Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.085409

ARTICLE

pFedUL: Layer-Aware Federated Unlearning for Personalized Federated Learning

Zhuodong Liu[1], **Xiangyu Li**[2,*] **and Zhihao Zhang**[1]

[1]Saunders College of Business, Rochester Institute of Technology, Rochester, NY, USA
[2]School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA

*Corresponding Author: Xiangyu Li. Email: xli985@gatech.edu

Received: 11 May 2026; Accepted: 15 June 2026; Published: 23 July 2026

ABSTRACT: Federated unlearning (FU) enables the removal of specific data contributions from federated learning (FL) models to comply with regulations such as the General Data Protection Regulation (GDPR). However, most existing FU methods are designed for the FedAvg paradigm, where all clients share a single global model. In practice, personalized federated learning (pFL) methods such as FedPer, FedRep, Ditto, and FedBN have become widely adopted due to their superior handling of non-IID data. These methods decompose the model into shared global layers and client-specific personalized layers, fundamentally altering the semantics of unlearning, yet this setting has received little attention. We formalize FU under the pFL paradigm, identifying a tension between unlearning completeness on shared layers and personalization preservation for remaining clients. We then propose pFedUL, a layer-aware selective unlearning framework comprising three components: (1) gradient-based layer-wise contribution attribution that separately quantifies the target client's influence on shared and personalized parameters, (2) adaptive selective unlearning that applies differentiated forgetting strategies across layer types, and (3) a lightweight recalibration protocol enabling remaining clients to restore personalization with minimal overhead. We further introduce two new metrics, Personalization Preservation Score (PPS) and Cross-client Fairness Index (CFI), to evaluate pFL-specific unlearning quality. Experiments on CIFAR-10, CIFAR-100, and FEMNIST under varying non-IID settings indicate that pFedUL achieves unlearning effectiveness comparable to full retraining while maintaining an average of 97.3% personalized accuracy for remaining clients. Compared with six state-of-the-art FU methods adapted to the pFL setting, pFedUL consistently achieves superior personalization preservation, improving over the best existing method by 6.3% in PPS on average with an 8.4× speedup, averaged across all tested pFL architectures and datasets.

KEYWORDS: Federated learning; federated unlearning; personalized federated learning; machine unlearning; layer-aware unlearning; right to be forgotten

1 Introduction

Federated learning (FL) has emerged as a prominent distributed machine learning paradigm that enables multiple clients to collaboratively train a shared model without directly exchanging raw data [1]. By keeping training data locally on each client and communicating only model updates, FL provides inherent privacy advantages over centralized learning [2]. This paradigm has been widely deployed in domains such as healthcare, mobile keyboard prediction, and recommendation systems, where data sensitivity and regulatory compliance are of paramount importance [3].

Despite its privacy-preserving design, FL does not fully satisfy the "right to be forgotten" mandated by data protection regulations such as the European General Data Protection Regulation (GDPR) [4] and

similar data protection regulations worldwide. These regulations require that, upon a data owner's request, all traces of their data contributions must be effectively removed from the trained model. In centralized settings, machine unlearning has been extensively studied to address this challenge. Representative methods include SISA [5], which partitions training data into disjoint shards and retrains only the affected subset, and influence-function-based approaches [6] that approximate the effect of removing specific data points. However, retraining from scratch in federated settings is prohibitively expensive due to communication overhead and the potential unavailability of participating clients [7].

To address this challenge, federated unlearning (FU) has attracted increasing research attention [8]. FedEraser [9] pioneered this direction by leveraging stored historical gradient updates to calibrate the global model without full retraining. Subsequent works have explored diverse unlearning strategies, including clustered aggregation for asynchronous client removal [10], differential privacy combined with gradient residual correction for certified unlearning [11], auxiliary modules for simultaneous training and unlearning [12], and orthogonal subspace descent to address gradient explosion [13]. Other contributions include class-discriminative pruning [14], projected gradient ascent (PGA) [15], and feature-level unlearning via Lipschitz sensitivity analysis [16]. Several comprehensive surveys [17,18] have systematically categorized these methods and identified evaluation metrics to assess unlearning quality.

In spite of this progress, a critical assumption shared by nearly all existing FU methods remains largely unexamined: they are designed for the standard FedAvg paradigm [1], where all clients collectively maintain a single global model. In practical deployments, however, statistical heterogeneity (non-IID distribution) of client data can severely degrade the performance of FedAvg [19], which has led to the widespread adoption of personalized federated learning (pFL) [20]. pFL methods decompose the model into shared global components and client-specific personalized components, achieving substantially improved performance under data heterogeneity. Representative pFL approaches include FedPer [21], which separates the model into a shared feature extractor and personalized classification heads; FedRep [22], which learns shared representations while allowing clients to independently optimize local heads; Ditto [23], which maintains personalized local models regularized toward the global model; and FedBN [24], which personalizes batch normalization (BN) statistics while sharing all other parameters. These methods have demonstrated consistent improvements over FedAvg in heterogeneous settings [25].

The architectural decomposition introduced by pFL fundamentally alters the semantics of FU, creating challenges that existing methods are not equipped to handle. Specifically, when a target client requests removal from the federation, the unlearning process must operate on two structurally distinct parameter spaces: the shared global layers, which aggregate contributions from all participating clients, and the personalized local layers, which encode each client's unique data distribution. This decomposition introduces a tension between two competing objectives. On the one hand, unlearning completeness requires thoroughly removing the target client's influence from the shared layers. On the other hand, aggressive modifications to these shared layers can disrupt the learned feature representations upon which remaining clients depend for their personalized performance, a phenomenon we term *personalization degradation*. Existing FU methods, which treat all model parameters uniformly, are inherently unable to navigate this trade-off.

To date, only a few works have tangentially touched upon the intersection of FU and personalization. ZeroFU [26] proposed a zero-shot data-free unlearning approach using GAN-based distillation with personalized client features via conditional computation. Mimir [27] introduced a prompt-based personalized unlearning framework that uses client-specific prompts to capture data distribution characteristics. FUSED [28] analyzed layer-wise adapter structures for reversible unlearning. However, none of these methods systematically address unlearning under the model-splitting pFL paradigm (e.g., FedPer, FedRep)

or regularization-based pFL (e.g., Ditto), nor do they provide formal analysis of the tension between shared and personalized layers.

Motivated by these observations, we propose pFedUL, a general layer-aware selective unlearning framework designed for pFL. The core insight is that different layer types require fundamentally different unlearning treatments: shared layers demand careful contribution removal with minimal collateral impact on remaining clients, while personalized layers of the target client can be directly discarded. pFedUL integrates Fisher information-based contribution attribution, adaptive selective correction, and lightweight local recalibration into a cohesive pipeline applicable across mainstream pFL architectures. We further introduce two pFL-specific evaluation metrics to assess unlearning quality dimensions not captured by existing benchmarks. The main contributions of this paper are summarized as follows:

- **Problem formulation:** We formalize the FU problem under the pFL paradigm, identifying a completeness-preservation trade-off between effective target-client removal on shared layers and personalization preservation for remaining clients.
- **General framework:** We propose pFedUL, a layer-aware unlearning framework that combines Fisher-based layer-wise contribution attribution, adaptive selective correction with differentiated intensity across layer types, and a lightweight recalibration protocol for remaining clients, applicable across four mainstream pFL architectures (FedPer, FedRep, Ditto, and FedBN) with architecture-specific instantiations.
- **Evaluation metrics:** We introduce two pFL-specific metrics—the Personalization Preservation Score (PPS) and the Cross-client Fairness Index (CFI)—to assess personalization preservation and cross-client fairness after unlearning, complementing existing MIA-based completeness measures.
- **Empirical validation:** We conduct experiments on CIFAR-10, CIFAR-100, and FEMNIST across four pFL architectures, comparing pFedUL with six state-of-the-art FU methods adapted to the pFL setting and showing that pFedUL approaches retraining-level forgetting while better preserving personalization and substantially reducing unlearning cost.

The remainder of this paper is organized as follows. Section 2 reviews related work on machine unlearning, FU, and pFL. Section 3 presents the proposed pFedUL framework, including the problem formulation and algorithmic details. Section 4 reports the experimental results and analysis. Section 5 discusses implications, limitations, and future directions. Finally, Section 6 concludes the paper.

2 Related Work

2.1 Machine Unlearning

Machine unlearning aims to remove the influence of specific training data from a trained model without retraining from scratch [29]. Exact unlearning methods retrain the model on the remaining data after excluding target samples. SISA [5] improved efficiency by partitioning training data into disjoint shards, each associated with an independently trained sub-model, so that only the affected shard requires retraining upon a removal request. However, this sharding strategy is not directly applicable to distributed settings where data partitioning is determined by client boundaries.

Approximate unlearning methods seek to efficiently modify model parameters to approximate the effect of exact retraining. Influence functions [6] estimate parameter changes caused by removing a single data point through second-order Hessian-based approximations, though at prohibitive cost for large-scale models. Sekhari et al. [30] proposed certified unlearning algorithms with formal indistinguishability guarantees. Thudi et al. [31] analyzed gradient ascent as an unlearning mechanism and showed that simply

reversing gradient updates does not guarantee complete forgetting under general conditions. Relatedly, Elastic Weight Consolidation (EWC) [32] uses the Fisher information matrix to quantify parameter importance for previously learned tasks, providing a foundational tool for identifying which parameters are most affected by specific data contributions. Despite their effectiveness in centralized settings, these methods share a fundamental limitation: they assume direct access to training data or its statistics, which is unavailable in FL where data remain distributed across clients.

2.2 FU

FU extends machine unlearning to distributed settings, aiming to remove a target client's data contributions from the jointly trained global model without requiring full retraining or access to raw client data [8]. A diverse set of FU methods have been proposed over the past several years, which can be broadly categorized by their core technical approach. Table 1 provides a comprehensive summary.

Gradient calibration and approximation. FedEraser [9] stores historical gradient updates during training and calibrates the global model by excluding the target client's contributions. Liu et al. [33] proposed a rapid retraining approach based on first-order Taylor expansion. These methods are computationally efficient but require substantial storage for historical updates.

Structural decomposition. Another line of research exploits federation structure to accelerate unlearning. KNOT [10] organizes clients into clusters with independently maintained sub-models, so that removal affects only the relevant cluster. MUFC [34] demonstrated that federated clustering can achieve exact unlearning with significant speedups. However, these clustering-based approaches do not account for the heterogeneous model structures found in pFL.

Direct parameter manipulation. Several methods directly modify model parameters to erase a target client's influence. FedOSD [13] constrains unlearning updates within orthogonal subspaces for stable convergence. Halimi et al. [15] proposed PGA, which reverses the target client's influence while projecting updates to preserve utility, and NoT [35] negates specific layer weights to disrupt inter-layer co-adaptation. These methods treat all model parameters uniformly, which becomes problematic when the model contains structurally distinct shared and personalized components.

Pruning, distillation, and specialized approaches. Wang et al. [14] proposed class-discriminative pruning using TF-IDF scoring for class-level unlearning. ZeroFU [26] proposed a zero-shot approach using GAN-based distillation with client-specific conditional features, making it one of the few methods that partially accounts for client-specific characteristics. Che et al. [36] developed FFMU based on nonlinear functional theory, enabling simultaneous training and unlearning. Additional specialized methods include FedRecovery [11] for certified removal, FedAU [12] for continuous contribution tracking, Ferrari [16] for feature-level unlearning, VeriFi [18] for unlearning verification, FRU [37] for recommendation domains, and SIFU [38] for provable bounds.

Gap analysis. As summarized in Table 1, the vast majority of existing FU methods operate under the assumption of a single global model, without awareness of the shared-personalized parameter distinction that characterizes pFL architectures. Only three recent works partially incorporate personalized elements: ZeroFU [26] incorporates client-specific conditional features but does not evaluate under model-splitting pFL; Mimir [27] uses prompt-based personalization but is restricted to prompt-tuning scenarios; and FUSED [28] analyzes layer-wise adapter structures but does not formalize the shared-personalized layer distinction. None of these methods directly addresses the tension between unlearning completeness on shared layers and personalization preservation for remaining clients, which motivates the present work.

Table 1: Summary of existing FU methods. All methods assume the standard FedAvg paradigm or its direct variants. The rightmost column describes each method's consideration of pFL architectures.

