# $f$−FUM: FEDERATED UNLEARNING VIA MIN–MAX AND $f$-DIVERGENCE

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Federated learning (FL) enables collaborative training while keeping raw data on clients. Deletion requests and the discovery of poisoned data create a need to remove selected contributions from an already trained model. This is challenging in FL because client data are decentralized and their influence is entangled through repeated aggregation. We present $f$−Fum, an active federated unlearning method that builds on teacher-student forget/retain optimization. Starting from a pretrained global model, the method keeps that model fixed as a teacher. Clients maximize an $f$-divergence between student and teacher predictive distributions on forget examples, then minimize KL-based teacher-student disagreement and supervised loss on retained examples. The updates are aggregated in separate, sample-weighted synchronization phases without moving raw data to the server or changing the model architecture. We evaluate forget-side divergence choices in client-level and data-level deletion settings involving backdoors, label confusion, and clean-data deletion. Under equal synchronization-phase budgets, divergence choice changes the forgetting-utility trade-off, yielding improvements in some settings and mixed results in others. In the evaluated configurations, $f$−Fum uses up to $8\times$ fewer synchronization phases than full federated retraining.

## 1 INTRODUCTION

The rapid progress of modern machine learning (ML) systems is largely driven by the abundance of data. However, the centralization of vast datasets raises critical privacy and security concerns, especially in sensitive domains such as healthcare, finance, and mobile computing. To mitigate these risks, Federated Learning (FL) has emerged as a powerful collaborative training paradigm. In FL, multiple clients jointly train a global model while keeping their raw data local. By sharing only model updates rather than the data itself, FL fundamentally changes the landscape of privacy-preserving ML and AI systems.

Although FL reduces the need to centralize private data, it does not inherently address what should happen after training when data must be removed. Such removal may be required due to user deletion requests enforcing the European Union's Right to Be Forgotten (EU, 2016), client withdrawal, or the discovery of poisoned and mislabeled data. These concerns have motivated the study of machine unlearning (MU), which aims to remove the influence of specific training data from an already trained model (Cao & Yang, 2015a). The most direct solution, known as exact unlearning, requires retraining the model from scratch without the targeted data. However, exact retraining is computationally prohibitive for deep models and is especially impractical in FL, where data is distributed and communication rounds are costly. To bypass these costs, approximate unlearning methods seek to modify an existing model to behave similarly to a retrained model, substantially reducing overhead. In the federated setting, this gives rise to federated unlearning (FU), where the objective is to efficiently remove the influence of selected samples, updates, or entire clients from the global model while preserving performance on retained data.

Achieving effective FU is fundamentally more complex than traditional centralized MU. In centralized ML, the model owner has direct access to both the full training dataset and the model, making unlearning straightforward. In contrast, FL operates under a decentralized approach where data remains strictly on clients' devices. Consequently, methods designed for centralized unlearning usually cannot be transferred directly to FL. Furthermore, the collaborative nature of FL deeply



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

integrates individual client updates into the global model, making it highly challenging to isolate and remove a single participant's effect without access to the original raw data. FU also necessitates a broader operational scope: rather than just deleting individual data points, it often requires the elimination of entire clients or groups of historical updates. Thus, FU must operate with severely limited information while strictly adhering to the communication constraints and privacy guarantees inherent to the federated ecosystem.

In practice, the success of unlearning also depends strongly on data heterogeneity, model architecture, federation size, and the severity of the unlearning request. Different unlearning scenarios vary significantly, ranging from benign client withdrawal (clean-data deletion) to the removal of adversarial contributions like poisoned updates or backdoors. Navigating these diverse and complex conditions requires a robust unlearning framework that can systematically isolate targeted information without causing a catastrophic collapse of the global model's utility.

Our method builds on the teacher-student forget/retain principle proposed by (Kurmanji et al., 2023). We study its use in active federated unlearning, where forget and retain examples remain at clients and the two updates use separate, sample-weighted aggregation phases. Our formulation permits different $f$-divergences for the retain and forget objectives. We analyze the local curvature of this objective under stated assumptions and evaluate how divergence choice affects the forgetting utility trade-off under a fixed synchronization-phase budget. Our main contributions are as follows:

- Inspired by $f$-divergences, we proposed the forget/retain objective for active federated unlearning through client-local updates and standard sample-weighted aggregation, supporting both client-level and data-level deletion requests without server access to raw data. Our algorithm avoids catastrophic collapse and provides a stable unlearning process.
- We study the choice of $f$-divergence for forgetting while retaining KL-based alignment, and derive a local curvature characterization that makes the role of the chosen divergence explicit. We show that the choice of divergence has a substantial effect on optimization stability, retained-data utility, and forgetting behavior.
- We evaluate our federated unlearning method across backdoor removal, label-confusion removal, and privacy-oriented deletion on several datasets and federation sizes. We report retained utility together with forgetting metrics and identify settings in which the method performs well or exhibits trade-offs.

## 2 Related Work

In this section, we will provide an overview of federated unlearning works focusing on *active FU*. More related works are provided in Appendix A.

Federated unlearning (FU) can broadly be divided into two categories: *active unlearning* and *passive unlearning*. In the active setting, the client requesting data deletion actively engages in the unlearning process, helping the system mitigate its own contribution to the global model. On the other hand, passive unlearning assumes that the client wishing to be forgotten is either unavailable or unwilling to participate. (e.g. (Liu et al., 2021b), (Su & Li, 2023), (Li et al., 2023a), (Cao et al., 2023))

**Passive Federated Unlearning:** Most passive unlearning approaches attempt to reconstruct the model without access to the forgetting client (e.g., (Liu et al., 2021a), (Jiang et al., 2024), (Huynh et al., 2024), (Yuan et al., 2023)). However, these methods have fundamental limitations. Primarily, they focus on removing an entire client, which may not be practical in many scenarios. Additionally, they rely on estimating the gradients of the removed client and subtracting them from the final model, which can be error-prone and computationally expensive.

**Active Federated Unlearning:** We divide active unlearning into two categories: (1) settings in which the server acts purely as an *accumulator* (i.e., it only aggregates client updates), and (2) settings in which the server performs additional computation beyond averaging e.g., leveraging a server dataset or replay memory, maintaining auxiliary state, or applying extra procedures such as special operators.

**Only Accumulator:** (Halimi et al., 2023) proposed Federated Unlearning, where a client locally reverses its contribution by maximizing empirical loss under constraints, followed by limited retrain-

ing across remaining clients. (Khalil et al., 2025) proposes perturbing the model's weights, strong enough to push the model away from its current solution yet resilient enough to allow rapid recovery, followed by brief fine-tuning to restore performance.

(Dhasade et al., 2024) integrates dataset distillation into the unlearning process, allowing clients to generate compact representations used for gradient ascent unlearning. (Meerza et al., 2024) combines confusion based updates and salience aware masking to weaken model memory of specific data (like (Ma et al., 2022)) By simulating memory degradation and avoiding full retraining, it enables lightweight, instance to client level unlearning in federated settings.

(Li et al., 2023b) proposes a novel Federated Active Forgetting (FedAF) framework that leverages teacher-student learning to generate new memories for overwriting old ones, addressing fine-grained unlearning challenges while mitigating catastrophic forgetting in FL settings. The (Gu et al., 2024) introduces FedAU, a streamlined Federated Machine Unlearning framework that integrates a lightweight auxiliary unlearning module during the training process, using a linear operation to enable efficient unlearning at sample, class, and client levels without additional time-consuming steps. You can find beyond accumulator literature in Appendix A

**Positioning of our work:** Our work belongs to the active and aggregator-only FU setting where the server is not assumed to hold auxiliary data, perform replay-based recovery, or apply architecture specific operations beyond aggregation. Unlike methods based primarily on direct weight perturbation, gradient reversal, or auxiliary unlearning modules, $f-$`Fum` formulates unlearning as a min-max optimization problem using $f$-divergence objectives.

## 3 PRELIMINARIES

**$f$-divergence:** The $f$-divergences are information measures that generalize various divergences, such as Kullback-Leibler (KL) divergence, through the use of a convex generator function $f$. Given two discrete distributions $P = \{p_i\}_{i=1}^k$ and $Q = \{q_i\}_{i=1}^k$, the $f$-divergence between them is defined as:

$$D_f(P \parallel Q) := \sum_{i=1}^{k} q_i f\left(\frac{p_i}{q_i}\right) \tag{1}$$

where $f : (0, \infty) \to \mathbb{R}$ is a convex function with the property that $f(1) = 0$. In this work, we focus on KL-divergence, JS-divergence and $\chi^2$-divergence. Therefore, we consider $D_f \in \{D_{\mathrm{KL}}, D_{\mathrm{JS}}, D_{\chi^2}\}$. The extra explanation and rationale behind this selection for JS and $\chi^2$ divergences is provided in Appendix C.

**Federated unlearning setup:** Let $C$ denote the number of clients and $S = \bigcup_{c=1}^{C} S^{(c)}$ the distributed training set. For client $c$, an unlearning request specifies $S_F^{(c)} \subseteq S^{(c)}$, with retained set $S_R^{(c)} = S^{(c)} \setminus S_F^{(c)}$. We write

$$S_F = \bigcup_{c=1}^{C} S_F^{(c)}, \qquad S_R = \bigcup_{c=1}^{C} S_R^{(c)}. \tag{2}$$

A federated learning algorithm $\mathcal{A}$ produces the original model $\theta_0 = \mathcal{A}(S)$, while an unlearning algorithm $\mathcal{U}$ produces

$$\theta_U = \mathcal{U}(\theta_0, S_F). \tag{3}$$

The retained-data retraining reference is $\theta_R = \mathcal{A}(S_R)$.

We use predictive discrepancy $\Delta_{\mathrm{pred}}(\theta_U, \theta_R)$ to characterize approximate unlearning:

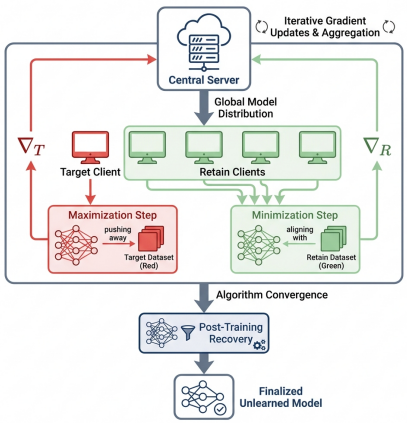
$$\Delta_{\mathrm{pred}}(\theta_U, \theta_R) \leq \epsilon_{\mathrm{unl}}, \tag{4}$$

where smaller $\epsilon_{\mathrm{unl}}$ indicates closer behavior to retained-data retraining.

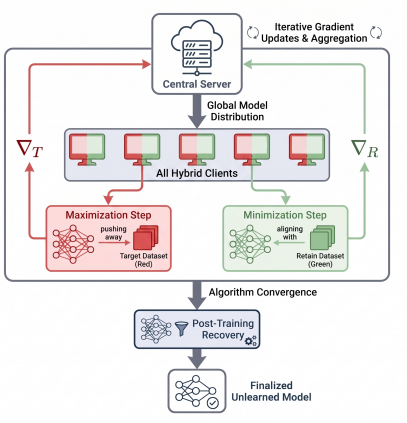
**Loss Functions:** Let $h(x; \theta) \in \Delta^{K-1}$ denote the predictive distribution. For student and teacher parameters $\theta^S$ and $\theta^T$, respectively, we define

$$d_f(x; \theta^S, \theta^T) = D_f\big(h(x; \theta^S) \| h(x; \theta^T)\big), \tag{5}$$

and use $\ell_f(h(x; \theta), \mathbf{Y}) = D_f(\mathbf{Y} \| h(x; \theta))$ as the supervised loss.




(a) Client-level unlearning.




(b) Data-level unlearning.

Figure 1: Overview of $f$−Fum under client-level and data-level unlearning. In client-level unlearning, the contribution of an entire target client is removed, whereas in data-level unlearning only selected samples are forgotten while the corresponding clients remain active. In both settings, forget-set maximization and retained-data minimization alternate for $T$ rounds, followed by retained-data post-training.

**Unlearning objectives:** We consider two evaluation regimes. In *robustness-oriented unlearning*, $S_F$ contains corrupted or poisoned samples, and effectiveness is measured by removal of the attack effect while preserving clean predictive utility. In *privacy-oriented unlearning*, $S_F$ contains samples requested for deletion, and residual information is evaluated using membership-inference attacks (MIAs).

For an attack $\hat{m}$ on the unlearned model,

$$\mathrm{MIA}(\theta_U) = \frac{1}{2}\left(\Pr_{x\sim S_F}[\hat{m}(\theta_U, x) = 1] + \Pr_{x'\sim S_{\text{test}}}[\hat{m}(\theta_U, x') = 0]\right), \tag{6}$$

with corresponding advantage

$$\mathrm{Adv}_{\mathrm{MIA}}(\theta_U) = |2\,\mathrm{MIA}(\theta_U) - 1|\,. \tag{7}$$

Values near zero indicate attack performance close to random guessing. The specific MIA variants used in our experiments are described in Section 5.

## 4 $f$−Fum Algorithm

In this section, we introduce our FU algorithm, $f$−Fum. In the $f$−Fum setting, a client may request unlearning due to poisoned data or privacy concerns, after which all clients and the server collaboratively perform the unlearning process, i.e., active unlearning. Our algorithm consists of three main steps: (i) Local Objectives, (ii) Federated Objectives, and (iii) Post-training Minimization. $f$−Fum performs $T$ max–min iterations, each comprising one forget-set maximization phase and one retained-data minimization phase, with a server aggregation after each phase. These iterations are followed by $T_{\min}$ minimization-only phases, giving $2T + T_{\min}$ synchronization phases in total.

Let $\theta$ denote the student model parameters. and $\theta_T$ be a reference model. Inspired by min-max problem, for client $c$, we define,

$$\mathcal{L}^{(c)}_{\max}(S^{(c)}_F, \theta, \theta_T) = \frac{1}{n^{(c)}_F} \sum_{x_f \in S^{(c)}_F} d_{f_F}(x_f; \theta, \theta_T),$$

$$\mathcal{L}^{(c)}_{\min}(S^{(c)}_R, \theta, \theta_T) = \frac{\alpha}{n^{(c)}_R} \sum_{x_r \in S^{(c)}_R} d_{f_R}(x_r; \theta, \theta_T) + \frac{\gamma}{n^{(c)}_R} \sum_{(x_r, y_r) \in S^{(c)}_R} \ell_{f_R}(h(x_r; \theta), y_r),$$




Figure 2: Catastrophic collapse under unconstrained forgetting. **Left:** the pre-trained model memorizes the poisoned borderline samples. **Middle:** f-divergence maximization without a retention anchor collapses the boundary and misclassifies clean samples. **Right:** recovery must effectively retrain from the degraded state.

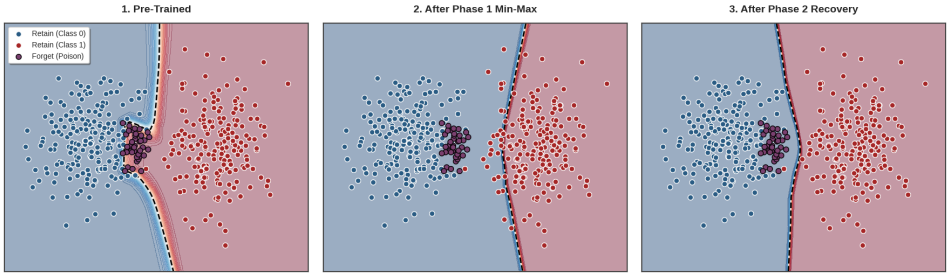



Figure 3: Stable unlearning under $f$−Fum. **Left:** the poisoned samples distort the pre-trained boundary. **Middle:** alternating min-max optimization removes the poison while retaining the global boundary structure. **Right:** post-training recovery smooths the residual distortion and restores clean class separation.

where $\alpha$ and $\gamma$ are loss weights. Inspired by a min-max objective, we aim to maximize the $f$-divergence between the outputs of the unlearned model and the reference model on the forgetting dataset, while simultaneously minimizing the $f$-divergence between their outputs on the remaining dataset. The $f$−Fum algorithm is presented in Algorithm 1 (provided in Appendix B). We further studied $f$−Fum from *theoretical perspective* in Appendix H, where we study the choice of $f$-divergence for forgetting, and derive a local curvature characterization that makes the role of the chosen divergence explicit. Next, we provide a toy example for motivation of our min-max algorithm, $f$−Fum algorithm.

**Toy Example; Min-Max Unlearning Dynamics:** We use a 2D toy classification task to visualize why $f$−Fum requires both alternating min-max optimization and post-training recovery. The retained set $S_R$ consists of two well-separated classes, while the forget set $S_F$ contains maliciously labeled borderline samples placed near the decision boundary. Pre-training on this poisoned data warps the boundary toward the wrong class.

Figure 2 shows that using only the maximization step leads to catastrophic collapse. Although the poisoned samples are pushed away from the learned hypothesis, the absence of a retention constraint causes the decision boundary to move indiscriminately, misclassifying clean retained samples. Subsequent recovery therefore resembles retraining rather than efficient unlearning.

In contrast, Figure 3 shows that max-min avoids this collapse. During Phase 1, the maximization step removes the influence of $S_F$, while the minimization step on $S_R$ anchors the boundary and preserves the global structure of the model. This produces only a localized temporary distortion. During Phase 2, post-training minimization smooths this distortion and restores a clean decision boundary. These results illustrate that the alternating minimization step is essential for stable unlearning, while the post-training phase acts as lightweight recovery rather than full retraining.

