TARGETED FORGETTING: A FRAMEWORK FOR CLIENT-CLASS CENTRIC KNOWLEDGE REMOVAL IN FEDERATED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated unlearning (FU) aims to remove the influence of specific clients or classes from a trained global model upon request, without accessing raw data or degrading overall performance. Existing FU methods often struggle to fully eliminate targeted information in a single unlearning round, particularly under non-IID data distributions, and may adversely affect non-target performance. In this work, we propose a hybrid federated unlearning framework that supports client-level, class-level, and client-class unlearning through parameter-level updates. It combines server-side gradient residual correction (proxy filtering) with client-side perturbation-based pruning (weight negation) to precisely remove targeted influences without requiring access to local training data. This design enables efficient, data-independent unlearning and generalizes across IID, non-IID, and label-shifted settings. Experiments on MNIST, FMNIST and CIFAR10 demonstrate that our framework achieves effective single-client, multi-client, single-class, and multi-class unlearning with minimal impact on non-target accuracy, while maintaining scalability and robustness across federated rounds.

1 INTRODUCTION

Machine learning (ML) has become a foundational tool for data-driven decision making, with widespread deployment in privacy-sensitive domains such as healthcare, finance, and personalized services Sinaci et al. (2024). However, trained models may implicitly retain sensitive, outdated, or erroneous information, raising concerns about data privacy, regulatory compliance, and unintended information leakage Wu et al. (2025); Alajaji (2025). Such risks are further formalized by regulations such as the GDPR and CCPA, which grant individuals the right to request deletion of their personal data even after model deployment Worden (2024); Sartor & Lagioia (2020); ccp (2018).

Federated learning (FL) partially addresses these concerns by enabling decentralized model training while keeping raw data on client devices Huang et al. (2023). Although FL improves privacy by design, it also restricts the server's ability to inspect or retrain on client data, making post-hoc data removal significantly more challenging Okolo et al. (2025); Haripriya et al. (2025). In realistic federated settings, clients may become unavailable, unwilling to participate in retraining, or contribute heterogeneous and non-IID data distributions, rendering conventional data deletion or retraining strategies impractical Zhang et al. (2025); Ribero & Vikalo (2024); Crawshaw & Liu (2024). Federated unlearning (FU) has therefore emerged as a critical mechanism for removing the influence of specific samples, classes, or clients from a federated model without accessing raw data or performing full retraining Xing et al. (2025); Huang et al. (2025); Jeong et al. (2024); Rishat et al. (2026). Despite recent advances, existing FU approaches often struggle to precisely erase targeted information while preserving non-target performance.

One of the most persistent challenges in federated unlearning is the inherent data heterogeneity across clients, which often appears as imbalance in local datasets and inconsistent label distributions Jimenez-Gutierrez et al. (2025); He et al. (2024). While IID data lead to coherent gradients and predictable forgetting behavior, real-world FL is overwhelmingly non-IID, with clients holding skewed feature distributions or single-class dominance Jimenez-Gutierrez et al. (2025); Zhang et al. (2024). Prior work shows that such conditions amplify gradient divergence, slow global convergence, and make class-specific knowledge harder to isolate Bai et al. (2025). The problem is further intensified under label-shift or flipped-label scenarios Zec et al. (2024), where inconsistent labeling distorts decision boundaries and causes partial forgetting or unintended suppression of non-target information Ye et al. (2025). In federated settings, iterative aggregation also disperses target-class influence across rounds, making direct gradient correction or parameter rollback difficult Pan et al. (2025); Jeong et al. (2024).

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

Together, these factors expose persistent issues in current FU methods, including instability under heterogeneity, non-target accuracy loss, and limited control over targeted forgetting.

To overcome the limitations of current federated unlearning methods especially their difficulty in fully removing targeted class or client information in a single unlearning pass, we propose a framework that improves unlearning efficiency while preserving non-target performance. This hybrid design reduces catastrophic forgetting and model drift across IID, non-IID, and label-shift settings. Our proposed framework evaluate its behavior across multiple, diverse benchmark datasets, targeting both single- and multi-class unlearning scenarios under varying data distributions. Our main contributions can be summarized as follows:

- We design a hybrid federated unlearning framework that combines server-side gradient residual correction with client-side perturbation-based pruning to support targeted client-based, class-based, and client-class-based knowledge removal without full retraining.
- We evaluate the framework across multiple datasets under IID, non-IID, and label-shift settings to analyze forgetting effectiveness, retention stability, and convergence behavior.
- We extend the framework to multi-class unlearning, demonstrating scalability and robustness across heterogeneous client distributions.
- Our proposed framework also supports selective removal of specific class–client pairs, enabling more precise unlearning when only certain clients contribute corrupted class information.
- We provide an in-depth analysis of how data heterogeneity, dataset variation, and unlearning granularity affect forgetting behavior, offering insights for future adaptive and privacy-preserving FU systems.

2 Related work

In distributed environments, FU has emerged as a promising extension of the FL paradigm Woisetschläger et al. (2025). The goal of FU is to enable a global model to selectively forget specific information, whether from individual samples, clients, or data classes, without requiring centralized retraining or direct access to raw data Zhong et al. (2025). First, decentralized data ownership prevents direct verification of deleted samples, making it difficult to guarantee complete erasure. The prior work highlighted that decentralized training and model aggregation in federated learning complicates the verification and durability of data modifications across distributed participants Nie & Lu (2024). Second, communication constraints and asynchronous client participation amplify inconsistency in model updates, which complicates rollback and gradient inversion Valadi et al. (2025). Author Valadi's study proposed that prior gradient inversion studies overstate privacy risks by relying on unrealistic inference-mode assumptions Valadi et al. (2025). Third, statistical heterogeneity arising from non-IID or imbalanced local data distributions causes model parameters to become tightly entangled across clients, leading to unintended knowledge retention or over-forgetting when unlearning is applied Chunduru et al. (2025). Gutierrez's and Wuerkaixi's study found that non-IID data distributions in federated learning cause parameters to become entangled across clients, resulting in catastrophic forgetting and unintended knowledge loss when heterogeneity is not properly managed Wuerkaixi et al. (2025); Gutierrez et al. (2024). Moreover, repeated aggregation over multiple training rounds disperses client-specific information throughout the global model, making exact gradient subtraction or parameter restoration infeasible Liu et al. (2025a). Recent FU studies have explored techniques such as influence estimation, server-side gradient residual correction, and perturbation-based mechanisms to mitigate these challenges Liu et al. (2025b). However, to address these inherent complexities, researchers have begun exploring distinct unlearning objectives and methodological paradigms.

Federated unlearning methods generally fall into two categories: client-level and class-level unlearning Romandini et al. (2024); Liu et al. (2024). Client-level approaches aim to remove the influence of entire clients. Liu's work presented FedEraser, which enables client-level unlearning by reconstructing the model using stored historical parameter updates, trading server storage for faster unlearning Liu et al. (2021). Methods such as FedEraser Liu et al. (2021), VeriFi Gao et al. (2024), SFU, and KNOT Su & Li (2023) offer strong guarantees but depend on full retraining or large additional computation at the server, making them expensive to scale. Class-level unlearning instead focuses on removing information tied to specific categories, a growing need in applications with ethical or policy constraints Li et al. (2025). Recent work, including FedRecover Cao et al. (2023),

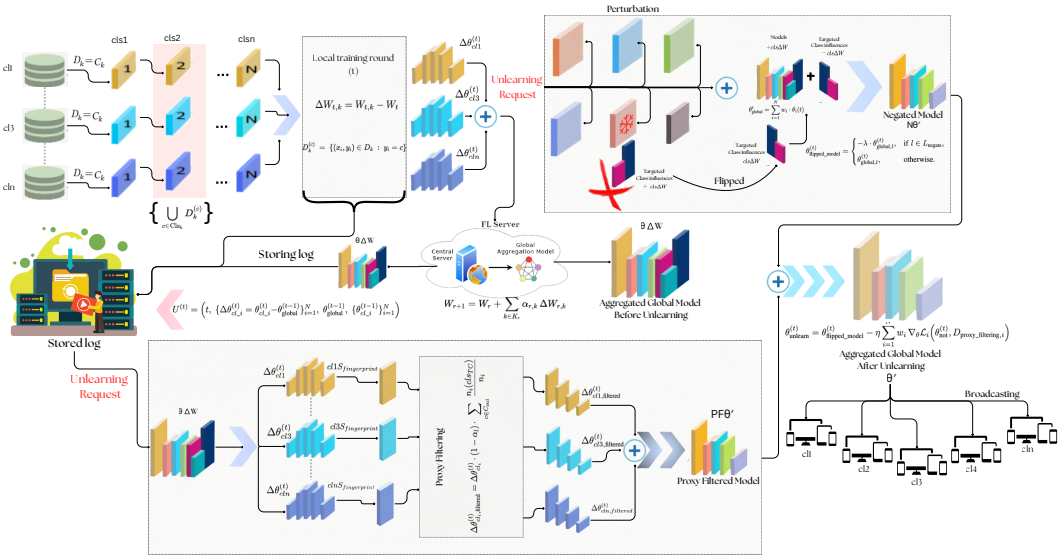


Figure 1: Workflow of the proposed unlearning framework, illustrating local training, unlearning request handling, perturbation-based updates, server aggregation, and model re-distribution

FedFilter Wang et al. (2023), and FedRecon Liu et al. (2025a) addresses this through gradient editing, distillation, or model-editing strategies. These challenges, along with the distinction between exact and approximate unlearning, reflect the diverse assumptions under which current FU methods are designed. However, most existing FU methods focus on a single unlearning granularity and often require multiple unlearning rounds or retraining to achieve effective forgetting, particularly under IID, non-IID, or poisoned data distributions. These limitations motivate the development of a more unified framework that can support client-level, class-level, and client-class pair unlearning while preserving non-target performance.

3 METHODOLOGY

The overall workflow of the proposed hybrid framework consists of two main phases: collaborative training and targeted federated unlearning. During the training phase, all participating clients collaboratively optimize a global federated model under the coordination of a central server. Local updates are aggregated using a weighted strategy to preserve generalization across heterogeneous data distributions. Once the model reaches convergence, the unlearning phase is triggered when a user or system requests the removal of specific client, class, or client-class information. Instead of retraining from scratch, the proposed hybrid procedure combines client-side perturbation-based pruning with server-side gradient residual correction. This coordinated process suppresses the targeted representations while reconstructing a stable global model with reduced influence from the requested client, class, or client-class contribution. The overall architecture is shown in Figure 1, which illustrates how the training and unlearning phases are integrated. A more detailed step-by-step interaction of the proposed two-phase unlearning workflow is provided in the sequence diagram in Appendix C (Figure 6). Let $\mathcal{D} = \bigcup_{k=1}^{K} \mathcal{D}_k$ denote the federated training dataset distributed across K clients, where $\mathcal{D}_k = \{(x_i^{(k)}, y_i^{(k)})\}_{i=1}^{n_k}$ represents the local dataset of client k with n_k samples. The global model $W_t \in \mathbb{R}^d$ at round t is obtained through federated aggregation:

$$W_{t+1} = W_t + \sum_{k=1}^{K} \alpha_{t,k} \Delta_{t,k} \tag{1}$$

where $\Delta_{t,k} = W_{t,k} - W_t$ denotes the local update from client k, and $\alpha_{t,k} \in [0,1]$ represents the aggregation weight satisfying $\sum_{k=1}^{K} \alpha_{t,k} = 1$. Given a target set $\mathcal{T}$ to unlearn (either classes T or client-class pairs (C,T)), we aim to obtain an unlearned model $\tilde{W}_t^*$ such that:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_f}[\ell(\tilde{W}_t^*, x, y)] \approx \mathbb{E}_{(x,y)\sim\mathcal{D}_f}[\ell(W_r, x, y)] \tag{2}$$

where $\mathcal{D}_f$ represents the forgotten data, and W_r is the retrained model trained exclusively on the remembered data $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_f$.

