# Trajectory-Aware Certified Decentralized Unlearning via SGD Stability

**Anonymous Authors**[1]

## Abstract

Decentralized Unlearning (DU) aims to remove the influence of specific clients from a collaboratively trained global model. However, existing methods suffer from strong reliance on static, problem-specific hyperparameters or restrictive convexity assumptions, limiting their general applicability. To overcome these limitations, we propose **TRA**jectory-aware **CE**rtified **D**ecentralized **U**nlearning (**TRACE-DU**), a generic unlearning framework for decentralized training. **TRACE-DU** introduces a fine-grained sensitivity analysis that leverages local SGD updates and decentralized training dynamics, thereby eliminating the need for convexity assumptions and reducing dependence on manually tuned parameters. By integrating strategic checkpoint selection with calibrated noise perturbation, the proposed framework enables efficient certified unlearning. Moreover, we exploit historical model trajectories to extend this framework, enabling it to naturally support sequential unlearning requests from an arbitrary number of clients. We provide theoretical guarantees for certified unlearning and derive sensitivity bounds under both convex and non-convex loss functions. Experimental results demonstrate that our framework outperforms state-of-the-art baselines across diverse metrics.

## 1. Introduction

Collaborative learning methods, including Federated Learning (FL) (McMahan et al., 2017) and Decentralized Learning (DL) (Lian et al., 2017), have advanced privacy-preserving machine learning. However, these methods face a significant challenge in complying with new privacy regulations (e.g., GDPR) (Voigt and Von dem Bussche, 2017), which guarantee clients the right to be forgotten (RTBF). Specifically, clients may request the removal of private data at any time during training. While a straightforward solution is to retrain the model on the retained data after a deletion request, this is often prohibitively expensive. Consequently, Machine Unlearning (MU) has emerged as a promising solution to avoid the cost of retraining while fulfilling privacy requirements (Cao and Yang, 2015). The formal privacy guarantee for MU requires that the unlearned model be statistically indistinguishable from the one retrained from scratch. Inspired by differential privacy, Guo et al. (2020) introduced the definition of $(\epsilon, \delta)$-unlearning to guarantee model indistinguishability (i.e., certified unlearning).

However, implementing MU in collaborative learning poses additional challenges. Existing work has shown that deleted data can be inferred by adversaries from the trained model (Tao et al., 2024). Consequently, effective unlearning in FL or DL requires removing not only the data but also its influence on the consensus model. In a system with multiple clients, this can be translated to removing the impact of specific clients who request complete deletion (i.e., client-wise unlearning) (Khalil et al., 2025; Zhong et al., 2025). Numerous efforts have been made to address machine unlearning in collaborative learning, but most of them focus on centralized settings with a central server (Liu et al., 2024).

The lack of a central coordinator prevents the straightforward application of centralized unlearning methodologies to decentralized regimes. Currently, only a handful of studies have addressed unlearning within fully decentralized settings. Despite these advancements, existing approaches are plagued by significant limitations, including lack of theoretical guarantees (Ye et al., 2024), heavy reliance on problem-specific parameters and convexity assumptions (Qiao et al., 2025a; Wu et al., 2026), and restrictions to fixed topologies (Lamri and Maniatakos, 2025). To address these challenges, we propose several key insights to advance certified and efficient decentralized unlearning.

**(1). Tracing sensitivity via decentralized training trajectories.** Beyond establishing model indistinguishability, certified unlearning requires rigorously bounding the error distance between the unlearned model and the exact retrained one. This requirement is technically known as *model sensitivity* bound. Model sensitivity is critical to certified un-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

learning since it directly determines the noise magnitude and therefore governs the attainable utility. Most existing studies upper bound sensitivity mainly through problem parameters (Sekhari et al., 2021; Lamri and Maniatakos, 2025; Wu et al., 2026). However, in decentralized scenarios, these static bounds fail to capture the complex dynamics characterizing how local updates contribute to the consensus model throughout the collaborative learning process. Such client-specific contributions can be naturally quantified by the deviation between clients' local models and consensus models along the training trajectory. Motivated by this observation, we introduce a fine-grained sensitivity analysis derived from the decentralized training dynamics and trajectory, which facilitates superior unlearning performance.

**(2). Unlocking the power of local SGD updates.** In conjunction with evaluating client contributions from a global view, we further investigate the process of local updates, which encompass local SGD computation and mixing aggregation between neighbors. While existing studies largely prioritize analyzing the influence propagation in peer-to-peer communication and mixing updates (Qiao et al., 2025a; Wu et al., 2026), we emphasize the significant potential of the local SGD step itself. To be specific, we take insights from the well-established stability of SGD (Hardt et al., 2016) and first leverage this property to achieve certified decentralized unlearning.

**(3). Going beyond the convexity setting.** Current certified unlearning methods often leverage second-order information for model correction (Sekhari et al., 2021; Suriyakumar and Wilson, 2022), a straightforward and effective approach that has been extended to decentralized settings (Wu et al., 2026). However, the theoretical guarantees for these methods typically rely on the convexity assumption, which often does not hold in practical scenarios such as neural networks (He et al., 2016). While empirical studies have demonstrated their effectiveness, we argue that a theoretical breakthrough beyond convexity assumptions is essential for generalizing certified decentralized unlearning.

**Our Contributions.** We propose a generic framework called **TRA**jectory-aware **CE**rtified **D**ecentralized **U**nlearning (**TRACE-DU**) to tackle the associated challenges in a unified, efficient manner. We start by leveraging both properties of local SGD update rules and individual contributions to establish a traceable sensitivity quantification, which is based on the model update trajectory. We then utilize it to select the checkpoint that exceeds the sensitivity threshold to perturb with carefully calibrated noise. Finally, we perform subsequent training initialized with the perturbed model. To better align with practical scenarios, we further extend our framework to handle sequential unlearning requests.

In summary, our main contributions are threefold:

- By leveraging model updates from a dual local-global perspective, we propose a generic certified DU framework under dynamic topologies with a traceable sensitivity bound and checkpoint selection. Notably, the proposed sensitivity bound relaxes the strict dependency on problem-specific parameters. Furthermore, our framework can be integrated into standard decentralized SGD frameworks and applies to both convex and non-convex optimization regimes.
- We provide a rigorous theoretical analysis demonstrating that our framework achieves $(\epsilon, \delta)$-unlearning. Moreover, we establish sensitivity bound and guarantee model utility under diverse settings. All our theoretical results are free from convexity assumptions, bridging a notable gap.
- Extensive experiments further confirm that our method outperforms state-of-the-art baselines across diverse metrics: model utility, unlearning quality, privacy protection, and unlearning efficiency.

## 2. Related Work

**Machine Unlearning** Machine Unlearning (MU) aims to efficiently remove the influence of specific data points from trained models without fully retraining. The first formal definition of MU was introduced by Cao and Yang (2015), which requires the outputs of an unlearning algorithm to match those of a model obtained through retraining (i.e., exact unlearning). Since then, various studies have proposed alternative definitions ensuring statistical indistinguishability between unlearned and retrained models (i.e., approximate unlearning) (Guo et al., 2020). A certified approximate unlearning algorithm should satisfy formal certified removal guarantees such as $(\epsilon, \delta)$-unlearning (Sekhari et al., 2021). Prior approaches often utilized the second-order information to scrub the trained model (Sekhari et al., 2021; Suriyakumar and Wilson, 2022) or used gradient-based methods for perturbation (Neel et al., 2021). These techniques are effective but rely heavily on the convexity of loss functions. Given the widespread application of neural networks and their non-convex nature (He et al., 2016), relaxing the convexity requirement is essential, while the absence of a unique global minimum in non-convex functions makes this particularly challenging. Prior research on certified unlearning in non-convex settings has yielded several effective methods, primarily through noise perturbation (Zhang et al., 2024; Chien et al., 2024; Qiao et al., 2025b) or by selecting checkpoints for subsequent training (Fraboni et al., 2024; Mu and Klabjan, 2025). However, these methods focus on centralized or single-node architectures, which pose fundamental challenges to their application in decentralized scenarios.

*Table 1.* Comparison of existing decentralized unlearning algorithms

| ALGORITHM | GUARANTEE | CONVEX | NON-CONVEX | DYN. TOPOLOGY | TRACEABLE SEN. |
|---|---|---|---|---|---|
| HDUS (YE ET AL., 2024) | × | × | × | √ | × |
| PDUDT (QIAO ET AL., 2025A) | √ | √ | √ | √ | × |
| RR-DU (LAMRI AND MANIATAKOS, 2025) | √ | √ | √ | × | × |
| CDU (WU ET AL., 2026) | √ | √ | × | √ | × |
| **TRACE-DU/SDU (OUR WORK)** | √ | √ | √ | √ | √ |

Note that *DYN. TOPOLOGY* denotes the decentralized network with dynamic topologies; *TRACEABLE SEN.* denotes the sensitivity bound derived from model update trajectories rather than static problem parameters;

**Decentralized Unlearning** Decentralized Unlearning (DU) aims to eliminate the impact of a specific client from a fully decentralized system (Qiao et al., 2025a). To the best of our knowledge, there are currently only a handful of DU algorithms. We discuss the representative ones as follows. Ye et al. (2024) introduced a DU mechanism that leverages distilled seed models to achieve exact unlearning. The lack of unlearning guarantee makes their algorithm theoretically unconvincing. Additionally, the involvement of model distillation presents challenges for integration with existing decentralized SGD (DSGD) algorithms (Lian et al., 2017; Koloskova et al., 2020) and introduces additional computational overhead. Qiao et al. (2025a) proposed the first provably approximate DU algorithm with certified unlearning guarantees. While PDUDT incorporates noise perturbation based on local model parameters, it suffers from a critical limitation that the sensitivity bound involves an excessive number of problem parameters, making it hard to quantify and theoretically insufficient (see Theorem 4.9 in (Qiao et al., 2025a)). Another existing certified DU algorithm, RR-DU (Lamri and Maniatakos, 2025), operates via a random walk mechanism that performs projected noisy gradient ascent, but it only considers decentralized networks with fixed topologies instead of dynamic ones. Recently, Wu et al. (2026) constructed corrective model updates with Newton-style approximations and proposed an efficient certified DU algorithm. Although this method is highly effective, its theoretical guarantees rely on the assumption of convexity, which limits its applicability to a broader range of problems (Wu et al., 2026). These limitations in existing methods motivate us to develop a generic DU framework that fulfills all the following requirements: certified unlearning guarantee, quantifiable sensitivity bounds, broad applicability and unlearning efficiency. Table 1 presents an intuitive comparison of the proposed approach with existing DU methods across multiple dimensions.