Method	Year	Venue	Technique	Granularity	Personalization Awareness
FedEraser [9]	2021	IWQoS	Gradient calibration	Client	Assumes a single global model shared by all clients
Rapid Retrain [33]	2022	INFOCOM	Taylor approximation	Sample	Operates on FedAvg; no personalized components considered
PGA [15]	2022	arXiv	PGA	Client	Uniform treatment of all parameters without layer distinction
Class-Pruning [14]	2022	WWW	Channel pruning	Class	Prunes global model channels; no client-specific layers involved
KNOT [10]	2023	INFOCOM	Clustered aggregation	Client	Clusters relate to grouped clients, not personalized architectures
MUFC [34]	2023	ICLR	Federated clustering	Client/Sample	Exact unlearning via cluster retraining; no pFL model structure
FedRecovery [11]	2023	IEEE TIFS	DP + residual correction	Client	Certified removal on a single global model
FRU [37]	2023	WSDM	Rollback + calibration	User	Designed for recommender systems under FedAvg
FFMU [36]	2023	ICML	Functional theory	Sample	Functional-space operations on a unified global model
SIFU [38]	2024	AISTATS	Sequential informed	Client	Provable bounds derived under FedAvg assumptions
FedAU [12]	2024	IJCAI	Auxiliary module	Client/Class	Auxiliary module tracks global contributions only
Ferrari [16]	2024	NeurIPS	Lipschitz sensitivity	Feature	Feature-level removal on a shared global model
VeriFi [18]	2024	IEEE TDSC	Verification framework	Client	Verifies unlearning on FedAvg; pFL verification unexplored
FedOSD [13]	2025	AAAI	Orthogonal subspace	Client	Orthogonal projection in global parameter space

(Continued)

Table 1 (continued)

Method	Year	Venue	Technique	Granularity	Personalization Awareness
NoT [35]	2025	CVPR	Weight negation	Client	Negates global weights; no shared/personalized distinction
ZeroFU [26]	2025	IJCAI	GAN distillation	Client	Embeds client-specific conditional features, but not evaluated on pFL architectures
Mimir [27]	2025	IEEE IoTJ	Prompt distillation	Client	Uses client-specific prompts for personalization, limited to prompt-tuning scenarios
FUSED [28]	2025	CVPR	Sparse adapter decomp	Client	Analyzes layer-wise adapters, but does not formalize shared-personalized decomposition

2.3 pFL

The standard FedAvg algorithm [1] trains a single global model by averaging locally updated parameters from participating clients. However, when client data distributions are highly heterogeneous (non-IID), this uniform aggregation leads to significant performance degradation [19,39]. pFL addresses this challenge by allowing each client to maintain model components tailored to its local data characteristics [20].

The most intuitive approach is to split the network architecture into shared and private components. FedPer [21] designates the lower layers as a shared feature extractor while keeping the upper classification layers local to each client. FedRep [22] adopts a similar split but introduces alternating optimization that decouples the update of shared representations from local heads. FedRoD [40] extends this line by maintaining both a generic and a personalized classifier head simultaneously.

Personalization can also be achieved through optimization-level mechanisms. Ditto [23] trains a global model via FedAvg and then optimizes a personalized local model for each client using proximal regularization that penalizes deviation from the global model. pFedMe [41] follows a similar philosophy but employs Moreau envelopes for stronger convergence guarantees. A more lightweight approach targets normalization layers: FedBN [24] keeps only BN layers local while sharing all other parameters, based on the observation that BN statistics encode domain-specific distributional information. More recently, FedALA [25] proposed learning element-wise adaptive aggregation weights for fine-grained personalization.

Despite their diversity, all pFL methods share a common structural property: the model parameters are decomposed into shared parameters θ^s that capture cross-client common knowledge, and personalized parameters θ^p that encode client-specific patterns. This decomposition is fundamental to their effectiveness but introduces a structural complexity that has been largely overlooked by the FU literature. When a client requests removal, the unlearning procedure must selectively operate on θ^s to erase the target client's contributions while preserving the integrity of the shared representation upon which all remaining clients' personalized parameters θ_i^p depend. To the best of our knowledge, no existing work has formally studied this problem or proposed a dedicated solution, which motivates the framework presented in this paper.

3 The Proposed pFedUL Framework

3.1 Problem Formulation

Consider an FL system consisting of N clients $\{c_1, c_2, \ldots, c_N\}$, where each client c_i has a private local dataset $\mathcal{D}_i$. The standard FL objective is to minimize the global empirical risk:

$$\min_{\theta} F(\theta) = \sum_{i=1}^{N} \frac{|\mathcal{D}_i|}{|\mathcal{D}|} F_i(\theta), \quad F_i(\theta) = \frac{1}{|\mathcal{D}_i|} \sum_{(x,y)\in\mathcal{D}_i} \ell(\theta; x, y), \tag{1}$$

where θ denotes the model parameters, $\mathcal{D} = \bigcup_{i=1}^{N} \mathcal{D}_i$ is the union of all local datasets, ℓ is the loss function, and $F_i(\theta)$ is the local objective of client c_i.

In pFL, the model parameters are decomposed into two functionally distinct groups: shared parameters θ^s that are aggregated across clients, and personalized parameters θ_i^p that are maintained locally by each client c_i. The pFL objective becomes:

$$\min_{\theta^s, \{\theta_i^p\}_{i=1}^N} \sum_{i=1}^{N} \frac{|\mathcal{D}_i|}{|\mathcal{D}|} F_i(\theta^s, \theta_i^p). \tag{2}$$

This general formulation encompasses the four mainstream pFL architectures considered in this work. In FedPer [21] and FedRep [22], θ^s corresponds to the feature extractor layers and θ_i^p to the classification head of client c_i. In Ditto [23], θ^s is the global model obtained through FedAvg, and θ_i^p is the local fine-tuned model regularized toward θ^s via a proximal term $\lambda\|\theta_i^p - \theta^s\|^2$. In FedBN [24], θ^s consists of all parameters except BN layers, while θ_i^p contains the local BN statistics.

To clarify the architecture-specific differences in how shared and personalized parameters are defined, Table 2 provides a detailed comparison across the four pFL methods. As shown, the nature and scale of personalized parameters vary substantially: in FedPer and FedRep, θ_i^p consists of a lightweight classification head (approximately 0.5% of total parameters for ResNet-18 on CIFAR-10); in FedBN, θ_i^p comprises the BN layers (approximately 1.2% of total parameters); while in Ditto, θ_i^p is a full copy of the entire model. These differences have direct implications for pFedUL's operation: the recalibration cost in Stage 3 is proportional to $|\theta_i^p|$, and the treatment of the target client's personalized parameters upon removal differs accordingly (discarding a classification head vs. discarding a full model copy).

Table 2: Architecture-specific decomposition of shared parameters θ^s and personalized parameters θ_i^p across the four pFL methods evaluated in this work. The table specifies the concrete parameter assignment, the approximate proportion of personalized parameters relative to the total model, and the pFedUL-specific treatment during unlearning and recalibration for each architecture.

Method	θ^s (Shared)	θ_i^p (Personalized)	$\lvert\theta_i^p\rvert/\lvert\theta\rvert$	Target $\theta_{c_t}^p$ Treatment	Recalibration
FedPer	All layers except final FC	Final FC layer (classifier head)	~0.5%	Delete head	Fine-tune head with $\hat{\theta}^s$ frozen
FedRep	All layers except final FC	Final FC layer (classifier head)	~0.5%	Delete head	Alternating opt. on head with $\hat{\theta}^s$ frozen
Ditto	Global model (full FedAvg model)	Full local model (ℓ_2-regularized toward θ^s)	100%	Delete full local model	Re-run proximal opt. (Eq. (13)) with $\hat{\theta}^s$ as anchor

(Continued)

Table 2 (continued)

Method	θ^s (Shared)	θ_i^p (Personalized)	$\lvert\theta_i^p\rvert/\lvert\theta\rvert$	Target $\theta_{c_t}^p$ Treatment	Recalibration
FedBN	All layers except BN layers	BN running mean and variance	~1.2%	Delete local BN stats	Re-estimate BN stats via forward pass on local data

We now formalize the FU problem under the pFL paradigm. Let c_t denote the target client that requests data removal. After the pFL training process converges, the system maintains the shared parameters θ^s and each remaining client c_i $(i \neq t)$ maintains personalized parameters θ_i^p. The objective of pFL-aware FU is to produce updated shared parameters $\hat{\theta}^s$ such that:

$$\hat{\theta}^s \approx \theta_{\setminus t}^{s,*} = \arg\min_{\theta^s} \sum_{i \neq t} \frac{|\mathcal{D}_i|}{|\mathcal{D}_{\setminus t}|} F_i(\theta^s, \theta_i^p), \tag{3}$$

where $\theta_{\setminus t}^{s,*}$ represents the shared parameters that would be obtained by retraining from scratch on the remaining clients, and $\mathcal{D}_{\setminus t} = \bigcup_{i \neq t} \mathcal{D}_i$. This formulation reveals a fundamental tension unique to the pFL setting. The personalized parameters θ_i^p of the remaining clients were optimized jointly with the original shared parameters θ^s, which contain the target client's contributions. Modifying θ^s to $\hat{\theta}^s$ for unlearning purposes disrupts this co-optimization, potentially degrading the personalized performance of remaining clients. We refer to this as the *completeness-preservation trade-off*: aggressive unlearning on θ^s improves forgetting completeness but risks damaging remaining clients' personalization, while conservative unlearning preserves personalization but may leave residual traces of the target client's data.

Remark 1 (Factors governing the trade-off severity). The severity of the completeness-preservation trade-off is governed by several factors that can be qualitatively characterized. First, when the target client's data volume $|\mathcal{D}_t|$ constitutes a larger fraction of the total data $|\mathcal{D}|$, its contributions to the shared parameters θ^s are more deeply embedded, requiring larger parameter modifications for complete removal and thus increasing the risk of disrupting remaining clients' personalization. Second, higher data heterogeneity (smaller α_{dir}) leads to more differentiated per-client contributions across layers, which paradoxically benefits layer-aware unlearning by concentrating the target client's influence in fewer layers and thereby reducing collateral perturbation. Third, when remaining clients' personalized parameters θ_i^p have been tightly co-optimized with θ^s over many communication rounds, the sensitivity of θ_i^p to changes in θ^s increases, amplifying the misalignment caused by unlearning. These observations motivate the three-stage design of pFedUL, which uses Fisher-based attribution to identify high-influence layers (addressing the first factor), applies selective correction to exploit the concentration of contributions (addressing the second), and performs lightweight recalibration to restore co-optimization alignment (addressing the third). The empirical sensitivity analysis in Section 4.7 provides quantitative support for these qualitative observations.

3.2 Framework Overview

To address the completeness-preservation trade-off, pFedUL adopts a three-stage pipeline, as illustrated in Fig. 1. Given a trained pFL system and a target client c_t requesting removal, pFedUL proceeds as follows.

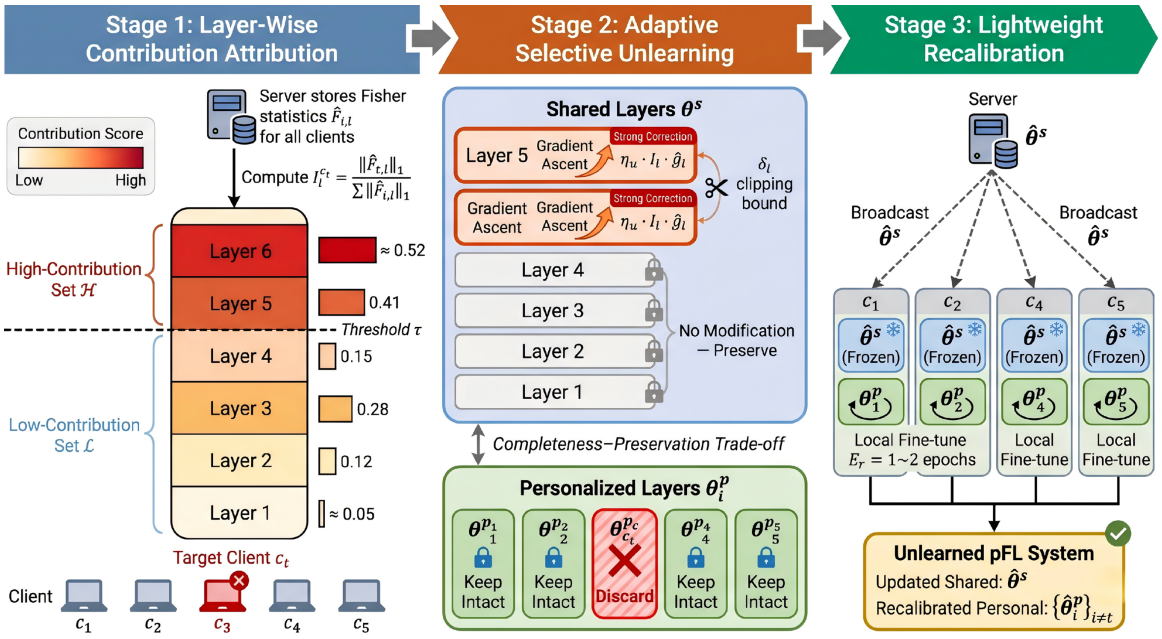

Figure 1: Overview of the proposed pFedUL framework. The framework operates in three stages: (1) Computing layer-wise contribution scores of the target client using Fisher-based attribution; (2) Applying adaptive unlearning with differentiated intensity across shared layers based on contribution scores, while directly discarding the target client's personalized parameters; (3) performing lightweight local recalibration for remaining clients to restore personalized performance.

