# PULSAR: WHO-FORGETS-WHOM PERSONALIZED UNLEARNING FOR FEDERATED LARGE LANGUAGE MODELS

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

## ABSTRACT

Federated unlearning (FU) has largely focused on producing a single unlearned global model, implicitly enforcing "forgetting-for-all" whenever any client requests deletion. In realistic federated deployments, however, data and model-sharing permissions are often recipient-specific and may change over time, requiring selective revocation of one client's contribution from a subset of other clients. We propose PULSAR, the first *who-forgets-whom* personalized FU framework that induces recipient-specific behavioral forgetting of designated source data in the models delivered to selected target clients, while preserving the model utility of non-target clients that remain authorized or cooperative. Building on parameter-efficient federated fine-tuning, PULSAR constructs a portable, plug-and-play adapter-style knowledge filter from an unlearning operation and applies it at model dispatch to enforce recipient-specific forgetting. To mitigate utility degradation on target clients, we further introduce a joint optimization procedure that balances effective unlearning and utility retention. Single-round experimental results on the TOFU benchmark demonstrate that PULSAR achieves a similar forgetting performance (FTR metric) on targeted clients to the *Retrain* baseline across multiple LLM backbones and FU settings, while maintaining similar model utility (MU metric) on non-target clients to the *Finetune* baseline.

## 1 INTRODUCTION

General Data Protection Regulation (GDPR) grants data subjects the right to erasure (i.e., the "right to be forgotten"). However, the implicit and distributed nature of knowledge representation within Large Language Model (LLM) parameters makes it challenging to localize and remove the influence of specific training data once model training is complete. Machine unlearning has therefore emerged as an active research area, aiming to eliminate the influence of particular data samples from LLM parameters without incurring the prohibitive cost of retraining the model from scratch, while preserving the utility of the remaining knowledge. Despite recent advances, most existing unlearning frameworks (Liu et al., 2025; Chundawat et al., 2023) assume centralized access to training data. This assumption is misaligned with practical deployments, where strict privacy regulations and commercial considerations (Thirunavukarasu et al., 2023; Huang et al., 2023) often result in data for LLM training and fine-tuning (FT) being distributed across multiple organizations.

In this context, Federated Learning (FL) (McMahan et al., 2017) and Federated Unlearning (FU) (Liu et al., 2024) offer a distributed paradigm in which multiple clients can train a global model without sharing raw data or removing knowledge from it. Recent studies, such as Oblivionis (Zhang et al., 2025), have begun integrating unlearning mechanisms into federated LLM pipelines. However, most existing approaches still assume indiscriminate global unlearning (i.e., "forgetting-for-all"), where forgetting is applied uniformly to all participants and thus lacks client-level specificity. In practical collaborative ecosystems, unlearning requirements are often asymmetric: a *source client* may require its contribution to be removed only from *target clients* (e.g., after licensing is revoked or partnerships deteriorate), while preserving collaboration with the rest of the federation. Enabling such selective, across-client targeted unlearning remains a significant open challenge.



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

Prior studies (Tan et al., 2024; Kostick-Quenet et al., 2025; Xu et al., 2021) have examined changes in data consent and collaboration–competition dynamics in federations, but they emphasize the expansion of authorization (e.g., widening access or admitting new participants). In contrast, authorization contraction, where access is restricted or revoked, remains underexplored despite its relevance. We highlight two contraction cases: ***(i)*** tightening data consent, where a data subject revises consent (e.g., to "non-commercial use only"), requiring commercial participants to remove the subject's data contribution from the collaboratively trained model; and ***(ii)*** participant withdrawal, where deteriorated or terminated collaboration (e.g., due to intensified competition), requires designated counterpart clients to remove source clients' contribution. In such scenarios, the federation should revoke access and enforce forgetting only for target-specific recipients, rather than applying global unlearning to all clients, causing unnecessary utility loss for the remaining authorized clients.

These motivate three research questions (RQs) that guide our study. **RQ1**: Unlike conventional FU, which produces an unlearned global model for "forgetting-for-all" (as shown in Figure 1 (a) and (b)), how can we achieve fine-grained control of targeted unlearning to induce source-specific forgetting behavior for designated target clients while preserving the corresponding contribution for authorized clients? **RQ2**: Given that deployments involve evolving consent and participation policies (e.g., revocations, withdrawals, and re-authorizations), how can an unlearning mechanism support updates efficiently and enable recoverable, plug-and-play enforcement of targeted forgetting for the intended recipients, without incurring prohibitive retraining or communication overhead? **RQ3**: Since removing one client's influence can degrade downstream performance and create distributional or behavioral divergence between target and non-target clients, how can solutions preserve task utility for targets and maintain stable collaboration with the rest of the federation under partial forgetting?

To address these questions, we propose a lightweight **P**ersonalized **U**n**L**earning with **S**elective **A**dapter-based **R**emoval (PULSAR) mechanism that consists of: (i) a source-derived filter constructed from the parameter displacement induced by an underlying unlearning operator to forget specified knowledge, and (ii) a local joint dual-objective optimization (JDOO) module to balance unlearning efficacy and model utility. We adopt Low-Rank Adaptation (LoRA) (Hu et al., 2022) for parameter-efficient FT, which updates only a small subset of adapter weights, achieving near-full FT performance while significantly reducing both computational and communication costs in unlearning. The main contributions of this paper are as follows:

- We propose PULSAR, the first *who-forgets-whom* personalized FU framework for LLMs, where designated target clients exhibit behavioral forgetting of source-specified data while source and authorized clients retain the unfiltered behavior. This role-aware formulation moves beyond conventional "forgetting-for-all" FU by enabling fine-grained source-to-target unlearning with limited impact on non-target clients. (**RQ1**)
- We design reusable adapter-style knowledge filters that encode source-specific unlearning effects and are coordinated through a persistent source–target map, recipient-specific composition and dispatch, and initialization-relative aggregation. This design supports plug-and-play activation, revocation, and recovery under evolving authorization policies without repeated retraining (**RQ2**). We further introduce target-side JDOO to mitigate filtering-induced utility loss while preserving the intended forgetting effect (**RQ3**).
- We evaluate PULSAR across multiple LLM backbones and FU configurations. Across the evaluated TOFU settings, PULSAR achieves target-side FTR/FOR close to the Retrain reference while preserving non-target utility close to the Finetune reference. Source-to-target and ablation analyses further support its targeted, recoverable behavior and the contribution of its core components.

## 2 System Model

We consider a federated system consisting of a server $\mathcal{S}$ and a set of $K$ clients $\mathcal{C} = \{k\}_{k=1}^{K}$. $\mathcal{S}$ acts as the coordinator, responsible for model aggregation and distribution. Each client $k$ holds a private dataset $\mathcal{D}_k = \{(x_i, y_i)\}_{i=1}^{N_k}$, where $x_i$ and $y_i$ are token sequences representing (input, output) or (prompt, response) pairs. These token sequences are used to fine-tune an LLM with parameters $\boldsymbol{\theta}_k \in \mathbb{R}^d$, where $d$ is the dimensionality of the model parameters. $N_k = |\mathcal{D}_k|$ is the number of data samples. $\mathcal{D} = \bigcup_{k=1}^{K} \mathcal{D}_k$ denotes the conceptual union of all clients' datasets, which is not directly accessible due to federated constraints. At each communication round $t$, $\mathcal{S}$ randomly samples a subset of clients $\mathcal{C}^t \subseteq \mathcal{C}$ for federated collaboration with sampling ratio $\lambda$, where $|\mathcal{C}^t| = \lambda|\mathcal{C}|$.

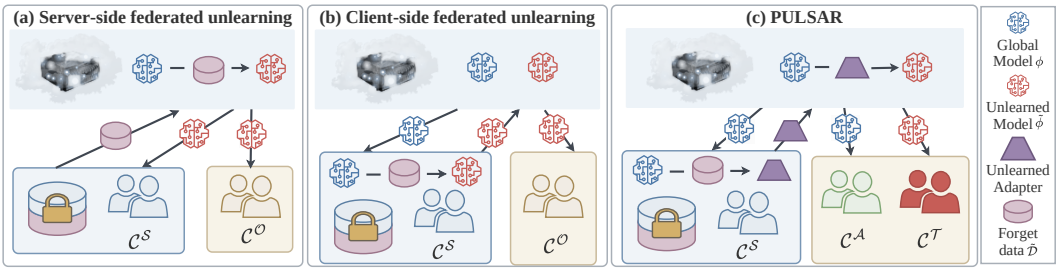



Figure 1: System Model. **(a) Server-side FU.** $\mathcal{C}^{\mathcal{S}}$ sends $\widetilde{\mathcal{D}}$ to $\mathcal{S}$ as an unlearning request. $\mathcal{S}$ applies the unlearning operator to $\phi$ and broadcasts $\widetilde{\phi}$ to all clients. **(b) Client-side FU.** $\mathcal{C}^{\mathcal{S}}$ locally applies the unlearning operator to the received $\phi$ on $\widetilde{\mathcal{D}}$, and sends the unlearned $\widetilde{\phi}$ to $\mathcal{S}$ as an unlearning request. $\mathcal{S}$ then broadcasts $\widetilde{\phi}$ to all clients. **(c) PULSAR.** $\mathcal{C}^{\mathcal{S}}$ raises an unlearning request by training an unlearning adapter on $\widetilde{\mathcal{D}}$, then uploads it to $\mathcal{S}$ to derive $\widetilde{\phi}$. $\mathcal{S}$ multicasts $\phi$ to $\mathcal{C}^{\mathcal{S}}$ and $\mathcal{C}^{\mathcal{A}}$, and $\widetilde{\phi}$ to $\mathcal{C}^{\mathcal{T}}$.

### 2.1 LoRA-based Federated Fine-tuning

We instantiate federated fine-tuning with LoRA (Hu et al., 2022). For client $k$ at communication round $t$, the trainable adapter parameters are denoted by $\boldsymbol{\phi}_k^t = \{\mathbf{A}_k^t, \mathbf{B}_k^t\}$, where the frozen pre-trained weight matrix $\mathbf{W}$ is adapted as

$$\mathbf{W}_k^t = \mathbf{W} + \Delta\mathbf{W}_k^t, \qquad \Delta\mathbf{W}_k^t = \mathbf{A}_k^t\mathbf{B}_k^t. \tag{1}$$

The corresponding model parameters are $\boldsymbol{\theta}_k^t = \boldsymbol{\theta}_{\text{base}} + \boldsymbol{\phi}_k^t$. Only the adapter parameters are updated during federated fine-tuning. For standard single-objective optimization (SOO), client $k$ minimizes its local fine-tuning loss $\mathcal{L}_k$ on $\mathcal{D}_k$. The federated objective and global aggregation are

$$\mathcal{L}^{\text{FT}} = \sum_{k=1}^{K} w_k\, \mathcal{L}_k(\boldsymbol{\phi}_k^t; \mathcal{D}_k), \qquad w_k = \frac{N_k}{\sum_{i=1}^{K} N_i}, \tag{2}$$

and

$$\boldsymbol{\phi}^{t+1} = \mathcal{A}\left(\boldsymbol{\phi}^t, \left\{\boldsymbol{\phi}_k^{(t,R)}, \mathcal{D}_k\right\}_{k\in\mathcal{C}^t}\right), \tag{3}$$

where $\mathcal{A}$ denotes a generic federated aggregation operator. Detailed local optimization and autoregressive training objectives are provided in Appendix A.1.

### 2.2 Federated Unlearning

Conventional FU paradigms, as shown in Figure 1 (a) and (b), divide sampled clients $\mathcal{C}$ into two groups: source clients $\mathcal{C}^{\mathcal{S}}$ (aka forgetting requester) and other clients $\mathcal{C}^{\mathcal{O}}$, i.e., $\mathcal{C} = \mathcal{C}^{\mathcal{S}} \cup \mathcal{C}^{\mathcal{O}}$. Each source client $m \in \mathcal{C}^{\mathcal{S}}$ holds a forget set $\widetilde{\mathcal{D}}_m$ (e.g., privacy-sensitive or commercially-confidential data) where $\widetilde{\mathcal{D}}_m \subseteq \mathcal{D}_m$. $\mathcal{C}^{\mathcal{O}}$ denotes the set of non-source clients that retain the implicit influence of $\mathcal{C}^{\mathcal{S}}$'s data contributions in their model parameters due to federated collaboration. When a source client $m$ requests to forget $\widetilde{\mathcal{D}}_m$, regardless of whether the unlearning procedure is performed on the server $\mathcal{S}$ (Figure 1 (a)) or locally (Figure 1 (b)), the objective is to derive unlearned global model parameters $\widetilde{\boldsymbol{\phi}}^t$ that approximate a model $\widetilde{\boldsymbol{\phi}}^{t,*}$ trained from scratch without $\widetilde{\mathcal{D}}_m$ as follows:

$$\widetilde{\boldsymbol{\phi}}^t = \mathcal{U}\left(\boldsymbol{\phi}^t, \widetilde{\mathcal{D}}_m\right), \tag{4}$$

$$\mathcal{L}^{\text{FT}}(\widetilde{\boldsymbol{\phi}}^{t,*}) = \sum_{k=1}^{K} w_k\, \mathcal{L}_k\left(\widetilde{\boldsymbol{\phi}}^{t,*};\, \mathcal{D}_k \backslash \widetilde{\mathcal{D}}_m\right). \tag{5}$$

where $\mathcal{U}$ denotes the unlearning operator, which utilizes a general unlearning approach. **However**, this mechanism is indiscriminate: it removes the influence of $\widetilde{\mathcal{D}}_m$ from the single global model $\boldsymbol{\phi}$ and thus propagates the forgetting effect to all local models $\{\boldsymbol{\phi}_k\}_{k=1}^K$, including the source client $m$.

### 2.3 Personalized Targeted Unlearning (PULSAR)

To enable personalized targeted *who-forgets-whom* unlearning, we further partition $\mathcal{C}^{\mathcal{O}}$ into a target-client (forgetting receiver) set $\mathcal{C}^{\mathcal{T}}$ and an authorized-client set $\mathcal{C}^{\mathcal{A}}$, i.e., $\mathcal{C}^{\mathcal{O}} = \mathcal{C}^{\mathcal{T}} \cup \mathcal{C}^{\mathcal{A}}$, as shown in

Figure 1 (c). Here, $\mathcal{C}^{\mathcal{T}} \subseteq \mathcal{C} \setminus \mathcal{C}^{\mathcal{S}}$, and $\mathcal{C}^{\mathcal{A}} = \mathcal{C} \setminus (\mathcal{C}^{\mathcal{S}} \cup \mathcal{C}^{\mathcal{T}})$. The three subsets $\mathcal{C}^{\mathcal{S}}$, $\mathcal{C}^{\mathcal{T}}$, and $\mathcal{C}^{\mathcal{A}}$ are pairwise disjoint, i.e., $\mathcal{C}^{\mathcal{S}} \cap \mathcal{C}^{\mathcal{T}} = \emptyset$, $\mathcal{C}^{\mathcal{S}} \cap \mathcal{C}^{\mathcal{A}} = \emptyset$, and $\mathcal{C}^{\mathcal{T}} \cap \mathcal{C}^{\mathcal{A}} = \emptyset$. We induce source-specific forgetting behavior on $\mathcal{C}^{\mathcal{T}}$, and aim to minimize model performance degradation in $\mathcal{C}^{\mathcal{A}}$ and $\mathcal{C}^{\mathcal{S}}$.

## 3 Proposed PULSAR Framework

This section presents PULSAR, a dispatch-time filtering mechanism for recipient-specific FU. Upon an unlearning request, the source client derives a lightweight adapter-style knowledge filter using an unlearning operator $\mathcal{U}$, and the server composes cached filters according to active source–target relations. Only target clients receive the filtered global model, while source and authorized clients retain the unfiltered one. Target clients further perform local JDOO to recover retained-task utility before aggregation. Built on generic $\mathcal{U}$ and aggregation operators $\mathcal{A}$, PULSAR flexibly integrates diverse methods and enables efficient targeted unlearning without transmitting the forgotten data.