## 3. Preliminaries

**Decentralized Learning** We consider a fully decentralized learning (DL) system operating over a network with potentially time-varying topology. Let $\mathcal{I}$ denote the set of indices of $N$ participating clients. Each client-$i$ ($i \in \mathcal{I}$) possesses its private local dataset $S_i$, which is drawn from a distinct local distribution $\mathcal{D}_i$. Unlike traditional FL, this system eliminates the central server and clients communicate exclusively with their current neighbors to train models collaboratively. Consistent with prior work (Qiao et al., 2025a; Wu et al., 2026), we also consider decentralized networks with dynamic topologies instead of fixed ones (Lamri and Maniatakos, 2025). This flexibility does not compromise our theoretical analysis, as shown in Appendix A.1. Specifically, the communication links between clients at round $t$ are captured by a time-varying symmetric doubly stochastic matrix $\mathbf{Q}^t \in \mathbb{R}^{N\times N}$, which means $\mathbf{Q}^t_{ij} = \mathbf{Q}^t_{ji}$ for all $i, j$ and $\sum_i \mathbf{Q}^t_{ji} = 1$ for all $j$. If $i$ and $j$ are not connected, $\mathbf{Q}^t_{ij} = 0$ (Lian et al., 2017).

We denote $F_i(x, \xi_{n_i})$ as the training loss assessed on the data sample $\xi_{n_i}$ and $f_i(x) = \frac{1}{|S_i|}\sum_{n_i=1}^{|S_i|} F_i(x, \xi_{n_i})$ as the local function. Then we formalize the learning objective as the following decentralized optimization problem:

$$\min_{x\in\mathbb{R}^d} f_{\mathcal{I}}(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x). \tag{1}$$

During the learning process, clients collaboratively train a consensus model $\bar{x}^t = \frac{1}{N}\sum_{i=1}^{N} x_i^t$, where $x_i^t$ is the local model of client-$i$ after performing $t$ communication iterations. In each communication round $t$, client-$i$ conducts a local update by $K$ steps of SGD. We assume that the consensus model $\bar{x}^T$ converges to a stationary solution of the optimization problem (1) after $T$ rounds. This process is similar to DSGD in (Koloskova et al., 2020) and represents a common paradigm in decentralized learning.

To facilitate our analysis, we make the following standard assumption regarding the loss functions. Our analysis accommodates a broad range of objective landscapes, including non-convex, convex, and $\lambda$-strongly convex settings.

**Assumption 3.1.** Loss functions $F_i(\cdot, \xi_{n_i})$ are $L$-smooth. Specifically, for any data sample $\xi_{n_i}$ and any $x, y \in \mathbb{R}^d$, we have $\|\nabla F_i(x, \xi_{n_i}) - \nabla F_i(y, \xi_{n_i})\|_2 \leq L\|x - y\|_2$.

**Decentralized Unlearning** Clients reserve the right to exit the system at any round $t$ ($t = 0, 1, \ldots, T$) and request

that the models trained using their data be removed. To accommodate this, a decentralized unlearning algorithm is executed to generate an unlearned model that satisfies unlearning guarantees.

To theoretically guarantee the effectiveness of unlearning, the unlearned model should be statistically indistinguishable from a model trained without using the deleted data. This requirement is in line with the principles of certified approximate unlearning. More specifically, certified decentralized unlearning should satisfy the notion of $(\epsilon, \delta)$-unlearning which is formalized in the following definition. We clarify that since our focus is on client-wise unlearning (Zhang et al., 2023; Qiao et al., 2025a), the following definition is adapted from the original $(\epsilon, \delta)$-unlearning (Sekhari et al., 2021).

**Definition 3.2.** Let $\mathcal{W} \subseteq \mathbb{R}^d$ denote the model space and $\mathcal{I}_-$ denote the retained client set obtained by deleting several clients. A learning algorithm $\mathcal{A}$ and an unlearning algorithm $\mathcal{M}$ achieve client-wise $(\epsilon, \delta)$-unlearning if, for any subset $W \in \mathcal{W}$, we have

$$\mathbb{P}\left(\mathcal{M}(\mathcal{X}) \in W\right) \leq e^{\epsilon}\mathbb{P}\left(\mathcal{A}(\mathcal{I}_-) \in W\right) + \delta,$$
$$\mathbb{P}\left(\mathcal{A}(\mathcal{I}_-) \in W\right) \leq e^{\epsilon}\mathbb{P}\left(\mathcal{M}(\mathcal{X}) \in W\right) + \delta,$$

where $\mathcal{X}$ denotes the data statistics available to $\mathcal{M}$.

Additionally, for any given client-$c$ ($c \in \mathcal{I}$) who requests deletion, we define the retained client set as $\mathcal{I}_c := \mathcal{I} \setminus \{c\}$. Similar to prior studies (Sekhari et al., 2021; Fraboni et al., 2024), we introduce the model sensitivity with respect to client-$c$ after $t$ rounds as

$$\mathcal{S}(t, c) := ||\bar{\omega}^t - \bar{x}^t||_2, \tag{2}$$

where $\bar{x}^t$ is the consensus model obtained by applying learning algorithm for $t$ iterations over the data of clients in $\mathcal{I}$ and $\bar{\omega}^t$ is the consensus unlearned model for set $\mathcal{I}_c$. The sensitivity must be strictly bounded to guarantee the utility of the unlearned model. Furthermore, this sensitivity bound is further employed to calibrate the Gaussian noise required for achieving $(\epsilon, \delta)$-unlearning (Dwork and Roth, 2014; Sekhari et al., 2021).

## 4. Our Method

### 4.1. Trajectory-Aware Decentralized Unlearning via SGD Stability

In the training phase, each client first executes local SGD updates and then performs a weighted aggregation of neighboring models according to the mixing matrix. This mechanism motivates us to investigate the underlying dynamics of these local optimization trajectories. We begin by leveraging the expansion properties of stochastic gradient descent to analyze the local updates of client-$i$ (Hardt et al., 2016). Let the one step SGD update operator be denoted by $\mathcal{G}(f_i, \gamma, x_i^{t,k}) = x_i^{t,k} - \gamma \nabla f(x_i^{t,k}, \xi_{n_i}^{t,k})$, we then obtain:

$$\begin{aligned}\|\omega_i^{t,k+1} - x_i^{t,k+1}\|_2 &= \|\mathcal{G}(f_i, \gamma, \omega_i^{t,k}) - \mathcal{G}(f_i, \gamma, x_i^{t,k})\|_2 \\ &\leq G(f_i, \gamma) \cdot \|\omega_i^{t,k} - x_i^{t,k}\|_2. \end{aligned} \tag{3}$$

Here $x_i^{t,k}$ corresponds to the local model of client-$i$ at local step $k$ of round $t$ before unlearning, $\omega_i^{t,k}$ denotes the updated model after the removal of client-$c$. $G(f_i, \gamma)$ is a constant related to smoothness and convexity that reflects the stability of SGD. We list the three cases of $G(f_i, \gamma)$ in Table 2, which align with the result of Lemma 3.7 in (Hardt et al., 2016). Full details are provided in Appendix A.1.1.

*Table 2.* Three cases of the stability factor $G(f_i, \gamma)$.

| Convexity | Learning Rate | $G(f_i, \gamma)$ |
|---|---|---|
| Non-convex | – | $1 + \gamma L$ |
| Convex | $\gamma \leq 2/L$ | $1$ |
| $\lambda$-strongly convex | $\gamma \leq 2/(L + \lambda)$ | $1 - \frac{\gamma L \lambda}{L + \lambda}$ |

Specifically, for a given client, the distance between the unlearned model and the original model is prevented from excessive expansion. Moreover, it is non-expansive under the convex setting and contractive when losses are strongly-convex, which guarantees the desired stability. However, it is infeasible to directly utilize Eq. (3) to bound sensitivity as we cannot directly access the unlearned model $\omega_i$.

For the next step, we introduce a proxy based on tracing decentralized training dynamics to deliver the sensitivity bound. To begin with, decentralized learning dynamics involve the interplay of local SGD updates, neighbor aggregation, and global model mixing. Building upon the analysis of local SGD, we proceed to examine neighbor aggregation and global mixing. For the neighbor aggregation process, the properties of the mixing matrix ensure that the peer-to-peer influence weights asymptotically converge to a uniform distribution as iterations progress (Lian et al., 2017). The global model mixing process implements distributed averaging over the communication graph, enabling clients to reach a consensus model via mechanisms like gossip protocols without central coordination, which is widely adopted in decentralized learning (Yuan et al., 2016; Lian et al., 2017; Koloskova et al., 2020). From a dual local-global perspective, the divergence between locally available model $x_i^t$ and consensus model $\bar{x}^t$ (i.e., $||x_i^t - \bar{x}^t||_2$) can serve as a natural metric for the client-specific contribution along the training process. Integrating the aforementioned analysis, we construct a traceable and weighted proxy along the model update trajectory, formulated as:

$$\Omega_i(\mathcal{I}, \bar{x}^t) := \frac{1}{|\mathcal{I} \setminus \{i\}|}||x_i^t - \bar{x}^t||_2 = \frac{1}{N-1}||x_i^t - \bar{x}^t||_2. \tag{4}$$

Finally, we can establish the upper bound for model sensitivity as in Theorem 4.1, tying it to the stability of SGD and client-specific contribution.