Phase 1: Client-Side Perturbation-Based Weight Negation
For each layer $l \in \mathcal{L}$, we decompose the weight matrix $W_{t^\star}[l]$ into target-specific and orthogonal components using projection operators:

$$W_{t^\star}[l] = P_{\perp\mathcal{T},l}(W_{t^\star}[l]) + P_{\mathcal{T},l}(W_{t^\star}[l]) \tag{3}$$

where $P_{\mathcal{T},l}(\cdot)$ projects onto the subspace spanned by target-specific features, and $P_{\perp\mathcal{T},l}(\cdot) = I - P_{\mathcal{T},l}(\cdot)$ projects onto its orthogonal complement. The projection operator is constructed from the class-specific gradient $g_{\mathcal{T},l}$:

$$g_{\mathcal{T},l} = \frac{1}{|\mathcal{D}_\mathcal{T}|} \sum_{(x,y)\in\mathcal{D}_\mathcal{T}} \nabla_{W[l]} \ell(W_{t^\star}, x, y) \tag{4}$$

where $\mathcal{D}_\mathcal{T}$ denotes samples belonging to target set $\mathcal{T}$. The normalized gradient direction is:

$$\hat{u}_{\mathcal{T},l} = \frac{g_{\mathcal{T},l}}{\|g_{\mathcal{T},l}\|_2} \tag{5}$$

The projection matrix is then $P_{\mathcal{T},l} = \hat{u}_{\mathcal{T},l}\hat{u}_{\mathcal{T},l}^T$. To determine the negation strength, we compute layer-specific sensitivity $\tilde{S}_l$ and selectivity R_l:

$$\tilde{S}_l = \frac{\|g_{\mathcal{T},l}\|_2}{\sum_{l'\in\mathcal{L}} \|g_{\mathcal{T},l'}\|_2} \tag{6}$$

$$R_l = \frac{\text{tr}(P_{\mathcal{T},l} \cdot \text{Cov}(W_{t^\star}[l]))}{\text{tr}(\text{Cov}(W_{t^\star}[l]))} \tag{7}$$

where $\text{Cov}(W_{t^\star}[l])$ denotes the covariance matrix of weights in layer l. The adaptive negation coefficient is:

$$\lambda_l = \text{clip}\left(\alpha \cdot \frac{\tilde{S}_l R_l}{\max_{j\in\mathcal{L}}(\tilde{S}_j R_j)}, 0, 1\right) \tag{8}$$

The negated weight matrix is:

$$W_{t^\star}^{(0)}[l] = P_{\perp\mathcal{T},l}(W_{t^\star}[l]) + (1 - 2\lambda_l) \cdot P_{\mathcal{T},l}(W_{t^\star}[l]) \tag{9}$$

This formulation ensures that: (1) orthogonal components remain unchanged, preserving non-target knowledge, and (2) target-specific components are negated with adaptive strength λ_l, disrupting learned representations while maintaining numerical stability.

Phase 2: Server-Side Gradient Residual Correction
We reconstruct the model state by replaying historical updates with target-specific filtering. Starting from checkpoint $\tau = \max\{t' : t' \bmod s = 0, t' \le t^\star\}$, $\tilde{W}_\tau = W_\tau$. For each round $t' \in [\tau, t^\star - 1]$, we compute the influence measure $\iota_{t',k}(\mathcal{T})$ quantifying the contribution of client k's update to target set $\mathcal{T}$:

$$\iota_{t',k}(\mathcal{T}) = \frac{1}{|\mathcal{D}_\mathcal{T}|} \sum_{(x,y)\in\mathcal{D}_\mathcal{T}} \left\langle \Delta_{t',k}, \nabla_W \ell(\tilde{W}_{t'}, x, y) \right\rangle \tag{10}$$

The server-side gradient residual correction weight follows an exponential decay:

$$w_{t',k}^{(\neg\mathcal{T})} = \exp(-\gamma \cdot \iota_{t',k}(\mathcal{T})) \tag{11}$$

where $\gamma > 0$ controls the decay rate. The filtered update is:

$$\tilde{W}_{t'+1} = \tilde{W}_{t'} + \sum_{k=1}^{K} w_{t',k}^{(\neg\mathcal{T})} \cdot \Delta_{t',k} \tag{12}$$

This mechanism ensures that updates with high target influence are exponentially suppressed, while preserving contributions from non-target data.

3.1 Class-Based Unlearning (Single and Multi Class)

We first describe the unlearning procedure for a *single target class* $T = \{c\}$, which forms the core operation of the proposed framework. Extensions to multi-class unlearning follow naturally from this formulation. For each layer l, we compute the class-specific gradient direction

$$g_{c,l} = \frac{1}{|\mathcal{D}_c|} \sum_{(x,y)\in\mathcal{D}_c} \nabla_{W[l]} \ell(W_{t^\star}, x, y), \qquad \hat{u}_{c,l} = \frac{g_{c,l}}{\|g_{c,l}\|_2}, \tag{13}$$

and define the projection onto the class-c subspace as

$$P_{c,l} = \hat{u}_{c,l}\hat{u}_{c,l}^T, \qquad P_{\perp c,l} = I - P_{c,l}. \tag{14}$$

The initial unlearned model is obtained via *layer-wise weight negation*

$$W_{t^\star}^{(0)}[l] = P_{\perp c,l}(W_{t^\star}[l]) + (1 - 2\lambda_l)\, P_{c,l}(W_{t^\star}[l]), \tag{15}$$

where $\lambda_l \in [0, 1]$ is an adaptive coefficient reflecting the sensitivity of layer l to the target class. When λ_l approaches 1, class-aligned features are strongly inverted, effectively disrupting class-specific representations, while orthogonal components are preserved. To remove historical contributions during replay, we estimate the influence of a client update $\Delta_{t',k}$ on the target class:

$$\iota_{t',k}(c) = \frac{1}{|\mathcal{D}_c|} \sum_{(x,y)\in\mathcal{D}_c} \left\langle \Delta_{t',k}, \nabla_W \ell(\tilde{W}_{t'}, x, y) \right\rangle, \tag{16}$$

and apply exponential filtering,

$$w_{t',k}^{(\neg c)} = \exp(-\gamma\, \iota_{t',k}(c)), \tag{17}$$

which suppresses updates that contributed strongly to learning the target class, while preserving updates unrelated to class c.

Multi-class unlearning is realized by extending the class-specific projection from a single gradient direction to the subspace spanned by multiple target-class gradients, i.e., by replacing $P_{c,l}$ with a summed projection over all $c \in T$. As detailed in algorithm 1.2 in Appendix C.1, the same client-side perturbation-based pruning and server-side gradient residual correction procedures are applied without further modification, enabling the simultaneous removal of multiple target classes within a unified unlearning step.

3.2 Client-Based Unlearning (Single and Multi Client)

Client-based unlearning removes the complete contribution of a selected set of clients C from the global model. Unlike class-based unlearning, where the target influence must be estimated across multiple clients, client attribution is directly known. Therefore, the server-side correction reduces to a hard client-selection indicator:

$$w_{t',k}^{(\neg C)} = \mathbf{1}[k \notin C], \tag{18}$$

where updates from target clients are completely excluded and updates from the remaining clients are preserved. Starting from the latest checkpoint before the unlearning request, the model is reconstructed by replaying only the historical updates of non-target clients:

$$\tilde{W}_{t'+1} = \tilde{W}_{t'} + \frac{1}{|\mathcal{K} \setminus C|} \sum_{k\notin C} \Delta_{t',k}. \tag{19}$$

After replay, a short calibration step is performed using only the retained clients to restore model stability and retained-task performance. Since the target is an entire client, no class-specific class-specific perturbation-based pruning is required. Thus, client-based unlearning can be viewed as the hard-filtering counterpart of the server-side gradient residual correction used for class-based and client-class-based unlearning.

3.3 Client-Class-Based Unlearning

While 3.1 removes a target class globally, our another main contribution is *client-class-based unlearning*, which forgets only the contribution of specific client-class pairs (C, T) while preserving the same class learned from non-target clients. The overall pipeline remains identical to section 3.1 (client-side perturbation-based pruning followed by server-side gradient residual correction); the only change is *what is considered "target"*: instead of using class-only gradients over $\mathcal{D}_c$, we compute client-conditioned class gradients over $\mathcal{D}_{k,c} = \{(x, y) \in \mathcal{D}_k : y = c\}$ and project onto their span. Concretely, for each layer l we form

$$g_{(k,c),l} = \frac{1}{|\mathcal{D}_{k,c}|} \sum_{(x,y)\in\mathcal{D}_{k,c}} \nabla_{W[l]} \ell(W_{t^\star}, x, y), \hat{u}_{(k,c),l} = \frac{g_{(k,c),l}}{\|g_{(k,c),l}\|_2}, \tag{20}$$

and define the target subspace as

$$P_{(C,T),l} = \sum_{k\in C} \sum_{c\in T} \hat{u}_{(k,c),l}\hat{u}_{(k,c),l}^T, P_{\perp(C,T),l} = I - P_{(C,T),l}. \tag{21}$$

Then, following the same client-side perturbation-based pruning (weight-negation) step in section 3.1, we initialize,

Algorithm 1 Federated Unlearning Procedure for Client, Class, and Client-Class-Based Unlearning

Require: Global model W_t, history $\{\Delta_{t,k}, \alpha_{t,k}\}$, checkpoints $\{W_\tau\}$, mode $M \in \{\text{client}, \text{class}, \text{client-class}\}$, targets C, T, layers $\mathcal{L}$, hyperparameters $\alpha, \gamma, s, E_{\text{cal}}$

A: Federated Training
while no unlearning request **do**
  Receive $\{W_{t,k}\}$; $\Delta_{t,k} \leftarrow W_{t,k} - W_t$
  $W_{t+1} \leftarrow W_t + \sum_k \alpha_{t,k} \Delta_{t,k}$
  Log $\{\Delta_{t,k}, \alpha_{t,k}\}$; store W_t if $t \bmod s = 0$
  $t \leftarrow t + 1$
end while
B: Unlearning at Round $t^\star$
$\tau \leftarrow \max\{\tau : \tau \le t^\star\}$; $\tilde{W}_\tau \leftarrow W_\tau$
if $M =$ client **then**
  Client-Based Unlearning
  $w_{t',k} \leftarrow \begin{cases} 0, & k \in C, \\ 1, & k \notin C \end{cases}$
else if $M =$ class **then**
  Class-Based Unlearning
  for each layer $l \in \mathcal{L}$ **do**
    Estimate target-class gradient $g_{T,l}$; $\hat{u}_{T,l} \leftarrow g_{T,l} / \|g_{T,l}\|$
    Compute sensitivity/selectivity and adaptive strength λ_l
    $W_{t^\star}^{(0)}[l] \leftarrow P_{\perp T,l}(W_{t^\star}[l]) + (1 - 2\lambda_l) P_{T,l}(W_{t^\star}[l])$
  end for
  $w_{t',k} \leftarrow \exp\big[-\gamma\, \iota_{t',k}(T)\big]$, where $\iota_{t',k}(T)$ is the class influence
else if $M =$ client-class **then**
  Client-Class-Based Unlearning
  for each layer $l \in \mathcal{L}$ **do**
    Estimate client-class gradient $g_{C,T,l}$; $\hat{u}_{C,T,l} \leftarrow g_{C,T,l} / \|g_{C,T,l}\|$
    Compute sensitivity/selectivity and adaptive strength λ_l
    $W_{t^\star}^{(0)}[l] \leftarrow P_{\perp(C,T),l}(W_{t^\star}[l]) + (1 - 2\lambda_l) P_{(C,T),l}(W_{t^\star}[l])$
  end for
  $w_{t',k} \leftarrow \begin{cases} \exp[-\gamma\, \iota_{t',k}(C,T)], & k \in C, \\ 1, & k \notin C \end{cases}$
  where $\iota_{t',k}(C,T)$ is the client-class influence
end if
C: Historical Replay / Filtering
for $t' = \tau$ to $t^\star - 1$ **do**
  $\tilde{W}_{t'+1} \leftarrow \tilde{W}_{t'} + \sum_k w_{t',k} \alpha_{t',k} \Delta_{t',k}$
end for
if $M =$ client **then**
  for $e = 1$ to E_{cal} **do**
    Calibrate using only non-target clients $k \notin C$
  end for
end if
return unlearned model $\tilde{W}_{t^\star}$

$$W_{t^\star}^{(0)}[l] = P_{\perp(C,T),l}(W_{t^\star}[l]) + (1 - 2\lambda_l)\, P_{(C,T),l}(W_{t^\star}[l]), \tag{22}$$

which selectively disrupts features attributable to (C, T) while leaving (i) non-target classes from clients in C and (ii) the target classes learned from clients outside C intact. During replay, we again reuse the server-side gradient residual correction mechanism from section 3.1, but compute influence only with respect to $\mathcal{D}_{C,T} = \bigcup_{k \in C} \bigcup_{c \in T} \mathcal{D}_{k,c}$ and apply filtering only to updates from target clients:

$$w_{t',k'}^{(\neg(C,T))} = \begin{cases} \exp(-\gamma\, \iota_{t',k'}(C,T)), & k' \in C, \\ 1, & k' \notin C. \end{cases} \tag{23}$$

This formulation reduces to single client-class unlearning when $|C| = |T| = 1$ and extends directly to multiple client-class pairs without changing the overall procedure. The complete algorithmic description is provided in Algorihm 1. More explicit description in Appendix C.1 (Algorithm 1.1, 1.2 and 1.3), and its effectiveness is supported by the confusion-matrix evidence reported under IID, non-IID, and label-flip settings. For clarity, preliminary definitions and background on the unlearning components are deferred to Appendix B. The complete theoretical analysis of the proposed hybrid federated unlearning operator, including smoothness, utility preservation, forgetting behavior, and membership-inference resilience, is provided in Appendix C.

