldsymbol{\theta}_u^n$ after server aggregation and noise perturbation. After standard federated training via FedAvg(I, N_0) the training history is $H(0) = (\boldsymbol{\theta}_0^0, \ldots, \boldsymbol{\theta}_0^{N_0})$. At request $u = 1$, the unlearning index is T_1, and the training history becomes $H(1) = (\boldsymbol{\theta}_0^0, \ldots, \boldsymbol{\theta}_0^{T_1}, \boldsymbol{\theta}_1^0, \ldots, \boldsymbol{\theta}_1^{N_1})$ with $\zeta_1 = 0$. At request $u = 2$, the unlearning index is T_2 and the training history becomes $H(2) = (\boldsymbol{\theta}_0^0, \ldots, \boldsymbol{\theta}_0^{T_1}, \boldsymbol{\theta}_1^0, \ldots, \boldsymbol{\theta}_1^{T_2}, \boldsymbol{\theta}_2^0, \ldots \boldsymbol{\theta}_2^{N_2})$ with $\zeta_2 = 1$. Finally, at request $u = 3$, the unlearning index is found at $T_3 < T_2$ in the branch of request $u = 1$. The updated training history is now $H(3) = (\boldsymbol{\theta}_0^0, \ldots, \boldsymbol{\theta}_0^{T_1}, \boldsymbol{\theta}_1^0, \ldots, \boldsymbol{\theta}_1^{T_3}, \boldsymbol{\theta}_3^0, \ldots \boldsymbol{\theta}_3^{N_3})$ with $\zeta_3 = 1$.

set of remaining clients after unlearning request u, i.e. $I_u \coloneqq I \setminus \cup_{s=1}^{u} W_s = I_{u-1} \setminus W_u$, with $I_0 = I$.

In case of multiple unlearning requests, the bounded sensitivity (9) for client i must be updated at each unlearning index u to account for the new history of global models resulting from retraining. With SIFU, the selection of the unlearning index T for a request u depends of the past history of unlearning requests. To track of the evolution of the unlearning procedure, we introduce the *model history* $H(u)$, which keeps track of each iteration of the global model across requests. Please note that this object is here introduced solely for illustration purposes, and is not actually stored when running SIFU. With reference to Figure 1, we start with the original sequence of global models obtained at each FL round, i.e. $H(0) = (\boldsymbol{\theta}_0^0, \ldots, \boldsymbol{\theta}_0^{N_0})$. Similarly to IFU, the first unlearning request requires to identify the unlearning index T_1 for which the corresponding global model $\boldsymbol{\theta}_0^{T_1}$ must be perturbed to obtain $\boldsymbol{\theta}_1^0$ and retrained until convergence, i.e. up to $\boldsymbol{\theta}_1^{N_1}$. In this case, the bounded sensitivity is computed according to equation (9).

After unlearning (i.e. Gaussian perturbation followed by re-training), the current training history is now $H(1) = (\boldsymbol{\theta}_0^0, \ldots, \boldsymbol{\theta}_0^{T_1}, \boldsymbol{\theta}_1^0, \ldots, \boldsymbol{\theta}_1^{N_1})$. More generally, we define $H(u)$ as the training history after u unlearning requests and N_u FL iterations steps from $\boldsymbol{\theta}_0^u$. We define the increment history of a given client c as the sequence obtained by computing $\Delta_c(I_k, \boldsymbol{\theta}_k^s)$ on every element of H, according to their order of appearance in the training history:

$$\Lambda_c^u = (\Delta_c(I_k, \boldsymbol{\theta}_k^s) \text{ for } \boldsymbol{\theta}_k^s \in H(u)), \tag{14}$$

The bounded sensitivity for client i should be updated to account for this new history of global models. We therefore generalize equation (9) to account for the entire training history:

$$\Psi_u(n, c) \coloneqq \sum_{s=1}^{n} B(f_I, \eta)^{\gamma_{s,n}} \cdot \Lambda_c^u[s], \tag{15}$$

where $\Lambda_c^u[s]$ is the s-th element of the sequence Λ_c^u. We extend this quantity to a set of clients S as

$$\Psi_u(n, S) \coloneqq \max_{c \in S} \Psi_u(n, c). \tag{16}$$

For a given unlearning request $u + 1$, we evaluate $\Psi_u(n, W_{u+1})$ along the clients' history and we identify the optimal parameters to initialize unlearning from:

$$\boldsymbol{\theta}_{\zeta_{u+1}}^{T_{u+1}} = H(u)[n_{u+1}], \tag{17}$$

where

$$n_{u+1} = \max\{n \ : \ \Psi_u(n, W_{u+1}) \leq \Psi^*\}. \tag{18}$$

We proceed by perturbing the parameters $\boldsymbol{\theta}_{\zeta_{u+1}}^{T_{u+1}}$ with Gaussian noise defined in Theorem 2 to obtain $\boldsymbol{\theta}_{u+1}^0$. A new FL routine is then operated with the remaining

clients to obtain parameters $\boldsymbol{\theta}_{u+1}^{N_{u+1}}$, and to update the training history as

$$H(u+1) = (\boldsymbol{\theta}_0^0, \dots, \boldsymbol{\theta}_{\zeta_{u+1}}^{T_{u+1}}, \boldsymbol{\theta}_{u+1}^0, \dots, \boldsymbol{\theta}_{u+1}^{N_{u+1}}). \quad (19)$$

As for IFU, performing SIFU requires the server to store the sensitivity associated to each client at each round, and one global model checkpoint per client. The computational cost for this operation is negligible. Theorem 3 shows that with SIFU we achieve unlearning guarantees for every client in a sequential unlearning request F_u.

Theorem 3. *The model $\boldsymbol{\theta}_u^{N_u}$ obtained with SIFU satisfies (ϵ, δ)-unlearning for every client in current and previous unlearning requests, i.e. in $F_u = \cup_{s=1}^{u} W_s$.*

Proof. See Appendix D. □

5 Experiments

In this section, we experimentally demonstrate the effectiveness of SIFU through a series of experiments introduced in Section 5.1. In Section 5.2, we illustrate and discuss our experimental results. Results and related code are publicly available at this GitHub URL.

5.1 Experimental Setup

Datasets. We report experiments on adapted versions of CIFAR-10 (Krizhevsky, 2009), CIFAR-100 (Krizhevsky, 2009), MNIST (LeCun et al., 1998), FashionMNIST (Xiao et al., 2017), and CelebA (Liu et al., 2015). For each dataset, we consider $M = 100$ clients, with 100 data points each. For MNIST and FashionMNIST, each client has data samples from only one class, so that each class is represented in 10 clients only. For CIFAR10 and CIFAR100, each client has data samples with ratio sampled from a Dirichlet distribution with parameter 0.1 (Harry Hsu et al., 2019). Finally, in CelebA, clients own data samples representing the same celebrity as done in LEAF (Caldas et al., 2018). With these five datasets, we consider different levels of heterogeneity based on labels and features distribution.

Models. For MNIST, we train a logistic regression model to consider a convex classification problem. For the other four datasets, we train a neural network with convolutional layers followed by fully connected ones. Further details are provided in Appendix E.