## 5 EXPERIMENTS

**Datasets & Models.** We conduct experiments on CIFAR-10/100, FashionMNIST, and MNIST using three model architectures: (i) ResNet-18, (ii) a lightweight CNN consisting of two convolutional layers with Layer Normalization, and (iii) LeNet5. All experiments were conducted on a cluster with one NVIDIA H100 GPU, 10 CPU cores, and 512 GB RAM. Unless stated otherwise, each reported unlearning result is averaged over three independent runs of the unlearning procedure.

**Experimental repetitions and model selection.** Within each dataset and federation setting, all methods are assigned the same predefined budget $B$ (Appendix D) of server-client synchronization phases. For $f-\texttt{Fum}$, a forget-retain iteration uses two phases and each post-training round uses one, so $B = 2 \times T + T_{\text{post}}$. Among checkpoints produced within this budget, we select the one with the highest clean accuracy on a held-out validation set. We evaluate all reported metrics at that same checkpoint. Neither test-set performance nor forgetting or MIA metrics are used for selection.

Across runs, stochastic operations during unlearning, such as minibatch or client sampling where applicable, use independent random seeds. Consequently, the reported standard deviations characterize variability conditional on the fixed pretrained checkpoint and deletion request. MIA samples are also selected independently for each run, so the corresponding standard deviations additionally include evaluation-sampling variability.

**Baselines.** We compare our method against: (1) **NoT** (Khalil et al., 2025); (2) **MoDE** (Zhao et al., 2024); and (3) **Halimi** (Halimi et al., 2023). Our codes are available here.

**Privacy Evaluation.** For privacy evaluation, we consider the membership inference attack (MIA) score for our models. We have different kind of MIAs, but we focused on the simple one and Yeom (Yeom et al., 2018) for CIFAR-100. The attacks evaluate whether the model retains information about the forget set by comparing its behavior on forgotten samples and non-member test samples. If the model treats forgotten data similarly to data it never saw, the unlearning was successful and privacy is preserved, with the membership inference accuracy approaching 50% (random guessing) being the ideal outcome; however, if the accuracy significantly deviates from 50%, this indicates the model's responses reveal distinguishable patterns for forgotten data, suggesting privacy leakage and incomplete removal of the data's influence. Another important fact is that under our setup we remove the clean data, based on the literature of Differential Privacy (DP) (El Ouadrhiri & Abdelhadi, 2022), we know that working with noisy data can help us to increase our privacy budget which should be avoided in the FU experiments.

**Utility Evaluation.** For the backdoor and label-confusion scenarios, we select checkpoints using clean accuracy on a held-out validation set that is disjoint from the optimization data. We then report clean accuracy for the selected checkpoint on the separate official test set; test-set results are not used for checkpoint selection. Clean test accuracy measures predictive utility, while attack success rate on triggered test examples, where reported, measures removal of backdoor behavior. Within each experimental setting, all methods receive the same predefined budget of server-client synchronization phases. Equal phase budgets do not imply equal transmitted bits, local computation, or wall-clock time; these quantities were not measured consistently for every baseline.

**Client-Level Unlearning:** Following (Halimi et al., 2023), we simulate full removal of one client, where the forgotten data may be **backdoored**, **label-confused**, or **clean** (in the privacy setting). Specifically, we consider two configurations: $N = 5$ clients with one client holding $66\%$ corrupted data, and $N = 10$ clients with one client holding $80\%$ corrupted data. Backdoor corruption follows the $3 \times 3$ pixel trigger pattern introduced by (Gu et al., 2019), while label confusion is implemented via deterministic class mapping to simulate annotation errors. The results are available in Table 1 and 2 . For each method, all reported metrics are computed at the checkpoint selected by held-out validation accuracy from those produced within its predefined synchronization-phase budget.

**Data-Level Unlearning:** Inspired by (Wang et al., 2024a), we evaluate unlearning when each client requests partial deletion of their local data. In this setup, forgotten samples are either clean, backdoored, or label-confused, and we vary the number of clients ($N \in \{5, 10\}$) as well as the fraction of forgotten data per client (2%). More fractions of forgotten data per client are provided in Appendix E. For CIFAR-100 removed percentage is 6% and 10% (Table 1) This setting reflects more realistic distributed deletion requests involving small, scattered portions of client datasets. For privacy

Table 1: CIFAR-100 results for client-level and data-level unlearning under backdoor, confused-data, and privacy scenarios. For client-level unlearning, the setting denotes the number of clients $N$. For data-level unlearning, $N/r$ denotes $N$ clients with $r\%$ of local data removed from each client. Acc.: higher is better; ASR: lower is better; MIA is best when closest to 50.

| Scenario | Setting | Method | Backdoor | | | Confused | | Privacy | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Pre. Acc. | Pre. ASR | Unl. Acc. | Unl. ASR | Pre. Acc. | Unl. Acc. | Pre. Acc. | Unl. Acc. | MIA | Yeom Pre. | Yeom Unl. |
| Client-Level | 5 | Ours (KL) | 47.90 | 72.05 | $50.10 \pm 0.40$ | $5.80 \pm 0.54$ | 45.62 | $49.80 \pm 0.44$ | 53.08 | $\mathbf{51.70 \pm 0.53}$ | $56.70 \pm 0.56$ | 65.75 | $\mathbf{57.40 \pm 1.10}$ |
| | | Ours (JS) | | | $\mathbf{50.22 \pm 0.43}$ | $7.98 \pm 0.40$ | | $49.48 \pm 0.32$ | | $51.44 \pm 0.33$ | $\mathbf{52.23 \pm 0.56}$ | | $58.59 \pm 0.17$ |
| | | NoT | | | $44.40 \pm 1.39$ | $\mathbf{0.59 \pm 0.14}$ | | $35.60 \pm 1.25$ | | $37.74 \pm 1.99$ | $53.35 \pm 1.03$ | | $59.32 \pm 0.95$ |
| | | Halimi | | | $48.01 \pm 0.48$ | $5.34 \pm 0.90$ | | $\mathbf{49.92 \pm 0.68}$ | | $51.00 \pm 0.44$ | $57.29 \pm 0.62$ | | $60.73 \pm 0.32$ |
| | 10 | Ours (KL) | 48.21 | 77.24 | $\mathbf{49.44 \pm 0.72}$ | $2.16 \pm 0.69$ | 45.32 | $\mathbf{49.12 \pm 0.48}$ | 52.11 | $\mathbf{51.38 \pm 0.95}$ | $58.00 \pm 0.51$ | 63.96 | $57.92 \pm 1.18$ |
| | | Ours (JS) | | | $49.06 \pm 0.11$ | $1.04 \pm 0.50$ | | $48.48 \pm 0.13$ | | $51.36 \pm 0.15$ | $56.60 \pm 0.25$ | | $57.71 \pm 0.14$ |
| | | NoT | | | $46.18 \pm 2.14$ | $\mathbf{0.38 \pm 0.11}$ | | $42.28 \pm 0.76$ | | $45.50 \pm 0.94$ | $\mathbf{49.20 \pm 1.22}$ | | $\mathbf{52.52 \pm 1.18}$ |
| | | Halimi | | | $49.40 \pm 0.34$ | $1.36 \pm 0.64$ | | $48.42 \pm 0.42$ | | $50.48 \pm 1.05$ | $61.00 \pm 0.72$ | | $59.66 \pm 0.58$ |
| Data-Level | 5/6 | Ours (KL) | 47.20 | 80.61 | $\mathbf{52.75 \pm 0.68}$ | $0.69 \pm 0.34$ | 51.24 | $\mathbf{53.18 \pm 1.13}$ | 52.32 | $51.14 \pm 1.07$ | $\mathbf{49.80 \pm 0.59}$ | 58.11 | $\mathbf{51.17 \pm 0.56}$ |
| | | Ours (JS) | | | $51.53 \pm 0.28$ | $0.77 \pm 0.29$ | | $52.48 \pm 0.18$ | | $50.98 \pm 0.25$ | $48.80 \pm 0.25$ | | $51.36 \pm 0.29$ |
| | | NoT | | | $35.30 \pm 1.08$ | $2.63 \pm 1.39$ | | $40.12 \pm 1.43$ | | $49.24 \pm 1.34$ | $50.44 \pm 0.95$ | | $52.18 \pm 1.44$ |
| | | Halimi | | | $52.10 \pm 0.45$ | $\mathbf{0.36 \pm 0.08}$ | | $53.16 \pm 0.51$ | | $\mathbf{52.56 \pm 0.73}$ | $54.44 \pm 0.61$ | | $56.16 \pm 0.38$ |
| | 5/10 | Ours (KL) | 45.16 | 81.07 | $\mathbf{51.03 \pm 0.82}$ | $\mathbf{0.34 \pm 0.17}$ | 49.42 | $\mathbf{50.76 \pm 0.96}$ | 53.11 | $\mathbf{51.05 \pm 0.81}$ | $\mathbf{49.89 \pm 1.02}$ | 59.08 | $\mathbf{50.10 \pm 0.58}$ |
| | | Ours (JS) | | | $50.35 \pm 0.34$ | $0.68 \pm 0.31$ | | $49.65 \pm 0.48$ | | $50.98 \pm 0.24$ | $52.60 \pm 0.49$ | | $53.57 \pm 0.43$ |
| | | NoT | | | $38.58 \pm 1.57$ | $34.30 \pm 1.05$ | | $39.93 \pm 0.57$ | | $50.30 \pm 0.84$ | $\mathbf{50.11 \pm 1.24}$ | | $49.63 \pm 1.33$ |
| | | Halimi | | | $50.23 \pm 0.46$ | $0.93 \pm 0.41$ | | $50.00 \pm 0.66$ | | $50.34 \pm 0.70$ | $56.33 \pm 0.51$ | | $57.50 \pm 0.70$ |
| | 10/6 | Ours (KL) | 47.62 | 74.09 | $48.00 \pm 0.45$ | $0.25 \pm 0.05$ | 48.43 | $\mathbf{50.64 \pm 1.29}$ | 51.08 | $48.64 \pm 0.70$ | $\mathbf{49.00 \pm 0.95}$ | 56.00 | $51.46 \pm 0.30$ |
| | | Ours (JS) | | | $47.50 \pm 0.04$ | $0.25 \pm 0.19$ | | $48.22 \pm 0.23$ | | $47.98 \pm 0.24$ | $51.45 \pm 0.20$ | | $51.86 \pm 0.17$ |
| | | NoT | | | $34.10 \pm 0.93$ | $21.06 \pm 1.48$ | | $36.78 \pm 1.06$ | | $46.30 \pm 0.72$ | $48.60 \pm 1.96$ | | $\mathbf{49.94 \pm 1.07}$ |
| | | Halimi | | | $\mathbf{50.28 \pm 0.42}$ | $\mathbf{0.20 \pm 0.04}$ | | $49.60 \pm 0.83$ | | $\mathbf{49.96 \pm 0.91}$ | $54.60 \pm 0.37$ | | $54.25 \pm 0.86$ |
| | 10/10 | Ours (KL) | 45.30 | 80.61 | $\mathbf{47.94 \pm 1.46}$ | $\mathbf{0.28 \pm 0.08}$ | 45.68 | $\mathbf{48.20 \pm 0.83}$ | 50.48 | $\mathbf{47.92 \pm 0.64}$ | $51.58 \pm 1.20$ | 55.47 | $49.56 \pm 0.79$ |
| | | Ours (JS) | | | $45.88 \pm 0.34$ | $0.51 \pm 0.32$ | | $45.70 \pm 0.24$ | | $44.32 \pm 0.39$ | $48.83 \pm 0.34$ | | $\mathbf{50.03 \pm 0.24}$ |
| | | NoT | | | $35.24 \pm 0.95$ | $10.61 \pm 0.84$ | | $37.83 \pm 1.84$ | | $45.56 \pm 1.39$ | $\mathbf{50.00 \pm 1.41}$ | | $49.84 \pm 1.57$ |
| | | Halimi | | | $46.96 \pm 0.79$ | $0.48 \pm 0.13$ | | $47.18 \pm 0.77$ | | $47.81 \pm 0.98$ | $53.10 \pm 0.78$ | | $53.64 \pm 0.81$ |

Table 2: Comparison of client-level unlearning performance under backdoor and confused-data scenarios. In each setting, one client is removed, with 66% corrupted data when $N = 5$ and 80% corrupted data when $N = 10$.

| Method | $N$ | Backdoor | | | | | | Confused | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | CIFAR-10 | | MNIST | | Fashion-MNIST | | CIFAR-10 | | MNIST | | Fashion-MNIST | |
| | | Pre. | Unl. | Pre. | Unl. | Pre. | Unl. | Pre. | Unl. | Pre. | Unl. | Pre. | Unl. |
| Ours (KL) | 10 | 63.64 | $71.84 \pm 1.29$ | 99.14 | $99.16 \pm 0.34$ | 73.85 | $73.10 \pm 2.10$ | 63.64 | $71.24 \pm 3.20$ | 98.93 | $99.11 \pm 0.12$ | 78.28 | $77.62 \pm 1.25$ |
| Ours (JS) | | | $72.74 \pm 0.19$ | | $98.99 \pm 0.27$ | | $\mathbf{75.13} \pm 1.45$ | | $72.90 \pm 0.44$ | | $99.08 \pm 0.19$ | | $\mathbf{77.87} \pm 1.02$ |
| Ours ($\chi^2$) | | | $\mathbf{72.92} \pm 0.89$ | | $\mathbf{99.21} \pm 0.29$ | | $73.82 \pm 1.04$ | | $\mathbf{77.43} \pm 1.23$ | | $99.06 \pm 0.27$ | | $77.43 \pm 1.51$ |
| NoT | | | $67.00 \pm 2.65$ | | $82.60 \pm 1.46$ | | $46.82 \pm 3.36$ | | $69.20 \pm 1.98$ | | $88.31 \pm 1.84$ | | $76.07 \pm 1.64$ |
| Halimi | | | $72.56 \pm 1.10$ | | $\mathbf{99.21} \pm 0.25$ | | $73.32 \pm 1.77$ | | $72.66 \pm 1.56$ | | $\mathbf{99.13} \pm 0.15$ | | $77.73 \pm 1.85$ |
| MoDE | | | $70.96 \pm 1.43$ | | $98.74 \pm 0.72$ | | $72.07 \pm 1.73$ | | $70.96 \pm 3.43$ | | $81.02 \pm 2.23$ | | $77.68 \pm 1.39$ |
| Ours (KL) | 5 | 52.22 | $69.62 \pm 1.35$ | 98.98 | $98.88 \pm 0.54$ | 31.32 | $39.55 \pm 2.95$ | 52.22 | $67.36 \pm 1.17$ | 97.93 | $98.66 \pm 0.14$ | 66.58 | $58.58 \pm 3.63$ |
| Ours (JS) | | | $\mathbf{77.50} \pm 0.32$ | | $98.81 \pm 0.09$ | | $55.93 \pm 1.08$ | | $\mathbf{77.22} \pm 0.91$ | | $98.14 \pm 0.06$ | | $\mathbf{78.12} \pm 1.40$ |
| Ours ($\chi^2$) | | | $77.22 \pm 1.43$ | | $\mathbf{98.95} \pm 0.91$ | | $48.48 \pm 3.12$ | | $61.07 \pm 2.40$ | | $\mathbf{99.13} \pm 0.34$ | | $61.07 \pm 3.93$ |
| NoT | | | $70.16 \pm 1.98$ | | $89.21 \pm 2.66$ | | $37.45 \pm 2.51$ | | $68.26 \pm 1.35$ | | $83.98 \pm 2.61$ | | $32.76 \pm 4.54$ |
| Halimi | | | $66.76 \pm 1.03$ | | $98.81 \pm 0.30$ | | $54.02 \pm 2.78$ | | $66.76 \pm 1.65$ | | $98.93 \pm 0.64$ | | $62.60 \pm 1.72$ |
| MoDE | | | $65.44 \pm 1.78$ | | $98.49 \pm 0.63$ | | $\mathbf{64.30} \pm 3.57$ | | $69.44 \pm 1.82$ | | $98.36 \pm 0.84$ | | $70.20 \pm 3.33$ |

oriented case we do not corrupt data as we explained why in Client-Level Unlearning discussion. The results are available in Tables 1 and 3. For the more difficult situations and unlearning settings, you can check their result in the Appendix E.

## 6 Discussion

**The effect of choosing $f$-divergence:** The choice of divergence changes the observed forgetting utility trade-off, but no variant is best in every setting. In client-level CIFAR-10 backdoor removal (Table 2), KL-JS[1] achieves $72.74\%$ clean accuracy with $N = 10$, compared with $71.84\%$ for KL-KL and $72.92\%$ for KL-$\chi^2$. With $N = 5$, KL-JS achieves $77.50\%$, compared with $69.62\%$ for KL-KL and $77.22\%$ for KL-$\chi^2$. CIFAR-100 shows why divergence choice must also be evaluated against forgetting metrics (Table 1). In client-level backdoor removal with $N = 10$, KL-JS achieves lower attack success rate (ASR) than KL-KL ($1.04\%$ versus $2.16\%$) at similar clean accuracy. With $N = 5$,

[1]In all configurations, the first loss component is fixed to KL. Thus, table notation **Ours ($D$)** denotes $(\mathrm{KL}, D, D)$; e.g., **Ours (JS)** corresponds to KL-JS in the discussion.