4 Experiments and Result Analysis

Dataset: We evaluate federated unlearning on three standard image classification benchmarks: MNIST LeCun et al. (2002), FMNIST Xiao et al. (2017) and CIFAR10 Krizhevsky & Hinton (2009). These datasets are chosen to present increasingly difficult classification scenarios, from simple gray-scale digits to visually complex clothing images and color natural images, so that we can assess

the generality of the proposed unlearning mechanism on data with different characteristics. They also provide controlled settings to simulate practical unlearning requirements, such as removing contributions from specific clients, deleting specific classes, or removing corrupted client-class information due to label errors or poisoning. Models are trained on the respective training sets and evaluated on held-out test sets. To simulate realistic federated settings, we consider IID, non-IID and label-shifted (flipped-label) client settings. Unlearning is activated at a fixed federated round and performance is evaluated for client-based, class-based and client-class specific based unlearning to evaluate targeted information removal under heterogeneous data conditions.

Baselines: We evaluate several representative federated unlearning methods. FedAF, NoT, and FUSED support class-level unlearning, while other methods are mainly limited to client-level removal and do not explicitly address client-class scenarios. FedAF applies a neuro-inspired active forgetting mechanism with teacher-student learning and elastic weight consolidation Yang et al. (2024). FUSED uses sparse adapters for reversible and computationally efficient unlearning Zhong et al. (2025), while NoT applies weight negation to reduce inter-layer co-adaptation without extra data or storage Khalil et al. (2025). Although these methods support class-based unlearning, none explicitly perform client-wise class-based unlearning. Additional details on the experimental environment, implementation settings, hyperparameters, and evaluation metrics are provided in the appendix E.

Table 1 compares class-based unlearning across FUSED, NoT, FedAF, and our framework. All methods increase Error$_f$ after unlearning, confirming target-class removal. However, FUSED and FedAF show larger Error$_r$ and Error$_t$, while NoT retains partial target information. In contrast, our framework achieves near-complete forgetting with lower retention and test errors across datasets. Table 2 further evaluates client-class-specific unlearning, where selective removal of client-dependent class contributions increases Error$_f$ while reducing Error$_r$ and Error$_t$, demonstrating stable performance under fine-grained unlearning. This capability extends beyond standard class-based unlearning to more precise federated unlearning scenarios.

Table 1: Class-based unlearning performance before and after unlearning.

Dataset	Model	Pre-Unlearning				Post-Unlearning			
		Acc.	$Error_f$	$Error_r$	$Error_t$	Acc.	$Error_f$	$Error_r$	$Error_t$
MNIST	FUSED	99.50%	0.28%	0.25%	0.50%	88.61%	100.00%	1.09%	11.39%
	FedAF	98.82%	0.56%	0.85%	1.18%	89.02%	100.00%	0.80%	10.98%
	NoT	98.68%	0.57%	0.73%	1.32%	88.63%	94.34%	1.18%	11.37%
	Proposed framework	98.87%	0.33%	0.87%	1.13%	**89.35%**	**100.00%**	**0.25%**	**10.65%**
FMNIST	FUSED	91.56%	5.62%	6.59%	8.44%	81.18%	100.00%	8.83%	18.82%
	FedAF	91.65%	6.23%	6.48%	8.35%	77.93%	100.00%	12.51%	22.07%
	NoT	91.33%	5.58%	6.69%	8.67%	80.98%	99.91%	9.09%	19.02%
	Proposed framework	90.93%	7.12%	7.35%	9.07%	**83.20%**	**100.00%**	**5.44%**	**16.80%**
CIFAR10	FUSED	82.34%	31.40%	16.14%	17.67%	82.34%	31.34%	16.14%	17.67%
	NoT	81.50%	35.70%	16.58%	18.50%	74.83%	59.40%	21.36%	25.17%
	Proposed framework	87.61%	28.5%	10.6%	12.39%	72.98%	**100.00%**	18.92%	27.02%

Table 2: Client–class specific unlearning performance before and after unlearning.

Model	Dataset	Before Unlearning				After Unlearning			
		Acc.	Error$_f$	Error$_r$	Error$_t$	Acc.	Error$_f$	Error$_r$	Error$_t$
Proposed framework	MNIST	87.64%	99.19%	11.16%	12.36%	98.87%	50.65%	0.56%	1.13%
	FMNIST	82.59%	14.13%	15.16%	17.41%	89.37%	50.58%	7.44%	10.63%
	CIFAR10	71.47%	71.45%	17.80%	28.54%	60.07%	92.40%	26.82%	39.94%

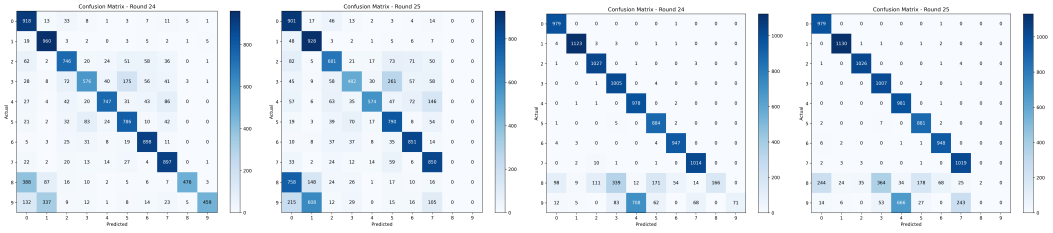
4.1 Analysis of Unlearning Trade-offs and Privacy Behavior

Table 3 compares class-based unlearning performance across FUSED, NoT, FedAF, and our proposed framework in terms of unlearning time, test loss, and membership inference attack (MIA) behavior. While baseline methods achieve favorable performance on individual metrics, their results reflect different trade-offs across efficiency, utility, and privacy. Some methods achieve lower MIA accuracy but require longer execution time or incur higher test loss, whereas others are computationally faster but show less favorable privacy or utility behavior. Our proposed framework does not achieve the best value on every individual metric, but maintains a reasonable margin across efficiency, utility, and privacy, with moderate unlearning time and test loss while keeping MIA accuracy close to chance level. Table 8 in Appendix D further evaluates client-class-specific unlearning, which is not implemented by the evaluated baseline methods, enabling selective removal of class contributions from targeted clients. Additional privacy and efficiency results are reported in Appendix D Tables 4, 5, 7, and 8, which provide detailed MIA-based analysis and client-class unlearning performance.

Table 3: Class-specific unlearning performance, reporting unlearning time, test loss, and membership inference attack (MIA) accuracy.

Dataset	Model	Time (s)	Test Loss	MIA Accuracy
MNIST	FUSED	130.08	**0.0182**	88.29%
	NoT	360.43	14.7425	**31.53**%
	FedAF	**41.63**	2.5137	93.17%
	Proposed framework	80.14	1.6307	50.08%
FMNIST	FUSED	205.49	**0.2486**	74.72%
	NoT	443.71	14.8022	44.48%
	FedAF	**92.82**	2.6054	77.91%
	Proposed framework	184.30	1.1485	**46.29**%
CIFAR10	FUSED	167.65	**0.5825**	55.40%
	NoT	**163.22**	0.7279	54.24%
	Proposed framework	341.48	1.1516	**52.05**%

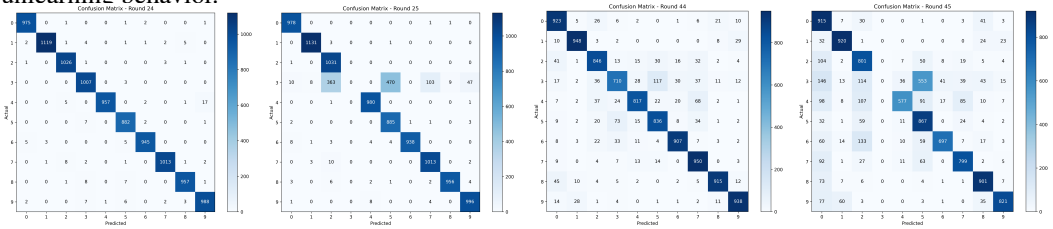
4.2 Effectiveness Analysis



(a) Pre-unlearn(CIFAR10) (b) Post-unlearn(CIFAR10) (c) Pre-unlearn(MNIST) (d) Post-unlearn(MNIST)

Figure 2: CIFAR10 and MNIST confusion matrices before and after Client Based Unlearning

Figure 2 shows the confusion matrices before and after client-specific unlearning. Before unlearning, the model retains strong predictive performance across the target and non-target classes. After unlearning, predictions associated with the targeted client-class contributions are strongly suppressed, while most non-target classes preserve their overall classification structure, demonstrating selective client-specific forgetting with limited impact on retained knowledge. Figures 3(a) and 3(c) show high class-based classification accuracy before unlearning for MNIST and CIFAR10 under IID settings, including the target class. After unlearning, Figures 3(b) and 3(d) show clear removal of the target class, while predictions for the remaining classes remain largely unchanged with minimal spillover effects. This behavior confirms that proposed framework achieves exact and isolated class-level forgetting without inducing unnecessary degradation in non-target classes across different dataset. The corresponding confusion matrices for FMNIST are presented in the Appendix, showing a similar unlearning behavior.

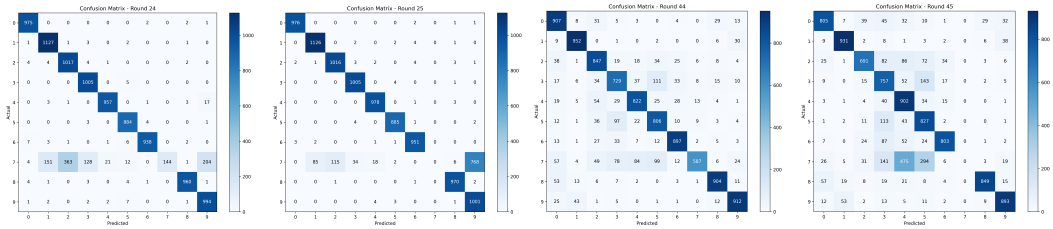


(a) Pre-unlearn(MNIST) (b) Post-unlearn(MNIST) (c) Pre-unlearn(CIFAR10) (d) Post-unlearn(CIFAR10)

Figure 3: Confusion matrices before & after class-based unlearning on MNIST & CIFAR10(IID)

We next evaluate under a Non-IID data distribution, where samples from a specific class are confined to a small subset of clients. This setting reflects realistic federated environments characterized by data heterogeneity and client bias. As shown in Figures 4(a) and 4(c), the proposed framework initially learns the target class for MNIST and CIFAR10, with corresponding FMNIST results provided in the Appendix. After unlearning, Figures 4(b) and 4(d) show that the model can no longer predict the target class. This result highlights the robustness of the method in handling heterogeneous and skewed data distributions, a key challenge in practical federated learning deployments.

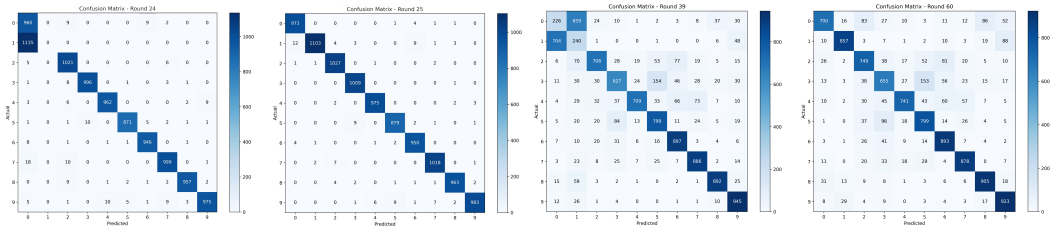
In the final analysis, we examine a challenging label-flipping scenario in which two adversarial clients inject corrupted samples from two classes, resulting in client-specific class noise. As shown in Figure 5(a), the presence of flipped labels introduces systematic bias into the global model, distorts decision boundaries, and causes a clear drop in overall accuracy. We then apply client-wise multi-class unlearning that selectively removes only the flipped class contributions from the affected clients. As



(a) Pre-unlearn(MNIST) (b) Post-unlearn(MNIST) (c) Pre-unlearn(CIFAR10) (d) Post-unlearn(CIFAR10)

Figure 4: Confusion matrices before & after class-based unlearning on MNIST & CIFAR10(non-IID)

illustrated in Figure 5(b), this targeted unlearning preserves clean class representations learned from benign clients while effectively eliminating adversarial influence, leading to a notable improvement in global accuracy and model stability. Crucially, existing federated unlearning methods do not support this form of client-specific class-level unlearning under flipped data distributions. This experiment demonstrating that precise client-wise class unlearning can not only correct bias but also actively strengthen federated models in adversarial and heterogeneous environments.



(a) Pre-unlearn(MNIST) (b) Post-unlearn(MNIST) (c) Pre-unlearn(CIFAR10) (d) Post-unlearn(CIFAR10)

Figure 5: Confusion matrices before & after client-class-specific unlearning on MNIST & CIFAR10(label-flipped)

5 Limitations

This study focuses on a controlled federated setting to enable clear evaluation of class-level, client-level and client–class-specific unlearning behavior. The experiments are conducted on MNIST, FMNIST and CIFAR10 with five and ten simulated clients, providing a consistent testbed for analyzing forgetting effectiveness, retention stability, and behavior under IID, non-IID, and label-flipped distributions. In addition, the current evaluation uses standard empirical indicators, including performance/error metrics, confusion matrices, and membership-inference attacks, to assess forgetting behavior.

