DRAUN: An Optimization-Agnostic Data Reconstruction Attack on Federated Unlearning

Hithem Lamri **Manaar Alam** **Haiyan Jiang** **Michail Maniatakos**
Center for Cyber Security, New York University Abu Dhabi

Abstract

Federated Unlearning (FU) enables clients to remove the influence of specific data from a collaboratively trained shared global model, addressing regulatory requirements such as GDPR and CCPA. However, this unlearning process introduces a new privacy risk: A malicious server may exploit unlearning updates to reconstruct the data requested for removal, a form of Data Reconstruction Attack (DRA). While DRAs for machine unlearning have been studied extensively in centralized Machine Learning-as-a-Service (MLaaS) settings, their applicability to FU remains unclear due to the decentralized, client-driven nature of FU. This work presents *DRAUN*, the first attack framework to reconstruct unlearned data in FU systems. *DRAUN* targets optimization-based unlearning methods, which are widely adopted for their efficiency. We theoretically demonstrate why existing DRAs targeting machine unlearning in MLaaS fail in FU and show how *DRAUN* overcomes these limitations. We validate our approach through extensive experiments on five datasets and five architectures, evaluating its performance against five popular unlearning methods, effectively demonstrating that state-of-the-art FU methods remain vulnerable to DRAs.

1 INTRODUCTION

Federated Learning (FL) enables multiple clients to collaboratively train a shared global model without exposing raw data (McMahan et al., 2017). Each client trains locally on its private dataset and sends local model updates to a central server. The server aggregates these updates over multiple communication rounds to iteratively improve the global model. Although FL preserves data privacy by keeping sensitive information on the client side, recent research shows that a malicious server can exploit the local model updates to reconstruct private client data through Data Reconstruction Attacks (DRA) (Zhu et al., 2019a; Geiping et al., 2020a; Yin et al., 2021a; Zhu and Blaschko, 2021a; Wen et al., 2022a; Zhao et al., 2024). These studies indicate that local model updates can be vectors for sensitive data leakage. Federated Unlearning (FU) extends the FL framework by allowing a client or a portion of its sensitive data to be fully removed from the global model even after several training rounds (Wu et al., 2022; Alam et al., 2024; Li et al., 2021; Halimi et al., 2022; Shaik et al., 2024; Wang et al., 2023a). To exercise the *right to be forgotten*, a client submits unlearning updates to erase the influence of the targeted data from the global model. In such a setting, *a malicious server can analyze these updates to launch DRAs targeting the samples requested for deletion.* This creates a new privacy threat inherent to the unlearning process, potentially violating data protection regulations like the GDPR (European Commission, 2024) and CCPA (Office of the Attorney General, State of California, 2024). Consequently, a detailed analysis of DRAs in FU is critical for understanding the scope of this risk and guiding the development of more robust FU methods.

Recent studies have investigated the feasibility of reconstructing unlearned data in traditional Machine Learning-as-a-Service (MLaaS) settings. Hu et al. (2024) introduced unlearning inversion attacks, demonstrating that even with black-box access, adversaries can recover labels or features of deleted data by comparing original and unlearned models. Bertran et al. (2024) further showed that even simple models like linear regression are vulnerable to exact reconstruction of deleted data after unlearning, highlighting that privacy leakage can occur without sophisticated ar-

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

chitectures or attack vectors. Both studies emphasize a key idea: *Naive unlearning methods can leave behind patterns in the model that can be used to recover unlearned samples.* However, extending these attacks directly to FU scenarios is much more challenging. In FU, unlearning is performed locally by the client, and the server sees only the resulting weight vector. It therefore lacks critical information such as which unlearning algorithm or hyper-parameters were applied. Unlike MLaaS, where model training and unlearning are centrally orchestrated and more easily observed, FU introduces several challenges: **(1)** client participation is random and changes over time, **(2)** the global model is overwritten after every aggregation step, so the signal from a single unlearning update is rapidly mixed with fresh training updates, and **(3)** the server lacks access to the specifics of local updates or unlearning operations. These factors make it difficult for an adversary to link specific changes in the model to particular unlearning requests. Moreover, FU typically requires multiple rounds to take effect, and any signal from deleted data becomes diluted over time. These challenges make it significantly harder to adapt existing DRA strategies from MLaaS to the FU setting. A brief background on FU and DRAs is provided in Appendix A. In this work, we investigate the following research question: *To what extent are federated unlearning algorithms susceptible to data reconstruction attacks by a malicious server?*

We present *DRAUN*, a novel DRA targeting FU systems. To our knowledge, this is the first work to investigate such attacks in the context of FU. *DRAUN* focuses on *optimization-based unlearning algorithms*, including both first-order (Wu et al., 2022; Alam et al., 2024; Li et al., 2021; Halimi et al., 2022) and second-order (Jin et al., 2024) methods, which are known for their high efficiency and are widely adopted in FU (Liu et al., 2025). A key strength of *DRAUN* is its ability to reconstruct unlearned data without any knowledge of the client's unlearning algorithm, which we consider an **optimization-agnostic** attack. To validate the effectiveness and generality of *DRAUN*, we conduct comprehensive experiments across *five datasets* and *five model architectures*. The qualitative results shown in Figure 1 illustrate the reconstruction performance of different methods across representative unlearned samples. We also assess a range of potential defense strategies, providing insights to guide the development of more resilient FU techniques.

Our **contributions** are as follows: **(1)** We introduce *DRAUN*, the first attack framework capable of reconstructing unlearned client data specifically in FU settings, whereas prior efforts have focused exclusively on centralized MLaaS environments. **(2)** We provide a formal theoretical analysis showing why conventional DRAs from MLaaS can be ineffective in optimization-based FU. **(3)** *DRAUN* operates in an optimization-agnostic manner, requiring no knowledge of the client's unlearning method. **(4)** We evaluate *DRAUN* across five datasets and five model architectures, and demonstrate its effectiveness against five widely used unlearning algorithms, four first-order and one second-order. **(5)** We open-source the implementation of *DRAUN* in this (repository).

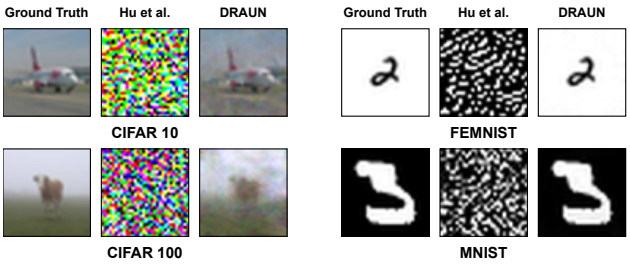

Figure 1: Reconstruction of deleted samples after unlearning using ABL (Li et al., 2021). **Column 1**: Ground truth images (one randomly selected sample each from CIFAR10, CIFAR100, FEMNIST, and MNIST). **Column 2**: Reconstructions using a state-of-the-art DRA on MLaaS unlearning (Hu et al., 2024) directly extended to FU, which fails to recover any meaningful images. **Column 3**: Reconstructions from *DRAUN* that closely resemble the ground truth.

2 *DRAUN* METHODOLOGY

2.1 Threat Model

We consider a FL setting where a central server coordinates training across multiple clients. The system supports optimization-based FU, allowing clients to remove specific subsets of their local data from the global model. Following the standard approach used in the literature (Liu et al., 2025), we consider the unlearning process as follows: **(1)** a client submits an unlearning request, **(2)** the server approves the request and ensures the client stays active in upcoming rounds, **(3)** in each round, the client performs a fixed number of local unlearning epochs on the data to be removed and sends the updated model to the server, and **(4)** once the unlearning is done, the client notifies the server. As unlearning requires access to the data being erased, all computations occur on the client side.

Client Capabilities: Clients initiate unlearning and perform the necessary local computations. Each client selects its own unlearning algorithm. They also share the expected metadata with the server, including the total size of their local dataset and the size of the subset being deleted (Dimitrov et al., 2022).

Server Capabilities: We assume an honest-but-curious server that follows the protocol but may analyze

client updates (Zhu et al., 2019a; Geiping et al., 2020a; Yin et al., 2021a; Zhu and Blaschko, 2021a; Wen et al., 2022a; Zhao et al., 2024; Dimitrov et al., 2022). The server has white-box access to the global model and to all received updates. It manages unlearning requests, tracks client participation, and performs model aggregation. Importantly, the server does not know which unlearning algorithm each client employs. In this paper, we use *Gradient Inversion Attack* (GIA) as a representative Data Reconstruction Attack (Wen et al., 2022a; Geiping et al., 2020a), a well-established approach showing that training inputs can often be accurately reconstructed from model weights or gradients even in the absence of their corresponding labels (Dimitrov et al., 2022; Yin et al., 2021b). Following standard practice in the GIA literature, we assume that the server knows the labels associated with each client's data.

2.2 Problem Landscape and Motivation

We consider a server aiming to train a neural network $F(\theta_s, \cdot)$ with parameters θ_s for image classification, using a set of clients $\mathcal{C}$. Each client c performs local training using Stochastic Gradient Descent (SGD) on its dataset $\mathcal{D}_c = \{(x_i, y_i)\}_{i=1}^{n}$, where x_i is an input image and y_i is its label. Upon receiving an unlearning request, the server sends the current global model θ_s to the requesting client, which aims to remove a subset $\mathcal{D}_u \subseteq \mathcal{D}_c$. The retained data is denoted $\mathcal{D}_r = \mathcal{D}_c \setminus \mathcal{D}_u$. Let $(x_u, y_u) \in \mathcal{D}_u$ represent the samples the client wishes to forget, and $(x_r, y_r) \in \mathcal{D}_r$ those it intends to retain. The client then applies a local unlearning algorithm $\mathcal{A}_c$, which takes $\mathcal{D}_u$, $\mathcal{D}_r$, and the current model θ_s as input, and returns an updated local model $\theta_c := \mathcal{A}_c(\mathcal{D}_u, \mathcal{D}_r, \theta_s)$. This algorithm typically involves optimization over a custom unlearning loss $\mathcal{L}_u$, controlled by hyperparameters such as the unlearning rate η, batch size m, and number of local epochs $\mathcal{E}$.

2.3 GIA Challenges in FU

GIA reconstructs private data by optimizing dummy inputs to match observed gradients (Zhu et al., 2019a; Geiping et al., 2020a; Yin et al., 2021a). Its success relies on the adversary knowing the client's loss function and replicating its gradient dynamics (see Appendix B). FU breaks this assumption since clients may adopt heterogeneous unlearning algorithms with modified objectives, additional loss terms, or different update rules as summarized in Table 1. These factors change the gradient dynamics in ways unknown to the server. Even if the server restricts attention to a finite set of candidate algorithms, the uncertainty introduced by hyperparameters makes reconstruction intractable. Learning rates, regularization strengths, batch sizes, and epoch counts all have a strong effect on the update distribution. As shown in Table 1, each unlearning method introduces its own hyperparameters, and these choices substantially influence the resulting model updates. To compensate, the server would need to run GIA across every algorithm–parameter configuration, requiring $k \times n$ reconstructions where k is the number of algorithms and n the number of parameter settings. This overhead is computationally infeasible and still only succeeds if the exact configuration is matched. The challenge becomes even harder because the space of unlearning algorithms is open-ended. Clients can freely adapt existing methods or implement proprietary routines, producing an effectively unbounded hypothesis space. In this setting, even small deviations in objectives or optimization schedules cause gradient dynamics to deviate from those assumed in GIA, leading to poor or meaningless reconstructions. These factors together make classical GIA unsuitable for FU and motivate the need for attacks that generalize across unknown objectives, hyperparameters, and algorithms.

Table 1: Local update rules for FU methods ($\alpha, \beta, \gamma, \delta, \eta$ are method-specific hyperparameters).

Algorithm	Local update rule
(Wu et al., 2022)	$\theta_u \leftarrow \theta_s + \eta \nabla_{\theta_s} \mathcal{L}(x_u, y_u)$
(Halimi et al., 2022)	$\theta_u \leftarrow \theta_s + \eta \nabla_{\theta_s} \mathcal{L}(x_u, y_u) + \delta \lVert \theta_s - \theta_u \rVert$
(Li et al., 2021)	$\theta_u \leftarrow \theta_s - \eta (\nabla_{\theta_s} \mathcal{L}(x_r, y_r) - \nabla_{\theta_s} \mathcal{L}(x_u, y_u))$
(Alam et al., 2024)	$\theta_u \leftarrow \theta_s - \eta(\gamma \lVert \frac{\theta_u}{\theta_s} \rVert + \alpha \nabla_{\theta_s} \mathcal{L}(x_r, y_r)$ - $\beta \nabla_{\theta_s} \mathcal{L}(x_u, y_u)$)

2.4 *DRAUN* Overview

DRAUN builds upon the standard structure of a GIA, where the attacker reconstructs private training data by minimizing the difference between true and simulated (dummy) gradients. Specifically, we adopt the similarity-based loss introduced by Geiping et al. (2020a), formulated in Equation (1). The loss minimizes the cosine distance between the client's true gradient ∇_c and a dummy gradient $\tilde{\nabla}_c$, with an additional Total Variation (TV) penalty (Rudin et al., 1992), controlled by hyperparameter λ_{TV}, that acts as a regularizer for image smoothness:

$$\begin{aligned} x &= \arg\min_{\tilde{x} \in \mathcal{X}} \mathcal{L}_{\text{sim}}(\nabla_c, \tilde{\nabla}_c) \\ &= \arg\min_{\tilde{x} \in \mathcal{X}} \Big(1 - \frac{\langle \nabla_c, \tilde{\nabla}_c \rangle}{\lVert \nabla_c \rVert \, \lVert \tilde{\nabla}_c \rVert} + \lambda_{\text{TV}} \, \text{TV}(\tilde{x})\Big) \end{aligned} \tag{1}$$

As discussed in Section 2.3, this standard GIA objective is effective only under specific conditions, when the client's local loss $\mathcal{L}_u$ depends solely on the unlearn dataset $\mathcal{D}_u$. However, many FU algorithms use update rules that depend on both $\mathcal{D}_u$ and a retain dataset $\mathcal{D}_r$ (see Table 1). To extend GIAs to these more general settings, *DRAUN* simulates the client's local update using a surrogate optimization procedure.

The server first estimates the client's average gradient

Algorithm 1 Overview of *DRAUN*

1: **Input:** θ_s, θ_c, $|\mathcal{D}_u|$, m $\mathcal{E}$, λ_{TV}, β, η_{unl}, η_{rec}, y_r, y_u
2: $U_c \leftarrow \frac{|\mathcal{D}_u| \cdot \mathcal{E}}{m}$ ▷ Number of local steps
3: $\nabla\bar{\theta}_c \leftarrow \frac{1}{U_c}(\theta_s - \theta_c)$
4: $\tilde{x}_r, \tilde{x}_u \leftarrow \mathcal{I}(|\mathcal{D}_u|)$
5: **for** $t = 1$ to T **do**
6: $\tilde{\nabla}^{(1)}, \tilde{\nabla}^{(0)} \leftarrow \mathcal{A}_{\text{approx}}(\theta_s, \tilde{x}_u, y_u, \tilde{x}_r, y_r, \mathcal{E}, \eta_{\text{unl}})$
7: $\ell_1 \leftarrow \mathcal{L}_{\text{sim}}(\nabla\bar{\theta}_c, \tilde{\nabla}^{(1)}) + \lambda_{\text{TV}} \cdot (\beta \cdot TV(\tilde{x}_u) + (1-\beta) \cdot TV(\tilde{x}_r))$ ▷ $\beta \approx 1$
8: $\ell_0 \leftarrow \mathcal{L}_{\text{sim}}(\nabla\bar{\theta}_c, \tilde{\nabla}^{(0)}) + \lambda_{\text{TV}} \cdot (\beta \cdot TV(\tilde{x}_u) + (1-\beta) \cdot TV(\tilde{x}_r))$ ▷ $\beta \approx 1$
9: $\ell \leftarrow \min(\ell_0, \ell_1)$
10: $\tilde{x}_u \leftarrow \tilde{x}_u - \eta_{\text{rec}} \cdot \frac{\partial \ell}{\partial \tilde{x}_u}$
11: $\tilde{x}_r \leftarrow \tilde{x}_r - \eta_{\text{rec}} \cdot \frac{\partial \ell}{\partial \tilde{x}_r}$
12: **end for**
13: **return** $\tilde{x}_u$

Algorithm 2 *DRAUN*'s Algorithm Approximation with $\mathcal{A}_{\text{approx}}$ (Algorithm 1, line 6)

Input: θ_s, $\tilde{x}_u$, y_u, $\tilde{x}_r$, y_r, $\mathcal{E}$, η_{unl}
Initialize two dummy models: $\tilde{\theta}_c^{(0)} \leftarrow \theta_s$, $\tilde{\theta}_c^{(1)} \leftarrow \theta_s$
for $t = 1$ to $\mathcal{E}$ **do**
  Update $\tilde{\theta}_c^{(1)}$ **with** $\alpha = 1$**:** ▷ gradient difference

$$\tilde{\theta}_c^{(1)} \leftarrow \tilde{\theta}_c^{(1)} - \eta_{\text{unl}}\Big[\nabla_\theta \mathcal{L}(\tilde{\theta}_c^{(1)}, \tilde{x}_r, \tilde{y}_r) - \nabla_\theta \mathcal{L}(\tilde{\theta}_c^{(1)}, \tilde{x}_u, \tilde{y}_u) + \delta\, \nabla_\theta \|\tilde{\theta}_c^{(1)} - \theta_s\|_2\Big]$$

  Update $\tilde{\theta}_c^{(0)}$ **with** $\alpha = 0$**:** ▷ gradient ascent

$$\tilde{\theta}_c^{(0)} \leftarrow \tilde{\theta}_c^{(0)} + \eta_{\text{unl}} \cdot \nabla_\theta \mathcal{L}(\tilde{\theta}_c^{(0)}, \tilde{x}_u, \tilde{y}_u) + \delta \cdot \nabla_\theta \|\tilde{\theta}_c^{(0)} - \theta_s\|_2$$

end for
Return:
  $\tilde{\nabla}^{(1)} \leftarrow \tilde{\theta}_c^{(1)} - \theta_s$ ▷ Averaged gradient for $\alpha = 1$
  $\tilde{\nabla}^{(0)} \leftarrow \tilde{\theta}_c^{(0)} - \theta_s$ ▷ Averaged gradient for $\alpha = 0$

using the received model θ_c and the global model θ_s as $\nabla\bar{\theta}_c = \frac{1}{U_c}(\theta_s - \theta_c)$ (Algorithm 1, line 3), where U_c is the number of client steps. The server then attempts to reconstruct the unlearned input x_u by optimizing dummy inputs $\tilde{x}_u$ and $\tilde{x}_r$, which represent proxy samples for the unlearn and retain subsets respectively. Since the server does not know the client's actual unlearning algorithm $\mathcal{A}_c$, it uses a surrogate procedure $\mathcal{A}_{approx}$ (Algorithm 2) for its approximation.