Unlearning schemes. We compare a variety of state-of-the-art FU schemes. First, we consider our method SIFU as described in Algorithm 3 and set $B = 1$. The choice for B is experimentally justified in Appendix F. In addition to SIFU, we consider the following unlearning schemes from the state-of-the-art: SCRATCH, where retraining of a new model is performed from scratch on the remaining clients; FINE-TUNING, where retraining is performed on the current global model with the remaining clients; DP (Dwork and Roth, 2014), where training with every client is performed with Differential Privacy, and both FEDERASER and FEDACCUM (Liu et al., 2021), where unlearning is performed by using the gradient history of clients to remove their contribution. Finally, we consider a freely adapted version of Neel et al. (2021)'s perturbed gradient descent by noising the final model with a standard deviation calculated from SIFU's theoretical analysis, which boils down to performing SIFU without the "roll-back" step.

Experimental scenario.

Since unlearning is about efficiency of information deletion, we first study the unlearning capabilities of every method under different time constraints. We limit the unlearning time allowed for each method to respectively 25%, 33% and 50% of the training time, and display the accuracy results both on the unlearned clients (forget set) and the remaining ones (retain set). Secondly, to account for the sequential unlearning setting and the adversarial case of watermarked data, we study the scenario proposed by (Sommer et al., 2020) in Section 5.3. Each unlearning method is applied with the same hyperparameters, i.e. local learning rate η, amount of SGD steps K and optimizer (Appendix E).

Unlearning quantification. We verify the success of a FU scheme by evaluating its running time and difference in accuracy with SCRATCH on the retain and forget set. When studying methods under a time constraint (Section 5.2), we compute: (a) The accuracy difference on the forget set, thus evaluating unlearning quality, and (b) the accuracy difference on the retain set, thus evaluating retained utility. When studying methods under a utility constraint (Section 5.3), we compute (a) The accuracy difference on the forget set, and (b) the number of required iterations to reach the fixed utility threshold. The differences are computed with respect to the performance of SCRATCH to identify the method associated with similar unlearning properties and reduced computation time.

5.2 Experimental Results

The results for the first experimental setting described in Sec. 5.1 are available in Fig 2. SIFU outperforms its competitors on 4 out of 5 datasets. In particular, on every experiment involving FU of a neural network, SIFU is the only method to achieve a good trade-off between forgetting quality and accuracy on the retain set, while other methods fail either by lack of forgetting

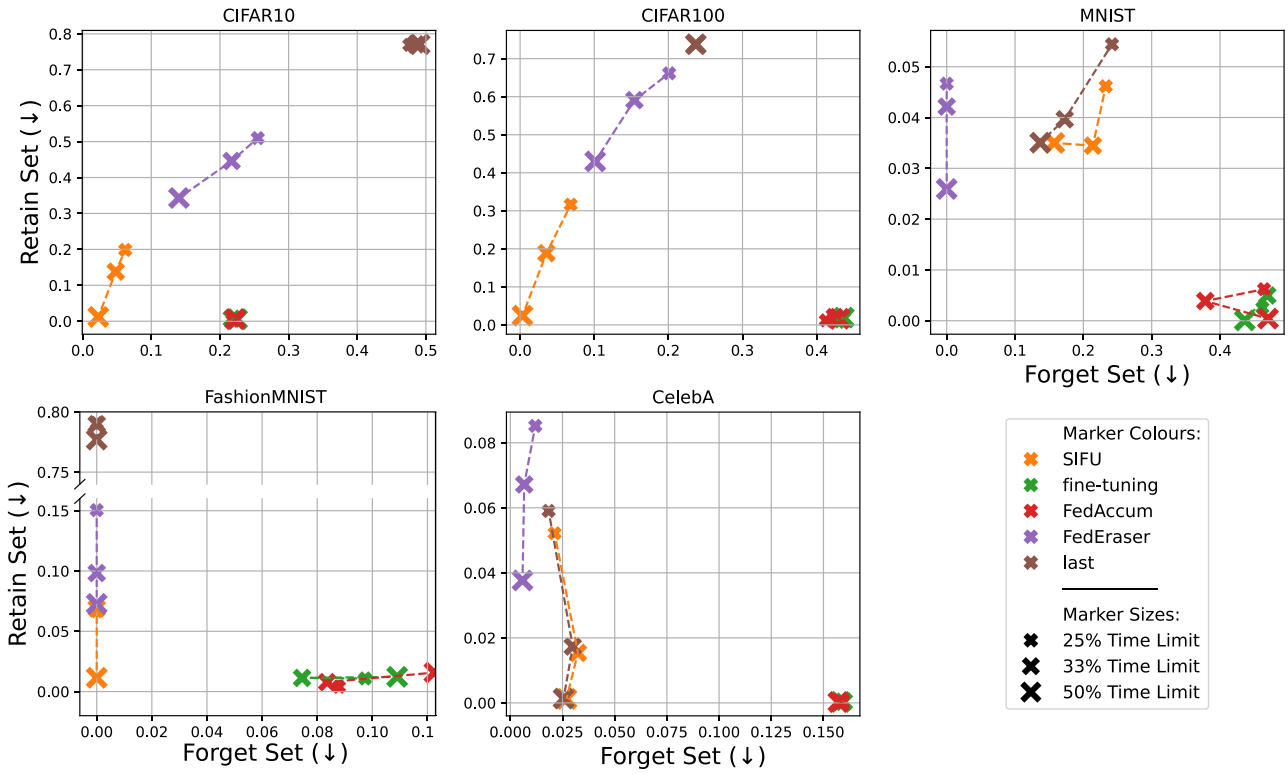

Figure 2: Difference in accuracy (absolute value) between SCRATCH and the considered unlearning methods, on both retain and forget sets (lower is better).

quality (Fine-Tuning, FedAccum) or by low retain set accuracy (FedEraser, LAST).

It is interesting to notice that while LAST performs quite well on very simple datasets such as CelebA and MNIST, the introduced noise becomes too large in more complex datasets such as CIFAR10(0), rendering it infeasible for the model to converge after noising. This motivates our method and underlines the importance of the "roll-back" step in SIFU. While FedEraser tends to be a good performer in terms of retain set accuracy, the imposed unlearning time limit forbids it from recovering satisfying utility on the retain set, as compared to SIFU. Thus, one can wonder whether the method would perform better if provided with more time. We answer this question in Section 5.3. Finally, we excluded some methods when their poor performances would hinder the figure's readability: DP was removed from Figures 2 and 3, and LAST from Figure 3. See Appendix E for DP performances.

5.3 Verifying Unlearning with Watermarking

The work of (Sommer et al., 2020) proposes an adversarial approach to verify the unlearning efficiency through watermarking. We follow this approach by applying watermarking to each client's data by randomly assigning the maximum possible value to 10 given pixels of each data sample. To ensure that clients' heterogeneity is only due to the modification of the pixels intensities, we define data partitioning across clients by randomly assigning the data according to a Dirichlet distribution with parameter $\alpha = 1$.

We consider a sequential unlearning scenario in which the server performs FL training and then receives $U = 3$ sequential unlearning requests to unlearn 10 random clients per request. In the special case of MNIST and FashionMNIST, the server must unlearn 10 clients owning the same class. The server performs unlearning with each method, before fine-tuning the obtained model on the remaining data until the global model accuracy on the remaining clients exceeds a fixed threshold specific to each dataset, namely: 93% for MNIST, 99.9% for CelebA, and 90% for FashionMNIST, CIFAR10 and CIFAR100. We impose a minimum of 50 FL aggregation rounds, and a maximum of 10000 rounds when the stopping accuracy threshold is not reached.