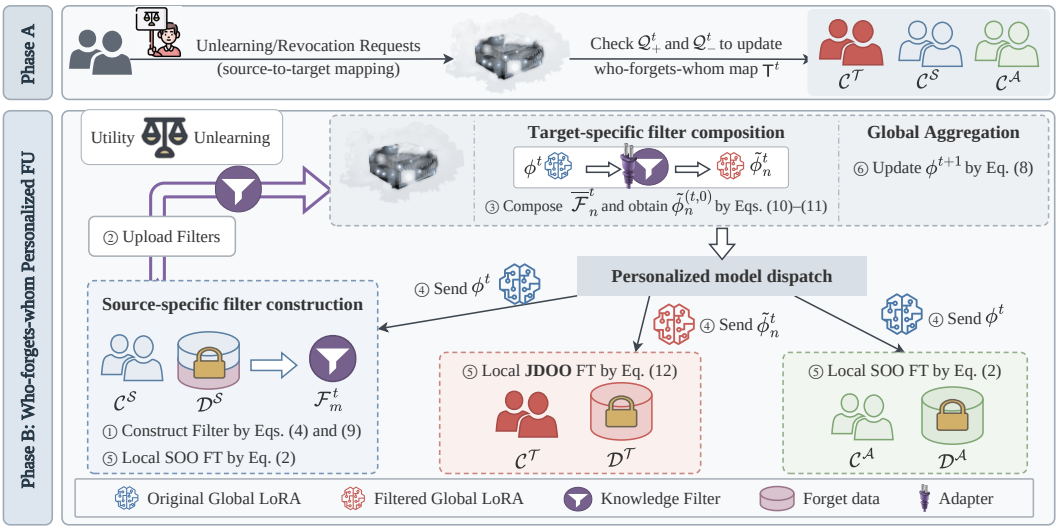



Figure 2: PULSAR overview. When clients issue unlearning/revocation requests, PULSAR is triggered. The server $\mathcal{S}$ processes requests in $\mathcal{Q}_+/\mathcal{Q}_-$, updates the *who-forgets-whom* map $\mathsf{T}$, and induces three client roles $\mathcal{C}^{\mathcal{S}}, \mathcal{C}^{\mathcal{T}}$ and $\mathcal{C}^{\mathcal{A}}$ (**Phase A**). For each activated source client $m \in \mathcal{C}^{\mathcal{S}}$, a source-specific filter $\mathcal{F}_m$ is constructed from its forget set $\widetilde{\mathcal{D}}_m$ and cached by $\mathcal{S}$. For each target client $n \in \mathcal{C}^{\mathcal{T}}$, $\mathcal{S}$ composes the corresponding source filters and dispatches the filtered initialization $\tilde{\phi}_n^t$. Meanwhile, clients in $\mathcal{C}^{\mathcal{S}}$ and $\mathcal{C}^{\mathcal{A}}$ receive the global parameter $\phi^t$ (**Phase B**).

### 3.1 Scope and Threat Model

PULSAR considers recipient-specific approximate unlearning under a trusted, policy-enforcing server. The server enforces a source-to-target authorization map and modifies dispatched model parameters using the displacement induced by $\mathcal{U}$. We assume an authenticated coordinator, protocol-compliant clients, and a valid authorization map. Our scope is model-level behavioral forgetting rather than adversarial security. Malicious servers or clients, collusion, unauthorized access to unfiltered checkpoints, filter theft, and authorization-map tampering are outside the scope.

### 3.2 PULSAR Workflow

Figure 2 and Algorithm 1 show the workflow of PULSAR. The convergence analysis is provided in Appendix D. We instantiate PULSAR with LoRA-based federated FT, as detailed below.

**Phase A: Requests Update.** At communication round $t$, clients may submit new targeted unlearning requests $\mathcal{Q}_+^t$ or revocation requests $\mathcal{Q}_-^t$ to the server $\mathcal{S}$. A new request from source client $m$ is written as $q_{m,+}^t = (m, \mathrm{id}(\widetilde{\mathcal{D}}_m), \mathcal{T}_{m,+}^t) \in \mathcal{Q}_+^t$, where $\mathrm{id}(\widetilde{\mathcal{D}}_m)$ denotes a local identifier for the forget

set (the raw set remains at source client $m$), and $\mathcal{T}^t_{m,+} \subseteq \mathcal{C} \setminus \{m\}$ denotes the target clients required to forget $m$'s contribution. A revocation request is written as $q^t_{m,-} = (m, \mathcal{T}^t_{m,-}) \in \mathcal{Q}^t_-$, where $\mathcal{T}^t_{m,-}$ specifies the targets for which $m$ cancels its active forgetting relations. $\mathcal{S}$ checks $\mathcal{Q}^t_+$ and $\mathcal{Q}^t_-$ to update a persistent source-to-target map $\mathsf{T}^t$ over the full client set $\mathcal{C}$, where $\mathsf{T}^t[m]$ records the currently active targets of $m$. For each involved $m$, the update is $\mathsf{T}^t[m] = \left(\mathsf{T}^{t-1}[m] \cup \mathcal{T}^t_{m,+}\right) \setminus \mathcal{T}^t_{m,-}$. Otherwise, $\mathsf{T}^t[m] = \mathsf{T}^{t-1}[m]$. Thus, each unlearning request persists across rounds until revoked by its source client. The updated mapping $\mathsf{T}^t$ induces three client groups over $\mathcal{C}$: source clients $\mathcal{C}^{\mathcal{S},t}$ with active forgetting requests, target clients $\mathcal{C}^{\mathcal{T},t}$ required to forget at least one source contribution, and authorized clients $\mathcal{C}^{\mathcal{A},t}$ that remain allowed to retain these contributions. This grouping is independent of the sampled client set $\mathcal{C}^t$: a source or target client may be inactive in round $t$, but its role persists and takes effect once it is sampled.

$\mathcal{S}$ then samples participating clients $\mathcal{C}^t \subseteq \mathcal{C}$ with ratio $\lambda$. If no sampled client belongs to $\mathcal{C}^{\mathcal{T},t}$, the dispatch step degenerates to standard federated FT: all sampled clients receive $\boldsymbol{\phi}^t$, perform SOO FT using (2), and $\mathcal{S}$ produces $\boldsymbol{\phi}^{t+1}$ by (3). Otherwise, the system enters **Phase B**.

**Phase B: Who-forgets-whom Personalized FU.** This phase proceeds in four steps. *(i) Source-specific filter update.* For each source client $m$ that appears in $\mathcal{Q}^t_+ \cup \mathcal{Q}^t_-$ and still has active targets after the relation update, PULSAR constructs or refreshes its source-specific filter $\mathcal{F}^t_m$ using (4 )and (9). The filter is uploaded to $\mathcal{S}$ and stored in the cache $\mathsf{F}$ for later reuse. If $\mathsf{T}^t[m] = \emptyset$, $\mathcal{S}$ removes $m$ from both $\mathsf{T}^t$ and $\mathsf{F}$. Source clients that do not issue or revoke requests in round $t$ keep their previous active relations and cached filters. *(ii) Target-specific filter composition.* For each sampled client $k \in \mathcal{C}^t$, $\mathcal{S}$ checks whether $k \in \mathcal{C}^{\mathcal{T},t}$. If so, it identifies the active source set $\mathcal{S}^t_k = \{m \in \mathcal{C}^{\mathcal{S},t} \mid k \in \mathsf{T}^t[m]\}$, and composes a target-specific filter $\overline{\mathcal{F}}^t_k$ from the cached filters $\{\mathsf{F}[m]\}_{m \in \mathcal{S}^t_k}$ using (10). *(iii) Personalized model dispatch.* Using (11), the server dispatches the filtered initialization $\widetilde{\boldsymbol{\phi}}^{(t,0)}_n$ to each sampled target client $n \in \mathcal{C}^t \cap \mathcal{C}^{\mathcal{T},t}$, and sends the unfiltered global parameters $\boldsymbol{\phi}^t$ to sampled non-target clients $n \in \mathcal{C}^t \setminus \mathcal{C}^{\mathcal{T},t}$. *(iv) Local FT and global aggregation.* Each sampled target client $n \in \mathcal{C}^{\mathcal{T},t}$ performs local JDOO FT using (12) to balance forgetting effectiveness and utility preservation, while sampled non-target clients perform SOO FT using (2). For each $k \in \mathcal{C}^t$, we define its dispatched initialization as (6). After local optimization, client $k$ returns the relative update as (7).

$$\boldsymbol{\phi}^{(t,0)}_k = \begin{cases} \widetilde{\boldsymbol{\phi}}^{(t,0)}_k, & k \in \mathcal{C}^t \cap \mathcal{C}^{\mathcal{T},t}, \\ \boldsymbol{\phi}^t, & k \in \mathcal{C}^t \setminus \mathcal{C}^{\mathcal{T},t}. \end{cases} \tag{6}$$

$$\Delta\boldsymbol{\phi}^t_k = \boldsymbol{\phi}^{(t,R)}_k - \boldsymbol{\phi}^{(t,0)}_k. \tag{7}$$

Finally, $\mathcal{S}$ aggregates these returned updates with an update-space aggregation operator $\mathcal{A}_\Delta$ and updates the global adapter as

$$\boldsymbol{\phi}^{t+1} = \boldsymbol{\phi}^t + \mathcal{A}_\Delta\left(\left\{\Delta\boldsymbol{\phi}^t_k\right\}_{k \in \mathcal{C}^t}\right). \tag{8}$$

**Recipient-specific Retrain reference.** For target client $n$ at round $t$, let $\mathcal{S}^t_n = \{m \in \mathcal{C}^{\mathcal{S},t} \mid n \in \mathsf{T}^t[m]\}$ and $\widetilde{\mathcal{D}}^t_n = \bigcup_{m \in \mathcal{S}^t_n} \widetilde{\mathcal{D}}_m$. Its recipient-specific *Retrain* reference follows (5), replacing $\widetilde{\mathcal{D}}_m$ with $\widetilde{\mathcal{D}}^t_n$ under the same federated training configuration.

### 3.3 Adaptive Filter Design

The filter $\mathcal{F}$ aims to suppress the representations associated with the forget set $\widetilde{\mathcal{D}}$ from the dispatched model. Once a source client $m$ initiates an unlearning request, it applies an unlearning operator $\mathcal{U}$ to its local forget set $\widetilde{\mathcal{D}}_m$ and obtains an unlearned adapter $\widetilde{\boldsymbol{\phi}}^{t-1}_m$ from the previous global adapter $\boldsymbol{\phi}^{t-1}$. The adapter-style filter captures the unlearning-induced discrepancy between $\widetilde{\boldsymbol{\phi}}^{t-1}_m$ and $\boldsymbol{\phi}^{t-1}$, serving as a parameter-space proxy for the effect induced by unlearning $\widetilde{\mathcal{D}}_m$. Formally, the filter $\mathcal{F}^t_m$ of source client $m$ at round $t$ is defined as (9), where $\widetilde{\boldsymbol{\phi}}^{t-1}_m$ is computed using (4).

$$\mathcal{F}^t_m = \boldsymbol{\phi}^{t-1} - \widetilde{\boldsymbol{\phi}}^{t-1}_m \tag{9}$$

The filter $\mathcal{F}_m$ is maintained by $\mathcal{S}$ as a deployable adapter and applied end-to-end to designated target clients, making PULSAR compatible with simultaneous multi-source and multi-target unlearning requests. For each active target client $n$, let $\mathcal{S}^t_n$ denote the set of source clients whose contributions should be removed from $n$ at round $t$. We define a target-specific combined filter as

$$\bar{\mathcal{F}}_n^t = \sum_{m \in \mathcal{S}_n^t} \omega_{n,m}^t \mathcal{F}_m^t \tag{10}$$

$$\widetilde{\boldsymbol{\phi}}_n^{(t,0)} = \boldsymbol{\phi}^t - \bar{\mathcal{F}}_n^t \tag{11}$$

where $\omega_{n,m}^t \geq 0$ and $\sum_{m \in \mathcal{S}_n^t} \omega_{n,m}^t = 1$. By default, the weights are sample-based and normalized over $\mathcal{S}_n^t$. At round $t$, the server dispatches the target-specific initialization $\widetilde{\boldsymbol{\phi}}_n^{(t,0)}$ to client $n$, thereby suppressing from $\boldsymbol{\phi}^t$ the components associated with the requested forget data $\widetilde{\mathcal{D}}_m$. To account for continued federated aggregation, the server dynamically updates and repeatedly applies the filters across rounds, mitigating the reintroduction of forgotten knowledge from non-target clients.

**Bounded composition.** Normalized non-negative weights constrain the magnitude of the composed parameter perturbation, but do not imply semantic orthogonality, equivalence to applying $\mathcal{U}$ to the union of forget sets, or invariance to LoRA gauge transformations. We therefore treat filter composition as a bounded parameter-space approximation.

**Filter overhead.** The server maintains one adapter-sized filter per active source client and composes target-specific filters on demand. Storage therefore grows linearly with the number of active sources rather than source–target pairs. Across the evaluated 1–8B backbones, each filter accounts for only $1.0\%$–$2.3\%$ of the base-model parameters (Appendix F.2). When a source filter is refreshed, it is constructed from the previous global adapter and applied to the current adapter, introducing a one-round temporal offset for that refresh. A cached filter may persist until the next refresh. Appendix F.1 examines the effect of temporal lag.

### 3.4 Local Fine-tuning with JDOO

The filtered initialization $\widetilde{\boldsymbol{\phi}}_n^{(t,0)}$ applies to target client $n$ the source-derived unlearning displacement specified by $\overline{\mathcal{F}}_n^t$, potentially perturbing representations important for $n$'s retained task distribution. Since the forget set $\widetilde{\mathcal{D}}_m$ is only available to source client $m$, target-side utility recovery relies solely on $n$'s local data $\mathcal{D}_n^{\mathcal{T}}$. To this end, we introduce a local JDOO module. Let $\widetilde{\boldsymbol{\phi}}_n^t \leftarrow \widetilde{\boldsymbol{\phi}}_n^{(t,0)}$ be the filtered target model and $\boldsymbol{\phi}_n^{\text{prev}}$ denote the locally trained model from the most recent participation of $n$ (equal to $\boldsymbol{\phi}_n^{(t-1,R)}$ under full participation), which serves as a retained-utility anchor. JDOO is formulated in (12), where $\mathcal{L}_n^{\text{FT}}$ is the local FT loss, $\mu \geq 0$ controls the anchoring strength, $\text{sg}(\cdot)$ denotes stop-gradient, and $|\boldsymbol{\phi}|$ is the number of trainable LoRA parameters. The factor $1/|\boldsymbol{\phi}|$ normalizes the proximal penalty into an average parameter-wise anchor, making a fixed $\mu$ comparable across adapter sizes.

$$\min_{\widetilde{\boldsymbol{\phi}}_n^t} \quad \mathcal{L}_n^{\text{FT}}\left(\widetilde{\boldsymbol{\phi}}_n^t; \mathcal{D}_n^{\mathcal{T}}\right) + \frac{\mu}{2|\boldsymbol{\phi}|}\left\|\widetilde{\boldsymbol{\phi}}_n^t - \text{sg}\left(\boldsymbol{\phi}_n^{\text{prev}}\right)\right\|_2^2 \tag{12}$$

The first term adapts the filtered model to the target client's local task distribution, while the second term explicitly constrains the update to remain close to the retained-utility anchor. Therefore, JDOO mitigates the utility degradation introduced by dispatch-time filtering while empirically preserving the intended source-to-target forgetting effect. We provide a theoretical analysis of JDOO in Appendix C.

## 4 Experiments

To evaluate whether PULSAR addresses the three RQs raised in 1, we organize our experiments to provide point-to-point evidence. First, we assess whether PULSAR achieves recipient-specific targeted unlearning (**RQ1**) in 4.2 and 4.3. Second, we examine whether PULSAR supports dynamic, recoverable unlearning under non-stationary requests (**RQ2**) through 4.3. Third, we evaluate whether PULSAR preserves model utility for $\mathcal{C}^{\mathcal{T}}$ under partial forgetting (**RQ3**) via 4.2. Additionally, we conduct ablation studies in 4.4 to isolate the effect of the JDOO, and sensitivity analyses in 4.5 to evaluate robustness to hyperparameter choices. More experimental results are shown in Appendix F.

### 4.1 Experimental Setup

We evaluate PULSAR on the **TOFU** benchmark (Dorna et al., 2025) with the stratified **Split90/95/99** protocols across six LLM backbones, including Llama-3.2-1B/3B, Llama-3.1-8B (Dubey et al., 2024), and Qwen2.5-1.5B/3B/7B (Hui et al., 2024). We compare against ten federated aggregation and eight underlying unlearning operators, with *Finetune* (i.e., without unlearning) and *Retrain* (i.e., retrained on the *Retain* set) reported as references. We defer method descriptions and the

discussion of PFU methods to Appendix A.4. We adopt a controlled full-participation setup with one client per role to isolate the mechanism, and report larger-scale multi-source-target results in Appendix F.10. When Mean±Std is reported across aggregation operators, the standard deviation reflects cross-operator dispersion rather than repeated-seed uncertainty. Appendix A.5 provides hyperparameter details. Following the official **TOFU** evaluation protocol, we report FTR, MU, and ROUGE-based scores (FOR/RER/RAR/WFR) to measure forgetting and utility preservation, with definitions given in Appendix A.3. A higher FTR/MU indicates stronger forgetting effectiveness and better utility preservation. Lower FOR indicates more effective forgetting, whereas higher RER/RAR/WFR indicates better retained and general knowledge utility.