**Theorem 4.1.** *For smooth clients' local loss functions, let $G(f_{\mathcal{I}}, \gamma)$ denote $G(f_i, \gamma)$ for every $i \in \mathcal{I}$ and $c$ denotes the index of unlearned client, we have*

$$\mathcal{S}(t, c) \leq \sum_{p=0}^{t-1} G(f_{\mathcal{I}}, \gamma)^{(t-p-1)K} \cdot \Omega_c(\mathcal{I}, \bar{x}^p), \quad (5)$$

*where $\gamma$ is the learning rate, $K$ is amount of local training steps, $p$ also denotes the index of the communication round and $\Omega_c(\mathcal{I}, \bar{x}^p) := \frac{1}{N-1}||x_c^p - \bar{x}^p||_2$.*

*Proof sketch.* Let $\Delta_t := \|\bar{\omega}_t - \bar{x}_t\|_2$. Using the uniform weights before/after deletion ($u_i = \frac{1}{N}$, $\nu_i = \frac{1}{N-1}(1 - \mathbb{1}_c(i))$), a direct reweighting decomposition gives $\Delta_{t+1} \leq \max_i \|\omega_i^{t+1} - x_i^{t+1}\|_2 + \Omega_c(\mathcal{I}, \bar{x}^{t+1})$. Mixing updates and SGD stability over $K$ local steps imply that $\|\omega_i^{t+1} - x_i^{t+1}\|_2 \leq \max_j \|\omega_j^t - x_j^t\|_2 \leq G(f_{\mathcal{I}}, \gamma)^K \|\omega_j^t - x_j^t\|_2$. Applying the recursive step together with $\Delta_0 = 0$ yields the desired result. We deliver full details in Appendix A.1.1. □

The sensitivity bound delivered above is further used to carefully calibrate Gaussian noise and certified unlearning can be achieved by perturbing the model with this noise. Upon receiving a deletion request from client-$c$ at $t_u$ round, a naive strategy is to directly perturb the consensus model at $t_u$ round with required noise and then continue training. However, it fails to address a critical trade-off inherent to the unlearning process. As the noise magnitude grows as training progresses, adding perturbation at $t_u$ round would severely damage model performance. Consequently, the subsequent recovery phase becomes prohibitively long to regain accuracy, compromising the efficiency of unlearning.

This motivates us to select an earlier checkpoint at which the required perturbation is smaller, while reducing the subsequent training cost to recover utility. To address the critical issue, we further leverage the sensitivity bound to select the unlearning time index. Specifically, we impose a threshold on the model sensitivity bound. Building upon the Gaussian mechanism (Dwork and Roth, 2014), the threshold is defined as $[2\ln(1.25/\delta)]^{-\frac{1}{2}} \epsilon\sigma$. Then we identify the most recent model time index $U$ (i.e., unlearning time index) which satisfies:

$$U = \arg\max_t (\Upsilon(t, c) \leq [2\ln(1.25/\delta)]^{-\frac{1}{2}} \epsilon\sigma), \quad (6)$$

where $\Upsilon(t, c) = \sum_{p=0}^{t-1} G(f_{\mathcal{I}}, \gamma)^{(t-p-1)K} \cdot \Omega_c(\mathcal{I}, \bar{x}^p)$. It can be updated by recurrence with $O(t)$ computational cost over $t$ rounds. After selecting the index $U$, we perturb the consensus model at round $U$ with the calibrated noise as $\tilde{x} = \bar{x}^U + \mathcal{N}(0, \sigma(U, c)^2 \mathbb{I}_d)$, where $\sigma(U, c) = \frac{\Upsilon(U,c)}{\epsilon} \cdot \sqrt{2\ln(1.25/\delta)}$. Then all retained clients continue training for $T_r$ rounds initialized with $\tilde{x}$, which is consistent with prior work (Qiao et al., 2025a). Algorithm 1 provides a detailed implementation of our framework.

Overall, **TRACE-DU** first maintains a fine-grained sensitivity bound grounded in the stability properties of SGD and a traceable proxy derived from locally accessible training information. Benefiting from this, we relax the reliance of the sensitivity bound on problem-specific parameters. Then this bound is utilized to identify the optimal checkpoint for injecting calibrated Gaussian noise.

Finally, leveraging the Gaussian mechanism, we establish the $(\epsilon, \delta)$-unlearning guarantee for our framework.

**Theorem 4.2.** *The learning algorithm $\mathcal{A}$ and the unlearning algorithm $\mathcal{M}$ (**TRACE-DU**) satisfy $(\epsilon, \delta)$-unlearning of client-$c$ with Gaussian noise $\mathcal{N}(0, \sigma(t, c)^2 \mathbb{I}_d)$, where $\sigma(t, c) = \frac{\Upsilon(t,c)}{\epsilon} \cdot \sqrt{2\ln(1.25/\delta)}$ and $\Upsilon(t, c) = \sum_{p=0}^{t-1} G(f_{\mathcal{I}}, \gamma)^{(t-p-1)K} \cdot \Omega_c(\mathcal{I}, \bar{x}^p)$.*

*Proof sketch.* We let $\sigma(t, c) = \frac{\Upsilon(t,c)}{\epsilon} \sqrt{2\ln(1.25/\delta)}$. Then, perturbing $\bar{x}^t$ with noise from $\mathcal{N}(0, \sigma(t, c)^2 \mathbb{I}_d)$ is exactly the Gaussian mechanism (Dwork and Roth, 2014), hence satisfies Definition 3.2. Any subsequent retraining is post-processing and preserves the guarantee. □

### 4.2. Extension to Sequential Unlearning Requests

In what follows, we extend **TRACE-DU** to issue sequential unlearning requests from different clients. We denote $\mathcal{V}_{\mathcal{U}_{total}} = \cup_{u=0}^{\mathcal{U}_{total}} \mathcal{Q}_u$ as a series of $\mathcal{U}_{total}$ unlearning requests, where $\mathcal{Q}_0 = \emptyset$. Each unlearning request can contain an arbitrary number of clients. Let $\mathcal{I}_u := \mathcal{I} \setminus \mathcal{V}_u$ ($\mathcal{I}_0 = \mathcal{I}$) denote retained client set. As discussed before, we need to identify the unlearning time index $U$. However, this becomes challenging under sequential unlearning setting due to the difficulty in checking the precise time index across the whole unlearning sequence.

To address this challenge, we introduce *historical model trajectory* $H$ to track and update the model history for each client. Each client can utilize $H$ to update its bounded sensitivity $\Upsilon(t, c)$ for each unlearning request. The selection of the unlearning index $U$ for a request $u$ depends on the past history of unlearning requests. Similarly to **TRACE-DU**, we identify the first unlearning index $U_1$ for the first unlearning request. Then we perturb the consensus model $\bar{x}_0^{U_1}$ and obtain the new initial model $\bar{x}_1^0$. Finally, we perform subsequent training of $T_r^1$ rounds and update the model trajectory as $H_1 = (\bar{x}_0^0, \ldots, \bar{x}_0^{U_1}, \bar{x}_1^0, \ldots, \bar{x}_1^{T_r^1})$. To handle the second unlearning request, we use the updated

**Algorithm 1 TRACE-DU**

**Input:** unlearning client index $c$, parameters $\epsilon, \delta, \sigma$, subsequent training steps $T_r$, initial local model $x_i^0 = \bar{x}^0$.

**Learning process:**
**for** each client-$i \in \mathcal{I}$ **in parallel do**
  **for** communication round $t = 0$ **to** $t_u - 1$ **do**
    **for** local update step $k = 0$ **to** $K - 1$ **do**
      $x_i^{t,k+1} \leftarrow x_i^{t,k} - \gamma \nabla f_i(x_i^{t,k}, \xi_{n_i})$;
    **end for**
    Client-$i$ sends $x_i^{t,K}$ to neighbors;
    Client-$i$ aggregates $x_i^{t+1} \leftarrow \sum_j \mathbf{Q}_{ij}^t x_j^{t,K}$;
    Client-$i$ computes $\Upsilon(t,i) \leftarrow \sum_{p=0}^{t-1} G(f_{\mathcal{I}}, \gamma)^{(t-p-1)K} \cdot \Omega_i(\mathcal{I}, \bar{x}^p)$;
  **end for**
**end for**
**Unlearning process:**
Client-$c$ submits unlearning request at round $t_u$;
Client-$c$ identifies the latest checkpoint $U = \arg\max_t \left(\Upsilon(t,c) \leq [2\ln(1.25/\delta)]^{-1/2}\epsilon\sigma\right)$;
Perturb the checkpoint as $\tilde{x} = \bar{x}^U + \mathcal{N}(0, \sigma(U,c)^2 \mathbb{I}_d)$;
**for** each retained client **do**
  Continue training for $T_r$ rounds initialized at $\tilde{x}$;
**end for**

model trajectory $H_1$ to identify the second unlearning index $U_2$ and repeat the previous steps.