This routine simulates the client update by minimizing a surrogate unlearning loss ($\tilde{\mathcal{L}}_u$), defined by the following constrained objective (Equation (2)):

$$\begin{aligned} &\min_{\theta_c \in \Theta} \tilde{\mathcal{L}}_u = \alpha\, \mathcal{L}(\theta_s, x_r, y_r) - \mathcal{L}(\theta_s, x_u, y_u), \\ &\text{s.t.} \qquad \|\theta_c - \theta_s\|_2 \le \delta. \end{aligned} \tag{2}$$

Here, $\alpha \in [0, 1]$ determines the contribution of the retain dataset, δ constrains the model divergence from the global model, and $\mathcal{L}$ is the cross-entropy loss. The constraint is relaxed into an unconstrained objective using a Lagrange multiplier, resulting in Equation (3):

$$\min_{\theta_c \in \Theta} \alpha \cdot \mathcal{L}(\theta_s, x_r, y_r) - \mathcal{L}(\theta_s, x_u, y_u) + \delta \cdot \|\theta_c - \theta_s\|_2 \tag{3}$$

Algorithm 2 optimizes two dummy models using an unlearning rate η_{unl} under this objective: One for $\alpha = 1$ (gradient difference-based updates), and another for $\alpha = 0$ (gradient ascent-based updates). After $\mathcal{E}$ optimization steps, it returns two surrogate gradients: $\tilde{\nabla}^{(1)}$ and $\tilde{\nabla}^{(0)}$. These are intended to approximate the gradient update a client would have produced under two extremes of unlearning behavior. Since the server has no knowledge of the client's actual unlearning algorithm, it uses both of these as candidates for matching the observed averaged gradient $\nabla\bar{\theta}_c$.

In each iteration of Algorithm 1, the server computes two loss values: ℓ_1 and ℓ_0, corresponding to the cosine similarity between $\nabla\bar{\theta}_c$ and the surrogate gradients $\tilde{\nabla}^{(1)}$ and $\tilde{\nabla}^{(0)}$, respectively (lines 7-8). These losses use the formulation in Equation (1), and each includes a TV regularization term applied separately to $\tilde{x}_u$ and $\tilde{x}_r$. A mixing coefficient $\beta \approx 1$ balances the contribution of the TV terms, prioritizing smoothness in $\tilde{x}_u$. The final loss ℓ is selected as the minimum of ℓ_0 and ℓ_1 (line 9). The dummy inputs are then updated via gradient descent with respect to this loss (lines 10-11), using a step size η_{rec}, and continuing for T iterations.

A critical design choice in *DRAUN* is the initialization of dummy inputs. Since the server lacks access to the true structure of the client's data, the dummy inputs for both $\mathcal{D}_u$ (unlearn) and $\mathcal{D}_r$ (retain) are initialized as random tensors. Specifically, we sample two sets of size $|\mathcal{D}_u|$ each from a uniform distribution over $[0, 1]$, denoted as $\mathcal{U}([0, 1], |\mathcal{D}_u|)$ in Algorithm 3. Our goal is not to reconstruct the retain set, but to use it to help denoise the gradients from the unlearn set and facilitate accurate reconstruction. To improve optimization and avoid convergence to poor local minima, a known issue in unlearning algorithms, we ensure that the dummy inputs for $\mathcal{D}_u$ and $\mathcal{D}_r$ are well separated at the start of reconstruction. This is achieved by iteratively adding multivariate Gaussian noise (with standard deviation σ and dimensionality d_x, matching that of a single input sample) to each image in $\mathcal{D}_r$, until the minimum Frobenius distance between corresponding image pairs in $\mathcal{D}_u$ and $\mathcal{D}_r$ exceeds a predefined threshold Δ. This guarantees the convex hulls of the unlearn and retain dummy inputs are disjoint: $\text{Conv}\{\tilde{\mathcal{D}}_u\} \cap \text{Conv}\{\tilde{\mathcal{D}}_r\} = \varnothing$. This initialization strategy, detailed in Algorithm 3, is particularly important for unlearning methods that rely on coupled update dynamics. A theoretical justification is provided in Appendix C.

3 THEORETICAL ANALYSIS

In this section, we demonstrate why classical GIAs fail to reconstruct unlearned inputs from unlearning updates involving two datasets, such as gradient difference methods (Li et al., 2021; Alam et al., 2024). Recall

Algorithm 3 *DRAUN*'s Input Initialization Function $\mathcal{I}$ (Algorithm 1, line 4)

Input: $|\mathcal{D}_u|$
$\tilde{x_u} \leftarrow \mathcal{U}([0,1], |\mathcal{D}_u|)$ ▷ Initialize dummy unlearn input
$\tilde{x_r} \leftarrow \mathcal{U}([0,1], |\mathcal{D}_u|)$ ▷ Initialize dummy retain input
for $i = 1$ to $|\mathcal{D}_u|$ **do** ▷ The i-th samples in $\tilde{x_u}, \tilde{x_r}$
  while $\|\tilde{x_u}^{(i)} - \tilde{x_r}^{(i)}\|_F \leq \Delta$ **do** ▷ $\|.\|_F$ Frobenius norm
    $\epsilon \sim \mathcal{N}(0, \sigma^2 \cdot I_{d_x})$
    $\tilde{x_r}^{(i)} \leftarrow \tilde{x_r}^{(i)} + \epsilon$
  end while
end for
Return: $\tilde{x_u}, \tilde{x_r}$

that $\mathcal{L}_u$ and $\tilde{\mathcal{L}}_u$ are client's local unlearning loss and the server's surrogate unlearning loss (used to generate dummy gradients), respectively. $\tilde{\mathcal{L}}_u$ depends on the dummy unlearn input $\tilde{x}_u$; hence, the similarity loss function $\mathcal{L}_{\text{sim}}$ from Equation 1 implicitly depends on $\tilde{x}_u$, not just model parameters θ. For clarity, we denote $\tilde{x}_u$ by x. To avoid high-order tensor calculations, we treat inputs as flattened vectors of dimension d_x. While we assume single unlearn/retain inputs for simplicity, this analysis extends to multiple inputs via averaging. To show that classical GIA fails, we adopt common theoretical assumptions—twice differentiability of the surrogate loss functions, Lipschitzness, and strong convexity. Rigorous details are provided in Appendix C.

Theorem 1 (GIA failure (Informal))**.** *Without loss of generality (WLOG), assume $\mathcal{L}_{sim} \triangleq \|\cdot\|_2^2$. Suppose the client's unlearning loss $\mathcal{L}_u$ depends on both x_u and x_r through $\mathcal{L}_u = \mathcal{L}(\theta, x_r, y_r) - \mathcal{L}(\theta, x_u, y_u)$. Then the ground-truth unlearn input x_u is not a local minimizer of the similarity objective $\mathcal{L}_{sim}$.*

Theorem 2 (GIA reconstruction error (Informal))**.** *Let x_u^* be a minimizer of the classical GIA objective as defined in Equation 4:*

$$x_u^* = \arg\min_{x \in \mathcal{X}} \mathcal{L}_{sim}\Big(\nabla_\theta \mathcal{L}_u, \nabla_\theta \tilde{\mathcal{L}}_u\Big). \tag{4}$$

The reconstruction error, defined as the ℓ_2-norm between the GIA output and the ground-truth input, satisfies the following inequality:

$$\|x_u^* - x_u\|_2 \geq \frac{\left\|J^\top \nabla_\theta \mathcal{L}(\theta, x_r, y_r)\right\|_2}{\mu_x \|J\|_F + 2\mu_\theta \left\|\nabla_\theta \mathcal{L}_u\right\|_2}. \tag{5}$$

Interpretation: Theorem 1 shows that classical GIA fails to converge to x_u when x_r influences $\mathcal{L}_u$, and Theorem 2 provides a nontrivial lower bound on its reconstruction error (Equation 5), implying that perfect reconstruction ($\|x_u^* - x_u\|_2 = 0$) is impossible.

The proofs of Theorems 1 and Theorem 2, together with a more detailed theoretical analysis of the failure of GIA for second-order-based FU methods, and the correctness of DRAUN, are provided in Appendix C.

4 EXPERIMENTS

4.1 Experimental Setup

Datasets and Models: We evaluate *DRAUN* on five standard image classification datasets: CIFAR10 (Krizhevsky, 2009), CIFAR100 (Krizhevsky, 2009), MNIST (LeCun et al., 1998), ImageNet (Deng et al., 2009) (a high-resolution dataset), and FEMNIST (Caldas et al., 2018), which is a standard *non-iid benchmark* (see Appendix D.1 for details on datasets). To ensure a fair comparison with the reconstruction performance of Hu et al. (2024), we use an 8-layer ConvNet for all main experiments (see Appendix D.2 for its architecture). In addition to ConvNet, the reconstruction results using other model architectures, such as MLP, LeNet (LeCun et al., 1998), ResNet18, are reported in Appendix E.1, and we have discussed the results of a particular study on the complex model architecture ResNet152 (He et al., 2016) in Section 4.2.

Federated Unlearning Setup: We simulate an FL environment with 100 clients. In each communication round, 10 clients are selected uniformly at random to participate. Each selected client trains a local model using stochastic gradient descent with a learning rate of 0.1, for 2 local epochs, and a batch size of 128. The global model is trained for 100 rounds and converges by round 90 based on clean accuracy. After convergence, we simulate FU by targeting the data of a single client for removal. We evaluate both single-step and multi-step FU methods, where the number of unlearning steps is denoted by $\mathcal{E} \in \{1, 2, 4\}$. We consider $\mathcal{E} = 1$ for the main experiments; results for $\mathcal{E} = 2$ and $\mathcal{E} = 4$ are provided in Appendix E.2. To evaluate the effect of target data size, we vary the size of the unlearned dataset as $|\mathcal{D}_u| \in 2^i$, where $i \in [0, 7]$. We evaluate four FU methods based on first-order optimization proposed by Wu et al. (2022), Halimi et al. (2022), Alam et al. (2024), and ABL (Li et al., 2021).

Reconstruction Setup: For reconstruction, *DRAUN* uses the Adam optimizer with $\eta_{\text{rec}} = 0.1$, $\lambda_{\text{reg}} = 10^{-6}$, and $\beta = 0.9$. Reconstruction iterations T varies between 6,000 and 24,000 across experiments. The other hyperparameters used in Algorithm 2 and Algorithm 3 are $\eta_{\text{unl}} = 0.1$, $\delta = 10.0$, $\Delta = 5.0$, and $\sigma = 1$.

Evaluation Metrics: To evaluate the quality of reconstructed images, we use standard image similarity metrics that assess luminance, contrast, and structure with the ground truth. Specifically, we use the Structural Similarity Index Measure (SSIM) (Wang et al., 2004), the Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018), and the Peak Signal-to-Noise Ratio (PSNR) (Hu et al., 2024). Lower LPIPS scores and higher PSNR and SSIM values (with SSIM close to 1) indicate better reconstruction quality.

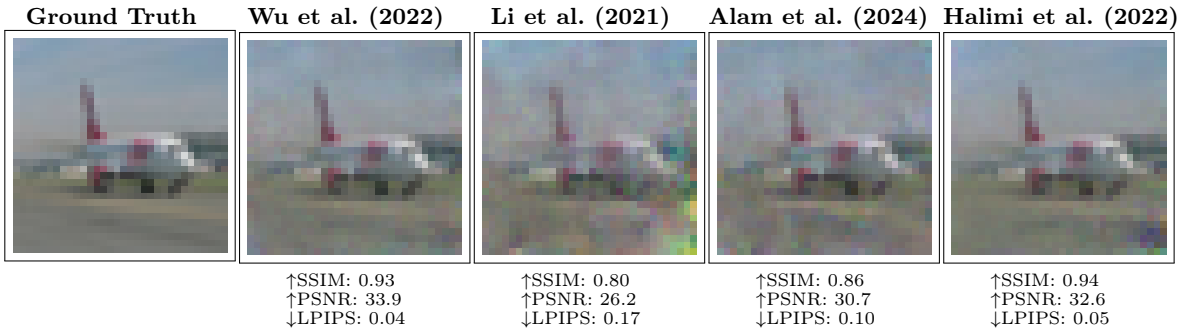

Figure 2: *DRAUN* reconstructions from local updates of four unlearning methods on CIFAR10. Higher SSIM and PSNR, and lower LPIPS scores indicate strong similarity to the original images, revealing residual information in the updates.

Table 2: ImageNet (224×224) with ResNet18: *DRAUN* reconstruction quality (mean ± std) across four unlearning algorithms.

Method	SSIM	PSNR	LPIPS
(Wu et al., 2022)	0.29 ± 0.07	14.99 ± 1.80	0.67 ± 0.06
(Halimi et al., 2022)	0.31 ± 0.07	15.86 ± 1.53	0.64 ± 0.05
(Li et al., 2021)	0.24 ± 0.04	14.32 ± 1.11	0.68 ± 0.07
Alam et al. (2024)	0.24 ± 0.04	14.70 ± 1.32	0.69 ± 0.06

4.2 *DRAUN* Reconstruction Efficiency

Visual Reconstruction Results: We first evaluate the effectiveness of *DRAUN* in reconstructing unlearned images from local updates produced by four unlearning algorithms. Figure 2 shows a randomly selected example from CIFAR10, together with reconstructions generated from updates produced by Wu et al. (2022), ABL Li et al. (2021), Alam et al. (2024), and Halimi et al. (2022). The reconstructed images closely resemble the originals, as reflected by high SSIM and PSNR values and low LPIPS scores. These results indicate that current unlearning methods leave substantial residual information in their local updates, which *DRAUN* can exploit to recover the original data with high fidelity.

To assess whether this vulnerability persists in more challenging settings, we extend the evaluation to ImageNet with 224×224 images using a ResNet18 model. Table 2 reports the average reconstruction quality over 9 randomly selected images across the same four unlearning algorithms. Although the numerical metrics are lower than those on CIFAR10, this degradation is expected given the higher resolution and complexity of ImageNet. Importantly, the reconstructions still preserve meaningful visual patterns and recognizable structure, suggesting that the privacy leakage remains significant even on a large-scale dataset.

We further evaluate *DRAUN* in an even more challenging setting on ImageNet with a ResNet152 model under the Alam et al. unlearning method. Even in this case, reconstruction quality remains notable, achieving SSIM = 0.30, PSNR = 15.8, and LPIPS = 0.62.

Additional reconstructed examples for CIFAR100, FEMNIST, and MNIST are provided in Appendix E.3.

Quantitative Comparison with State-of-the-Art: Table 3 provides a quantitative comparison between *DRAUN* and the state-of-the-art DRA by Hu et al. (2024), originally developed for MLaaS unlearning and directly adapted here to the FU setting. The evaluation includes all four unlearning algorithms discussed earlier, applied to CIFAR10, CIFAR100, MNIST, and FEMNIST datasets. Each entry shows the average performance over reconstructions of 10 randomly selected images per dataset, using the same metrics described previously. On all datasets, *DRAUN* clearly outperforms Hu et al. (2024) when using updates from ABL (Li et al., 2021) and Alam et al. (2024). For example, on CIFAR10 with ABL (Li et al., 2021), Hu et al. (2024) achieves near-zero SSIM (0.0038) and low PSNR (5.48) values and extremely high LPIPS (0.9358), indicating failed reconstructions (see Figure 1 for examples of such failed reconstructions). In contrast, *DRAUN* achieves a better reconstruction (SSIM: 0.6407, PSNR: 21.11, LPIPS: 0.3175). A similar gap is observed for Alam et al. (2024). However, for the unlearning algorithms by Wu et al. (2022) and Halimi et al. (2022), Hu et al. (2024) slightly outperforms or matches *DRAUN* in some cases (e.g., CIFAR100 with Wu et al. (2022)). This is due to the design of these specific unlearning algorithms, which do not rely on coupled update dynamics of the retained and unlearned datasets (see Table 1). As a result, Hu et al. (2024) can still reconstruct meaningful images in these cases. Nonetheless, *DRAUN* consistently achieves better performance than Hu et al. (2024) across most datasets and unlearning algorithms, especially on complex datasets such as CIFAR10 and CIFAR100. These results validate the robustness and generalizability of *DRAUN* in FU contexts.

Reconstruction Convergence Analysis: Figure 3 shows how the reconstruction loss changes over reconstruction steps for *DRAUN* and Hu et al. (2024), evaluated with four unlearning algorithms on the CIFAR10

Table 3: Reconstruction quality comparison between *DRAUN* and the method by Hu et al. (2024) across four unlearning algorithms and datasets. Higher SSIM and PSNR, and lower LPIPS scores indicate better reconstruction quality.