Figure 3 shows our results for this experimental scenario. On every dataset where a CNN is used, SIFU

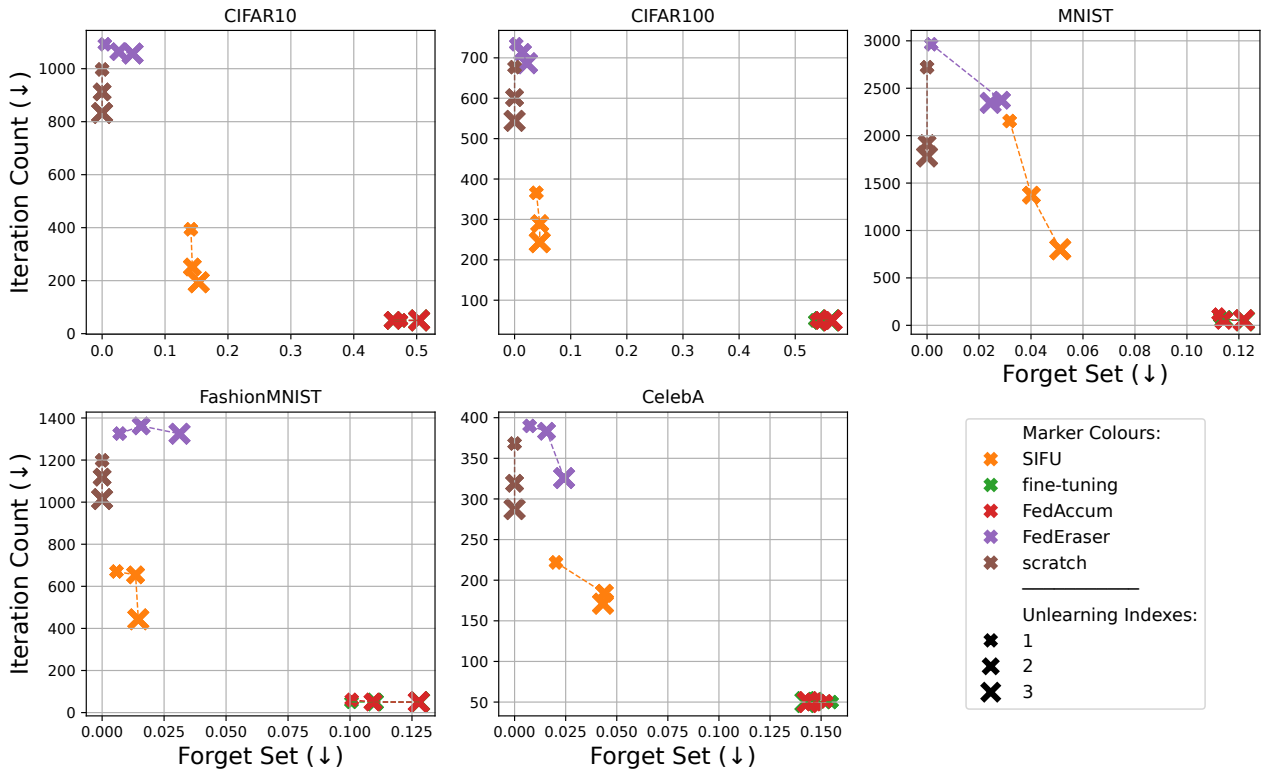

Figure 3: Total amount of aggregation rounds (1st row) and model accuracy of unlearned clients (2nd row) for the unlearning of watermarked data from MNIST, FashionMNIST, CIFAR10, CIFAR100, and CelebA (the lower the better).

outperforms every other method, confirming the conclusions drawn from Figure 2. Indeed, it offers a unique trade-off between efficiency and unlearning, while FedEraser provides satisfying unlearning but is even more costly than Scratch in terms of iteration count, thus rendering the method of limited interest in our experimental scenario.

5.4 Verifying the consistency of SIFU.

We provide additional experiments in Appendix E assessing the unlearning results depending on varying training conditions, including the choice of the clients to be unlearnt, and the amplitude of the model perturbation. Our findings are consistent in showing that SIFU is the best performing method in terms of computational efficiency and unlearning capabilities. In particular, we note that when unlearning with low (resp. high) values of σ, SIFU has identical behavior to Scratch (resp. Last), as the unlearning is applied to the initial model $\boldsymbol{\theta}_0^0$ (resp. final $\boldsymbol{\theta}_u^{N_u}$). Moreover, independently from the chosen batch of clients to be unlearnt, Supplementary Figure 6 shows that SIFU consistently leads to effective unlearning with lower computational cost as compared to Scratch.

6 Conclusions

In this work, we introduce SIFU, a general FU scheme allowing unlearning of clients contributions from a model trained with FedAvg. Upon receiving an unlearning request from a given client, SIFU identifies the optimal FL iteration from which to re-initialise the optimisation. We prove that SIFU accounts for sequences of requests while satisfying the unlearning guarantees. SIFU is scalable with respect to model size and FL iterations, and generalizes beyond the convex assumption on the local loss functions, thus relaxing the strong assumptions typically adopted in the MU literature.

A further contribution of this work consists in a new theory for bounding the clients contribution in FL, which can be computed by the server without major overhead, and no additional communication nor computation on the client side.

7 Acknowledgements

This work was supported by the French government managed by the Agence Nationale de la Recherche (ANR) through France 2030 program with the reference ANR-23-PEIA-005 (REDEEM project), and through the Franco-German research program with reference ANR-22-FAI1-0003 (TRAIN project). It was also funded in part by the Groupe La Poste, sponsor of the Inria Foundation, in the framework of the FedMalin Inria Challenge.

References

Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C. A., Jia, H., Travers, A., Zhang, B., Lie, D., and Papernot, N. (2021). Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE.

Caldas, S., Duddu, S. M. K., Wu, P., Li, T., Konečný, J., McMahan, H. B., Smith, V., and Talwalkar, A. (2018). LEAF: A Benchmark for Federated Settings. *NeurIPS*.

Cao, Y. and Yang, J. (2015). Towards making systems forget with machine unlearning. *2015 IEEE Symposium on Security and Privacy*, pages 463–480.

Chen, X., Wu, S. Z., and Hong, M. (2020). Understanding gradient clipping in private sgd: A geometric perspective. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 13773–13782. Curran Associates, Inc.

Dwork, C. and Roth, A. (2014). The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407.

Geng, J., Mou, Y., Li, Q., Li, F., Beyan, O., Decker, S., and Rong, C. (2023). Improved gradient inversion attacks and defenses in federated learning. *IEEE Transactions on Big Data*.

Ginart, A., Guan, M., Valiant, G., and Zou, J. Y. (2019). Making ai forget you: Data deletion in machine learning. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Golatkar, A., Achille, A., Ravichandran, A., Polito, M., and Soatto, S. (2021). Mixed-privacy forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 792–801.

Golatkar, A., Achille, A., and Soatto, S. (2020a). Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Golatkar, A., Achille, A., and Soatto, S. (2020b). Forgetting outside the box: Scrubbing deep networks of information accessible from input-output observations.