In the first stage (Section 3.3), pFedUL computes a layer-wise contribution score for each shared layer, quantifying how much the target client c_t has influenced each layer's parameters during training. This is achieved through a Fisher information-based attribution mechanism that operates on stored gradient statistics without requiring access to the target client's raw data.

In the second stage (Section 3.4), pFedUL applies adaptive selective unlearning to shared parameters θ^s. Layers with high contribution scores from c_t receive aggressive correction through a constrained gradient ascent, whereas layers with low contribution scores undergo minimal or no modification. The personalized parameters of the target client $\theta_{c_t}^p$ are immediately discarded, and all remaining personalized parameters of the clients θ_i^p $(i \neq t)$ are kept intact during this stage.

In the third stage (Section 3.5), a lightweight recalibration protocol allows each remaining client to local fine-tune its personalized parameters θ_i^p with respect to the updated shared parameters $\hat{\theta}^s$, using only 1 to 2 epochs of local training. This restores the co-optimization relationship between shared and personalized components without requiring additional global communication rounds.

3.3 Layer-Wise Contribution Attribution

The first stage of pFedUL aims to quantify the target client's influence on each layer of the shared parameters. We propose a Fisher information-based attribution mechanism that provides a principled measure of each layer's sensitivity to the target client's data.

Let $\theta^s = \{\theta_1^s, \theta_2^s, \ldots, \theta_L^s\}$ denote the shared parameters decomposed into L layers. During the federated training process, we maintain running estimates of the diagonal Fisher information matrix for each client and each layer. The diagonal Fisher information captures the expected squared gradient of the loss with respect

to each parameter, serving as a measure of how sensitive a given layer is to the data of a particular client. Specifically, for client c_i and layer l, the diagonal Fisher information is defined as:

$$\mathbf{F}_{i,l} = \frac{1}{|\mathcal{D}_i|} \sum_{(x,y)\in\mathcal{D}_i} \left(\nabla_{\theta_l^s} \ell(\theta^s, \theta_i^p; x, y)\right)^2. \tag{4}$$

In practice, computing (Eq. (4)) exactly requires access to client data, which is unavailable at unlearning time. Instead, we accumulate an exponential moving average (EMA) of the squared gradients during training. In each communication round r, when client c_i uploads its local gradient update $g_{i,l}^r$ for layer l, the server maintains:

$$\hat{\mathbf{F}}_{i,l} \leftarrow \beta \hat{\mathbf{F}}_{i,l} + (1-\beta)(g_{i,l}^r)^2, \tag{5}$$

where $\beta \in (0,1)$ is the momentum coefficient that controls the trade-off between the recent and historical gradient information. Over the course of training, this running estimate converges to an approximation of the diagonal Fisher information.

Given the accumulated Fisher statistics, the layer-wise contribution score of the target client c_t for layer l is defined as the ratio of the target client's Fisher information to the aggregate Fisher information across all clients:

$$I_l^{c_t} = \frac{\|\hat{\mathbf{F}}_{t,l}\|_1}{\sum_{i=1}^N \|\hat{\mathbf{F}}_{i,l}\|_1}, \tag{6}$$

where $\|\cdot\|_1$ denotes the L_1 norm. Intuitively, $I_l^{c_t} \in [0,1]$ measures the proportion of the total parameter sensitivity at layer l that is attributable to the target client. A high value of $I_l^{c_t}$ indicates that the target client has strongly shaped the parameters of layer l, suggesting that more aggressive unlearning is warranted at this layer.

This attribution mechanism offers two practical advantages. First, it only requires storing L vectors of the same dimensionality as each layer's parameters per client, which incurs modest memory overhead. Second, the running average in Eq. (5) can be computed incrementally during training without requiring any post-hoc data access, making it naturally compatible with the privacy constraints of FL.

Remark 2 (Theoretical connection between Fisher attribution and unlearning). The use of the diagonal Fisher information matrix as a contribution attribution metric is grounded in its well-established role as a measure of parameter importance with respect to specific data [6,32]. For a given client c_i and layer l, the diagonal Fisher $\mathbf{F}_{i,l}$ captures the expected curvature of the loss landscape along each parameter dimension: parameters with large Fisher values are those whose perturbation would most strongly affect the loss evaluated on client c_i's data, and hence are the parameters that most encode c_i's data characteristics. Consequently, the contribution score $I_l^{c_t}$ (Eq. (6)) quantifies the relative proportion of parameter sensitivity at layer l that is attributable to the target client c_t. A high $I_l^{c_t}$ value indicates that the parameters of layer l are disproportionately sensitive to the target client's data distribution, implying that the target client's influence is concentrated in that layer. By directing unlearning corrections to layers with high $I_l^{c_t}$, pFedUL targets the parameter subspace where the target client's data patterns are most strongly encoded, while preserving layers where the target client's influence is negligible relative to other clients. This reasoning is consistent with the EWC framework [32,42], which demonstrates that the Fisher information matrix reliably identifies parameters critical to specific tasks, and with influence function theory [6], which shows that parameter importance weighted by curvature provides accurate approximations of data removal effects. We provide empirical validation of this theoretical connection in Section 4.6, where we demonstrate

that selectively unlearning high-attribution layers achieves substantially better forgetting than unlearning low-attribution layers.

3.4 Adaptive Selective Unlearning

Given the layer-wise contribution scores $\{I_l^{c_t}\}_{l=1}^{L}$ computed in the attribution stage, pFedUL applies differentiated unlearning strategies to different layers of the shared parameters.

We first classify each shared layer into one of two categories based on an adaptive threshold τ. Layers with contribution scores exceeding τ are designated as *high-contribution layers*, while the remaining layers are designated as *low-contribution layers*:

$$\mathcal{H} = \{l \mid I_l^{c_t} > \tau\}, \quad \mathcal{L} = \{l \mid I_l^{c_t} \leq \tau\}. \tag{7}$$

The threshold τ is set adaptively based on the distribution of contribution scores:

$$\tau = \mu_I + \alpha \cdot \sigma_I, \tag{8}$$

where $\mu_I = \frac{1}{L}\sum_{l=1}^{L} I_l^{c_t}$ and $\sigma_I = \sqrt{\frac{1}{L}\sum_{l=1}^{L}(I_l^{c_t} - \mu_I)^2}$ are the mean and standard deviation of the contribution scores across all layers, and $\alpha \geq 0$ is a sensitivity hyperparameter. A larger α yields a more conservative unlearning strategy that modifies only the most heavily influenced layers, while $\alpha = 0$ applies correction to all above-average layers.

For high-contribution layers ($l \in \mathcal{H}$), we apply constrained gradient ascent to actively erase the target client's influence. The update for layer l is:

$$\hat{\theta}_l^s = \theta_l^s + \eta_u \cdot I_l^{c_t} \cdot \hat{g}_l^{c_t}, \tag{9}$$

where η_u is the unlearning step size and $\hat{g}_l^{c_t}$ is the estimated unlearning direction for layer l. The unlearning direction is derived from the accumulated gradient information of the target client. Specifically, it combines the Fisher sensitivity of each parameter with its deviation from initialization, under the heuristic that parameters exhibiting both high sensitivity to the target client and large displacement from their initial values are most likely to encode that client's data patterns:

$$\hat{g}_l^{c_t} = \hat{\mathbf{F}}_{t,l} \odot (\theta_l^s - \theta_{l,\text{init}}^s), \tag{10}$$

where $\theta_{l,\text{init}}^s$ denotes the initial parameters of layer l before training, and $\odot$ denotes element-wise multiplication.

The multiplication by $I_l^{c_t}$ in Eq. (9) ensures that the correction intensity is proportional to the layer's contribution score, implementing the adaptive nature of the approach. To prevent the unlearning update from excessively degrading model utility, we apply a projection constraint that bounds the magnitude of the modification at each layer:

$$\hat{\theta}_l^s \leftarrow \theta_l^s + \text{clip}\left(\hat{\theta}_l^s - \theta_l^s,\ -\delta_l,\ \delta_l\right), \tag{11}$$

where $\delta_l = \gamma \cdot \|\theta_l^s\|_2 / \sqrt{d_l}$ is the per-layer clipping bound, d_l is the dimensionality of layer l, and γ is a global clipping coefficient. This projection ensures that the unlearning update remains within a trust region around the original parameters, reducing the risk of catastrophic degradation of the shared representation.

For low-contribution layers ($l \in \mathcal{L}$), no modification is applied, i.e., $\hat{\theta}_l^s = \theta_l^s$. This selective approach minimizes unnecessary perturbation to the shared representation, which is the key mechanism for preserving remaining clients' personalization.

Regarding personalized parameters, the target client's personalized parameters $\theta_{c_t}^p$ are directly deleted from the system, as they are exclusively owned by the departing client. The personalized parameters θ_i^p of all remaining clients ($i \neq t$) are kept unchanged during this stage. Any potential misalignment with the updated $\hat{\theta}^s$ is addressed in the subsequent recalibration stage.

Remark. The layer-aware strategy is expected to outperform uniform unlearning (applying the same correction to all layers) for two reasons. First, in deep neural networks, the influence of a specific client's data is typically concentrated in certain layers rather than uniformly distributed [32]. Second, modifying low-contribution layers introduces unnecessary noise that degrades the shared representation without improving forgetting completeness, whereas the selective approach avoids this wasteful perturbation. This reasoning is empirically validated in the ablation study (Section 4.5).

3.5 Lightweight Recalibration Protocol

After the selective unlearning stage modifies the shared parameters from θ^s to $\hat{\theta}^s$, the personalized parameters θ_i^p of remaining clients may no longer be optimally aligned with the updated shared representation. This misalignment arises because θ_i^p was jointly optimized with the original θ^s, and the unlearning-induced modifications, even when carefully controlled, shift the feature space encoded by the shared layers.

To restore this alignment, pFedUL employs a lightweight recalibration protocol. Each remaining client c_i ($i \neq t$) receives the updated shared parameters $\hat{\theta}^s$ and performs a small number of local training epochs E_r to re-optimize its personalized parameters:

$$\hat{\theta}_i^p = \arg\min_{\theta_i^p} F_i(\hat{\theta}^s, \theta_i^p), \tag{12}$$

where $\hat{\theta}^s$ is frozen during this local optimization. In practice, we find that $E_r = 1$ to 2 local epochs are sufficient to restore personalized performance to within approximately 1% of the pre-unlearning level, as shown in our experiments.

The recalibration protocol is efficient for three reasons. First, only the personalized parameters θ_i^p are updated, which typically constitute a small fraction of the total model parameters (e.g., only the classification head in FedPer/FedRep, or only the BN layers in FedBN). Second, the shared parameters $\hat{\theta}^s$ are frozen, so no global communication is required during recalibration. Third, the number of recalibration epochs E_r is very small because the selective unlearning strategy in Section 3.4 has already minimized disruption to low-contribution layers, reducing the magnitude of misalignment that recalibration needs to correct.

For the special case of Ditto, where the personalized model θ_i^p is a full local model regularized toward θ^s, the recalibration takes the form:

$$\hat{\theta}_i^p = \arg\min_{\theta_i^p} \left[F_i(\theta_i^p) + \frac{\lambda}{2} \|\theta_i^p - \hat{\theta}^s\|^2 \right], \tag{13}$$

which re-runs the Ditto local optimization with the updated global model $\hat{\theta}^s$ as the new regularization anchor. This requires E_r local epochs with no additional communication, maintaining the same overhead as in the model-splitting case.