To be general, we define $H_u$ as the model trajectory after $u$ unlearning requests. For a set of clients $\mathcal{Q}_{u+1}$ in the next unlearning request, we need to compute the maximum bounded sensitivity according to $H_u$:

$$\Upsilon_{u+1}(t,c) := \sum_{p=0}^{t-1} G(f_{\mathcal{I}}, \gamma)^{(t-p-1)K} \cdot \Omega_{c,p}(\mathcal{I}_s, \bar{x}_s^p), \quad (7)$$

$$\Upsilon_{u+1}(t, \mathcal{Q}_{u+1}) := \max_{c \in \mathcal{Q}_{u+1}} \Upsilon_{u+1}(t,c), \quad (8)$$

where $\Omega_{c,p}(\mathcal{I}_s, \bar{x}_s^p)$ is the $p$-th element of the sequence $Seq_c^u = (\Omega_c(\mathcal{I}_s, \bar{x}_s^p)$ for $\bar{x}_s^p \in H_u)$ and $s = 0, 1, \ldots, u$.

After evaluating $\Upsilon_{u+1}(t, \mathcal{Q}_{u+1})$ along the model trajectory, we can identify the optimal time index for perturbing as

$$U_{u+1} = \arg\max_t (\Upsilon_{u+1}(t, \mathcal{Q}_{u+1}) \leq [2\ln(1.25/\delta)]^{-\frac{1}{2}} \epsilon\sigma). \quad (9)$$

We proceed by perturbing the $U_{u+1}$-th model $H_u(U_{u+1})$ in $H_u$ with Gaussian noise defined in Theorem 4.2 to obtain the new initial model $\bar{x}_{u+1}^0$, where $\sigma(U_{u+1}, \mathcal{Q}_{u+1}) = \frac{\Upsilon_{u+1}(U_{u+1}, \mathcal{Q}_{u+1})}{\varepsilon} \sqrt{2\ln(1.25/\delta)}$. Then a subsequent training process is operated on retained clients to get $\bar{x}_{u+1}^{T_r^{u+1}}$ and to update the historical model trajectory as

$$H_{u+1} = (\bar{x}_0^0, \ldots, H_u(U_{u+1}), \bar{x}_{u+1}^0, \ldots, \bar{x}_{u+1}^{T_r^{u+1}}). \quad (10)$$

**Algorithm 2 TRACE-SDU** (Sequential DU)

**Input:** unlearning client sets $\{\mathcal{Q}_u\}_{u=0}^{\mathcal{U}_{\text{total}}}$, parameters $\epsilon, \delta, \sigma$, subsequent training steps $\{T_r^u\}_{u=0}^{\mathcal{U}_{total}}$
**for** $u = 0$ **to** $\mathcal{U}_{total}$ **do**
  Update $\mathcal{I}_{u+1} \leftarrow \mathcal{I}_u \setminus \mathcal{Q}_{u+1}$;
  Compute $\Upsilon_{u+1}(t, \mathcal{Q}_{u+1})$ with Eq. (8);
  Compute unlearning time index $U_{u+1}$ with Eq. (9);
  Perturb to get the new initial model: $\bar{x}_{u+1}^0 = H_u(U_{u+1}) + \mathcal{N}(0, \sigma(U_{u+1}, \mathcal{Q}_{u+1})^2 \mathbb{I}_d)$;
  Continue decentralized training for $T_r^{u+1}$ rounds starting from $\bar{x}_{u+1}^0$;
  Update historical model trajectory $H_{u+1}$ with Eq. (10)
**end for**

Finally, we also establish certified unlearning guarantees for the sequential extension **TRACE-SDU**. Details of all our theoretical results are provided in Appendix A.1.

**Theorem 4.3.** *The learning algorithm $\mathcal{A}$ and the unlearning algorithm $\mathcal{M}'$ (**TRACE-SDU**) also satisfy $(\epsilon, \delta)$-unlearning for clients in sequential unlearning requests.*

*Proof sketch.* Building upon Theorem 4.2, the result follows directly by induction on the request index $u$. □

## 5. Experiments

### 5.1. Experimental Setup

**Datasets and Models.** We use standard benchmark datasets in our experiments, including MNIST (Lecun et al., 1998), Fashion-MNIST (Xiao et al., 2017) (denoted as F-MNIST), and CIFAR-10 (Krizhevsky, 2009). For the convex task, we employ Logistic Regression on the MNIST dataset. For non-convex tasks, we utilize the CNN model on the Fashion-MNIST dataset and the ResNet-18 model (He et al., 2016) on the CIFAR-10 dataset.

**Baselines and Evaluation Metrics.** We compare our proposed framework with three existing baselines: *Retrain*, which means retraining from scratch with retained clients by D-PSGD (Lian et al., 2017), *PDUDT* (Qiao et al., 2025a) and *Certified Hessian-based DU Algorithm* (Wu et al., 2026). For brevity, we denote the last one as *CDU*. Following prior studies (Khalil et al., 2025; Qiao et al., 2025a; Wu et al., 2026), we employ the following metrics: *Retain Accuracy*, *Forget Accuracy*, measuring the model's performance on retained client set and unlearned client set, respectively. *Membership Inference Attack (MIA)* indicates the extent to which the data of unlearned clients remains recognizable in the model. *Unlearning Time* denotes the runtime required to perform the unlearning process. We further investigate the impact of varying noise scales to demonstrate the algorithmic robustness of our framework in Appendix A.3.3.

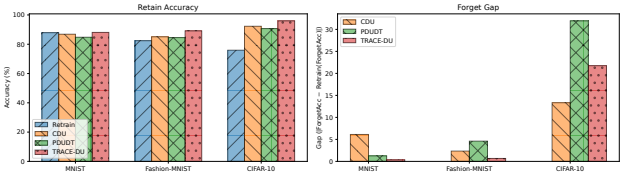


*Figure 1.* Comparison of retain accuracy and forget accuracy gap across different datasets under *Single-DU*.

**Implementation Details.** We consider a fully decentralized system with *50* clients and no central server under non-IID data distributions, where the connections between clients are randomly generated. For **TRACE-DU**, we focus on a single unlearning request issued by one randomly selected client. For **TRACE-SDU**, we assess performance under the sequential unlearning setting. Specifically, the experimental protocol includes 10 sequential requests for single-client unlearning (one client per request) and 5 sequential requests for multi-client unlearning (3 clients per request). We refer to these two scenarios as *Single-DU* and *Seq-DU*, respectively. For *Single-DU*, the unlearning request is set to occur at round $t_u = 40$ and we also set the number of subsequent training rounds to be equal to $t_u$ (i.e., $T_r = 40$) for fairness. For *Seq-DU*, the first unlearning request is set to occur at round $t_1 = 40$. Following a consistent evaluation protocol, we allocate $40$ rounds of subsequent training for every unlearning request, which implies that unlearning requests occur sequentially every $40$ rounds. Under this setting, we do not assume that the training procedure has converged when an unlearning request arrives. This formulation captures a more realistic scenario where clients may request deletion at any intermediate training stage, ensuring better compliance with privacy regulations. Due to space constraints, we provide full details of our setup and additional experimental results in Appendix A.2 and A.3.

### 5.2. Unlearning Performance

**Model Utility** We measure the test accuracy on the retained clients to assess model utility (denoted as *Retain Accuracy*), where higher accuracy indicates better preservation of model performance. For *Single-DU* scenario, the results in Figure 1 demonstrate the improved performance of our proposed framework across different datasets. For *Seq-DU* scenario, we first ensure that all methods reach the same accuracy before the issuance of the first unlearning request and denote the related unlearning request index as 0. We also reduce the interval between unlearning requests to a small number of rounds to characterize frequent sequential unlearning. Under this realistic setting, results in Figure 2 reveal that our framework maintains exceptional advantages over all baselines. Specifically, we record the retain accuracy after each method fully processes an unlearning request (i.e., the accuracy measured after the unlearning phase followed by subsequent training). As shown in Figure 2, all baselines suffer from performance degradation and fail to restore model utility during frequent sequential unlearning. In contrast, benefiting from our strategic checkpoint mechanism, **TRACE-SDU** consistently achieves superior accuracy within the same recovery budget. We provide more details of Figure 2 in Appendix A.3.1, including a justification for our method's performance advantage over *Retrain* baseline.

*Table 3.* Comparison of MIA results on different datasets

| METHOD | MIA PRECISION (%) | | |
|---|---|---|---|
| | MNIST | F-MNIST | CIFAR-10 |
| RETRAIN | $56.31_{\pm 3.29}$ | $46.56_{\pm 3.00}$ | $51.63_{\pm 1.38}$ |
| PDUDT | $54.32_{\pm 1.61}$ | $49.60_{\pm 0.27}$ | $55.19_{\pm 1.44}$ |
| CDU | $54.37_{\pm 3.47}$ | $48.79_{\pm 1.09}$ | $50.12_{\pm 1.21}$ |
| **TRACE-DU** | $51.74_{\pm 0.82}$ | $49.22_{\pm 0.72}$ | $48.59_{\pm 2.30}$ |

**Unlearning Quality** We adopt the accuracy on the unlearned clients (forget set) as a primary metric for assessing unlearning quality (denoted as *Forget Accuracy*). Specifically, aligning with prior research (Khalil et al., 2025), we utilize the forget accuracy of the model retrained from scratch as the gold standard. Then we benchmark various methods by calculating the gap between their forget accuracy and this standard, where a minimal difference denotes superior unlearning quality. The results in Figure 1 demonstrate the effectiveness of our unlearning framework. Finally, we argue that forget accuracy alone is insufficient to fully characterize unlearning quality. Membership Inference Attack (MIA) metrics are also commonly employed for this evaluation, which we will discuss in the following part.

**Privacy Protection** Consistent with prevailing research standards (Tao et al., 2024; Zhong et al., 2025; Qiao et al., 2025a; Wu et al., 2026), we utilize MIA to assess the privacy protection performance of our framework. MIA aims to infer whether a specific data sample was part of the model's training set. In the context of unlearning, this entails assessing whether the target data can still be distinguished as a training sample by the unlearned model. Higher MIA precision implies severe privacy leakage, whereas the precision approaching 50% (equivalent to random guessing) indicates effective and secure unlearning. In our experiment, we conduct membership inference attacks across different datasets under *Single-DU* scenario, reporting the average precision and standard deviation. As shown in Table 3, the MIA precision results of our method consistently approximate the random guessing. This implies that attackers cannot reliably distinguish training membership, thereby validating the privacy protection performance and unlearning quality of our unlearning framework.