Gong, J., Kang, J., Simeone, O., and Kassab, R. (2022). Forget-svgd: Particle-based bayesian federated unlearning. In *2022 IEEE Data Science and Learning Workshop (DSLW)*, pages 1–6. IEEE.

Guo, C., Goldstein, T., Hannun, A., and Van Der Maaten, L. (2020). Certified data removal from machine learning models. In III, H. D. and Singh, A., editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3832–3842. PMLR.

Gupta, V., Jung, C., Neel, S., Roth, A., Sharifi-Malvajerdi, S., and Waites, C. (2021). Adaptive machine unlearning. In Ranzato, M., Beygelzimer, A., Nguyen, K., Liang, P. S., Vaughan, J. W., and Dauphin, Y., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 16319–16330. Curran Associates, Inc.

Halimi, A., Kadhe, S., Rawat, A., and Baracaldo, N. (2022). Federated unlearning: How to efficiently erase a client in fl? *arXiv preprint arXiv:2207.05521*.

Harding, E. L., Vanto, J. J., Clark, R., Hannah Ji, L., and Ainsworth, S. C. (2019). Understanding the scope and impact of the california consumer privacy act of 2018. *Journal of Data Protection & Privacy*, 2(3):234–253.

Hardt, M., Recht, B., and Singer, Y. (2016). Train faster, generalize better: Stability of stochastic gradient descent. In *International conference on machine learning*, pages 1225–1234. PMLR.

Harry Hsu, T. M., Qi, H., and Brown, M. (2019). Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*.

Izzo, Z., Anne Smart, M., Chaudhuri, K., and Zou, J. (2021). Approximate data deletion from machine learning models. In Banerjee, A. and Fukumizu, K., editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 2008–2016. PMLR.

Jin, R., Chen, M., Zhang, Q., and Li, X. (2023). Forgettable federated linear learning with certified data removal. *arXiv preprint arXiv:2306.02216*.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images.

LeCun, Y., Bottou, L., Bengio, Y., and Ha, P. (1998). LeNet. *Proceedings of the IEEE*.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. (2020). On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*.

Li, Y., Wang, C., and Cheng, G. (2021). Online forgetting process for linear regression models. In Banerjee, A. and Fukumizu, K., editors, *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*, volume 130 of *Proceedings of Machine Learning Research*, pages 217–225. PMLR.

Liu, G., Ma, X., Yang, Y., Wang, C., and Liu, J. (2021). Federaser: Enabling efficient client-level data removal from federated learning models. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pages 1–10.

Liu, Y., Xu, L., Yuan, X., Wang, C., and Li, B. (2022). The right to be forgotten in federated learning: An efficient realization with rapid retraining. *Proceedings - IEEE INFOCOM*, 2022-May:1749–1758.

Liu, Z., Luo, P., Wang, X., and Tang, X. (2015). Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Mahadevan, A. and Mathioudakis, M. (2021). Certifiable machine unlearning for linear models. *arXiv preprint arXiv:2106.15093*.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. In Singh, A. and Zhu, J., editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282, Fort Lauderdale, FL, USA. PMLR.

Neel, S., Roth, A., and Sharifi-Malvajerdi, S. (2021). Descent-to-delete: Gradient-based methods for machine unlearning. In Feldman, V., Ligett, K., and Sabato, S., editors, *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pages 931–962. PMLR.

Pan, C., Sima, J., Prakash, S., Rana, V., and Milenkovic, O. (2023). Machine unlearning of federated clusters. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Sommer, D. M., Song, L., Wagh, S., and Mittal, P. (2020). Towards probabilistic verification of machine unlearning. *arXiv preprint arXiv:2003.04247*, abs/2003.04247.

Voigt, P. and Von dem Bussche, A. (2017). The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555.

Wang, J., Guo, S., Xie, X., and Qi, H. (2022). Federated unlearning via class-discriminative pruning.

Wang, J., Liu, Q., Liang, H., Joshi, G., and Poor, H. V. (2020). Tackling the objective inconsistency problem in heterogeneous federated optimization. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Wang, W., Tian, Z., Zhang, C., Liu, A., and Yu, S. (2023). Bfu: Bayesian federated unlearning with parameter self-sharing. In *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*, ASIA CCS '23, page 567–578, New York, NY, USA. Association for Computing Machinery.

Wu, C., Zhu, S., and Mitra, P. (2022). Federated unlearning with knowledge distillation. *arXiv preprint arXiv:2201.09441*.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, abs/1708.07747.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]

(c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A When fine tuning does not guarantee unlearning: example on linear regression

Let us consider a linear regression optimization, with feature matrix $\boldsymbol{X}$ and predictions $\boldsymbol{y}$ such that the loss function f is defined as

$$f(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\theta}) = \frac{1}{2} \left[\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\right]^T \left[\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\right]. \tag{20}$$

In this example, we assume there are more features than data samples, which makes $\boldsymbol{X}^T\boldsymbol{X}$ a singular matrix. While f is convex, f has more than one global optimum. Any model with parameter $\boldsymbol{\theta}^*$ such that

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}^* = \boldsymbol{X}^T\boldsymbol{y} \tag{21}$$

is a global optimum. When $\boldsymbol{X}^T\boldsymbol{X}$ is non-singular, we retrieve the unique optimum in close-form $\boldsymbol{\theta}^* = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$. We show with this simple example that, upon unlearning a data sample, no amount of fine-tuning on the model $\boldsymbol{\theta}^*$ can lead to the same model obtained when retraining from a random initial model. We differentiate between $(\boldsymbol{X}, \boldsymbol{y})$ and $(\boldsymbol{X}_{-1}, \boldsymbol{y}_{-1})$ our data with and without a given data point.

Optimizing f, as defined in equation (20), with N steps of gradient descent, learning rate η, and initial model $\boldsymbol{\theta}_0$ gives model parameters $\boldsymbol{\theta}^N$ defined as

$$\boldsymbol{\theta}^N = \underbrace{\left[I - \eta\boldsymbol{X}^T\boldsymbol{X}\right]^N}_{A(\boldsymbol{X},N)}\boldsymbol{\theta}^0 + \underbrace{\eta\sum_{n=0}^{N-1}\left[I - \eta\boldsymbol{X}^T\boldsymbol{X}\right]^n\boldsymbol{X}^T\boldsymbol{y}}_{B(\boldsymbol{X},\boldsymbol{y},N)}. \tag{22}$$

We first note that we retrieve the standard form for the global optimum of linear regression when $\boldsymbol{X}^T\boldsymbol{X}$ is non-singular as $\lim_{n\to\infty} A(\boldsymbol{X}, n) = 0$ and $\lim_{n\to\infty} B(\boldsymbol{X}, \boldsymbol{y}, n) = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$. In the general form accounting for the singular case, at least one eigenvalue of $A(\boldsymbol{X}, N)$ is equal to 1 independently from the amount of gradient descent steps N. Hence, the parameters of the model obtained with gradient descent optimization always depend from the ones of the initial model $\boldsymbol{\theta}^0$. Hence, when unlearning our data sample from $\boldsymbol{\theta}^N$, the resulting trained model still depends of that data samples. Indeed, if we compare the model $\boldsymbol{\theta}_{-1}^{\tilde{N}}$ trained on the data samples $(\boldsymbol{X}_{-1}, \boldsymbol{y}_{-1})$, to the model $\phi_{-1}^{\tilde{N}}$ obtained after fine-tuning the model $\boldsymbol{\theta}^N$ with $\tilde{N}$ server aggregations, we have

$$\phi_{-1}^{\tilde{N}} - \boldsymbol{\theta}_{-1}^{\tilde{N}} = A(\boldsymbol{X}_{-1}, \tilde{N})A(\boldsymbol{X}, N)\boldsymbol{\theta}^0 + A(\boldsymbol{X}_{-1}, \tilde{N})B(\boldsymbol{X}, \boldsymbol{y}, N). \tag{23}$$

B Forgetting a Single Client with IFU, Theorem 1

In this section, we provide the proof of Theorem 1 and derive 3 different results when considering 3 different sets of assumptions: f is smooth, f is smooth and convex, and f is smooth and strongly-convex.