6 Conclusions and Future Works

In this paper, we propose a hybrid federated unlearning framework for targeted client-level, class-level, and client-class-specific knowledge removal without full retraining. The proposed framework integrates client-side perturbation-based pruning with server-side gradient residual correction to effectively reduce target-class influence while preserving non-target performance. Experiments on MNIST, FMNIST and CIFAR-10 in IID, non-IID and label-flipped settings show consistent forgetting behavior and highlight the usefulness of fine-grained client-class unlearning.

The proposed mechanism is not tied to a specific dataset or model architecture, as it operates on model updates, target-specific gradient directions, and historical information. Thus, the same framework can be adapted to different neural architectures, larger client populations, and other federated tasks by defining the corresponding target subspaces and replay filters. The framework is not limited to the current experimental settings. More extensive evaluation is required for further validation of scalability and generalizability. Future work will expand the evaluation to larger datasets, more clients, and more complex model architectures. We also plan to explore certified unlearning guaranties, instance-level deletion, and more robust privacy-preserving mechanisms, such as differential privacy, secure aggregation, and formal privacy verification protocols for large-scale federated deployments.

AI use statement

In this work we employed generative AI tools for language refinement, organization and limited help in debugging. All text, code, and technical content generated by AI was reviewed, verified, and revised by the authors for accuracy, originality, and agreement with the experiments. The main

scientific analysis and experiment were conducted by the authors themselves. Generative AI was not responsible for the experimental results and the final scientific conclusions. The authors bear full responsibility for all content, claims, results and artifacts associated with this work.

ETHICS STATEMENT

The experiments use publicly available benchmark datasets such as MNIST, FMNIST and CIFAR10 and do not involve human subjects, private user data or sensitive personal information. The proposed federated unlearning framework is intended to support privacy-preserving data deletion and to mitigate the impact of undesired or corrupted data in federated models. Yet, unlearning mechanisms can be misused to delete information that should be preserved for auditing or accountability purposes. Therefore, practical implementation should involve suitable authorization, logging and verification of unlearning requests. All experiments were conducted for research purposes. The authors take responsibility for the integrity and reporting of the study.

REPRODUCIBILITY STATEMENT

For reproducibility, the full formulation of the proposed federated unlearning framework, including the class-based, client-based and client-class-based procedures, is given in the paper. Detailed algorithms and theoretical derivations are presented in the Appendix . The experimental section and supplementary material describe the datasets, client settings, model architectures, training configurations, hyperparameters, evaluation metrics, and unlearning protocols used in our experiments. We also provide implementation details and code in the supplementary material to facilitate the reproduction of the reported results.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the PRISM Lab at the University of North Texas for providing the resources and research support that contributed to this work.

REFERENCES

California consumer privacy act (ccpa/cpra), civil code 1798.105: Right to deletion, 2018. URL https://leginfo.legislature.ca.gov/faces/codes_displaySection.xhtml?lawCode=CIV&sectionNum=1798.105. Accessed 2025-12-18.

A. Alajaji. Fortinids: Defending smart city iot infrastructures against transferable adversarial poisoning in machine learning-based intrusion detection systems. *Sensors*, 25(19):6056, 2025. doi: 10.3390/s25196056.

Y. Bai, Y. Sun, T. Chen, W. Chen, S. Zhou, and Z. Niu. Fedteddi: Temporal drift and divergence aware scheduling for timely federated edge learning. *arXiv preprint arXiv:2509.07342*, 2025.

X. Cao, J. Jia, Z. Zhang, and N. Z. Gong. Fedrecover: Recovering from poisoning attacks in federated learning using historical information. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pp. 1366–1383, 2023.

A. Chunduru, M. Morafah, M. Morafah, V. P. Chellapandi, and A. Li. Avoid forgetting by preserving global knowledge gradients in federated learning with non-iid data. *arXiv preprint arXiv:2505.20485*, 2025.

M. Crawshaw and M. Liu. Federated learning under periodic client participation and heterogeneous data: A new communication-efficient algorithm and analysis. *Advances in Neural Information Processing Systems*, 37:8240–8299, 2024.

X. Gao, X. Ma, J. Wang, Y. Sun, B. Li, S. Ji, et al. Verifi: Towards verifiable federated unlearning. *IEEE Transactions on Dependable and Secure Computing*, 21(6):5720–5736, 2024.

D. M. J. Gutierrez, D. Solans, M. A. Heikkilä, A. Vitaletti, N. Kourtellis, A. Anagnostopoulos, and I. Chatzigiannakis. Non-iid data in federated learning: A survey with taxonomy, metrics, methods, frameworks and future directions. *CoRR*, 2024.

R. Haripriya, N. Khare, and M. Pandey. Privacy-preserving federated learning for collaborative medical data mining in multi-institutional settings. *Scientific Reports*, 15(1):12482, 2025. URL https://doi.org/10.1038/s41598-025-97565-4.

Z. He, Y. Li, D. Seo, and Z. Cai. Fedcpd: Addressing label distribution skew in federated learning with class proxy decoupling and proxy regularization. *Information Fusion*, 110:102481, 2024.

D. Huang, Y. Yang, and Y. Huang. Decentralized federated learning privacy-preserving framework based on blockchain. In *Proceedings of the 2023 3rd International Conference on Electronic Information Engineering and Computer Communication (EIECC)*, pp. 517–520, 2023.

Z. Huang, D. Wu, J. Bai, J. Zhang, S. Cao, J. Zhang, and Y. Hu. Federated unlearning in the wild: Rethinking fairness and data discrepancy. *arXiv preprint arXiv:2510.07022*, 2025.

H. Jeong, S. Ma, and A. Houmansadr. A survey on federated unlearning: Challenges and opportunities. *arXiv preprint arXiv:2403.02437*, 2024.

D. M. Jimenez-Gutierrez, M. Hassanzadeh, A. Anagnostopoulos, I. Chatzigiannakis, and A. Vitaletti. A thorough assessment of the non-iid data impact in federated learning. *arXiv preprint arXiv:2503.17070*, 2025.

Y. H. Khalil, L. Brunswic, S. Lamghari, X. Li, M. Beitollahi, and X. Chen. Not: Federated unlearning via weight negation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 25759–25769, 2025.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002.

C. Li, H. Jiang, J. Chen, Y. Zhao, S. Fu, F. Jing, and Y. Guo. An overview of machine unlearning. *High-Confidence Computing*, 5(2):100254, 2025. URL https://doi.org/10.1016/j.hcc.2024.100254.

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

J. Liu, G. Zeng, D. Wang, Y. Gao, Y. Yin, and Y. Jin. Fedrecon: Missing modality reconstruction in distributed heterogeneous environments. *arXiv preprint arXiv:2504.09941*, 2025a.

Y. Liu, H. Chen, W. Huang, Y. Ni, and M. Imani. Recover-to-forget: Gradient reconstruction from lora for efficient llm unlearning. In *Proceedings of the NeurIPS Workshop on Socially Responsible and Trustworthy Foundation Models*, 2025b.

Z. Liu, Y. Jiang, J. Shen, M. Peng, K. Y. Lam, X. Yuan, and X. Liu. A survey on federated unlearning: Challenges, methods, and future directions. *ACM Computing Surveys*, 57(1):1–38, 2024.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1273–1282. PMLR, 2017.

H. Nie and S. Lu. Persistverify: Federated model ownership verification with spatial attention and boundary sampling. *Knowledge-Based Systems*, 293:111675, 2024. URL https://doi.org/10.1016/j.knosys.2024.111675.

J. N. Okolo, A. A. G. Arowogbadamu, S. A. Adeniji, and R. K. Tasie. Federated learning for privacy-preserving data analytics in mobile applications. *World Journal of Advanced Research and Reviews*, 2025. URL https://doi.org/10.30574/wjarr.2025.26.1.1099.

Z. Pan, Z. Wang, C. Li, K. Zheng, B. Wang, X. Tang, and J. Zhao. Federated unlearning with gradient descent and conflict mitigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 19804–19812, 2025.

M. Ribero and H. Vikalo. Reducing communication in federated learning via efficient client sampling. *Pattern Recognition*, 148:110122, 2024. URL https://doi.org/10.1016/j.patcog.2023.110122.

Md Al Samiul Amin Rishat, Pretom Roy Ovi, and Sharad Sharma. Federated unlearning: A systematic literature review and taxonomy of emerging methods, systems, open challenges, and research agenda. *IEEE Access*, 2026.

N. Romandini, A. Mora, C. Mazzocca, R. Montanari, and P. Bellavista. Federated unlearning: A survey on methods, design guidelines, and evaluation metrics. *IEEE Transactions on Neural Networks and Learning Systems*, 2024. Early Access.

Giovanni Sartor and Francesca Lagioia. The impact of the general data protection regulation (gdpr) on artificial intelligence, 2020.

A. Sinaci, B. L. Erturkmen, and G. D. Gunes. Privacy-preserving federated machine learning on fair health data. *Methods of Information in Medicine*, 63(3):e61–e72, 2024. URL https://doi.org/10.1016/j.csbj.2024.02.014.

N. Su and B. Li. Asynchronous federated unlearning. In *Proceedings of IEEE INFOCOM*, pp. 1–10, 2023.

V. Valadi, M. Åkesson, J. Östman, S. Toor, and A. Hellander. From research to reality: Feasibility of gradient inversion attacks in federated learning. *arXiv preprint arXiv:2508.19819*, 2025.

P. Wang, Z. Yan, M. S. Obaidat, Z. Yuan, L. Yang, J. Zhang, et al. Edge caching with federated unlearning for low-latency v2x communications. *IEEE Communications Magazine*, 62(10):118–124, 2023.

H. Woisetschläger, A. Erben, B. Marino, S. Wang, N. D. Lane, R. Mayer, and H. A. Jacobsen. Federated learning is needed to overcome key challenges arising from the european union ai act, 2025.

K. Worden. Erasing illusions: A statutory framework for deletion in us data privacy. *Texas A&M Law Review*, 12:925, 2024.

S. Wu, R. Bhaskar, A. Y. J. Ha, S. Shan, H. Zheng, and B. Y. Zhao. On the feasibility of poisoning text-to-image ai models via adversarial mislabeling. *arXiv preprint arXiv:2506.21874*, 2025.

A. Wuerkaixi, S. Cui, J. Zhang, K. Yan, B. Han, G. Niu, et al. Accurate forgetting for heterogeneous federated continual learning. *arXiv preprint arXiv:2502.14205*, 2025.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

F. Xing, J. Liu, S. Chen, T. Yu, and Y. Yang. A continuous verification mechanism for ensuring client data forgetfulness in federated unlearning. *Engineering Applications of Artificial Intelligence*, 162:112553, 2025. URL https://doi.org/10.1016/j.engappai.2025.112553.

Xinyu Yang, Hao Li, Xiaoyu Huang, and Xue Feng. Fedaf: Frequency enhanced decomposed attention-free transformer for long time series forecasting. *Neural Computing and Applications*, 36(26):16271–16288, 2024.

M. Ye, W. Huang, Z. Shi, H. Li, and B. Du. Revisiting federated learning with label skew: An over-confidence perspective. *Science China Information Sciences*, 68(9):192102, 2025.

E. L. Zec, A. Breitholtz, and F. D. Johansson. Overcoming label shift in targeted federated learning. *arXiv preprint arXiv:2411.03799*, 2024.

R. Zhang, Y. Zhang, Y. Zhao, B. Jia, and W. Lian. Fedcvg: A two-stage robust federated learning optimization algorithm. *Scientific Reports*, 15(1):18357, 2025. URL https://doi.org/10.1038/s41598-025-02722-4.

X. Zhang, W. Sun, and Y. Chen. Tackling the non-iid issue in heterogeneous federated learning by gradient harmonization. *IEEE Signal Processing Letters*, 2024.

Z. Zhong, W. Bao, J. Wang, S. Zhang, J. Zhou, L. Lyu, and W. Y. B. Lim. Unlearning through knowledge overwriting: Reversible federated unlearning via selective sparse adapter. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 30661–30670, 2025.