Table 3: Comparison of data-level unlearning performance under backdoor and confused-data scenarios. In all settings, 2% of the data is removed from each client.

| Method | $N$ | Backdoor | | | | Confused | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | CIFAR-10 | | Fashion-MNIST | | CIFAR-10 | | MNIST | | Fashion-MNIST | |
| | | Pre. | Unl. | Pre. | Unl. | Pre. | Unl. | Pre. | Unl. | Pre. | Unl. |
| Ours (KL) | | | 68.62 ± 1.54 | | 69.70 ± 0.06 | | 70.66 ± 0.15 | | 97.75 ± 0.61 | | 67.67 ± 0.34 |
| Ours (JS) | | | **73.40** ± 0.86 | | 76.70 ± 0.83 | | **73.98** ± 0.11 | | **98.81** ± 0.23 | | **77.40** ± 0.04 |
| Ours ($\chi^2$) | 10 | 53.38 | 57.86 ± 2.80 | 49.07 | 61.10 ± 1.37 | 67.02 | 69.10 ± 1.21 | 93.49 | 97.33 ± 0.90 | 42.35 | 64.60 ± 2.21 |
| NoT | | | 62.00 ± 2.12 | | 12.37 ± 4.32 | | 69.32 ± 1.61 | | 86.58 ± 2.92 | | 39.92 ± 5.13 |
| Halimi | | | 62.00 ± 1.27 | | **79.53** ± 0.39 | | 70.00 ± 0.62 | | 97.80 ± 0.76 | | 68.52 ± 0.60 |
| Ours (KL) | | | 66.64 ± 0.94 | | 71.58 ± 0.16 | | 73.00 ± 0.74 | | 95.69 ± 0.51 | | 70.70 ± 0.20 |
| Ours (JS) | | | **74.50** ± 0.72 | | **77.95** ± 0.43 | | **76.62** ± 0.55 | | **98.41** ± 0.14 | | 77.85 ± 1.05 |
| Ours ($\chi^2$) | 5 | 58.54 | 67.26 ± 1.80 | 45.42 | 48.17 ± 2.94 | 68.70 | 72.12 ± 2.11 | 84.06 | 94.05 ± 1.07 | 50.77 | 67.80 ± 3.80 |
| NoT | | | 69.92 ± 1.93 | | 52.82 ± 1.78 | | 72.00 ± 2.00 | | 81.64 ± 3.91 | | 68.40 ± 2.91 |
| Halimi | | | 68.00 ± 0.96 | | 71.22 ± 0.58 | | 74.00 ± 0.83 | | 73.52 ± 1.16 | | **95.48** ± 0.62 |

Table 4: Data-level privacy-oriented unlearning: pretrained (Pre.), unlearned (Unl.) accuracy, and MIA performance when 2% of clean data is removed from each client. MoDE is omitted because it is explicitly designed for client-level unlearning.

| Method | $N$ | CIFAR-10 | | | MNIST | | | Fashion-MNIST | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Pre. | Unl. | MIA | Pre. | Unl. | MIA | Pre. | Unl. | MIA |
| Ours (KL) | | | 78.38 ± 0.66 | 60.77 ± 0.90 | | 77.40 ± 1.25 | 59.89 ± 6.50 | | 79.62 ± 0.11 | 49.90 ± 0.26 |
| Ours (JS) | | | 78.32 ± 0.34 | 58.44 ± 1.10 | | **77.86** ± 0.87 | 59.33 ± 4.32 | | **80.02** ± 0.34 | **50.20** ± 0.12 |
| Ours ($\chi^2$) | 10 | 79.00 | 78.12 ± 1.83 | 60.56 ± 1.43 | 99.18 | 77.56 ± 1.63 | 59.78 ± 6.78 | 80.53 | 79.28 ± 0.87 | 49.70 ± 0.55 |
| NoT | | | 73.30 ± 2.31 | **49.66** ± 3.45 | | 72.92 ± 2.73 | **50.11** ± 2.33 | | 57.77 ± 4.50 | 49.50 ± 2.73 |
| Halimi | | | **79.02** ± 0.86 | 59.33 ± 2.34 | | 77.72 ± 1.03 | 59.67 ± 3.02 | | 79.65 ± 0.27 | 49.20 ± 0.47 |
| Ours (KL) | | | **78.68** ± 0.81 | 58.55 ± 0.56 | | 77.66 ± 1.24 | 59.33 ± 5.55 | | 79.33 ± 0.50 | 49.33 ± 0.19 |
| Ours (JS) | | | 77.50 ± 0.23 | 59.00 ± 0.61 | | 77.72 ± 1.01 | 58.00 ± 4.37 | | **79.80** ± 0.19 | 48.38 ± 0.22 |
| Ours ($\chi^2$) | 5 | 80.00 | 77.66 ± 0.90 | 60.44 ± 1.13 | 98.93 | 77.52 ± 2.32 | 61.00 ± 5.18 | 79.92 | 79.10 ± 1.15 | 48.57 ± 1.23 |
| NoT | | | 72.78 ± 3.12 | **50.00** ± 0.00 | | 72.96 ± 2.45 | **50.89** ± 2.50 | | 72.03 ± 2.74 | 49.17 ± 2.76 |
| Halimi | | | 77.98 ± 1.49 | 61.88 ± 2.23 | | **77.88** ± 1.17 | 61.89 ± 3.96 | | 79.18 ± 0.64 | **50.19** ± 0.75 |

their ASR ordering reverses: KL-KL achieves 5.80% ASR versus 7.98% for KL-JS. These results support setting-dependent divergence selection rather than a uniformly optimal choice. Although JS divergence has a bounded value, that property alone does not establish bounded parameter gradients or guarantee stable training.

**Data-level versus client-level unlearning:** Client-level deletion removes one participant's contribution, whereas data-level deletion targets a fraction of examples held by each affected client. In the CIFAR-10 data-level backdoor setting, removing $2\%$ of local data with $N = 10$ raises clean accuracy from $53.38\%$ before unlearning to $73.40\%$ with KL-JS (Table 3). Halimi reaches $62.00\%$ in the same setting, so this result does not establish dominance over the baselines. CIFAR-100 includes larger data-level deletion fractions of $6\%$ and $10\%$ (Table 1). With five clients and $10\%$ of local data deleted, KL-KL achieves $0.34\%$ ASR and $51.03\%$ clean accuracy, compared with $34.30\%$ ASR and $38.58\%$ accuracy for NoT. The CIFAR-100 results illustrate why we do not use ASR to compare methods across datasets. In client-level unlearning with $N = 5$, NoT achieves a lower ASR than KL-KL (0.59% versus 5.80%), but also lower clean test accuracy (44.40% versus 50.10%). The same pattern appears with $N = 10$: NoT achieves 0.38% ASR and 46.18% accuracy, compared with 2.16% ASR and 49.44% accuracy for KL-KL (Table 1). A low ASR can arise when a model rarely predicts the attack target, even if its useful predictions have deteriorated; these results do not establish that the model has unlearned the poison data. Motivated by this trade-off, we report clean test accuracy for the other dataset comparisons and interpret those results as comparisons of retained utility.

For clean-data deletion, predictive utility and membership-inference attack (MIA) accuracy should be considered together (Tables 1, 4, 5). On CIFAR-100 client-level deletion with $N = 5$, KL-JS gives MIA accuracy of $52.23\%$, closer to random guessing than KL-KL's $56.70\%$. With $N = 10$, NoT's $49.20\%$ is closer to random guessing than KL-JS's $56.60\%$. Thus, the privacy results do not support

Table 5: Client-level privacy-oriented unlearning: pretrained (Pre.), unlearned (Unl.) accuracy, and MIA performance when one client containing only clean data is removed.

| Method | $N$ | CIFAR-10 | | | MNIST | | | Fashion-MNIST | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Pre. | Unl. | MIA | Pre. | Unl. | MIA | Pre. | Unl. | MIA |
| Ours (KL) | 10 | 78.04 | $73.74 \pm 2.40$ | **52.40** $\pm 1.65$ | 99.14 | $99.20 \pm 0.03$ | $50.33 \pm 0.57$ | 81.58 | **80.37** $\pm 2.15$ | $52.67 \pm 3.58$ |
| Ours (JS) | | | $63.16 \pm 2.90$ | $55.20 \pm 5.10$ | | $99.13 \pm 0.04$ | **50.00** $\pm 0.42$ | | $80.25 \pm 1.30$ | $46.67 \pm 2.35$ |
| Ours ($\chi^2$) | | | $77.66 \pm 3.12$ | $64.60 \pm 3.40$ | | **99.25** $\pm 0.06$ | $50.50 \pm 0.58$ | | $79.67 \pm 1.78$ | $54.83 \pm 2.50$ |
| NoT | | | $73.48 \pm 1.83$ | $52.80 \pm 2.74$ | | $95.25 \pm 0.12$ | $49.50 \pm 0.85$ | | $41.22 \pm 0.95$ | **48.67** $\pm 2.15$ |
| Halimi | | | **78.88** $\pm 1.96$ | $64.80 \pm 4.20$ | | $99.21 \pm 0.05$ | **50.00** $\pm 1.50$ | | $80.15 \pm 1.12$ | $54.00 \pm 2.45$ |
| MoDE | | | $72.30 \pm 3.12$ | $64.20 \pm 6.30$ | | $98.98 \pm 0.08$ | $51.00 \pm 2.72$ | | $73.48 \pm 1.60$ | $44.83 \pm 2.25$ |
| Ours (KL) | 5 | 79.72 | **78.92** $\pm 1.38$ | $63.60 \pm 6.25$ | 98.98 | $98.88 \pm 0.44$ | $50.75 \pm 0.12$ | 81.33 | $80.80 \pm 2.35$ | $49.08 \pm 1.05$ |
| Ours (JS) | | | $78.68 \pm 0.87$ | **49.90** $\pm 5.13$ | | **99.13** $\pm 0.05$ | **50.25** $\pm 0.30$ | | **81.32** $\pm 1.25$ | $48.42 \pm 1.15$ |
| Ours ($\chi^2$) | | | $78.24 \pm 2.10$ | $63.40 \pm 4.32$ | | $98.88 \pm 0.04$ | $49.75 \pm 0.26$ | | $80.57 \pm 1.85$ | $50.75 \pm 1.40$ |
| NoT | | | $72.94 \pm 2.54$ | $55.10 \pm 3.45$ | | $96.59 \pm 1.15$ | $48.75 \pm 0.55$ | | $38.95 \pm 0.92$ | $51.33 \pm 1.70$ |
| Halimi | | | $77.52 \pm 1.14$ | $64.20 \pm 4.56$ | | $98.96 \pm 0.06$ | $50.67 \pm 0.35$ | | $80.65 \pm 1.08$ | **49.67** $\pm 1.25$ |
| MoDE | | | $70.60 \pm 3.86$ | $58.40 \pm 4.62$ | | $98.58 \pm 0.49$ | $49.12 \pm 0.44$ | | $68.77 \pm 1.45$ | $50.25 \pm 1.85$ |

a general advantage for one method across deletion settings. MIA accuracy near $50\%$ measures the outcome of the reported attack; it is not a guarantee that all information about the deleted data has been removed.

**Comparison with baseline methods:** The baselines reveal different trade-offs across datasets. For client-level MNIST backdoor removal with $N = 10$ (Table 2), NoT achieves $82.60\%$ clean accuracy, while our KL-$\chi^2$ variant and Halimi both achieve $99.21\%$. On CIFAR-100 client-level backdoor removal, NoT achieves lower ASR than our variants but also lower clean accuracy: for $N = 5$ and $N = 10$, its ASR is $0.59\%$ and $0.38\%$, while its clean accuracy is $44.40\%$ and $46.18\%$, respectively (Table 1). This pattern is consistent with weight negation affecting useful predictions as well as backdoor behavior, although these results do not isolate the cause of the accuracy loss.

Halimi (Halimi et al., 2023) is competitive in several settings. On CIFAR-10 client-level backdoor removal with $N = 10$, it achieves $72.56\%$ clean accuracy, compared with $72.74\%$ for our KL-JS variant. In contrast, for data-level label-confusion removal on MNIST with $N = 5$, KL-JS achieves $98.41\%$ versus Halimi's $73.52\%$ (Table 3). MoDE (Zhao et al., 2024) also has mixed results: on CIFAR-10 client-level backdoor removal with $N = 10$, it achieves $70.96\%$ versus $72.92\%$ for our KL-$\chi^2$ variant, but on Fashion-MNIST with $N = 5$ in client-level backdoor removal , MoDE achieves $64.30\%$ versus $55.93\%$ for our best variant in that column (Table 2). These comparisons identify settings where $f-$`Fum` performs well and settings where a baseline performs better. Additional results appear in Appendix E.

## 7 Conclusion and Future Works

We introduced $f-$`Fum`, an active federated unlearning framework formulated as a min–max optimization problem using $f$-divergences such as Jensen–Shannon and $\chi^2$. The method increases the discrepancy between the unlearned and original models on forgotten data while preserving performance on retained data, thereby addressing both privacy-oriented and robustness-oriented unlearning in decentralized settings. Overall, $f-$`Fum` provides a principled and flexible mechanism for balancing forgetting effectiveness and retained utility with limited trade-offs. Future work includes extending the framework to $\alpha$-Rényi divergences and combining it with the theoretical framework of (Ding et al., 2024) to derive stronger guarantees.

### AI USE STATEMENT

Generative AI tools were used solely for limited editorial and presentation support. Specifically, they were used to assist with rewriting and improving the clarity, grammar, and presentation of portions of the manuscript, and to assist in generating and refining scientific figures based on descriptions provided by the authors.

All research ideas, methodological choices, mathematical derivations, theoretical results, experimental design, analyses, interpretations, and conclusions were developed and verified by the authors. The authors reviewed and take full responsibility for all AI-assisted text and figures included in the paper.

### ETHICS STATEMENT

This work studies federated machine unlearning with the aim of reducing the influence of data requested for deletion while preserving model utility. Our experiments use only standard public benchmark datasets and simulated deletion, poisoning, and corruption scenarios; no private or personally identifiable user data are used. Membership-inference attacks are employed solely as evaluation tools and should not be interpreted as formal privacy guarantees. Likewise, the proposed method does not by itself certify exact deletion, differential privacy, or legal compliance with a right to erasure. All results should therefore be interpreted within the assumptions and evaluation settings stated in the paper.

### REPRODUCIBILITY STATEMENT

All experiments use publicly available benchmark datasets, including CIFAR-10, CIFAR-100, MNIST, and FashionMNIST. We report the model architectures, federated-learning configurations, client- and data-level unlearning settings, data-corruption procedures, evaluation metrics, and optimization hyperparameters used in the experiments. Implementation details such as batch size, learning rates, numbers of pre-training and unlearning rounds, maximization/minimization schedules, and post-training configurations are provided in the appendix. The implementation is based on the Flower federated-learning framework. Code for reproducing the experiments is provided through the anonymous repository accompanying the submission. The repository contains the implementation of the proposed method and the experimental pipeline used to generate the reported results.

## REFERENCES


Gholamali Aminian, Amirhossien Bagheri, Mahyar JafariNodeh, Radmehr Karimian, and Mohammad-Hossein Yassaee. Robust semi-supervised learning via $f$-divergence and $\alpha$-rényi divergence, 2024. URL https://arxiv.org/abs/2405.00454.

Peter L. Bartlett, Michael I. Jordan, and Jon D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006. doi: 10.1198/016214505000000907.

Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Kwing Hei Li, Titouan Parcollet, Pedro Porto Buarque de Gusmão, and Nicholas D. Lane. Flower: A friendly federated learning research framework, 2022. URL https://arxiv.org/abs/2007.14390.

Jacopo Bonato, Marco Cotogni, and Luigi Sabetta. Is retain set all you need in machine unlearning? restoring performance of unlearned models with out-of-distribution images. In *European Conference on Computer Vision*, pp. 1–19. Springer, 2025.

Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning, 2018. URL https://arxiv.org/abs/1606.04838.

Xiaoyu Cao, Jinyuan Jia, Zaixi Zhang, and Neil Zhenqiang Gong. Fedrecover: Recovering from poisoning attacks in federated learning using historical information. In *2023 IEEE Symposium on Security and Privacy (SP)*, pp. 1366–1383. IEEE, 2023.

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pp. 463–480, 2015a. doi: 10.1109/SP.2015.35.

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, pp. 463–480. IEEE, 2015b.

Tianshi Che, Yang Zhou, Zijie Zhang, Lingjuan Lyu, Ji Liu, Da Yan, Dejing Dou, and Jun Huan. Fast federated machine unlearning with nonlinear functional theory. In *International conference on machine learning*, pp. 4241–4268. PMLR, 2023.

Zhipeng Deng, Luyang Luo, and Hao Chen. Enable the right to be forgotten with federated client unlearning in medical imaging. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 240–250. Springer, 2024.

Akash Dhasade, Yaohong Ding, Song Guo, Anne marie Kermarrec, Martijn De Vos, and Leijie Wu. Quickdrop: Efficient federated unlearning by integrated dataset distillation, 2024. URL https://arxiv.org/abs/2311.15603.

Meng Ding, Rohan Sharma, Changyou Chen, Jinhui Xu, and Kaiyi Ji. Understanding fine-tuning in approximate unlearning: A theoretical perspective. *arXiv preprint arXiv:2410.03833*, 2024.

John Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization, 2020. URL https://arxiv.org/abs/1810.08750.