B.1 Definitions

We define by $\boldsymbol{\theta}^N = \textsc{FedAvg}(I, N)$ and $\boldsymbol{\phi}^N = \textsc{FedAvg}(I_{-c}, N)$ the models trained with FedAvg initialized at $\boldsymbol{\theta}_0$ with respectively all the clients, i.e. I, and all the clients but client c, i.e. I_{-c}, performing K GD steps.

When clients perform $K = 1$ GD steps, two consecutive global models can be related, when training with clients in I as a simple GD step, i.e.

$$\boldsymbol{\theta}^{n+1} = \boldsymbol{\theta}^n - \eta\nabla f_I(\boldsymbol{\theta}^n). \tag{24}$$

Let us define the gradient step operator for function f at learning rate η:

$$G(f, \eta, \boldsymbol{\theta}) = \boldsymbol{\theta} - \eta\nabla f(\boldsymbol{\theta})$$

B.2 General case

B.2.1 Main observation

The following results use Lemma 3.7 of (Hardt et al., 2016) under its 3 possible hypothesis. Let us first notice that, with $K = 1$ and without any hypothesis on f besides its differentiability, we have:

$$\begin{aligned}\phi^{n+1} - \theta^{n+1} &= \phi^n - \theta^n - \eta[\nabla f_{I\setminus\{c\}}(\phi^n) \\ &\quad - \nabla f_{I\setminus\{c\}}(\theta^n) + \nabla f_{I\setminus\{c\}}(\theta^n) - \nabla f_I(\theta^n)] \\ &= G(f_{I\setminus\{c\}}, \eta, \phi^n) - G(f_{I\setminus\{c\}}, \eta, \theta^n) \\ &\quad + \eta(\nabla f_{I\setminus\{c\}}(\theta^n) - \nabla f_I(\theta^n))\end{aligned}$$

Then, depending on the assumptions made on f, we get 3 different results, all taking the same form:

$$\begin{aligned}\|\phi^{n+1} - \theta^{n+1}\| &\leq B(f, \eta)\|\phi^n - \theta^n\| \\ &\quad + \eta\|\nabla f_{I\setminus\{c\}}(\theta^n) - \nabla f_I(\theta^n)\|\end{aligned} \tag{25}$$

Where we consider 3 distinct cases, each with their respective assumptions and definition of B:

1. If f_i is β-smooth for every $i \in I$, then
$$B(f_I, \eta) = 1 + \eta.\beta \tag{26}$$
2. If f_i is β-smooth and convex for every $i \in I$ and $\eta \leq 2/\beta$, then
$$B(f, \eta) = 1 \tag{27}$$

3. If f_i is β-smooth and μ-strongly-convex for every $i \in I$ and $\eta \leq \frac{2}{\beta+\mu}$, then

$$B(f,\eta) = 1 - \frac{\eta\beta\mu}{\beta+\mu} \tag{28}$$

B.2.2 Generic proof

Let us prove the desired results with a generic function B. The specific results in the 3 different cases will then be derived directly by specifying B depending on the hypothesis.

Let $p_i = \frac{N_i}{N_I}$ and $q_i = \frac{p_i}{1-p_c} \cdot (1 - \mathbb{1}_c(i))$. Then,

$$\begin{aligned}
\|\phi^{n+1} - \theta^{n+1}\| &= \Big\| \sum_{i=1}^{M} q_i \phi_i^{n+1} - \sum_{i=1}^{M} p_i \theta_i^{n+1} \Big\| \\
&= \Big\| \sum_{i=1}^{M} q_i (\phi_i^{n+1} - \theta_i^{n+1}) \\
&\quad + \underbrace{\sum_{i=1}^{M} q_i (\theta_i^{n+1} - \theta^n) - \sum_{i=1}^{M} p_i (\theta_i^{n+1} - \theta^n)}_{\Delta_c(I,\theta^n)} \\
&\leq \sum_{i=1}^{M} q_i \|\phi_i^{n+1} - \theta_i^{n+1}\| + \Delta_c(I,\theta^n) \\
&\leq \max_{i \in I} \|\phi_i^{n+1} - \theta_i^{n+1}\| + \Delta_c(I,\theta^n)
\end{aligned}$$

where the last inequality follows from the fact that $\sum q_i = 1$.

Now, for any $i \in I$, let us give an upper bound for $\|\phi_i^{n+1} - \theta_i^{n+1}\|$:

$$\begin{aligned}
\|\phi_i^{n,k+1} - \theta_i^{n,k+1}\| &= \|G(f_{\{i\}}, \eta, \phi_i^{n,k}) - G(f_{\{i\}}, \eta, \theta_i^{n,k})\| \\
&\leq B(f,\eta) \cdot \|\phi_i^{n,k} - \theta_i^{n,k}\|
\end{aligned} \tag{29}$$

where the last inequality follows from the contractivity of one step of gradient descent (see Lemma 3.7 of Hardt et al. (2016) for all 3 cases). By applying this equation recursively K times, we get

$$\|\phi_i^{n+1} - \theta_i^{n+1}\| \leq B(f,\eta)^K \cdot \|\phi_i^n - \theta_i^n\| \tag{30}$$

From Eq. (29) and Eq. (30), we get:

$$\|\phi^{n+1} - \theta^{n+1}\| \leq B(f,\eta)^K \cdot \|\phi^n - \theta^n\| + \Delta_c(I,\theta^n) \tag{31}$$

Now, let us prove via recurrence that:

$$\|\phi^n - \theta^n\| \leq \sum_{p=0}^{n-1} B(f,\eta)^{(n-p-1)K} \cdot \Delta_c(I,\theta^p) \tag{32}$$

The initialization is trivial since $\phi^0 = \theta^0$.

We provide the proof of the recurring property in equation (33), thus verifying equation (32) and proving Theorem 1.

$$\begin{aligned}
\|\phi^{n+1} - \theta^{n+1}\| &\leq B(f,\eta)^K \cdot \\
&\quad \left[\sum_{p=0}^{n-1} B(f,\eta)^{(n-p-1)K} \Delta_c(I,\theta^p) \right] \\
&\quad + \Delta_c(I,\theta^n) \\
&\leq \sum_{p=0}^{(n+1)-1} B(f,\eta)^{(n+1-p-1)K} \cdot \Delta_c(I,\theta^p)
\end{aligned} \tag{33}$$

Let us now give the specific formulations for each set of assumptions.