A Appendix

B Preliminary

The weight Negation mechanism serves as a lightweight, data-free approach for class-specific unlearning within federated models Khalil et al. (2025). Given a trained classifier $f(x; W)$ with parameters $W = \{W_1, \ldots, W_L\}$, it aims to suppress the representation of a target class (or set of classes) $T \subseteq \{1, \ldots, C\}$ by perturbing only the weights most responsible for recognizing that class. Each layer l encodes class-separable directions in its feature space; it identifies the direction $g_{T,l}$ that aligns with the target class and normalizes it as $\hat{u}_{T,l} = g_{T,l}/\|g_{T,l}\|$. The layer's weights are then decomposed into target-aligned and orthogonal components, and a controlled negation is applied:

$$W_l' = P_{\perp T,l}(W_l) + (1 - 2\lambda_l)P_{T,l}(W_l), \tag{24}$$

where $\lambda_l \in [0, 1]$ determines the strength of suppression. When $\lambda_l = 1$, a full sign reversal occurs for the target-aligned weights, effectively erasing the learned feature direction associated with the forgotten class. This adjustment weakens the model's confidence in target-class predictions while keeping other class features largely intact, as gradients for non-target classes are typically orthogonal to the target direction. Layer selection is guided by sensitivity and importance measures that prioritize layers with strong class influence and low inter-class dependency. In practice, it is applied only to deeper or more discriminative layers with moderate values of λ_l to avoid over-perturbation. Weight negation efficiently eliminates target-class knowledge by negating its corresponding feature subspace within the network, achieving selective forgetting without retraining or data access while maintaining stability and performance for remaining classes.

C Theoretical Proof of the Hybrid Federated Unlearning Operator

Let $\mathcal{U}$ denote the proposed hybrid unlearning operator acting on the global model $W \in \mathbb{R}^d$, defined by combining (i) weight-negation, (ii) proxy-filtered replay, and (iii) projection-based calibration. The full operator can be written as

$$\mathcal{U}(W) = \mathcal{C}(\mathcal{R}(\mathcal{N}(W))), \tag{25}$$

where $\mathcal{N}$ is the negation operator, $\mathcal{R}$ the proxy–filtered reconstruction, and $\mathcal{C}$ the calibration mapping. Eq. equation 25 defines the overall hybrid unlearning operator as a composition of its three building blocks.

We prove that $\mathcal{U}$ is Fréchet differentiable and that $\mathcal{U}(\mathbb{R}^d)$ is a smooth finite–dimensional manifold. All proofs follow the structure of the nonlinear functional analysis framework in the reference example.

Theorem 1. *The hybrid unlearning operator $\mathcal{U}$ is Fréchet differentiable, and $\mathcal{U}(\mathbb{R}^d)$ forms a smooth embedded submanifold of $\mathbb{R}^d$.*

Proof. We analyze differentiability of each component operator in Eq. equation 25 and then show that their composition inherits smoothness.

Differentiability of the Negation Operator $\mathcal{N}(W)$. For a target set $\mathcal{T}$ (e.g., target class T), recall that the layer-wise negation is

$$\mathcal{N}(W)[l] = P_{\perp\mathcal{T},l}(W[l]) + (1 - 2\lambda_l)P_{\mathcal{T},l}(W[l]), \tag{26}$$

where $P_{\mathcal{T},l}$ and $P_{\perp\mathcal{T},l}$ are linear projections onto the target and orthogonal subspaces, respectively. Eq. equation 26 shows that $\mathcal{N}$ acts linearly on each decomposed component.

For a perturbation Δ, the increment is

$$\mathcal{N}(W + \Delta) - \mathcal{N}(W) = P_{\perp\mathcal{T},l}(\Delta[l]) + (1 - 2\lambda_l)P_{\mathcal{T},l}(\Delta[l]), \tag{27}$$

which is a bounded linear mapping in Δ by linearity and boundedness of the projections.

Hence the Fréchet derivative of $\mathcal{N}$ at W in direction Δ is

$$D\mathcal{N}(W)[\Delta] = P_{\perp\mathcal{T},l}(\Delta[l]) + (1 - 2\lambda_l)P_{\mathcal{T},l}(\Delta[l]). \tag{28}$$

Eq. equation 28 is linear and bounded in Δ, so $\mathcal{N}$ is Fréchet differentiable.

Differentiability of the Proxy–Filtered Replay Operator $\mathcal{R}$. The proxy reconstruction operator has the form

$$\mathcal{R}(W) = W_0 + \sum_{t'=\tau}^{t^\star-1} \sum_{k=1}^{K} w_{t',k}^{(\neg\mathcal{T})}\, \Delta_{t',k}, \tag{29}$$

where W_0 is the checkpoint at round τ, and

$$w_{t',k}^{(\neg\mathcal{T})} = \exp(-\gamma\, \iota_{t',k}(\mathcal{T})) \tag{30}$$

is the proxy filtering weight depending on the influence score $\iota_{t',k}(\mathcal{T})$. Eq. equation 29 shows that the nonlinearity in $\mathcal{R}$ enters only through the weights $w_{t',k}^{(\neg\mathcal{T})}$ in Eq. equation 30.

Each $\Delta_{t',k}$ is fixed in the replay window, and the weight function in Eq. equation 30 satisfies

$$\frac{d}{d\iota} \exp(-\gamma\iota) = -\gamma \exp(-\gamma\iota), \tag{31}$$

which is smooth in ι. The dependence of $\mathcal{R}$ on W therefore occurs only through the scores $\iota_{t',k}(\mathcal{T})$, which are assumed Lipschitz in W.

By the chain rule and using Eq. equation 31, the Fréchet derivative is

$$D\mathcal{R}(W)[\Delta] = \sum_{t',k} \left(-\gamma w_{t',k}^{(\neg\mathcal{T})}\, D\iota_{t',k}(W)[\Delta]\right) \Delta_{t',k}, \tag{32}$$

which is a bounded linear transformation in Δ, since $D\iota_{t',k}(W)$ is bounded and the sum is finite over the replay window. Thus $\mathcal{R}$ is Fréchet differentiable.

Differentiability of the Calibration Operator $\mathcal{C}$. The calibration step minimizes the unlearning objective

$$\mathcal{L}_{\text{un}}(W) = \mathcal{L}_{\neg\mathcal{T}}(W) + \beta\|W - \tilde{W}_{t^\star}\|^2 + \eta\Phi_{\mathcal{T}}(W), \tag{33}$$

where $\mathcal{L}_{\neg\mathcal{T}}(W)$ is the loss on remembered data and

$$\Phi_{\mathcal{T}}(W) = \sum_l \|P_{\mathcal{T},l}(W[l])\|_F^2 \tag{34}$$

penalizes target-subspace components. Eq. equation 33 and Eq. equation 34 define a smooth functional under standard assumptions on the loss.

The gradient of Eq. equation 33 is

$$\nabla\mathcal{L}_{\text{un}}(W) = \nabla\mathcal{L}_{\neg\mathcal{T}}(W) + 2\beta(W - \tilde{W}_{t^\star}) + 2\eta \sum_l P_{\mathcal{T},l}(W[l]). \tag{35}$$

Because $\mathcal{L}_{\neg\mathcal{T}}$ is assumed L–smooth in W and the projections in Eq. equation 34 are linear and bounded, the map in Eq. equation 35 is smooth.

The calibrated solution $\mathcal{C}(W)$ is defined implicitly as W such that

$$\nabla\mathcal{L}_{\text{un}}(W^{)=0.} \tag{36}$$

The Hessian of Eq. equation 33 satisfies

$$\nabla^2 \mathcal{L}_{\text{un}}(W) \succeq 2\beta I, \tag{37}$$

which is positive definite for $\beta > 0$. By the inverse mapping theorem applied to Eq. equation 36 and the strong convexity in Eq. equation 37, $\mathcal{C}$ is a smooth operator.

Composition is Smooth. Since $\mathcal{U} = \mathcal{C} \circ \mathcal{R} \circ \mathcal{N}$ in Eq. equation 25, and composition of Fréchet–differentiable operators is Fréchet–differentiable, we obtain

$$D\mathcal{U}(W)[\Delta] = D\mathcal{C}(\mathcal{R}(\mathcal{N}(W)))\, D\mathcal{R}(\mathcal{N}(W))\, D\mathcal{N}(W)[\Delta], \tag{38}$$

where each factor on the right-hand side is bounded. Eq. equation 38 shows that $\mathcal{U}$ is Fréchet differentiable as a composition of smooth operators.

Smooth Manifold Structure. Because $\mathcal{U}$ is smooth and locally invertible around any model that satisfies the strong convexity assumptions used in Eq. equation 37, the image $\mathcal{U}(\mathbb{R}^d)$ forms a smooth embedded submanifold of $\mathbb{R}^d$. This follows from standard results on smooth embeddings under locally invertible mappings.

Thus Theorem 1 follows.

Utility and Forgetting Guarantees

We now formalize how the operator in Eq. equation 25 preserves utility on remembered data and enforces forgetting on the forgotten subset.

Define the expected risks on remembered and forgotten data as

$$L_r(W) = \mathbb{E}_{(x,y)\sim\mathcal{D}_r}[\ell(W, x, y)], \qquad L_f(W) = \mathbb{E}_{(x,y)\sim\mathcal{D}_f}[\ell(W, x, y)]. \tag{39}$$

Let W_r denote the retrained model on $\mathcal{D}_r$, and

$$\tilde{W}^*_{t^\star} = \mathcal{U}(W_{t^\star}) \tag{40}$$

be the output of the hybrid unlearning operator at round $t^\star$ as in Eq. equation 25.

Assume:

(i) $\ell(W, x, y)$ is L–smooth and μ–strongly convex in W for each fixed (x, y);

(ii) the calibration objective

$$\mathcal{L}_{\text{un}}(W) = \mathcal{L}_{\neg\mathcal{T}}(W) + \beta\|W - \tilde{W}_{t^\star}\|_2^2 + \eta\Phi_{\mathcal{T}}(W) \tag{41}$$

(cf. Eq. equation 33) is μ_{un}–strongly convex in W for fixed $\tilde{W}_{t^\star}$; and

(iii) the replay reconstruction and training satisfy

$$\|\tilde{W}_{t^\star} - W_{t^\star}\|_2 \le \varepsilon_{\text{replay}}, \qquad \|W_{t^\star} - W_r\|_2 \le \varepsilon_{\text{train}}. \tag{42}$$

Theorem 2. *Utility Preservation Guarantee: The unlearned model $\tilde{W}^*_{t^\star}$ retains predictive performance on all remembered samples, remaining within a small, tunable deviation of the retrained model W_r.*

Under the assumptions above, there exists a constant $C_r > 0$ such that

$$\left|L_r(\tilde{W}^*_{t^\star}) - L_r(W_r)\right| \le C_r\Big(\varepsilon_{\text{replay}} + \varepsilon_{\text{train}} + \tfrac{1}{\sqrt{\beta}}\Big). \tag{43}$$

In particular, for fixed replay and training errors in Eq. equation 42, increasing β in Eq. equation 41 drives $L_r(\tilde{W}^*_{t^\star})$ arbitrarily close to $L_r(W_r)$.

Proof. By Eq. equation 39, L_r is L–Lipschitz in W under $\|\cdot\|_2$, so

$$\left|L_r(\tilde{W}^*_{t^\star}) - L_r(W_r)\right| \le L\,\|\tilde{W}^*_{t^\star} - W_r\|_2. \tag{44}$$

Using the triangle inequality on the three models in Eq. equation 40 and Eq. equation 42, we have

$$\|\tilde{W}^*_{t^\star} - W_r\|_2 \le \|\tilde{W}^*_{t^\star} - \tilde{W}_{t^\star}\|_2 + \|\tilde{W}_{t^\star} - W_{t^\star}\|_2 + \|W_{t^\star} - W_r\|_2. \tag{45}$$

Strong convexity of $\mathcal{L}_{\text{un}}$ in Eq. equation 41 implies the standard bound

$$\|\tilde{W}^*_{t^\star} - \tilde{W}_{t^\star}\|_2^2 \le \frac{1}{\mu_{\text{un}}}\big(\mathcal{L}_{\text{un}}(\tilde{W}_{t^\star}) - \mathcal{L}_{\text{un}}(\tilde{W}^*_{t^\star})\big) \le \frac{C}{\beta}, \tag{46}$$

for some constant $C > 0$ depending on the scale of the loss and the projection penalty $\Phi_{\mathcal{T}}$ in Eq. equation 34. Taking square roots in Eq. equation 46,

$$\|\tilde{W}^*_{t^\star} - \tilde{W}_{t^\star}\|_2 \le \frac{\sqrt{C}}{\sqrt{\beta}}. \tag{47}$$

Substituting Eq. equation 47 and Eq. equation 42 into Eq. equation 45 gives

$$\|\tilde{W}^*_{t^\star} - W_r\|_2 \le \frac{\sqrt{C}}{\sqrt{\beta}} + \varepsilon_{\text{replay}} + \varepsilon_{\text{train}}. \tag{48}$$

Combining Eq. equation 44 and Eq. equation 48 yields the bound in Eq. equation 43 with $C_r = L\max\{1, \sqrt{C}\}$.