Ahmed El Ouadrhiri and Ahmed Abdelhadi. Differential privacy for deep and federated learning: A survey. *IEEE access*, 10:22359–22380, 2022.

EU. General data protection regulation (GDPR), April 2016. URL https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32016R. Official Journal of the European Union, L 119/1.

Kristian Georgiev, Roy Rinberg, Sung Min Park, Shivam Garg, Andrew Ilyas, Aleksander Madry, and Seth Neel. Attribute-to-delete: Machine unlearning via datamodel matching, 2024. URL https://arxiv.org/abs/2410.23232.

Saeed Ghadimi and Guanghui Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming, 2013. URL https://arxiv.org/abs/1309.5549.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. URL https://arxiv.org/abs/1406.2661.

Hanlin Gu, Gongxi Zhu, Jie Zhang, Xinyuan Zhao, Yuxing Han, Lixin Fan, and Qiang Yang. Unlearning during learning: An efficient federated machine unlearning method. *arXiv preprint arXiv:2405.15474*, 2024.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain, 2019. URL https://arxiv.org/abs/1708.06733.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

Anisa Halimi, Swanand Kadhe, Ambrish Rawat, and Nathalie Baracaldo. Federated unlearning: How to efficiently erase a client in fl?, 2023. URL https://arxiv.org/abs/2207.05521.

Jamie Hayes, Ilia Shumailov, Eleni Triantafillou, Amr Khalifa, and Nicolas Papernot. Inexact unlearning needs more careful evaluations to avoid a false sense of privacy, 2024a. URL https://arxiv.org/abs/2403.01218.

Jamie Hayes, Ilia Shumailov, Eleni Triantafillou, Amr Khalifa, and Nicolas Papernot. Inexact unlearning needs more careful evaluations to avoid a false sense of privacy, 2024b. URL https://arxiv.org/abs/2403.01218.

Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54(11s): 1–37, 2022.

Thanh Trung Huynh, Trong Bang Nguyen, Phi Le Nguyen, Thanh Tam Nguyen, Matthias Weidlich, Quoc Viet Hung Nguyen, and Karl Aberer. Fast-fedul: A training-free federated unlearning with provable skew resilience, 2024. URL https://arxiv.org/abs/2405.18040.

Dongjae Jeon, Wonje Jeung, Taeheon Kim, Albert No, and Jonghyun Choi. An information theoretic metric for evaluating unlearning models. *arXiv preprint arXiv:2405.17878*, 2024.

Yu Jiang, Jiyuan Shen, Ziyao Liu, Chee Wei Tan, and Kwok-Yan Lam. Towards efficient and certified recovery from poisoning attacks in federated learning, 2024. URL https://arxiv.org/abs/2401.08216.

Ruinan Jin, Minghui Chen, Qiong Zhang, and Xiaoxiao Li. Forgettable federated linear learning with certified data unlearning, 2024. URL https://arxiv.org/abs/2306.02216.

Yasser H. Khalil, Leo Brunswic, Soufiane Lamghari, Xu Li, Mahdi Beitollahi, and Xi Chen. Not: Federated unlearning via weight negation, 2025. URL https://arxiv.org/abs/2503.05657.

Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning, 2023. URL https://arxiv.org/abs/2302.09880.

Guanghao Li, Li Shen, Yan Sun, Yue Hu, Han Hu, and Dacheng Tao. Subspace based federated unlearning. *arXiv preprint arXiv:2302.12448*, 2023a.

Na Li, Chunyi Zhou, Yansong Gao, Hui Chen, Anmin Fu, Zhi Zhang, and Yu Shui. Machine unlearning: Taxonomy, metrics, applications, challenges, and prospects. *arXiv preprint arXiv:2403.08254*, 2024.

Yuyuan Li, Chaochao Chen, Xiaolin Zheng, and Jiaming Zhang. Federated unlearning via active forgetting. *arXiv preprint arXiv:2307.03363*, 2023b.

Gaoyang Liu, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu. Federaser: Enabling efficient client-level data removal from federated learning models. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pp. 1–10. IEEE, 2021a.

Yang Liu, Zhuo Ma, Yilong Yang, Ximeng Liu, Jianfeng Ma, and Kui Ren. Revfrf: Enabling cross-domain random forest training with revocable federated learning. *IEEE Transactions on Dependable and Secure Computing*, 19(6):3671–3685, 2021b.

Yi Liu, Lei Xu, Xingliang Yuan, Cong Wang, and Bo Li. The right to be forgotten in federated learning: An efficient realization with rapid retraining. In *IEEE INFOCOM 2022-IEEE conference on computer communications*, pp. 1749–1758. IEEE, 2022.

Zhuo Ma, Yang Liu, Ximeng Liu, Jian Liu, Jianfeng Ma, and Kui Ren. Learn to forget: Machine unlearning via neuron masking. *IEEE Transactions on Dependable and Secure Computing*, 20(4): 3194–3207, 2022.

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*, pp. 1273–1282. PMLR, 2017.

Syed Irfan Ali Meerza, Amir Sadovnik, and Jian Liu. Confuse: Confusion-based federated unlearning with salience exploration. In *2024 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 427–432, 2024. doi: 10.1109/ISVLSI61997.2024.00083.

Quoc Phong Nguyen, Bryan Kian Hsiang Low, and Patrick Jaillet. Variational bayesian unlearning. *Advances in Neural Information Processing Systems*, 33:16025–16036, 2020.

XuanLong Nguyen, Martin J. Wainwright, and Michael I. Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, November 2010. ISSN 1557-9654. doi: 10.1109/tit.2010.2068870. URL http://dx.doi.org/10.1109/TIT.2010.2068870.

Nicola Novello and Andrea M. Tonello. $f$-divergence based classification: Beyond the use of cross-entropy, 2024. URL https://arxiv.org/abs/2401.01268.

Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization, 2016. URL https://arxiv.org/abs/1606.00709.

Yury Polyanskiy and Yihong Wu. Lecture notes on information theory. *Lecture Notes for ECE563 (UIUC) and*, 6(2012-2016):7, 2014.

Jose Miguel Lara Rangel, Stefan Schoepf, Jack Foster, David Krueger, and Usman Anwar. Learning to forget using hypernetworks. *arXiv preprint arXiv:2412.00761*, 2024.

Mark D. Reid and Robert C. Williamson. Information, divergence and risk for binary experiments, 2009. URL https://arxiv.org/abs/0901.0356.

Nicolò Romandini, Alessio Mora, Carlo Mazzocca, Rebecca Montanari, and Paolo Bellavista. Federated unlearning: A survey on methods, design guidelines, and evaluation metrics. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

Vincent Roulet, Tianlin Liu, Nino Vieillard, Michael E. Sander, and Mathieu Blondel. Loss functions and operators generated by f-divergences, 2025. URL https://arxiv.org/abs/2501.18537.

Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning, 2021. URL https://arxiv.org/abs/2103.03279.

Ningxin Su and Baochun Li. Asynchronous federated unlearning. In *IEEE INFOCOM 2023-IEEE conference on computer communications*, pp. 1–10. IEEE, 2023.

Anvith Thudi, Gabriel Deza, Varun Chandrasekaran, and Nicolas Papernot. Unrolling sgd: Understanding factors influencing machine unlearning. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pp. 303–319. IEEE, 2022.

Houzhe Wang, Xiaojie Zhu, Chi Chen, and Paulo Esteves-Veríssimo. Goldfish: An efficient federated unlearning framework, 2024a. URL https://arxiv.org/abs/2404.03180.

Yaxuan Wang, Jiaheng Wei, Chris Yuhao Liu, Jinlong Pang, Quan Liu, Ankit Parag Shah, Yujia Bao, Yang Liu, and Wei Wei. Llm unlearning via loss adjustment with only forget data. *arXiv preprint arXiv:2410.11143*, 2024b.

Chen Wu, Sencun Zhu, and Prasenjit Mitra. Federated unlearning with knowledge distillation. *arXiv preprint arXiv:2201.09441*, 2022.

Zuobin Xiong, Wei Li, Yingshu Li, and Zhipeng Cai. Exact-fun: an exact and efficient federated unlearning approach. In *2023 IEEE International Conference on Data Mining (ICDM)*, pp. 1439–1444. IEEE, 2023.

Zuobin Xiong, Wei Li, and Zhipeng Cai. Appro-fun: Approximate machine unlearning in federated setting. In *2024 33rd International Conference on Computer Communications and Networks (ICCCN)*, pp. 1–9. IEEE, 2024.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting, 2018. URL https://arxiv.org/abs/1709.01604.

Wei Yuan, Hongzhi Yin, Fangzhao Wu, Shijie Zhang, Tieke He, and Hao Wang. Federated unlearning for on-device recommendation. In *Proceedings of the sixteenth ACM international conference on web search and data mining*, pp. 393–401, 2023.

Yian Zhao, Pengfei Wang, Heng Qi, Jianguo Huang, Zongzheng Wei, and Qiang Zhang. Federated unlearning with momentum degradation. *IEEE Internet of Things Journal*, 11(5):8860–8870, 2024. doi: 10.1109/JIOT.2023.3321594.

## A Other Related Works

**Machine Unlearning**: Machine unlearning seeks to remove the influence of specific data from trained models, a task that has grown increasingly complex with the rise of deep learning. Early efforts established a mathematical framework inspired by differential privacy, achieving success in convex optimization problems(Sekhari et al., 2021). However, modern deep learning models, with their non-convex objectives, are prone to memorization, retaining specific training data within their parameters (Li et al., 2024). This persistence of sensitive or malicious data complicates unlearning, as standard removal techniques often fail to fully erase these traces. Further complicating the task, research has shown that models trained on non-overlapping datasets can converge to nearly identical weights, suggesting that a specific parameter state does not guarantee effective unlearning (Thudi et al., 2022).

Two primary approaches have emerged: exact unlearning (Cao & Yang, 2015b) and approximate unlearning (Nguyen et al., 2020). Exact unlearning involves retraining the model from scratch using only the remaining data, a computationally expensive process impractical for large-scale models (Cao & Yang, 2015b; Thudi et al., 2022). In contrast, approximate unlearning adjusts the trained model to approximate the outcome of retraining, requiring robust theoretical guarantees to ensure reliability (Guo et al., 2019). The non-convex nature of deep neural networks poses a significant hurdle, with heuristic methods yielding inconsistent results across benchmarks (Li et al., 2024).

To address these challenges, researchers have explored data-driven unlearning techniques. The OracleMatching method enhances stability during unlearning (Georgiev et al., 2024), while sparsity-regularized fine-tuning reduces computational overhead. Bayesian and variational inference provide probabilistic frameworks to estimate the impact of forgotten data (Nguyen et al., 2020). Data attribution techniques identify and remove targeted data influences but risk leaking information in federated settings. A critical issue, the missing targets problem, arises in fine-tuning-based methods. When unlearning a data point, gradient ascent is applied to the forget set, while gradient descent preserves performance on the retain set. Without knowing the ideal loss—equivalent to a model trained only on retained data—the process may overshoot or undershoot, destabilizing the model (Hayes et al., 2024a).

Recent advances leverage f-divergences, such as Jensen-Shannon divergence, to design loss functions that mitigate these issues, ensuring more stable unlearning (Bonato et al., 2025; Jeon et al., 2024; Rangel et al., 2024). These tools, previously used to validate machine learning tasks (Aminian et al., 2024; Roulet et al., 2025; Novello & Tonello, 2024), show particular promise for unlearning in large language models (Wang et al., 2024b). Together, these efforts reflect a dynamic field, navigating the delicate balance of efficiency, privacy, and model integrity in the pursuit of effective machine unlearning.

**Beyond Accumulator:** In the domain of active federated unlearning, (Jin et al., 2024) proposed Forgettable Federated Linear Learning with Certified Data Removal, enabling provable data removal in linear models by exploiting their analytical properties. While computationally efficient and privacy-preserving in theory, the method holds a dataset on server for calculating Hessian approximation. Several early works, such as (Xiong et al., 2023) and (Xiong et al., 2024), addressed unlearning in convex optimization settings. However, given that most modern deep learning systems involve non-convex objectives, the conclusions drawn from these convex approaches are of limited utility in the most ML frameworks.

To address unlearning in more complex models, (Wang et al., 2024a) introduced Goldfish, a framework that removes client influence without full retraining. It incorporates a novel loss balancing retained accuracy, removal bias, and confidence. However, the method needs a portion of data on the server side. In (Liu et al., 2022), the model scrubbing technique is applied locally on the target client to unlearn specific portions of data. This process involves Hessian matrix computations, with improved computational efficiency achieved by approximating the diagonal of the empirical Fisher Information Matrix (FIM).

(Che et al., 2023) introduces FFMU, an efficient federated unlearning algorithm that leverages nonlinear functional analysis specifically, the Nemytskii operator to transform local models into smooth, quantized representations, enabling the server to aggregate them into a global model with certified unlearning guarantees. (Deng et al., 2024), presents a novel Federated Client Unlearning

**Algorithm 1** $f-\texttt{Fum}$: Federated Unlearning via Divergence Optimization (Gradient Version with Local Rounds)

**Require:** Initial model $\theta^0$; iterations $T$, $T_{\text{post}}$; loss weights $\alpha, \gamma$; learning rates $\eta_{\max}, \eta_{\min}$; local epochs $E_{\max}, E_{\min}, E_{\text{post}}$; client-local forget and retain sets $\mathcal{S}_F^{(c)}, \mathcal{S}_R^{(c)}$

**function** LocalUpdate($\theta$, $\theta_{\text{ref}}$, $\eta$, $\mathcal{L}$, $E$)
  **for** $e = 1$ **to** $E$ **do**
    $g \leftarrow \nabla_\theta \mathcal{L}(\theta; \theta_{\text{ref}})$
    $\theta \leftarrow \theta - \eta g$
  **end for**
  **return** $\theta$
**end function**
$\theta_{\text{ref}} \leftarrow \theta^0$ {fixed (evaluation mode) throughout unlearning and post-training }
$\mathcal{C}_F \leftarrow \{c \in [C] : n_F^{(c)} > 0\}, \quad \mathcal{C}_R \leftarrow \{c \in [C] : n_R^{(c)} > 0\}$
$n_F \leftarrow \sum_{c \in \mathcal{C}_F} n_F^{(c)}, \quad n_R \leftarrow \sum_{c \in \mathcal{C}_R} n_R^{(c)}$
**for** $t = 0$ **to** $T - 1$ **do**
  **for all** clients $c \in \mathcal{C}_F$ in parallel **do**
    $\theta_{\max}^{(c)} \leftarrow \textsc{LocalUpdate}(\theta^t, \theta_{\text{ref}}, -\eta_{\max}, \mathcal{L}_{\max}^{(c)}, E_{\max})$
  **end for**
  $\theta^{t+\frac{1}{2}} \leftarrow \sum_{c \in \mathcal{C}_F} \frac{n_F^{(c)}}{n_F} \theta_{\max}^{(c)}$
  **for all** clients $c \in \mathcal{C}_R$ in parallel **do**
    $\theta_{\min}^{(c)} \leftarrow \textsc{LocalUpdate}(\theta^{t+\frac{1}{2}}, \theta_{\text{ref}}, \eta_{\min}, \mathcal{L}_{\min}^{(c)}(\alpha, \gamma), E_{\min})$
  **end for**
  $\theta^{t+1} \leftarrow \sum_{c \in \mathcal{C}_R} \frac{n_R^{(c)}}{n_R} \theta_{\min}^{(c)}$
**end for**
**for** $\tau = 0$ **to** $T_{\text{post}} - 1$ **do**
  **for all** clients $c \in \mathcal{C}_R$ in parallel **do**
    $\theta_{\text{post}}^{(c)} \leftarrow \textsc{LocalUpdate}(\theta^{T+\tau}, \theta_{\text{ref}}, \eta_{\min}, \mathcal{L}_{\min}^{(c)}(\alpha, \gamma), E_{\text{post}})$
  **end for**
  $\theta^{T+\tau+1} \leftarrow \sum_{c \in \mathcal{C}_R} \frac{n_R^{(c)}}{n_R} \theta_{\text{post}}^{(c)}$
**end for**
$\theta_U \leftarrow \theta^{T+T_{\text{post}}}$
**return** $\theta_U$

framework to address the right to be forgotten in medical imaging, utilizing Model Contrastive Unlearning (MCU) to align the unlearned model with a downgraded model at the feature level and Frequency Guided Memory Preservation (FGMP) to preserve general knowledge. (Wu et al., 2022) proposes a novel federated unlearning method that removes a client's contribution from a trained global model by subtracting its historical updates and recovering performance through knowledge distillation on the server side. (Zhao et al., 2024) approach splits unlearning into (i) knowledge erasure via momentum degradation and (ii) memory guidance via guided fine-tuning to restore discriminability on retained data.

## B ALGORITHM

The $f-\texttt{Fum}$ algorithm is presented in Algorithm 1.

The $f-\texttt{Fum}$ algorithm introduces several hyperparameters to control the dynamics of federated unlearning. Specifically, it uses $T$ max–min unlearning iterations and $T_{\text{post}}$ post-training iterations to refine the final model. Learning rates $\eta_{\max}$ and $\eta_{\min}$ are used during the maximization (forgetting) and minimization (retention and post-training) phases, respectively. The number of local training epochs for each client is controlled by $E_{\max}$, $E_{\min}$, and $E_{\text{post}}$, corresponding to the forgetting, retention, and post-training phases.