### 4.2 Unlearning Effectiveness and Model Utility

Table 1 reports the single-round unlearning results of PULSAR on $\mathcal{C}^{\mathcal{T}}$ at $T{=}11$ following the 10-round federated warm-up using Llama-3.2-1B and Qwen2.5-1.5B. Detailed per-aggregation results are provided in Appendix F.4. We use a combined set of metrics for evaluation: MU and RER assess retained-knowledge utility, FTR and FOR measure target-knowledge forgetting, and RAR/WFR evaluate generalization on the external *RA/WF* subsets.

PULSAR consistently improves forgetting across both backbones. On Llama-3.2-1B, *GradAscent* increases FTR from $0.600$ under *Finetune* to $0.684$, approaching *Retrain* at $0.688$. On Qwen2.5-1.5B, it reaches $0.689$, slightly surpassing *Retrain* at $0.686$. These results show that target-side FTR approaches the *Retrain* reference with a single unlearning round. The modest MU reduction relative to *Finetune* and *Retrain* reflects the expected forgetting-utility trade-off, while the lower MU variation indicates less sensitivity to the evaluated aggregation choices $\mathcal{A}$ in this experiment. ROUGE results further confirm this trade-off: methods with lower FOR typically show a moderate decrease in RER. Among all unlearning operators, *CEU* achieves the lowest FOR on Qwen-1.5B ($0.430$), while *GradDiff* performs best on Llama-1B ($0.458$), both approaching *Retrain*. Meanwhile, RAR and WFR indicate that general knowledge is largely preserved, whereas target-answer retention is reduced across the evaluated forgetting metrics. Most operator-based methods outperform both *Finetune* and *Retrain* on external generalization. For example, on Llama-1B, *PDU* and *GradDiff* achieve the highest RAR of $0.798$, and *SimNPO* obtains the highest WFR of $0.854$. On Qwen-1.5B, *CEU* reaches the highest RAR of $0.676$, while *UNDIAL* achieves the highest WFR of $0.839$. These results indicate that external generalization is largely preserved under the evaluated unlearning configurations.

Table 1: Utility and unlearning performance on **Split90**. Results are averaged over 10 aggregation operators and reported as **Mean±Std**. We compare 8 unlearning operators using MU/FTR/FOR/RER/RAR/WFR, with *Finetune* and *Retrain* as baselines. All metrics are higher-is-better (↑) except for FOR (↓). **Bold**/underline indicates the best/second-best scores, respectively.

| **Llama-1B-Instruct with LoRA** ($T = 11$) | | | | | | |
|---|---|---|---|---|---|---|
| **Unlearning** | MU ↑ | FTR ↑ | FOR ↓ | RER ↑ | RAR ↑ | WFR ↑ |
| Finetune | $0.538 \pm 0.022$ | $0.600 \pm 0.036$ | $0.492 \pm 0.017$ | $0.486 \pm 0.024$ | $0.729 \pm 0.034$ | $0.835 \pm 0.021$ |
| GradAscent | $0.500 \pm 0.020$ | $\mathbf{0.684 \pm 0.014}$ | $0.465 \pm 0.010$ | $\underline{0.466 \pm 0.012}$ | $\underline{0.797 \pm 0.027}$ | $0.841 \pm 0.014$ |
| GradDiff | $\mathbf{0.507 \pm 0.013}$ | $0.680 \pm 0.015$ | $\mathbf{0.458 \pm 0.009}$ | $0.465 \pm 0.010$ | $\mathbf{0.798 \pm 0.022}$ | $0.837 \pm 0.013$ |
| NPO | $\underline{0.505 \pm 0.013}$ | $0.663 \pm 0.018$ | $0.474 \pm 0.012$ | $0.465 \pm 0.013$ | $0.790 \pm 0.022$ | $0.836 \pm 0.017$ |
| SimNPO | $0.504 \pm 0.013$ | $0.639 \pm 0.017$ | $0.487 \pm 0.009$ | $0.465 \pm 0.012$ | $0.787 \pm 0.025$ | $\mathbf{0.854 \pm 0.015}$ |
| CEU | $0.496 \pm 0.013$ | $\underline{0.681 \pm 0.016}$ | $\underline{0.461 \pm 0.013}$ | $0.460 \pm 0.012$ | $0.796 \pm 0.018$ | $0.845 \pm 0.016$ |
| PDU | $0.500 \pm 0.015$ | $0.671 \pm 0.017$ | $0.472 \pm 0.008$ | $0.463 \pm 0.008$ | $\mathbf{0.798 \pm 0.022}$ | $\underline{0.847 \pm 0.010}$ |
| UNDIAL | $\underline{0.505 \pm 0.014}$ | $0.657 \pm 0.019$ | $0.481 \pm 0.011$ | $\underline{0.466 \pm 0.012}$ | $0.788 \pm 0.015$ | $0.847 \pm 0.012$ |
| WGA | $0.504 \pm 0.013$ | $0.663 \pm 0.016$ | $0.476 \pm 0.011$ | $\mathbf{0.467 \pm 0.013}$ | $0.782 \pm 0.019$ | $0.843 \pm 0.010$ |
| Retrain | $0.518 \pm 0.036$ | $0.688 \pm 0.025$ | $0.441 \pm 0.013$ | $0.486 \pm 0.057$ | $0.715 \pm 0.107$ | $0.826 \pm 0.049$ |
| **Qwen-1.5B-Instruct with LoRA** ($T = 11$) | | | | | | |
| **Unlearning** | MU ↑ | FTR ↑ | FOR ↓ | RER ↑ | RAR ↑ | WFR ↑ |
| Finetune | $0.480 \pm 0.034$ | $0.619 \pm 0.075$ | $0.469 \pm 0.042$ | $0.463 \pm 0.030$ | $0.640 \pm 0.119$ | $0.805 \pm 0.054$ |
| GradAscent | $0.447 \pm 0.021$ | $\mathbf{0.689 \pm 0.046}$ | $0.444 \pm 0.017$ | $\mathbf{0.456 \pm 0.016}$ | $0.640 \pm 0.138$ | $0.826 \pm 0.054$ |
| GradDiff | $\underline{0.455 \pm 0.016}$ | $\underline{0.686 \pm 0.042}$ | $\underline{0.437 \pm 0.016}$ | $0.452 \pm 0.017$ | $\underline{0.663 \pm 0.136}$ | $0.826 \pm 0.052$ |
| NPO | $0.454 \pm 0.018$ | $0.674 \pm 0.046$ | $0.450 \pm 0.021$ | $0.451 \pm 0.016$ | $0.639 \pm 0.131$ | $0.829 \pm 0.045$ |
| SimNPO | $\mathbf{0.457 \pm 0.017}$ | $0.654 \pm 0.044$ | $0.462 \pm 0.023$ | $0.450 \pm 0.014$ | $0.649 \pm 0.110$ | $0.824 \pm 0.050$ |
| CEU | $0.453 \pm 0.017$ | $0.683 \pm 0.042$ | $\mathbf{0.430 \pm 0.015}$ | $0.454 \pm 0.015$ | $\mathbf{0.676 \pm 0.125}$ | $\underline{0.835 \pm 0.044}$ |
| PDU | $0.452 \pm 0.023$ | $0.684 \pm 0.042$ | $0.451 \pm 0.021$ | $\underline{0.455 \pm 0.019}$ | $0.661 \pm 0.123$ | $0.818 \pm 0.053$ |
| UNDIAL | $0.454 \pm 0.018$ | $0.668 \pm 0.045$ | $0.455 \pm 0.025$ | $0.450 \pm 0.019$ | $0.646 \pm 0.121$ | $\mathbf{0.839 \pm 0.042}$ |
| WGA | $0.454 \pm 0.016$ | $0.671 \pm 0.045$ | $0.453 \pm 0.023$ | $0.450 \pm 0.016$ | $0.644 \pm 0.113$ | $0.831 \pm 0.051$ |
| Retrain | $0.486 \pm 0.035$ | $0.686 \pm 0.030$ | $0.422 \pm 0.004$ | $0.472 \pm 0.034$ | $0.635 \pm 0.124$ | $0.791 \pm 0.051$ |

### 4.3 Source-to-Target Unlearning Analysis

Figure 3 shows the source-to-target unlearning dynamics of the three client groups ($\mathcal{C}^{\mathcal{S}}$, $\mathcal{C}^{\mathcal{T}}$, and $\mathcal{C}^{\mathcal{A}}$) under $\mathcal{A} = \textit{FedAvg}, \mathcal{U} = \textit{SimNPO}$, and **Split90**. We report $\Delta$MU = MU(PULSAR) − MU(*Finetune*) and $\Delta$FTR = FTR(PULSAR) − FTR(*Retrain*). The filter $\mathcal{F}$ is activated in rounds 1–5 for *unlearning* and deactivated in rounds 6–10 for *recovery*. Here, *recovery* refers to policy rollback after deactivating a source–target forgetting relation. Across all subplots, $\mathcal{C}^{\mathcal{T}}$ exhibits lower MU but higher FTR than $\mathcal{C}^{\mathcal{S}}$ and $\mathcal{C}^{\mathcal{A}}$ at round 1. This reflects initialization from the knowledge-filtered global model and JDOO optimization under a forgetting constraint, which immediately strengthens forgetting while introducing a utility cost. During rounds 1–5, the MU of $\mathcal{C}^{\mathcal{T}}$ gradually approaches its *Finetune* level, while its FTR remains higher than that of non-target clients but slowly decreases, indicating a better utility–forgetting balance. In contrast, $\mathcal{C}^{\mathcal{S}}$ and $\mathcal{C}^{\mathcal{A}}$, which do not directly receive $\mathcal{F}$, remain close with higher MU and lower FTR. Once $\mathcal{F}$ is deactivated in *recovery* (round 6), $\mathcal{C}^{\mathcal{T}}$ quickly restores MU, and its FTR drops sharply, converging to non-target clients within one round. These trends across the Llama/Qwen families and 1–8B scales show that $\mathcal{F}$ is targeted, controlled during unlearning, and rapidly recoverable after deactivation. More results are provided in Appendix F.6.

Together with Section 4.2, the single-round results confirm that PULSAR achieves target-specific behavioral forgetting metrics close to the *Retrain* reference while preserving *Finetune*-level non-target utility, addressing **RQ1** and **RQ3**. The rapid MU/FTR rollback after filter deactivation shows that the unlearning effect in PULSAR is plug-and-play rather than permanently baked into target models, enabling recoverable unlearning under non-stationary unlearning requests, as required by **RQ2**.

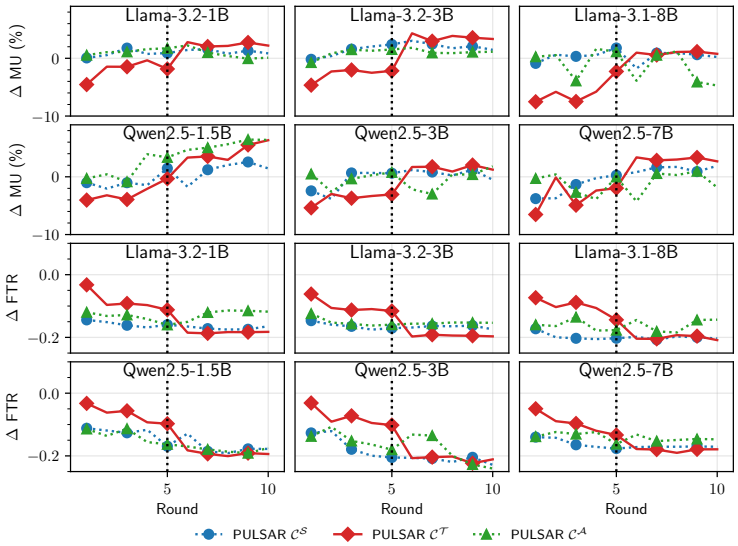



Figure 3: Source-to-Target Unlearning Analysis.

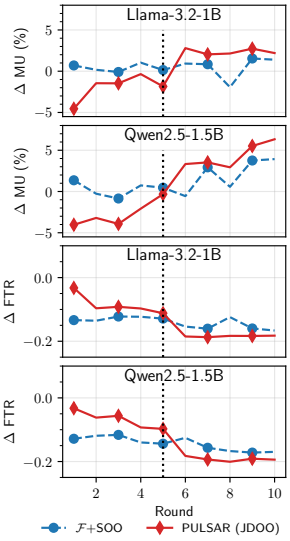



Figure 4: Ablation Study.

### 4.4 Ablation Study

Figure 4 reports ablation results on $\mathcal{C}^{\mathcal{T}}$ under $\mathcal{A} = \textit{FedAvg}$, $\mathcal{U} = \textit{SimNPO}$, and **Split90**. We examine the two core components of PULSAR: the filter $\mathcal{F}$ and JDOO, comparing $\mathcal{F}$+SOO with full PULSAR ($\mathcal{F}$+JDOO). We report $\Delta$MU = MU(PULSAR) − MU(*Finetune*) and $\Delta$FTR = FTR(PULSAR) − FTR(*Retrain*). During *unlearning*, PULSAR starts with lower MU but higher FTR, reflecting the utility cost of enforcing targeted forgetting on $\mathcal{C}^{\mathcal{T}}$. As training proceeds, it rapidly restores MU to the corresponding *Finetune* level, showing that JDOO mitigates the utility degradation introduced by $\mathcal{F}$. In contrast, $\mathcal{F}$+SOO maintains lower FTR, indicating weaker forgetting. During *recovery*, both methods recover MU within one round, but PULSAR achieves larger utility gains and a sharper FTR reduction, suggesting that JDOO further enables faster rollback and more stable recovery after removing the forgetting constraint. To further isolate the contributions of the source-derived filter and JDOO, we provide additional matched controls in Appendix F.7. The results show that PULSAR's source-derived filter achieves stronger target-side forgetting, while JDOO consistently improves forgetting performance across all six evaluated unlearning operators. More ablation results with varying $U$ and $A$ are provided in Appendix F.8.

### 4.5 Hyperparameter Analysis: Filter Training Epoch

Figure 5 reports MU and FTR for the three client groups ($\mathcal{C}^{\mathcal{S}}$, $\mathcal{C}^{\mathcal{A}}$, and $\mathcal{C}^{\mathcal{T}}$) under different filter-training rounds on Qwen2.5-1.5B and Llama-3.2-1B, with $\mathcal{A} = \textit{FedAvg}$, $\mathcal{U} = \textit{SimNPO}$, and **Split90**. *Finetune* and *Retrain* are included as references. Across both backbones, $\mathcal{C}^{\mathcal{T}}$ consistently achieves higher FTR and remains closest to *Retrain*, following the ranking $\mathcal{C}^{\mathcal{T}} > \mathcal{C}^{\mathcal{A}} \approx \mathcal{C}^{\mathcal{S}}$. This shows that the filter $\mathcal{F}$ and JDOO jointly strengthen forgetting for target clients $\mathcal{C}^{\mathcal{T}}$, while $\mathcal{C}^{\mathcal{S}}$ and $\mathcal{C}^{\mathcal{A}}$ remain closer to *Finetune* without forgetting constraints. Meanwhile, $\mathcal{C}^{\mathcal{T}}$ maintains MU comparable to the non-target groups $\mathcal{C}^{\mathcal{S}}$ and $\mathcal{C}^{\mathcal{A}}$, indicating a balanced trade-off between utility preservation and forgetting effectiveness. Varying the filter-training rounds from 1 to 5 causes only minor changes in MU and FTR across all groups, demonstrating that PULSAR is robust to this hyperparameter. More results are provided in Appendix F.9.

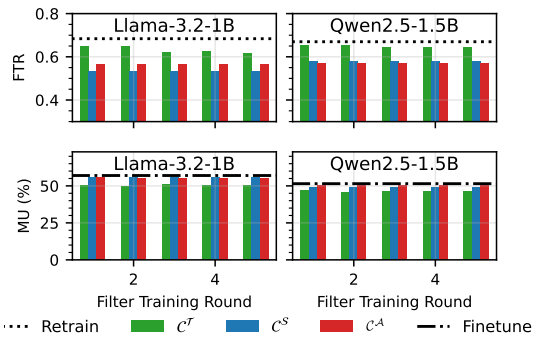



Figure 5: Hyperparameter Analysis.