3.6 Evaluation Metrics for pFL Unlearning

Standard FU evaluation relies on metrics such as membership inference attack (MIA) accuracy [42] for unlearning completeness and remaining accuracy for model utility preservation. While these metrics remain

relevant, they do not capture two quality dimensions that are important in the pFL setting: the preservation of per-client personalized performance and the fairness of unlearning impact across remaining clients. We introduce two complementary metrics to address these gaps.

The PPS measures the extent to which each remaining client retains its personalized performance after unlearning. Let A_i^{pre} and A_i^{post} denote the personalized test accuracy of client c_i before and after unlearning, respectively. The PPS is defined as:

$$\text{PPS} = \frac{1}{N-1} \sum_{i \neq t} \frac{A_i^{\text{post}}}{A_i^{\text{pre}}}. \tag{14}$$

A PPS of 1.0 indicates perfect personalization preservation, meaning that no remaining client has experienced any performance degradation due to unlearning. Values below 1.0 indicate degradation, with lower values signifying more severe impact. Note that the Retrain baseline is assigned PPS = 1.0 and CFI = 1.0 by definition, since retraining from scratch produces the ideal reference model; all other methods are evaluated relative to this upper bound. Unlike global remaining accuracy, PPS explicitly accounts for each client's individual performance trajectory, making it sensitive to cases where the overall average accuracy is maintained but specific clients suffer disproportionate losses.

The CFI quantifies the uniformity of unlearning impact across remaining clients. An ideal unlearning procedure should affect all remaining clients similarly, rather than disproportionately harming certain clients. The CFI is defined as:

$$\text{CFI} = 1 - \sqrt{\frac{1}{N-1} \sum_{i \neq t} \left(\frac{A_i^{\text{post}}}{A_i^{\text{pre}}} - \text{PPS} \right)^2}. \tag{15}$$

The CFI subtracts the standard deviation of per-client preservation ratios from 1, yielding values close to 1.0 when the unlearning impact is uniformly distributed and lower values when certain clients are disproportionately affected. Together, PPS and CFI provide a more comprehensive assessment of pFL unlearning quality that complements the standard MIA-based completeness metrics.

Remark 3 (Limitations of PPS and CFI). While PPS and CFI capture important evaluation dimensions, several caveats should be noted. First, both metrics are ratio-based, which makes them sensitive to clients with very low initial accuracy A_i^{pre}: a small absolute change in accuracy can produce a disproportionately large ratio deviation. In our experiments, this issue is mitigated by the fact that pFL methods generally achieve reasonable personalized accuracy for all clients, but in extreme heterogeneity settings where some clients have very few samples, this sensitivity warrants caution. Second, clients with small local test sets may exhibit high variance in their accuracy estimates, which propagates into both PPS and CFI. To account for this, we report standard deviations across five independent runs, and in Section 4.6, we additionally present client-level performance distributions to complement the averaged metrics. Third, PPS treats all clients equally regardless of their data volume, which may not always align with practical deployment priorities where certain clients may be more important than others. Despite these limitations, PPS and CFI provide a substantially more informative evaluation than global remaining accuracy alone, and we consider them useful additions to the FU evaluation toolkit.

3.7 Overall Algorithm

The complete pFedUL procedure is summarized in Algorithm 1. The algorithm takes as input the trained pFL system (shared parameters, personalized parameters, and stored Fisher statistics) and the identity of the

target client, and outputs the unlearned shared parameters along with recalibrated personalized parameters for all remaining clients.

Algorithm 1: The pFedUL framework

Require: Shared parameters $\theta^s = \{\theta_1^s, \ldots, \theta_L^s\}$; personalized parameters $\{\theta_i^p\}_{i=1}^N$; stored Fisher statistics $\{\hat{\mathbf{F}}_{i,l}\}$ for all clients i and layers l; target client c_t; hyperparameters α, η_u, γ, E_r

Ensure: Unlearned shared parameters $\hat{\theta}^s$; recalibrated personalized parameters $\{\hat{\theta}_i^p\}_{i\neq t}$

// Stage 1: Layer-Wise Contribution Attribution

for each shared layer $l = 1, 2, \ldots, L$ **do**

  Compute contribution score $I_l^{c_t} \leftarrow \|\hat{\mathbf{F}}_{t,l}\|_1 \,/\, \sum_{i=1}^N \|\hat{\mathbf{F}}_{i,l}\|_1$ ▷ Eq. (6)

end for

Compute threshold $\tau \leftarrow \mu_I + \alpha \cdot \sigma_I$ ▷ Eq. (8)

Classify layers: $\mathcal{H} \leftarrow \{l \mid I_l^{c_t} > \tau\}$, $\mathcal{L} \leftarrow \{l \mid I_l^{c_t} \leq \tau\}$ ▷ Eq. (7)

// Stage 2: Adaptive Selective Unlearning

for each layer $l \in \mathcal{H}$ **do**

  Compute unlearning direction $\hat{g}_l^{c_t} \leftarrow \hat{\mathbf{F}}_{t,l} \odot (\theta_l^s - \theta_{l,\text{init}}^s)$ ▷ Eq. (10)

  Compute update $\Delta_l \leftarrow \eta_u \cdot I_l^{c_t} \cdot \hat{g}_l^{c_t}$ ▷ Eq. (9)

  Compute clipping bound $\delta_l \leftarrow \gamma \cdot \|\theta_l^s\|_2 / \sqrt{d_l}$

  Apply projection $\hat{\theta}_l^s \leftarrow \theta_l^s + \text{clip}(\Delta_l, -\delta_l, \delta_l)$ ▷ Eq. (11)

end for

for each layer $l \in \mathcal{L}$ **do**

  $\hat{\theta}_l^s \leftarrow \theta_l^s$ ▷ No modification

end for

Delete target client's personalized parameters $\theta_{c_t}^p$

// Stage 3: Lightweight Recalibration

Broadcast $\hat{\theta}^s$ to all remaining clients

for each remaining client c_i $(i \neq t)$ **in parallel do**

  for epoch $= 1, 2, \ldots, E_r$ **do**

    Update $\theta_i^p \leftarrow \theta_i^p - \eta_r \nabla_{\theta_i^p} F_i(\hat{\theta}^s, \theta_i^p)$ ▷ Eq. (12)

  end for

  $\hat{\theta}_i^p \leftarrow \theta_i^p$

end for

return $\hat{\theta}^s$, $\{\hat{\theta}_i^p\}_{i\neq t}$

The computational complexity of Algorithm 1 is dominated by Stages 2 and 3. Stage 1 requires only $O(L)$ vector norm computations. Stage 2 performs element-wise operations on at most $|\mathcal{H}|$ layers, with cost $O(\sum_{l\in\mathcal{H}} d_l)$, which is linear in the number of parameters in the high-contribution layers. Stage 3 requires each remaining client to perform E_r local epochs on their personalized parameters, with cost $O(E_r \cdot |\theta_i^p| \cdot |\mathcal{D}_i|)$ per client. Since $|\theta_i^p| \ll |\theta^s|$ in typical pFL architectures and E_r is very small (1 to 2), the total overhead of pFedUL is substantially lower than retraining from scratch, which would require $O(T \cdot |\theta| \cdot |\mathcal{D}_{\setminus t}|)$ for T communication rounds over the full parameter set.

4 Experiments

4.1 Experimental Setup

We evaluate pFedUL on three widely used FL benchmarks. CIFAR-10 consists of 60,000 color images across 10 classes, with 50,000 for training and 10,000 for testing. CIFAR-100 follows the same image

dimensions but spans 100 fine-grained classes, each containing 600 images. FEMNIST is a character recognition dataset from the LEAF benchmark [43] with naturally heterogeneous data distributions across writers, containing 62 classes (digits, uppercase, and lowercase letters). To simulate controlled non-IID data heterogeneity, we partition CIFAR-10 and CIFAR-100 across clients using the Dirichlet distribution $\mathrm{Dir}(\alpha_{\mathrm{dir}})$ [44], where smaller α_{dir} values indicate higher heterogeneity. Unless otherwise stated, we set $\alpha_{\mathrm{dir}} = 0.5$ as the default non-IID setting. For FEMNIST, we use the natural writer-based partition provided by the benchmark.

We deploy $N = 20$ clients in all experiments and designate one randomly selected client as the target client c_t for unlearning. All models use ResNet-18 as the backbone architecture. The federated training runs for $T = 200$ communication rounds with full client participation, using SGD with learning rate 0.01, momentum 0.9, and local epochs $E = 5$. For pFedUL, the key hyperparameters are set as follows: Fisher momentum $\beta = 0.99$, threshold sensitivity $\alpha = 0.5$, unlearning step size $\eta_u = 0.1$, clipping coefficient $\gamma = 0.01$, and recalibration epochs $E_r = 2$. The selection of these hyperparameters is guided by the following considerations. The Fisher momentum $\beta = 0.99$ follows the standard EMA practice used in Adam-style optimizers, ensuring that the accumulated Fisher statistics reflect a smoothed estimate over the full training trajectory. The threshold sensitivity $\alpha = 0.5$ is selected based on the trade-off analysis presented in Section 4.7, which shows that $\alpha = 0.5$ provides a favorable balance between unlearning completeness and personalization preservation. The unlearning step size $\eta_u = 0.1$ is set to be 10× the training learning rate (0.01), consistent with the common practice that unlearning typically requires larger step sizes than learning to effectively reverse parameter updates within a single pass [31]. The clipping coefficient $\gamma = 0.01$ constrains per-layer modifications to approximately 1% of the layer's parameter norm, which we found sufficient to prevent catastrophic degradation while allowing meaningful correction. The recalibration epoch count $E_r = 2$ is chosen because our experiments show diminishing returns beyond this point: increasing E_r from 1 to 2 recovers the majority of misalignment-induced PPS loss (from 0.948 to 0.975), while further increasing E_r to 5 yields negligible additional improvement (0.977). All experiments are repeated five times with different random seeds, and we report the mean and standard deviation.

We evaluate pFedUL under four mainstream pFL architectures that span the major personalization paradigms. FedPer [21] splits the model at the last fully connected layer, sharing the feature extractor and keeping the classifier head local. FedRep [22] uses the same split but employs alternating optimization between shared and personalized components. Ditto [23] trains personalized local models with ℓ_2 regularization ($\lambda = 0.1$) toward the global model. FedBN [24] keeps BN layers local while sharing all other parameters.

We compare pFedUL against five basic baselines and six state-of-the-art FU methods. The basic baselines are as follows. *Retrain* removes the target client and retrains the entire pFL system from scratch, serving as the gold-standard upper bound. *FedEraser-adapted* applies the FedEraser [9] gradient calibration method to the shared parameters θ^s of each pFL architecture. *GA-Naive* performs uniform gradient ascent on all shared layers without layer-wise differentiation. *Fine-tune* first applies gradient ascent for unlearning and then fine-tunes the entire model on remaining clients' data for 10 rounds. *No-Unlearn* makes no modifications after the target client departs, serving as a lower bound for unlearning completeness.

To provide a more comprehensive comparison, we further adapt six representative FU methods to the pFL setting by applying their unlearning operations to the shared parameters θ^s while preserving the personalized parameters of remaining clients. These include: *KNOT-adapted* [10], which performs clustered aggregation-based unlearning on shared parameters; *FedRecovery-adapted* [11], which combines differential privacy with gradient residual correction on shared layers; *PGA-adapted* [15], which applies PGA on shared layers; *ZeroFU-adapted* [26], which uses GAN-based distillation with client-specific conditional features;

Mimir-adapted [27], which employs prompt-based knowledge distillation for personalized unlearning; and *FUSED-adapted* [28], which applies sparse adapter decomposition for layer-wise unlearning. For methods that incorporate personalization-aware components (ZeroFU, Mimir, FUSED), we use their original designs where applicable and adapt only the incompatible components to the pFL architectures under evaluation.

To ensure fair comparison, Table 3 details the specific adaptation protocol applied to each baseline method, including which parameters are modified during unlearning, which are frozen, how the target client's personalized parameters are handled, and whether a recalibration step is included. For methods that do not originally include any form of post-unlearning recalibration, we do not add one, as doing so would conflate the baseline's intrinsic capability with the benefit of recalibration. This design ensures that the observed improvements of pFedUL are attributable to its complete framework rather than an unfair advantage from recalibration alone. The ablation study in Section 4.5 separately quantifies the contribution of recalibration.