**Federated Updates (Gradient Version):** At iteration $t$, the server broadcasts the current global student model $\theta^t$. Clients first perform the forgetting update on their local forget sets:

$$g_{\max}^{(c)} = \nabla_\theta \mathcal{L}_{\max}^{(c)} \left( S_F^{(c)}, \theta^t, \theta_{\text{ref}} \right), \tag{8}$$

$$\theta_{\max}^{(c)} = \theta^t + \eta_{\max} g_{\max}^{(c)}, \tag{9}$$

$$\theta^{t+\frac{1}{2}} = \sum_{c \in \mathcal{C}_F} \frac{n_F^{(c)}}{n_F} \theta_{\max}^{(c)}. \tag{10}$$

The aggregated model $\theta^{t+\frac{1}{2}}$ is then broadcast to the clients participating in the retention phase:

$$g_{\min}^{(c)} = \nabla_\theta \mathcal{L}_{\min}^{(c)} \left( S_R^{(c)}, \theta^{t+\frac{1}{2}}, \theta_{\text{ref}} \right), \tag{11}$$

$$\theta_{\min}^{(c)} = \theta^{t+\frac{1}{2}} - \eta_{\min} g_{\min}^{(c)}, \tag{12}$$

$$\theta^{t+1} = \sum_{c \in \mathcal{C}_R} \frac{n_R^{(c)}}{n_R} \theta_{\min}^{(c)}. \tag{13}$$

Following Algorithm 1, the reference (teacher) model is fixed to the original pretrained model throughout both the unlearning and post-training phases:

$$\theta_{\text{ref}} = \theta^0. \tag{14}$$

The student model evolves across the unlearning iterations, whereas $\theta_{\text{ref}}$ remains unchanged.

**Post-training Minimization (Gradient Version):** After $T$ max–min unlearning iterations, the model undergoes $T_{\text{post}}$ minimization-only iterations:

$$\begin{aligned} g_{\min}^{(c)} &= \nabla_\theta \mathcal{L}_{\min}^{(c)} \left( S_R^{(c)}, \theta^{T+\tau}, \theta_{\text{ref}} \right), \\ \theta_{\text{post}}^{(c)} &= \theta^{T+\tau} - \eta_{\min} g_{\min}^{(c)}, \\ \theta^{T+\tau+1} &= \sum_{c \in \mathcal{C}_R} \frac{n_R^{(c)}}{n_R} \theta_{\text{post}}^{(c)}, \qquad \tau = 0, \ldots, T_{\text{post}} - 1. \end{aligned} \tag{15}$$

The final unlearned model is therefore

$$\theta_U = \theta^{T+T_{\text{post}}}. \tag{16}$$

For client-level unlearning, the target client $k$ participates only in the forgetting phase and is excluded from the retained-data phase. Accordingly,

$$n_R^{(k)} = 0. \tag{17}$$

Conversely, any client that does not participate in the forgetting phase has

$$n_F^{(c)} = 0. \tag{18}$$

Note that when the student and reference models have identical parameters, the first maximization gradient may be zero if their predictive distributions are identical. In our implementation, the reference model is fixed and evaluated in evaluation mode, while the student is evaluated in training mode during local updates. Consequently, training-mode-dependent components can produce different predictive distributions even when the two models initially share identical parameters, and the initial forget gradient therefore need not be zero.

## C Motivations for JS-divergence and $\chi^2$-divergence

**$f$-divergence:** The $f$-divergences are information measures that generalize various divergences, such as Kullback-Leibler (KL) divergence, through the use of a convex generator function $f$. Given two discrete distributions $P = \{p_i\}_{i=1}^k$ and $Q = \{q_i\}_{i=1}^k$, the $f$-divergence between them is defined as:

$$D_f(P \parallel Q) := \sum_{i=1}^{k} q_i f\left( \frac{p_i}{q_i} \right) \tag{19}$$

where $f : (0, \infty) \to \mathbb{R}$ is a convex function with the property that $f(1) = 0$. In this work, we focus on KL-divergence, JS-divergence and $\chi^2$-divergence (Table 6. Therefore, we consider $D_f \in \{D_{\mathrm{KL}}, D_{\mathrm{JS}}, D_{\chi^2}\}$. This definition implies that $D_f(P \parallel Q) = 0$ if and only if $P = Q$. By choosing different forms for $f$, we obtain different types of divergences. For example, when $f(t) = t\log(t)$, we get the Kullback-Leibler (KL) divergence, which measures the difference between two probability distributions.

Table 6: Divergences and their corresponding generator functions

| **Divergence** | **Generator Function** |
|---|---|
| KL-divergence | $t \log t$ |
| $\chi^2$-divergence | $(1-t)^2$ |
| JS-divergence | $\frac{1}{2}(t \log\left(\frac{2t}{1+t}\right) + \log\left(\frac{2}{1+t}\right))$ |

In this section, we study some motivations behind choosing JS-divergence and $\chi^2$ divergence. These information measures offer several advantages over KL divergence, particularly in applications involving generative modeling and robust regularization.

JS-divergence is widely used as a loss function in Generative Adversarial Networks (GANs) due to its symmetric and bounded nature, which provides a stable measure of similarity between distributions (Goodfellow et al., 2014). Unlike KL divergence, which can diverge to infinity when the two distributions have disjoint supports, JS-divergence remains finite and well-behaved, making it particularly effective for comparing empirical distributions ((Nowozin et al., 2016)). This property is especially beneficial in our context, as it helps mitigate overshoot and undershoot problems, particularly in scenarios where exact loss values for removed data points are unavailable.

On the other hand, $\chi^2$ divergence emphasizes large discrepancies due to its squared difference term, making it particularly useful in outlier detection and robust learning frameworks (Reid & Williamson, 2009). Regularizing with $\chi^2$ divergence can also help prevent models from becoming overly biased toward majority classes by strongly penalizing large probability gaps (Duchi & Namkoong, 2020). This property makes it particularly effective in imbalanced learning scenarios, where standard loss functions may fail to capture significant disparities between class distributions.

Thus, by leveraging JS-divergence for stable probability comparisons and $\chi^2$ divergence for strong regularization and outlier sensitivity, we can achieve a more robust and balanced learning framework compared to using KL divergence alone.

Building on this, we modify our loss functions and introduce $f$-`Fum`, where we select different $f$-divergences for the retain set and the forget set. Each divergence term, $d(x_r; \theta^u)$ and $d(x_f; \theta^u)$, can be chosen from JS, KL, or $\chi^2$ divergences (Nguyen et al., 2010).

# D Implementation Details

## D.1 Federated Data Partitioning

We evaluate *f-FUM* on CIFAR-10, CIFAR-100, MNIST, and FashionMNIST under federations containing either $N = 5$ or $N = 10$ clients. The training data are partitioned across clients using a class-balanced IID partition: samples from every class are divided as evenly as possible among the clients. Consequently, client dataset sizes are approximately equal, up to rounding when a class size is not divisible by $N$. All available clients participate in every federated round.

Within each client partition, the data are split in a class-stratified manner into a local optimization set and a held-out validation set. The validation data are clean, are not used for local optimization, and are disjoint from the forget, retained-training, and official test sets. The same saved partition and forget-set indices are used by all compared methods for a given experimental setting.

Table 7: Dataset-specific model architectures and federated pretraining settings. The reported pretraining rounds correspond to budgets for which all observed pretraining runs had reached convergence. Unless otherwise specified, each client performs one local epoch per communication round; for CIFAR-100, each client performs two local epochs per round.

| Dataset | Model | Local epochs | Pretraining rounds |
|---|---|---|---|
| CIFAR-10 | Reduced ResNet-18 | 1 | 200 |
| CIFAR-100 | ResNet-18 | 2 | 300 |
| MNIST | LeNet-5 | 1 | 50 |
| FashionMNIST | Lightweight CNN | 1 | 50 |

### D.2 Models and Pretraining Protocol

We use ResNet-18 for CIFAR-100, the reduced ResNet-18 architecture for CIFAR-10, LeNet-5 for MNIST, and the lightweight two-convolutional-layer network for FashionMNIST. The federated system is implemented using Flower (Beutel et al., 2022). Flower executes the participating clients in parallel: all ten clients are run in parallel for $N = 10$, and all five clients are run in parallel for $N = 5$.

Client-side optimization is performed using Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use a batch size of $512$ throughout the reported experiments. Pretraining continues until the training loss has converged. In particular, CIFAR-10 is pretrained for $200$ global communication rounds, while MNIST and FashionMNIST are pretrained for $50$ rounds. CIFAR-100 is Pretrain on 300 rounds but with two local epoch rather than one.

For each experimental configuration, every unlearning method starts from exactly the same pretrained global checkpoint. All methods also receive the same deletion request and exactly the same forget-set indices. Thus, differences between methods cannot be attributed to different pretrained models or different sampled deletion requests.

### D.3 Construction of Unlearning Requests

We consider privacy-oriented deletion, label-confusion removal, and backdoor removal.

**Privacy.** The forget set consists of clean training examples selected for deletion. No corruption or label noise is added in this setting. This avoids confounding unlearning with the privacy benefits that may arise from training on noisy data. In data-level unlearning, the requested fraction is removed from every affected client's local data. In client-level unlearning, the contribution of the target client is removed.

**Label confusion.** In the *confuse* scenario, labels are deterministically swapped within paired classes, for example $1 \leftrightarrow 2$ and $3 \leftrightarrow 4$. The same class-pair mapping is used throughout a given experiment. During evaluation, the model is assessed on both the original clean labels and the confused-label view of the forget set.

**Backdoor.** Following Halimi et al. (2023), a fixed fraction of the target client's images is modified by inserting a $3 \times 3$ pixel-pattern trigger. The triggered examples are relabeled to target class $0$. Backdoor unlearning is evaluated using both clean test accuracy and attack success rate (ASR), where lower ASR is better.

For every method and repetition, the underlying pretrained checkpoint and deletion request remain fixed. Only stochastic operations performed during unlearning, such as minibatch ordering and applicable client sampling, vary between repetitions.

### D.4 $f$-FUM Optimization

The maximization phase increases the predictive divergence between the student and reference models on the forget set. The minimization phase anchors the student to the reference model on retained

data while also minimizing the supervised retained-data loss. The corresponding local retained-data objective is weighted by $\alpha$ and $\gamma$: $\alpha$ controls teacher–student alignment on retained samples, whereas $\gamma$ controls the supervised retained-data loss.

We select

$$\alpha = 0.5, \qquad \gamma = 1.$$

These values were selected using centralized, single-client cross-validation. The selected values were then frozen and transferred to the $N = 5$ and $N = 10$ federated experiments. This strategy avoids requiring the complete federation to participate repeatedly during hyperparameter selection.

The KL, Jensen–Shannon, and $\chi^2$ variants use the same data partitions, initialization, communication budget, optimizer protocol, and checkpoint-selection rule. The divergence is the experimental factor being changed; it is not accompanied by a different deletion request or checkpoint-selection criterion.

For CIFAR-10, we perform $T = 5$ max-min iterations, followed by $T_{\text{post}} = 10$ minimization-only recovery phases. Because each max-min iteration contains one maximization synchronization and one minimization synchronization, the total cost is

$$B = 2T + T_{\text{post}} = 2(5) + 10 = 20$$

synchronization phases.

For MNIST and FashionMNIST, we use $T = 3$ and $T_{\text{post}} = 4$, giving

$$B = 2(3) + 4 = 10$$

synchronization phases.

For CIFAR-100, we use $T = 2$ max-min iterations. These iterations are followed by $T_{\text{post}} = 26$ retained-data recovery phases. The total communication budget is therefore

$$B = 2T + T_{\text{post}} = 2(2) + 26 = 30$$

synchronization phases.

The original pretrained model is used as the teacher/reference model during unlearning. This fixes the target distribution and ensures that all divergence variants are compared against the same reference.

### D.5 Baseline Fairness and Communication Budget

We compare against NoT, Halimi, and MoDE. For each fixed dataset and federation size, all approximate-unlearning methods receive the same predefined synchronization-phase budget $B$. A synchronization phase consists of

$$\text{server broadcast} \rightarrow \text{local client computation} \rightarrow \text{client upload} \rightarrow \text{server aggregation}.$$

Thus, the comparison controls the number of opportunities each method has to communicate with the federation.

For $f$-FUM, NoT, Halimi, and MoDE, respectively, the phase counts can be written as

$$\begin{aligned} C_{f\text{-FUM}} &= 2T + T_{\text{post}}, \\ C_{\text{NoT}} &= T_{\text{neg}} + T_{\text{fine}}, \\ C_{\text{Halimi}} &= T_{\text{ascent}} + T_{\text{fine}}, \\ C_{\text{MoDE}} &= T_{\text{deg}} + T_{\text{guid}}. \end{aligned}$$

Here, $T_{\text{neg}}$ and $T_{\text{fine}}$ refer to the weight-negation and fine-tuning phases of NoT (Khalil et al., 2025), $T_{\text{ascent}}$ and $T_{\text{fine}}$ to the gradient-ascent and fine-tuning phases of Halimi (Halimi et al., 2023), and $T_{\text{deg}}$ and $T_{\text{guid}}$ to the degradation and guidance phases of MoDE (Zhao et al., 2024). The experimental protocol enforces

$$C_{f\text{-FUM}} = C_{\text{NoT}} = C_{\text{Halimi}} = C_{\text{MoDE}} = B.$$

Exact retraining is different: it repeats the complete federated pretraining schedule on the retained dataset. Consequently, approximate unlearning requires substantially fewer synchronization phases than exact retraining.

Equal synchronization-phase counts do not necessarily imply equal transmitted bytes, local computation, or wall-clock time. Because these quantities were not measured consistently for every baseline, we do not claim a head-to-head byte-level or wall-clock advantage over NoT, Halimi, or MoDE. The communication reduction above is specifically relative to exact retraining.

Table 8: Comparison of data-level unlearning performance under backdoor and confused-data scenarios for $N = 10$ clients. The setting $N/r$ denotes $N$ clients with $r\%$ of poisoned data removed from each client. Higher accuracy is better.

| Method | Setting | Backdoor | | | | Confused | |
|---|---|---|---|---|---|---|---|
| | | CIFAR-10 | | Fashion-MNIST | | CIFAR-10 | |
| | | Pre. | Unl. | Pre. | Unl. | Pre. | Unl. |
| Ours (KL) | 10/6 | 28.42 | 52.62 | 65.93 | 77.10 | 60.44 | 67.84 |
| Ours (JS) | | | **73.74** | | **79.90** | | **70.72** |
| Ours ($\chi^2$) | | | 44.78 | | 76.25 | | 61.30 |
| NoT | | | 43.00 | | 26.67 | | 68.00 |
| Halimi | | | 43.00 | | 77.82 | | 68.00 |
| Ours (KL) | 10/10 | 38.42 | 52.74 | 41.98 | 49.20 | 57.10 | 64.06 |
| Ours (JS) | | | **72.32** | | 70.48 | | **67.26** |
| Ours ($\chi^2$) | | | 52.34 | | 48.17 | | 61.40 |
| NoT | | | 42.00 | | 32.13 | | 65.00 |
| Halimi | | | 42.00 | | **79.58** | | 65.00 |

Table 9: Data-level unlearning performance under the confused-data scenario for $N = 5$ clients. The setting $N/r$ denotes $N$ clients with $r\%$ of poisoned data removed from each client. Higher accuracy is better.

| Method | Setting | CIFAR-10 | |
|---|---|---|---|
| | | Pre. | Unl. |
| Ours (KL) | 5/6 | 56.86 | **74.20** |
| Ours (JS) | | | 69.80 |
| Ours ($\chi^2$) | | | 62.22 |
| NoT | | | 63.00 |
| Halimi | | | 73.00 |
| Ours (KL) | 5/10 | 54.38 | 63.24 |
| Ours (JS) | | | **69.70** |
| Ours ($\chi^2$) | | | 60.10 |
| NoT | | | 64.00 |
| Halimi | | | 64.00 |

### D.6 Model Selection, Repetitions, and Metrics

Every method is evaluated within its fixed communication budget. Among the checkpoints produced within that budget, we select the checkpoint with the highest accuracy on the held-out validation set. All reported metrics including clean accuracy, forget-set accuracy, confusion accuracy, ASR, the standard MIA score, and Yeom's attack are evaluated using that same selected checkpoint. Neither test-set performance nor MIA performance is used for checkpoint selection.

The revised principal experiments use three independent unlearning seeds, $\{7, 17, 101\}$. We report the mean and standard deviation over these three repetitions. The pretrained global checkpoint, client partition, and forget-set indices remain fixed. The seeds vary stochastic operations during unlearning and MIA evaluation. Therefore, the reported variability measures uncertainty conditional on a fixed pretrained model and fixed deletion request.

### D.7 Runtime

All experiments were run on a machine equipped with one NVIDIA H100 GPU, ten CPU cores, and 512 GB of RAM. With Flower executing clients in parallel, CIFAR-10 pretraining took approximately 2.30 hours and unlearning approximately 20 minutes. For MNIST and FashionMNIST, pretraining took approximately 25 minutes and unlearning approximately 2.5 minutes.

**Why $f$-divergences and why Jensen–Shannon for maximization.** We formulate the maximization step using an $f$-divergence, and in particular the Jensen–Shannon (JS) divergence, for stability reasons. Unlike KL divergence, Jensen–Shannon divergence is bounded above by $\log 2$. Thus, the forget-set JS objective cannot increase without bound. This motivates its use as a potentially more conservative forgetting objective, and is consistent with the favorable forgetting-utility trade-offs observed in some of our experiments. Boundedness of the divergence alone does not guarantee bounded parameter updates or prevent model drift; the optimization behavior also depends on model gradients, learning rates, and the number of updates. Empirically, this stabilizes the alternating optimization and reduces the risk of irreversible drift that would otherwise degrade recovery.