## 5 Related Work

The distributed, multi-round training nature of FL presents unique challenges for unlearning. Existing FU frameworks include FedAF (Li et al., 2025), which employs active unlearning via a teacher-student component to overwrite old memories with new ones, while FUSED (Zhong et al., 2025b) introduces sparse adapters to overwrite forgettable knowledge with retained data for low-cost unlearning. FULM (Zhong et al., 2025a) decouples unlearning and retention tasks and uses hierarchical clustering to remove obsolete knowledge from the LLMs. The recent Oblivionis framework (Zhang et al., 2025) is the first to combine federated LLMs with targeted unlearning, thereby balancing data removal and model utility. Building on these foundations, Personalized FU (PFU) introduces client-level customization into FU by adapting the unlearning scope, strength, or local objectives under heterogeneous data. ZeroFU and Mimir (Wu et al., 2025b;a) model client-specific representations with synthetic data and knowledge distillation, aiming to better identify and remove a forgotten client's contribution from the global model. ConPFU (Pan et al., 2025) allows users to adjust the scope and intensity of unlearning according to privacy preferences, focusing on how much to forget.

By contrast, PULSAR personalizes unlearning along a recipient-specific axis defined by source–target authorization relations. It targets authorization-contraction scenarios in which designated target recipients should forget source-specified data, while still-authorized recipients retain the corresponding unfiltered behavior. This avoids the unnecessary utility loss caused by federation-wide "forgetting-for-all" and captures selective forget/retain requirements arising from consent tightening, licensing revocation, or partial withdrawal.

## 6 Conclusions

We propose PULSAR, a *who-forgets-whom* personalized unlearning framework for recipient-specific behavioral forgetting in federated LLMs. PULSAR maintains persistent source–target relations, converts the displacement induced by an underlying unlearning operator into reusable source-specific filters, and dynamically composes and dispatches recipient-specific model initializations, while initialization-relative aggregation prevents these filter offsets from being permanently written into the shared model. JDOO further mitigates the resulting target-side utility loss. Across the evaluated TOFU settings, PULSAR strengthens behavioral forgetting on designated target clients, with FTR/FOR often approaching the corresponding *Retrain* reference, while preserving non-target utility close to the *Finetune* reference. Additional experiments on component isolation, MUSE-News, and severe Non-IID partitions further support the recipient-specific nature of this behavior. PULSAR therefore provides a flexible framework for policy-driven approximate unlearning, while certified removal of training-data influence and adversarially secure revocation remain beyond the current scope.

## AI USE STATEMENT

Generative AI tools were used to assist with language editing, manuscript organization, and presentation, including improving wording, clarity, and formatting. The authors reviewed and verified all AI-assisted content. The research questions, methodological design, theoretical claims, experimental protocols, data analysis, and conclusions presented in this paper remain the responsibility of the authors. Generative AI outputs were not treated as scientific evidence or used as a substitute for experimental validation. The authors take full responsibility for the accuracy and integrity of the final manuscript.

## ETHICS STATEMENT

This work studies recipient-specific behavioral unlearning in federated large language models. Our experiments use established unlearning benchmarks and do not involve newly collected human-subject data or user studies. In particular, the primary TOFU benchmark consists of synthetic fictitious-author profiles and associated question–answer pairs. PULSAR is designed to support selective revocation of learned information under changing authorization or collaboration policies, which may be relevant to privacy- and consent-sensitive federated deployments. However, the method should not be interpreted as providing certified deletion, cryptographic revocation, or a formal privacy guarantee. Our threat model assumes a trusted, authenticated, and policy-enforcing coordinator together with protocol-compliant clients; malicious servers or clients, collusion, unauthorized access to unfiltered checkpoints, filter theft, and authorization-map tampering are outside the present scope. The reported forgetting metrics therefore measure behavioral forgetting under the evaluated protocols rather than proof that all influence of the requested training data has been removed. Deployment in privacy- or legally sensitive settings would require application-specific validation and appropriate technical and organizational safeguards beyond those evaluated here.

## REPRODUCIBILITY STATEMENT

We provide the information required to reproduce PULSAR and the reported experiments throughout the main paper and appendices. Section 3 specifies the PULSAR workflow, source-specific filter construction, recipient-specific filter composition, initialization-relative aggregation, and the JDOO objective, while Algorithm 1 in Appendix B gives the complete training and dispatch procedure. Appendix A describes the evaluated datasets and model backbones, formal definitions of metrics, baseline unlearning and federated aggregation methods, and default hyperparameters. Theoretical assumptions and proofs for JDOO, PULSAR convergence, and conditional inheritance of the underlying unlearning operator are provided in Appendices C–E. Appendix F reports additional experiments covering temporal filter lag, filter size, forget/retain splits, model and aggregation configurations, source-to-target dynamics, ablations, hyperparameter sensitivity, and larger-scale settings. The main experiments use the publicly available TOFU benchmark, six LLM backbones, ten federated aggregation methods, and eight unlearning operators. Default training configurations, including LoRA settings, optimization hyperparameters, communication-round settings, and computing resources, are documented in Appendix A.5.

## REFERENCES

Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. Zero-shot machine unlearning. *IEEE Transactions on Information Forensics and Security*, 18:2345–2354, 2023.

Yijiang River Dong, Hongzhou Lin, Mikhail Belkin, Ramon Huerta, and Ivan Vulić. Undial: Self-distillation with adjusted logits for robust unlearning in large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8827–8840, 2025.

Vineeth Dorna, Anmol Mekala, Wenlong Zhao, Andrew McCallum, Zachary C Lipton, J Zico Kolter, and Pratyush Maini. Openunlearning: Accelerating llm unlearning via unified benchmarking of methods and metrics. *arXiv preprint arXiv:2506.12618*, 2025.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Taha Entesari, Arman Hatami, Rinat Khaziev, Anil Ramakrishna, and Mahyar Fazlyab. Constrained entropic unlearning: A primal-dual framework for large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Chongyu Fan, Jiancheng Liu, Licong Lin, Jinghan Jia, Ruiqi Zhang, Song Mei, and Sijia Liu. Simplicity prevails: Rethinking negative preference optimization for llm unlearning. In *Neurips Safe Generative AI Workshop 2024*, 2024.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Allen H Huang, Hui Wang, and Yi Yang. Finbert: A large language model for extracting information from financial text. *Contemporary Accounting Research*, 40(2):806–841, 2023.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Kristin M Kostick-Quenet, Marcelo Corrales Compagnucci, Mateo Aboy, and Timo Minssen. Patient-centric federated learning: automating meaningful consent to health data sharing with smart contracts. *Journal of Law and the Biosciences*, 12(1):lsaf003, 2025.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Yuyuan Li, Jiaming Zhang, Yixiu Liu, and Chaochao Chen. Class-wise federated unlearning: Harnessing active forgetting with teacher–student memory generation. *Knowledge-Based Systems*, 316:113353, 2025.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.

Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Yuguang Yao, Chris Yuhao Liu, Xiaojun Xu, Hang Li, et al. Rethinking machine unlearning for large language models. *Nature Machine Intelligence*, pp. 1–14, 2025.

Ziyao Liu, Yu Jiang, Jiyuan Shen, Minyi Peng, Kwok-Yan Lam, Xingliang Yuan, and Xiaoning Liu. A survey on federated unlearning: Challenges, methods, and future directions. *ACM Comput. Surv.*, 57(1), October 2024. ISSN 0360-0300.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary Chase Lipton, and J Zico Kolter. TOFU: A task of fictitious unlearning for LLMs. In *First Conference on Language Modeling*, 2024.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 20–22 Apr 2017.

Yanghe Pan, Yuntao Wang, Zhou Su, Yuan Gao, Shaolong Guo, Qinnan Hu, Ruidong Li, and WeiWei Li. The right to be forgotten vs. the need to be remembered: Efficient personalized federated unlearning with optimal incentives. *IEEE Transactions on Network Science and Engineering*, 2025.

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H. Brendan McMahan. Adaptive federated optimization. In *The 9th International Conference on Learning Representations*, 2021.

Changlong Shi, Jinmeng Li, He Zhao, Dan dan Guo, and Yi Chang. Fedlws: Federated learning with adaptive layer-wise weight shrinking. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Changlong Shi, He Zhao, Bingjie Zhang, Mingyuan Zhou, Dandan Guo, and Yi Chang. Fedawa: adaptive optimization of aggregation weights in federated learning using client vectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 30651–30660, 2025b.

Shanli Tan, Hao Cheng, Xiaohu Wu, Han Yu, Tiantian He, Yew Soon Ong, Chongjun Wang, and Xiaofeng Tao. Fedcompetitors: Harmonious collaboration in federated learning with competing participants. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 15231–15239, 2024.

Yitong Tang. Adapted weighted aggregation in federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 23763–23765, 2024.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature medicine*, 29(8): 1930–1940, 2023.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.

Qizhou Wang, Jin Peng Zhou, Zhanke Zhou, Saebyeol Shin, Bo Han, and Kilian Q Weinberger. Rethinking llm unlearning objectives: A gradient perspective and go beyond. In *The Thirteenth International Conference on Learning Representations*, 2025.

Wenhan Wu, Huanghuang Liang, Tianyu Tu, Jiawei Jiang, Chuang Hu, and Dazhao Cheng. Mimir: Data-free federated unlearning through client-specific prompt generation for personalized models. *IEEE Transactions on Mobile Computing*, 2025a.

Wenhan Wu, Huanghuang Liang, Jingling Yuan, Jiawei Jiang, Kanye Ye Wang, Chuang Hu, Xiaobo Zhou, and Dazhao Cheng. Zero-shot federated unlearning via transforming from data-dependent to personalized model-centric. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 6588–6596, 2025b.

Jie Xu, Benjamin S Glicksberg, Chang Su, Peter Walker, Jiang Bian, and Fei Wang. Federated learning for healthcare informatics. *Journal of healthcare informatics research*, 5(1):1–19, 2021.

Bo Yang. Ce-u: Cross entropy unlearning. *arXiv preprint arXiv:2503.01224*, 2025.

Fuyao Zhang, Xinyu Yan, Tiantong Wu, Wenjie Li, Tianxiang Chen, Yang Cao, Ran Yan, Longtao Huang, Wei Yang Bryan Lim, and Qiang Yang. Oblivionis: A lightweight learning and unlearning framework for federated large language models. *arXiv preprint arXiv:2508.08875*, 2025.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. In *First Conference on Language Modeling*, 2024.

Yisheng Zhong, Zhengbang Yang, and Zhuangdi Zhu. Hierarchical federated unlearning for large language models. *arXiv preprint arXiv:2510.17895*, 2025a.

Zhengyi Zhong, Weidong Bao, Ji Wang, Shuai Zhang, Jingxuan Zhou, Lingjuan Lyu, and Wei Yang Bryan Lim. Unlearning through knowledge overwriting: Reversible federated unlearning via selective sparse adapter. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 30661–30670, 2025b.

# A Experimental Setup

## A.1 LoRA Fine-tuning Details

For the LoRA parameterization in (1), $\mathbf{A}_k^t \in \mathbb{R}^{a\times r}$ and $\mathbf{B}_k^t \in \mathbb{R}^{r\times b}$, where $r \ll \min(a, b)$. The adapter therefore contains substantially fewer trainable parameters than the frozen base model.

At communication round $t$, the server broadcasts the global adapter $\boldsymbol{\phi}^{t-1}$, which initializes each participating client as $\boldsymbol{\phi}_k^{(t,0)} = \boldsymbol{\phi}^{t-1}$. Client $k$ then performs $R$ local optimization steps. At the local step $s \in \{1, \dots, R\}$,

$$\boldsymbol{\phi}_k^{(t,s)} = \boldsymbol{\phi}_k^{(t,s-1)} - \eta \nabla_{\boldsymbol{\phi}} \mathcal{L}_k \left( \boldsymbol{\phi}_k^{(t,s-1)}; \mathcal{B}_k \right), \tag{13}$$

where $\eta$ is the learning rate and $\mathcal{B}_k \subseteq \mathcal{D}_k$ is a mini-batch.

For autoregressive language-model fine-tuning, the mini-batch objective is

$$\mathcal{L}_k \left( \boldsymbol{\phi}_k; \mathcal{B}_k \right) = -\frac{1}{|\mathcal{B}_k|} \sum_{(x_i, y_i) \in \mathcal{B}_k} \sum_{j=1}^{|y_i|} \log p(y_{i,j} \mid x_i \oplus y_{i,<j}; \boldsymbol{\theta}_{\text{base}} + \boldsymbol{\phi}_k), \tag{14}$$

where $y_{i,<j}$ denotes the preceding output tokens and $\oplus$ denotes concatenation. After $R$ local steps, the resulting adapters $\{\boldsymbol{\phi}_k^{(t,R)}\}$ are returned to the server and combined using the aggregation operator in (3).

## A.2 Models and Datasets

We evaluate six base models spanning different architectures and scales: (i) the Llama family, including Llama-3.1-8B-Instruct, Llama-3.2-1B-Instruct, and Llama-3.2-3B-Instruct (Dubey et al., 2024); and (ii) the Qwen family, including Qwen2.5-1.5B, Qwen-3B, and Qwen-7B (Hui et al., 2024). We fine-tune and evaluate all models on the **TOFU** benchmark (Dorna et al., 2025; Maini et al., 2024), which comprises four subsets: Forget Set (*Forget*), Retain Set (*Retain*), Real Authors (*RA*), and World Facts (*WF*). We adopt stratified splits **Split90**, **Split95**, and **Split99**, where 10%, 5%, and 1% of instances are assigned to the *Forget* set, respectively, and the remaining instances form the *Retain* set.

**TOFU** (Task of Fictitious Unlearning), proposed by Maini et al. (Maini et al., 2024), is a fine-grained benchmark for evaluating knowledge unlearning in LLMs. It comprises 200 diverse fictitious author profiles—generated via GPT-4 as synthetic autobiographies with structured attributes—each accompanied by 20 question–answer pairs targeting entity-centric facts such as identity traits, personal history, and biographical details. The benchmark is structured into four distinct sets: (1) *Forget Set*, targeted for unlearning and comprising 1%, 5%, or 10% of the data (corresponding to 2, 10, or 20 authors); (2) *Retain Set*, which should be preserved and comprises the remaining 99%, 95%, or 90%; (3) *Real Authors*, used to assess retention of knowledge about real-world entities; and (4) *World Facts*, used to evaluate general factual knowledge retention. TOFU further provides a chat-oriented LLM finetuned on the TOFU data as the starting point for unlearning, offering a controlled environment to precisely evaluate whether a model can remove accessible traces of the *Forget* set while maintaining overall utility and minimizing collateral degradation on unrelated tasks.

## A.3 Measurements

We evaluate unlearning using **TOFU**'s official definitions of Forgetting Truth Ratio (FTR) and Model Utility (MU). MU measures post-unlearning model utility and is defined as the harmonic mean of nine metrics across *Retain*, *RA*, and *WF*, with probability, Recall-Oriented Understudy for Gisting Evaluation (ROUGE), and TR evaluated at each level.

**Truth Ratio (TR) and Forget Truth Ratio (FTR).** TR measures whether the model assigns more probability to the correct answer than to a structurally similar incorrect perturbation. For a prompt $x$, paraphrased correct answer $y^{\text{para}}$, and perturbed incorrect answer $y^{\text{pert}}$, we define

$$\text{TR}(x) = \frac{p(y^{\text{para}} \mid x)}{p(y^{\text{para}} \mid x) + p(y^{\text{pert}} \mid x)}. \tag{15}$$

By construction, $\mathrm{TR}(x) \in [0, 1]$. A larger TR means that the model prefers the correct answer over the perturbation, while a smaller TR means that this preference is weakened.

The desired direction of TR depends on the evaluation split. On non-forget data such as *Retain*, *RA*, and *WF*, a high TR is desirable because the model should still prefer correct answers after unlearning. In contrast, on the *Forget* set, a high TR indicates that the model continues to assign high probability to the ground-truth answer that should have been forgotten. Therefore, for privacy evaluation, lower TR on the *Forget* set corresponds to stronger forgetting. To make the forgetting score directionally consistent, where higher values always indicate stronger privacy-oriented forgetting performance under the TOFU protocol, we define FTR as

$$\mathrm{FTR}(x) = 1 - \mathrm{TR}(x). \tag{16}$$

A larger FTR means that the model assigns relatively less probability mass to the forgotten ground-truth answer compared with the confounding alternative, indicating a stronger forgetting effect.

**Model Utility (MU).** MU evaluates whether the model preserves useful behavior on non-forget data after unlearning. Following **TOFU**, MU is computed over three non-forget regimes: *Retain*, *RA*, and *WF*. For each regime, we evaluate three utility-oriented metrics: probability-based score, ROUGE, and TR, resulting in nine metrics in total.

All nine metrics are oriented such that higher values indicate better utility. A higher probability-based score means that the model assigns more likelihood to the correct reference answer. A higher ROUGE score means that the generated response has greater lexical overlap with the ground truth. A higher TR means that the model prefers the correct answer over a plausible incorrect perturbation. Since these metrics are evaluated on non-forget data, higher values indicate that the unlearning method better preserves the model's original capabilities. We aggregate the nine metrics using the harmonic mean, reported as follows:

$$U_{\text{model}} = \frac{9}{\sum_{m \in \mathcal{M}} \frac{1}{m}}. \tag{17}$$

The harmonic mean penalizes methods with poor performance on any individual component, because a small metric value leads to a large reciprocal term. Therefore, a high MU requires consistently strong performance across all utility dimensions rather than improvement on only a subset of them. A larger MU thus indicates better preservation of model utility after unlearning.