Table 3: Adaptation protocol for baseline FU methods under the pFL setting. For each method, the table specifies which parameters are subject to unlearning operations, which are frozen, how the target client's personalized parameters are treated, and whether post-unlearning recalibration is applied. All adapted methods delete the target client's personalized parameters $\theta_{c_t}^p$ and preserve remaining clients' personalized parameters θ_i^p $(i \neq t)$ unless otherwise noted.

Method	Parameters Modified	Recalib.	Adaptation Notes
GA-Naive	All shared layers θ^s (uniform GA)	No	Gradient ascent applied uniformly to all shared parameters
Fine-tune	All shared layers θ^s (GA then FT)	No	10 rounds of fine-tuning on remaining clients after GA
FedEraser-adapt.	Shared layers θ^s (gradient calibration)	No	Historical gradients used to calibrate θ^s only
PGA-adapt.	Shared layers θ^s (projected GA)	No	Projection constraint applied in θ^s space
KNOT-adapt.	Shared layers θ^s (cluster retrain)	No	Cluster formed on shared parameters; affected cluster retrained
FedRecovery-adapt.	Shared layers θ^s (DP + residual)	No	DP noise and gradient residual correction on θ^s
ZeroFU-adapt.$^{\leqq}$	Shared layers θ^s (GAN distillation)	No	Client-specific conditional features retained from original design
Mimir-adapt.$^{\leqq}$	Shared layers θ^s (prompt distillation)	No	Client-specific prompts retained; distillation on θ^s
FUSED-adapt.$^{\leqq}$	Shared layers θ^s (sparse adapters)	No	Layer-wise adapter decomposition applied to θ^s
pFedUL	High-contribution shared layers (selective)	Yes	Fisher-based selective correction + $E_r = 2$ recalibration

Note: $^{\leqq}$Methods with partial personalization awareness in their original designs.

We evaluate all methods using four metrics: MIA accuracy [42] (closer to 0.5 indicates better unlearning), Remaining Accuracy (test accuracy of remaining clients after unlearning), PPS (Eq. (14)), and CFI (Eq. (15)). Additionally, we measure the wall-clock time of each unlearning method.

4.2 Unlearning Effectiveness

We first evaluate unlearning completeness by measuring MIA accuracy and remaining model utility. Table 4 presents the results across all datasets, pFL architectures, and methods.

Table 4: Unlearning effectiveness comparison across three datasets and four pFL architectures. MIA Acc. (↓, closer to 0.50 is better) measures unlearning completeness; Rem. Acc. (↑) measures remaining model utility. Best results (excluding Retrain) are in **bold**. Results are averaged over 5 runs.

Method	FedPer		FedRep		Ditto		FedBN	
	MIA↓	Rem.↑	MIA↓	Rem.↑	MIA↓	Rem.↑	MIA↓	Rem.↑
CIFAR-10 (α_{dir} = 0.5)								
No-Unlearn	78.4	84.7	79.1	85.3	77.6	86.1	76.8	80.5
Fine-tune	58.3	82.1	59.1	83.0	57.5	84.3	58.7	78.6
GA-Naive	50.2	72.4	49.8	73.1	50.6	74.5	49.5	69.8
FedEraser-adapt.	53.6	78.9	54.1	79.5	53.2	80.8	54.8	75.3
pFedUL	**51.3**	**83.5**	**51.1**	**84.2**	**51.5**	**85.4**	**51.8**	**79.6**
Retrain	50.1	84.3	50.2	84.9	50.3	85.8	50.0	80.1
CIFAR-100 (α_{dir} = 0.5)								
No-Unlearn	81.2	58.6	82.0	59.4	80.3	60.7	79.5	55.2
Fine-tune	60.7	56.1	61.3	56.8	59.8	58.2	61.5	52.7
GA-Naive	51.4	44.3	50.8	45.1	51.9	46.8	50.3	41.6
FedEraser-adapt.	55.9	51.8	56.4	52.5	55.1	53.9	57.2	48.3
pFedUL	**52.1**	**57.5**	**51.8**	**58.3**	**52.4**	**59.8**	**52.7**	**54.1**
Retrain	50.3	58.1	50.1	59.0	50.5	60.3	50.2	54.8
FEMNIST (natural partition)								
No-Unlearn	75.6	91.2	76.3	91.8	74.9	92.4	74.1	89.3
Fine-tune	56.4	89.5	57.2	90.1	55.8	90.9	57.1	87.5
GA-Naive	50.8	82.6	50.3	83.2	51.2	84.1	49.7	80.4
FedEraser-adapt.	53.1	86.3	53.8	86.9	52.7	87.8	54.3	83.7
pFedUL	**51.0**	**90.4**	**50.8**	**91.0**	**51.3**	**91.7**	**51.6**	**88.5**
Retrain	50.2	90.8	50.0	91.5	50.4	92.1	50.1	89.0

Several observations can be drawn from Table 4. First, pFedUL consistently achieves MIA accuracy within 1.0 to 1.8 percentage points of the Retrain baseline across all dataset–architecture combinations, suggesting that the layer-aware unlearning strategy effectively erases the target client's data influence. In contrast, Fine-tune and FedEraser-adapted leave substantially higher MIA accuracy (7 to 11 points above 0.5), indicating incomplete forgetting. GA-Naive achieves MIA accuracy close to 0.5 but at the cost of severe remaining accuracy degradation, as the uniform gradient ascent indiscriminately damages all shared layers regardless of their contribution from the target client.

Second, pFedUL maintains remaining accuracy within 0.5 to 1.2 percentage points of Retrain, whereas GA-Naive suffers 10 to 15 point drops and FedEraser-adapted drops 4 to 7 points. This suggests that the selective nature of pFedUL's layer-aware correction successfully preserves shared representation quality. The advantage is particularly notable on CIFAR-100, where the larger number of classes makes the shared representation more sensitive to indiscriminate perturbation.

Third, the results are consistent across all four pFL architectures, supporting the generalizability of pFedUL's unified formulation. The slight variations across architectures (e.g., FedBN showing marginally lower remaining accuracy) likely reflect inherent differences in personalization capacity rather than limitations of pFedUL.

4.3 Personalization Preservation

While Table 4 shows pFedUL's effectiveness in balancing unlearning completeness and remaining accuracy, the PPS and CFI metrics provide deeper insight into per-client personalization quality. Table 5 presents these results with standard deviations across five runs. Note that Retrain serves as the normalized upper bound (PPS = 1.0, CFI = 1.0), and all other methods are evaluated against this reference.

Table 5: PPS (↑) and CFI (↑) across datasets and pFL architectures. Mean ± std over 5 runs. Best results (excluding Retrain) are in **bold**. Δ denotes the improvement of pFedUL over the second-best baseline.

Method	FedPer		FedRep		Ditto		FedBN	
	PPS↑	CFI↑	PPS↑	CFI↑	PPS↑	CFI↑	PPS↑	CFI↑
CIFAR-10 (α_{dir} = 0.5)								
Fine-tune	0.941 ± 0.012	0.903 ± 0.018	0.946 ± 0.010	0.911 ± 0.015	0.952 ± 0.009	0.918 ± 0.014	0.934 ± 0.014	0.895 ± 0.021
GA-Naive	0.827 ± 0.021	0.784 ± 0.032	0.833 ± 0.019	0.791 ± 0.028	0.841 ± 0.018	0.802 ± 0.025	0.819 ± 0.024	0.771 ± 0.035
FedEraser-adapt.	0.862 ± 0.016	0.831 ± 0.023	0.869 ± 0.014	0.839 ± 0.020	0.878 ± 0.013	0.851 ± 0.019	0.853 ± 0.018	0.822 ± 0.026
pFedUL	**0.971** ± 0.006	**0.958** ± 0.008	**0.975** ± 0.005	**0.963** ± 0.007	**0.978** ± 0.004	**0.967** ± 0.006	**0.968** ± 0.007	**0.951** ± 0.009
Retrain	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000
Δ	+3.0%	+5.5%	+2.9%	+5.2%	+2.6%	+4.9%	+3.4%	+5.6%
CIFAR-100 (α_{dir} = 0.5)								
Fine-tune	0.928 ± 0.015	0.886 ± 0.022	0.932 ± 0.013	0.893 ± 0.019	0.939 ± 0.011	0.901 ± 0.017	0.921 ± 0.017	0.878 ± 0.025
GA-Naive	0.793 ± 0.028	0.741 ± 0.039	0.801 ± 0.025	0.752 ± 0.035	0.812 ± 0.023	0.768 ± 0.031	0.784 ± 0.031	0.729 ± 0.042
FedEraser-adapt.	0.841 ± 0.019	0.807 ± 0.027	0.849 ± 0.017	0.816 ± 0.024	0.858 ± 0.015	0.828 ± 0.022	0.832 ± 0.021	0.795 ± 0.030
pFedUL	**0.963** ± 0.008	**0.944** ± 0.011	**0.968** ± 0.007	**0.950** ± 0.009	**0.972** ± 0.006	**0.956** ± 0.008	**0.959** ± 0.009	**0.938** ± 0.012
Retrain	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000
Δ	+3.5%	+5.8%	+3.6%	+5.7%	+3.3%	+5.5%	+3.8%	+6.0%
FEMNIST (natural partition)								
Fine-tune	0.952 ± 0.009	0.921 ± 0.013	0.956 ± 0.008	0.928 ± 0.011	0.961 ± 0.007	0.934 ± 0.010	0.947 ± 0.011	0.914 ± 0.016
GA-Naive	0.851 ± 0.019	0.812 ± 0.027	0.858 ± 0.017	0.821 ± 0.024	0.866 ± 0.015	0.832 ± 0.021	0.843 ± 0.021	0.801 ± 0.030
FedEraser-adapt.	0.884 ± 0.014	0.856 ± 0.020	0.891 ± 0.012	0.864 ± 0.017	0.898 ± 0.011	0.873 ± 0.016	0.876 ± 0.016	0.845 ± 0.023
pFedUL	**0.978** ± 0.004	**0.966** ± 0.006	**0.981** ± 0.003	**0.971** ± 0.005	**0.984** ± 0.003	**0.975** ± 0.004	**0.974** ± 0.005	**0.960** ± 0.007
Retrain	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000	1.00 ± 0.000
Δ	+2.6%	+4.5%	+2.5%	+4.3%	+2.3%	+4.1%	+2.7%	+4.6%

Note: No-Unlearn is omitted from PPS/CFI comparison as it trivially achieves PPS = 1.0 by performing no unlearning. Δ is computed relative to the best-performing baseline (Fine-tune for PPS, Fine-tune for CFI).

The results in Table 5 reveal several findings. First, pFedUL achieves an average PPS of 0.973 across all dataset–architecture combinations, indicating that remaining clients retain approximately 97.3% of their original personalized accuracy after unlearning. Compared to the best baseline among the approximate methods (Fine-tune, average PPS of approximately 0.944 on CIFAR-10), pFedUL shows consistent improvements across all settings, with the improvement over FedEraser-adapted (average PPS of approximately

0.864 on CIFAR-10) being particularly pronounced. This supports the effectiveness of the layer-aware unlearning and recalibration protocol.

Second, the CFI scores suggest that pFedUL distributes the unlearning impact more uniformly across remaining clients. GA-Naive exhibits the lowest CFI values (0.73 to 0.80), indicating that its indiscriminate gradient ascent disproportionately harms certain clients, particularly those whose data distributions overlap more with the target client's data. pFedUL's selective approach mitigates this disparity by preserving low-contribution layers that may be important for specific clients.

Third, the standard deviations of pFedUL are consistently smaller than those of all baselines across all metrics, suggesting more stable and predictable unlearning outcomes. This stability likely stems from the principled Fisher-based attribution, which provides a data-driven basis for unlearning decisions rather than relying on uniform heuristics.

4.4 Comparison with State-of-the-Art FU Methods

To further evaluate pFedUL against existing FU methods, we compare it with six representative approaches adapted to the pFL setting. Since the basic baselines in Tables 4 and 5 already establish pFedUL's advantage over generic strategies, this section focuses on published FU methods that employ more sophisticated unlearning mechanisms. We use FedRep as the representative pFL architecture and report all four metrics across the three datasets. Results are averaged over 5 runs.

The results in Table 6 yield several observations. First, among the six adapted methods, the three that incorporate partial personalization awareness (ZeroFU, Mimir, FUSED) generally outperform the three that do not (PGA, KNOT, FedRecovery) in terms of PPS and CFI, suggesting that some degree of personalization consideration is beneficial. ZeroFU-adapted achieves the highest PPS among the existing methods (0.912 on CIFAR-10, 0.889 on CIFAR-100, 0.921 on FEMNIST), likely due to its client-specific conditional features that partially capture individual data distributions.