B.3 Case f smooth, not necessarily convex

Let f be β-smooth. In this case, the result from Eq. (32) applies with B as described in Eq. (26). Therefore:

$$\|\phi^n - \theta^n\| \leq \sum_{p=0}^{n-1} (1+\eta\beta)^{(n-p-1)K} \cdot \Delta_c(I,\theta^p) \tag{34}$$

B.4 Case f smooth & convex

Let f be convex and β-smooth, let $\eta \leq \frac{2}{\beta}$. In this case, the result from Eq. (32) applies with B as described in Eq. (27). Therefore:

$$\|\phi^n - \theta^n\| \leq \sum_{p=0}^{n-1} \Delta_c(I,\theta^p) \tag{35}$$

B.5 Case f smooth & strongly convex

Let f be μ-strongly convex and β-smooth, let $\eta \leq \frac{2}{\beta+\mu}$. In this case, the result from Eq. (32) applies with B as described in Eq. (28). Therefore:

$$\|\phi^n - \theta^n\| \leq \sum_{p=0}^{n-1} \left(1 - \frac{\eta\beta\mu}{\beta+\mu}\right)^{(n-p-1)K} \cdot \Delta_c(I,\theta^p) \tag{36}$$

C Unlearning certification, Proof of Theorem 2

Proof. According to theorem A.1 of (Dwork and Roth, 2014), the Gaussian mechanism with sensitivity α, is (ϵ,δ)-Differentially Private if its noise parameter σ verifies :

$$\sigma > \left[2\left(\ln(1.25) - \ln(\delta)\right)\right]^{1/2} \epsilon^{-1} \alpha. \tag{37}$$

In our case, the sensitivity with respect to any client c at step n is $\alpha(n,c)$ by construction. Additionally, Theorem 1 provides us with the bound

$$\alpha(n,c) \leq \Psi(n,c). \tag{38}$$

Thus, picking any noise $\sigma(n,c)$ such that

$$\sigma(n,c) > \left[2\left(\ln(1.25) - \ln(\delta)\right)\right]^{1/2} \epsilon^{-1} \Psi(n,c) \tag{39}$$

ensures the differential privacy of our forgetting mechanism with respect to the unlearned clients. This concludes the proof. □

D Convergence of SIFU, Theorem 3

D.1 Proof of Theorem 3

Proof. We prove by induction that $\boldsymbol{\theta}^0_{u+1}$ (ϵ,δ)-unlearns every client in $F_{u+1} = \cup_{s=1}^{u+1} W_s = F_u \cup W_{u+1}$. The initialization $(u=1)$ directly follows from IFU, Algorithm 2, with Theorem 2. With reference to equation (17), we now assume that for every unlearning request $u \leq u$, the perturbed model $\boldsymbol{\theta}^0_u$ (ϵ,δ)-unlearns every client in F_u, and prove that $\boldsymbol{\theta}^0_{u+1}$ (ϵ,δ)-unlearns every client in F_{u+1}.

- Case 1: $\forall u \leq u$, $n_u \leq n_{u+1}$. The model $\boldsymbol{\theta}^{T_{u+1}}_{\zeta_{u+1}} = H(u)[n_{u+1}]$ appears later in the training history than models $\boldsymbol{\theta}^{T_u}_{\zeta_u}$ and, thanks to the induction property, provides (ϵ,δ)-unlearning of every client in F_u. Thus, the model $\boldsymbol{\theta}^0_{u+1}$ guarantees the unlearning of every client in F_{u+1}.

- Case 2: $\exists$ unlearning request $u^* \leq u$ such that $n_{u+1} < n_{u*}$. By construction of the training history, the sequence $H(u^*)$ contains the model $\boldsymbol{\theta}^{T_{u+1}}_{\zeta_{u+1}} = H(u^*)[n_{u+1}] = H(u)[n_{u+1}]$, which appears earlier than model $\boldsymbol{\theta}^{T_{u*}}_{\zeta_{u^*}} = H(u^*)[n_{u^*}]$. Perturbing the model $\boldsymbol{\theta}^{T_{u+1}}_{\zeta_{u+1}}$ with noise $N(\mathbf{0}, \sigma \boldsymbol{I}_{\boldsymbol{\theta}})$, guarantees (ϵ,δ)-unlearning of the clients in W_{u^*}, since

  $$\Psi_{u^*}(n_{u+1}, c^*) \leq \Psi_{u^*}(n_{u^*}, c^*) \leq \Psi^*,$$

  for every client c^* in W_{u*}. By extending this reasoning to all learning requests u such that $n_{u+1} < n_u$, and by the induction property for the remaining ones, the model $\boldsymbol{\theta}^0_{u+1}$ guarantees the unlearning of every client in F_{u+1}.

□

E Experiments

For every benchmark, we consider the number of SGD steps K, batch size B, number of clients M, the number of sampled clients m, the standard deviation σ of the noise perturbation, and the local learning rate η given in Table 2. Also, for our unlearning scheme SIFU, DP, and LAST, we consider an unlearning budget of $\epsilon = 10$ and $\delta = 0.01$. The unlearning budget plays the important role of identifying in the training history the global model to perturb. Theorem 2 shows that ϵ and σ are linearly related. Hence, to unlearn a client c from a global model c, a smaller σ can be considered, but at the cost of a higher unlearning budget (ϵ,δ), Definition 1. Also, for fair comparison of DP with other FU schemes, we select the best clipping value C, in a range from 0.001 to 1, for which the global model reaches the target accuracy in the smallest amount of aggregation rounds. Finally, for FashionMNIST, CIFAR10, CIFAR100, and CelebA, we consider model architectures composed of three convolutional layers followed by two fully connected layers, with implementation at this GitHub url.

Table 2: Hyperparameters used for our different unlearning benchmarks described in Section 5.1.

Dataset	K	B	M	m	σ	η	C
CIFAR10	5	20	100	5	0.05	0.01	0.2
CIFAR100	5	20	100	5	0.05	0.02	0.2
MNIST	10	100	100	10	0.05	0.01	0.5
FMNIST	5	20	100	10	0.1	0.02	0.5
CelebA	10	20	100	20	0.1	0.01	0.5

The training and retraining depends on the initial model $\boldsymbol{\theta}^0_0$ and the clients' batches of data used at every aggregation to compute their local SGDs. Hence, we replicate each unlearning scenario on 10 different seeds and plot in Figure 4 to 6 their averaged results. For the unlearning benchmarks described in Section 5.1 and used in Figure 4, to 6, the stopping accuracies considered are 93% for MNIST, 90% for FashionMNIST, CIFAR10, and CIFAR100, and 99.9% for CelebA.

We provide several figures to further the experimental evaluation of our method.