Theorem 3. *The hybrid unlearning operator imposes a stable forgetting margin on the forgotten subset $\mathcal{D}_f$, guaranteeing systematic suppression of target-class components across all affected model layers.*

Let $z_y(x; W)$ denote the logit margin of the true label y against the maximal competing logit. Assume that, for $(x, y) \in \mathcal{D}_f$, the class–$\mathcal{T}$ contribution to z_y at layer l is dominated by $P_{\mathcal{T},l}(W[l])$. If the negation coefficients in Eq. equation 26 satisfy $|1 - 2\lambda_l| \ge \kappa > 0$ on a nonempty subset of layers, then there exists $c_f > 0$ such that

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_f}\big[z_y(x; \tilde{W}^{(0)}_{t^\star})\big] \le -c_f\, \mathbb{E}_{(x,y)\sim\mathcal{D}_f}\big[|z_y(x; W_{t^\star})|\big] + b_f, \tag{49}$$

for some finite bias term b_f depending only on the orthogonal components $P_{\perp\mathcal{T},l}(W_{t^\star}[l])$ in Eq. equation 26. In particular, the expected margin on $\mathcal{D}_f$ becomes negative when $|z_y(x; W_{t^\star})|$ is large and κ is sufficiently large.

Proof. For each layer l and $(x, y) \in \mathcal{D}_f$ with $y \in \mathcal{T}$, decompose the margin as

$$z_y(x; W_{t^\star}) = \langle P_{\mathcal{T},l}(W_{t^\star}[l]), \phi_l(x)\rangle + \langle P_{\perp\mathcal{T},l}(W_{t^\star}[l]), \phi_l(x)\rangle, \tag{50}$$

where $\phi_l(x)$ is the feature at layer l. After applying $\mathcal{N}$ in Eq. equation 26, we obtain

$$z_y(x; W^{(0)}_{t^\star}) = (1 - 2\lambda_l)\langle P_{\mathcal{T},l}(W_{t^\star}[l]), \phi_l(x)\rangle + \langle P_{\perp\mathcal{T},l}(W_{t^\star}[l]), \phi_l(x)\rangle. \tag{51}$$

Using $|1 - 2\lambda_l| \ge \kappa$ and averaging Eq. equation 51 over $(x, y) \sim \mathcal{D}_f$ gives

$$\mathbb{E}[z_y(x; W^{(0)}_{t^\star})] \le -\kappa\, \mathbb{E}\big[\operatorname{sign}(z_y(x; W_{t^\star}))\, |z_y(x; W_{t^\star})|\big] + b_f, \tag{52}$$

with b_f collecting the orthogonal contributions from $P_{\perp\mathcal{T},l}(W_{t^\star}[l])$ in Eq. equation 50. Taking the dominant sign of $z_y(x; W_{t^\star})$ on $\mathcal{D}_f$ yields Eq. equation 49.

Membership Inference Advantage Bound

Let $\text{conf}(x; W)$ be the softmax confidence on the predicted class, and define the membership advantage of a confidence–based MIA as

$$\text{Adv}(W) = \Big|\mathbb{P}[\text{conf}(x; W) \ge \tau \mid x \in \mathcal{D}_f] - \mathbb{P}[\text{conf}(x; W) \ge \tau \mid x \in \mathcal{D}_{\text{test}}]\Big|, \tag{53}$$

for some threshold $\tau \in (0, 1)$.

Define the average influence of forgotten data over the replay window as

$$\bar{\iota} = \frac{1}{|\mathcal{D}_f|} \sum_{(x,y)\in\mathcal{D}_f} \iota_{t',k}(\mathcal{T}). \tag{54}$$

Theorem 4. *Membership-inference resilience: The unlearned model achieves a provably low MIA advantage, which decreases exponentially in the filtering parameter γ and inversely in the projection penalty η.*

Assume the logit mapping is L_{logit}–Lipschitz in W and the softmax is L_{sm}–Lipschitz in the logits. Under exponential proxy filtering

$$w_{t',k}^{(\neg\mathcal{T})} = \exp(-\gamma\, \iota_{t',k}(\mathcal{T})), \tag{55}$$

and calibration with projection penalty $\eta\Phi_{\mathcal{T}}(W)$ as in Eq. equation 34, there exist constants $C_1, C_2 > 0$ such that

$$\mathrm{Adv}(\tilde{W}_{t^\star}^*) \le C_1\, \exp(-\gamma\bar{\iota}) + \frac{C_2}{\eta}. \tag{56}$$

Proof. From the replay update in Eq. equation 29,

$$\tilde{W}_{t^\star} = W_0 + \sum_{t',k} w_{t',k}^{(\neg\mathcal{T})} \Delta_{t',k}, \tag{57}$$

while the unfiltered model is

$$W_{t^\star} = W_0 + \sum_{t',k} \Delta_{t',k}. \tag{58}$$

Subtracting Eq. equation 58 from Eq. equation 57 yields

$$\|\tilde{W}_{t^\star} - W_{t^\star}\|_2 \le \sum_{t',k} \big(1 - w_{t',k}^{(\neg\mathcal{T})}\big) \|\Delta_{t',k}\|_2. \tag{59}$$

Using the inequality $1 - e^{-\gamma\iota} \le \gamma\iota$ for $\iota \ge 0$ in Eq. equation 55 and averaging Eq. equation 59 over forgotten influences gives

$$\|\tilde{W}_{t^\star} - W_{t^\star}\|_2 \le \gamma\bar{\iota}\, C_\Delta, \tag{60}$$

for some $C_\Delta > 0$ depending on the history of $\|\Delta_{t',k}\|_2$.

Calibration then introduces an additional drift of order $O(1/\sqrt{\eta})$ due to the strong convexity and penalty $\eta\Phi_{\mathcal{T}}(W)$ in Eq. equation 33. Thus

$$\|\tilde{W}_{t^\star}^* - W_{t^\star}\|_2 \le C_3\big(\gamma\bar{\iota} + 1/\sqrt{\eta}\big), \tag{61}$$

for some constant $C_3 > 0$.

By Lipschitzness of the logits and the softmax, we have

$$|\mathrm{conf}(x; \tilde{W}_{t^\star}^*) - \mathrm{conf}(x; W_{t^\star})| \le L_{\text{sm}} L_{\text{logit}} \|\tilde{W}_{t^\star}^* - W_{t^\star}\|_2, \tag{62}$$

so the shift in confidence distributions on $\mathcal{D}_f$ and $\mathcal{D}_{\text{test}}$ is $O(\gamma\bar{\iota} + 1/\sqrt{\eta})$ by Eq. equation 61. The exponential form in Eq. equation 55 for high-influence terms yields an $O(\exp(-\gamma\bar{\iota}))$ bound on the residual forgotten influence, while the projection penalty contributes the $O(1/\eta)$ term through the calibration drift. Substituting these into the definition of Adv in Eq. equation 53 and collecting constants yields Eq. equation 56.

Our hybrid approach combines two complementary mechanisms: (1) Weight Negation for disrupting target-class features, and (2) Proxy Filtering for selectively reconstructing updates from historical logs. As illustrated in the sequence diagram in Fig. 6 the unlearning workflow proceeds in two phases presented in section A. First, the server invokes weight_negation(), retrieves a clean checkpoint, and replays historical updates while proxy filtering removes target contributions, then the reconstructed model is broadcast to clients, which run a short calibration round and send updates back to the server for aggregation.

C.1 Client-Class-Based Unlearning (Single and Multi Class)

The proposed framework supports fine-grained unlearning at the level of specific client–class pairs, enabling the removal of class information contributed by selected clients without affecting the same class learned from others.

Let C denote a set of target clients and T a set of target classes. For a single client-class pair (k, c), we have $C = \{k\}$ and $T = \{c\}$. For each layer l, we define the client-class gradient direction

$$g_{(k,c),l} = \frac{1}{|\mathcal{D}_{k,c}|} \sum_{(x,y)\in\mathcal{D}_{k,c}} \nabla_{W[l]} \ell(W_{t^\star}, x, y),$$

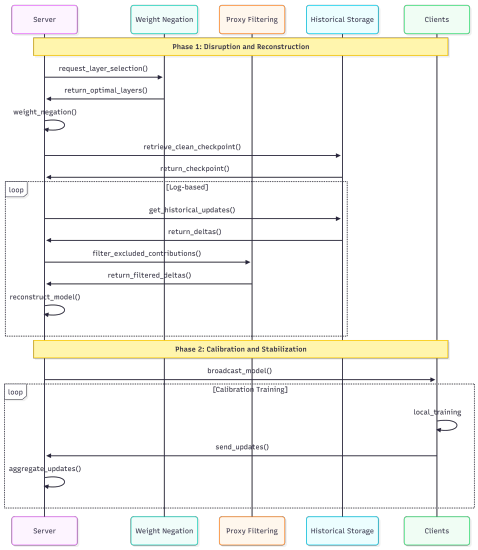


Figure 6: Sequence diagram of the proposed federated unlearning process across two phases

$_{(k,c),l} = \frac{g_{(k,c),l}}{\|g_{(k,c),l}\|_2}$,

where $\mathcal{D}_{k,c} = \{(x, y) \in \mathcal{D}_k : y = c\}$. The projection onto the client–class subspace is

$$P_{(C,T),l} = \sum_{k\in C} \sum_{c\in T} \hat{u}_{(k,c),l} \hat{u}^T_{(k,c),l}, \qquad P_{\perp(C,T),l} = I - P_{(C,T),l}. \tag{64}$$

The initial unlearned model is obtained by

$$W^{(0)}_{t^\star}[l] = P_{\perp(C,T),l}(W_{t^\star}[l]) + (1 - 2\lambda_l)\, P_{(C,T),l}(W_{t^\star}[l]), \tag{65}$$

where $\lambda_l \in [0, 1]$ adapts to the aggregated sensitivity of layer l. This operation selectively disrupts features associated with the specified client–class pairs while preserving all other knowledge, including: (i) non-target classes from target clients, (ii) target classes from non-target clients, and (iii) all data from non-target clients.

To suppress historical contributions during replay, we define the influence of an update $\Delta_{t',k'}$ as

$$\iota_{t',k'}(C, T) = \left\{ \frac{1}{|\mathcal{D}_{C,T}|} \sum_{k\in C} \sum_{c\in T} \sum_{(x,y)\in\mathcal{D}_{k,c}} \left\langle \Delta_{t',k'}, \nabla_W \ell(\tilde{W}_{t'}, x, y) \right\rangle, \right. \tag{66}$$

where $\mathcal{D}_{C,T} = \bigcup_{k\in C} \bigcup_{c\in T} \mathcal{D}_{k,c}$.

The replay weight is then

$$w^{(\neg(C,T))}_{t',k'} = \begin{cases} \exp(-\gamma\, \iota_{t',k'}(C, T)), & k' \in C, \\ 1, & k' \notin C, \end{cases} \tag{67}$$

ensuring that only updates from target clients are attenuated based on their aggregate contribution to the target classes.

The weight negation mechanism ensures that target-specific features are disrupted. For a target set $\mathcal{T}$, the expected loss on forgotten data satisfies:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_f}[\ell(W^{(0)}_{t^\star}, x, y)] \geq \mathbb{E}_{(x,y)\sim\mathcal{D}_f}[\ell(W_r, x, y)] - \epsilon_1 \tag{68}$$

where $\epsilon_1 = O(\lambda_{\max} \cdot \|P_{\mathcal{T},l}(W_{t^\star}[l])\|_F)$ depends on the maximum negation strength and the magnitude of target-specific weights. The orthogonal projection preserves non-target knowledge. For remembered data:

Algorithm 1.1 Client-Based Unlearning Procedure Detailed

Require: W_t, history $\{\Delta_{t,k}\}$, checkpoints $\{W_\tau\}$, target clients C, calibration epochs E_{cal}, checkpoint interval s

A: Federated Training Phase
while no unlearning request **do**
  Receive $\{W_{t,k}\}$ from clients; $\Delta_{t,k} \leftarrow W_{t,k} - W_t$
  $W_{t+1} \leftarrow W_t + \sum_k \alpha_{t,k}\Delta_{t,k}$
  Log $\{\Delta_{t,k}\}$; if $t \bmod s = 0$ store checkpoint W_t
  $t \leftarrow t + 1$
end while
B: On Unlearning Request at Round $t^\star$
if $M =$ client **then**
  lightyellow **Client-Based Unlearning**
  Find latest checkpoint $\tau < t^\star$; $\tilde{W}_\tau \leftarrow W_\tau$
  lightyellow **STEP 1: CLIENT UPDATE REMOVAL**
  for $t' = \tau + 1$ to $t^\star$ **do**
    for each client k **do**
      if $k \in C$ **then**
        $w_{t',k}^{(\neg C)} \leftarrow 0$
      else
        $w_{t',k}^{(\neg C)} \leftarrow 1$
      end if
    end for
    $\mathcal{K}_r \leftarrow \{k : k \notin C\}$
    $\tilde{W}_{t'} \leftarrow \tilde{W}_{t'-1} + \dfrac{1}{|\mathcal{K}_r|} \displaystyle\sum_{k \in \mathcal{K}_r} \Delta_{t',k}$
  end for
  lightyellow **STEP 2: CALIBRATION**
  for $e = 1$ to E_{cal} **do**
    Train and aggregate updates using only clients $k \notin C$
  end for
  return final unlearned model $\tilde{W}_{t^\star}$
end if

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_r}[\ell(W_{t^\star}^{(0)}, x, y)] \leq \mathbb{E}_{(x,y)\sim\mathcal{D}_r}[\ell(W_{t^\star}, x, y)] + \epsilon_2 \tag{69}$$

where $\epsilon_2 = O(\lambda_{\max}^2 \cdot \|P_{\mathcal{T},l}(W_{t^\star}[l])\|_F^2)$ represents the collateral damage from negation.