Table 6: Comparison with state-of-the-art FU methods adapted to the pFL setting (FedRep backbone). MIA Acc. (↓, closer to 0.50 is better), Rem. Acc. (↑), PPS (↑), and CFI (↑). Best results (excluding Retrain) are in **bold**. Methods marked with $^{\leqslant}$ incorporate partial personalization-aware components in their original designs. Results are averaged over 5 runs.

Method	CIFAR-10				CIFAR-100				FEMNIST			
	MIA↓	Rem.↑	PPS↑	CFI↑	MIA↓	Rem.↑	PPS↑	CFI↑	MIA↓	Rem.↑	PPS↑	CFI↑
PGA-adapt. [15]	50.4	74.1	0.838	0.795	50.6	46.3	0.808	0.759	50.5	83.9	0.863	0.826
KNOT-adapt. [10]	54.3	80.6	0.886	0.852	56.1	53.2	0.856	0.822	53.6	87.4	0.897	0.868
FedRecovery-adapt. [11]	52.5	79.3	0.873	0.838	54.8	52.1	0.843	0.809	52.3	86.8	0.889	0.857
ZeroFU-adapt.$^{\leqslant}$ [26]	52.1	81.8	0.912	0.879	53.4	55.1	0.889	0.852	51.8	88.6	0.921	0.893
Mimir-adapt.$^{\leqslant}$ [27]	53.7	80.1	0.894	0.861	55.2	53.8	0.862	0.829	53.1	87.1	0.901	0.872
FUSED-adapt.$^{\leqslant}$ [28]	52.4	81.3	0.905	0.871	53.9	54.6	0.881	0.845	52.1	88.1	0.915	0.886
pFedUL	**51.1**	**84.2**	**0.975**	**0.963**	**51.8**	**58.3**	**0.968**	**0.950**	**50.8**	**91.0**	**0.981**	**0.971**
Retrain	50.2	84.9	1.00	1.00	50.1	59.0	1.00	1.00	50.0	91.5	1.00	1.00

Second, pFedUL consistently outperforms all six adapted methods across all datasets and metrics. Compared to ZeroFU-adapted (the strongest existing method), pFedUL improves PPS by 6.3% on CIFAR-10, 7.9% on CIFAR-100, and 6.0% on FEMNIST, averaged across datasets, while simultaneously achieving lower MIA accuracy (closer to the ideal 0.5). This improvement is attributable to pFedUL's explicit modeling of the shared-personalized layer structure, whereas ZeroFU's conditional features only implicitly capture client-specific characteristics without formally distinguishing between shared and personalized parameters.

Third, PGA-adapted achieves MIA accuracy close to 0.5 (similar to its GA-Naive counterpart in Table 4), confirming that gradient ascent-based methods are effective at forgetting. However, the PGA mechanism does not prevent the severe personalization degradation observed in PPS (0.838 on CIFAR-10), as the projection operates on the global parameter space without awareness of the shared-personalized decomposition. KNOT-adapted and FedRecovery-adapted show moderate performance across all metrics but exhibit incomplete forgetting (MIA accuracy 52.3 to 56.1), as their original designs assume a single global model structure that does not align well with the pFL setting.

Fourth, the CFI gap between pFedUL and existing methods is particularly pronounced. The best existing method (ZeroFU-adapted) achieves CFI values of 0.879 to 0.893, while pFedUL reaches 0.950 to 0.971. This indicates that pFedUL's layer-aware selective unlearning distributes the forgetting impact more evenly across remaining clients, whereas adapted methods that do not explicitly account for the shared-personalized structure tend to cause uneven degradation.

4.5 Ablation Study

To validate the contribution of each component in pFedUL, we conduct an ablation study by systematically removing individual components. We evaluate four variants: the full pFedUL framework, *w/o Attribution* (replacing layer-wise contribution scores with uniform weights across all layers), *w/o Adaptive* (applying the same unlearning intensity to all high-contribution layers regardless of their individual scores), and *w/o Recalibration* (skipping the local fine-tuning stage for remaining clients). Experiments are conducted on CIFAR-10 with $\alpha_{\text{dir}} = 0.5$ under the FedRep backbone.

The ablation results in Fig. 2 indicate that each component plays a distinct and complementary role. Removing the Fisher-based attribution (w/o Attribution) causes the most notable PPS degradation (from 0.975 to 0.921), as uniform unlearning weights lead to unnecessary perturbation of layers where the target client had minimal influence. Removing the adaptive intensity mechanism (w/o Adaptive) primarily increases MIA accuracy (from 0.511 to 0.536), suggesting that applying uniform correction intensity to all high-contribution layers results in under-forgetting on the most critical layers. Removing the recalibration protocol (w/o Recalibration) reduces PPS from 0.975 to 0.948, confirming that even the selective unlearning strategy introduces some misalignment between shared and personalized layers that benefits from local fine-tuning. Notably, the MIA accuracy remains stable when recalibration is removed, consistent with the expectation that recalibration operates on personalized parameters only and does not compromise unlearning completeness.

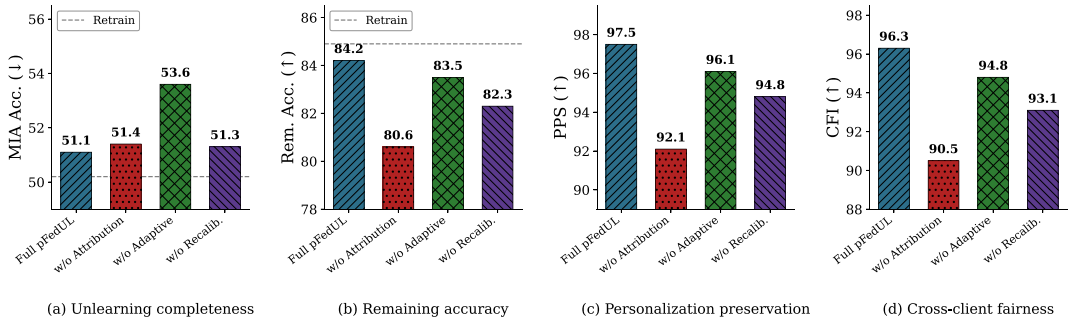

Figure 2: Ablation study on CIFAR-10 with FedRep ($\alpha_{\text{dir}} = 0.5$). (**a**) Unlearning completeness measured by MIA Acc. (↓). (**b**) Remaining accuracy (↑). (**c**) Personalization preservation measured by PPS (↑). (**d**) Cross-client fairness measured by CFI (↑). Removing any single component degrades at least one metric, suggesting that all three components contribute to the overall performance.

4.6 Attribution Validity and Client-Level Analysis

To directly validate whether the Fisher-based contribution scores reliably identify layers most associated with the target client's influence, we conduct a controlled experiment that compares three layer selection strategies for unlearning: (1) *High-attribution*, which applies unlearning only to the top-k layers ranked by $I_l^{c_t}$; (2) *Low-attribution*, which applies unlearning only to the bottom-k layers; and (3) *Random*, which applies unlearning to k randomly selected layers. We fix k to match the number of high-contribution layers identified by pFedUL's adaptive threshold ($\alpha = 0.5$), and use the same unlearning intensity (Eq. (9)) for all three strategies. This experiment is conducted on CIFAR-10 with FedRep, averaged over 5 runs.

The results in Table 7 provide direct evidence that the Fisher-based attribution scores reliably identify the layers most responsible for encoding the target client's data influence. Unlearning only the high-attribution layers achieves MIA accuracy of 51.1%, which is close to the ideal 0.5 and comparable to uniform all-layer unlearning (49.8%), while preserving substantially higher remaining accuracy (84.2% vs. 73.1%) and PPS (0.975 vs. 0.833). In contrast, unlearning only low-attribution layers results in MIA accuracy of 68.7%, indicating that the target client's data influence remains largely intact when low-contribution layers are modified. The random selection strategy falls between these two extremes. These results confirm that the target client's influence is concentrated in specific layers rather than uniformly distributed, and that the Fisher-based contribution score $I_l^{c_t}$ effectively identifies these critical layers.

Table 7: Validation of Fisher-based attribution on CIFAR-10 with FedRep ($\alpha_{\text{dir}} = 0.5$). Three layer selection strategies are compared using the same number of layers ($k = 6$ out of $L = 17$ shared layers) and the same unlearning intensity. MIA Acc. (↓, closer to 0.50 is better), Rem. Acc. (↑), PPS (↑). Mean ± std over 5 runs.

Layer Selection Strategy	MIA Acc. (↓)	Rem. Acc. (↑)	PPS (↑)
High-attribution (top-k by $I_l^{c_t}$)	51.1 ± 0.4	84.2 ± 0.3	0.975 ± 0.005
Low-attribution (bottom-k by $I_l^{c_t}$)	68.7 ± 1.2	81.3 ± 0.6	0.936 ± 0.011
Random (k random layers)	59.4 ± 2.1	79.8 ± 0.9	0.924 ± 0.016
All layers (uniform, $k = L$)	49.8 ± 0.3	73.1 ± 0.5	0.833 ± 0.019

To complement the averaged PPS and CFI metrics and address potential sensitivity to individual client performance variations, we report the per-client preservation ratio $A_i^{\text{post}}/A_i^{\text{pre}}$ distribution across all 19 remaining clients on CIFAR-10 with FedRep. For pFedUL, the per-client ratios range from 0.961 to 0.993 (interquartile range: 0.968–0.982), indicating that no individual client experiences more than a 3.9% accuracy reduction. In comparison, GA-Naive yields per-client ratios ranging from 0.742 to 0.923 (interquartile range: 0.793–0.871), with the most affected client losing over 25% of its personalized accuracy. FedEraser-adapted shows an intermediate range of 0.821 to 0.918. The narrow spread of pFedUL's per-client ratios confirms that the averaged PPS value of 0.975 is representative of all individual clients and is not inflated by a few well-preserved clients masking significant degradation in others.

4.7 Sensitivity Analysis

We analyze the sensitivity of pFedUL to three key factors: the threshold sensitivity parameter α, the degree of non-IID data heterogeneity, and the number of simultaneously unlearning clients. All experiments use CIFAR-10 with FedRep unless otherwise stated.

The sensitivity analysis in Fig. 3 reveals the following. For the threshold parameter α in Fig. 3a, increasing α from 0 to 0.5 steadily improves PPS (from 0.953 to 0.975) with only a marginal increase in MIA accuracy (from 0.505 to 0.511), suggesting that a moderate threshold effectively filters out low-contribution

layers without compromising forgetting quality. Beyond $\alpha = 1.0$, the PPS improvement plateaus while MIA accuracy begins to rise more steeply (to 0.538 at $\alpha = 1.5$), as too few layers are selected for unlearning. The default setting $\alpha = 0.5$ offers a favorable balance.

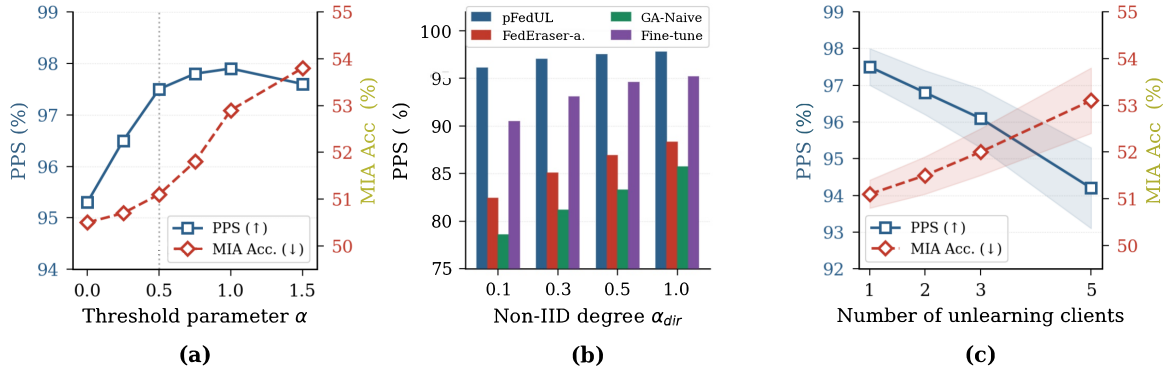

Figure 3: Sensitivity analysis on CIFAR-10 with FedRep. (**a**) Trade-off between unlearning completeness and personalization preservation controlled by the threshold parameter α. The vertical dashed line indicates the default setting ($\alpha = 0.5$) used in all other experiments. (**b**) Impact of non-IID heterogeneity on personalization preservation across methods. (**c**) Scalability of pFedUL as the number of simultaneously unlearning clients increases.