**ROUGE.** ROUGE is used to measure the similarity between model-generated answers and the ground truth. Specifically, we adopt ROUGE-L (Lin, 2004) recall, which computes overlap based on the longest common subsequence (LCS), making it relatively robust to minor phrasing variations while providing an approximate measure of answer correctness in question answering. Formally, given the reference answer $a$ (i.e., $y$) and the generated response $\hat{a}$ (i.e., $f(x)$), ROUGE-L is defined as:

$$\mathrm{ROUGE}_L(x) = \frac{\mathrm{LCS}(a, \hat{a})}{|a|}, \tag{18}$$

where $\mathrm{LCS}(a, \hat{a})$ denotes the length of the longest common subsequence between $a$ and $\hat{a}$, and $|a|$ is the length of the reference answer. In our evaluation, ROUGE serves as a complementary utility metric, capturing whether unlearning preserves the model's ability to generate responses that retain key reference content on non-forget data under various evaluation conditions, including paraphrased references and adversarial prompts. ROUGE is evaluated on non-forget data in the utility assessment, a higher ROUGE-L recall indicates that the model-generated answer covers more of the ground-truth content, and therefore suggests better preservation of question-answering ability after unlearning.

### A.4 BASELINES

We adopt ten well-established federated optimization algorithms and eight unlearning-operator-based algorithms as baselines. For FL aggregation, we consider adaptive server-side optimization methods (*FedAdagrad*, *FedAdam*, and *FedYogi* (Reddi et al., 2021)), classical weighted/proximal averaging methods (*FedAvg* (McMahan et al., 2017), *FedAvgM* (Hsu et al., 2019), and *FedProx* (Li et al., 2020)), normalized aggregation for heterogeneous local updates (*FedNova* (Wang et al., 2020)), adaptive reweighting methods (*FedAW* (Tang, 2024) and *FedAWA* (Shi et al., 2025b)), and layer-wise shrinkage

aggregation (*FedLWS* (Shi et al., 2025a)). For operator-based unlearning, we include gradient-ascent-based methods (*GradAscent*, *GradDiff* (Maini et al., 2024), and *WGA* (Wang et al., 2025)), preference-based methods (*NPO* (Zhang et al., 2024) and *SimNPO* (Fan et al., 2024)), logit/distribution-based methods (*CEU* (Yang, 2025) and *UNDIAL* (Dong et al., 2025)), and constrained optimization methods (*PDU* (Entesari et al., 2025)). We further report *Finetune*, the model without unlearning, and *Retrain*, the model retrained on the *Retain* set, as reference points.

We do not include PFU methods, such as ZeroFU, Mimir, and ConPFU, as direct empirical baselines because they do not support the targeted unlearning setting studied in this paper. In PULSAR, the same source contribution can be revoked for the target clients $\mathcal{C}^{\mathcal{T}}$ while being preserved for the authorized clients $\mathcal{C}^{\mathcal{A}}$, rather than being uniformly forgotten by all clients as in prior work. Directly comparing with PFU methods would therefore require a substantial redesign of their unlearning pipelines, instead of a simple drop-in replacement. Without such redesign, the comparison would not reflect the intended operating assumptions of these methods.

### A.4.1 Unlearning Operators $\mathcal{U}$

**Gradient Ascent (*GradAscent*).** Gradient Ascent (Maini et al., 2024) is a straightforward unlearning baseline that enforces forgetting by maximizing the training loss on the forget set, thereby directly reducing the model's likelihood of producing the targeted responses. Given forget samples $(x, y_f) \sim \mathcal{D}_{\text{forget}}$ and an unlearning model $f_{\text{unl}}$, *GradAscent* optimizes:

$$\mathcal{L}_{\textit{GradAscent}} = -\gamma\, \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}}\, \ell(y_f \mid x; f_{\text{unl}})\,. \tag{19}$$

Here, $\ell(y \mid x; f)$ denotes the standard negative log-likelihood (cross-entropy) loss, while the negative sign converts gradient descent into gradient ascent on the forget loss. The scalar $\gamma > 0$ controls forgetting strength: larger $\gamma$ enforces more aggressive suppression of $y_f$. Intuitively, *GradAscent* pushes the model parameters toward regions where forget answers become less probable, but since the loss is unbounded, this mechanism can also induce instability and substantial drift in model behavior.

**Gradient Difference (*GradDiff*).** Gradient Difference (Maini et al., 2024) extends *GradAscent* by explicitly incorporating utility preservation through *Retain*-set training. Rather than only suppressing forget data, *GradDiff* jointly performs gradient ascent on $\mathcal{D}_{\text{forget}}$ and gradient descent on $\mathcal{D}_{\text{retain}}$:

$$\mathcal{L}_{\textit{GradDiff}} = -\gamma\, \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}}\, \ell(y_f \mid x; f_{\text{unl}}) + \alpha\, \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}}\, \ell(y \mid x; f_{\text{unl}})\,. \tag{20}$$

The first term enforces forgetting by decreasing the likelihood of forget answers, while the second term maintains model utility by continuing standard maximum-likelihood training on the *Retain* set. The hyperparameters $\gamma$ and $\alpha$ balance the forget–retain trade-off: increasing $\gamma$ strengthens forgetting, whereas increasing $\alpha$ prioritizes retention. This formulation provides a simple mechanism to stabilize *GradAscent* by anchoring optimization to non-forget samples, though it may still suffer from optimization sensitivity when the ascent component dominates.

**Negative Preference Optimization (*NPO*).** Negative Preference Optimization (Zhang et al., 2024) formulates unlearning as a preference-style objective that penalizes the model for preferring the forget answer relative to a reference model $f_{\text{target}}$ (typically the pre-unlearning model). For forget samples $(x, y_f) \sim \mathcal{D}_{\text{forget}}$, NPO minimizes:

$$\mathcal{L}_{\textit{NPO}} = -\frac{2}{\beta}\, \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}}\, \log\sigma\left(-\beta \log\left(\frac{p(y_f \mid x; f_{\text{unl}})}{p(y_f \mid x; f_{\text{target}})}\right)\right) + \alpha\, \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}}\, \ell(y \mid x; f_{\text{unl}})\,. \tag{21}$$

The key mechanism lies in the log-likelihood ratio $\log \frac{p(y_f|x;f_{\text{unl}})}{p(y_f|x;f_{\text{target}})}$, which measures how much more (or less) the unlearned model assigns probability mass to the forget answer compared with the reference. The negative sign inside the sigmoid encourages $p(y_f \mid x; f_{\text{unl}}) \ll p(y_f \mid x; f_{\text{target}})$, i.e., the forget answer becomes substantially less likely after unlearning. The temperature $\beta$ controls the sharpness of the penalty, while the retain-set term (weighted by $\alpha$) preserves general utility. Compared with *GradAscent*-style ascent, the sigmoid-based loss provides a smoother and bounded optimization signal, often resulting in improved stability.

***SimNPO*.** *SimNPO* (Fan et al., 2024) is a simplified variant of *NPO* that removes the explicit dependence on a reference model by replacing the likelihood ratio term with a reference-free margin constraint. Specifically, *SimNPO* optimizes:

$$\mathcal{L}_{\textit{SimNPO}} = -\frac{2}{\beta}\,\mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}}\,\log\sigma\!\left(-\frac{\beta}{|y_f|}\log p(y_f \mid x; f_{\text{unl}}) - \delta\right) + \alpha\,\mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}}\,\ell(y \mid x; f_{\text{unl}})\,. \tag{22}$$

Instead of comparing against $f_{\text{target}}$, *SimNPO* directly suppresses the model likelihood of forget answers via $\log p(y_f \mid x; f_{\text{unl}})$, normalized by the answer length $|y_f|$ to avoid bias toward shorter responses. The margin parameter $\delta$ defines how strongly the forget likelihood must be reduced before the penalty saturates, enabling a more uniform allocation of forgetting effort across examples. By eliminating the reference model, *SimNPO* avoids potential reference-induced bias while retaining the core optimization behavior of *NPO*, yielding a compact yet effective objective for scalable LLM unlearning.

**Cross Entropy Unlearning (*CEU*).** *CEU* (Yang, 2025) formulates unlearning as a modified cross-entropy objective in logit space, avoiding the unstable gradient-ascent update commonly used to decrease the likelihood of forget samples. For a target token $y_t$ with model logits $z_t \in \mathbb{R}^{|\mathcal{V}|}$ and output distribution $p_t(i) = \mathrm{softmax}(z_t)_i$, *CEU* first constructs a suppressed target distribution by setting the ground-truth logit to $-\infty$:

$$\tilde{z}_{t,i} = \begin{cases} -\infty, & i = y_t, \\ z_{t,i}, & i \neq y_t, \end{cases} \qquad q_t(i) = \mathrm{sg}(\mathrm{softmax}(\tilde{z}_t)_i)\,, \tag{23}$$

where $\mathrm{sg}(\cdot)$ denotes the stop-gradient operator. The resulting unlearning loss is

$$\mathcal{L}_{\textit{CEU}} = -\mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}}\frac{1}{|y_f|}\sum_{t=1}^{|y_f|}\sum_{i\in\mathcal{V}} q_t(i)\log p_t(i). \tag{24}$$

Since $q_t(y_t) = 0$, the objective explicitly removes probability mass from the memorized target token while preserving the model-induced relative distribution over alternative tokens. Unlike gradient ascent, whose signal vanishes when the model is highly confident in the forget token, *CEU* yields a gradient on the target logit proportional to $p_t(y_t)$, thereby applying a strong corrective update when memorization is severe and a weak update once the target has already been forgotten.

**Primal Dual Unlearning (*PDU*).** *PDU* (Entesari et al., 2025) formulates LLM unlearning as a constrained optimization problem rather than a manually weighted trade-off between forgetting and retention. Specifically, it minimizes a forgetting objective on $\mathcal{D}_{\text{forget}}$ while explicitly constraining the degradation on the retain set:

$$\min_{\theta}\ \mathcal{L}_{\text{fgt}}(\pi_\theta, \mathcal{D}_{\text{forget}}) \quad \text{s.t.} \quad \mathcal{L}_{\text{rtn}}(\pi_\theta, \mathcal{D}_{\text{retain}}) \leq \epsilon, \tag{25}$$

where $\epsilon = (1+\alpha)\mathcal{L}_{\text{rtn}}(\pi_{\text{ref}}, \mathcal{D}_{\text{retain}})$ specifies an explicit retention budget relative to the reference model. The retention loss is the standard autoregressive cross-entropy,

$$\mathcal{L}_{\text{rtn}} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}}[-\log\pi_\theta(y \mid x)]. \tag{26}$$

For forgetting, *PDU* introduces a logit-margin flattening loss that drives the model output distribution toward uniformity on forget samples without computing entropy explicitly:

$$\mathcal{L}^{\text{LM}}_{\text{fgt}} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{forget}}}\left[\frac{1}{|y|}\sum_{t=1}^{|y|}\left(\max_k z_{t,k} - \frac{1}{V}\sum_{k=1}^{V} z_{t,k}\right)^2\right], \tag{27}$$

where $z_t = \pi_\theta^{\text{logits}}(y_t \mid x, y_{<t})$ and $V$ is the vocabulary size. Minimizing this loss suppresses logit peakedness: zero loss is achieved only when all token logits are equal, which corresponds to a uniform predictive distribution after softmax and thus maximal uncertainty over the forgotten response. The constrained problem is optimized through the Lagrangian

$$\mathcal{L}(\theta,\lambda) = \mathcal{L}^{\text{LM}}_{\text{fgt}} + \lambda\big(\mathcal{L}_{\text{rtn}} - \epsilon\big), \tag{28}$$

with primal-dual updates

$$\theta \leftarrow \theta - \eta_\theta \nabla_\theta \mathcal{L}(\theta, \lambda), \qquad \lambda \leftarrow \big[\lambda + \eta_\lambda(\mathcal{L}_{\mathrm{rtn}} - \epsilon)\big]_+. \tag{29}$$

The dual variable $\lambda$ adaptively increases when the retention constraint is violated, strengthening utility preservation, and decreases when the model remains within the retention budget, allowing more aggressive forgetting. In this way, ***PDU*** replaces opaque scalarization with an interpretable retention constraint, while its softmax-free logit-flattening loss provides stable, non-saturating forgetting signals and avoids the instability of gradient-ascent-based unlearning.

***UNDIAL.*** *UNDIAL* (Dong et al., 2025) performs unlearning through self-distillation on adjusted logits, replacing unstable loss maximization with a fixed, well-defined target distribution. Given a frozen original model $\pi_{\mathrm{orig}}$ and a forget sequence $y_f$, let $z_t^{\mathrm{orig}} = \pi_{\mathrm{orig}}^{\mathrm{logits}}(\cdot \mid x, y_{<t})$ denote the original pre-softmax logits at position $t$. To suppress the token $y_t$, *UNDIAL* constructs adjusted logits by subtracting a one-hot penalty:

$$\tilde{z}_t = z_t^{\mathrm{orig}} - \gamma\, \mathbf{e}_{y_t}, \qquad \tilde{p}_t = \mathrm{softmax}(\tilde{z}_t), \tag{30}$$

where $\mathbf{e}_{y_t}$ is the one-hot vector of the target token and $\gamma > 0$ controls the unlearning strength. The model is then optimized to match this adjusted distribution:

$$\mathcal{L}_{\textit{UNDIAL}} = \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\mathrm{forget}}}\left[\sum_{t=1}^{|y_f|} H(\mathrm{sg}(\tilde{p}_t), p_\theta(\cdot \mid x, y_{<t}))\right], \tag{31}$$

where $H(\cdot,\cdot)$ denotes cross-entropy and $\mathrm{sg}(\cdot)$ stops gradients through the adjusted teacher distribution. Intuitively, *UNDIAL* preserves the original model's full predictive structure while selectively demoting the memorized token, often shifting probability mass toward the next most likely alternative. Unlike gradient-ascent-based methods, which lack a finite convergence target and may continue pushing the model toward collapse, *UNDIAL* distills the student toward a stable adjusted distribution, yielding controlled forgetting with smooth optimization and reduced over-unlearning.

***WGA.*** *WGA* (Wang et al., 2025) improves gradient-ascent-based unlearning by mitigating its tendency toward excessive forgetting. Standard gradient ascent decreases the likelihood of forget tokens, but its gradient contains an inverse-confidence effect: as a token probability becomes smaller, already-forgotten tokens can still receive disproportionately large updates, leading to over-unlearning and degradation on retained knowledge. *WGA* addresses this issue by introducing a confidence-dependent token weight:

$$\mathcal{L}_{\textit{WGA}} = \mathbb{E}_{s_u\sim\mathcal{D}_{\mathrm{forget}}} \sum_{i=2}^{|s_u|} w_{u,i}^{\textit{WGA}} \log p(s_u^i \mid s_u^{<i}; \theta), \qquad w_{u,i}^{\textit{WGA}} = p(s_u^i \mid s_u^{<i}; \theta)^\alpha, \tag{32}$$

where $\alpha$ controls the strength of confidence reweighting. Since $w_{u,i}^{\textit{WGA}}$ decreases as the model assigns lower probability to the forget token, tokens that have already been sufficiently suppressed contribute less to subsequent updates. This makes *WGA* behave as an adaptive token-level early-stopping mechanism: it preserves the strong removal ability of gradient ascent on high-confidence memorized tokens while reducing unnecessary updates after forgetting has occurred.

### A.4.2 Aggregation Strategies $\mathcal{A}$

**FedAvg.** FedAvg (McMahan et al., 2017) is a widely used baseline in FL. In each communication round $t$, the server broadcasts the current global model (or the trainable subset of parameters) to a set of participating clients. Each client trains the received model on its own local dataset for several steps and then sends the locally updated parameters (or the corresponding update) back to the server. The server combines client contributions by weighted averaging to obtain the next global model. This training protocol keeps raw client data on-device, which reduces server-side data collection and supports privacy by design. In parameter-efficient FL with LoRA, FedAvg is typically applied to aggregate only the LoRA parameters. Let $\phi_k^{(t,R)}$ denote the LoRA parameters of client $k$ after $R$ local training steps at round $t$. FedAvg updates the global LoRA parameters as

$$\phi^t \;=\; \sum_{k=1}^{K} \alpha_k\, \phi_k^{(t,R)}, \tag{33}$$

where $\alpha_k$ is the aggregation weight for client $k$. A common choice is to set weights proportional to local data sizes:

$$\alpha_k \;=\; \frac{n_k}{\sum_{j=1}^{K} n_j}, \tag{34}$$

with $n_k$ being the number of local training examples at client $k$. This weighting makes clients with more data contribute more to the aggregated model, which matches the goal of optimizing performance over the union of all client data. In practice, FedAvg is simple and communication-efficient, but it can be sensitive to non-IID client data and large numbers of local steps, which may cause client updates to move in different directions and slow down convergence.