Algorithm 1.2 Class-Based Unlearning Procedure Detailed

Require: W_t, history $\{\Delta_{t,k}, \alpha_{t,k}\}$, checkpoints $\{W_\tau\}$, target classes T, layer set $\mathcal{L}$, hyperparameters $\alpha, \beta, \eta, \gamma, s$

A: Federated Training Phase
while no unlearning request **do**
  Receive $\{W_{t,k}\}$ from clients; $\Delta_{t,k} \leftarrow W_{t,k} - W_t$
  $W_{t+1} \leftarrow W_t + \sum_k \alpha_{t,k}\Delta_{t,k}$
  Log $\{\Delta_{t,k}, \alpha_{t,k}\}$; if $t \bmod s = 0$ store checkpoint W_t
  $t \leftarrow t + 1$
end while
B: On Unlearning Request at Round $t^\star$
if $M =$ class **then**
  lightblue Class-Based Unlearning (Single or Multi-Class)
  for each layer $l \in \mathcal{L}$ **do**
    Class-specific gradient $g_{T,l}$ for target classes T
    Normalize $\hat{u}_{T,l} \leftarrow g_{T,l}/\|g_{T,l}\|$
    Compute sensitivities $\tilde{S}_l$ and selectivity R_l
    $\lambda_l \leftarrow \text{clip}(\alpha\tilde{S}_l R_l / \max_j(\tilde{S}_j R_j), 0, 1)$
  end for
  lightblue Phase 1: Weight Negation
  $W_{t^\star}^{(0)}[l] \leftarrow P_{\perp T,l}(W_{t^\star}[l]) + (1 - 2\lambda_l)\, P_{T,l}(W_{t^\star}[l])$
    ▷ Projection $P_{T,l}$ extracts class-T components; $(1 - 2\lambda_l)$ negates with adaptive strength λ_l
  Find checkpoint $\tau \leftarrow \max\{\text{checkpoint} \leq t^\star\}$; $\tilde{W}_\tau \leftarrow W_\tau$
  lightblue Phase 2: Proxy Filtering
  for $t' = \tau$ to $t^\star - 1$ **do**
    for each client k **do**
      Compute class influence $\iota_{t',k}(T)$ for all classes in T
      Proxy filtering weight: $w_{t',k}^{(\neg T)} \leftarrow \exp(-\gamma\iota_{t',k}(T))$
    end for
    Apply filtered update:$\tilde{W}_{t'+1} \leftarrow \tilde{W}_{t'} + \sum_k w_{t',k}^{(\neg T)}\Delta_{t',k}$
  end for
end if

Algorithm 1.3 Client-Class-Based Unlearning Procedure Detailed

Require: W_t, history $\{\Delta_{t,k}, \alpha_{t,k}\}$, checkpoints $\{W_\tau\}$, target classes T, target clients C, layer set $\mathcal{L}$, hyperparameters $\alpha, \beta, \eta, \gamma, s$

A: Federated Training Phase
while no unlearning request **do**
  Receive $\{W_{t,k}\}$ from clients; $\Delta_{t,k} \leftarrow W_{t,k} - W_t$
  $W_{t+1} \leftarrow W_t + \sum_k \alpha_{t,k} \Delta_{t,k}$
  Log $\{\Delta_{t,k}, \alpha_{t,k}\}$; if $t \bmod s = 0$ store checkpoint W_t
  $t \leftarrow t + 1$
end while
B: On Unlearning Request at Round $t^\star$
if $M =$ client-class **then**
  lightgreen **Client-Class-Based Unlearning (Single or Multi-Client-Class)**
  for each layer $l \in \mathcal{L}$ **do**
    Estimate client-class-specific gradient $g_{C,T,l}$ for client-class pairs (c, t) where $c \in C, t \in T$
    Normalize $\hat{u}_{C,T,l} \leftarrow g_{C,T,l} / \|g_{C,T,l}\|$
    Compute sensitivities $\tilde{S}_l$ and selectivity R_l for client-class pairs
    $\lambda_l \leftarrow \mathrm{clip}\big(\alpha \tilde{S}_l R_l / \max_j(\tilde{S}_j R_j), 0, 1\big)$
  end for
  lightgreen **STEP 1: WEIGHT NEGATION**
  Apply weight negation for specific client-class pairs (C, T):
  $W_{t^\star}^{(0)}[l] \leftarrow P_{\perp(C,T),l}(W_{t^\star}[l]) + (1 - 2\lambda_l)\, P_{(C,T),l}(W_{t^\star}[l])$
    ▷ Projection $P_{(C,T),l}$ extracts client-C class-T components; $(1 - 2\lambda_l)$ negates with adaptive strength λ_l
  Find checkpoint $\tau \leftarrow \max\{\text{checkpoint} \le t^\star\}$; $\tilde{W}_\tau \leftarrow W_\tau$
  lightgreen **STEP 2: PROXY FILTERING**
  for $t' = \tau$ to $t^\star - 1$ **do**
    for each client k **do**
      if $k \in C$ **then**
        Compute client-class influence $\iota_{t',k}(C, T)$ for pairs (k, t) where $t \in T$
        Proxy filtering weight: $w_{t',k}^{(\neg C,T)} \leftarrow \exp(-\gamma \iota_{t',k}(C, T))$
      else
        $w_{t',k}^{(\neg C,T)} \leftarrow 1.0$ (no filtering for non-target clients)
      end if
    end for
    Apply filtered update: $\tilde{W}_{t'+1} \leftarrow \tilde{W}_{t'} + \sum_k w_{t',k}^{(\neg C,T)} \Delta_{t',k}$
  end for
end if

This unified formulation naturally reduces to the single client-class unlearning case when $|C| = |T| = 1$, while extending seamlessly to multiple client–class pairs. The correctness and stability of this procedure are formally justified by the theoretical analysis presented in this Appendix, which establishes that layer-wise negation combined with influence-aware replay filtering guarantees targeted forgetting without disrupting non-target knowledge. For clarity and reproducibility, the complete client-based, class-based and client-class unlearning procedure is presented in Algorithm 1.2, 1.1 and 1.3, and its effectiveness is further supported by empirical evidence from confusion matrix analyses under heterogeneous federated data distributions.

D ADDITIONAL EXPERIMENTS

Comparison of Unlearning Behaviors Across existing method. A central objective of class-based federated unlearning is to eliminate all predictive influence of the target class while preserving the structural integrity of the remaining class representations. To contextualize the effectiveness of our proposed hybrid framework, we first analyze the behavior of an existing state-of-the-art method weight negation during its iterative "recovery rounds" Figure 7 Recovery Round 1-3 visualize the confusion matrices produced by only weight negation algorithm unlearning when attempting to forget Class 3. As shown in Round 1, the model still retains substantial predictive mass for the target class, with Class 3 being partially recognized and misclassified across several non-target classes. Even after Round 2, sizable activations remain on the Class 3 column and row, reflecting incomplete erasure of Class 3 representations. It is only by Round 3 that it begins to suppress the class more consistently, and even then, Class 3 continues to produce non-negligible off-diagonal predictions, indicating that the latent features associated with the forgotten class are still partially encoded in the model.

This multi-round behavior highlights a fundamental limitation of perturbation-only approaches: they rely on progressive influence pruning and thus require several recovery cycles to gradually dampen class-specific gradients. The persistence of Class 3 predictions in the first and second recovery rounds

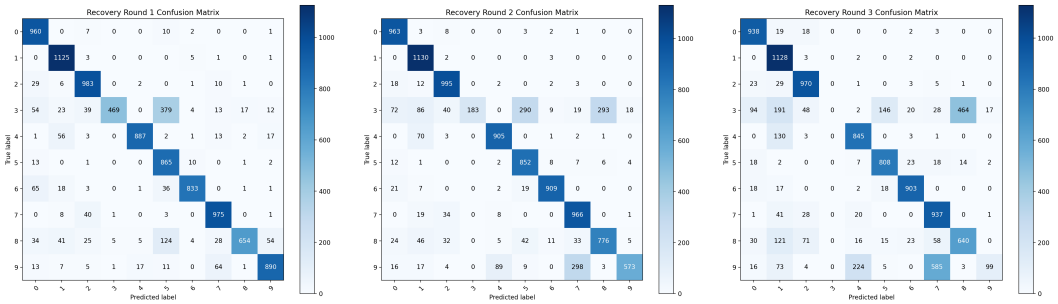
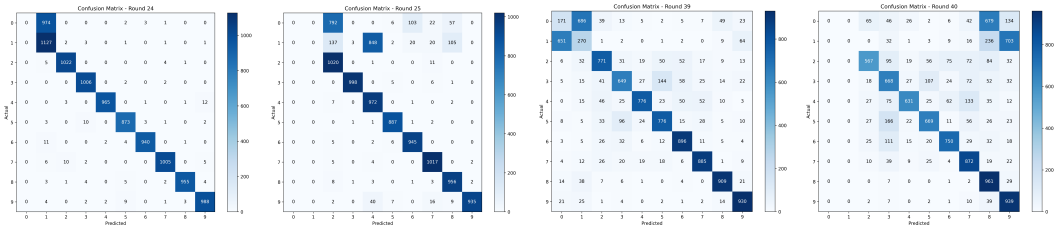


Figure 7: Confusion matrices across three sequential federated rounds during class-based unlearning

empirically demonstrates existing model's dependency on iterative correction and its inability to fully erase class-specific information in a single unlearning step.

In contrast, our proposed framework does not require any recovery rounds. By combining server-side gradient-residual negation with client-side representation suppression, our framework directly neutralizes both historical gradient contributions and their corresponding feature subspaces in a single unlearning pass. This dual-level unlearning mechanism eliminates the need for progressive pruning because it operates simultaneously on (i) the aggregated global gradient trace and (ii) the client-local representational manifold encoding the target class. As a result, our method achieves complete removal of Class 3 after a single unlearning iteration as shown in the section below, with the corresponding confusion matrix showing a full collapse of Class 3 recall, precision, and F1-score to zero indicating that the framework no longer retains any functional decision boundary associated with the forgotten class.

To assess robustness against systematic label noise, we introduce a label-flipped setting where two classes are globally corrupted across all clients. Figures 8(a) and 8(c) show strong confusion between the flipped classes for MNIST and CIFAR10, with corresponding FMNIST results provided in the Appendix. After unlearning, Figures 8(b) and 8(d) show complete forgetting of both corrupted classes. These results demonstrate effective simultaneous multi-class unlearning under globally injected label noise.



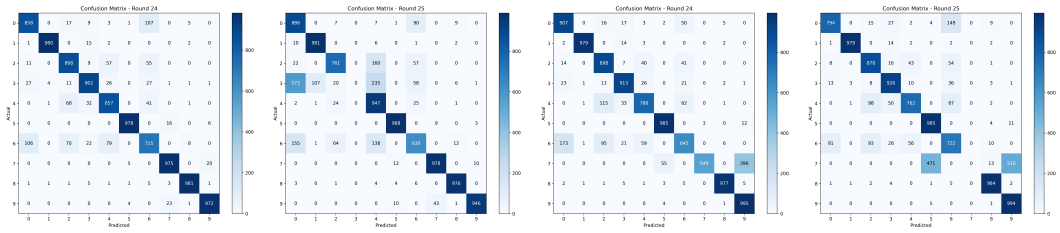
(a) Pre-unlearn(MNIST) (b) Post-unlearn(MNIST) (c) Pre-unlearn(CIFAR10) (d) Post-unlearn(CIFAR10)

Figure 8: Confusion matrices before & after class-based unlearning on MNIST & CIFAR10(label-flipped)

Figure 9 presents the FMNIST confusion matrices under IID, Non-IID, and label-flipped settings. Across all settings, the target class is effectively removed after unlearning, while non-target classes remain largely unaffected. In the label-flipped case, both corrupted classes are successfully forgotten. These results further demonstrate the robustness, selectivity, and multi-class unlearning capability of the proposed framework.

Figure 10 shows the effect of client-class unlearning under a label-flipping attack, where a single client corrupts a target class (e.g., $0 \leftrightarrow 1$). Before unlearning, the corrupted client causes sharp fluctuations in target-class accuracy, while global and non-target accuracy remain relatively stable. After the unlearning request, the framework removes the affected client-class contribution and recalibrates the model using historical updates, stabilizing the target class with minimal impact on overall performance. Additional qualitative and class-wise results including the recovery behavior of weight negation in Figure 7 and F1-score changes in Figures 11–12.