For non-IID heterogeneity in Fig. 3b, pFedUL maintains a PPS above 0.960 across all tested α_{dir} values, while baseline methods show increasing degradation under higher heterogeneity ($\alpha_{\text{dir}} = 0.1$). This robustness is expected: higher heterogeneity leads to stronger differentiation in layer-wise contributions, which makes the Fisher-based attribution more discriminative and the selective unlearning more precise.

For multi-client unlearning in Fig. 3c, pFedUL's PPS decreases gradually from 0.975 (1 client) to 0.961 (3 clients) and 0.942 (5 clients), while MIA accuracy remains within 2 percentage points of 0.5 up to 3 clients. This gradual degradation is inherent to the increased cumulative perturbation of shared parameters and is observed across all methods. pFedUL's advantage over baselines widens as the number of unlearning clients increases, since layer-aware selectivity becomes more important when multiple clients' contributions must be disentangled.

4.8 Efficiency Analysis

Table 8 compares the wall-clock time and communication overhead of all methods across the three datasets, using FedRep as the pFL backbone. As shown in Table 8, pFedUL achieves an 8.4× speedup over Retrain across all three datasets (measured on the FedRep backbone), with the efficiency advantage remaining consistent regardless of dataset complexity. This speedup arises from two factors. First, the attribution and selective unlearning stages (Stages 1 and 2) operate entirely on the server side using pre-computed Fisher statistics, requiring no client participation or iterative communication. Second, the recalibration stage (Stage 3) involves only a single broadcast of the updated shared parameters followed by 1 to 2 local epochs on each client, which is substantially less expensive than the 200 rounds of full retraining.

Compared to other approximate baselines, pFedUL offers a distinct advantage in communication cost: it requires zero global aggregation rounds, while GA-Naive and Fine-tune still require 20 and 30 rounds of iterative server-client communication, respectively. FedEraser-adapted is the slowest among the approximate methods (3.2 to 3.6× speedup) due to its dependence on iterative gradient calibration. Although GA-Naive achieves a comparable speedup (9.8 to 9.9×), its severe remaining accuracy and PPS degradation (shown in Tables 4 and 5) limits its practical usefulness. Combined with the results in Table 6, pFedUL is

the only method among those tested that simultaneously achieves strong unlearning completeness, high personalization preservation, and practical efficiency.

Table 8: Efficiency comparison on FedRep backbone. Time is measured in seconds for the complete unlearning procedure. Speedup is relative to Retrain. Comm. Best results among approximate unlearning methods (excluding Retrain) are highlighted in **bold**. Rounds denotes the number of global communication rounds required.

Method	CIFAR-10			CIFAR-100			FEMNIST		
	Time (s)	Speedup	Rounds	Time (s)	Speedup	Rounds	Time (s)	Speedup	Rounds
Retrain	4826	1.0×	200	5143	1.0×	200	3917	1.0×	200
FedEraser-adapt.	1438	3.4×	58	1592	3.2×	62	1089	3.6×	55
GA-Naive	486	9.9×	20	524	9.8×	20	398	9.8×	20
Fine-tune	762	6.3×	30	831	6.2×	30	615	6.4×	30
pFedUL	**574**	**8.4×**	**0***	**613**	**8.4×**	**0***	**468**	**8.4×**	**0***

Notes: *pFedUL requires zero global communication rounds for unlearning (Stages 1 and 2 are server-side operations). The recalibration stage (Stage 3) uses a single broadcast of $\hat{\theta}^s$ followed by local-only computation, which is not counted as a communication round.

4.9 Robustness under Challenging Scenarios

The experiments in Sections 4.2–4.8 evaluate pFedUL under the default setting where one randomly selected client with a typical data volume is removed. To more comprehensively assess robustness, we evaluate pFedUL under three additional challenging scenarios on CIFAR-10 with the FedRep backbone: (1) *imbalanced data volume*, where the target client holds a disproportionately large share of the total data; (2) *rare-class target client*, where the target client is the primary holder of certain minority classes that are underrepresented among other clients; and (3) *sequential unlearning*, where multiple clients are removed sequentially rather than simultaneously. We compare pFedUL against the two strongest baselines (ZeroFU-adapted and Fine-tune) and Retrain. Results are averaged over 5 runs.

The results in Table 9 reveal that pFedUL maintains strong performance across all three challenging scenarios, though with expected modest degradation compared to the default setting. In the *imbalanced* scenario, where the target client contributes 20% of the total data (4× the expected share under uniform partitioning), pFedUL achieves MIA accuracy of 52.6% and PPS of 0.958, compared to 51.1% and 0.975 in the default setting. The slightly higher MIA accuracy reflects the increased difficulty of removing a heavily contributing client, while the PPS remains high because the Fisher-based attribution effectively identifies the larger number of high-contribution layers associated with this client, allowing targeted correction without broad collateral damage. In contrast, ZeroFU-adapted shows MIA of 55.8% and PPS of 0.886, indicating both more incomplete forgetting and more severe personalization degradation.

In the *rare-class* scenario, where the target client is the dominant holder of two minority classes, pFedUL achieves PPS of 0.951 and MIA of 52.1%. The primary challenge in this scenario is that removing the target client can cause the shared feature extractor to lose discriminative capacity for the rare classes, potentially affecting remaining clients that also hold small numbers of these class samples. pFedUL mitigates this by limiting modifications to high-contribution layers, preserving the broader feature representation. The CFI remains high (0.932), indicating that the remaining clients' rare-class accuracy does not degrade disproportionately.

Table 9: Robustness evaluation under challenging scenarios on CIFAR-10 with FedRep. *Imbalanced*: the target client holds 20% of the total training data (4× the average). *Rare-class*: the target client holds >80% of samples from 2 rare classes. *Sequential*: 3 clients are removed one after another (metrics reported after the final removal). MIA Acc. (↓), PPS (↑), CFI (↑). Best results among approximate unlearning methods (excluding Retrain) are highlighted in **bold**. Mean over 5 runs.

Method	Imbalanced			Rare-Class			Sequential		
	MIA↓	PPS↑	CFI↑	MIA↓	PPS↑	CFI↑	MIA↓	PPS↑	CFI↑
Fine-tune	62.3	0.918	0.872	61.5	0.901	0.856	63.8	0.892	0.841
ZeroFU-adapt.	55.8	0.886	0.843	54.6	0.873	0.831	57.1	0.861	0.818
pFedUL	**52.6**	**0.958**	**0.939**	**52.1**	**0.951**	**0.932**	**53.4**	**0.943**	**0.921**
Retrain	50.3	1.00	1.00	50.1	1.00	1.00	50.2	1.00	1.00

In the *sequential* scenario, where three clients are removed one after another with a full pFedUL unlearning procedure applied at each step, the cumulative effect on PPS (0.943) and MIA (53.4%) is modest. Each sequential removal builds upon the updated $\hat{\theta}^s$ from the previous step, and the Fisher statistics are recalculated to reflect the reduced client pool before each subsequent unlearning. The gradual PPS degradation is consistent with the multi-client results in Fig. 3c and arises from the cumulative perturbation of shared parameters. Importantly, the sequential procedure does not exhibit compounding instability, as each step operates independently on fresh attribution scores. This suggests that pFedUL can support practical deployment scenarios where unlearning requests arrive over time, though very long sequences of removals may eventually necessitate periodic full retraining to restore optimal shared representations.

5 Discussion

5.1 Implications

The results presented in this paper carry implications for the deployment of FL systems in privacy-sensitive environments. Our finding that existing FU methods—including recent approaches with partial personalization awareness such as ZeroFU, Mimir, and FUSED—do not adequately preserve personalization when adapted to pFL architectures exposes a notable gap in current research. As pFL methods increasingly replace FedAvg in practical deployments due to their superior handling of data heterogeneity, the inability to perform effective unlearning under these architectures represents a potential regulatory compliance risk. pFedUL addresses this gap by showing that layer-aware unlearning can simultaneously satisfy the right to be forgotten and maintain service quality for remaining participants, contributing toward making privacy-preserving personalized FL more practically viable.

Beyond the immediate technical contributions, the conceptual framework introduced in this work—namely the decomposition of unlearning into shared-layer forgetting and personalized-layer preservation—offers a useful lens for understanding the interplay between collaborative knowledge and individual adaptation in distributed learning systems. The two proposed metrics, PPS and CFI, formalize evaluation dimensions that prior work has not explicitly addressed, and we anticipate they may be useful in future FU benchmarks. In particular, the CFI metric highlights the importance of fairness in unlearning, ensuring that the cost of one client's departure is not disproportionately borne by specific remaining clients. This fairness perspective connects FU to broader discussions on equitable machine learning, where the actions of individual participants should not unfairly impact others in the system.

5.2 Limitations and Future Work

Despite the effectiveness of pFedUL, several limitations should be acknowledged. First, the Fisher-based attribution mechanism requires storing per-client, per-layer gradient statistics throughout the training process, introducing additional memory overhead on the server. While this overhead is modest relative to storing full model checkpoints (as required by FedEraser), it may become a concern in extremely large-scale federations with thousands of clients and very deep models. Moreover, the storage of per-client Fisher statistics on the server introduces potential privacy considerations that merit discussion. Although the stored quantities are diagonal Fisher information vectors (i.e., aggregated squared gradient norms) rather than raw gradients or data, they nonetheless encode information about each client's data distribution, specifically which parameters are most sensitive to that client's data. In principle, an adversary with access to these statistics could infer certain characteristics of a client's data distribution, such as which classes dominate the local dataset, by analyzing which layers exhibit high Fisher values. Several mitigation strategies can be adopted to address this risk: (a) applying calibrated noise to the Fisher statistics before storage, following differential privacy principles, at the cost of slightly reduced attribution precision; (b) deleting all per-client Fisher statistics immediately after the unlearning procedure completes, retaining only the aggregate Fisher across remaining clients for potential future use; and (c) computing and storing Fisher statistics only for the shared layers rather than the full model, since personalized parameters are not subject to attribution. We note that this privacy consideration is not unique to pFedUL—any gradient-calibration-based FU method (e.g., FedEraser) faces analogous or greater exposure by storing full historical gradient updates—but it warrants explicit acknowledgment in the context of regulatory compliance, which is a primary motivation for FU. Second, the current framework assumes that the target client's identity is known and that unlearning occurs after training has converged. Extending pFedUL to support unlearning during ongoing training or handling sequential unlearning requests from multiple clients over time remains an open challenge, though the sequential unlearning experiment in Section 4.9 provides initial evidence that pFedUL can handle sequential removals without compounding instability. Third, our evaluation focuses on classification tasks with convolutional architectures, and the generalizability of the layer-wise contribution patterns to other tasks (e.g., natural language processing, generative models) and architectures (e.g., transformers) warrants further investigation.

Several promising directions emerge from this work. First, integrating pFedUL with formal verification mechanisms [18] could enable certified unlearning guarantees under the pFL paradigm, bridging the gap between approximate and provably correct forgetting. Second, extending the framework to support vertical FL, where features rather than samples are partitioned across clients, would require rethinking the notion of layer-wise contributions and is a natural next step. Third, the emergence of federated foundation models and large language models introduces new challenges for unlearning, as the scale and complexity of these models amplify both the importance and the difficulty of selective forgetting. Adapting the layer-aware principle of pFedUL to parameter-efficient fine-tuning paradigms such as LoRA and prompt tuning represents a timely research direction. Finally, investigating the theoretical connections between the completeness-preservation trade-off identified in this paper and the broader privacy-utility trade-off in differential privacy could yield deeper insights into the fundamental limits of FU.

6 Conclusion

In this paper, we identified a critical yet largely overlooked challenge in FL: most existing FU methods are designed for the FedAvg paradigm and do not account for the structural decomposition inherent in pFL architectures. To address this gap, we formalized the FU problem under the pFL paradigm and revealed a fundamental tension between unlearning completeness on shared layers and personalization

preservation for remaining clients. We proposed pFedUL, a layer-aware selective unlearning framework that integrates three synergistic components: Fisher-based layer-wise contribution attribution, adaptive selective unlearning with differentiated intensity across layer types, and a lightweight recalibration protocol for restoring personalized performance. We further introduced two evaluation metrics, PPS and CFI, to capture pFL-specific unlearning quality dimensions not addressed by existing benchmarks. Experiments on CIFAR-10, CIFAR-100, and FEMNIST across four mainstream pFL architectures (FedPer, FedRep, Ditto, and FedBN) indicate that pFedUL achieves unlearning effectiveness comparable to full retraining while maintaining an average of 97.3% personalized accuracy for remaining clients across all tested settings. Compared with six state-of-the-art FU methods adapted to the pFL setting, pFedUL consistently achieves superior personalization preservation and cross-client fairness, with an 8.4× speedup and zero global communication rounds.