Table 10: Data-level unlearning results under the privacy scenario for CIFAR-10, Fashion-MNIST, and MNIST. The setting $N/r$ denotes $N$ clients with $r\%$ of local clean data removed from each client. Accuracy is higher-is-better, while MIA is best when closest to 50.

| Setting | Method | CIFAR-10 | | | Fashion-MNIST | | | MNIST | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Pre. Acc. | Unl. Acc. | MIA | Pre. Acc. | Unl. Acc. | MIA | Pre. Acc. | Unl. Acc. | MIA |
| 10/6 | Ours (KL) | 79.00 | 78.10 | 61.44 | 99.13 | 77.65 | 49.03 | 99.04 | 78.68 | 61.04 |
| | Ours (JS) | | 77.92 | 61.33 | | 78.17 | 48.94 | | 78.66 | 59.74 |
| | Ours ($\chi^2$) | | 77.82 | 62.00 | | 77.23 | 49.02 | | 78.52 | 62.11 |
| | NoT | | 74.14 | 51.77 | | 73.38 | 49.13 | | 73.44 | 52.11 |
| | Halimi | | 78.52 | 61.66 | | 77.43 | 48.97 | | 78.44 | 62.48 |
| 10/10 | Ours (KL) | 79.00 | 78.84 | 59.88 | 99.20 | 79.40 | 48.57 | 99.13 | 78.34 | 59.78 |
| | Ours (JS) | | 78.64 | 59.33 | | 80.10 | 48.96 | | 78.80 | 58.91 |
| | Ours ($\chi^2$) | | 78.08 | 60.22 | | 78.87 | 48.33 | | 78.12 | 60.44 |
| | NoT | | 70.14 | 51.42 | | 38.52 | 48.52 | | 70.32 | 51.42 |
| | Halimi | | 78.28 | 61.06 | | 79.18 | 48.48 | | 78.86 | 60.44 |
| 5/6 | Ours (KL) | 80.00 | 80.00 | 58.96 | 99.03 | 81.43 | 50.91 | 99.01 | 80.06 | 57.89 |
| | Ours (JS) | | 78.32 | 59.29 | | 80.77 | 50.50 | | 78.40 | 59.593 |
| | Ours ($\chi^2$) | | 79.66 | 59.67 | | 80.18 | 51.31 | | 77.54 | 57.74 |
| | NoT | | 74.82 | 54.00 | | 44.68 | 51.00 | | 75.02 | 54.19 |
| | Halimi | | 80.62 | 58.55 | | 77.07 | 51.34 | | 80.20 | 58.30 |
| 5/10 | Ours (KL) | 80.00 | 77.64 | 61.24 | 99.08 | 78.73 | 49.41 | 99.08 | 78.48 | 61.53 |
| | Ours (JS) | | 78.32 | 60.86 | | 49.56 | 49.56 | | 78.26 | 60.89 |
| | Ours ($\chi^2$) | | 77.90 | 61.62 | | 78.07 | 49.32 | | 77.68 | 61.78 |
| | NoT | | 72.74 | 52.68 | | 72.03 | 49.17 | | 71.44 | 52.64 |
| | Halimi | | 77.70 | 62.26 | | 78.30 | 49.48 | | 78.04 | 60.71 |

# E Additional Results

## E.1 Data-Level Experiments

In the tables 8- 10 we see the result of our model for the harder unlearning setup; there are couple things to mention; 1) Some attacks like Confuse attack for MNIST dataset under 10 clients and 10% data removal would not let the model converge and the accuracy and the model are not being practical; 2) on the other hand; there exist some situation where the accuracy is too low but under our unlearning setup we achieve more than 90% accuracy. (see table 3); This leads to a main question of how our model is robust through the model architecture and unlearning situation.

## E.2 Network Architecture Sensitivity

On CIFAR-10 with ResNet-18, all methods face challenges due to the dataset's inherent difficulty and the model's capacity. Shallow architectures present different challenges. This architectural sensitivity likely stems from shallow networks' limited capacity to independently represent both corrupted and clean patterns, making the unlearning objective more critical for success. Our divergence-based approach appears to be better suited to guide this limited capacity toward the desired solution. (e.g. See The NoT performance in the Table 8 and Table 9)

## E.3 The Privacy-Performance dilemma

The privacy-oriented unlearning scenarios revealed in Tables 1, 4, and 5, and 10 expose a fundamental tension inherent to federated unlearning: achieving strong privacy guarantees (measured by MIA scores approaching 50%) often comes at the cost of model utility. Unlike robustness-oriented scenarios where removing corrupted data improves performance, privacy-oriented unlearning inherently involves sacrificing useful information, making this trade-off unavoidable. Our empirical analysis across Tables 4, 7, and 10 reveals that this dilemma manifests differently depending on task complexity and federation structure: on simple tasks like MNIST, all divergence variants achieve near-perfect privacy (MIA $\approx$ 50%) with minimal utility loss, while complex tasks like CIFAR-10

Table 11: Confused-data deletion configurations for the $N = 1$ case study. Each scenario specifies the affected classes and the number of samples included in the forget set.

| Scenario | Forget set | Samples removed |
|---|---|---|
| Complete | Entire class 5 | All |
| Moderate | Class 5 | 500 |
| Dual | Classes 4 and 5 | 500/class |
| Broad | Classes 1–5 | 500/class |
| Extended | Classes 1–6 | 500/class |

Table 12: CIFAR-10 test error (%) after data-level unlearning in the $N = 1$ centralized setting with ResNet-18. The Vanilla and Full-Capacity models have pre-unlearning test errors of 16% and 4%, respectively. Results are reported as mean $\pm$ standard deviation; lower is better.

| Method | Scenario | Vanilla: Test Error | | Full Capacity: Test Error | |
|---|---|---|---|---|---|
| | | Pretrain | Unlearn | Pretrain | Unlearn |
| Ours (KL) | Complete | 16.00 | $82.28 \pm 7.10$ | 4.00 | $43.11 \pm 3.99$ |
| Ours (JS) | | | $83.01 \pm 2.47$ | | $56.34 \pm 0.69$ |
| Ours (KL) | Moderate | 16.00 | $73.05 \pm 4.55$ | 4.00 | $18.64 \pm 1.47$ |
| Ours (JS) | | | $61.79 \pm 1.90$ | | $21.68 \pm 0.14$ |
| Ours (KL) | Dual | 16.00 | $79.59 \pm 0.21$ | 4.00 | $26.27 \pm 0.09$ |
| Ours (JS) | | | $79.56 \pm 0.09$ | | $34.23 \pm 0.40$ |
| Ours (KL) | Broad | 16.00 | $74.36 \pm 3.71$ | 4.00 | $59.88 \pm 1.82$ |
| Ours (JS) | | | $80.11 \pm 0.07$ | | $59.55 \pm 1.08$ |
| Ours (KL) | Extended | 16.00 | $82.81 \pm 5.48$ | 4.00 | $55.38 \pm 4.27$ |
| Ours (JS) | | | $79.04 \pm 0.47$ | | $63.76 \pm 2.82$ |

expose a clear Pareto frontier where practitioners must choose between aggressive forgetting (KL-JS achieving 55.20% MIA but 63.16% accuracy) and utility preservation (KL-$\chi^2$ maintaining 77.66% accuracy but 64.60% MIA). Notably, we observe that fewer clients ($N = 5$) generally yield better outcomes in both dimensions compared to more fragmented settings ($N = 10$), and that fine-grained data-level unlearning (2-10% per client) persistently struggles to achieve ideal privacy metrics even when utility is well-preserved, suggesting fundamental limitations in current federated unlearning approaches for distributed deletion requests.

## F Case study: Single Client Setting ($N = 1$) with Confuse setup

We further evaluate our method in the setting where only a single client participates ($N = 1$), which reduces to *centralized unlearning*. Our motivation is practical: hyperparameter tuning can be performed efficiently in the single-client (centralized) setting, and the resulting choices can then be broadcast to the full federated setup. This approach reduces computational cost and, in real-world deployments, avoids requiring participation from the entire distributed population of clients during hyperparameter tuning.

We also report additional empirical results for this setting, which provide an expanded treatment of the discussion.

We consider two versions of the model (ResNet-18). The first, which we call the vanilla model, was trained on CIFAR-10, achieving an accuracy of 0.84. We refer to this as the vanilla original model. Notably, this model does not operate at full capacity. Since we believe that the unlearning frameworks should be independent of the original model's training procedure, we also evaluate a full-capacity original model, which is a Torchvision pre-trained model with a precision of 0.96. For the unlearning process, we apply one policy. We perform two epochs of maximization, each followed by a minimization step, with an additional minimization step at the end. This is a very extreme recovery treatment with only one round of minimization.

Since $N$ is equal to one we evaluate data level deletion. We consider confuse scenario summarized in Table 11, to support our empirical results. Following setup in (Kurmanji et al., 2023) which are already mentioned.

We analyze the performance of the models across different confuse scenarios. In this section, we focus on KL-JS and KL-KL method variance. The best performance is achieved when the error in the test dataset is minimized. As shown in Table 12, using JS loss as the maximization loss generally results in **lower variance** across almost most of all scenarios.

To interpret these results, we define the best loss function as the one where test error is lowest. This is not always the case. With a more nuanced analysis, we argue that KL-JS performs better in most

scenarios with confidence. In cases where it does not, model degradation complicates the analysis and introduces significant complexity.

In the complete forgetting scenario, where the goal is to forget all data from a specific class, the error on the forget set rapidly reaches 100%. After two max-min epochs, minimizing with the largest possible batch size yields strong results, as demonstrated in (Kurmanji et al., 2023). The key point here is that because the entire class is absent, post-minimization does not affect the forget error.

As shown in Table 12, in the Moderate case, KL-JS clearly outperforms the baseline in the Vanilla setup. In the full capacity case for Broad, the KL−JS outperforming baseline is determined.

Our observations indicate that in the Vanilla case for Moderate (forgetting 500 samples) and up to Extended cases, the degradation in model performance is so severe that one could argue the model has effectively lost its knowledge. This highlights a critical limitation based on current literature when the number of deleted samples per class increases. This issue presents emerging challenges in the field, which are highly relevant to real world scenarios. A similar problem is also addressed in (Sekhari et al., 2021). Broad and Extended cases in full capacity also suffer the same problem mentioned here. As you can see, there is not a single case where KL−KL outperforms KL−JS with full confidence (lower variance) and not degraded model.

# G Communication-Cost Analysis

Federated unlearning should be evaluated not only in terms of predictive utility and forgetting effectiveness, but also in terms of the communication required to obtain the unlearned model. This distinction is particularly important because, in federated learning, communication between the server and clients can dominate the cost of local computation.

We therefore analyze communication using two complementary measures. The first measure is implementation independent and is the primary measure used in our experiments. The second provides a more fine-grained analytical view of the communication generated by $f\text{-}\texttt{Fum}$.

## G.1 Synchronization-Phase Accounting

A synchronization phase consists of

$$\text{server broadcast} \;\rightarrow\; \text{local client computation} \;\rightarrow\; \text{client upload} \;\rightarrow\; \text{server aggregation}.$$

For $f\text{-}\texttt{Fum}$, a single max–min iteration contains two distinct synchronization phases. The first phase performs forget-set maximization, while the second performs retained-data minimization. Hence, if $T$ denotes the number of max–min iterations and $T_{\text{post}}$ denotes the number of post-training minimization iterations, the total number of synchronization phases is

$$B_{f\text{-}\texttt{Fum}} = 2T + T_{\text{post}}. \tag{20}$$

This accounting is consistent with Algorithm 1: the server aggregates once after the maximization update and once again after the retained-data minimization update.

For comparison, we denote the corresponding phase decomposition of the baselines as

$$B_{\text{Halimi}} = T_{\text{ascent}} + T_{\text{fine}}, \tag{21}$$

$$B_{\text{NoT}} = T_{\text{neg}} + T_{\text{fine}}, \tag{22}$$

$$B_{\text{MoDE}} = T_{\text{deg}} + T_{\text{guid}}, \tag{23}$$

where $T_{\text{neg}}$ and $T_{\text{fine}}$ refer to the weight-negation and fine-tuning phases of NoT (Khalil et al., 2025), $T_{\text{ascent}}$ and $T_{\text{fine}}$ to the gradient-ascent and fine-tuning phases of Halimi (Halimi et al., 2023), and $T_{\text{deg}}$ and $T_{\text{guid}}$ to the degradation and guidance phases of MoDE (Zhao et al., 2024).

To avoid confounding unlearning quality with additional communication, all approximate-unlearning methods are evaluated under the same predefined synchronization-phase budget,

$$B_{f\text{-}\texttt{Fum}} = B_{\text{Halimi}} = B_{\text{NoT}} = B_{\text{MoDE}} = B. \tag{24}$$

**Interpretation of the equal-budget comparison.** We assign $f-\texttt{Fum}$ and each approximate-unlearning baseline the same predefined budget $B$ of server–client synchronization phases. A phase comprises a server broadcast, local client computation, client upload, and server aggregation. In $f-\texttt{Fum}$, maximization and minimization are separate phases, so its total is

$$2T + T_{\text{post}} = B.$$

This protocol controls the number of synchronization opportunities available to each method. Consequently, the head-to-head experiments compare forgetting effectiveness and retained-model utility at an equal phase budget; they do not claim that $f-\texttt{Fum}$ uses fewer phases than the other approximate-unlearning methods. Equal phase counts do not necessarily imply equal transmitted bytes, local computation, or wall-clock time. Since these quantities were not measured consistently for every baseline, we make no head-to-head efficiency claim on those measures.

Thus, comparisons between approximate-unlearning methods are performed under equal communication opportunity. The communication advantage relative to exact retraining arises because exact retraining must repeat the complete federated training schedule on the retained data. Importantly, we do not claim a measured transmitted-byte advantage over NoT, Halimi, or MoDE. Total transmitted bytes were not explicitly logged for these methods and may depend on implementation details including broadcast mechanisms, compression, secure aggregation, and client availability. Instead, the head-to-head experimental comparison controls communication through the common synchronization budget in Eq. equation 24.

The finer-grained analysis in Eqs. equation 20–equation 23 shows, however, that synchronization-phase count can be conservative for $f-\texttt{Fum}$ in client-level unlearning: a forgetting phase may require an upload from only the departing client, whereas an ordinary retained-data training phase requires updates from all retained clients.

# H Theoretical Results

In this section, we provide three main theoretical results for our approach. The first theoretical question is whether $f-\texttt{Fum}$ is merely a heuristic alternation of ascent and descent steps, or whether it optimizes a well-defined objective. This point is important because the algorithm maximizes a forget-set divergence and then minimizes a retained-data objective, which could otherwise look like two unrelated procedures. The second theoretical question is why $f-\texttt{Fum}$ should alternate forget-set maximization with retained-data minimization. This is central point because max-only unlearning can move the decision boundary away from the forget set while also destroying the retained-data structure. The third theoretical question is whether the model produced after $f-\texttt{Fum}$ is close to a retained-data optimization target. Exact unlearning is defined by equivalence to training on $S_R$ (Cao & Yang, 2015b); in the nonconvex analysis below, we therefore use a local retained-data ERM minimizer as a tractable reference target rather than claiming parameter equality with an arbitrary fresh retraining run. In the following, we first provide notations and preliminaries for our main theoretical results. Then, we provide main theoretical results.

## H.1 Notations

We repeat the notations in this section. Let $C$ be the number of clients. Client $c$ has local data

$$S^{(c)} = S_R^{(c)} \cup S_F^{(c)}, \qquad S_R^{(c)} \cap S_F^{(c)} = \emptyset, \tag{25}$$

where $S_F^{(c)}$ is the local forget set and $S_R^{(c)}$ is the local retained set [2]. Define

$$n_F^{(c)} := |S_F^{(c)}|, \qquad n_R^{(c)} := |S_R^{(c)}|, \tag{26}$$

$$n_F := \sum_{c=1}^{C} n_F^{(c)}, \qquad n_R := \sum_{c=1}^{C} n_R^{(c)}. \tag{27}$$

[2]Throughout, expressions normalized by $n_R$ or $n_F$ are understood only when the corresponding set is nonempty. If $S_R = \emptyset$ or $S_F = \emptyset$, the associated loss/divergence term is omitted from the objective

Let

$$q_\theta(x) := h(x;\theta) \in \Delta^{K-1} \tag{28}$$

denote the predictive probability vector of the model. Let $f_F$ be the divergence generator used on the forget set and $f_R$ the generator used on the retained set. For a teacher or reference parameter $\theta_T$, define

$$d_F(x;\theta,\theta_T) := D_{f_F}\big(q_\theta(x)\|q_{\theta_T}(x)\big), \tag{29}$$

$$d_R(x;\theta,\theta_T) := D_{f_R}\big(q_\theta(x)\|q_{\theta_T}(x)\big). \tag{30}$$

The client-level forget objective is

$$\mathcal{L}^{(c)}_{\max}(\theta;\theta_T) := \frac{1}{n_F^{(c)}} \sum_{x\in S_F^{(c)}} d_F(x;\theta,\theta_T), \tag{31}$$

and the client-level retain objective is

$$\mathcal{L}^{(c)}_{\min}(\theta;\theta_T) := \frac{\alpha}{n_R^{(c)}} \sum_{x\in S_R^{(c)}} d_R(x;\theta,\theta_T) + \frac{\gamma}{n_R^{(c)}} \sum_{(x,y)\in S_R^{(c)}} \ell_{f_R}(q_\theta(x),y). \tag{32}$$

The corresponding global objectives are

$$\mathcal{L}_{\max}(\theta;\theta_T) := \sum_{c=1}^{C} \frac{n_F^{(c)}}{n_F} \mathcal{L}^{(c)}_{\max}(\theta;\theta_T) = \frac{1}{n_F} \sum_{c=1}^{C} \sum_{x\in S_F^{(c)}} d_F(x;\theta,\theta_T), \tag{33}$$

$$\mathcal{L}_{\min}(\theta;\theta_T) := \sum_{c=1}^{C} \frac{n_R^{(c)}}{n_R} \mathcal{L}^{(c)}_{\min}(\theta;\theta_T) \tag{34}$$

with the explicit form

$$\mathcal{L}_{\min}(\theta;\theta_T) = \frac{\alpha}{n_R} \sum_{c=1}^{C} \sum_{x\in S_R^{(c)}} d_R(x;\theta,\theta_T) + \frac{\gamma}{n_R} \sum_{c=1}^{C} \sum_{(x,y)\in S_R^{(c)}} \ell_{f_R}(q_\theta(x),y). \tag{35}$$

Define the contrastive unlearning potential

$$\Psi_\rho(\theta;\theta_T) := \mathcal{L}_{\min}(\theta;\theta_T) - \rho\mathcal{L}_{\max}(\theta;\theta_T), \qquad \rho := \frac{\eta_{\max}}{\eta_{\min}}. \tag{36}$$

The retain term preserves utility and output behavior on $S_R$, while the negative forget term encourages the unlearned model to move away from the original model on $S_F$.