**FedAvgM.** FedAvgM (Hsu et al., 2019) extends FedAvg by adding server-side momentum to smooth aggregated updates across communication rounds. Instead of directly replacing the global parameters by a weighted average of client models, FedAvgM first computes the aggregated update direction as the average difference between the locally trained client parameters and the previous global parameters. Letting $\phi_k^{(t,R)}$ denote the LoRA parameters of client $k$ after $R$ local training steps in round $t$, and $\phi^{t-1}$ denote the global LoRA parameters from the previous round, the aggregated update is

$$\Delta_t \;=\; \sum_{k=1}^{K} \alpha_k \Big(\phi_k^{(t,R)} - \phi^{t-1}\Big), \tag{35}$$

where $\alpha_k$ is the aggregation weight (commonly proportional to local data size). FedAvgM then maintains a server momentum vector $m_t$ that accumulates update directions over rounds:

$$m_t \;=\; \begin{cases} \Delta_t, & t = 0, \\ \beta_1 m_{t-1} + \Delta_t, & t > 0, \end{cases} \tag{36}$$

where $\beta_1 \in [0, 1)$ is the server momentum coefficient. Finally, the global parameters are updated using the momentum term:

$$\phi^t \;=\; \phi^{t-1} + m_t. \tag{37}$$

By combining the current aggregated update with a running momentum, FedAvgM reduces oscillations caused by client sampling noise and heterogeneous data distributions, which often leads to more stable training and faster convergence than vanilla FedAvg.

**FedAdagrad.** FedAdagrad (Reddi et al., 2021) is a server-optimized federated method that introduces Adagrad-style adaptive updates into federated learning. The motivation is that standard FL algorithms such as FedAvg can be difficult to tune and may converge slowly or unstably under heterogeneous (non-IID) client data. FedAdagrad follows the FedOpt framework, where clients still perform local training using SGD, while the server applies an adaptive optimizer to aggregate client updates. Let $\phi_k^{(t,R)}$ denote the LoRA parameters obtained by client $k$ after $R$ local training steps in communication round $t$, and let $\phi^{t-1}$ be the global LoRA parameters from the previous round. The server first computes the aggregated update as the weighted average of client displacements:

$$\Delta_t \;=\; \sum_{k=1}^{K} \alpha_k \Big(\phi_k^{(t,R)} - \phi_{t-1}\Big), \tag{38}$$

where $\alpha_k$ is the aggregation weight (commonly proportional to each client's local data size). FedAdagrad then maintains an accumulated squared update vector $v_t$, which tracks the historical magnitude of updates for each parameter coordinate:

$$v_t \;=\; v_{t-1} + \Delta_t^2, \tag{39}$$

where the square is applied element-wise. Finally, the server updates the global parameters using coordinate-wise normalization:

$$\phi^t \;=\; \phi^{t-1} + \epsilon \cdot \frac{\Delta_t}{\sqrt{v_t} + \tau}, \tag{40}$$

where $\epsilon$ is the server learning rate and $\tau > 0$ is a small constant for numerical stability. This adaptive scaling reduces the effective step size for coordinates that have received consistently large updates, while allowing relatively larger steps on infrequently updated coordinates, which often improves training robustness under heterogeneous client updates.

**FedAdam.** FedAdam (Reddi et al., 2021) extends the FedOpt family by applying the Adam optimizer at the server side. Similar to FedAdagrad, clients still perform local training using SGD, while the server maintains adaptive statistics to aggregate client updates more robustly under heterogeneous (non-IID) data. In particular, FedAdam keeps exponential moving averages of both the *first moment* and *second moment* of the aggregated updates, which helps smooth noisy round-to-round updates and automatically scale parameter-wise step sizes. Let $\phi_k^{(t,R)}$ denote the LoRA parameters of client $k$ after $R$ local training steps in round $t$, and let $\phi_{t-1}$ be the global LoRA parameters at the previous round. The server first computes the aggregated update direction as

$$\Delta_t \;=\; \sum_{k=1}^{K} \alpha_k \Big( \phi_k^{(t,R)} - \phi^{t-1} \Big), \tag{41}$$

where $\alpha_k$ is the aggregation weight. FedAdam then updates the server-side first moment accumulator $m_t$ as

$$m_t \;=\; \begin{cases} \Delta_t, & t = 0, \\ \beta_1 m_{t-1} + (1-\beta_1)\Delta_t, & t > 0, \end{cases} \tag{42}$$

and updates the second moment accumulator $v_t$ by

$$v_t \;=\; \beta_2 v_{t-1} + (1-\beta_2)\Delta_t^2, \tag{43}$$

where $\beta_1, \beta_2 \in [0, 1)$ are exponential decay rates and the square is taken element-wise. Finally, the global LoRA parameters are updated by the Adam-style rule

$$\phi^t \;=\; \phi^{t-1} + \epsilon \cdot \frac{m_t}{\sqrt{v_t} + \tau}, \tag{44}$$

where $\epsilon$ is the server learning rate and $\tau > 0$ is a small constant for numerical stability and to control adaptivity. By combining momentum-based smoothing with coordinate-wise normalization, FedAdam often improves training stability and convergence speed compared to FedAvg, especially when client updates have different magnitudes across parameters or vary strongly across rounds.

**FedYogi.** FedYogi (Reddi et al., 2021) is a server-optimized federated method in the FedOpt family. It uses the same overall structure as FedAdam—clients run local SGD and the server aggregates client updates—but it changes how the server updates the second-moment accumulator. The key goal is to prevent the variance estimate from growing too quickly, which can otherwise make the effective step size shrink rapidly and destabilize training under heterogeneous client updates. Let $\phi_k^{(t,R)}$ denote the LoRA parameters of client $k$ after $R$ local training steps in round $t$, and let $\phi^{t-1}$ be the global LoRA parameters from the previous round. The server first computes the aggregated update

$$\Delta_t \;=\; \sum_{k=1}^{K} \alpha_k \Big( \phi_k^{(t,R)} - \phi^{t-1} \Big), \tag{45}$$

where $\alpha_k$ is the aggregation weight. FedYogi then updates the first-moment accumulator in the same way as FedAdam:

$$m_t \;=\; \begin{cases} \Delta_t, & t = 0, \\ \beta_1 m_{t-1} + (1-\beta_1)\Delta_t, & t > 0, \end{cases} \tag{46}$$

with $\beta_1 \in [0, 1)$ controlling the momentum strength. The main difference lies in the second-moment update:

$$v_t \;=\; v_{t-1} - (1-\beta_2)\Delta_t^2 \odot \operatorname{sign}\big(v_{t-1} - \Delta_t^2\big)\,, \tag{47}$$

where $\beta_2 \in [0, 1)$ is a decay rate, $\Delta_t^2$ is taken element-wise, and $\operatorname{sign}(\cdot)$ is applied coordinate-wise. This sign-based correction prevents $v_t$ from increasing monotonically; instead, it allows $v_t$ to adapt up or down depending on whether the current squared update is larger or smaller than the previous estimate. Finally, the server updates the global parameters by

$$\phi^t \;=\; \phi^{t-1} + \epsilon \cdot \frac{m_t}{\sqrt{v_t} + \tau}, \tag{48}$$

where $\epsilon$ is the server learning rate and $\tau > 0$ is a small constant for numerical stability. By controlling the growth of the second-moment estimator, FedYogi often yields more stable training dynamics than FedAdam in federated settings with strong heterogeneity.

**FedProx.** FedProx (Li et al., 2020) modifies FedAvg to better handle practical issues such as heterogeneous (non-IID) client data and partial participation. The main idea is to *regularize local training* so that a client model does not move too far away from the current global model within one communication round. This is done by adding a proximal term to the client-side objective, which directly penalizes large deviations from the global parameters and therefore reduces large, inconsistent local updates that can destabilize training. The server aggregation step remains the same as FedAvg: after receiving locally trained LoRA parameters $\phi_k^{(t,R)}$ from clients, the server computes the global LoRA parameters by weighted averaging,

$$\phi^t \;=\; \sum_{k=1}^{K} \alpha_k \, \phi_k^{(t,R)}, \tag{49}$$

where $\alpha_k$ is the aggregation weight (often proportional to $n_k$, the local data size). The key difference lies in the local optimization problem solved by each client $k$. Instead of minimizing only the empirical loss on its local dataset $\mathcal{D}_k$, FedProx augments the objective with a proximal regularizer:

$$\min_{\phi_k} \;\; \mathcal{L}_k(\phi_k; \mathcal{D}_k) \;+\; \frac{\mu}{2} \big\| \phi_k - \phi^{t-1} \big\|_2^2, \tag{50}$$

where $\mu \geq 0$ is the regularization coefficient and $\phi^{t-1}$ is the global LoRA parameters broadcast by the server at the beginning of round $t$. This proximal term encourages local solutions to stay close to the global model, which reduces client drift caused by non-IID data and allows more stable training when client availability is intermittent or when clients run different amounts of local computation.

**FedNova.** FedNova (Wang et al., 2020) addresses the *objective inconsistency* caused by heterogeneous local progress in federated optimization. In FedAvg, if client $i$ performs $\tau_i$ local updates in one communication round, its cumulative update

$$\Delta_i^t = x_i^{(t,\tau_i)} - x^t = -\eta \sum_{s=0}^{\tau_i - 1} g_i(x_i^{(t,s)}) \tag{51}$$

is directly averaged by the server:

$$x^{t+1} - x^t = \sum_{i=1}^{m} p_i \Delta_i^t. \tag{52}$$

When $\tau_i$ differs across clients, clients with more local steps implicitly contribute larger update magnitudes, so FedAvg no longer optimizes the desired global objective

$$F(x) = \sum_{i=1}^{m} p_i F_i(x), \tag{53}$$

but instead converges toward a biased surrogate objective whose weights depend on local progress.

FedNova removes this bias by normalizing each client's cumulative update before aggregation. For a general local solver, let

$$\Delta_i^t = -\eta G_i^t a_i^t, \qquad d_i^t = \frac{G_i^t a_i^t}{\|a_i^t\|_1} = -\frac{\Delta_i^t}{\eta \|a_i^t\|_1}, \tag{54}$$

where $G_i^t$ stacks the local stochastic gradients and $a_i^t$ describes how they are accumulated by the local optimizer. The server aggregates the normalized directions as

$$x^{t+1} = x^t - \tau_{\text{eff}}^t \sum_{i=1}^{m} p_i \eta d_i^t. \tag{55}$$

For vanilla local SGD, $\|a_i^t\|_1 = \tau_i$, so this becomes

$$x^{t+1} - x^t = \left( \sum_{j=1}^{m} p_j \tau_j \right) \sum_{i=1}^{m} p_i \frac{\Delta_i^t}{\tau_i}. \tag{56}$$

Thus, each client contributes according to its data weight $p_i$, while its update is normalized by the amount of local computation $\tau_i$. The effective step size $\tau_{\text{eff}}^t = \sum_i p_i \tau_i$ preserves the global update scale, whereas the normalization removes the spurious overweighting of fast or more locally trained clients. Consequently, FedNova restores consistency with the true federated objective while retaining the efficiency benefits of heterogeneous local computation.

**FedAW.** FedAW (Tang, 2024) modifies FedAvg by making the server aggregation sensitive to the *quality and bias level* of each client's local dataset. The main idea is that, under non-IID and biased client distributions, not all local updates should contribute equally: clients trained on more representative or less biased data should exert larger influence on the global model, while highly biased clients should be downweighted to prevent bias propagation. To estimate client quality without supervision, FedAW trains local biased/debiased models and assigns each sample a bias-conflict score:

$$s_i = \frac{\mathrm{CE}(x_i, y_i; \phi_k^B)}{\mathrm{CE}(x_i, y_i; \phi_k^B) + \mathrm{CE}(x_i, y_i; \phi_k^D)}, \tag{57}$$

where $\phi_k^B$ and $\phi_k^D$ denote the biased and debiased local models of client $k$, respectively. Bias-aligned samples are easily fitted by the biased model and thus receive smaller scores, whereas bias-conflicting samples are harder for the biased model and receive larger scores. The client-level quality score is then obtained by aggregating sample scores over its local dataset:

$$S_k = \sum_{(x_i, y_i) \in \mathcal{D}_k} s_i. \tag{58}$$

A larger $S_k$ indicates that client $k$ contains more bias-conflicting, informative, or less shortcut-dominated samples, and should therefore be trusted more during aggregation. FedAW replaces the data-size-based averaging weight in FedAvg with a score-adaptive weight:

$$\alpha_k^{(t)} = \frac{g(S_k)}{\sum_{j=1}^K g(S_j)}, \qquad \theta^{t+1} = \sum_{k=1}^K \alpha_k^{(t)} \phi_k^{(t)}, \tag{59}$$

where $\phi_k^{(t)}$ is the locally updated model from client $k$, and $g(\cdot)$ is a weighting function, e.g., a linear mapping for aggressive reweighting or a softmax mapping for smoother aggregation. Unlike FedAvg, which treats clients mainly according to data quantity, FedAW aggregates according to estimated data quality. This adaptive weighting reduces the negative influence of highly biased local models, mitigates client-induced bias propagation.

**FedAWA.** FedAWA (Shi et al., 2025b) improves FedAvg by adaptively optimizing server-side aggregation weights according to the *update directions* of clients rather than assigning fixed weights based only on local data size. The key observation is that the parameter change produced by local training reflects information about the client's data distribution. Therefore, FedAWA defines the client vector of client $k$ at round $t$ as

$$\tau_k^t = \theta_k^t - \theta_g^t, \qquad \tau_g^t = \sum_{k=1}^K \lambda_k^t \tau_k^t, \tag{60}$$

where $\theta_g^t$ is the global model broadcast by the server, $\theta_k^t$ is the locally updated model, and $\tau_k^t$ represents the direction in which client $k$ moves the global model after local training. The aggregated vector $\tau_g^t$ serves as a proxy for the global update direction induced by all clients.

FedAWA assigns larger aggregation weights to clients whose update directions are more aligned with this global direction. Specifically, the server optimizes the aggregation weights by minimizing the weighted deviation between each client vector and the global vector:

$$\boldsymbol{\lambda}^t = \arg\min_{\boldsymbol{\lambda}} \sum_{k=1}^K \lambda_k \left\| \tau_k^t - \tau_g^t \right\|_2, \qquad \text{s.t.} \quad \|\boldsymbol{\lambda}\|_1 = 1. \tag{61}$$

To further stabilize training, FedAWA adds a global alignment regularizer that prevents the newly aggregated model from drifting too far from the current global model:

$$\boldsymbol{\lambda}^t = \arg\min_{\boldsymbol{\lambda}} \left( \sum_{k=1}^K \lambda_k \left\| \tau_k^t - \tau_g^t \right\|_2 + d\left( \sum_{k=1}^K \lambda_k \theta_k^t, \theta_g^t \right) \right), \qquad \text{s.t.} \quad \|\boldsymbol{\lambda}\|_1 = 1, \tag{62}$$

where $d(\cdot, \cdot)$ is implemented as 1 minus cosine similarity. After obtaining $\boldsymbol{\lambda}^t$, the server updates the global model by

$$\theta_g^{t+1} = \sum_{k=1}^K \lambda_k^t \theta_k^t. \tag{63}$$

Thus, unlike FedAvg, which aggregates clients according to data quantity, FedAWA aggregates according to the consistency between each client's update direction and the overall optimization direction. This dynamic weighting reduces the influence of clients whose updates deviate from the global trajectory, mitigates client drift under non-IID data, and improves convergence stability without requiring proxy data or additional privacy-sensitive information.

**FedLWS.** FedLWS (Shi et al., 2025a) improves federated aggregation by applying adaptive *layer-wise weight shrinking* to the aggregated global model. The key idea is that client-update disagreement reflects data heterogeneity: when local gradients vary greatly, the server should shrink the global update more strongly to avoid unstable aggregation; when clients agree, the update can remain close to standard FedAvg. Let the pre-shrunk aggregated model be

$$\widehat{w}_g^{t+1} = \sum_{k=1}^{K} \lambda_k w_k^t = w_g^t - \eta_g g_g^t, \qquad g_g^t = \sum_{k=1}^{K} \lambda_k g_k^t. \tag{64}$$

FedLWS computes a shrinking factor from the variance of local gradients,

$$\tau^t = \frac{1}{K} \sum_{k=1}^{K} \|g_k^t - g_{\text{mean}}^t\|, \qquad g_{\text{mean}}^t = \frac{1}{K} \sum_{k=1}^{K} g_k^t, \tag{65}$$

and balances the optimization term with a weight-shrinking regularizer:

$$\beta\tau^t = \frac{(1-\gamma^t)\|w_g^t\|}{\gamma^t \|\eta_g g_g^t\|}, \qquad \gamma^t = \frac{\|w_g^t\|}{\beta\tau^t \|\eta_g g_g^t\| + \|w_g^t\|}. \tag{66}$$

Thus, larger gradient variance yields smaller $\gamma^t$ and stronger shrinkage, while smaller variance makes $\gamma^t$ close to 1, recovering ordinary aggregation.