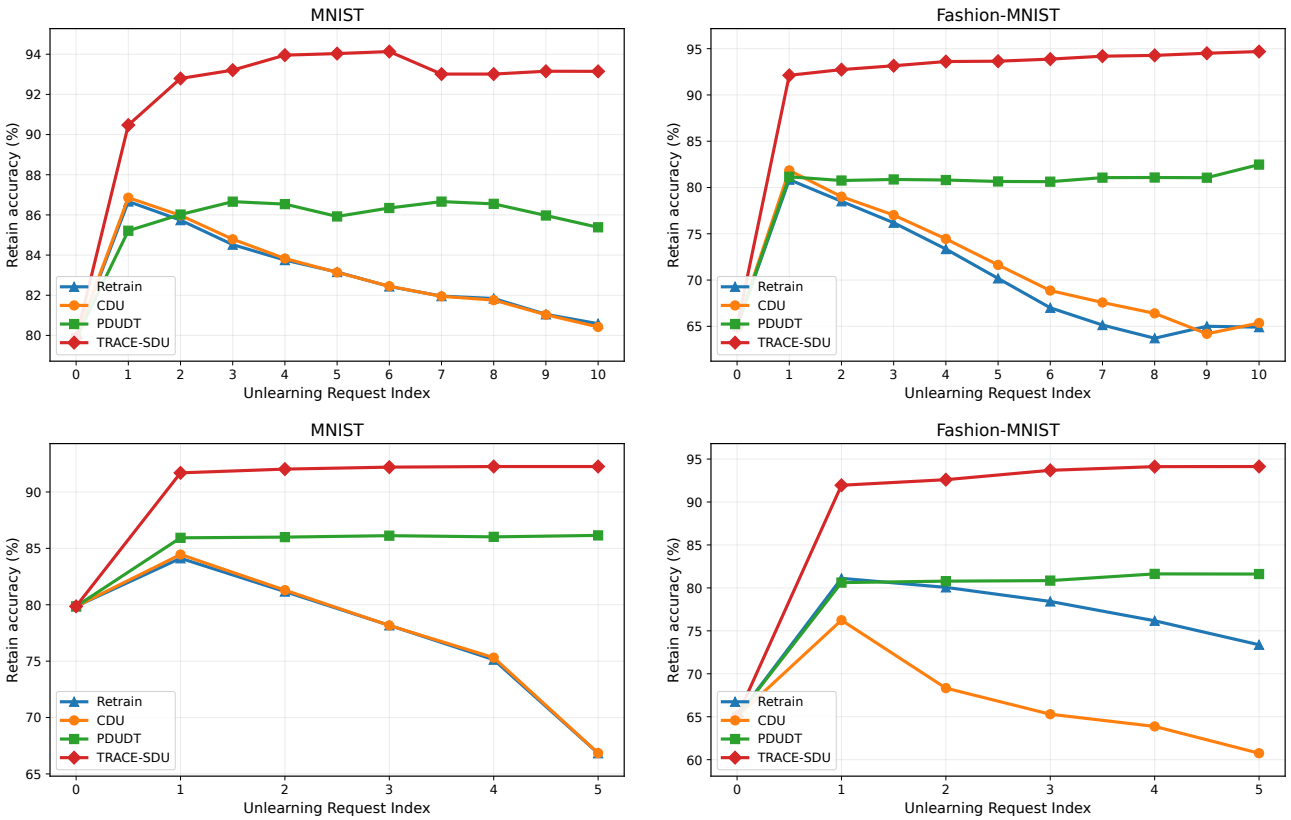


*Figure 2.* Comparison of the retain accuracy of different methods on different datasets under *Seq-DU*. *Top*: 10 sequential unlearning requests, removing one client per request. *Bottom*: 5 sequential unlearning requests, removing 3 clients per request.

*Table 4.* Unlearning time on different datasets

| Method | Unlearning Time (s) | | |
|---|---|---|---|
| | MNIST | F-MNIST | CIFAR-10 |
| Retrain | 3136 | 2996 | 12290 |
| PDUDT | 6.23 | 11.23 | 18.02 |
| CDU | 4.91 | 8.92 | 20.84 |
| **TRACE-DU** | **0.02** | **0.11** | **2.21** |

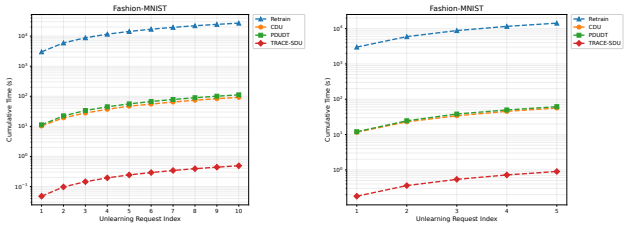

*Figure 3.* Comparison of cumulative unlearning time across different methods on the Fashion-MNIST dataset. *Left*: 10 sequential unlearning requests, removing one client per request. *Right*: 5 sequential unlearning requests, removing 3 clients per request.

**Unlearning Efficiency** To evaluate the unlearning efficiency, we measure the runtime required for the unlearning process. Consistent with prior work (Qiao et al., 2025a), the reported runtime excludes training sessions before and after the unlearning request. Note that for *Retrain* method, the reported time reflects the full time of training from scratch, as it does not decouple unlearning from training. Table 4 reports the unlearning time for *Single-DU* scenario, while Figure 3 illustrates the cumulative unlearning time for *Seq-DU* scenario on F-MNIST datasets. Other results can be found in Appendix A.3. Experimental results demonstrate that our algorithm not only maintains great unlearning performance but also exhibits superior efficiency.

## 6. Conclusion

In this paper, we propose **TRACE-DU**, a generic certified DU framework that exploits local SGD stability and a trajectory-based contribution proxy to derive a traceable sensitivity bound. This bound guides principled checkpoint selection and calibrated noise injection for efficient unlearning. Furthermore, **TRACE-SDU** leverages historical trajectories to handle sequential requests from arbitrary clients. We theoretically establish $(\epsilon, \delta)$-unlearning guarantees across diverse settings, with extensive experiments demonstrating superiority over state-of-the-art baselines.

## Impact Statement

This paper presents work whose goal is to advance the field of machine learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

## References


Yinzhi Cao and Junfeng Yang. Towards Making Systems Forget with Machine Unlearning. In *2015 IEEE Symposium on Security and Privacy*, pages 463–480, 2015. doi: 10.1109/SP.2015.35.

Eli Chien, Haoyu Peter Wang, Ziang Chen, and Pan Li. Langevin Unlearning: A New Perspective of Noisy Gradient Descent for Machine Unlearning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=3LKuC8rbyV.

Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and trends® in theoretical computer science*, 9(3–4):211–407, 2014.

Yann Fraboni, Martin Van Waerebeke, Kevin Scaman, Richard Vidal, Laetitia Kameni, and Marco Lorenzi. SIFU: Sequential Informed Federated Unlearning for Efficient and Provable Client Unlearning in Federated Optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 3457–3465. PMLR, 2024.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified Data Removal from Machine Learning Models. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3832–3842, 2020.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International conference on machine learning*, pages 1225–1234. PMLR, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Yasser H. Khalil, Leo Brunswic, Soufiane Lamghari, Xu Li, Mahdi Beitollahi, and Xi Chen. NoT: Federated Unlearning via Weight Negation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 25759–25769, June 2025.

Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. A Unified Theory of Decentralized sgd with Changing Topology and Local Updates. In *International conference on machine learning*, pages 5381–5393. PMLR, 2020.

Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, Toronto, ON, Canada, 2009.

Hithem Lamri and Michail Maniatakos. Fully Decentralized Certified Unlearning, 2025. URL https://arxiv.org/abs/2512.08443.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. *Advances in neural information processing systems*, 30, 2017.

Ziyao Liu, Yu Jiang, Jiyuan Shen, Minyi Peng, Kwok-Yan Lam, Xingliang Yuan, and Xiaoning Liu. A Survey on Federated Unlearning: Challenges, Methods, and Future Directions. *ACM Computing Surveys*, 57(1):1–38, 2024.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

Siqiao Mu and Diego Klabjan. Rewind-to-delete: Certified machine unlearning for nonconvex functions. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=FgjcLXIUjr.

Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-Delete: Gradient-Based Methods for Machine Unlearning. In *Algorithmic Learning Theory*, pages 931–962. PMLR, 2021.

Jing Qiao, Yu Liu, Zengzhe Chen, Mingyi Li, Yuan Yuan, Xiao Zhang, and Dongxiao Yu. PDUDT: Provable Decentralized Unlearning under Dynamic Topologies. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 50142–50170. PMLR, 2025a.

Xinbao Qiao, Meng Zhang, Ming Tang, and Ermin Wei. Hessian-Free Online Certified Unlearning. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=C3TrHWanh5.

Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember What You Want to Forget: Algorithms for Machine Unlearning. *Advances in Neural Information Processing Systems*, 34:18075–18086, 2021.

Vinith M. Suriyakumar and Ashia C. Wilson. Algorithms that Approximate Data Removal: New Results and Limitations. *Advances in Neural Information Processing Systems*, 35:18892–18903, 2022.

Youming Tao, Cheng-Long Wang, Miao Pan, Dongxiao Yu, Xiuzhen Cheng, and Di Wang. Communication Efficient and Provable Federated Unlearning. *Proceedings of the VLDB Endowment*, 17(5):1119–1131, 2024.

Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A practical guide, 1st ed., Cham: Springer International Publishing*, 10(3152676): 10–5555, 2017.

Hengliang Wu, Youming Tao, Anhao Zhou, Shuzhen Chen, Falko Dressler, and Dongxiao Yu. Certified Unlearning in Decentralized Federated Learning, 2026. URL https://arxiv.org/abs/2601.06436.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms, 2017. URL https://arxiv.org/abs/1708.07747.

Guanhua Ye, Tong Chen, Quoc Viet Hung Nguyen, and Hongzhi Yin. Heterogeneous decentralised machine unlearning with seed model distillation. *CAAI Transactions on Intelligence Technology*, 9(3):608–619, 2024.

Kun Yuan, Qing Ling, and Wotao Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

Binchi Zhang, Yushun Dong, Tianhao Wang, and Jundong Li. Towards Certified Unlearning for Deep Neural Networks. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 58800–58818. PMLR, 2024.

Lefeng Zhang, Tianqing Zhu, Haibin Zhang, Ping Xiong, and Wanlei Zhou. FedRecovery: Differentially Private Machine Unlearning for Federated Learning Frameworks. *IEEE Transactions on Information Forensics and Security*, 18:4732–4746, 2023.

Zhengyi Zhong, Weidong Bao, Ji Wang, Shuai Zhang, Jingxuan Zhou, Lingjuan Lyu, and Wei Yang Bryan Lim. Unlearning through Knowledge Overwriting: Reversible Federated Unlearning via Selective Sparse Adapter. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 30661–30670, 2025.

*Table 5.* Important notions used in our paper.

| Notations | Descriptions |
|---|---|
| $N$ | Total number of clients (nodes) in the decentralized network. |
| $i, j, c$ | Client indices; $c$ is the target client to unlearn. |
| $\mathbf{Q}_{ij}$ | Mixing weight between client-$j$ and $i$. |
| $\mathcal{I}$ | Total client set participating in training. |
| $\mathcal{I}_c$ | Retained client set after removing client $c$. |
| $t$ | Communication round index. |
| $t_u$ | Round when an unlearning request is issued. |
| $K$ | Number of local SGD steps per round. |
| $\gamma$ | Learning rate. |
| $f_i(\cdot)$ | Local function at client-$i$. |
| $\xi_{n_i}$ | Data sample at client-$i$. |
| $x_i^{t,k}$ | Local model of client-$i$ at round $t$ after $k$ local steps. |
| $x_i^t$ | Local aggregated model of client-$i$ at round $t$. |
| $\bar{x}^t$ | Consensus (average) model at round $t$. |
| $\tilde{x}$ | Perturbed checkpoint model (i.e., initial model for subsequent training). |
| $\Omega_c(\mathcal{I}, \bar{x}^t)$ | A proxy for the contribution of client-$c$ at round $t$. |
| $\Upsilon(t, c)$ | Sensitivity upper bound w.r.t. client-$c$ at round $t$. |
| $U$ | Selected checkpoint round in **TRACE-DU**. |
| $T_r$ | Number of subsequent training rounds after perturbation. |
| $\mathcal{Q}_u$ | Client subset removed in the $u$-th request. |
| $\mathcal{U}_{total}$ | Total number of sequential unlearning requests. |
| $\mathcal{I}_u$ | Retained client set after $u$ unlearning requests. |
| $U_u$ | Selected checkpoint for the $u$-th unlearning request. |
| $H_u$ | Historical model trajectory after $u$ unlearning requests. |
| $T_r^u$ | Number of subsequent training rounds for the $u$-th unlearning request. |

# A. Appendix

## A.1. Proof Details of Theoretical Results

Here we deliver the notation table and full details of our theoretical results.

### A.1.1. Proof of Theorem 4.1

*Proof.* Before we begin the proof of Theorem 4.1, we make the following assumption of the time-varying symmetric doubly stochastic matrix $\mathbf{Q}^t$, which is standard assumption in decentralized learning (Lian et al., 2017).

**Assumption A.1.** Given the symmetric doubly stochastic matrix $\mathbf{Q}^t$, and let $\rho_k(\mathbf{Q}^t)$ denote the $k$-th largest eigenvalue of $\mathbf{Q}^t$. Define $\rho := (\max\{|\rho_2(\mathbf{Q}^t)|, |\rho_N(\mathbf{Q}^t)|\})^2$. We assume $\rho < 1$.

Following the analyses in prior work (Wu et al., 2026), we can utilize a uniform weighting scheme to estimate the peer-to-peer influence after $t$ iterations.

Specifically, let $u_i = \frac{1}{N}$ denote the weight of peer-to-peer influence of each client before the unlearning process and $\nu_i = \frac{1}{N-1} \cdot (1 - \mathbb{1}_c(i))$ denote the updated weight of each client (including client-$c$) after client-$c$ requests deletion. Then we have

$$
\begin{aligned}
\|\bar{\omega}^{t+1} - \bar{x}^{t+1}\| &= \|\sum_{i=1}^{N} \nu_i \omega_i^{t+1} - \sum_{i=1}^{N} u_i x_i^{t+1}\| \\
&= \|\sum_{i=1}^{N} \nu_i (\omega_i^{t+1} - x_i^{t+1}) + \sum_{i=1}^{N} (\nu_i - u_i) x_i^{t+1}\| \\
&= \|\sum_{i=1}^{N} \nu_i (\omega_i^{t+1} - x_i^{t+1}) + \sum_{i \neq c} \frac{1}{N-1} (x_i^{t+1} - \bar{x}^{t+1}) - \sum_{i=1}^{N} \frac{1}{N} (x_i^{t+1} - \bar{x}^{t+1})\|
\end{aligned}
$$

$$
\begin{aligned}
&\leq \|\sum_{i=1}^{N}\nu_i(\omega_i^{t+1}-x_i^{t+1})\| + \|\sum_{i\neq c}\frac{1}{N-1}(x_i^{t+1}-\bar{x}^{t+1}) - \sum_{i=1}^{N}\frac{1}{N}(x_i^{t+1}-\bar{x}^{t+1})\| \\
&\leq \|\sum_{i=1}^{N}\nu_i(\omega_i^{t+1}-x_i^{t+1})\| + \|\frac{1}{N-1}(\sum_{i=1}^{N}x_i^{t+1}-x_c^{t+1}) - \frac{1}{N}\sum_{i=1}^{N}x_i^{t+1}\| \\
&\leq \|\sum_{i=1}^{N}\nu_i(\omega_i^{t+1}-x_i^{t+1})\| + \|(\frac{1}{N-1}-\frac{1}{N})\sum_{i=1}^{N}x_i^{t+1} - \frac{1}{N-1}x_c^{t+1}\| \\
&= \|\sum_{i=1}^{N}\nu_i(\omega_i^{t+1}-x_i^{t+1})\| + \frac{1}{N-1}\|\frac{1}{N}\sum_{i=1}^{N}x_i^{t+1} - x_c^{t+1}\| \\
&= \|\sum_{i=1}^{N}\nu_i(\omega_i^{t+1}-x_i^{t+1})\| + \frac{1}{N-1}\|\bar{x}^{t+1} - x_c^{t+1}\| \\
&\leq \sum_{i=1}^{N}\nu_i\|\omega_i^{t+1}-x_i^{t+1}\| + \Omega_c(\mathcal{I},\bar{x}^{t+1}) \\
&\leq \max_{i\in\mathcal{I}}\|\omega_i^{t+1}-x_i^{t+1}\| + \Omega_c(\mathcal{I},\bar{x}^{t+1})
\end{aligned} \tag{11}
$$

Now, for any $i \in \mathcal{I}$, let us give an upper bound for $\|\omega_i^{t+1} - x_i^{t+1}\|$ with a generic function $G$. The specific results in the 3 different cases will then be derived directly by specifying $G$ depending on the hypothesis (Hardt et al., 2016).

First, we define the one SGD step as $\mathcal{G}(f_i,\gamma,x_i^{t,k}) = x_i^{t,k} - \gamma\nabla f(x_i^{t,k},\xi_{n_i})$. Leveraging the contractivity of one step gradient descent for the local update of client-$i$, we have:

$$
\begin{aligned}
\|\omega_i^{t,k+1} - x_i^{t,k+1}\| &= \|\mathcal{G}(f_i,\gamma,\omega_i^{t,k}) - \mathcal{G}(f_i,\gamma,x_i^{t,k})\| \\
&\leq G(f_i,\gamma)\cdot\|\omega_i^{t,k} - x_i^{t,k}\|
\end{aligned} \tag{12}
$$

By applying this equation recursively $K$ times, we get

$$
\|\omega_i^{t,K} - x_i^{t,K}\| \leq G(f_i,\gamma)^K\cdot\|\omega_i^{t} - x_i^{t}\| \tag{13}
$$

Depending on the assumptions made on $f$, we can get 3 distinct forms of $G(f_i,\gamma)$ with their respective assumptions. Let $G(f_{\mathcal{I}},\gamma)$ denote $G(f_i,\gamma)$ for every $i \in \mathcal{I}$. The following results come from the stability of stochastic gradient descent, which have been discussed in (Hardt et al., 2016) (Lemma 3.7). We utilize this result as follows:

1. If $f_i$ is $L$-smooth, then $G(f_{\mathcal{I}},\gamma) = 1 + \gamma\cdot L$.

2. If $f_i$ is $L$-smooth, convex and $\gamma \leq \frac{2}{L}$, then $G(f_{\mathcal{I}},\gamma) = 1$.

3. If $f_i$ is $L$-smooth, $\lambda$-strongly convex and $\gamma \leq \frac{2}{L+\lambda}$, then $G(f_{\mathcal{I}},\gamma) = 1 - \frac{\gamma L\lambda}{L+\lambda}$.