Dataset	Methods	Wu et al. (2022)			ABL (Li et al., 2021)			Alam et al. (2024)			Halimi et al. (2022)		
		↑SSIM	↑PSNR	↓LPIPS	↑SSIM	↑PSNR	↓LPIPS	↑SSIM	↑PSNR	↓LPIPS	↑SSIM	↑PSNR	↓LPIPS
CIFAR10	Hu et al. (2024)	0.8698	27.17	0.1350	0.0038	5.48	0.9358	-0.0062	5.43	0.9201	0.8488	27.04	0.1523
	DRAUN	0.8503	26.06	0.1509	0.6407	21.11	0.3175	0.6114	20.85	0.3328	0.8374	25.49	0.1669
CIFAR100	Hu et al. (2024)	0.8786	25.68	0.1294	0.0008	4.67	0.9007	-0.0066	4.87	0.8966	0.7883	23.42	0.2122
	DRAUN	0.8775	24.98	0.1374	0.6478	19.32	0.3357	0.7341	21.86	0.2704	0.8795	26.26	0.1170
MNIST	Hu et al. (2024)	0.9402	37.93	0.0041	-0.0148	2.04	0.8080	-0.0097	2.19	0.8209	0.9425	38.94	0.0029
	DRAUN	0.9247	36.39	0.0075	0.7603	28.46	0.1400	0.7626	29.01	0.1157	0.9283	36.67	0.0056
FEMNIST	Hu et al. (2024)	0.9986	53.57	0.0001	-0.0000	1.77	0.8880	0.0016	1.39	0.8991	0.9620	45.36	0.0070
	DRAUN	0.9978	52.25	0.0003	0.9929	44.27	0.0007	0.9947	45.44	0.0004	0.9970	51.15	0.0003

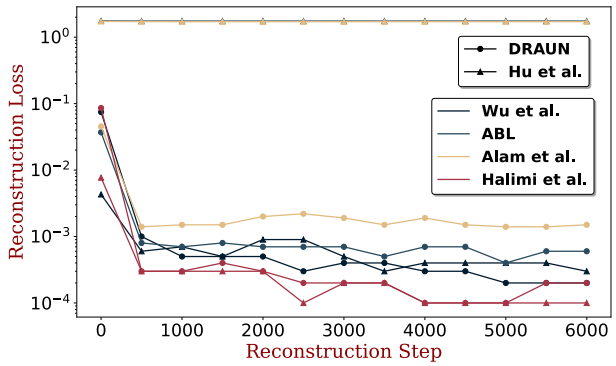

Figure 3: Reconstruction loss over reconstruction steps for *DRAUN* and Hu et al. (2024) across four unlearning algorithms on CIFAR10.

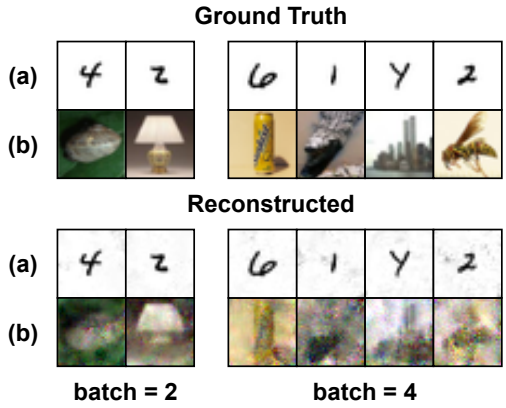

Figure 4: Reconstructed image batches using *DRAUN* from ABL (Li et al., 2021) with batch sizes of 2 and 4 for: **(a)** FEMNIST and **(b)** CIFAR100.

dataset. *DRAUN* consistently achieves low reconstruction loss across all algorithms, showing that it can efficiently recover the data. In contrast, Hu et al. (2024) fails to converge for ABL (Li et al., 2021) and Alam et al. (2024), getting stuck in high-loss regions. This suggests that *DRAUN* can effectively approximate the client's unlearning algorithm (see $\mathcal{A}_{\text{approx}}$ in Algorithm 2), regardless of which specific method is used, and produce accurate reconstructions.

Batch Image Reconstruction: Batch reconstruction is a well-known challenge in DRA literature (Geiping et al., 2020a). When gradients are generated from multiple images, their signals become entangled, making it difficult to separate and recover individual examples. To further probe the limits of *DRAUN*, we evaluate its performance in batch unlearning settings. While all previous experiments focused on unlearning and reconstructing a single image, Figure 4 shows how *DRAUN* performs on FEMNIST and CIFAR100 when the client unlearns a randomly selected batch of images with batch sizes of 2 and 4. We use ABL (Li et al., 2021) to perform unlearning and apply *DRAUN* to reconstruct each image in the batch.

On FEMNIST, which consists of non-iid grayscale images with relatively low visual complexity, *DRAUN* reconstructs the images with high fidelity even when the batch size is greater than one. Fluctuations in client participation do not affect this behavior. *DRAUN* performs reconstruction immediately after the first unlearning round using only the model delta; it does not depend on successful training or unlearning, nor does it require assumptions about client behavior or participation frequency. This makes *DRAUN* robust to participation variability and data heterogeneity. On CIFAR100, which contains more diverse and visually complex images, the reconstructions become increasingly noisy as the batch size grows from 2 to 4. Nevertheless, the recovered images still preserve recognizable structure and salient visual features from the ground truth. By contrast, Hu et al. (2024) do not produce meaningful reconstructions even in the single-image setting, and are therefore not applicable to the more difficult problem of recovering multiple images from mixed gradients.

Additional batch-reconstruction results for other datasets and batch sizes are provided in Appendix E.4, while the corresponding reconstruction-time results for different model architectures and batch sizes are summarized in Tables 4 and 5. Although data reconstruction is inherently compute-intensive, *DRAUN* remains efficient and scales substantially better than classical Gradient Inversion Attacks (GIA), especially on high-resolution inputs and deeper architectures. Table 4 re-

Table 4: Single-image reconstruction times (seconds) for classical GIA vs. *DRAUN* across representative architectures.

Model	GIA Time (s)	*DRAUN* Time (s)
ConvNet64 (CIFAR10)	655.0	754.2
ResNet18 (ImageNet)	8044.1	6963.6
ResNet152 (ImageNet)	61492.9	18949.2

Table 5: *DRAUN* batch reconstruction time (seconds).

Batch Size	Time (s)
8	1037
64	1513
128	9530

ports single-image reconstruction times (in seconds) for classical GIA and *DRAUN* across three representative architectures. Despite being more robust and generalizable, *DRAUN* is up to 3× faster than classical GIA on deeper models such as ResNet152, while maintaining comparable runtime on smaller architectures such as ConvNet64. Importantly, *DRAUN* also achieves substantially better reconstruction quality across all settings, particularly in FU where classical GIA often fails because its gradient assumptions are violated.

We also evaluate the efficiency of *DRAUN* in the batch setting. Table 5 shows the total reconstruction time for batch sizes of 8, 64, and 128. Although total runtime increases with batch size, the growth is far from exponential, and the amortized per-image cost remains substantially lower for larger batches than in the small-batch regime. These results indicate that *DRAUN* scales favorably to batch reconstruction while preserving practical efficiency.

DRAUN's Components Evaluation: We separately evaluated individual contribution of DRAUN: (1) *TV-norm regularization:* represented in Equation (1), effect on the reconstruction using same setup as in Figure 2. The results shows that when we disabled the TV loss by setting $\lambda_{TV} = 0$. The reconstruction quality degraded substantially: SSIM dropped from 0.86 to 0.6330, PSNR from 30.7 to 25.43, and LPIPS increased from 0.10 to 0.2260. These results quantitatively demonstrate that TV regularization plays a crucial role in stabilizing the optimization and improving visual fidelity. (2) *Input separation*: We conducted additional ablation experiments by varying the Frobenius norm threshold Δ. The results show a consistent trend: increasing Δ improves reconstruction quality. With $\Delta = 0.0$ (no enforced separation), the reconstruction quality is poor (SSIM: 0.3769, PSNR: 18.25, LPIPS: 0.4279). At $\Delta = 0.001$, quality improves noticeably (SSIM: 0.6690, PSNR: 25.45, LPIPS: 0.2639). Further increases to $\Delta = 0.1$ and $\Delta = 1.0$ produce continued gains, with SSIM rising to 0.7333 and 0.8051, PSNR to 26.48 and 27.47, and LPIPS dropping to 0.2154 and 0.1626, respectively. This empirical trend aligns with the theoretical motivation in Section 2.4 and Appendix C, where we show that well-separated initializations reduce gradient coupling and help avoid convergence to poor local minima.

5 DISCUSSION

***DRAUN*'s Effectiveness against Second-Order Unlearning:** Optimization-based FU typically relies on first-order methods, as second-order approaches involve costly Hessian inversions. Nonetheless, Jin et al. (2024) has recently explored second-order unlearning in FU. We evaluate *DRAUN*'s ability to reconstruct data from such second-order unlearning updates. Reconstruction from second-order unlearning follows the same overall approach as in the first-order case, but instead of matching gradients, the server simulates Newton updates by computing Hessian-vector products (HVPs) using dummy inputs and minimizes the differences between the true and simulated HVPs to recover the original client data. A detailed discussion on this second-order reconstruction mechanism is provided in Appendix F. Figure 5(a) shows two random examples from the CIFAR10 dataset reconstructed using *DRAUN* from second-order unlearned updates, demonstrating strong visual reconstruction. The model used in this experiment is an MLP, consistent with the setup used by Jin et al. (2024).

Potential Defense against *DRAUN*: Given the strong reconstruction results above, we next examine whether simple perturbation-based defenses can mitigate *DRAUN*. We consider two standard techniques: (1) adding Gaussian noise to unlearning updates before they are sent to the server (Zhu et al., 2019a), and (2) threshold-based pruning, which removes gradient components whose magnitudes fall below a fixed threshold (Lin et al., 2018). We evaluate both defenses using the unlearning algorithm of Alam et al. (2024) on CIFAR10. As shown in Figure 5(b), increasing the noise level or pruning threshold σ from 10^{-5} to 10^{-3} substantially degrades the quality of the reconstructed images, indicating that both defenses can hinder *DRAUN*.

However, this improvement in privacy comes at a clear cost in model utility. To quantify this trade-off, we evaluate pruning and Gaussian noise on CIFAR-10 under Alam et al. (2024) unlearning method, using 90 training rounds followed by 10 unlearning rounds. As shown in Figure 6, both defenses cause a noticeable drop in model accuracy on both retained and unlearned data, while only moderately increasing reconstruction difficulty. This suggests that, although these defenses reduce leakage, their utility loss makes them unattrac-

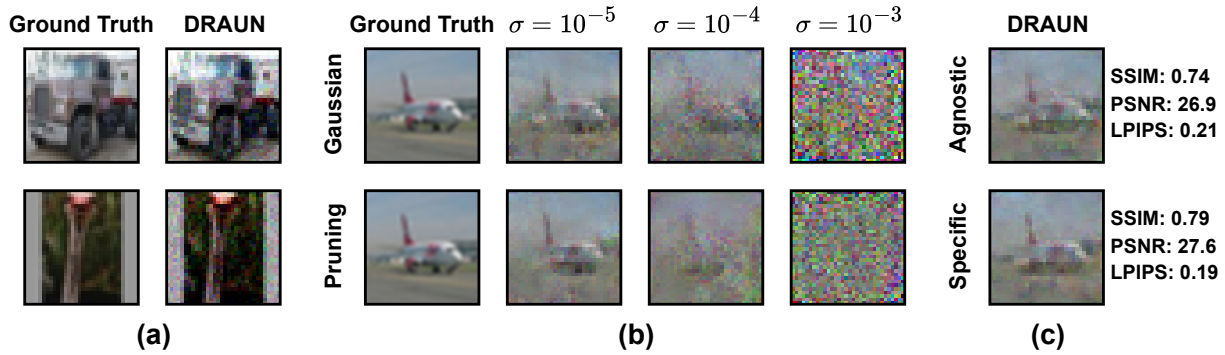

Figure 5: **(a)** Examples of CIFAR10 images reconstructed by *DRAUN* from second-order unlearning updates (Jin et al., 2024). **(b)** Effects of two defense strategies: Gaussian noise addition (top row) and gradient pruning (bottom row) at varying noise levels and pruning thresholds σ. **(c)** Comparison of *DRAUN*'s reconstruction performance under optimization-agnostic and optimization-specific settings.

tive for practical federated unlearning settings where maintaining accuracy is essential.

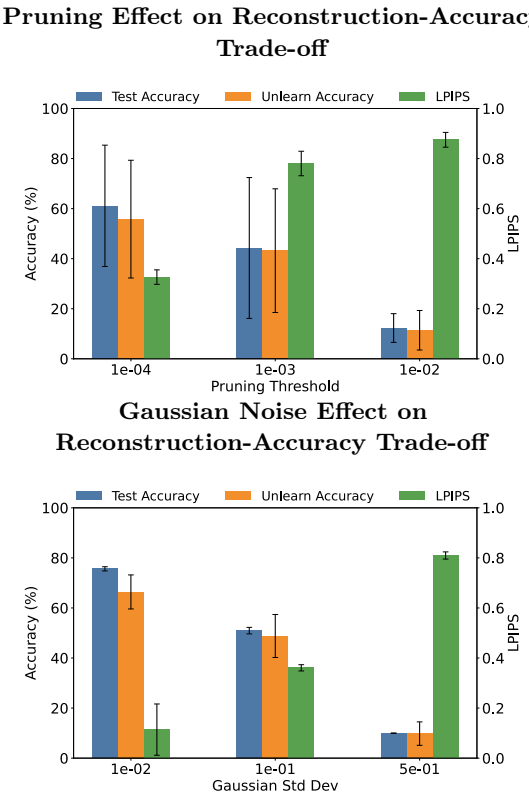

Figure 6: *DRAUN* reconstruction-accuracy trade-off under pruning and Gaussian noise defenses.

***DRAUN*'s Performance in Optimization-Specific Setting:** We also consider a relaxed threat model where the server knows the client's unlearning algorithm, referred to as the **optimization-specific** setting. Here, the server simulates the exact unlearning updates, removing the need for surrogate optimization. Figure 5(c) shows *DRAUN*'s reconstruction performance on CIFAR10 in both agnostic and specific modes using the unlearning algorithm proposed by Alam et al. (2024). The optimization-specific mode yields significantly better reconstructions of the unlearned image, closely resembling the ground truth. This demonstrates that knowing the exact unlearning dynamics enables the attacker to construct more accurate gradient surrogates, leading to faster convergence and improved reconstruction quality. A detailed comparison between optimization-specific and optimization-agnostic modes of *DRAUN* is provided in Appendix E.5.

Limitations and Future Work: (1) *DRAUN* is specifically designed for optimization-based federated unlearning algorithms. However, there also exist non-optimization-based unlearning techniques (Liu et al., 2025) that fall outside the current scope of our attack. In future work, we plan to extend *DRAUN* to support these alternative classes of unlearning algorithms. **(2)** *DRAUN* relies on GIA as its backbone and therefore inherits its limitations, including reduced effectiveness on batch data reconstruction. We plan to develop a GIA-Free *DRAUN* to address these issues.

6 CONCLUSION

We present *DRAUN*, the first DRA specifically designed for FU systems. Existing DRAs for unlearning were developed in centralized MLaaS settings and do not directly translate to federated learning. *DRAUN* addresses the unique challenges of FU and is capable of reconstructing deleted data from local unlearning updates without requiring knowledge of the client's unlearning method. Our theoretical and empirical analysis shows that optimization-based unlearning methods leak exploitable signals, which *DRAUN* effectively uses to recover the deleted data. We demonstrate *DRAUN*'s effectiveness across a range of datasets, model architectures, and unlearning algorithms.

References

Alam, M., Lamri, H., and Maniatakos, M. (2024). Get rid of your trail: Remotely erasing backdoors in federated learning. *IEEE Trans. Artif. Intell.*, 5(12):6683–6698.

Balle, B., Cherubin, G., and Hayes, J. (2022). Reconstructing training data with informed adversaries. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 1138–1156. IEEE.

Balunovic, M., Dimitrov, D. I., Staab, R., and Vechev, M. T. (2022). Bayesian framework for gradient leakage. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Bertran, M., Tang, S., Kearns, M., Morgenstern, J. H., Roth, A., and Wu, S. Z. (2024). Reconstruction attacks on machine unlearning: Simple models are vulnerable. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Boenisch, F., Dziedzic, A., Schuster, R., Shamsabadi, A. S., Shumailov, I., and Papernot, N. (2023). When the curious abandon honesty: Federated learning is not private. In *8th IEEE European Symposium on Security and Privacy, EuroS&P 2023, Delft, Netherlands, July 3-7, 2023*, pages 175–199. IEEE.

Caldas, S., Wu, P., Li, T., Konecny, J., McMahan, H. B., Smith, V., and Talwalkar, A. (2018). Leaf: A benchmark for federated settings. In *Workshop on Federated Learning for Data Privacy and Confidentiality, NeurIPS*.

Cao, Y. and Yang, J. (2015). Towards Making Systems Forget with Machine Unlearning. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 463–480. IEEE Computer Society.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255.

Dimitrov, D. I., Baader, M., Müller, M. N., and Vechev, M. T. (2024). SPEAR: exact gradient inversion of batches in federated learning. *CoRR*, abs/2403.03945.

Dimitrov, D. I., Balunovic, M., Konstantinov, N., and Vechev, M. T. (2022). Data leakage in federated averaging. *Trans. Mach. Learn. Res.*, 2022.

European Commission (2024). Data protection in the EU. `https://commission.europa.eu/law/law-topic/data-protection/data-protection-eu_en`.

Fowl, L. H., Geiping, J., Czaja, W., Goldblum, M., and Goldstein, T. (2022). Robbing the fed: Directly obtaining private data in federated learning with modified models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Gao, W., Guo, S., Zhang, T., Qiu, H., Wen, Y., and Liu, Y. (2021). Privacy-preserving collaborative learning with automatic transformation search. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 114–123. Computer Vision Foundation / IEEE.

Garov, K., Dimitrov, D. I., Jovanovic, N., and Vechev, M. T. (2024). Hiding in plain sight: Disguising data stealing attacks in federated learning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. (2020a). Inverting gradients - how easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. (2020b). Inverting gradients - how easy is it to break privacy in federated learning? In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Geng, J., Mou, Y., Li, F., Li, Q., Beyan, O., Decker, S., and Rong, C. (2021). Towards general deep leakage in federated learning. *CoRR*, abs/2110.09074.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. (2014). Generative adversarial networks. *CoRR*, abs/1406.2661.

Haim, N., Vardi, G., Yehudai, G., Shamir, O., and Irani, M. (2022). Reconstructing training data from trained neural networks. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing*

Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022.

Halimi, A., Kadhe, S., Rawat, A., and Baracaldo, N. (2022). Federated unlearning: How to efficiently erase a client in fl? In *Updatable Machine Learning (part of ICML 2022), UpML 2022, Baltimore, USA, July 23, 2022*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

He, Z., Zhang, T., and Lee, R. B. (2019). Model inversion attacks against collaborative inference. In Balenson, D. M., editor, *Proceedings of the 35th Annual Computer Security Applications Conference, ACSAC 2019, San Juan, PR, USA, December 09-13, 2019*, pages 148–162. ACM.