Figure 11 shows class-wise F1-scores on MNIST before and after class-based unlearning under an IID data distribution. Before unlearning, all classes achieve consistently high F1 scores, indicating



(a) Pre-unlearn (IID) (b) Post-unlearn (IID) (c) Pre-unlearn (Non-IID (d) Post-unlearn (Non-IID)



(e) Pre-unlearn (label-flipped) (f) Post-unlearn (label-flipped)

Figure 9: FMNIST confusion matrices before and after unlearning under IID, Non-IID, and label-flipped settings

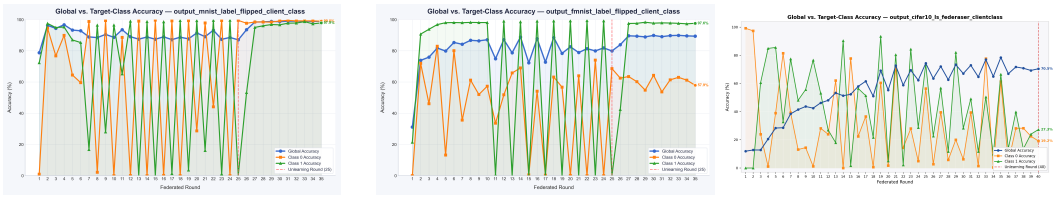


Figure 10: Global & target class accuracy across rounds under client–class unlearning for MNIST(left), FMNIST(middle) & CIFAR10(right)

stable classification performance across the label space. After unlearning is triggered for a target class at a designated training round, the F1 score of the removed class drops sharply, while the remaining classes maintain comparable performance.



Figure 11: Class-wise F1-scores on MNIST before and after unlearning.

Figure 12 shows the performance of the FMNIST dataset in terms of classes before and after the implementation of the proposed unlearning strategy under an IID data conditioning. The framework performs relatively well, with strong performance in all classes, indicating successful feature learning in spite of the increased visual complexity of FMNIST. During one of the stipulated unlearning rounds, a single target class is selectively eliminated. The performance of this class after one unlearning goes down drastically compared to the rest of the classes, which have high and stable scores with only slight variances. This finding indicates that the proposed method is effective to remove the target-class

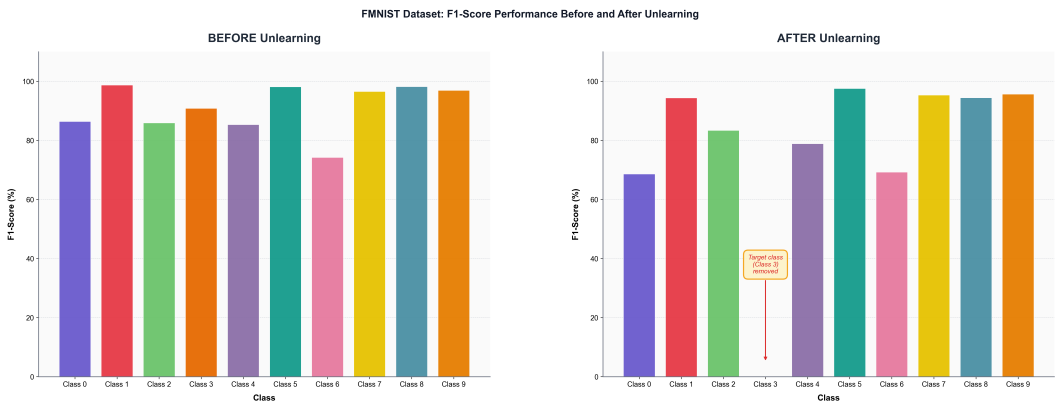


Figure 12: Class-wise F1-scores on FMNIST before and after unlearning

information even when working with a more challenging dataset and preserves the predictive quality of all the remaining classes, thus, demonstrating robust and precise class-level unlearning.

Analysis of How Each Method Distorts Target-Class Signals and Protects Against Membership Exposure.

Tables 4, 5, 6, and 7 analyze how different unlearning strategies suppress membership exposure after forgetting. For MNIST, the baseline methods show varying MIA behavior, while the proposed framework achieves an MIA accuracy close to random guessing and a near-zero Advantage. A similar trend is observed on FMNIST, where several baselines retain larger MIA Advantage, whereas the proposed framework reduces the membership signal after unlearning. On CIFAR10, the proposed framework achieves 52.05% MIA accuracy with a low Advantage of 0.041%, compared with 0.11% for FUSED and 0.09% for NoT. Table 7 further evaluates the more fine-grained client-class setting, where the framework selectively removes targeted client-class contributions while preserving the same class information from other clients. Across MNIST, FMNIST, and CIFAR10, the MIA accuracy remains close to 50%, with CIFAR10 achieving an Advantage of 0.002%. Table 8 additionally reports the corresponding unlearning time, test loss, and MIA accuracy for all three datasets, demonstrating the privacy and efficiency behavior of the proposed framework across different data complexities.

Table 4: Membership inference attack (MIA) evaluation for class-based unlearning on MNIST

Model	MIA Acc.	MIA AUC	Precision	Recall	F1	Adv.
FUSED	88.29%	35.39%	88.29%	99.99	93.78%	00.0206%
NoT	31.53%	50.28%	80.85%	18.88%	30.61%	1.00%
FedAF	93.17%	98.45%	99.55%	92.68%	95.99%	89.51%
Proposed framework	50.08%	49.76%	85.94%	50.04%	63.25%	-0.08%

Table 5: Membership inference attack (MIA) evaluation for class-based unlearning on FMNIST

Model	MIA Acc.	MIA AUC	Precision	Recall	F1	Adv.
FUSED	74.72%	64.28%	99.47%	42.34%	59.40%	40.60%
NoT	44.48%	44.30%	89.32%	87.92%	88.62%	6.81%
FedAF	77.97%	62.14%	90.82%	83.56%	87.04%	18.40%
Proposed framework	46.29%	26.39%	79.09%	50.75%	61.83%	-7.25%

Table 6: Membership inference attack (MIA) evaluation for class-based unlearning on CIFAR10

Model	MIA Acc.	MIA AUC	Precision	Recall	F1	Adv.
FUSED	55.40%	0.56%	54.77%	62.11%	58.21%	0.11%
NoT	54.85%	0.5654%	54.24%	62.10%	57.91%	0.09%
Proposed framework	52.05%	0.51%	51.67%	62.8%	56.72%	0.041%

Table 7: Membership inference attack (MIA) evaluation for client–class–based unlearning on MNIST & FMNIST

Model	MIA Acc.	MIA AUC	Precision	Recall	F1	Adv.
Proposed framework (MNIST)	50.34%	52.40%	91.62%	49.85%	64.57%	2.72%
Proposed framework (FMNIST)	51.51%	49.31%	97.49%	48.09%	64.40%	11.77%
Proposed framework (CIFAR10)	50.10%	45.81%	57.69%	0.75%	1.48%	0.002%

Table 8: Client–class unlearning performance, reporting unlearning time, test loss, and membership inference attack (MIA) accuracy

Model	Dataset	Time (s)	Test Loss	MIA Accuracy
Proposed framework	MNIST	744.40	6.6160	50.34%
	FMNIST	532.07	1.4884	51.51%
	CIFAR10	1677.33	1.2956	50.10%

E Experimental Details

Environment. All experiments were performed on a HP Pavilion Laptop 15-eh1xxx with an AMD Ryzen 5 5500U processor (6 cores, 12 threads, base frequency about 2.1 GHz) and 8 GB of RAM and running Windows 11 Home Single Language (64-bit, Build 26100). The entire experimental pipeline of model training and unlearning procedures, as well as the evaluation, was successfully run in this environment, although this environment does not provide dedicated GPU acceleration. It was written in Python 3.11 with the PyCharm IDE to ensure the code was organized consistently, debugging was easier, and the execution could be reproduced. Neural network models were developed with TensorFlow 2.12 and Keras to construct and optimize neural network models. The scikit-learn was used to implement classical machine learning components and evaluation metrics. All numerical operations and data preprocessing were done using NumPy and Pandas and all visualizations such as performance curves, confusion matrices, and feature-space plots were done using Matplotlib and Seaborn.

Settings and Hyperparameters

We simulate 5-client and 10-client federated environments, where each client holds a disjoint partition of the MNIST, FMNIST, and CIFAR10 datasets. Global aggregation uses accuracy-weighted FedAvg McMahan et al. (2017) by default, while uniform aggregation is used for non-IID and label-flipped settings when appropriate. Each client trains for $E = 1$ local epoch per communication round. We evaluate three data regimes: (i) IID, with a uniform class distribution across clients; (ii) non-IID, where one client is heavily concentrated on a single class (e.g., class 7) while others exclude it; and (iii) label shift, where clients 2 and 5 swap labels $0 \leftrightarrow 1$ and receive a majority allocation of the swapped classes with minority allocation of others. We use 35 - 60 communication rounds, with unlearning triggered at round $t^\star = 25$ and $t^\star = 40$. MNIST and FMNIST use batch size 32 with learning rates of 0.001 and 0.00125, respectively, whereas CIFAR10 uses batch size 128 and learning rate 0.05. MNIST/FMNIST are optimized using Adam, while CIFAR10 uses SGD with Nesterov momentum 0.9. Server checkpoints are stored every 5 rounds, and replay starts from the most recent checkpoint before the unlearning request. Calibration runs for $E_{\text{cal}} = 5$ epochs with a learning rate of 0.001. The weight-negation coefficients λ_l remain depth-aware with $\lambda_l \in [0.4, 1.0]$, while $\beta = 0.005$, $\eta = 0.001$, and $\gamma = 2.0$ control replay stabilization, orthogonality regularization, and target-influence filtering, respectively.

The experiments use a structured set of hyperparameters governing federated training, unlearning, calibration, and membership-inference evaluation. These settings are applied across MNIST, FMNIST, and CIFAR10 while maintaining consistent evaluation across client-based, class-based, and client-class-based unlearning. Table 9 summarizes the main hyperparameter settings.

Evaluation metrics. To remain consistent with previous studies in machine unlearning our evaluation is based on the standard measures used to evaluate the effectiveness of an unlearning method in removing forgotten information and preserving retained knowledge. We report the accuracy of both the forgotten set D_f (Error$_f$) and the remembered set D_r (Error$_r$) and the overall test error D_t (Error$_t$). As in the literature, the retrained model W_r that has never been trained on the forgotten data, is the gold standard of comparing the quality of unlearning. Ideally, the performance of the unlearned

Table 9: Hyperparameter settings for federated training and unlearning

Parameter	Value	Short Description
Federated rounds	35-60	Total number of global communication rounds
Client participation	5, 10	All clients participate in each round
Unlearning request	25-40	Round at which unlearning is triggered
Checkpoint interval	5	Server checkpoint stored every 5 rounds
Calibration epochs	5	Number of post-unlearning calibration epochs
Calibration learning rate	0.001	Learning rate used during calibration
Batch size (MNIST/FMNIST)	32	Mini-batch size for MNIST and FMNIST
Batch size (CIFAR10)	128	Mini-batch size for CIFAR training
Learning rate (MNIST)	0.001	Base learning rate for MNIST
Learning rate (FMNIST)	0.00125	FMNIST learning rate (25% higher than MNIST)
Learning rate (CIFAR10)	0.05	Base learning rate for CIFAR training
Optimizer (MNIST/FMNIST)	Adam	Adam optimizer with default $\beta_1 = 0.9$, $\beta_2 = 0.999$
Optimizer (CIFAR10)	SGD	SGD with Nesterov momentum 0.9
Weight decay (MNIST)	1×10^{-4}	Weight regularization for MNIST
Weight decay (FMNIST)	7×10^{-5}	Weight regularization for FMNIST
Weight decay (CIFAR10)	5×10^{-4}	Weight regularization for CIFAR
Gradient clipping (MNIST/FMNIST)	1.0	Global gradient norm clipping
Gradient clipping (CIFAR10)	5.0	Global gradient norm clipping
MIA thresholds	512	Thresholds scanned during MIA evaluation
MIA feature weights	0.40/0.30/0.15/0.10/0.05	Weights for confidence, probability, correctness, entropy, and loss

model $\tilde{W}_r$ should be comparable to that of W_r implying that the method is effective in eliminating the effect of D_f without impairing the performance of the method on D_r. Besides these fundamental measures, we compute the so-called confusion matrices during training and post-unlearning to learn on the level of classes. These matrices enable us to analyze the ways in which forgetting applies to separate classes and to obtain per-class recall, precision and F1-scores. Monitoring these matrices over federated rounds can give us a fine-grained picture of the dynamics of class-wise. During training, performance changes and after the unlearning operation. In addition to the classic measures, our framework introduces some of the additional evaluation layers that further describe the unlearning behavior. We feature membership inference attack (MIA) metrics accuracy, AUC, precision, recall, F1-score, and advantage to quantify the degree to which forgotten samples are still distinguishable against non-members. We report as well the computational efficiency with execution time per method, and loss-based metrics (separately tracking forgotten loss, remembered loss and test loss). These measures enable us to not only know whether the model forgets, but also how it forgets and at what cost of computation.