### H.2 Preliminaries

This subsection collects the definitions, assumptions, and auxiliary lemmas used by the three main results. The preliminaries are grouped according to the result they support.

#### H.2.1 Federated contrastive optimization

The contrastive unlearning potential is

$$\Psi_\rho(\theta;\theta_T) := \mathcal{L}_{\min}(\theta;\theta_T) - \rho\mathcal{L}_{\max}(\theta;\theta_T), \qquad \rho = \frac{\eta_{\max}}{\eta_{\min}}. \tag{37}$$

The term $\mathcal{L}_{\min}$ preserves retained-data behavior and supervised utility, while $-\rho\mathcal{L}_{\max}$ creates a directed removal pressure on the forget set. The federated averaging notation follows FedAvg (McMahan et al., 2017); the retain/forget split adapts standard machine-unlearning and federated-unlearning formulations (Cao & Yang, 2015b; Sekhari et al., 2021; Romandini et al., 2024).

**Assumption 1** (Smoothness)**.** *The functions $\mathcal{L}_{\max}$ and $\mathcal{L}_{\min}$ are differentiable and have Lipschitz gradients on the local parameter region visited by $f$-`Fum`. That is, there exist $L_F, L_R > 0$ such that*

$$\|\nabla\mathcal{L}_{\max}(\theta) - \nabla\mathcal{L}_{\max}(\theta')\| \le L_F\|\theta-\theta'\|, \tag{38}$$

$$\|\nabla\mathcal{L}_{\min}(\theta) - \nabla\mathcal{L}_{\min}(\theta')\| \le L_R\|\theta-\theta'\|. \tag{39}$$

*Consequently, $\Psi_\rho$ is $L_\Psi$-smooth with*

$$L_\Psi \le L_R + \rho L_F. \tag{40}$$

**Assumption 2** (Bounded gradients)**.** *There exists $B_F > 0$ such that*

$$\|\nabla \mathcal{L}_{\max}(\theta)\| \le B_F \tag{41}$$

*for all iterates considered in the analysis.*

**Lemma 1** (FedAvg equivalence for one full-batch $f$−Fum step)**.** *Assume each client performs one full-batch forget-ascent step*

$$\theta_{c,+}^F = \theta_t + \eta_{\max} \nabla \mathcal{L}_{\max}^c(\theta_t; \theta_T). \tag{42}$$

*If the server aggregates the resulting client models with weights $n_F^{(c)}/n_F$, then*

$$\theta_{t+1/2} = \sum_{c=1}^{C} \frac{n_F^{(c)}}{n_F} \theta_{c,+}^F = \theta_t + \eta_{\max} \nabla \mathcal{L}_{\max}(\theta_t; \theta_T). \tag{43}$$

*Similarly, if each client performs one full-batch retain-descent step*

$$\theta_{c,+}^R = \theta_{t+1/2} - \eta_{\min} \nabla \mathcal{L}_{\min}^c(\theta_{t+1/2}; \theta_T) \tag{44}$$

*and the server aggregates with weights $n_R^{(c)}/n_R$, then*

$$\theta_{t+1} = \sum_{c=1}^{C} \frac{n_R^{(c)}}{n_R} \theta_{c,+}^R = \theta_{t+1/2} - \eta_{\min} \nabla \mathcal{L}_{\min}(\theta_{t+1/2}; \theta_T). \tag{45}$$

This lemma is an exact identity only for one common-start, full-batch local step. With multiple local steps, client iterates generally drift apart and the aggregated update acquires an additional client-drift error.

*Proof.* By the definition of $\mathcal{L}_{\max}$,

$$\sum_{c=1}^{C} \frac{n_F^{(c)}}{n_F} \theta_{c,+}^F = \sum_{c=1}^{C} \frac{n_F^{(c)}}{n_F} \left(\theta_t + \eta_{\max} \nabla \mathcal{L}_{\max}^c(\theta_t; \theta_T)\right) \tag{46}$$

$$= \theta_t + \eta_{\max} \sum_{c=1}^{C} \frac{n_F^{(c)}}{n_F} \nabla \mathcal{L}_{\max}^c(\theta_t; \theta_T) \tag{47}$$

$$= \theta_t + \eta_{\max} \nabla \mathcal{L}_{\max}(\theta_t; \theta_T). \tag{48}$$

The retained-data step follows identically from the definition of $\mathcal{L}_{\min}$. □

### H.2.2 Local $f$-divergence geometry

Let

$$J_x := \nabla_\theta q_\theta(x)\big|_{\theta=\theta_T} \in \mathbb{R}^{K \times d} \tag{49}$$

be the Jacobian of the model output at the teacher parameter. Define the empirical Fisher-type matrices

$$I_R(\theta_T) := \frac{1}{n_R} \sum_{x \in S_R} J_x^\top \operatorname{Diag}(q_{\theta_T}(x))^{-1} J_x, \tag{50}$$

$$I_F(\theta_T) := \frac{1}{n_F} \sum_{x \in S_F} J_x^\top \operatorname{Diag}(q_{\theta_T}(x))^{-1} J_x. \tag{51}$$

Finally, let

$$H_R := \nabla^2 \mathcal{L}_R(\theta_T), \qquad \mathcal{L}_R(\theta) := \frac{1}{n_R} \sum_{(x,y) \in S_R} \ell(q_\theta(x), y), \tag{52}$$

and let

$$c_R := f_R''(1), \qquad c_F := f_F''(1). \tag{53}$$

**Assumption 3** (Interior softmax probabilities)**.** *There exists $\kappa > 0$ such that, for all relevant samples $x$, classes $k$, and parameters $\theta$ in the local region of interest,*

$$q_{\theta,k}(x) \geq \kappa. \tag{54}$$

*This local condition is automatically satisfied if probabilities are clipped away from zero.*

**Assumption 4** (Local differentiability of the predictive map)**.** *For every relevant sample $x$, the map $\theta \mapsto q_\theta(x)$ is twice continuously differentiable on a neighborhood of $\theta_T$. More generally, it is sufficient that the iterates remain in a local region on which the first- and second-order derivatives used below exist and are continuous.*

**Lemma 2** (Local Fisher expansion of $f$-divergences (Polyanskiy & Wu, 2014))**.** *Assume Assumptions 3 and 4. Let $f$ be twice differentiable at 1, with $f(1) = 0$, and normalize it so that $f'(1) = 0$. For a small perturbation $\delta \in \mathbb{R}^d$,*

$$D_f\big(q_{\theta_T+\delta}(x) \| q_{\theta_T}(x)\big) = \frac{f''(1)}{2} \delta^\top J_x^\top \operatorname{Diag}(q_{\theta_T}(x))^{-1} J_x \delta + o(\|\delta\|^2). \tag{55}$$

*For the divergences used in $f$-`Fum`,*

$$f''_{D_{\mathrm{KL}}}(1) = 1, \qquad f''_{\mathrm{JS}}(1) = \frac{1}{4}, \qquad f''_{\chi^2}(1) = 2. \tag{56}$$

*Moreover, with natural logarithms,*

$$0 \leq D_{\mathrm{JS}}(p\|q) \leq \log 2. \tag{57}$$

### H.2.3 Retraining target

We use the retained-data empirical-risk minimization (ERM) target

$$\theta_R^\star := \arg\min_\theta \mathcal{L}_R(\theta), \tag{58}$$

where $\mathcal{L}_R$ is the retained supervised loss in Equation equation 52. This target is a local optimization reference for the analysis; in a nonconvex neural network it need not coincide in parameter space with the output of rerunning the full stochastic federated training algorithm on $S_R$. The post-training objective is

$$\mathcal{P}_\alpha(\theta) := \gamma \mathcal{L}_R(\theta) + \alpha \mathcal{D}_R(\theta; \theta_T), \tag{59}$$

where

$$\mathcal{D}_R(\theta; \theta_T) := \frac{1}{n_R} \sum_{x \in S_R} D_{f_R}\big(q_\theta(x) \| q_{\theta_T}(x)\big). \tag{60}$$

Let

$$\theta_\alpha^\star := \arg\min_\theta \mathcal{P}_\alpha(\theta). \tag{61}$$

The following local optimization assumptions are standard for analyzing optimization problems.

**Assumption 5** (Local strong convexity for distance guarantees)**.** *There exists a convex local neighborhood $\mathcal{B}$ containing the relevant post-training iterates and the local minimizers $\theta_\alpha^\star$ and $\theta_R^\star$ such that $\mathcal{P}_\alpha$ is $\mu_P$-strongly convex on $\mathcal{B}$. In particular, whenever Proposition 2 is invoked, $\theta_U$, $\theta_\alpha^\star$, and $\theta_R^\star$, together with the line segments joining them, are assumed to lie in $\mathcal{B}$. In the special case $\alpha = 0$, $\mathcal{L}_R$ is locally $\mu_R$-strongly convex on the same neighborhood, so that $\gamma\mathcal{L}_R$ is $\gamma\mu_R$-strongly convex.*

**Assumption 6** (Local PL condition for optimization-rate guarantees)**.** *The post-training objective $\mathcal{P}_\alpha$ is $L_P$-smooth and satisfies the Polyak–Lojasiewicz condition in the local basin reached by $f$-`Fum`: for some $\mu_{PL} > 0$,*

$$\frac{1}{2}\|\nabla \mathcal{P}_\alpha(\theta)\|^2 \geq \mu_{PL}\big(\mathcal{P}_\alpha(\theta) - \mathcal{P}_\alpha^\star\big). \tag{62}$$

**Lemma 3** (Evaluation transfer from parameter/output closeness)**.** *The following two consequences will be used to interpret approximate retraining.*

*(a) If the supervised loss is cross-entropy,*

$$\ell(q_\theta(x), y) = -\log q_{\theta,y}(x), \tag{63}$$

*then the empirical retained-data classification error*

$$\widehat{\mathrm{err}}_R(\theta) := \frac{1}{n_R} \sum_{(x,y)\in S_R} \mathbb{1}\left[\arg\max_k q_{\theta,k}(x) \neq y\right] \tag{64}$$

*satisfies*

$$\widehat{\mathrm{err}}_R(\theta) \leq \frac{\mathcal{L}_R(\theta)}{\log 2}. \tag{65}$$

*(b) Let $a : \Delta^{K-1} \to [0,1]$ be an output-based membership-inference attack score that is $L_a$-Lipschitz in $\ell_1$ distance:*

$$|a(p) - a(q)| \leq L_a \|p - q\|_1. \tag{66}$$

*Assume also that the model output is locally $L_h$-Lipschitz in parameters:*

$$\|q_\theta(x) - q_{\theta'}(x)\|_1 \leq L_h \|\theta - \theta'\|. \tag{67}$$

*Then, for any forget set $S_F$,*

$$\left|\frac{1}{n_F}\sum_{x\in S_F} a(q_\theta(x)) - \frac{1}{n_F}\sum_{x\in S_F} a(q_{\theta'}(x))\right| \leq L_a L_h \|\theta - \theta'\|. \tag{68}$$

*Proof.* For part (a), if the classifier predicts incorrectly, then the true-class probability cannot exceed $1/2$, i.e., $q_{\theta,y}(x) \leq 1/2$. Hence $-\log q_{\theta,y}(x) \geq \log 2$, and

$$\mathbb{1}\left[\arg\max_k q_{\theta,k}(x) \neq y\right] \leq \frac{-\log q_{\theta,y}(x)}{\log 2}. \tag{69}$$

Averaging over $S_R$ proves Equation equation 65.

For part (b), by the triangle inequality, the Lipschitz property of $a$, and the Lipschitz property of $q_\theta$,

$$\left|\frac{1}{n_F}\sum_{x\in S_F} a(q_\theta(x)) - \frac{1}{n_F}\sum_{x\in S_F} a(q_{\theta'}(x))\right| \tag{70}$$

$$\leq \frac{1}{n_F}\sum_{x\in S_F} |a(q_\theta(x)) - a(q_{\theta'}(x))| \tag{71}$$

$$\leq \frac{L_a}{n_F}\sum_{x\in S_F} \|q_\theta(x) - q_{\theta'}(x)\|_1 \tag{72}$$

$$\leq L_a L_h \|\theta - \theta'\|. \tag{73}$$

□

The classification-error bound is an instance of the standard implication from surrogate-risk consistency to classification-risk consistency; a general treatment of surrogate calibration is given by (Bartlett et al., 2006). The membership-inference part is stated for output-based attacks, following the attack setting introduced by (Hu et al., 2022). The need to evaluate unlearning against the retained-data retraining model is emphasized by (Hayes et al., 2024b).

### H.3 Main Results

#### H.3.1 $f$-Fum is an inexact optimizer of a contrastive unlearning objective

The result below shows that, after the weighted aggregation justified in Lemma 1, one idealized $f$-Fum round is an inexact gradient step on the contrastive potential $\Psi_\rho = \mathcal{L}_{\min} - \rho\mathcal{L}_{\max}$. In the unlearning context, this result is needed to justify that $f$-Fum has a principled optimization target rather than only empirical motivation.

Throughout the theoretical analysis in this subsection, the teacher parameter is fixed across unlearning rounds. In particular, one may take

$$\theta_T = \theta_0. \tag{74}$$

A round-dependent teacher produces a time-varying objective and requires an additional teacher-drift term; that setting is not covered by the stationarity result below.

**Theorem 1** (Inexact contrastive optimization for the fixed-teacher one-step update)**.** *Let $\eta := \eta_{\min}$ and $\rho := \eta_{\max}/\eta_{\min}$. Consider the idealized aggregated $f\text{-}\mathtt{Fum}$ round*

$$\theta_{t+1/2} = \theta_t + \eta_{\max} \nabla \mathcal{L}_{\max}(\theta_t; \theta_T), \tag{75}$$

$$\theta_{t+1} = \theta_{t+1/2} - \eta_{\min} \nabla \mathcal{L}_{\min}(\theta_{t+1/2}; \theta_T). \tag{76}$$

*Under Assumptions 1 and 2, the update can be written as*

$$\theta_{t+1} = \theta_t - \eta \nabla \Psi_\rho(\theta_t; \theta_T) + e_t, \tag{77}$$

*where*

$$\|e_t\| \le \rho \eta^2 L_R B_F. \tag{78}$$

*Furthermore, suppose that the fixed-teacher potential $\Psi_\rho(\cdot; \theta_T)$ is lower bounded by $\Psi_{\inf}$ and that the inexact update satisfies the more general bound $\|e_t\| \le \eta \varepsilon_{\text{alg}}$. If*

$$0 < \eta \le \frac{1}{2L_\Psi}, \tag{79}$$

*then after $T$ rounds,*

$$\frac{1}{T} \sum_{t=0}^{T-1} \|\nabla_\theta \Psi_\rho(\theta_t; \theta_T)\|^2 \le \frac{4(\Psi_\rho(\theta_0; \theta_T) - \Psi_{\inf})}{\eta T} + 8\varepsilon_{\text{alg}}^2, \tag{80}$$

*and therefore*

$$\min_{0 \le t < T} \|\nabla_\theta \Psi_\rho(\theta_t; \theta_T)\|^2 \le \frac{4(\Psi_\rho(\theta_0; \theta_T) - \Psi_{\inf})}{\eta T} + 8\varepsilon_{\text{alg}}^2. \tag{81}$$

*For the one-step full-batch version in Equations equation 75–equation 76, one may take*

$$\varepsilon_{\text{alg}} = \rho \eta L_R B_F. \tag{82}$$

*Proof.* The analysis is inspired by standard smooth nonconvex optimization tools (Ghadimi & Lan, 2013; Bottou et al., 2018).