We define the set of clients requesting unlearning as:

$$F_u = \cup_{s=1}^{u} W_s. \tag{40}$$

In our experimental scenario, we have $|F_0| = 0$ during training, and $|F_1| = 10$, $|F_2| = 20$, and $|F_3| = 30$ after each unlearning request. We consider this setting both within an usual and an adversarial scenario with backdoored data (as proposed in (Sommer et al., 2020)). In

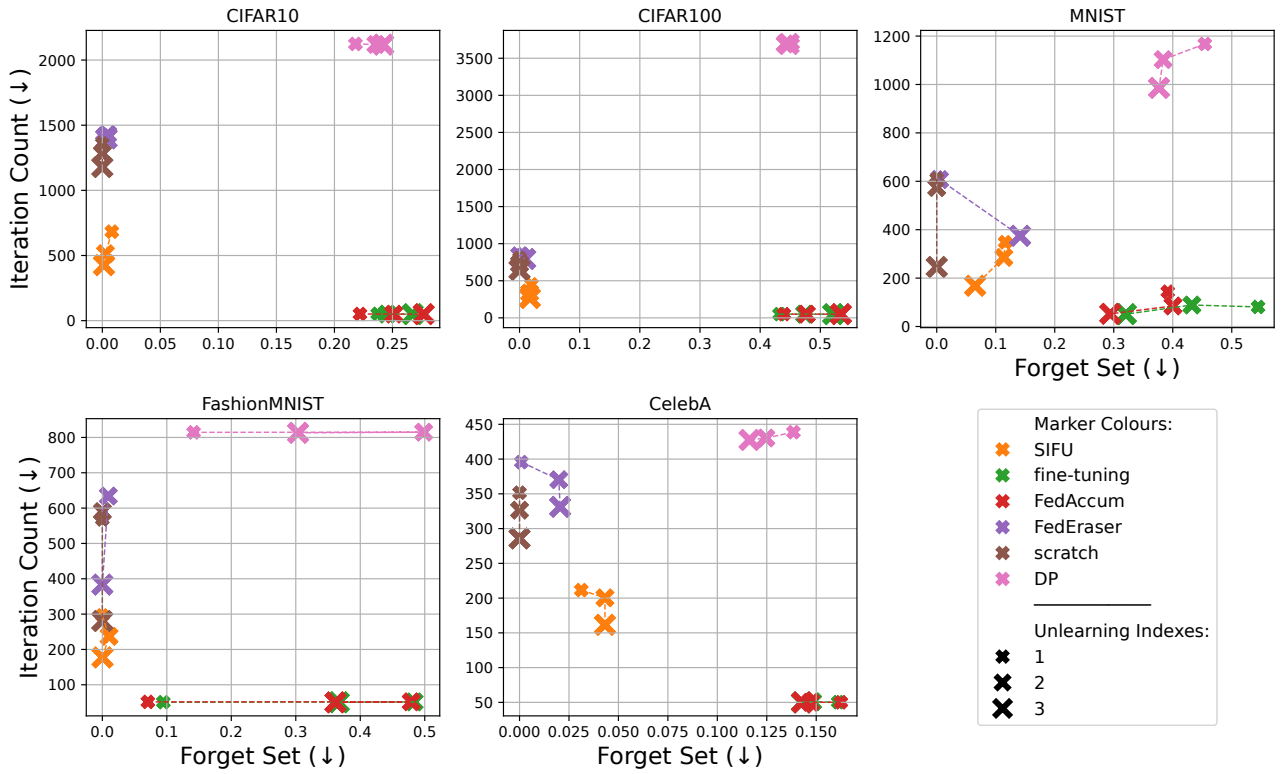

Figure 4: Total amount of aggregation rounds (1st row) and model accuracy of unlearned clients (2nd row) for MNIST, FashionMNIST, CIFAR10, CIFAR100, and CelebA (the lower the better). The server runs a federated routine with $M = 100$ clients, and unlearns 10 of them at each unlearning request ($U = 3$). Results are reported with variability estimated on 10 seeds.

Figure 3 and 4, we compare the performances of SIFU with other methods from the literature. Rather than operating within a limited time budget, we fine-tune after the unlearning until a certain accuracy threshold is reached. Thus, all compared methods have equivalent performances on the retain set and the evaluated quantities are now forget set accuracy and unlearning (+ fine-tuning) budge, as described in Sec 5.1. Figure 4 illustrates the computational cost (1st row), and unlearning capabilities (2ns row) of the tested FU methods across dataset. We note that SIFU requires a sensibly lower number of iterations than SCRATCH (52% faster on average) to achieve similar unlearning performances. FEDERASER also provides comparable unlearning capabilities, while however requiring a higher number of iterations than SCRATCH (12% on average). The other approaches are generally associated with poor unlearning results, independently from the required computational cost. We notice that the model accuracy of SIFU is slightly higher than the one of SCRATCH, with overlap only for FashionMNIST. This behavior is natural and can be explained by the privacy budget (ϵ, δ) trading unlearning capabilities for retraining cost. With the highest unlearning budget, i.e. $\epsilon = 1$ and $\delta = 0$, SIFU would require to retrain from the initial model $\boldsymbol{\theta}_0^0$, thus reducing to SCRATCH.

The poor unlearning performance of DP can be explained by the fact that it provides privacy guarantees with respect to every client, while FU only aims at removing the contribution of a few specific clients.

As observed previously, only SIFU and FEDERASER provide satisfactory unlearning, but SIFU is significantly faster than its counterpart: FEDERASER is even slower than SCRATCH and is thus not a relevant unlearning method in our experimental scenario.

Finally, when unlearning with LAST, we observed that the model always converged to a local optimum with accuracy inferior to our target. This behavior is likely due to the magnitude of the noise being added to guarantee unlearning. Hence, we decided to exclude LAST from Figure 4.

Experiments were performed using a 1080TI GPU from Nvidia.

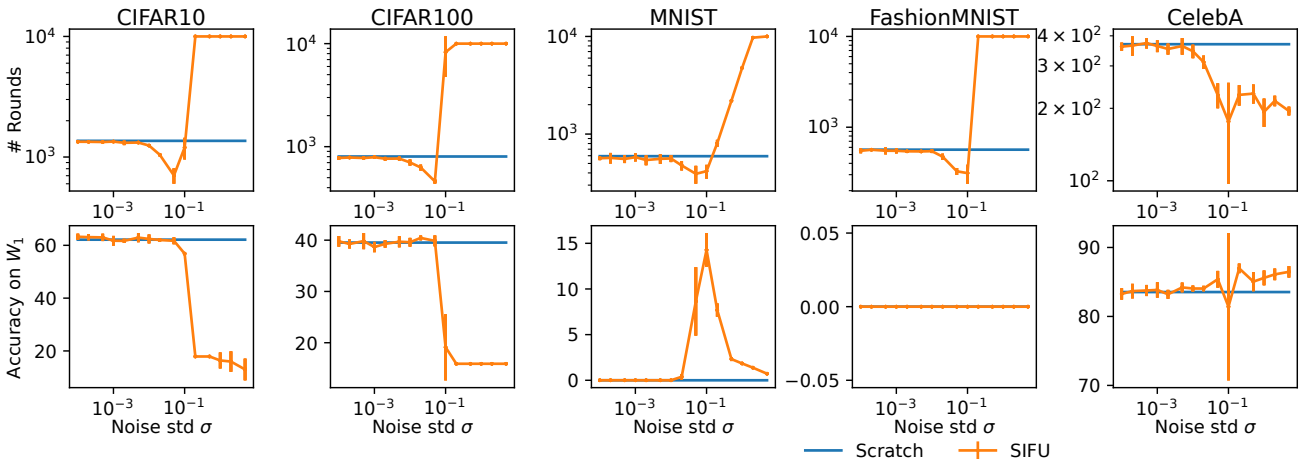

Figure 5: Impact of the noise standard deviation σ when unlearning with SIFU for the unlearning budget $(\epsilon, \delta) = (10, 0.01)$. Total amount of aggregation rounds (1^{st} row) and model accuracy of unlearned clients (2^{nd} row) for MNIST, FashionMNIST, CIFAR10, CIFAR100, and CelebA (the lower the better). Speed-ups at optimal sigma are between two-fold and five-fold.