Acknowledgement: Not applicable.

Funding Statement: This research received no external funding.

Author Contributions: Conceptualization: Zhuodong Liu, Xiangyu Li and Zhihao Zhang; Methodology: Zhuodong Liu and Xiangyu Li; Software: Zhuodong Liu and Zhihao Zhang; Validation: Zhuodong Liu and Zhihao Zhang; Formal Analysis: Zhuodong Liu, Xiangyu Li and Zhihao Zhang; Investigation: Zhuodong Liu, Xiangyu Li and Zhihao Zhang; Data Curation: Zhuodong Liu and Xiangyu Li; Writing—Original Draft Preparation: Zhuodong Liu and Zhihao Zhang; Visualization: Zhuodong Liu and Zhihao Zhang; Writing—Review and Editing: Zhuodong Liu and Xiangyu Li; Supervision, Xiangyu Li; Project Administration, Xiangyu Li. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Dataset available on request from the authors.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. McMahan B, Moore E, Ramage D, Hampson S, Arcas BA. Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS); 2017 Apr 20–22; Fort Lauderdale, FL, USA. p. 1273–82.
2. Kairouz P, McMahan HB, Avent B, Bellet A, Bennis M, Bhagoji AN, et al. Advances and open problems in federated learning. Found Trends Mach Learn. 2021;14(1–2):1–210. doi:10.1561/2200000083.
3. Li T, Sahu AK, Talwalkar A, Smith V. Federated learning: challenges, methods, and future directions. IEEE Signal Process Mag. 2020;37(3):50–60. doi:10.1109/MSP.2020.2975749.
4. European Parliament, Council of the European Union. Regulation (EU) 2016/679 of the European parliament and of the council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data. Off J Eur Union. 2016;L119:1–88.
5. Bourtoule L, Chandrasekaran V, Choquette-Choo CA, Jia H, Travers A, Zhang B, et al. Machine unlearning. In: Proceedings of the 2021 IEEE Symposium on Security and Privacy (S&P); 2021 May 24–27; San Francisco, CA, USA. p. 141–59. doi:10.1109/SP40001.2021.00019.
6. Koh PW, Liang P. Understanding black-box predictions via influence functions. In: Proceedings of the 34th International Conference on Machine Learning (ICML); 2017 Aug 6–11; Sydney, Australia. p. 1885–94.
7. Xu H, Zhu T, Zhang L, Zhou W, Yu PS. Machine unlearning: a survey. ACM Comput Surv. 2023;56(1):9. doi:10.1145/3603620.

8. Romandini N, Mora A, Mazzocca C, Montanari R, Bellavista P. Federated unlearning: a survey on methods, design guidelines, and evaluation metrics. IEEE Trans Neural Netw Learn Syst. 2024;35(12):17513–30. doi:10.1109/TNNLS.2024.3478334.
9. Liu G, Ma X, Yang Y, Liu J. FedEraser: enabling efficient client-level data removal from federated learning. In: Proceedings of the 2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQoS); 2021 Jun 25–28; Tokyo, Japan. p. 1–10. doi:10.1109/IWQOS52092.2021.9521274.
10. Su L, Li J. Asynchronous federated unlearning. In: Proceedings of the IEEE Conference on Computer Communications (INFOCOM); 2023 May 17–20; New York, NY, USA. p. 1–10. doi:10.1109/INFOCOM53939.2023.10228894.
11. Zhang L, Zhu T, Calandrini P, Yin J. FedRecovery: differentially private machine unlearning for federated learning frameworks. IEEE Trans Inf Forensics Secur. 2023;18:4732–46. doi:10.1109/TIFS.2023.3297905.
12. Gu H, Zhu G, Zhang J, Zhao X, Han Y, Fan L, et al. Unlearning during learning: an efficient federated machine unlearning method. In: Proceedings of the 33rd International Joint Conference on Artificial Intelligence (IJCAI); 2024 Aug 3–9; Jeju, Republic of Korea. p. 4035–43. doi:10.24963/ijcai.2024/446.
13. Pan Z, Wang Z, Li C, Zheng K, Wang B, Tang X, et al. Federated unlearning with gradient descent and conflict mitigation. In: Proceedings of the 39th AAAI Conference on Artificial Intelligence; 2025 Feb 25–Mar 4; Philadelphia, PA, USA. p. 19804–12. doi:10.1609/aaai.v39i19.34181.
14. Wang J, Guo S, Xie X, Qi H. Federated unlearning via class-discriminative pruning. In: Proceedings of the ACM Web Conference (WWW); 2022 Apr 25–29; Lyon, France. p. 622–32. doi:10.1145/3485447.3512222.
15. Halimi A, Kadhe S, Rawat A, Baracaldo N. Federated unlearning: how to efficiently erase a client in FL? arXiv:2207.05521. 2022. doi:10.48550/arxiv.2207.05521.
16. Gu H, Ong WK, Chan CS, Fan L. Ferrari: federated feature unlearning via optimizing feature sensitivity. In: Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS); 2024 Dec 10–15; Vancouver, BC, Canada. p. 1–12. doi:10.52202/079017-0761.
17. Jeong G, Ma K. A survey on federated unlearning: challenges and opportunities. IEEE Trans Big Data. 2026;12(3):702–20. doi:10.1109/tbdata.2026.3668538.
18. Gao Y, Liu Y, Xiong H, Han Z. VeriFi: towards verifiable federated unlearning. IEEE Trans Dependable Secure Comput. 2024;21(5):4236–49. doi:10.1109/TDSC.2024.3354630.
19. Li T, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A, Smith V. Federated optimization in heterogeneous networks. In: Proceedings of Machine Learning and Systems (MLSys); 2020 Mar 2–4; Austin, TX, USA. p. 429–50.
20. Tan AZ, Yu H, Cui L, Yang Q. Towards personalized federated learning. IEEE Trans Neural Netw Learn Syst. 2023;34(12):9587–603. doi:10.1109/TNNLS.2022.3160699.
21. Arivazhagan MG, Aggarwal V, Singh AK, Choudhary S. Federated learning with personalization layers. arXiv:1912.00818. 2019.
22. Collins L, Hassani H, Mokhtari A, Shakkottai S. Exploiting shared representations for personalized federated learning. In: Proceedings of the 38th International Conference on Machine Learning (ICML); 2021 Jul 18–24; Virtual. p. 2089–99.
23. Li T, Hu S, Beirami A, Smith V. Ditto: fair and robust federated learning through personalization. In: Proceedings of the 38th International Conference on Machine Learning (ICML); 2021 Jul 18–24; Virtual. p. 6357–68.
24. Li X, Jiang M, Zhang X, Kamp M, FedBN DT. FedBN: federated learning on non-IID features via local batch normalization. In: Proceedings of the 9th International Conference on Learning Representations (ICLR); 2021 May 3–7; Virtual. p. 1–16.
25. Zhang J, Hua Y, Wang H, Song T, Xue Z, Ma R, et al. FedALA: adaptive local aggregation for personalized federated learning. In: Proceedings of the 37th AAAI Conference on Artificial Intelligence; 2023 Feb 7–14; Washington, DC, USA. p. 11237–44. doi:10.1609/aaai.v37i9.26330.
26. Wu W, Liang H, Yuan J, Jiang J, Wang KY, Hu C, et al. Zero-shot federated unlearning via transforming from data-dependent to personalized model-centric. In: Proceedings of the 34th International Joint Conference on Artificial Intelligence (IJCAI); 2025 Aug 16–22; Montreal, QC, Canada. p. 6588–96. doi:10.24963/ijcai.2025/733.

27. Wu W, Liang H, Tu T, Jiang J, Hu C, Mimir C D. Data-free federated unlearning through client-specific prompt generation for personalized models. IEEE Trans Mob Comput. 2025;24(10):10537–56. doi:10.1109/TMC.2025.3570018.
28. Zhong Z, Bao W, Wang J, Zhang S, Zhou J, Lyu L, et al. 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); 2025 Jun 10–17; Nashville, TN, USA. p. 30661–70. doi:10.1109/CVPR52734.2025.02855.
29. Nguyen TT, Huynh TT, Nguyen PL, Liew AWC, Yin H, Nguyen QVH. A survey of machine unlearning. ACM Comput Surv. 2024;56(12):1–37. doi:10.1145/3749987.
30. Sekhari A, Acharya J, Kamath G, Suresh AT. Remember what you want to forget: algorithms for machine unlearning. In: Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS); 2021 Dec 6–14; Virtual. p. 18075–86.
31. Thudi A, Deza G, Chandrasekaran V, Papernot N. Unrolling SGD: understanding factors of influence in machine unlearning. In: Proceedings of the 2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P); 2022 Jun 6–10; Genoa, Italy. p. 303–19. doi:10.1109/eurosp53844.2022.00027.
32. Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, et al. Overcoming catastrophic forgetting in neural networks. Proc Natl Acad Sci U S A. 2017;114(13):3521–6. doi:10.1073/pnas.1611835114.
33. Liu Y, Xu L, Yuan X, Wang C, Li B. The right to be forgotten in federated learning: an efficient realization with rapid retraining. In: Proceedings of the IEEE Conference on Computer Communications (INFOCOM); 2022 May 2–5; London, UK. p. 1749–58. doi:10.1109/INFOCOM48880.2022.9796721.
34. Pan C, Sima J, Prakash S, Rana V, Milenkovic O. Machine unlearning of federated clusters. In: Proceedings of the 11th International Conference on Learning Representations (ICLR); 2023 May 1–5; Kigali, Rwanda. p. 1–27.
35. Khalil YH, Brunswic L, Lamghari S, Li X, Beitollahi M, Chen X. Federated unlearning via weight negation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2025 Jun 10–17; Nashville, TN, USA. p. 30661–70. doi:10.1109/cvpr52734.2025.02399.
36. Che T, Zhou Y, Zhang Z, Lyu Y, Liu J, Yan D, et al. Fast federated machine unlearning with nonlinear functional theory. In: Proceedings of the 40th International Conference on Machine Learning (ICML); 2023 Jul 23–29; Honolulu, HI, USA. p. 4241–68.
37. Yuan W, Yin H, Wu F, Zhang S, He T, Wang H. Federated unlearning for on-device recommendation. In: Proceedings of the 16th ACM International Conference on Web Search and Data Mining (WSDM); 2023 Feb 27–Mar 3; Singapore. p. 393–401. doi:10.1145/3539597.3570462.
38. Fraboni Y, Waerebeke MV, Scaman K, Vidal R, Kameni L, Lorenzi M. SIFU: sequential informed federated unlearning: efficient and provable client unlearning in federated optimization. In: Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS); 2024 May 2–4; Valencia, Spain. p. 1–12
39. Li Q, Diao Y, Chen Q, He B. Federated learning on non-IID data silos: an experimental study. In: Proceedings of the 38th IEEE International Conference on Data Engineering (ICDE); 2022 May 9–12; Malaysia: Kuala Lumpur. p. 965–78. doi:10.1109/ICDE53745.2022.00077.
40. Chen HY, Chao WL. On bridging generic and personalized federated learning for image classification. arXiv:2107.00778. 2022.
41. T.Dinh C, Tran NH, Nguyen TD. Personalized federated learning with Moreau envelopes. In: Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS); 2020 Dec 6–12; Virtual. p. 21394–405. doi:10.1109/icme59968.2025.11209227.
42. Shokri R, Stronati M, Song C, Shmatikov V. Membership inference attacks against machine learning models. In: Proceedings of the 2017 IEEE Symposium on Security and Privacy (S&P); 2017 May 22–26; San Jose, CA, USA. p. 3–18. doi:10.1109/SP.2017.41.
43. Caldas S, Duddu SMK, Wu P, Li T, Konecny J, McMahan HB, et al. LEAF: a benchmark for federated settings. arXiv:1812.01097. 2019.
44. Hsu TMH, Qi H, Brown M. Measuring the effects of non-independently and identically distributed data on federated learning. arXiv:2009.09890. 2022.