Next, based on the forms of $\omega_i^{t+1}$ and $\omega_i^{t+1}$, we can obtain:

$$
\begin{aligned}
\|\omega_i^{t+1} - x_i^{t+1}\| &= \|\sum_{j\in\mathcal{I}_c}\mathbf{Q}_{ij}^{t}{}'\omega_j^{t,K} - \sum_{j\in\mathcal{I}}\mathbf{Q}_{ij}^{t}x_j^{t,K}\| \\
&\leq \max_j\|\omega_j^{t,K} - x_j^{t,K}\| \\
&\leq G(f_{\mathcal{I}},\gamma)^K\cdot\max_j\|\omega_j^{t} - x_j^{t}\|
\end{aligned} \tag{14}
$$

Now we get:

$$
\|\bar{\omega}^{t+1} - \bar{x}^{t+1}\| \leq G(f_{\mathcal{I}},\gamma)^K\cdot\max_j\|\omega_j^{t} - x_j^{t}\| + \Omega_c(\mathcal{I},\bar{x}^{t}) \tag{15}
$$

Finally, since $\omega_j^0 = x_j^0$ at the initial time, we can have the following bound via recurrence:

$$\|\bar{\omega}^t - \bar{x}^t\| \leq \sum_{p=0}^{t-1} G(f_{\mathcal{I}}, \gamma)^{(t-p-1)K} \cdot \Omega_c(\mathcal{I}, \bar{x}^p) \tag{16}$$

□

### A.1.2. Proof of Theorem 4.2

*Proof.* We denote **TRACE-DU** (Algorithm 1) as unlearning algorithm $\mathcal{M}$ in Definition 3.2 and our learning algorithm $\mathcal{A}$ follows the standard decentralized SGD (Lian et al., 2017; Koloskova et al., 2020).

Then, similar to previous studies (Sekhari et al., 2021; Suriyakumar and Wilson, 2022; Fraboni et al., 2024; Qiao et al., 2025a; Wu et al., 2026), we follow the same proof as in (Dwork and Roth, 2014) (Theorem A.1). Specifically, note that

$$\sigma(t, c) = \frac{\Upsilon(t, c)}{\epsilon} \cdot \sqrt{2\ln(1.25/\delta)} \tag{17}$$

and in Theorem 4.1 we have proved the model sensitivity

$$\mathcal{S}(t, c) \leq \Upsilon(t, c) = \sum_{p=0}^{t-1} G(f_{\mathcal{I}}, \gamma)^{(t-p-1)K} \cdot \Omega_c(\mathcal{I}, \bar{x}^p). \tag{18}$$

Thus, we get the following inequality with the calibrated Gaussian noise scaled to $\mathcal{N}(0, \sigma(t, c)^2 \mathbb{I}_d)$:

$$\mathbb{P}(\bar{\omega} \in W) \leq e^{\epsilon} \mathbb{P}(\bar{x} \in W) + \delta, \tag{19}$$

$$\mathbb{P}(\bar{x} \in W) \leq e^{\epsilon} \mathbb{P}(\bar{\omega} \in W) + \delta. \tag{20}$$

This concludes that the unlearning process without retraining and learning algorithm satisfy $(\epsilon, \delta)$-unlearning guarantee. Finally, we further clarify that the additional retraining process in Algorithm 1 does not hurt this guarantee. It states that the composition of a mapping $m$ with an $(\epsilon, \delta)$-unlearning algorithm is also $(\epsilon, \delta)$-unlearning. Specifically, we denote the $(\epsilon, \delta)$-unlearning algorithm without retraining as $\mathcal{M}^- : \mathcal{P} \times \mathcal{Z} \to \mathcal{W}$ and denote $m : \mathcal{W} \to \mathcal{W}'$ be a randomized mapping (e.g., the retraining process). Then we let $m$ be a deterministic function, fix any pair of neighboring sets $a, b$ with $\|a - b\|_1 \leq 1$, and fix any event $\mathcal{X} \subseteq \mathcal{W}'$. Let $\mathcal{T} = \{r \in \mathcal{W} : m(r) \in \mathcal{X}\}$. We then have:

$$\begin{aligned} \mathbb{P}\big[m(\mathcal{M}^-(a)) \in \mathcal{X}\big] &= \mathbb{P}\big[\mathcal{M}^-(a) \in \mathcal{T}\big] \\ &\leq e^{\epsilon} \mathbb{P}\big[\mathcal{M}^-(b) \in \mathcal{T}\big] + \delta \\ &= e^{\epsilon} \mathbb{P}\big[m(\mathcal{M}^-(b)) \in \mathcal{X}\big] + \delta. \end{aligned} \tag{21}$$

This result proves that $m \circ \mathcal{M}^-$ (i.e., Algorithm 1) satisfies $(\epsilon, \delta)$-unlearning with the learning algorithm and aligns with that differential privacy is immune to post-processing (Dwork and Roth, 2014). □

### A.1.3. Proof of Theorem 4.3

*Proof.* For sequential unlearning requests, we begin with $u = 1$ (i.e., the first unlearning request $\mathcal{Q}_1$). From Theorem 4.2, the first perturbed model $\bar{x}_1^0$ from the checkpoint $\bar{x}_0^{U_1}$.

First we consider the simpler case: $\forall u \leq u, U_u \leq U_{u+1}$. It implies that the model $H_u(U_{u+1})$ that requires adding noise at $(u+1)$-th unlearning appears later in the training history than previous perturbed model $\bar{x}_u^0$ (which is obtained by adding noise to $H_{u-1}(U_u)$). We assume that $(\epsilon, \delta)$-unlearning guarantees are achieved for every client in $\mathcal{V}_u$. Then it is obvious that the next perturbed model $\bar{x}_{u+1}^0$ guarantees $(\epsilon, \delta)$-unlearning of every client in $\mathcal{V}_{u+1}$.

Next, we consider another case: $\exists$ unlearning request $u' \leq u$ such that $U_{u+1} < U_{u'}$. It implies that the model $H_u(U_{u+1})$ requires adding noise at $(u+1)$-th unlearning request appears earlier than previous perturbed model $\bar{x}_{u'}^0$ (which is obtained by adding noise to $H_{u'-1}(U_{u'})$). Thus we have $\Upsilon_{u'}(U_{u+1}, \mathcal{Q}_{u'}) \leq \Upsilon_{u'}(U_{u'}, \mathcal{Q}_{u'}) \leq \Upsilon^*$. It actually means that perturbing the model $H_u(U_{u+1})$ with noise guarantees $(\epsilon, \delta)$-unlearning of the clients in $\mathcal{Q}_{u'}$. We extend this to all unlearning requests $u$ such that $U_{u+1} < U_u$ and can conclude that the next perturbed model $\bar{x}_{u+1}^0$ guarantees $(\epsilon, \delta)$-unlearning of every client in $\mathcal{V}_{u+1}$ by induction. □

## A.2. Missing Details in Experiments

### A.2.1. Implementation Details

For the sake of experimental fairness, we maintain consistent settings for the hyperparameters listed below across all evaluated scenarios and methods.

To better simulate practical scenarios, we evaluate all methods under non-IID data distributions following a Dirichlet distribution with $\alpha = 0.3$.

Each client trains with a batch size of 64 for 2 epochs per round. Consistent with the configurations reported in their original papers (Qiao et al., 2025a; Wu et al., 2026), we set the learning rate to 0.001 for all baselines, ensuring they achieve optimal performance. For the unlearning budget, we set $\epsilon = 50$ and $\delta = 10^{-5}$. We also discuss the algorithmic robustness under different noise scales from $0.025$ to $0.2$ in the following part.

### A.2.2. Baseline Methods

In our experiments, we compare our framework against the following baseline DU methods:

**Retrain (D-PSGD) (Lian et al., 2017)** refers to the process of retraining from scratch using standard decentralized SGD solely on retained clients when specific clients issue unlearning requests. This approach represents exact unlearning and serves as the gold standard for verifying unlearning performance.

**PDUDT (Qiao et al., 2025a)** was the first provable decentralized unlearning algorithm with certified unlearning guarantee. This method utilizes cached statistics of each client such as gradients and intermediate model parameters to compute the approximations of the gradient residuals and add related noise perturbation. Its outstanding unlearning performance was verified by extensive experiments.

**CDU (Wu et al., 2026)** introduced a certified decentralized unlearning algorithm leveraging Newton-style model update. When a client requests deletion, it first performs model correction with curvature information, then the updated model is further perturbed with Gaussian noise and broadcast to retained clients. It is quite efficient due to the effective Newton-style model correction. Although the theoretical results in their paper heavily rely on convex loss settings, the authors empirically demonstrated its performance for non-convex tasks.

### A.2.3. Evaluation Metrics

**Retain Accuracy** represents the test accuracy on the retained clients and serves as a quantitative measure of model utility. Specifically, retain accuracy indicates how well the unlearned models preserve its performance relative to the original state (i.e., before unlearning). Therefore, higher values are desirable for this metric.

**Forget Accuracy** is the test accuracy on the unlearned clients. Since a drop in forget accuracy implies the effective removal of the specific clients' influence, it should be as low as possible. Instead of adopting the simplest comparison, we utilize a different approach to compare forget accuracy. Specifically, we can calculate the gap compared to *RT*, with a smaller gap implying more effective unlearning. This approach has been adopted by prior work (Fraboni et al., 2024; Khalil et al., 2025), confirming its fairness and effectiveness.

**Membership Inference Attack (MIA)** aims to determine whether specific data samples were used during model training. For effective unlearning, it is imperative to prevent attackers from discerning whether data from unlearned clients participated in the training process. Specifically, MIA success rates approaching 50% imply that the attack performs no better than random guessing, indicating an inability to reliably identify training membership. Such results effectively demonstrate effective data removal. Therefore, we utilize MIA on the unlearned model to verify whether the algorithm has successfully eliminated the unlearned clients' influence. It serves as a dual metric for privacy protection performance and unlearning quality.