To capture layer-specific heterogeneity, FedLWS computes a separate factor for each layer:

$$\gamma_l^t = \frac{\|w_{g,l}^t\|}{\beta\tau_l^t \|\eta_g g_{g,l}^t\| + \|w_{g,l}^t\|}, \qquad \tau_l^t = \frac{1}{K} \sum_{k=1}^{K} \|g_{k,l}^t - g_{\text{mean},l}^t\|. \tag{67}$$

The final global model is obtained by

$$w_{g,l}^{t+1} = \gamma_l^t \widehat{w}_{g,l}^{t+1}. \tag{68}$$

In this way, FedLWS regularizes each layer according to the degree of client disagreement, reducing aggregation bias under non-IID data without using proxy data or altering the local training process. Since it is applied after server aggregation, it can serve as a plug-and-play module for FedAvg, FedProx, FedDyn, and other FL methods.

### A.5 Default Hyperparameters

We run the main simulated PULSAR experiments with three client groups under full participation. For the supplementary experiments with overlapping multi-source and multi-target unlearning requests, we use 100 clients and sample clients at a rate of $\lambda = 0.1$ in each round. We warm up the model for 10 rounds before unlearning to obtain a stable starting state. For controlled evaluation, a source client $m$ issues a targeted unlearning request $q_{m,+}$ against target clients $\mathcal{C}^{\mathcal{T}}$ for five rounds. We fine-tune the base model with LoRA for one local epoch per round with a batch size of 8, using rank 32, $\alpha = 64$, and a dropout ratio of 0.05. We set the learning rate to $8 \times 10^{-5}$ and the weight decay to 0.01. Each filter $\mathcal{F}$ is trained for one epoch. For JDOO, we use the normalized proximal anchoring form in (12) and fix $\mu = 2$ across all experiments. All experiments run on a cloud server with eight NVIDIA A100 GPUs and 1 TB RAM.

## B ALGORITHM

**Algorithm 1:** PULSAR: Who-Forgets-Whom Personalized Federated Unlearning

**Data:** Clients $\mathcal{C}$; total rounds $T$; sampling ratio $\lambda$; local steps $R$; unlearning operator $\mathcal{U}$; aggregation operator $\mathcal{A}$.
**Result:** Global LoRA parameters $\boldsymbol{\phi}^T$.

Initialize global LoRA parameters $\boldsymbol{\phi}^0$; source-to-target map $\mathsf{T} \leftarrow \emptyset$; source-filter cache $\mathsf{F} \leftarrow \emptyset$;
**for** $t = 0$ **to** $T-1$ **do**
  Server receives new requests $\mathcal{Q}_+^t$ and revocation requests $\mathcal{Q}_-^t$;
  **foreach** *source client* $m$ *appearing in* $\mathcal{Q}_+^t \cup \mathcal{Q}_-^t$ **do**
    Let $\mathcal{T}_{m,+}^t$ and $\mathcal{T}_{m,-}^t$ denote the newly requested and revoked targets of $m$, respectively;
    $\mathsf{T}[m] \leftarrow \left(\mathsf{T}[m] \cup \mathcal{T}_{m,+}^t\right) \setminus \mathcal{T}_{m,-}^t$;
    **if** $\mathsf{T}[m] = \emptyset$ **then**
      Remove $m$ from $\mathsf{T}$ and $\mathsf{F}$;
    **else**
      Source client $m$ constructs or refreshes $\mathcal{F}_m^t$ using (4) and (9);
      Server caches $\mathsf{F}[m] \leftarrow \mathcal{F}_m^t$;
  $\mathcal{C}^{\mathcal{S},t} \leftarrow \{m \in \mathcal{C} \mid \mathsf{T}[m] \neq \emptyset\}, \quad \mathcal{C}^{\mathcal{T},t} \leftarrow \bigcup_{m \in \mathcal{C}^{\mathcal{S},t}} \mathsf{T}[m]$;
  Server samples participating clients $\mathcal{C}^t \subseteq \mathcal{C}$ with ratio $\lambda$;
  **foreach** $k \in \mathcal{C}^t$ **do**
    **if** $k \in \mathcal{C}^{\mathcal{T},t}$ **then**
      $\mathcal{S}_k^t \leftarrow \{m \in \mathcal{C}^{\mathcal{S},t} \mid k \in \mathsf{T}[m]\}$;
      Compose $\overline{\mathcal{F}}_k^t$ from $\{\mathsf{F}[m]\}_{m \in \mathcal{S}_k^t}$ using (10);
      Initialize client $k$ with filtered parameters $\widetilde{\boldsymbol{\phi}}_k^{(t,0)}$ using (11);
      Client $k$ performs local JDOO fine-tuning for $R$ steps using (12);
    **else**
      Initialize client $k$ with unfiltered global parameters $\boldsymbol{\phi}_k^{(t,0)} \leftarrow \boldsymbol{\phi}^t$;
      Client $k$ performs standard SOO fine-tuning for $R$ steps using (2);
  Each client $k$ returns $\Delta\boldsymbol{\phi}_k^t = \boldsymbol{\phi}_k^{(t,R)} - \boldsymbol{\phi}_k^{(t,0)}$ as in (7);
  Server aggregates $\{\Delta\boldsymbol{\phi}_k^t\}_{k \in \mathcal{C}^t}$ using (8) to obtain $\boldsymbol{\phi}^{t+1}$;
**return** $\boldsymbol{\phi}^T$;

## C THEORETICAL ANALYSIS OF JDOO

This section provides a theoretical analysis of the local JDOO objective in (12). The goal is to show that the proximal anchoring term explicitly controls the retained-utility drift of the filtered target model. For compactness, let $f_n(\boldsymbol{\phi}) = \mathcal{L}_n^{\mathrm{FT}}\left(\boldsymbol{\phi}; \mathcal{D}_n^{\mathcal{T}}\right), \boldsymbol{a}_n^t = \mathrm{sg}\left(\boldsymbol{\phi}_n^{\mathrm{prev}}\right)$, Then (12) can be written as

$$J_n^t(\boldsymbol{\phi}) = f_n(\boldsymbol{\phi}) + \frac{\mu}{2d}\left\|\boldsymbol{\phi} - \boldsymbol{a}_n^t\right\|_2^2. \tag{69}$$

Define the normalized adapter distance as

$$\Delta(\boldsymbol{\phi}, \boldsymbol{a}_n^t) = \frac{1}{\sqrt{d}}\left\|\boldsymbol{\phi} - \boldsymbol{a}_n^t\right\|_2. \tag{70}$$

Under this notation, the proximal term in (69) becomes $\frac{\mu}{2}\Delta^2(\boldsymbol{\phi}, \boldsymbol{a}_n^t)$, i.e., it penalizes the average parameter-wise deviation from the retained-utility anchor. This explains the normalization factor $1/|\boldsymbol{\phi}|$ in (12).

**Assumption 1** (Local regularity)**.** *In the local region visited by JDOO, the target-client FT loss $f_n(\boldsymbol{\phi})$ is lower bounded by $f_n^\star$ and is $G_n$-Lipschitz continuous with respect to the normalized adapter distance, i.e.,*

$$|f_n(\boldsymbol{\phi}) - f_n(\boldsymbol{\phi}')| \le G_n \Delta(\boldsymbol{\phi}, \boldsymbol{\phi}') \quad \textit{for any } \boldsymbol{\phi}, \boldsymbol{\phi}'. \tag{71}$$

**Assumption 2** (Approximate local optimization)**.** *Let $\widehat{\boldsymbol{\phi}}_n^t$ be the target model returned by JDOO after local training. We assume that it is $\epsilon$-accurate relative to the anchor solution:*

$$J_n^t(\widehat{\boldsymbol{\phi}}_n^t) \le J_n^t(\boldsymbol{a}_n^t) + \epsilon = f_n(\boldsymbol{a}_n^t) + \epsilon, \tag{72}$$

*where $\epsilon \ge 0$ captures finite-step local optimization error. For an exact minimizer of (69), $\epsilon = 0$.*

**Proposition 1** (Retained-utility drift control)**.** *Under Assumptions 1 and 2, the normalized drift between the JDOO-updated target model $\widehat{\boldsymbol{\phi}}_n^t$ and the retained-utility anchor $\boldsymbol{a}_n^t$ satisfies*

$$\Delta\left(\widehat{\boldsymbol{\phi}}_n^t, \boldsymbol{a}_n^t\right) \le \sqrt{\frac{2\left(f_n(\boldsymbol{a}_n^t) - f_n^\star + \epsilon\right)}{\mu}}. \tag{73}$$

*Consequently, the retained-utility loss drift is bounded by*

$$\left|f_n(\widehat{\boldsymbol{\phi}}_n^t) - f_n(\boldsymbol{a}_n^t)\right| \le G_n \sqrt{\frac{2\left(f_n(\boldsymbol{a}_n^t) - f_n^\star + \epsilon\right)}{\mu}}. \tag{74}$$

*Proof.* From Assumption 2, we have

$$f_n(\widehat{\boldsymbol{\phi}}_n^t) + \frac{\mu}{2}\Delta^2\left(\widehat{\boldsymbol{\phi}}_n^t, \boldsymbol{a}_n^t\right) \le f_n(\boldsymbol{a}_n^t) + \epsilon. \tag{75}$$

Since $f_n(\widehat{\boldsymbol{\phi}}_n^t) \ge f_n^\star$, it follows that

$$\frac{\mu}{2}\Delta^2\left(\widehat{\boldsymbol{\phi}}_n^t, \boldsymbol{a}_n^t\right) \le f_n(\boldsymbol{a}_n^t) - f_n^\star + \epsilon. \tag{76}$$

Rearranging gives (73). Applying the Lipschitz condition in Assumption 1 with $\boldsymbol{\phi} = \widehat{\boldsymbol{\phi}}_n^t$ and $\boldsymbol{\phi}' = \boldsymbol{a}_n^t$ yields (74). □

**Interpretation.** Proposition 1 shows that JDOO explicitly controls how far the filtered target model can move away from the previous local model of the same target client. The bound decreases as $\mu$ increases, indicating that the anchoring term reduces retained-utility drift. When $\mu = 0$, (12) degenerates to ordinary local SOO fine-tuning from the filtered initialization, corresponding to the F+SOO ablation. When $\mu > 0$, the update is coupled to the retained-utility anchor.

This can also be seen from the gradient of (69):

$$\nabla J_n^t(\boldsymbol{\phi}) = \nabla f_n(\boldsymbol{\phi}) + \frac{\mu}{d}\left(\boldsymbol{\phi} - \boldsymbol{a}_n^t\right). \tag{77}$$

Thus, each JDOO step consists of the standard local FT gradient plus a pull-back force toward the retained-utility anchor. This explains why JDOO can recover target-client utility after dispatch-time filtering while avoiding unconstrained drift from the target client's previously learned local behavior.

# D Theoretical Convergence Analysis of PULSAR

We provide a convergence analysis for PULSAR under a FedAvg-style aggregation rule. We show that PULSAR preserves the standard non-convex FedAvg convergence behavior up to an additional controllable neighborhood determined by the active target mass, the filter magnitude, and the JDOO anchoring strength.

## D.1 Notation and PULSAR Update

Let $d = |\phi|$ denote the number of trainable adapter parameters. For client $k$, define the local fine-tuning objective as

$$f_k(\phi) := \mathcal{L}_k^{\mathrm{FT}}(\phi; D_k), \tag{78}$$

and the global federated objective as

$$f(\phi) := \sum_{k=1}^{K} p_k f_k(\phi), \qquad p_k = \frac{N_k}{\sum_{j=1}^{K} N_j}. \tag{79}$$

At communication round $t$, let $\mathcal{S}_t \subseteq \mathcal{C}$ be the sampled client set, and let $p_k^t$ be the normalized aggregation weight over $\mathcal{S}_t$.

PULSAR assigns each sampled client an initialization

$$x_{k,0}^t = \phi^t + \delta_k^t, \tag{80}$$

where

$$\delta_k^t = \begin{cases} -\bar{F}_k^t, & k \in \mathcal{C}^{T,t}, \\ 0, & k \notin \mathcal{C}^{T,t}. \end{cases} \tag{81}$$

Here, $\bar{F}_k^t$ is the target-specific composed filter. Thus, non-target clients are initialized from the global adapter $\phi^t$, while target clients are initialized from the filtered adapter $\phi^t - \bar{F}_k^t$.

For target clients, PULSAR optimizes the JDOO objective

$$h_k^t(\phi) = f_k(\phi) + \mathbf{1}\{k \in \mathcal{C}^{T,t}\} \frac{\mu}{2d} \|\phi - a_k^t\|_2^2, \tag{82}$$

where $a_k^t$ denotes the retained-utility anchor, i.e., the locally trained adapter from the last participation of client $k$. For non-target clients, $h_k^t(\phi) = f_k(\phi)$.

Each sampled client performs $R$ local stochastic gradient steps:

$$x_{k,r+1}^t = x_{k,r}^t - \eta g_k^t(x_{k,r}^t; \xi_{k,r}^t), \qquad r = 0, \ldots, R-1, \tag{83}$$

where $g_k^t(x; \xi)$ is a stochastic gradient estimator of $\nabla h_k^t(x)$.

**Dispatch-time filtering and aggregation convention.** A subtle difference between PULSAR and standard FedAvg is that sampled clients may start local training from different initializations. Non-target clients start from $\phi^t$, whereas target clients start from $\phi^t - \bar{F}_k^t$. Therefore, directly averaging the returned absolute parameters would mix the dispatch-time filter offsets into the next global adapter. This is undesirable for PULSAR, because the filter is intended to be a recipient-specific access-control perturbation rather than a permanent modification of the global adapter.

To capture the plug-and-play nature of PULSAR, we formulate aggregation in terms of the local training update relative to each client's dispatched initialization. Specifically, after receiving $x_{k,0}^t$ and performing $R$ local steps, client $k$ contributes

$$\Delta_k^t := x_{k,R}^t - x_{k,0}^t. \tag{84}$$

The server then updates the global adapter by aggregating these local training updates:

$$\phi^{t+1} = \phi^t + \sum_{k \in \mathcal{S}_t} p_k^t \Delta_k^t. \tag{85}$$

Under this convention, the filter $\bar{F}_k^t$ affects the target client's local optimization trajectory through its initialization, but the filter offset itself is not directly averaged into the global model. This preserves the recoverable dispatch-time semantics of PULSAR: once a source-to-target forgetting relation is revoked, the server can stop applying the corresponding filter without needing to undo a permanent global offset.

Using (83), (85) can be equivalently written as

$$\phi^{t+1} = \phi^t - \eta R v_t, \tag{86}$$

where

$$v_t = \frac{1}{R} \sum_{r=0}^{R-1} \sum_{k \in \mathcal{S}_t} p_k^t g_k^t(x_{k,r}^t; \xi_{k,r}^t). \tag{87}$$

### D.2 Assumptions

We make the following standard assumptions for non-convex federated optimization.

**Assumption 1: Smoothness and lower boundedness.** Each local objective $f_k$ is $L$-smooth, i.e., for any $\phi, \phi'$,

$$\|\nabla f_k(\phi) - \nabla f_k(\phi')\|_2 \le L\|\phi - \phi'\|_2. \tag{88}$$

The global objective $f$ is lower bounded by $f^\star$. Since JDOO adds a quadratic anchor, $h_k^t$ is $L_\mu$-smooth with

$$L_\mu = L + \frac{\mu}{d}. \tag{89}$$

**Assumption 2: Unbiased stochastic gradients and bounded variance.** For each client $k$, round $t$, local step $s$, and model parameter $x$,

$$\mathbb{E}_\xi\left[g_k^t(x;\xi)\right] = \nabla h_k^t(x), \tag{90}$$

and

$$\mathbb{E}_\xi\left[\|g_k^t(x;\xi) - \nabla h_k^t(x)\|_2^2\right] \le \sigma^2. \tag{91}$$

We also assume a bounded second moment:

$$\mathbb{E}_\xi\|g_k^t(x;\xi)\|_2^2 \le G^2. \tag{92}$$

**Assumption 3: Bounded client heterogeneity.** There exists $\zeta \ge 0$ such that for any $\phi$,

$$\sum_{k=1}^{K} p_k\|\nabla f_k(\phi) - \nabla f(\phi)\|_2^2 \le \zeta^2. \tag{93}$$

**Assumption 4: Bounded PULSAR perturbations.** For every active target client $k \in \mathcal{C}^{T,t}$, the composed filter and the retained-utility anchor satisfy

$$\|\bar{F}_k^t\|_2 \le \rho_F, \qquad \|\phi^t - a_k^t\|_2 \le \rho_A. \tag{94}$$

Let the sampled target mass be

$$\gamma_t = \sum_{k \in \mathcal{S}_t \cap \mathcal{C}^{T,t}} p_k^t. \tag{95}$$

We assume $\gamma_t \le \gamma$ for all $t$. Here, $\gamma$ measures the fraction of the aggregated update affected by active target-side unlearning.