F Bound from theorem 1

To investigate whether it is legitimate to pick $B(f, \eta) = 1$ for our experiments, we empirically measure α and Ψ for all our datasets with a large set of hyper-parameter choices, ranging from the ones used in the experiments to less adapted ones, even included some that do not allow convergence. The bound from Theorem 1 holds for every experimental scenario we tried. The results are displayed in figure 7.

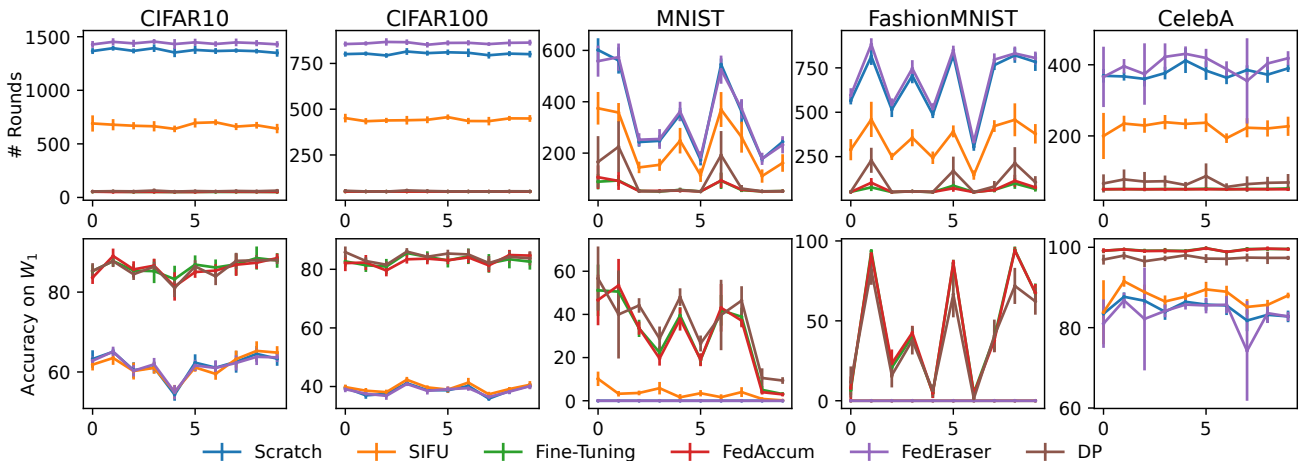

Figure 6: Total amount of aggregation rounds (1st row) and model accuracy of unlearned clients (2nd row) for MNIST, FashionMNIST, CIFAR10, CIFAR100, and CelebA (the lower the better). This figure displays the unlearning capabilities of the unlearning benchmarks introduced in Section 5.1 after training on clients in I and unlearning $|W_1| = 10$ clients. For each integer on the x-axis, a different set of clients to unlearn is considered. Each unlearning request is composed of 10 random clients for CIFAR10, CIFAR100, and CelebA. For MNIST and FashionMNIST, each unlearning request $|W_1|$ has 10 clients of the same class such that the x-axis is the class forgotten. The integers on the x-axis corresponds to the class of the clients to unlearn. We retrieve the same conclusions made in 4: SIFU is the only unlearning method offering an unlearning speed-up while maintaining an accuracy close to SCRATCH on unlearned clients. DP seems very fast since the displayed number of rounds does not include the initial training.

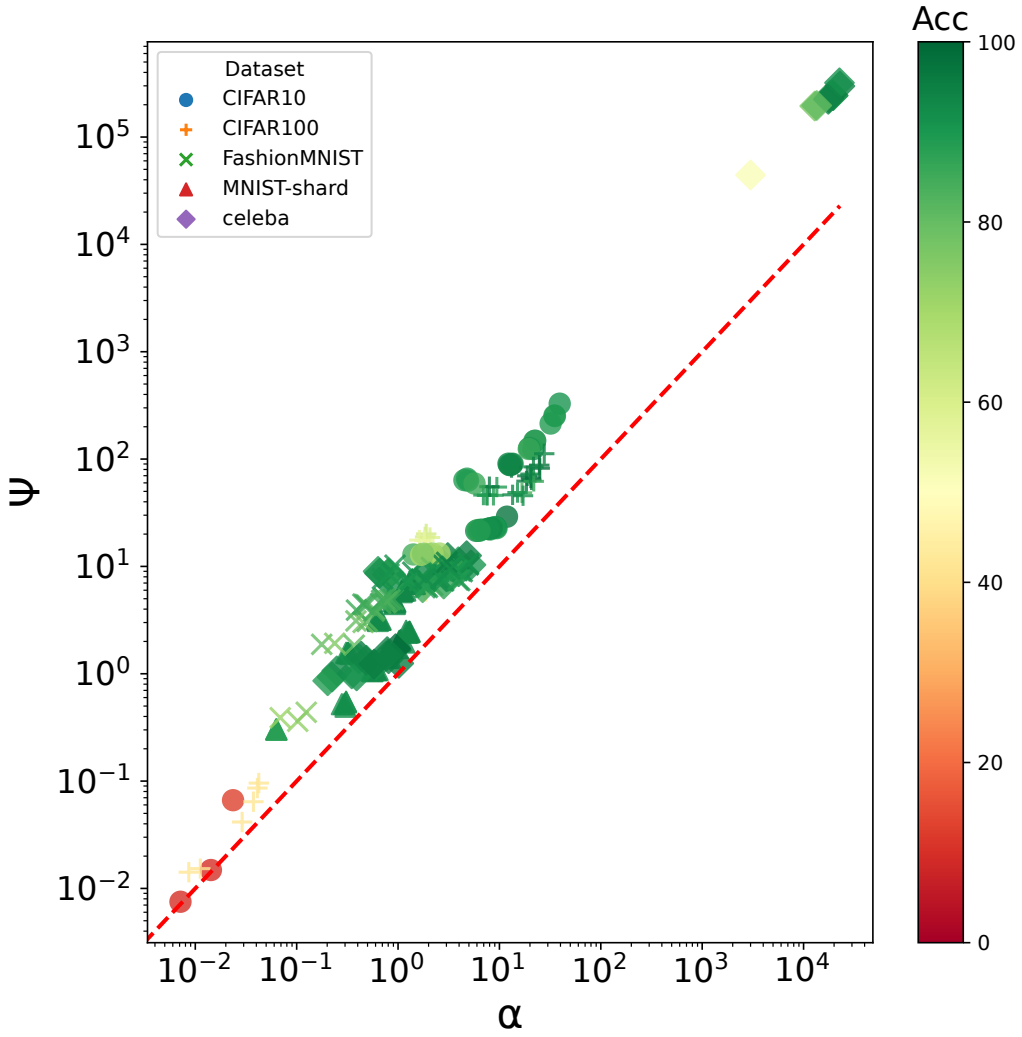

Figure 7: Comparison between Ψ and α for $B = 1$ for all datasets and various hyper-parameters. We observe that the bound demonstrated in Theorem 1 holds for all the considered experimental scenarios, even when setting $B = 1$ in neural networks.