Hitaj, B., Ateniese, G., and Pérez-Cruz, F. (2017). Deep models under the GAN: information leakage from collaborative deep learning. In Thuraisingham, B., Evans, D., Malkin, T., and Xu, D., editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 603–618. ACM.

Hu, H., Wang, S., Dong, T., and Xue, M. (2024). Learn what you want to unlearn: Unlearning inversion attacks against machine unlearning. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*, pages 3257–3275. IEEE.

Huang, Y., Gupta, S., Song, Z., Li, K., and Arora, S. (2021). Evaluating gradient inversion attacks and defenses in federated learning. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 7232–7241.

Jacot, A., Hongler, C., and Gabriel, F. (2018). Neural tangent kernel: Convergence and generalization in neural networks. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8580–8589.

Jeon, J., Kim, J., Lee, K., Oh, S., and Ok, J. (2021). Gradient inversion with generative image prior. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 29898–29908.

Ji, Z. and Telgarsky, M. (2020). Directional convergence and alignment in deep learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 17176–17186. Curran Associates, Inc.

Jin, R., Chen, M., Zhang, Q., and Li, X. (2024). Forgettable federated linear learning with certified data unlearning.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical report, University of Toronto.

Kuhn, H. W. and Tucker, A. W. (1951). Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1950*, pages 481–492, Berkeley and Los Angeles. University of California Press.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324.

Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B., and Ma, X. (2021). Anti-backdoor learning: Training clean models on poisoned data. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 14900–14912.

Lin, Y., Han, S., Mao, H., Wang, Y., and Dally, B. (2018). Deep gradient compression: Reducing the communication bandwidth for distributed training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

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

Liu, Z., Jiang, Y., Shen, J., Peng, M., Lam, K., Yuan, X., and Liu, X. (2025). A survey on federated unlearning: Challenges, methods, and future directions. *ACM Comput. Surv.*, 57(1):2:1–2:38.

Lyu, K. and Li, J. (2020). Gradient descent maximizes the margin of homogeneous neural networks. In *International Conference on Learning Representations*.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR.

Noorbakhsh, S. L., Zhang, B., Hong, Y., and Wang, B. (2024). Inf2guard: An information-theoretic framework for learning privacy-preserving representations against inference attacks. In Balzarotti, D. and Xu, W., editors, *33rd USENIX Security Symposium, USENIX Security 2024, Philadelphia, PA, USA, August 14-16, 2024*. USENIX Association.

Office of the Attorney General, State of California (2024). California Consumer Privacy Act (CCPA). `https://oag.ca.gov/privacy/ccpa`.

Rudin, L. I., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268.

Salem, A., Bhattacharya, A., Backes, M., Fritz, M., and Zhang, Y. (2020). Updates-leak: Data set inference and reconstruction attacks in online learning. In Capkun, S. and Roesner, F., editors, *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pages 1291–1308. USENIX Association.

Scheliga, D., Mäder, P., and Seeland, M. (2022). PRECODE - A generic model extension to prevent deep gradient leakage. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022, Waikoloa, HI, USA, January 3-8, 2022*, pages 3605–3614. IEEE.

Shaik, T., Tao, X., Li, L., Xie, H., Cai, T., Zhu, X., and Li, Q. (2024). FRAMU: attention-based machine unlearning using federated reinforcement learning. *IEEE Trans. Knowl. Data Eng.*, 36(10):5153–5167.

Sun, J., Li, A., Wang, B., Yang, H., Li, H., and Chen, Y. (2021). Soteria: Provable defense against privacy leakage in federated learning from representation perspective. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 9311–9319. Computer Vision Foundation / IEEE.

Wang, W., Tian, Z., Zhang, C., Liu, A., and Yu, S. (2023a). BFU: bayesian federated unlearning with parameter self-sharing. In *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security, ASIA CCS 2023, Melbourne, VIC, Australia, July 10-14, 2023*, pages 567–578. ACM.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612.

Wang, Z., Lee, J., and Lei, Q. (2023b). Reconstructing training data from model gradient, provably. In Ruiz, F. J. R., Dy, J. G., and van de Meent, J., editors, *International Conference on Artificial Intelligence and Statistics, 25-27 April 2023, Palau de Congressos, Valencia, Spain*, volume 206 of *Proceedings of Machine Learning Research*, pages 6595–6612. PMLR.

Wang, Z., Song, M., Zhang, Z., Song, Y., Wang, Q., and Qi, H. (2019). Beyond inferring class representatives: User-level privacy leakage from federated learning. In *2019 IEEE Conference on Computer Communications, INFOCOM 2019, Paris, France, April 29 - May 2, 2019*, pages 2512–2520. IEEE.

Wen, Y., Geiping, J., Fowl, L., Goldblum, M., and Goldstein, T. (2022a). Fishing for user data in large-batch federated learning via gradient magnification. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 23668–23684. PMLR.

Wen, Y., Geiping, J. A., Fowl, L., Goldblum, M., and Goldstein, T. (2022b). Fishing for user data in large-batch federated learning via gradient magnification. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23668–23684. PMLR.

Wu, L., Guo, S., Wang, J., Hong, Z., Zhang, J., and Ding, Y. (2022). Federated unlearning: Guarantee the right of clients to forget. *IEEE Netw.*, 36(5):129–135.

Yin, H., Mallya, A., Vahdat, A., Álvarez, J. M., Kautz, J., and Molchanov, P. (2021a). See through gradients: Image batch recovery via gradinversion. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 16337–16346. Computer Vision Foundation / IEEE.

Yin, H., Mallya, A., Vahdat, A., Álvarez, J. M., Kautz, J., and Molchanov, P. (2021b). See through gradients: Image batch recovery via gradinversion. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 16337–16346. Computer Vision Foundation / IEEE.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA,*

June 18-22, 2018, pages 586–595. Computer Vision Foundation / IEEE Computer Society.

Zhang, S., Huang, J., Zhang, Z., and Qi, C. (2022). Compromise privacy in large-batch federated learning via malicious model parameters. In Meng, W., Lu, R., Min, G., and Vaidya, J., editors, *Algorithms and Architectures for Parallel Processing - 22nd International Conference, ICA3PP 2022, Copenhagen, Denmark, October 10-12, 2022, Proceedings*, volume 13777 of *Lecture Notes in Computer Science*, pages 63–80. Springer.

Zhao, B., Mopuri, K. R., and Bilen, H. (2020). idlg: Improved deep leakage from gradients. *CoRR*, abs/2001.02610.

Zhao, J. C., Sharma, A., Elkordy, A. R., Ezzeldin, Y. H., Avestimehr, S., and Bagchi, S. (2023). Secure aggregation in federated learning is not private: Leaking user data at large scale through model modification. *CoRR*, abs/2303.12233.

Zhao, J. C., Sharma, A., Elkordy, A. R., Ezzeldin, Y. H., Avestimehr, S., and Bagchi, S. (2024). Loki: Large-scale data reconstruction attack against federated learning through model manipulation. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*, pages 1287–1305. IEEE.

Zhou, X., Fan, S., and Jaggi, M. (2025). HyperINF: Unleashing the hyperpower of the schulz's method for data influence estimation.

Zhu, J. and Blaschko, M. B. (2021a). R-GAP: recursive gradient attack on privacy. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Zhu, J. and Blaschko, M. B. (2021b). R-GAP: recursive gradient attack on privacy. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Zhu, L., Liu, Z., and Han, S. (2019a). Deep leakage from gradients. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14747–14756.

Zhu, L., Liu, Z., and Han, S. (2019b). Deep leakage from gradients. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14747–14756.

Checklist

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

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

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [No]

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

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots. [Not Applicable]

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Supplementary Materials

A Background

A.1 Federated Unlearning

Federated Unlearning (FU) extends the concept of Machine Unlearning (MU) (Cao and Yang, 2015) to federated learning (FL) settings. MU is the process of reversing the influence of specific data samples—often referred to as the unlearning dataset—on a trained model, enabling it to "forget" such data while maintaining performance on the remaining (retained) data. This is increasingly important in light of privacy regulations and the need to mitigate the effect of corrupted or poisoned data. A straightforward but computationally intensive solution is to retrain the model from scratch without the unlearning dataset, known as exact unlearning. To reduce the overhead, approximate unlearning methods have been proposed to yield models that are statistically close to those obtained via exact retraining. First-order (gradient-based) methods are among the most widely used in FL (Halimi et al., 2022; Alam et al., 2024; Wu et al., 2022; Li et al., 2021). These methods typically perform gradient ascent on the unlearning dataset, or optimize a loss difference between retained and unlearned data, often combined with regularization terms that constrain the local model to remain close to the global model. The decentralized nature of FL introduces additional complexities to unlearning, leading to the exploration of different scenarios. For example, second-order methods based on Hessian inversion may not be directly suitable for FL due to computational and communication constraints. Nevertheless, (Jin et al., 2024) have applied such techniques in the Neural Tangent Kernel (NTK)(Jacot et al., 2018) regime of trained neural networks. The unlearning process in FL can also vary in terms of scope and participation: some settings require client participation, while others aim to unlearn data from entire clients (Liu et al., 2021; Jin et al., 2024). In more fine-grained scenarios, clients may wish to selectively forget specific samples while continuing to participate in the protocol.While optimization-based methods are widely used, other techniques such as knowledge distillation and null subspace projection have also been explored Liu et al. (2025), contributing to the diverse landscape of federated unlearning approaches.

A.2 Data Reconstruction Attack

Extensive research has revealed the vulnerability of machine learning models, particularly deep neural networks (DNNs) (Goodfellow et al., 2016), to data reconstruction attacks (DRAs), also known as model inversion attacks (MIAs). Studies such as (Zhu et al., 2019b; Zhao et al., 2020; Geiping et al., 2020b; Yin et al., 2021b; Wen et al., 2022b; Geng et al., 2021; Zhu and Blaschko, 2021b) have demonstrated the capability of a malicious server to reconstruct a client's private training data from the gradients of their updates. This is often achieved through gradient matching, which involves optimizing the discrepancy between the true gradient derived from training on actual data and a synthesized (dummy) gradient. Concurrently, (Boenisch et al., 2023; Fowl et al., 2022; Zhao et al., 2023; Zhang et al., 2022) have explored methods that utilize modified models to directly extract training datasets, bypassing the need for optimization. Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) have also been leveraged in (Salem et al., 2020; Hitaj et al., 2017; Jeon et al., 2021; Wang et al., 2019) to reconstruct training data by conditioning the generator on prior knowledge about the dataset. Furthermore, Balle et al. (2022); He et al. (2019) presented approaches that train a DNN to learn the inverse mapping from labels to their corresponding inputs. Beyond optimization and generative methods, some DRAs are grounded in theoretical findings. Haim et al. (2022) employed the implicit bias approximation results (Lyu and Li, 2020; Ji and Telgarsky, 2020) to extract training data from the algebraic equations of the Karush-Kuhn-Tucker (KKT) conditions of trained DNNs (Kuhn and Tucker, 1951). Additionally, (Balunovic et al., 2022) constructed a Bayes optimal data reconstructor, and (Wang et al., 2023b) utilized tensor theory and the third moment of the dataset for reconstruction. Dimitrov et al. (2024) exploited the low rankness and ReLU-induced sparsity of gradients within a sampling-based algorithm to reconstruct data. Despite the significant body of work on DRAs, Huang et al. (2021) highlighted that many existing attacks rely on often unrealistic assumptions, such as access to batch normalization statistics, knowledge of prior data distributions, and known labels. On the defensive front, various mitigation and

detection strategies have been proposed, ranging from information-theoretic frameworks to gradient obfuscation and model modification (Gao et al., 2021; Sun et al., 2021; Scheliga et al., 2022; Noorbakhsh et al., 2024; Garov et al., 2024).

B Gradient Inversion Attack

Gradient Inversion Attack (GIA) was first introduced by Zhu et al. (2019b), where they demonstrated that training data can be reconstructed from model gradients in the FedSGD protocol by optimizing a similarity loss function $\mathcal{L}_{\text{sim}}$ (typically the ℓ_2 norm) between the true gradient and a dummy gradient computed on dummy inputs and labels, as shown in Algorithm 4.Geiping et al. (2020b) chose $\mathcal{L}_{\text{sim}}$ to be the cosine similarity and added a Total Variation norm as regularization to enhance the quality of the reconstructed input. They also targeted the FedAvg protocol, where clients are allowed to perform multiple local training steps.Yin et al. (2021b) introduced group regularization to improve batch reconstruction, and further showed that labels can be retrieved analytically from the model's linear layer instead of being treated as optimization variables.

Algorithm 4 Gradient Inversion Attack

Input: $\nabla\theta_c := \nabla_\theta \mathcal{L}_\theta(x, y)$ ▷ Client's gradient of the cross-entropy loss
$\tilde{x} \leftarrow \mathcal{N}(0,1)$, $\tilde{y} \leftarrow \mathcal{N}(0,1)$ ▷ Initialize dummy inputs and labels
for t in [1..T] **do**
  $\nabla\tilde{\theta}_c \leftarrow \nabla_\theta \mathcal{L}_\theta(\tilde{x}, \tilde{y})$ ▷ Compute dummy gradient
  $\ell \leftarrow \mathcal{L}_{\text{sim}}(\nabla\theta_c, \nabla\tilde{\theta}_c) + \lambda_{\text{TV}} \cdot TV(\tilde{x})$
  $\tilde{x} \leftarrow \tilde{x} - \eta_{\text{rec}} \cdot \frac{\partial \ell}{\partial \tilde{x}}$
  $\tilde{y} \leftarrow \tilde{y} - \eta_{\text{rec}} \cdot \frac{\partial \ell}{\partial \tilde{y}}$
end for
return $(\tilde{x}, \tilde{y})$

C Detailed Theoretical Analysis

C.1 Failure of Classical GIA in First-Order Federated Unlearning Algorithms

We recall that in Section 3, we adopted the following assumptions :

Assumption 1. *Without loss of generality, we assume the similarity loss $\mathcal{L}_{sim}$ from Equation 1 is the squared ℓ_2-norm:*

$$\mathcal{L}_{sim}(\nabla_\theta \mathcal{L}_u, \nabla_\theta \tilde{\mathcal{L}}_u) = \left\| \nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u \right\|_2^2.$$

Thus, the classical GIA objective becomes:

$$x^* = \arg\min_{x \in \mathcal{X}} \left\| \nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u \right\|_2^2.$$

Assumption 2. *The surrogate loss $\tilde{\mathcal{L}}_u$ is twice differentiable in x and θ. The Jacobian*

$$J(x) = \frac{\partial \nabla_\theta \tilde{\mathcal{L}}_u}{\partial x} \in \mathbb{R}^{d_\theta \times d_x},$$

where d_x and d_θ are the dimensions of x and θ, exists everywhere.

Assumption 3. *We assume $d_\theta \geq d_x$ and that $J(x)$ has full column rank (i.e., rank$(J(x)) = d_x$) $\forall x$.*

Assumption 4. *The surrogate loss $\tilde{\mathcal{L}}_u$ is: μ_x-smooth in x: $\left\| \nabla_x \tilde{\mathcal{L}}_u(x_1) - \nabla_x \tilde{\mathcal{L}}_u(x_2) \right\|_2 \leq \mu_x \|x_1 - x_2\|_2$, and μ_θ-smooth in θ: $\left\| \nabla_\theta \tilde{\mathcal{L}}_u(\theta_1) - \nabla_\theta \tilde{\mathcal{L}}_u(\theta_2) \right\|_2 \leq \mu_\theta \|\theta_1 - \theta_2\|_2$.*

Theoretical justification of Assumption 1. We assumed, *without loss of generality*, that the distance function is the squared ℓ_2-norm to simplify the derivations in our theoretical analysis. It is important to note that the minimizers of the squared ℓ_2 distance are a subset of those of the cosine distance. Specifically, for a fixed

nonzero vector $y \in \mathbb{R}^d$, the squared ℓ_2 distance $\|x-y\|^2$ is uniquely minimized at $x=y$, while the cosine distance $1-\frac{\langle x,y\rangle}{\|x\|\cdot\|y\|}$ is minimized by all positive scalar multiples of y, i.e., all $x=\alpha y$ for $\alpha>0$. Therefore, we have:

$$\arg\min_x \|x-y\|^2 = \{y\} \subset \arg\min_x \left(1-\frac{\langle x,y\rangle}{\|x\|\cdot\|y\|}\right) = \{\alpha y \mid \alpha>0\}.$$

Thus, any solution that minimizes the squared ℓ_2 distance also minimizes the cosine distance, but the converse is not necessarily true. This shows that cosine distance preserves the same *argmin* as the squared ℓ_2-norm, so it does not affect the minimization objective. Meanwhile, the TV-norm term only shifts the minimum value but does not alter the structure of the loss landscape. Therefore, the analysis presented in **Figure 3** remains valid.

C.2 Proof of Theorem 1

Proof. We show that Fermat's stationary condition does not hold at $x=x_u$.

First, we compute the gradient of $\mathcal{L}_{\text{sim}}$ with respect to x:

$$\frac{\partial \mathcal{L}_{\text{sim}}}{\partial x} = \frac{\partial}{\partial x}\left\|\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u\right\|_2^2 = -2J(x)^T\left(\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u\right),$$

where $J(x)=\frac{\partial \nabla_\theta \tilde{\mathcal{L}}_u}{\partial x}$. Substituting $\mathcal{L}_u=\mathcal{L}(\theta,x_r,y_r)-\mathcal{L}(\theta,x_u,y_u)$ and $\tilde{\mathcal{L}}_u=\mathcal{L}(\theta,x,y_u)$:

$$\frac{\partial \mathcal{L}_{\text{sim}}}{\partial x} = -2J(x)^T\left(\nabla_\theta \mathcal{L}(\theta,x_r,y_r)-\nabla_\theta \mathcal{L}(\theta,x_u,y_u)-\nabla_\theta \mathcal{L}(\theta,x,y_u)\right).$$

Then, we evaluate the expression at $x=x_u$:

$$\left.\frac{\partial \mathcal{L}_{\text{sim}}}{\partial x}\right|_{x=x_u} = -2J(x_u)^T\left(\nabla_\theta \mathcal{L}(\theta,x_r,y_r)-2\nabla_\theta \mathcal{L}(\theta,x_u,y_u)\right).$$

By Assumption 3, $J(x_u)$ has full column rank. Since $\nabla_\theta \mathcal{L}(\theta,x_r,y_r) \neq 2\nabla_\theta \mathcal{L}(\theta,x_u,y_u)$ as $x_u \neq x_r$, the gradient at x_u is non-zero. By Fermat's optimality condition, x_u cannot be a local minimizer. □

C.3 Proof of Theorem 2

Proof. Since $\tilde{\mathcal{L}}_u$ is twice differentiable and μ_x-smooth in x, we apply a first-order Taylor expansion around x_u:

$$\nabla_\theta \tilde{\mathcal{L}}_u(x_u^*) = \nabla_\theta \tilde{\mathcal{L}}_u(x_u) + J(x_u)(x_u^*-x_u) + R,$$

where $\|R\|_2 \leq \frac{\mu_x}{2}\|x_u^*-x_u\|_2^2$.