We first prove the inexact-gradient representation. From Equations equation 75–equation 76,

$$\theta_{t+1} = \theta_t + \rho\eta \nabla \mathcal{L}_{\max}(\theta_t; \theta_T) - \eta \nabla \mathcal{L}_{\min}(\theta_{t+1/2}; \theta_T). \tag{83}$$

Adding and subtracting $\eta \nabla \mathcal{L}_{\min}(\theta_t; \theta_T)$ gives

$$\theta_{t+1} = \theta_t - \eta \left( \nabla \mathcal{L}_{\min}(\theta_t; \theta_T) - \rho \nabla \mathcal{L}_{\max}(\theta_t; \theta_T) \right) + e_t, \tag{84}$$

where

$$e_t := -\eta \left( \nabla \mathcal{L}_{\min}(\theta_{t+1/2}; \theta_T) - \nabla \mathcal{L}_{\min}(\theta_t; \theta_T) \right). \tag{85}$$

Since $\nabla \Psi_\rho = \nabla \mathcal{L}_{\min} - \rho \nabla \mathcal{L}_{\max}$, Equation equation 77 follows. By $L_R$-smoothness of $\mathcal{L}_{\min}$,

$$\|e_t\| \le \eta L_R \|\theta_{t+1/2} - \theta_t\| \tag{86}$$

$$= \eta L_R \eta_{\max} \|\nabla \mathcal{L}_{\max}(\theta_t; \theta_T)\| \tag{87}$$

$$\le \rho \eta^2 L_R B_F, \tag{88}$$

which proves Equation equation 78.

We next prove the stationarity bound. Let $g_t := \nabla_\theta \Psi_\rho(\theta_t; \theta_T)$. By $L_\Psi$-smoothness of the fixed-teacher potential,

$$\Psi_\rho(\theta_{t+1}; \theta_T) \le \Psi_\rho(\theta_t; \theta_T) + \langle g_t, \theta_{t+1} - \theta_t \rangle + \frac{L_\Psi}{2} \|\theta_{t+1} - \theta_t\|^2. \tag{89}$$

Using $\theta_{t+1} - \theta_t = -\eta g_t + e_t$ and $\| - \eta g_t + e_t\|^2 \le 2\eta^2\|g_t\|^2 + 2\|e_t\|^2$, we obtain

$$\Psi_\rho(\theta_{t+1};\theta_T) \le \Psi_\rho(\theta_t;\theta_T) - \eta\|g_t\|^2 + \langle g_t, e_t\rangle + L_\Psi\eta^2\|g_t\|^2 + L_\Psi\|e_t\|^2. \tag{90}$$

Because $\eta \le 1/(2L_\Psi)$,

$$-\eta\|g_t\|^2 + L_\Psi\eta^2\|g_t\|^2 \le -\frac{\eta}{2}\|g_t\|^2. \tag{91}$$

Using $\|e_t\| \le \eta\varepsilon_{\mathrm{alg}}$ and Young's inequality,

$$\langle g_t, e_t\rangle \le \eta\varepsilon_{\mathrm{alg}}\|g_t\| \le \frac{\eta}{4}\|g_t\|^2 + \eta\varepsilon_{\mathrm{alg}}^2. \tag{92}$$

Also,

$$L_\Psi\|e_t\|^2 \le L_\Psi\eta^2\varepsilon_{\mathrm{alg}}^2 \le \frac{\eta}{2}\varepsilon_{\mathrm{alg}}^2. \tag{93}$$

Therefore,

$$\Psi_\rho(\theta_{t+1};\theta_T) \le \Psi_\rho(\theta_t;\theta_T) - \frac{\eta}{4}\|g_t\|^2 + 2\eta\varepsilon_{\mathrm{alg}}^2. \tag{94}$$

Summing from $t = 0$ to $T-1$ and using the lower bound $\Psi_\rho(\theta^{(T)};\theta_T) \ge \Psi_{\mathrm{inf}}$ for the final iterate $\theta^{(T)}$ gives

$$\frac{\eta}{4}\sum_{t=0}^{T-1}\|g_t\|^2 \le \Psi_\rho(\theta_0;\theta_T) - \Psi_{\mathrm{inf}} + 2\eta T\varepsilon_{\mathrm{alg}}^2. \tag{95}$$

Dividing by $\eta T/4$ proves Equation equation 80. Equation equation 81 follows because the minimum is no larger than the average. □

Note that the bound in Theorem 1 is for the idealized one-full-batch-step update. With mini-batches or multiple local epochs, additional stochastic-gradient and client-drift terms enter the inexactness error. Quantifying those terms requires further variance and client-heterogeneity assumptions and is outside the present theorem.

**Remark 1** (Lower boundedness for JS forget maximization)**.** *When the forget objective uses the standard Jensen–Shannon divergence,* $0 \le D_{\mathrm{JS}} \le \log 2$*, and hence* $\mathcal{L}_{\max}(\theta;\theta_T) \le \log 2$*. Since the retained-data objective* $\mathcal{L}_{\min}$ *is composed of the nonnegative divergence and supervised-loss terms used here,* $\mathcal{L}_{\min}(\theta;\theta_T) \ge 0$*,*

$$\Psi_\rho(\theta;\theta_T) = \mathcal{L}_{\min}(\theta;\theta_T) - \rho\mathcal{L}_{\max}(\theta;\theta_T) \ge -\rho\log 2. \tag{96}$$

*Thus, for JS forget maximization the lower-boundedness assumption in Theorem 1 holds automatically with* $\Psi_{\mathrm{inf}} \ge -\rho\log 2$*. For unbounded forget divergences such as KL or* $\chi^2$*, lower boundedness must instead be imposed through the local region of analysis or an appropriate bounded-output/parameter assumption.*

### H.3.2 Retained-data minimization prevents local collapse

The toy example illustrates this phenomenon visually: the max-only phase can collapse the clean decision boundary, whereas the min–max phase preserves the global class structure and leaves only a local distortion to be repaired by post-training. The following Theorem shows that forget-set maximization introduces negative curvature in forget-sensitive directions, while the retained-data objective contributes positive curvature that can dominate this destabilizing effect.

**Theorem 2** (Local curvature stabilization by retained-data minimization)**.** *Under Assumptions 3 and 4 and the conditions of Lemma 2, the local Hessian of the contrastive objective around* $\theta_T$ *is*

$$\nabla^2\Psi_\rho(\theta_T) = \gamma H_R + \alpha c_R I_R(\theta_T) - \rho c_F I_F(\theta_T). \tag{97}$$

*If*

$$\lambda_{\min}\left(\gamma H_R + \alpha c_R I_R(\theta_T)\right) > \rho c_F \lambda_{\max}\left(I_F(\theta_T)\right), \tag{98}$$

*where* $\lambda_{\min}(\cdot)$ *and* $\lambda_{\max}(\cdot)$ *are minimum eigenvalue and maximum eigenvalue, respectively. then*

$$\nabla^2\Psi_\rho(\theta_T) \succ 0. \tag{99}$$

*Equivalently, positive local curvature is guaranteed whenever*

$$\rho < \frac{\lambda_{\min}\left(\gamma H_R + \alpha c_R I_R(\theta_T)\right)}{c_F\lambda_{\max}\left(I_F(\theta_T)\right)}. \tag{100}$$

*If the retained-data objective is removed, i.e., $\alpha = \gamma = 0$, then*

$$\nabla^2 \Psi_\rho(\theta_T) = -\rho c_F I_F(\theta_T), \tag{101}$$

*which is negative semidefinite and has strictly negative curvature along any forget-sensitive direction $v$ satisfying $v^\top I_F(\theta_T)v > 0$. Thus, pure divergence maximization contributes destabilizing curvature in forget-sensitive directions, whereas the retained-data objective acts as a curvature stabilizer.*

**Proposition 1** (Restricted curvature version)**.** *The full-space condition in Equation equation 98 is sufficient and may be conservative in heavily overparameterized networks, where the minimum eigenvalue of the full Hessian can be zero. The same argument applies on any subspace with orthonormal basis $U$. In particular, positive curvature on that subspace is guaranteed if*

$$\lambda_{\min}\big(U^\top[\gamma H_R + \alpha c_R I_R(\theta_T)]\,U\big) > \rho c_F\,\lambda_{\max}\big(U^\top I_F(\theta_T)U\big)\,. \tag{102}$$

*A natural choice is a forget-sensitive subspace contained in the range of $I_F(\theta_T)$.*

*Proof.* By Lemma 2, the Hessian of the forget divergence at $\theta_T$ is $c_F I_F(\theta_T)$, and the Hessian of the retained divergence at $\theta_T$ is $c_R I_R(\theta_T)$. The supervised retained loss contributes $H_R$. Since

$$\Psi_\rho = \mathcal{L}_{\min} - \rho\mathcal{L}_{\max}, \tag{103}$$

the Hessian is exactly Equation equation 97.

For any unit vector $v$,

$$v^\top \nabla^2 \Psi_\rho(\theta_T)v = v^\top\left(\gamma H_R + \alpha c_R I_R(\theta_T)\right)v - \rho c_F v^\top I_F(\theta_T)v \tag{104}$$

$$\geq \lambda_{\min}\left(\gamma H_R + \alpha c_R I_R(\theta_T)\right) - \rho c_F \lambda_{\max}\left(I_F(\theta_T)\right). \tag{105}$$

If Equation equation 98 holds, the right-hand side is positive for every unit vector $v$, which proves positive definiteness. □

**Remark 2** (Why JS often gives more stable maximization)**.** *Equation equation 100 shows that the destabilizing curvature scales with $c_F = f_F''(1)$. Since*

$$c_F = \frac{1}{4} \quad \textit{for JS}, \qquad c_F = 1 \quad \textit{for KL}, \qquad c_F = 2 \quad \textit{for } \chi^2, \tag{106}$$

*JS contributes less local negative curvature than KL and $\chi^2$. In addition, $D_{\mathrm{JS}}$ is bounded by $\log 2$. Therefore, JS can be interpreted as a more conservative forget-set maximization loss, which supports the empirical observation that KL-JS variants can be more stable in difficult unlearning settings. This is a local curvature interpretation, not a claim that JS is uniformly optimal in all regimes.*

### H.3.3 Post-training approaches the retained-data ERM target

The alternating min-max phase is designed to remove the forget-set influence while avoiding collapse, but it may still leave residual distortion. The post-training phase is therefore needed to move the model back toward the retained-data solution. The result below formalizes this role: under a local strong-convexity condition, the final distance to the retained-data ERM target $\theta_R^\star$ is controlled by an optimization error term and a bias term caused by retaining the teacher-divergence penalty. The PL corollary then justifies the empirical use of only a small number of post-training rounds.

**Proposition 2** (Local closeness to the retained-data ERM target after post-training)**.** *Let $\theta_R^\star$ and $\theta_\alpha^\star$ be defined by Equations equation 58 and equation 61. Assume Assumption 5. Suppose also that*

$$\|\nabla \mathcal{D}_R(\theta_R^\star;\theta_T)\| \leq B_D. \tag{107}$$

*If post-training returns $\theta_U$ satisfying*

$$\mathcal{P}_\alpha(\theta_U) - \mathcal{P}_\alpha(\theta_\alpha^\star) \leq \varepsilon_{\min}, \tag{108}$$

*then*

$$\|\theta_U - \theta_R^\star\| \leq \sqrt{\frac{2\varepsilon_{\min}}{\mu_P}} + \frac{\alpha B_D}{\mu_P}. \tag{109}$$

*If the final post-training objective sets $\alpha = 0$ and returns $\theta_U$ satisfying*

$$\mathcal{P}_0(\theta_U) - \mathcal{P}_0(\theta_R^\star) \leq \varepsilon_{\min}, \qquad \mathcal{P}_0 = \gamma\mathcal{L}_R, \tag{110}$$

*then the teacher-divergence bias term is absent and*

$$\|\theta_U - \theta_R^\star\| \le \sqrt{\frac{2\varepsilon_{\min}}{\gamma\mu_R}}. \tag{111}$$

*Furthermore, under the Lipschitz conditions in Lemma 3, the difference between any Lipschitz output-based membership score evaluated at $\theta_U$ and at the retained-data ERM target $\theta_R^\star$ is bounded by*

$$\left|\frac{1}{n_F}\sum_{x\in S_F} a(q_{\theta_U}(x)) - \frac{1}{n_F}\sum_{x\in S_F} a(q_{\theta_R^\star}(x))\right| \le L_a L_h\left(\sqrt{\frac{2\varepsilon_{\min}}{\mu_P}} + \frac{\alpha B_D}{\mu_P}\right). \tag{112}$$

*Thus, post-training gives a local parameter-space closeness guarantee to the retained-data ERM target, and this closeness transfers to Lipschitz output-based membership scores. This statement does not by itself cover discontinuous threshold attacks or establish equality with the parameter vector returned by a fresh nonconvex federated retraining run.*

*Proof.* By $\mu_P$-strong convexity of $\mathcal{P}_\alpha$,

$$\mathcal{P}_\alpha(\theta_U) - \mathcal{P}_\alpha(\theta_\alpha^\star) \ge \frac{\mu_P}{2}\|\theta_U - \theta_\alpha^\star\|^2. \tag{113}$$

Combining this inequality with Equation equation 108 gives

$$\|\theta_U - \theta_\alpha^\star\| \le \sqrt{\frac{2\varepsilon_{\min}}{\mu_P}}. \tag{114}$$

Next, because $\theta_R^\star$ minimizes $\mathcal{L}_R$ locally,

$$\nabla\mathcal{L}_R(\theta_R^\star) = 0. \tag{115}$$

Therefore,

$$\nabla\mathcal{P}_\alpha(\theta_R^\star) = \alpha\nabla\mathcal{D}_R(\theta_R^\star;\theta_T). \tag{116}$$

Since $\theta_\alpha^\star$ minimizes $\mathcal{P}_\alpha$,

$$\nabla\mathcal{P}_\alpha(\theta_\alpha^\star) = 0. \tag{117}$$

Strong monotonicity of the gradient of a $\mu_P$-strongly convex differentiable function gives

$$\mu_P\|\theta_\alpha^\star - \theta_R^\star\| \le \|\nabla\mathcal{P}_\alpha(\theta_R^\star) - \nabla\mathcal{P}_\alpha(\theta_\alpha^\star)\|. \tag{118}$$

Hence,

$$\|\theta_\alpha^\star - \theta_R^\star\| \le \frac{\alpha\|\nabla\mathcal{D}_R(\theta_R^\star;\theta_T)\|}{\mu_P} \le \frac{\alpha B_D}{\mu_P}. \tag{119}$$

Combining Equations equation 114 and equation 119 using the triangle inequality proves Equation equation 109. For the special case $\alpha = 0$, Equation equation 110 and $\gamma\mu_R$-strong convexity of $\mathcal{P}_0 = \gamma\mathcal{L}_R$ directly give Equation equation 111. Finally, substituting Equation equation 109 into the MIA transfer bound in Lemma 3 proves Equation equation 112. □

**Corollary 1** (Recovery for a fixed post-training objective)**.** *Assume Assumption 6. Fix $\alpha$ and $\theta_T$ throughout these $K$ post-training steps. If post-training uses gradient descent with*

$$0 < \eta_{\min} \le \frac{1}{L_P}, \tag{120}$$

*then after $K$ post-training steps,*

$$\mathcal{P}_\alpha(\theta_K) - \mathcal{P}_\alpha^\star \le (1 - \eta_{\min}\mu_{PL})^K\left(\mathcal{P}_\alpha(\theta_0) - \mathcal{P}_\alpha^\star\right). \tag{121}$$

*Consequently, to reach optimization accuracy $\varepsilon_{\min}$, it suffices that*

$$K \ge \frac{1}{\eta_{\min}\mu_{PL}}\log\left(\frac{\mathcal{P}_\alpha(\theta_0) - \mathcal{P}_\alpha^\star}{\varepsilon_{\min}}\right). \tag{122}$$

*Proof.* By smoothness of $\mathcal{P}_\alpha$ and the gradient-descent update,

$$\mathcal{P}_\alpha(\theta_{k+1}) \le \mathcal{P}_\alpha(\theta_k) - \eta_{\min}\left(1 - \frac{L_P \eta_{\min}}{2}\right)\|\nabla \mathcal{P}_\alpha(\theta_k)\|^2. \tag{123}$$

Since $\eta_{\min} \le 1/L_P$,

$$\mathcal{P}_\alpha(\theta_{k+1}) \le \mathcal{P}_\alpha(\theta_k) - \frac{\eta_{\min}}{2}\|\nabla \mathcal{P}_\alpha(\theta_k)\|^2. \tag{124}$$

Applying the PL condition in Equation equation 62 yields

$$\mathcal{P}_\alpha(\theta_{k+1}) - \mathcal{P}_\alpha^\star \le (1 - \eta_{\min}\mu_{PL})\left(\mathcal{P}_\alpha(\theta_k) - \mathcal{P}_\alpha^\star\right). \tag{125}$$

Iterating this recursion proves Equation equation 121. Solving for $K$ gives Equation equation 122, using $\log(1/(1-a)) \ge a$ for $a \in (0,1)$. □

**Remark 3** (Utility consequence)**.** *If the post-training phase also makes $\mathcal{L}_R(\theta_U)$ small, Lemma 3 gives the retained-data error control*

$$\widehat{\mathrm{err}}_R(\theta_U) \le \frac{\mathcal{L}_R(\theta_U)}{\log 2}. \tag{126}$$

*Thus, the same post-training mechanism that moves the model toward $\theta_R^\star$ also has a direct retained-utility interpretation.*