**Unlearning Time** denotes the runtime required to execute the unlearning operations. As the subsequent training phase employs an identical number of rounds for all algorithms, its runtime is excluded from the measurement, in alignment with prior works (Qiao et al., 2025a; Wu et al., 2026). For our proposed method, the unlearning time comprises checkpoint selection and noise perturbation. For *PDUDT*, it accounts for the computation of gradient residuals, weighting factors, and noise addition. For *CDU*, it encompasses the time for model correction, noise injection, and broadcasting.

*Table 6.* Comparison of best retain accuracy on different datasets under *Single-DU*.

| Method | Best Retain Accuracy (%) | | |
|---|---|---|---|
| | MNIST | F-MNIST | CIFAR-10 |
| Retrain | 87.89 | 82.41 | 75.95 |
| PDUDT | 84.80 | 84.53 | 90.68 |
| CDU | 86.86 | 85.09 | 92.29 |
| TRACE-DU | **88.09** | **89.12** | **95.95** |

*Table 7.* Comparison of forget accuracy gap on different datasets under *Single-DU*.

| Method | Forget Gap $\Delta$ (%) | | |
|---|---|---|---|
| | MNIST | F-MNIST | CIFAR-10 |
| Retrain | 0.00 | 0.00 | 0.00 |
| PDUDT | 1.25 | 4.61 | 31.93 |
| CDU | 6.12 | 2.35 | **13.33** |
| TRACE-DU | **0.35** | **0.67** | 21.75 |

## A.3. Additional Experimental Results

### A.3.1. Unlearning Performance

In this section, we provide supplementary experimental results and discussion to complement the main text.

***Single-DU*.** We first discuss the results in *Single-DU* scenario. Table 6 reports the optimal retain accuracy achieved by various methods across different datasets under the *Single-DU* scenario. In Table 7, we present the absolute difference between the accuracy of each method and *Retrain* baseline. This evaluation metric aligns with established protocols in prior work (Khalil et al., 2025).

***Seq-DU*.** Next, we provide more details and address potential points of confusion regarding Figure 2. First, consistent with prior studies (Qiao et al., 2025a; Wu et al., 2026), we standardize the learning algorithm as D-PSGD (Lian et al., 2017) across all methods. Under identical hyperparameter settings, we conduct a pre-training phase of 40 rounds before the issue of the first unlearning request. At this stage, all methods achieve the same initial accuracy, denoted as unlearning request index 0 (where the retained client set $\mathcal{I}_0 = \mathcal{I}$). To be specific, the initial accuracy is $79.86\%$ on MNIST and $65.01\%$ on Fashion-MNIST, corresponding to the points at request index 0 in Figure 2.

Furthermore, it is necessary to clarify the pronounced sharp increase in retain accuracy observed between request indices 0 and 1 in Figure 2. Specifically, the reported accuracy for each request is evaluated on the retained clients after executing the unlearning algorithm and completing 40 rounds of subsequent training. At index 0, the model has undergone only 40 initial training rounds, which is insufficient to reach optimal accuracy. However, upon the first unlearning request (index 1), the model receives 40 extra rounds of subsequent training beyond the unlearning process, allowing it to approach convergence. Although a small fraction of clients depart (1 or 3 out of 50 clients), the minor performance degradation caused by their exit is vastly outweighed by the substantial accuracy gains achieved through the additional optimization rounds.

**Explanation for the Inferior Performance of *Retrain* in *Seq-DU*.** To begin with, we argue that the constraints imposed by prior works are impractical for real-world applications. *CDU* requires model convergence before unlearning (Wu et al., 2026), while *PDUDT* necessitates an excessively long subsequent training phase ($T_r = 200$) (Qiao et al., 2025a).

Aiming to reflect real-world dynamics, our setting allows clients to initiate unlearning at any training stage and supports high-frequency unlearning requests. Specifically, the initial unlearning request is scheduled after 40 rounds of training. Thereafter, the interval between subsequent unlearning requests is also fixed at 40 rounds.

While the initial performance boost (Index 0 to 1) has been clarified, we proceed to explain the subsequent performance decline of *Retrain* as sequential unlearning requests accumulate. In non-IID settings, the cumulative withdrawal of clients exacerbates data scarcity, rendering the remaining dataset less representative. Consequently, the distribution learned by the model diverges increasingly from the true global distribution, leading to a decline in retain accuracy. Furthermore, for

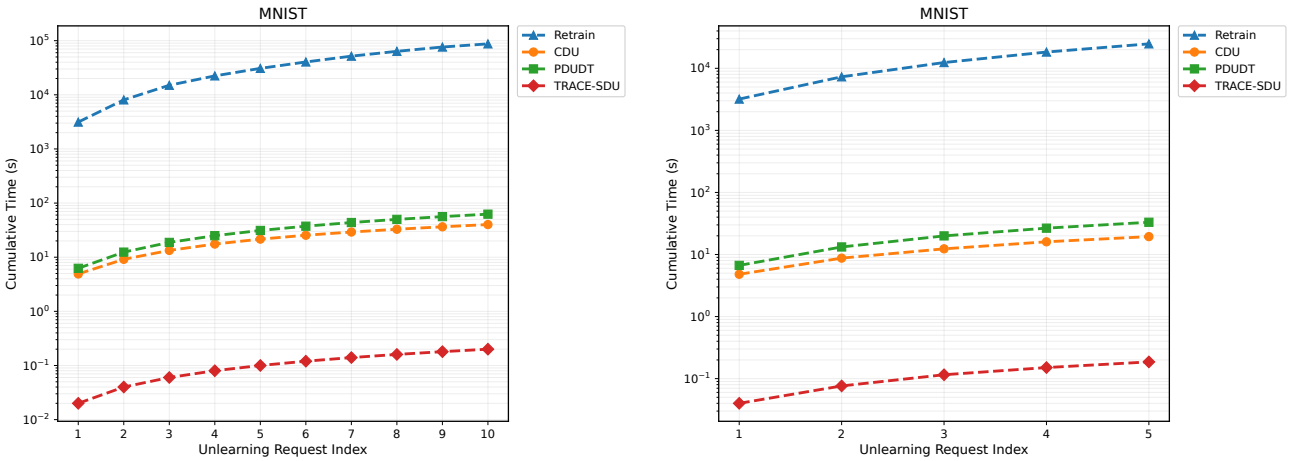


*Figure 4.* Comparison of cumulative unlearning time across different methods for sequential unlearning requests on the MNIST dataset. *Left*: 10 sequential unlearning requests, removing one client per request. *Right*: 5 sequential unlearning requests, removing 3 clients per request.

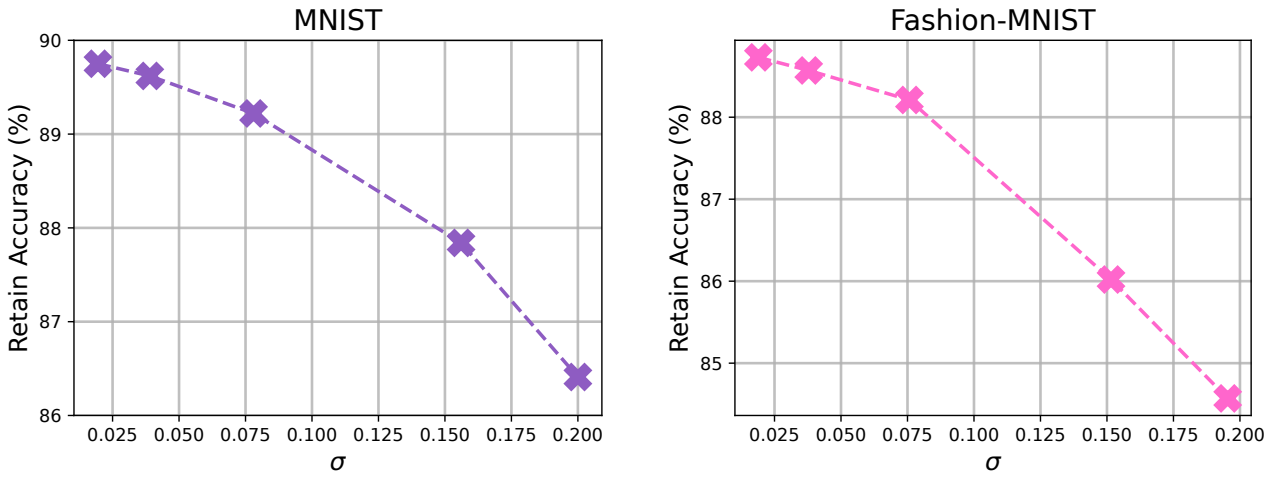


*Figure 5.* Retain accuracy of unlearned models using TRACE-DU under varying noise scales.

each unlearning request, the *Retrain* method necessitates training from scratch on retained clients. However, the frequent withdrawal of clients makes it challenging to fully restore optimal accuracy within the limited interval between requests. Consequently, the combined effect of distribution shifts and frequent unlearning makes the performance of *Retrain* method unsatisfactory in *Seq-DU* scenarios.

### A.3.2. Unlearning Efficiency

Here in Figure 4, we present the cumulative unlearning runtime results for sequential requests on the MNIST dataset.

### A.3.3. Algorithmic Robustness

To demonstrate the robustness of our algorithm, we investigate model performance under varying noise scales. Specifically, we vary the noise parameter $\sigma$ from 0.025 to 0.200 and evaluate retain accuracy across convex and non-convex settings. Figure 5 illustrates that **TRACE-DU** maintains stable performance despite variations in $\sigma$. Even when the noise scale is amplified by a factor of eight, the retain accuracy does not suffer from significant degradation, thereby effectively preserving model utility.