Using the identities:

$$\nabla_\theta \tilde{\mathcal{L}}_u(x_u) = \nabla_\theta \mathcal{L}(\theta,x_u,y_u), \quad \nabla_\theta \mathcal{L}_u = \nabla_\theta \mathcal{L}(\theta,x_r,y_r) - \nabla_\theta \mathcal{L}(\theta,x_u,y_u),$$

we have:

$$\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(x_u) = \nabla_\theta \mathcal{L}(\theta,x_r,y_r),$$

and therefore:

$$\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(x_u^*) = \nabla_\theta \mathcal{L}(\theta,x_r,y_r) - J(x_u)(x_u^*-x_u) - R.$$

Taking norms and applying the triangle inequality:

$$\left\|\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(x_u^*)\right\|_2 \geq \|\nabla_\theta \mathcal{L}(\theta,x_r,y_r)\|_2 - \|J(x_u)\|_F\|x_u^*-x_u\|_2 - \frac{\mu_x}{2}\|x_u^*-x_u\|_2^2.$$

To eliminate the quadratic term, we conservatively bound:

$$\frac{\mu_x}{2}\|x_u^*-x_u\|_2^2 \leq \mu_x\|J(x_u)\|_F\|x_u^*-x_u\|_2 + 2\mu_\theta\|\nabla_\theta \mathcal{L}_u\|_2.$$

Substituting, we obtain:

$$\left\|\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(x_u^*)\right\|_2 \geq \|\nabla_\theta \mathcal{L}(\theta, x_r, y_r)\|_2 - (\|J(x_u)\|_F + \mu_x \|J(x_u)\|_F)\|x_u^* - x_u\|_2 - 2\mu_\theta \|\nabla_\theta \mathcal{L}_u\|_2.$$

Combining the Jacobian terms:

$$\left\|\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(x_u^*)\right\|_2 \geq \|\nabla_\theta \mathcal{L}(\theta, x_r, y_r)\|_2 - \mu_x \|J(x_u)\|_F \|x_u^* - x_u\|_2 - 2\mu_\theta \|\nabla_\theta \mathcal{L}_u\|_2.$$

Projecting onto $J(x_u)^\top$ and rearranging:

$$\|x_u^* - x_u\|_2 \geq \frac{\|J(x_u)^\top \nabla_\theta \mathcal{L}(\theta, x_r, y_r)\|_2}{\mu_x \|J(x_u)\|_F + 2\mu_\theta \|\nabla_\theta \mathcal{L}_u\|_2}.$$

□

C.4 Failure of Classical GIA in Second-Order Federated Unlearning Algorithms

Theorem 3 (Non-Optimality of x_u in Second-Order GIA)**.** *Under Assumptions 1–3, suppose the client's unlearning loss $\mathcal{L}_u$ is defined via a Newton update:*

$$\nabla_\theta \mathcal{L}_u = H_r^{-1} \nabla_\theta \mathcal{L}(\theta, x_u, y_u),$$

and the server uses a first-order surrogate loss:

$$\nabla_\theta \tilde{\mathcal{L}}_u(x) = \nabla_\theta \mathcal{L}(\theta, x, y_u).$$

Then the ground truth $x = x_u$ is not a local minimizer of $\mathcal{L}_{sim} = \|\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(x)\|_2^2$.

Proof. The gradient of $\mathcal{L}_{\text{sim}}$ with respect to x is:

$$\nabla_x \mathcal{L}_{\text{sim}} = -2J(x)^\top \left(\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(x)\right),$$

where $J(x) = \frac{\partial \nabla_\theta \tilde{\mathcal{L}}_u}{\partial x}$ is the Jacobian. At $x = x_u$, we substitute:

$$\nabla_\theta \tilde{\mathcal{L}}_u(x_u) = \nabla_\theta \mathcal{L}(\theta, x_u, y_u), \quad \nabla_\theta \mathcal{L}_u = H_r^{-1} \nabla_\theta \mathcal{L}(\theta, x_u, y_u).$$

This gives:

$$\nabla_x \mathcal{L}_{\text{sim}}\big|_{x=x_u} = -2J(x_u)^\top \left(H_r^{-1} - I\right) \nabla_\theta \mathcal{L}(\theta, x_u, y_u).$$

By Assumption 3, $J(x_u)$ has full column rank. If $H_r^{-1} \neq I$ and $\nabla_\theta \mathcal{L}(\theta, x_u, y_u) \neq 0$, the gradient is non-zero. By Fermat's theorem, x_u cannot be a local minimizer. □

Assumption 5. *The Hessian H_r is μ_H-Lipschitz in the input, i.e.,*

$$\|H_r(x_1) - H_r(x_2)\| \leq \mu_H \|x_1 - x_2\| \quad \forall x_1, x_2.$$

Assumption 6. *The inverse Hessian is bounded: $\|H_r^{-1}\| \leq \kappa$.*

Theorem 4 (Second-Order Reconstruction Error Bound)**.** *Let x_u^* be a minimizer of $\mathcal{L}_{sim}$ under Assumptions 1–6. Then:*

$$\|x_u^* - x_u\|_2 \geq \frac{\left\|J^\top H_r^{-1} \nabla_\theta \mathcal{L}(\theta, x_r, y_r)\right\|_2}{\mu_x \|J\|_F + \mu_H \kappa \|\nabla_\theta \mathcal{L}_u\|_2}.$$

Proof. Define $\Delta x = x_u^* - x_u$. Expand $\nabla_\theta \tilde{\mathcal{L}}_u(x_u^*)$ via Taylor series:

$$\nabla_\theta \tilde{\mathcal{L}}_u(x_u^*) = \nabla_\theta \mathcal{L}(\theta, x_u, y_u) + J(x_u)\Delta x + R,$$

where $\|R\|_2 \leq \frac{\mu_x}{2}\|\Delta x\|_2^2$. Substituting into $\mathcal{L}_{\text{sim}}$:

$$\mathcal{L}_{\text{sim}} = \left\|H_r^{-1}\nabla_\theta \mathcal{L}(\theta, x_u, y_u) - (\nabla_\theta \mathcal{L}(\theta, x_u, y_u) + J(x_u)\Delta x + R)\right\|_2^2.$$

Simplify using $\nabla_\theta \mathcal{L}_u = H_r^{-1}\nabla_\theta \mathcal{L}(\theta, x_u, y_u)$:

$$\mathcal{L}_{\text{sim}} = \left\|\left(H_r^{-1} - I\right)\nabla_\theta \mathcal{L}(\theta, x_u, y_u) - J(x_u)\Delta x - R\right\|_2^2.$$

Using $\|H_r^{-1}\| \leq \kappa$ (Assumption 6) and the Hessian Lipschitz property (Assumption 5):

$$\|R\|_2 \leq \frac{\mu_x}{2}\|\Delta x\|_2^2, \quad \|H_r^{-1} - I\| \leq \kappa + 1.$$

Isolate $\|\Delta x\|_2$ via Cauchy-Schwarz and linearize:

$$\left\|J^\top H_r^{-1}\nabla_\theta \mathcal{L}(\theta, x_r, y_r)\right\|_2 \leq (\mu_x\|J\|_F + \mu_H\kappa\|\nabla_\theta \mathcal{L}_u\|_2)\,\|\Delta x\|_2.$$

Rearranging yields the stated bound. □

C.5 Correctness of DRAUN for First- and Second-Order Algorithms

As shown previously, classical GIA fails to reconstruct the ground truth unlearn input x_u because the server's surrogate loss $\tilde{\mathcal{L}}_u$ depends only on the dummy unlearn input $\tilde{x}_u$. This mismatch leads to a biased local minimum of the similarity loss $\mathcal{L}_{\text{sim}}$, especially when the client's unlearning loss $\mathcal{L}_u$ implicitly depends on both x_u and x_r.

To address this, DRAUN incorporates the retain input x_r into the GIA optimization problem. The surrogate loss $\tilde{\mathcal{L}}_u$ is then defined over both dummy inputs $(\tilde{x}_u, \tilde{x}_r)$, making $\mathcal{L}_{\text{sim}}$ a function of two variables. Theorems 5 and 6 formalize this intuition by showing that the pair (x_u, x_r) is a local minimizer of the similarity loss under both first-order and second-order formulations.

Theorem 5 (First-Order DRAUN Correctness)**.** *Under Assumptions 1–3, let the client's unlearning loss be:*

$$\nabla_\theta \mathcal{L}_u = \nabla_\theta \mathcal{L}(\theta, x_r, y_r) - \nabla_\theta \mathcal{L}(\theta, x_u, y_u),$$

and the server's surrogate loss be:

$$\nabla_\theta \tilde{\mathcal{L}}_u(\tilde{x}_u, \tilde{x}_r) = \nabla_\theta \mathcal{L}(\theta, \tilde{x}_r, y_r) - \nabla_\theta \mathcal{L}(\theta, \tilde{x}_u, y_u).$$

*Then, the ground truth pair $(\tilde{x}_u, \tilde{x}_r) = (x_u, x_r)$ is a **local minimizer** of:*

$$\mathcal{L}_{sim}(\tilde{x}_u, \tilde{x}_r) = \left\|\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(\tilde{x}_u, \tilde{x}_r)\right\|_2^2.$$

Proof. Compute the gradient of $\mathcal{L}_{\text{sim}}$ at $(\tilde{x}_u, \tilde{x}_r)$:

$$\nabla_{\tilde{x}_u, \tilde{x}_r}\mathcal{L}_{\text{sim}} = -2\begin{bmatrix} J_u^\top \\ -J_r^\top \end{bmatrix}\left(\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(\tilde{x}_u, \tilde{x}_r)\right),$$

where $J_u = \frac{\partial \nabla_\theta \mathcal{L}(\theta, \tilde{x}_u, y_u)}{\partial \tilde{x}_u}$ and $J_r = \frac{\partial \nabla_\theta \mathcal{L}(\theta, \tilde{x}_r, y_r)}{\partial \tilde{x}_r}$.

At $(\tilde{x}_u, \tilde{x}_r) = (x_u, x_r)$, we have:

$$\nabla_\theta \tilde{\mathcal{L}}_u(x_u, x_r) = \nabla_\theta \mathcal{L}_u \implies \nabla_{\tilde{x}_u, \tilde{x}_r}\mathcal{L}_{\text{sim}}\big|_{(x_u, x_r)} = 0.$$

The Hessian at (x_u, x_r) is:

$$\nabla^2_{\tilde{x}_u, \tilde{x}_r}\mathcal{L}_{\text{sim}}\big|_{(x_u, x_r)} = 2\begin{bmatrix} J_u^\top J_u & -J_u^\top J_r \\ -J_r^\top J_u & J_r^\top J_r \end{bmatrix}.$$

By Assumption 3, J_u and J_r have full column rank, so the Hessian is positive definite. Thus, (x_u, x_r) is a local minimizer. □

Theorem 6 (Second-Order DRAUN Correctness)**.** *Under Assumptions 1–6, let the client's unlearning loss be:*

$$\nabla_\theta \mathcal{L}_u = H_r^{-1} \nabla_\theta \mathcal{L}(\theta, x_u, y_u),$$

and the server's surrogate loss be:

$$\nabla_\theta \tilde{\mathcal{L}}_u(\tilde{x}_u, \tilde{x}_r) = \tilde{H}_r^{-1} \nabla_\theta \mathcal{L}(\theta, \tilde{x}_u, y_u),$$

where $\tilde{H}_r = \nabla_\theta^2 \mathcal{L}(\theta, \tilde{x}_r, y_r)$*. Then,* $(\tilde{x}_u, \tilde{x}_r) = (x_u, x_r)$ *is a* ***local minimizer*** *of:*

$$\mathcal{L}_{sim}(\tilde{x}_u, \tilde{x}_r) = \left\| \nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u(\tilde{x}_u, \tilde{x}_r) \right\|_2^2.$$

Proof. At $(\tilde{x}_u, \tilde{x}_r) = (x_u, x_r)$, we have $\tilde{H}_r = H_r$, so:

$$\nabla_\theta \tilde{\mathcal{L}}_u(x_u, x_r) = H_r^{-1} \nabla_\theta \mathcal{L}(\theta, x_u, y_u) = \nabla_\theta \mathcal{L}_u.$$

This implies $\mathcal{L}_{\text{sim}}(x_u, x_r) = 0$. To verify (x_u, x_r) is a local minimizer, compute the gradient and Hessian of $\mathcal{L}_{\text{sim}}$. The gradient is:

$$\nabla_{\tilde{x}_u, \tilde{x}_r} \mathcal{L}_{\text{sim}} = -2 \begin{bmatrix} J_u^\top H_r^{-1} \\ -J_r^\top H_r^{-1} \end{bmatrix} \left(\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u \right),$$

where $J_u = \frac{\partial \nabla_\theta \mathcal{L}(\theta, x_u, y_u)}{\partial x_u}$ and $J_r = \frac{\partial \nabla_\theta \mathcal{L}(\theta, x_r, y_r)}{\partial x_r}$. At (x_u, x_r), the term $\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u$ vanishes, giving $\nabla_{\tilde{x}_u, \tilde{x}_r} \mathcal{L}_{\text{sim}} \big|_{(x_u, x_r)} = 0$. The Hessian at (x_u, x_r) is:

$$\nabla^2_{\tilde{x}_u, \tilde{x}_r} \mathcal{L}_{\text{sim}} \approx 2 \begin{bmatrix} J_u^\top H_r^{-2} J_u & 0 \\ 0 & J_r^\top H_r^{-2} J_r \end{bmatrix},$$

which is positive definite by Assumption 3 (full column rank J_u, J_r) and Assumption 6 ($H_r \succ 0$). Thus, (x_u, x_r) is a local minimizer. □

C.6 Bounding Reconstruction Error for First- and Second-Order DRAUN

As discussed in Section 2, DRAUN does not aim to reconstruct the true retain samples x_r. Instead, as described in the initialization step of Algorithm 3, the server simply generates dummy retain inputs of the same size as the unlearn dataset. Although $\tilde{x}_r$ is not meant to match x_r, we aim for it to be close. This motivates the analysis of how proximity to x_r impacts the reconstruction of x_u.

In this section, we provide upper bounds on the reconstruction error $\|\tilde{x}_u - x_u\|$ under both first-order and second-order DRAUN formulations, assuming that the client loss depends on both unlearn and retain inputs.

Assumption 7 (Proximity of Optimized Dummy Inputs)**.** *Let* $\tilde{x}_r^{(T)}$ *be the dummy retain input after* T *optimization steps in (Algorithm 1). We assume:*

$$\tilde{x}_r^{(T)} \in B(x_r, \epsilon(T)), \quad \text{where } \epsilon(T) = \mathcal{O}(1/\sqrt{T}),$$

and the gradient $\nabla_\theta \mathcal{L}$ *is* μ_x*-Lipschitz in* x *(per Assumption 4).*

Theorem 7 (First-Order Reconstruction Bound)**.** *Under Assumptions 1-4, 7, the reconstruction error after* T *steps satisfies:*

$$\|\tilde{x}_u^{(T)} - x_u\|_2 \leq \frac{\mu_x \epsilon(T) + \mathcal{L}_{sim}^{1/2}(T)}{\sigma_{\min}(J_u)},$$

where: $J_u = \frac{\partial \nabla_\theta \mathcal{L}(\theta, x_u, y_u)}{\partial x_u}$ *with* $\sigma_{\min}(J_u) > 0$ *(Assumption 3), and* $\mathcal{L}_{sim}(T) = \|\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u^{(T)}\|_2^2$ *is the similarity loss at step* T *where* $\nabla_\theta \tilde{\mathcal{L}}_u^{(T)} = \nabla_\theta \mathcal{L}(\theta, \tilde{x}_r^{(T)}, y_r) - \nabla_\theta \mathcal{L}(\theta, \tilde{x}_u^{(T)}, y_u)$

Proof. Using μ_x-Lipschitz gradients (Assumption 4):

$$\|\nabla_\theta \mathcal{L}(\theta, x_r, y_r) - \nabla_\theta \mathcal{L}(\theta, \tilde{x}_r^{(T)}, y_r)\|_2 \leq \mu_x \epsilon(T).$$

Expand the unlearn term around x_u:

$$\nabla_\theta \mathcal{L}(\theta, \tilde{x}_u^{(T)}, y_u) = \nabla_\theta \mathcal{L}(\theta, x_u, y_u) + J_u(\tilde{x}_u^{(T)} - x_u) + R,$$

with $\|R\|_2 \leq \frac{\mu_x}{2}\|\tilde{x}_u^{(T)} - x_u\|_2^2$. From the similarity objective:

$$\mathcal{L}_{\text{sim}}(T) = \|(\nabla_\theta \mathcal{L}(\theta, x_r, y_r) - \nabla_\theta \mathcal{L}(\theta, \tilde{x}_r^{(T)}, y_r)) - (J_u(\tilde{x}_u^{(T)} - x_u) + R)\|_2^2.$$

Apply triangle inequality and use $\sigma_{\min}(J_u)$-lower bound (Assumption 3):

$$\mathcal{L}_{\text{sim}}^{1/2}(T) \geq \sigma_{\min}(J_u)\|\tilde{x}_u^{(T)} - x_u\|_2 - \mu_x \epsilon(T).$$

Rearranging gives the bound. □

Theorem 8 (Second-Order Reconstruction Error Bound)**.** *Under Assumptions 1-6 and 7, the reconstruction error satisfies:*

$$\|\tilde{x}_u^{(T)} - x_u\|_2 \leq \frac{\kappa(\mu_x \epsilon(T) + \mu_H \epsilon(T)) + \mathcal{L}_{sim}^{1/2}(T)}{\sigma_{\min}(J_u)},$$

where $\mathcal{L}_{sim}(T) = \|\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u^{(T)}\|_2^2$.

Proof. Let $H_r = \nabla_\theta^2 \mathcal{L}(\theta, x_r, y_r)$ and $\tilde{H}_r = \nabla_\theta^2 \mathcal{L}(\theta, \tilde{x}_r^{(T)}, y_r)$. By Assumption 5:

$$\|H_r - \tilde{H}_r\| \leq \mu_H \|x_r - \tilde{x}_r^{(T)}\|_2 \leq \mu_H \epsilon(T).$$

Using the matrix inversion identity $A^{-1} - B^{-1} = A^{-1}(B - A)B^{-1}$ and Assumption 6 ($\|H_r^{-1}\| \leq \kappa$):

$$\|H_r^{-1} - \tilde{H}_r^{-1}\| \leq \|H_r^{-1}\| \cdot \|H_r - \tilde{H}_r\| \cdot \|\tilde{H}_r^{-1}\| \leq \kappa \cdot \mu_H \epsilon(T) \cdot \kappa = \kappa^2 \mu_H \epsilon(T).$$

Decompose the gradient difference:

$$\begin{aligned}\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u^{(T)} = &\underbrace{(H_r^{-1} - \tilde{H}_r^{-1})\nabla_\theta \mathcal{L}(\theta, x_u, y_u)}_{\text{Hessian error}} \\ &+ \tilde{H}_r^{-1} \underbrace{\left(\nabla_\theta \mathcal{L}(\theta, x_u, y_u) - \nabla_\theta \mathcal{L}(\theta, \tilde{x}_u^{(T)}, y_u)\right)}_{\text{Gradient error}}.\end{aligned}$$

For the Hessian error term:

$$\|(H_r^{-1} - \tilde{H}_r^{-1})\nabla_\theta \mathcal{L}(\theta, x_u, y_u)\|_2 \leq \kappa^2 \mu_H \epsilon(T) \cdot \|\nabla_\theta \mathcal{L}(\theta, x_u, y_u)\|_2.$$

By Assumption 4 (μ_θ-smoothness), $\|\nabla_\theta \mathcal{L}(\theta, x_u, y_u)\|_2 \leq C$ for some constant C, giving:

$$\|\text{Hessian error}\|_2 \leq \kappa^2 \mu_H \epsilon(T) C.$$

For the gradient error term, expand via Taylor series with Jacobian $J_u = \frac{\partial \nabla_\theta \mathcal{L}}{\partial x_u}$:

$$\nabla_\theta \mathcal{L}(\theta, \tilde{x}_u^{(T)}, y_u) = \nabla_\theta \mathcal{L}(\theta, x_u, y_u) + J_u(\tilde{x}_u^{(T)} - x_u) + R,$$

where $\|R\|_2 \leq \frac{\mu_x}{2}\|\tilde{x}_u^{(T)} - x_u\|_2^2$ by Assumption 4. Thus:

$$\|\text{Gradient error}\|_2 \leq \kappa \left(\|J_u(\tilde{x}_u^{(T)} - x_u)\|_2 + \frac{\mu_x}{2}\|\tilde{x}_u^{(T)} - x_u\|_2^2\right).$$

Combining terms via triangle inequality:

$$\mathcal{L}_{\text{sim}}^{1/2}(T) \geq \sigma_{\min}(J_u)\|\tilde{x}_u^{(T)} - x_u\|_2 - \kappa\left(\mu_x \epsilon(T) + \mu_H \epsilon(T)\right).$$

Rearranging yields:

$$\|\tilde{x}_u^{(T)} - x_u\|_2 \leq \frac{\kappa(\mu_x \epsilon(T) + \mu_H \epsilon(T)) + \mathcal{L}_{\text{sim}}^{1/2}(T)}{\sigma_{\min}(J_u)}.$$

□

C.7 Impact of Initialization Proximity in DRAUN

Assumption 8 (Bounded Reconstruction Step)**.** *The reconstruction step in Algorithm 1 satisfies $\eta_{rec} \leq \frac{1}{2\mu_x}$, where μ_x is the smoothness constant from Assumption 4.*

Assumption 9 (Proximity at Initialization)**.** *The initial dummy inputs satisfy $\|\tilde{x}_u^{(0)} - \tilde{x}_r^{(0)}\| \leq \Delta$.*

Assumption 10 (Non-Degenerate Gradients)**.** *There exists $c > 0$ such that $\|\nabla_\theta \mathcal{L}(\theta, x_u, y_u)\| \geq c$ for all x_u.*

Theorem 9 (Proximity Preservation over Optimization)**.** *Under Assumptions 4 (μ_x-smoothness of $\mathcal{L}_{sim}$), 8, and 9, after T gradient steps:*

$$\|\tilde{x}_u^{(T)} - \tilde{x}_r^{(T)}\| \leq \Delta \cdot (1 + 2\eta_{rec}\mu_x)^T.$$

Proof. Let $\delta^{(t)} = \|\tilde{x}_u^{(t)} - \tilde{x}_r^{(t)}\|$. By Assumption 4, the similarity loss gradient satisfies:

$$\|\nabla_{\tilde{x}_u} \mathcal{L}_{\text{sim}}\| \leq \mu_x \delta^{(t)}, \quad \|\nabla_{\tilde{x}_r} \mathcal{L}_{\text{sim}}\| \leq \mu_x \delta^{(t)}.$$

Each gradient update step then obeys:

$$\delta^{(t+1)} \leq \delta^{(t)} + \eta_{\text{rec}} \left(\|\nabla_{\tilde{x}_u} \mathcal{L}_{\text{sim}}\| + \|\nabla_{\tilde{x}_r} \mathcal{L}_{\text{sim}}\|\right) \leq \delta^{(t)}(1 + 2\eta_{\text{rec}}\mu_x).$$

Unrolling over T steps yields:

$$\delta^{(T)} \leq \Delta \cdot (1 + 2\eta\mu_x)^T.$$

□

Theorem 10 (First-Order Gradient Collapse)**.** *Under Assumptions 4 (μ_x-smoothness) and $\|J\| \leq C_J$, if $\|\tilde{x}_u - \tilde{x}_r\| \leq \Delta$:*

$$\|\nabla_\theta \tilde{\mathcal{L}}_u\| \leq \mu_x \Delta, \quad \|\nabla_x \mathcal{L}_{sim}\| \leq 2\mu_x C_J \Delta.$$

Proof. By μ_x-Lipschitz continuity (Assumption 4):

$$\|\nabla_\theta \tilde{\mathcal{L}}_u\| = \|\nabla_\theta \mathcal{L}(\theta, \tilde{x}_r, y_r) - \nabla_\theta \mathcal{L}(\theta, \tilde{x}_u, y_u)\| \leq \mu_x \|\tilde{x}_r - \tilde{x}_u\| \leq \mu_x \Delta.$$

The similarity loss gradient becomes:

$$\nabla_x \mathcal{L}_{\text{sim}} = -2J^\top (\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u),$$

with norm bounded by:

$$\|\nabla_x \mathcal{L}_{\text{sim}}\| \leq 2\|J\| \cdot \|\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u\| \leq 2\mu_x C_J \Delta.$$

This implies that as the proximity $\Delta = \|\tilde{x}_u - \tilde{x}_r\|$ approaches zero, the gradient $\nabla_x \mathcal{L}_{\text{sim}}$ vanishes. Consequently, the optimization process may converge prematurely to a false minimum.

□

Theorem 11 (Second-Order Proximity-Inverse Error)**.** *Under Assumptions 5-6, 10, and $\|\tilde{x}_u - \tilde{x}_r\| \leq \Delta$:*

$$\|\tilde{x}_u - x_u\| \geq \frac{\kappa \mu_H c}{\sigma_{\min}(J_u)\Delta + \mu_x},$$

where $\kappa = \|H_r^{-1}\|$, $\sigma_{\min}(J_u) > 0$, and $c > 0$ is from Assumption 10.

Proof. Decompose the gradient mismatch:

$$\nabla_\theta \mathcal{L}_u - \nabla_\theta \tilde{\mathcal{L}}_u = \underbrace{(H_r^{-1} - \tilde{H}_r^{-1})\nabla_\theta \mathcal{L}(\theta, x_u, y_u)}_{\text{Hessian Error}} + \underbrace{\tilde{H}_r^{-1} J_u e}_{\text{Reconstruction Term}} + \mathcal{O}(\mu_x \|e\|^2),$$

where $e = \tilde{x}_u - x_u$.

By Assumption 5 and Neumann series we have:

$$\|H_r^{-1} - \tilde{H}_r^{-1}\| \geq \frac{\mu_H \Delta}{2\kappa} \implies \|\text{Hessian Error}\| \geq \frac{\mu_H \Delta}{2\kappa} \|\nabla_\theta \mathcal{L}\| \geq \frac{\mu_H \Delta c}{2\kappa}.$$

And by Jacobian action:

$$\|\tilde{H}_r^{-1} J_u e\| \leq \frac{\sigma_{\min}(J_u)}{\kappa} \|e\|.$$

From $\mathcal{L}_{\text{sim}} \geq 0$:

$$\frac{\mu_H \Delta c}{2\kappa} \leq \frac{\sigma_{\min}(J_u)}{\kappa} \|e\| + \mu_x \|e\|.$$

Solving for $\|e\|$ gives:

$$\|e\| \geq \frac{\kappa \mu_H c}{2(\sigma_{\min}(J_u) + 2\kappa\mu_x)} \geq \frac{\kappa \mu_H c}{\sigma_{\min}(J_u)\Delta + \mu_x}.$$

□

D Additional Details of Datasets and Models

D.1 Dataset Details

Table 6 lists the datasets used in our experiments. Each dataset was uniformly split across 100 clients, except FEMNIST, which is already partitioned by user.

Table 6: Summary of datasets used in our experiments.

Dataset	Samples	Classes	Input Shape	Description
CIFAR-10	60,000	10	$3 \times 32 \times 32$	Tiny color images of everyday objects
CIFAR-100	60,000	100	$3 \times 32 \times 32$	Similar to CIFAR-10 but with 100 fine-grained classes
MNIST	70,000	10	$1 \times 28 \times 28$	Grayscale handwritten digits (0–9)
FEMNIST	∼800,000	62	$1 \times 28 \times 28$	Extended MNIST with digits and letters, partitioned by user
ImageNet (ILSVRC-2012)	∼1.28M (train), 50k (val)	1,000	$3 \times 224 \times 224$	Large-scale natural images across 1,000 categories (224×224)

CIFAR-10 and CIFAR-100: Both datasets consist of small natural images from 10 or 100 categories, respectively. The images are RGB, and each has a resolution of 32×32 pixels. CIFAR-10 has 6,000 images per class, while CIFAR-100 provides only 600 per class, making it more challenging due to the finer granularity.

MNIST: MNIST contains grayscale images of handwritten digits from 0 to 9. Each image is 28×28 pixels. The dataset is balanced and often used for benchmarking classification models on simple tasks.

FEMNIST: FEMNIST is an extended version of MNIST built for federated learning. It includes digits (0–9), uppercase (A–Z), and lowercase (a–z) letters, for a total of 62 classes. It is partitioned by writer ID, enabling realistic non-IID client splits in federated settings.

ImageNet (ILSVRC-2012): A large-scale benchmark of natural images spanning 1,000 object categories. We use the standard ILSVRC-2012 split (approximately 1.28M training images and 50k validation images), with RGB inputs resized and center-cropped to 224×224.

D.2 MLP and ConvNet Model Architecture

In addition to the ResNet18 (He et al., 2016) and LeNet (LeCun et al., 1998) models, we consider two additional neural network architectures: a Multi-Layer Perceptron (MLP) and a deep convolutional network referred to as ConvNet64. The details of these two models are provided in Table 7 and Table 8, respectively

MLP: The MLP consists of three hidden layers, each with width 1024 and ReLU activation functions. The input is first flattened before being passed through fully connected layers. While the base architecture assumes an input shape of $3 \times 32 \times 32$ (for datasets like CIFAR), we adapt the input layer to match the shape of other datasets such as MNIST and FEMNIST, which have smaller grayscale images.

Table 7: Architecture of the MLP model. The input dimension varies depending on the dataset.

Layer	**Description**
Input	Flatten ($C \times H \times W$)
Linear 1	Linear(input_dim $\to$ 1024)
ReLU 1	ReLU()
Linear 2	Linear(1024 $\to$ 1024)
ReLU 2	ReLU()
Linear 3	Linear(1024 $\to$ 1024)
ReLU 3	ReLU()
Output	Linear(1024 $\to$ num_classes)

ConvNet64: The ConvNet64 architecture is a deep convolutional network composed of 8 convolutional layers, batch normalization, ReLU activations, and two max pooling layers. The model operates on 2D image inputs with `num_channels` input channels and uses a base width parameter.

Table 8: Feature extractor architecture of ConvNet64. Width is denoted by w.

Layer	**Description**
Conv 0	Conv2D($C \to 1w$), 3×3, padding=1
BN + ReLU	BatchNorm + ReLU
Conv 1	Conv2D($1w \to 2w$), 3×3, padding=1
BN + ReLU	BatchNorm + ReLU
Conv 2	Conv2D($2w \to 2w$), 3×3, padding=1
BN + ReLU	BatchNorm + ReLU
Conv 3	Conv2D($2w \to 4w$), 3×3, padding=1
BN + ReLU	BatchNorm + ReLU
Conv 4	Conv2D($4w \to 4w$), 3×3, padding=1
BN + ReLU	BatchNorm + ReLU
Conv 5	Conv2D($4w \to 4w$), 3×3, padding=1
BN + ReLU	BatchNorm + ReLU
MaxPool 0	MaxPool2D(3×3)
Conv 6	Conv2D($4w \to 4w$), 3×3, padding=1
BN + ReLU	BatchNorm + ReLU
Conv 7	Conv2D($4w \to 4w$), 3×3, padding=1
BN + ReLU	BatchNorm + ReLU
MaxPool 1	MaxPool2D(3×3)

E Additional Reconstruction Results

E.1 Reconstruction on Different Model Architectures

We evaluated DRAUN across various datasets and model architectures. Figures 7, 8, and 9 present single-image reconstructions on the FEMNIST dataset using LeNet (LeCun et al., 1998), the MNIST dataset using an MLP, and the CIFAR-100 dataset using ResNet18 (He et al., 2016), respectively. Higher SSIM and PSNR scores, along with lower LPIPS values, indicate greater similarity to the ground truth. We observe that DRAUN achieves strong reconstructions on simpler models; however, recovering data from gradients becomes increasingly challenging with complex architectures such as ResNet18—an inherited limitation from prior gradient inversion attacks (Geiping et al., 2020b). igures

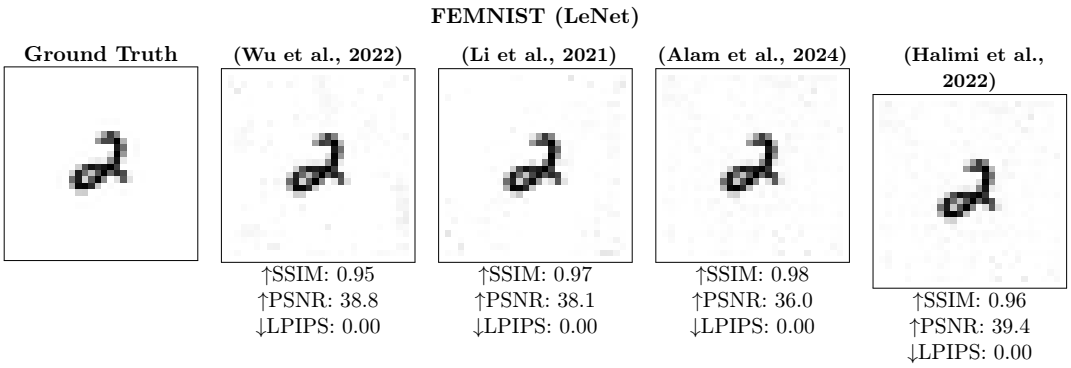

Figure 7: Single-image reconstructions on FEMNIST using LeNet. We compare DRAUN reconstructions from local updates of four unlearning methods. Higher SSIM/PSNR and lower LPIPS indicate better visual fidelity.

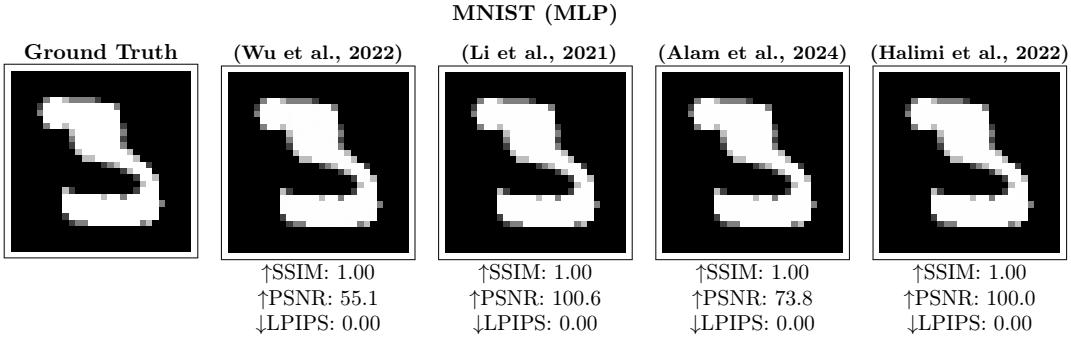

Figure 8: Single-image reconstructions on MNIST using MLP. We compare DRAUN reconstructions from local updates of four unlearning methods.

E.2 Reconstruction for Different Unlearning Step Size

We evaluated DRAUN under the FedAvg protocol, where the client is allowed to perform more than one local optimization step ($\mathcal{E} > 1$). Figures 10 and 11 show DRAUN reconstructions on the CIFAR-10 dataset using the ConvNet64 model for $\mathcal{E} = 2$ and $\mathcal{E} = 4$, respectively, across four unlearning algorithms. We observe that reconstruction efficacy deteriorates as the number of local epochs increases.

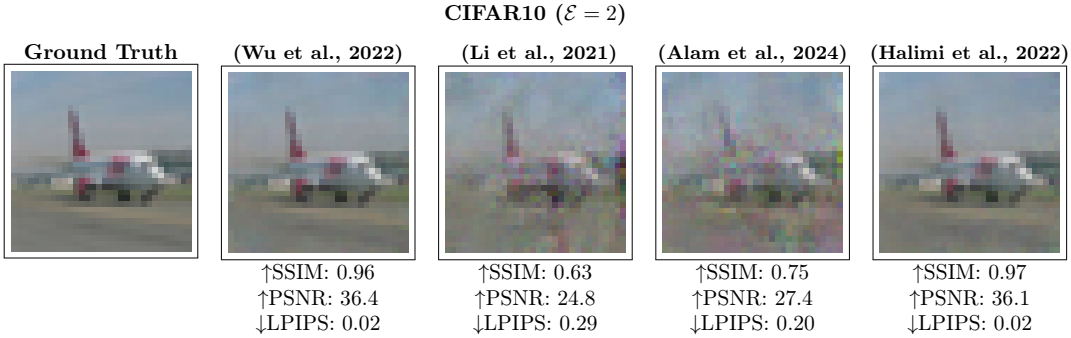

Figure 10: Reconstruction results from local updates with $\mathcal{E} = 2$ steps on CIFAR-10 using ConvNet64. DRAUN is applied to four unlearning methods. Higher SSIM/PSNR and lower LPIPS indicate better reconstruction quality.

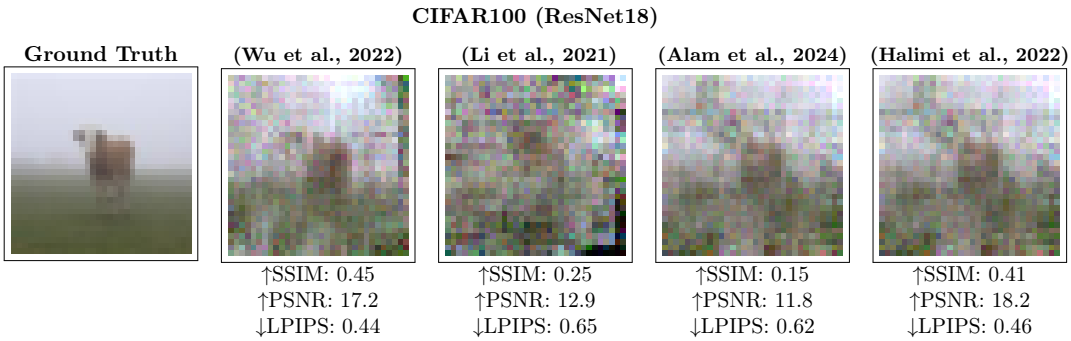

Figure 9: Single-image reconstructions on CIFAR100 using ResNet18. We compare DRAUN reconstructions from local updates of four unlearning methods.

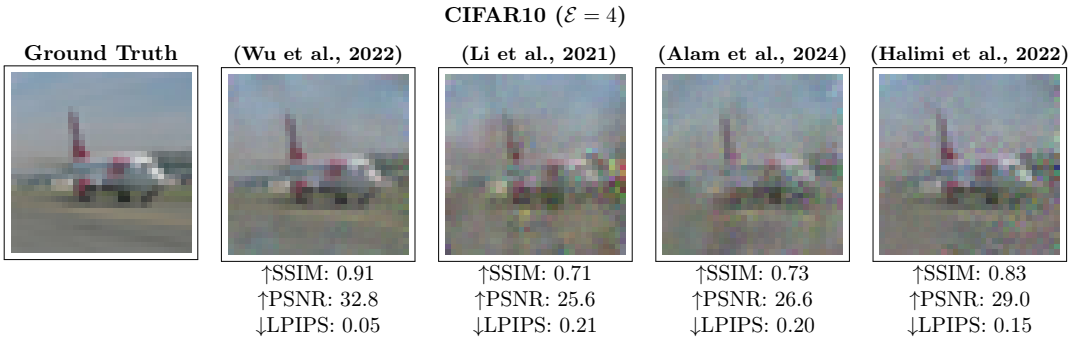

Figure 11: Reconstruction results from local updates with $\mathcal{E} = 4$ steps on CIFAR-10 using ConvNet64. DRAUN is applied to four unlearning methods. Higher SSIM/PSNR and lower LPIPS indicate better reconstruction quality.

E.3 Reconstructed Images for CIFAR100, FEMNIST and MNIST

Figures 12, 13, and 14 demonstrate the capability of DRAUN to reconstruct images across different datasets and unlearning algorithms using the ConvNet64 model.

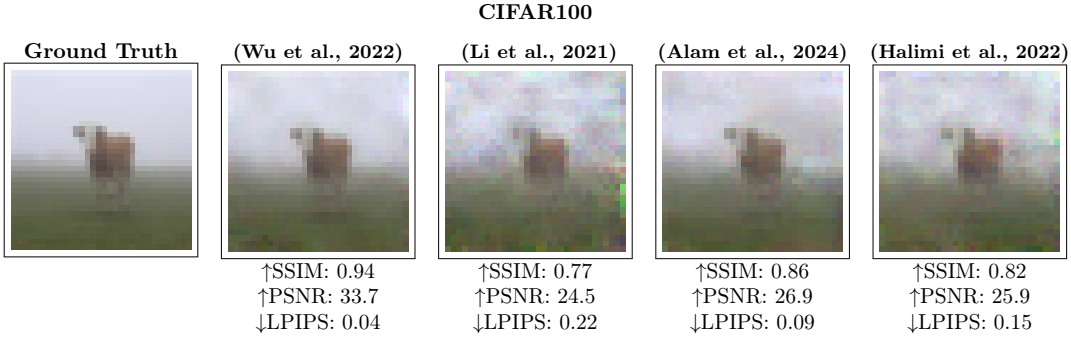

Figure 12: *DRAUN* reconstructions from local updates of four unlearning methods on CIFAR100.

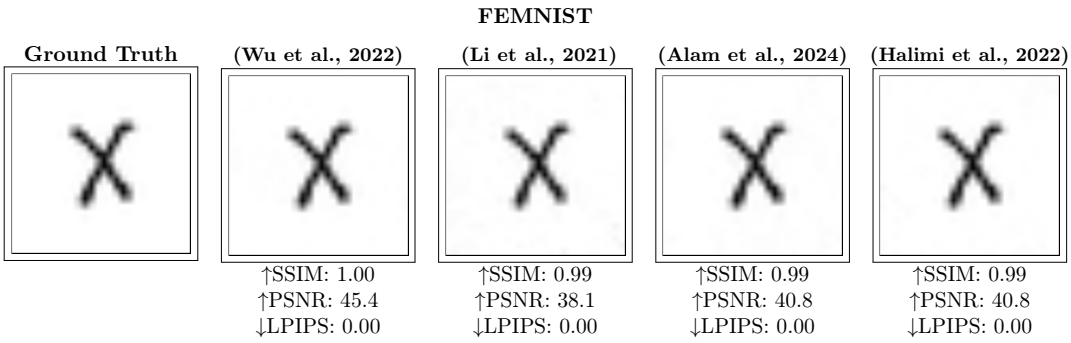

Figure 13: *DRAUN* reconstructions from local updates of four unlearning methods on FEMNIST.

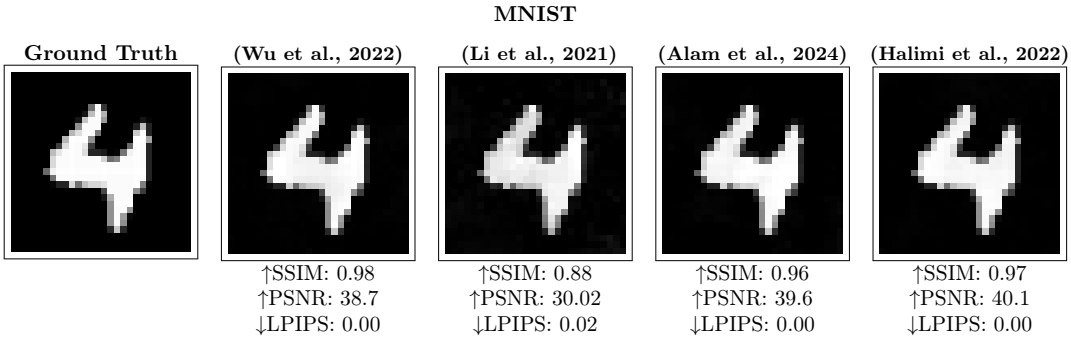

Figure 14: *DRAUN* reconstructions from local updates of four unlearning methods on MNIST.

E.4 Batch Reconstructions for Datasets and Batch Sizes

We evaluate DRAUN by reconstructing images from local model updates produced by two unlearning methods—ABL Li et al. (2021) and Halimi et al. (2022)—across four datasets: CIFAR-10, CIFAR-100, FEMNIST, and MNIST. Figures 15 to 24 show how reconstruction quality degrades as the batch size of removed samples increases from 4 to 128. While these figures focus on visual results from two representative methods, the accompanying metric plots (Figures 17 to 26) provide a broader comparison across all evaluated algorithms. They show consistent trends—declining SSIM and PSNR, and increasing LPIPS—highlighting the growing difficulty of unlearning larger forget sets.

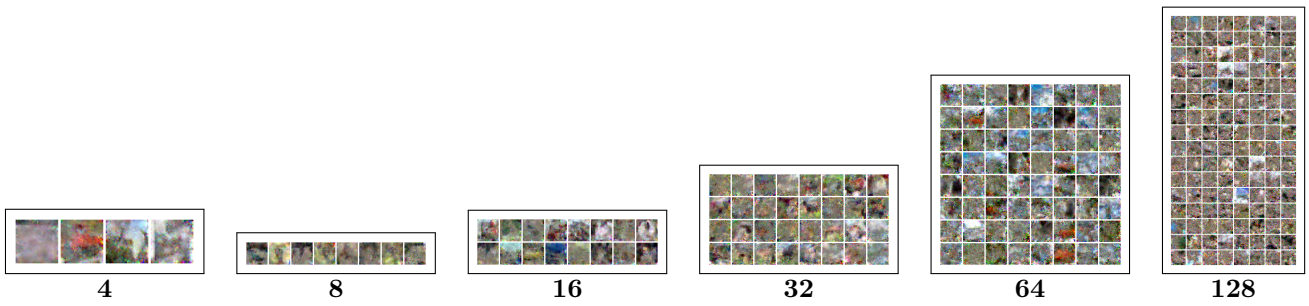

Figure 15: Multi image reconstruction from (Halimi et al., 2022) on CIFAR10.

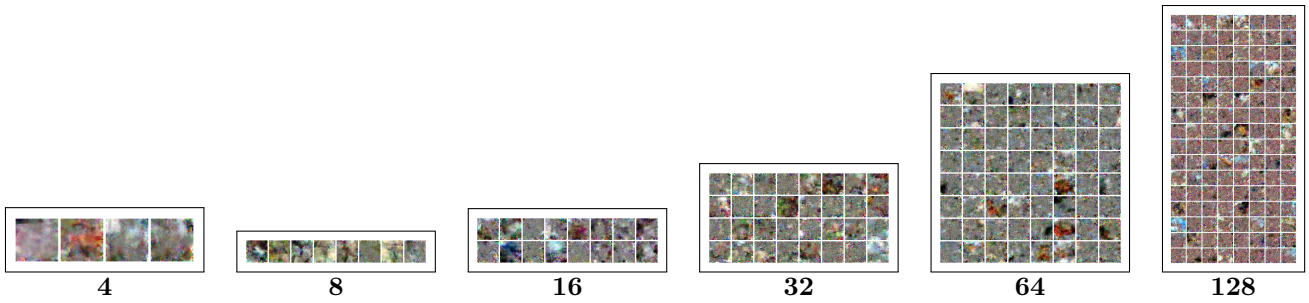

Figure 16: Multi image reconstruction from ABL (Li et al., 2021) on CIFAR10.

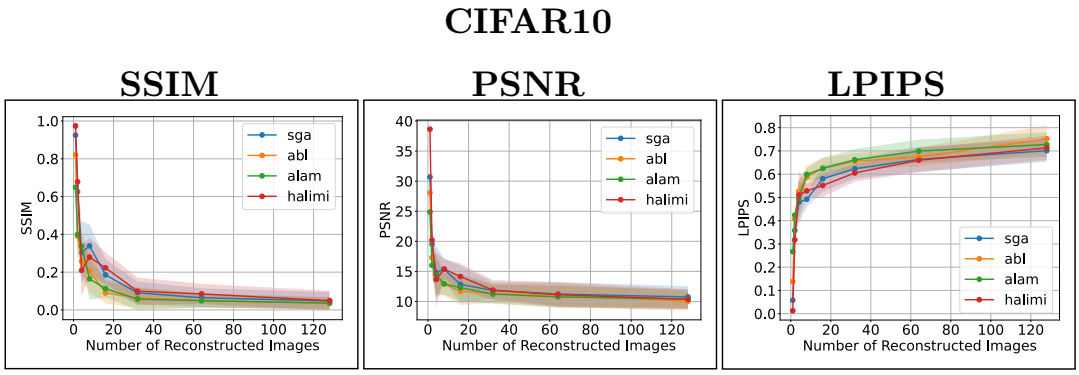

Figure 17: Effect of batch size scaling on reconstruction quality for the CIFAR10 dataset on four unlearning algorithms, measured using SSIM, PSNR, and LPIPS. Larger batch sizes tend to degrade reconstruction fidelity, as indicated by decreasing SSIM/PSNR and increasing LPIPS.

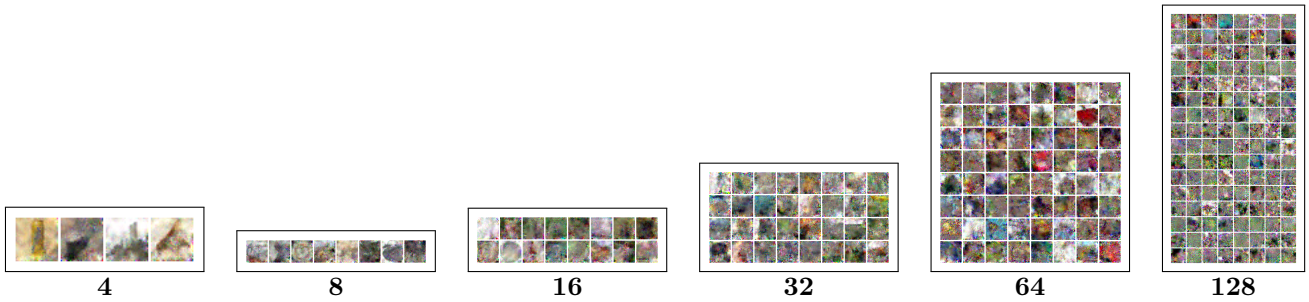

Figure 18: Multi image reconstruction from (Halimi et al., 2022) on CIFAR100.

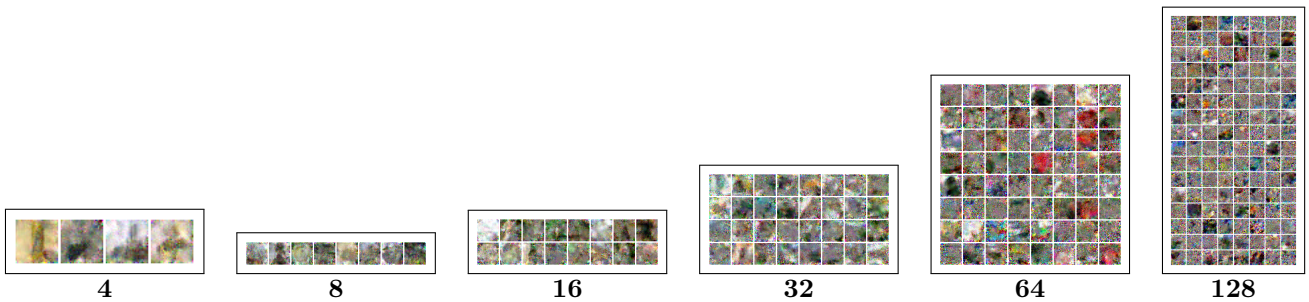

Figure 19: Multi image reconstruction from ABL (Li et al., 2021) on CIFAR100.

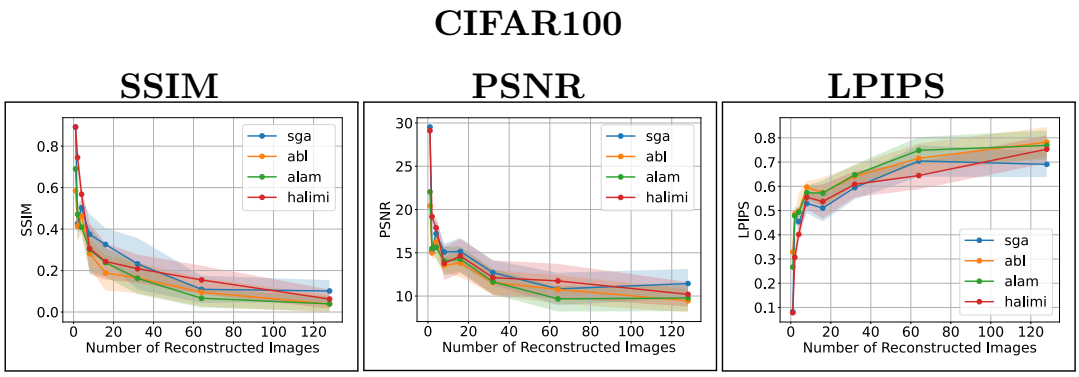

Figure 20: Effect of batch size scaling on reconstruction quality for the CIFAR100 dataset on four unlearning algorithms, measured using SSIM, PSNR, and LPIPS. Larger batch sizes tend to degrade reconstruction fidelity, as indicated by decreasing SSIM/PSNR and increasing LPIPS.

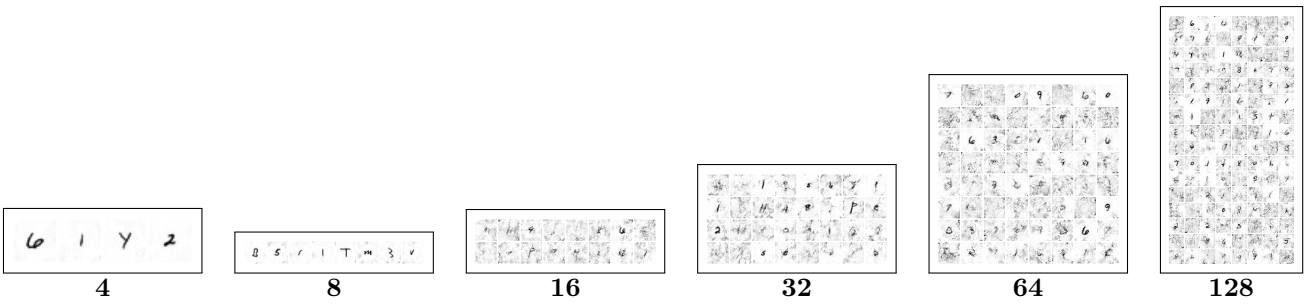

Figure 21: Multi image reconstruction from (Halimi et al., 2022) on FEMNIST.

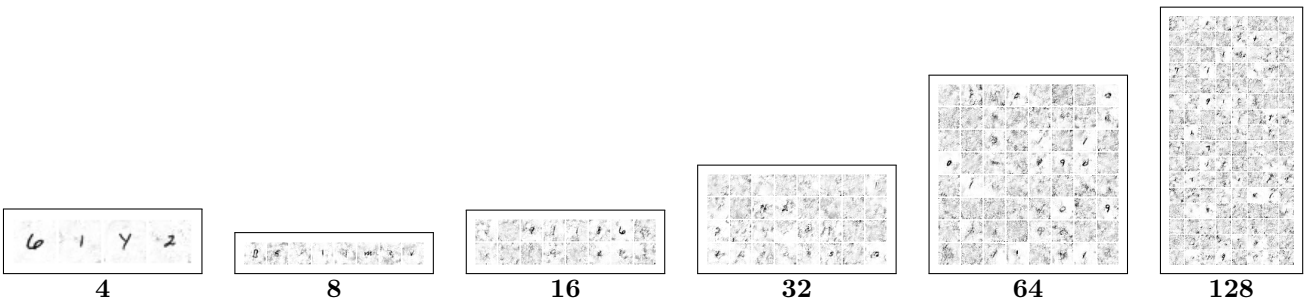

Figure 22: Multi image reconstruction from ABL (Li et al., 2021) on FEMNIST.

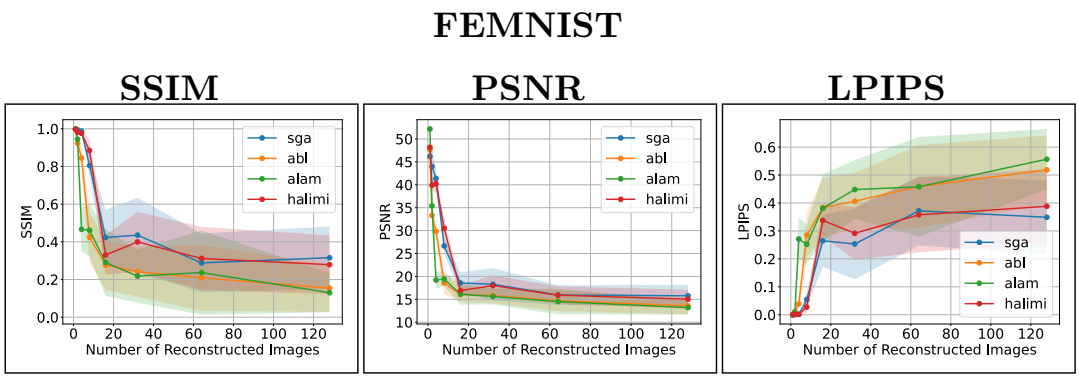

Figure 23: Effect of batch size scaling on reconstruction quality for the FEMNIST dataset on four unlearning algorithms, measured using SSIM, PSNR, and LPIPS. Larger batch sizes tend to degrade reconstruction fidelity, as indicated by decreasing SSIM/PSNR and increasing LPIPS.

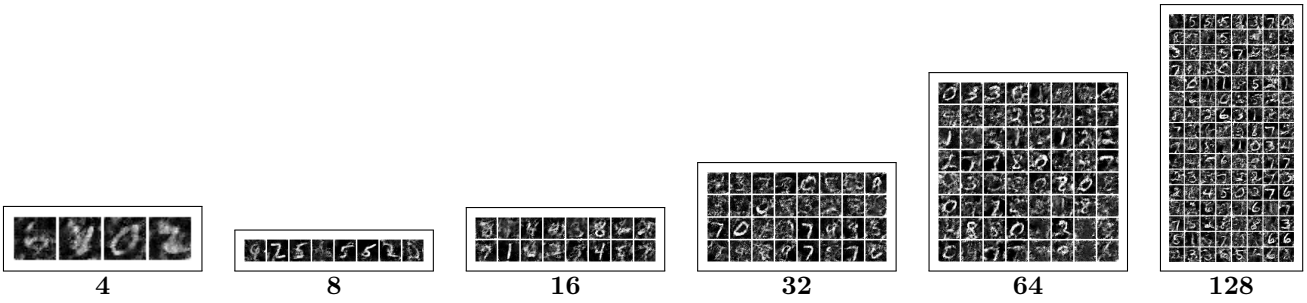

Figure 24: Multi image reconstruction from (Halimi et al., 2022) on MNIST.

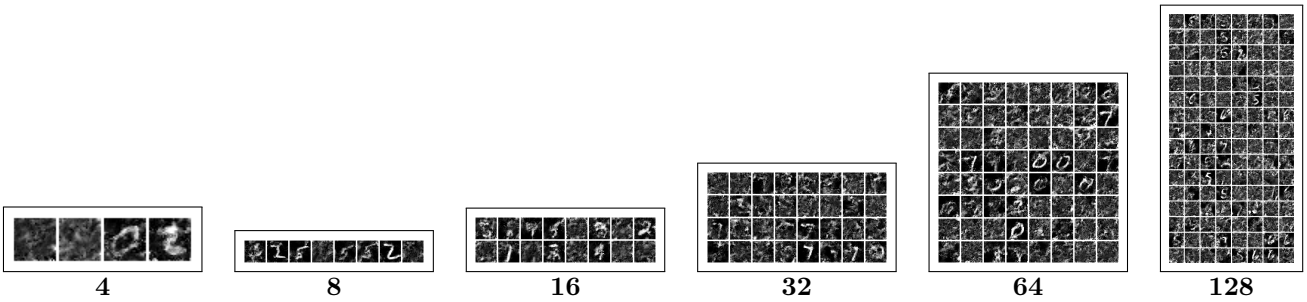

Figure 25: Multi image reconstruction from ABL (Li et al., 2021) on MNIST.

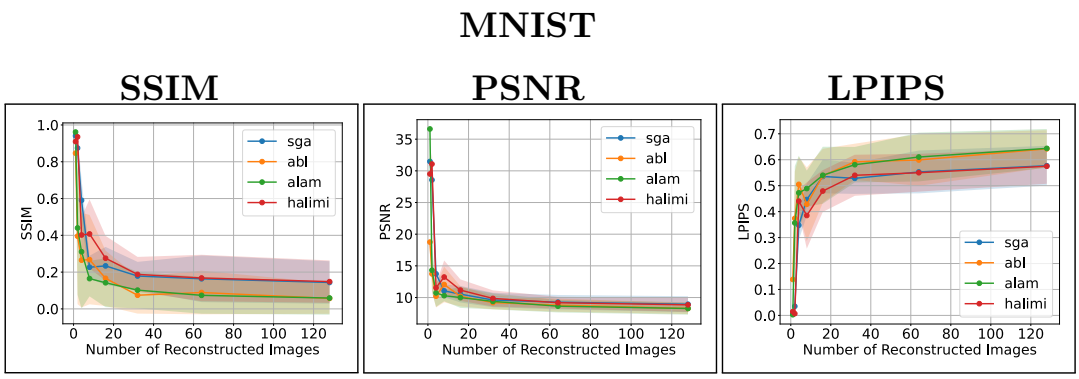

Figure 26: Effect of batch size scaling on reconstruction quality for the MNIST dataset on four unlearning algorithms, measured using SSIM, PSNR, and LPIPS. Larger batch sizes tend to degrade reconstruction fidelity, as indicated by decreasing SSIM/PSNR and increasing LPIPS.

E.5 Reconstruction Comparison between Optimization-Specific and Optimization-Agnostic

To evaluate the performance of DRAUN under different unlearning configurations, we compare reconstructions obtained using Alam Alam et al. (2024) in both agnostic and specific modes across four datasets: MNIST, FEMNIST, CIFAR-10, and CIFAR-100 (Figures 27–30). While the agnostic mode assumes minimal knowledge about the internal details of the unlearning algorithm, the specific mode leverages full access to its optimization logic and hyperparameters. As expected, the specific setup generally leads to better reconstructions, benefiting from targeted penalties and aligned update dynamics.

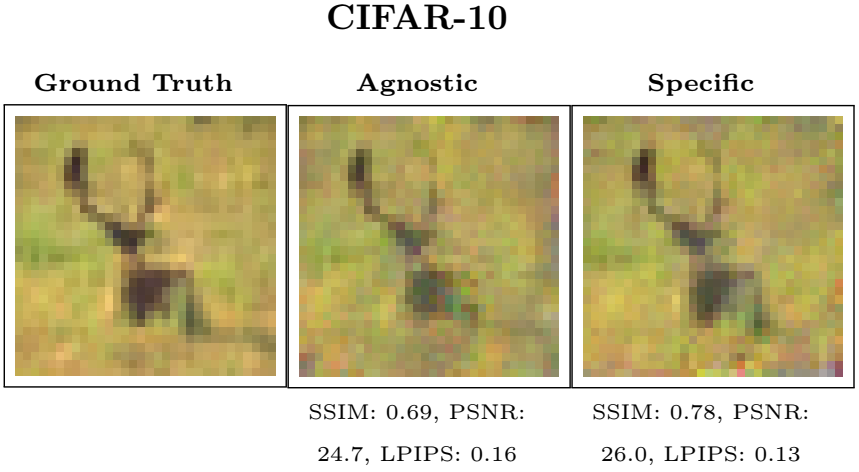

Figure 27: *DRAUN*: Agnostic vs Specific reconstruction on CIFAR-10 using Alam (Alam et al., 2024) unlearning.

CIFAR-100

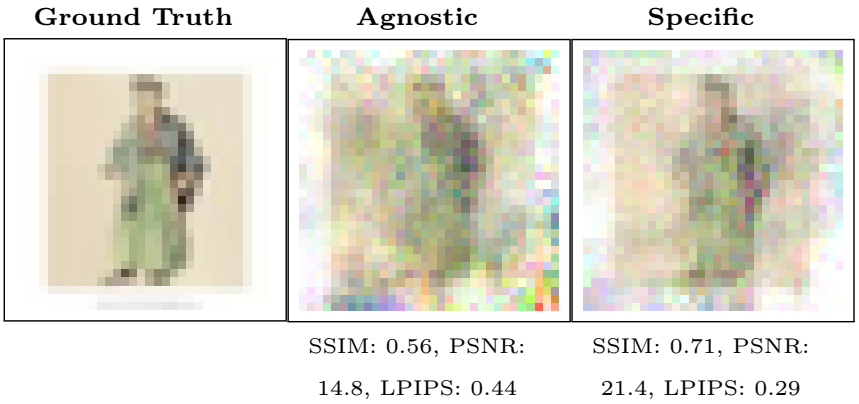

Figure 28: *DRAUN*: Agnostic vs Specific reconstruction on CIFAR-100 using Alam (Alam et al., 2024) unlearning.

FEMNIST

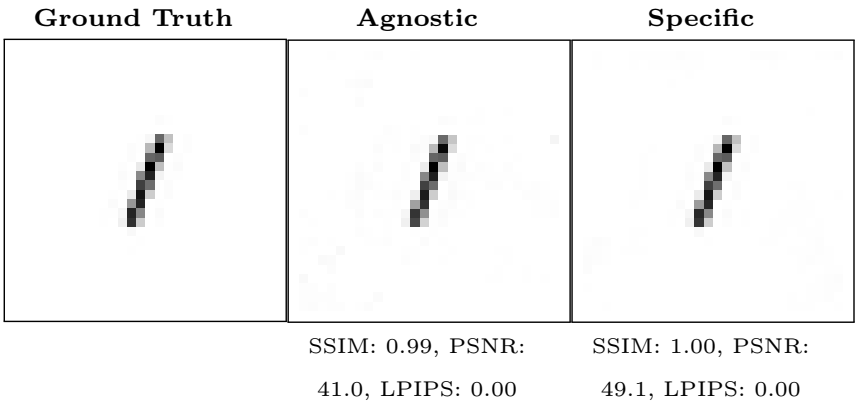

Figure 29: *DRAUN*: Agnostic vs Specific reconstruction on FEMNIST using Alam (Alam et al., 2024) unlearning.

MNIST

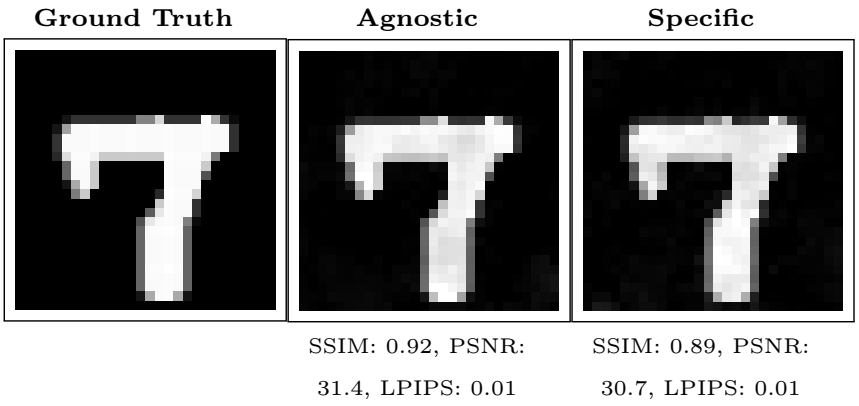

Figure 30: *DRAUN*: Agnostic vs Specific reconstruction on MNIST using Alam (Alam et al., 2024) unlearning.

F *DRAUN* for Second-Order Unlearning Algorithms

We extend DRAUN to the setting where the client employs second-order unlearning algorithms. While uncommon in federated learning due to their high computational cost, (Jin et al., 2024) show that Newton-style updates can be applied to linearized MLPs. To reconstruct the unlearned data, DRAUN generalizes classical gradient inversion attacks by simulating second-order updates on dummy data, as outlined in Algorithm 5. Specifically, the server approximates the client's update by computing Hessian-vector products (HVPs) using dummy inputs and aligns them with the true HVPs by minimizing their cosine distance. To reduce the computational burden of exact Hessian inversion, we follow the approximation strategy from (Zhou et al., 2025). Reconstruction is optimized via SGD with total variation (TV) regularization. For simplicity, we assume access to unlearn labels during reconstruction. Theoretical justifications for the correctness of this approach and the failure modes of classical GIA under second-order updates are provided in Appendix C.

Algorithm 5 DRAUN for Second-Order Algorithms

Client Input: Local model θ_c, datasets $(x_u, y_u), (x_r, y_r)$, learning rate η
$\nabla\mathcal{L}_u \leftarrow \nabla\mathcal{L}(x_u, y_u; \theta_s)$
$H_r \leftarrow H(x_r, y_r; \theta_s)$
$\Delta\theta \leftarrow H_r^{-1} \cdot \nabla\mathcal{L}_u$

Server Input: Global model θ_s, reconstruction step size η_{rec}, similarity loss $\mathcal{L}_{\text{sim}}$, TV weight λ_{TV}
Initialize dummy variables $\tilde{x}_u, \tilde{y}_u, \tilde{x}_r, \tilde{y}_r$ randomly
for $t = 1$ to T **do**
    $\nabla\tilde{\mathcal{L}}_u \leftarrow \nabla\mathcal{L}(\tilde{x}_u, \tilde{y}_u; \theta_s)$
    $\tilde{H}_r \leftarrow H(\tilde{x}_r, \tilde{y}_r; \theta_s)$
    $\tilde{\Delta\theta} \leftarrow \tilde{H}_r^{-1} \cdot \nabla\tilde{\mathcal{L}}_u$
    $\ell \leftarrow \mathcal{L}_{\text{sim}}(\Delta\theta, \tilde{\Delta\theta}) + \lambda_{\text{TV}} \cdot TV(\tilde{x}_u)$
    $\tilde{x}_u \leftarrow \tilde{x}_u - \eta_{\text{rec}} \cdot \nabla_{\tilde{x}_u}\ell$
    $\tilde{x}_r \leftarrow \tilde{x}_r - \eta_{\text{rec}} \cdot \nabla_{\tilde{x}_r}\ell$
end for
return $(\tilde{x}_u, \tilde{y}_u, \tilde{x}_r, \tilde{y}_r)$

G Broader Impacts

Recent works (Hu et al., 2024; Bertran et al., 2024) have shown that unlearning algorithms are vulnerable to data reconstruction attacks in Machine Learning as a Service (MLaaS) settings. However, this vulnerability remains unexplored in the context of federated unlearning—a setting where unlearning is critical due to data ownership and regulatory constraints. In this work, we demonstrate that federated unlearning is highly vulnerable when facing a malicious server. In realistic industrial scenarios, such as computer vision classification tasks, we show that the server can silently reconstruct the raw images that a client requests to unlearn—without interfering with the unlearning process. These images are often highly sensitive (e.g., medical X-rays) and typically involve a small number of samples, making accurate reconstruction more feasible. Our findings expose a serious privacy risk in current federated unlearning methods and call for a rethinking of their trust assumptions. We hope this work motivates future research on the privacy-utility trade-offs in federated unlearning, as well as the development of detection and mitigation strategies specifically tailored to this emerging threat model.