### D.3 PULSAR-Induced Bias

PULSAR differs from standard FedAvg in two aspects. First, target clients evaluate gradients from the filtered initialization $\phi^t - \bar{F}_k^t$ instead of $\phi^t$. Second, target clients optimize the JDOO objective $h_k^t$, whose gradient contains an anchor term.

For a target client $k$, we have

$$\nabla h_k^t(x) = \nabla f_k(x) + \frac{\mu}{d}(x - a_k^t). \tag{96}$$

At the beginning of local training, $x_{k,0}^t = \phi^t - \bar{F}_k^t$. Hence,

$$\|x_{k,0}^t - a_k^t\|_2 \le \|\phi^t - a_k^t\|_2 + \|\bar{F}_k^t\|_2 \le \rho_A + \rho_F. \tag{97}$$

Therefore, the aggregate JDOO anchor contributes a bounded gradient bias:

$$\left\| \sum_{k \in \mathcal{S}_t \cap \mathcal{C}^{T,t}} p_k^t \frac{\mu}{d}(x_{k,0}^t - a_k^t) \right\|_2 \le \gamma \frac{\mu}{d}(\rho_A + \rho_F). \tag{98}$$

Similarly, by $L$-smoothness,

$$\|\nabla f_k(\phi^t - \bar{F}_k^t) - \nabla f_k(\phi^t)\|_2 \le L\rho_F. \tag{99}$$

Thus, the aggregate gradient bias caused by evaluating target-client gradients at filtered initializations is bounded by

$$L\gamma\rho_F. \tag{100}$$

Collecting these two PULSAR-specific terms, define the PULSAR perturbation level as

$$B_{\mathrm{PULSAR}}^2 := L^2\gamma^2\rho_F^2 + \gamma^2\frac{\mu^2}{d^2}(\rho_A + \rho_F)^2. \tag{101}$$

### D.4 Main Convergence Result

**Theorem 1** (Non-convex convergence of PULSAR-FedAvg)**.** *Suppose Assumptions 1–4 hold. Let PULSAR run for $T$ communication rounds with $R$ local steps per round. If the local learning rate satisfies*

$$\eta R \le \frac{1}{4L}, \tag{102}$$

*then the global iterates generated by PULSAR satisfy*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\phi^t)\|_2^2 \le \mathcal{O}\left(\frac{f(\phi^0)-f^\star}{\eta RT}\right) + \mathcal{O}\left(\frac{L\eta R}{|\mathcal{S}_t|}(\sigma^2+\zeta^2)\right) + \mathcal{O}\left(\eta^2 L_\mu^2 R^2(\sigma^2+\zeta^2+G^2)\right) + \mathcal{O}\left(B_{\mathrm{PULSAR}}^2\right). \tag{103}$$

*Equivalently,*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\phi^t)\|_2^2 \le \mathcal{O}\left(\frac{f(\phi^0)-f^\star}{\eta RT} + \frac{L\eta R}{|\mathcal{S}_t|}(\sigma^2+\zeta^2) + \eta^2 L_\mu^2 R^2(\sigma^2+\zeta^2+G^2)\right) + \mathcal{O}\left(L^2\gamma^2\rho_F^2 + \gamma^2\frac{\mu^2}{d^2}(\rho_A+\rho_F)^2\right). \tag{104}$$

*Proof.* By the $L$-smoothness of $f$, we have

$$f(\phi^{t+1}) \le f(\phi^t) + \langle \nabla f(\phi^t), \phi^{t+1}-\phi^t\rangle + \frac{L}{2}\|\phi^{t+1}-\phi^t\|_2^2. \tag{105}$$

Using $\phi^{t+1} = \phi^t - \eta R v_t$, this gives

$$f(\phi^{t+1}) \le f(\phi^t) - \eta R\langle \nabla f(\phi^t), v_t\rangle + \frac{L\eta^2R^2}{2}\|v_t\|_2^2. \tag{106}$$

Let

$$e_t := v_t - \nabla f(\phi^t). \tag{107}$$

Then

$$\langle \nabla f(\phi^t), v_t\rangle = \|\nabla f(\phi^t)\|_2^2 + \langle \nabla f(\phi^t), e_t\rangle. \tag{108}$$

By Young's inequality,

$$-\langle \nabla f(\phi^t), e_t\rangle \le \frac{1}{4}\|\nabla f(\phi^t)\|_2^2 + \|e_t\|_2^2. \tag{109}$$

Moreover,

$$\|v_t\|_2^2 = \|\nabla f(\phi^t)+e_t\|_2^2 \le 2\|\nabla f(\phi^t)\|_2^2 + 2\|e_t\|_2^2. \tag{110}$$

Substituting these inequalities into (106) and using $\eta R \le 1/(4L)$, we obtain

$$\mathbb{E}f(\phi^{t+1}) \le \mathbb{E}f(\phi^t) - c_1\eta R\mathbb{E}\|\nabla f(\phi^t)\|_2^2 + c_2\eta R\mathbb{E}\|e_t\|_2^2, \tag{111}$$

for some universal constants $c_1, c_2 > 0$.

It remains to bound $\mathbb{E}\|e_t\|_2^2$. We decompose $e_t$ into standard FedAvg errors and PULSAR-specific errors:

$$e_t = e_t^{\mathrm{stoch}} + e_t^{\mathrm{sample}} + e_t^{\mathrm{drift}} + e_t^{\mathrm{filter}} + e_t^{\mathrm{jdoo}}. \tag{112}$$

The first three terms correspond to stochastic gradient noise, client-sampling error, and local update drift. Under Assumptions 1–3, standard FedAvg arguments yield

$$\mathbb{E}\|e_t^{\mathrm{stoch}} + e_t^{\mathrm{sample}}\|_2^2 \le \mathcal{O}\left(\frac{\sigma^2+\zeta^2}{|\mathcal{S}_t|}\right), \tag{113}$$

and

$$\mathbb{E}\|e_t^{\mathrm{drift}}\|_2^2 \le \mathcal{O}\left(\eta^2 L_\mu^2 R^2(\sigma^2+\zeta^2+G^2)\right). \tag{114}$$

For the PULSAR-specific terms, (100) gives

$$\mathbb{E}\|e_t^{\mathrm{filter}}\|_2^2 \le \mathcal{O}\left(L^2\gamma^2\rho_F^2\right). \tag{115}$$

Similarly, (98) gives

$$\mathbb{E}\|e_t^{\mathrm{jdoo}}\|_2^2 \le \mathcal{O}\left(\gamma^2 \frac{\mu^2}{d^2}(\rho_A + \rho_F)^2\right). \tag{116}$$

Combining (113)–(116), we obtain

$$\begin{aligned}\mathbb{E}\|e_t\|_2^2 \le \mathcal{O}\left(\frac{\sigma^2+\zeta^2}{|\mathcal{S}_t|}\right) + \mathcal{O}\left(\eta^2 L_\mu^2 R^2(\sigma^2+\zeta^2+G^2)\right) \\ + \mathcal{O}\left(B_{\mathrm{PULSAR}}^2\right).\end{aligned} \tag{117}$$

Substituting (117) into (111), summing over $t = 0, \dots, T-1$, and using $f(\phi^T) \ge f^\star$, we obtain

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\phi^t)\|_2^2 \le \mathcal{O}\left(\frac{f(\phi^0)-f^\star}{\eta R T}\right) + \mathcal{O}\left(\frac{L\eta R}{|\mathcal{S}_t|}(\sigma^2+\zeta^2)\right) \\ + \mathcal{O}\left(\eta^2 L_\mu^2 R^2(\sigma^2+\zeta^2+G^2)\right) + \mathcal{O}\left(B_{\mathrm{PULSAR}}^2\right),\end{aligned} \tag{118}$$

which completes the proof. □

### D.5 Consequences

**Corollary 1** (Same-order convergence with bounded PULSAR perturbation)**.** *Assume $R$ is fixed. Choosing $\eta = \mathcal{O}(1/\sqrt{T})$ gives*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\phi^t)\|_2^2 = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right) + \mathcal{O}\left(L^2\gamma^2\rho_F^2 + \gamma^2\frac{\mu^2}{d^2}(\rho_A+\rho_F)^2\right). \tag{119}$$

Corollary 1 shows that PULSAR converges to a stationary neighborhood of the standard federated fine-tuning objective. The radius of this neighborhood is controlled by the active target mass $\gamma$, the filter magnitude $\rho_F$, and the JDOO anchoring strength $\mu$. When there is no active target client, i.e., $\gamma = 0$, the additional PULSAR term vanishes and the result reduces to the standard FedAvg convergence behavior.

**Corollary 2** (Effect of dispatch-time filtering)**.** *Under the update-relative aggregation in (85), the filter offset does not directly enter the global update. Filters affect convergence only through gradient evaluation at personalized target initializations and through the JDOO anchor.*

Corollary 2 formalizes the plug-and-play property of PULSAR. The filter changes the target client's local optimization trajectory, but is not directly averaged into the global adapter. This explains why the forgetting effect can be revoked by deactivating the corresponding source-to-target filter relation.

**Corollary 3** (Utility anchoring effect of JDOO)**.** *Let $\hat{\phi}_k^t$ be an $\epsilon$-approximate minimizer of the target-side JDOO objective*

$$h_k^t(\phi) = f_k(\phi) + \frac{\mu}{2d}\|\phi - a_k^t\|_2^2. \tag{120}$$

*Then*

$$\|\hat{\phi}_k^t - a_k^t\|_2 \le \sqrt{\frac{2d\left(f_k(a_k^t) - f_k^\star + \epsilon\right)}{\mu}}. \tag{121}$$

*If $f_k$ is $G_k$-Lipschitz, the retained-utility drift is bounded by*

$$|f_k(\hat{\phi}_k^t) - f_k(a_k^t)| \le G_k\sqrt{\frac{2d\left(f_k(a_k^t) - f_k^\star + \epsilon\right)}{\mu}}. \tag{122}$$

*Proof.* Since $\hat{\phi}_k^t$ is an $\epsilon$-approximate minimizer of $h_k^t$, we have

$$h_k^t(\hat{\phi}_k^t) \le h_k^t(a_k^t) + \epsilon = f_k(a_k^t) + \epsilon. \tag{123}$$

Therefore,

$$f_k(\hat{\phi}_k^t) + \frac{\mu}{2d}\|\hat{\phi}_k^t - a_k^t\|_2^2 \le f_k(a_k^t) + \epsilon. \tag{124}$$

Since $f_k(\hat{\phi}_k^t) \geq f_k^\star$, we obtain

$$\frac{\mu}{2d}\|\hat{\phi}_k^t - a_k^t\|_2^2 \leq f_k(a_k^t) - f_k^\star + \epsilon. \tag{125}$$

Taking square roots gives the first claim. The second claim follows directly from the Lipschitz continuity of $f_k$. □

Corollary 3 shows that increasing $\mu$ restricts how far the target model can drift from its retained-utility anchor. Therefore, JDOO mitigates the utility degradation induced by dispatch-time filtering, while Theorem 1 shows that this anchoring introduces only a bounded optimization bias.

### D.6 Discussion

The above analysis provides three insights. First, PULSAR preserves the standard FedAvg convergence structure. The first terms in Theorem 1 correspond to the usual optimization error, stochastic gradient noise, client sampling, local drift, and data heterogeneity. Thus, PULSAR does not fundamentally change the convergence order of federated fine-tuning.

Second, the additional PULSAR-specific convergence neighborhood is controlled by

$$L^2\gamma^2\rho_F^2 + \gamma^2\frac{\mu^2}{d^2}(\rho_A + \rho_F)^2. \tag{126}$$

This term is small when the active target mass $\gamma$ is small, the composed filters are bounded, and the anchor strength $\mu$ is not excessively large. This matches the intended personalized unlearning setting, where only a subset of clients receives filtered models.

Third, the analysis explains why PULSAR can support recoverable dispatch-time unlearning. Since the global adapter is updated by aggregating local training deltas relative to each client's dispatched initialization, the filter offset is not directly written into the global adapter. Therefore, when a forgetting relation is revoked, the server can simply stop composing and dispatching the corresponding filter for the target client, without requiring an explicit global-model rollback.

## E Conditional Inheritance of Unlearning-Operator Properties

PULSAR is designed as an operator-general framework and therefore does not, by itself, establish an unlearning guarantee for an arbitrary underlying operator $\mathcal{U}$. Such guarantees generally depend on the properties of $\mathcal{U}$, the training procedure, data distribution, and the corresponding evaluation or adversarial assumptions. Nevertheless, PULSAR admits an exact inheritance result in a restricted single-source, temporally aligned, pre-JDOO setting.

**Aligned single-source dispatch.** Consider a source client $m$ with forget set $\widetilde{\mathcal{D}}_m$. Given a global adapter $\boldsymbol{\phi}$, let $\boldsymbol{\phi}_{\mathcal{U}} = \mathcal{U}\left(\boldsymbol{\phi}, \widetilde{\mathcal{D}}_m\right)$, and define the corresponding source-specific filter as $\mathcal{F}_m = \boldsymbol{\phi} - \boldsymbol{\phi}_{\mathcal{U}}$. For a target client $n$ associated only with source $m$, suppose that the filter is constructed from and applied to the same global checkpoint. The target initialization dispatched by PULSAR is then

$$\begin{aligned}\widetilde{\boldsymbol{\phi}}_n^0 &= \boldsymbol{\phi} - \mathcal{F}_m \\ &= \boldsymbol{\phi} - (\boldsymbol{\phi} - \boldsymbol{\phi}_{\mathcal{U}}) \\ &= \boldsymbol{\phi}_{\mathcal{U}} = \mathcal{U}\left(\boldsymbol{\phi}, \widetilde{\mathcal{D}}_m\right).\end{aligned} \tag{127}$$

Therefore, in this aligned setting, the model dispatched by PULSAR is exactly the output of the underlying unlearning operator.

**Conditional property inheritance.** Let $\mathcal{L}(\cdot)$ denote the distribution of the resulting model under any randomness in the underlying procedure, and let $\mathcal{R}_m$ denote the corresponding recipient-specific retraining reference distribution. For any distance or discrepancy measure $d(\cdot,\cdot)$, if the underlying operator satisfies

$$d\left(\mathcal{L}\left(\mathcal{U}\left(\boldsymbol{\phi}, \widetilde{\mathcal{D}}_m\right)\right), \mathcal{R}_m\right) \leq \epsilon, \tag{128}$$

under a given set of assumptions, then (127) immediately gives

$$d\left(\mathcal{L}\left(\widetilde{\phi}_n^0\right), \mathcal{R}_m\right) \leq \epsilon. \tag{129}$$

Hence, temporally aligned PULSAR preserves any such property already established for $\mathcal{U}$ under exactly the same assumptions. This is an equivalence and inheritance result rather than a new guarantee of unlearning, privacy, or security for $\mathcal{U}$.

**One-round-lag refresh case.** The exact equality above does not hold for a lagged refresh in which the filter constructed from $\phi^{t-1}$ is applied to the next global adapter $\phi^t$. Specifically,

$$\mathcal{F}_m^t = \phi^{t-1} - \mathcal{U}\left(\phi^{t-1}, \widetilde{\mathcal{D}}_m\right), \tag{130}$$

and therefore

$$\begin{aligned}\widetilde{\phi}_n^{(t,0)} &= \phi^t - \mathcal{F}_m^t \\ &= \mathcal{U}\left(\phi^{t-1}, \widetilde{\mathcal{D}}_m\right) + \left(\phi^t - \phi^{t-1}\right).\end{aligned} \tag{131}$$

Thus, extending (129) to the lagged pipeline requires an additional stability assumption with respect to the inter-round displacement $\|\phi^t - \phi^{t-1}\|$. Our temporal-lag experiments in Appendix F.1 empirically evaluate this discrepancy. However, the same exact inheritance result also does not extend automatically to multi-source filter composition or to subsequent JDOO updates.

# F Experimental Results

## F.1 Temporal Inconsistency of Filter

When PULSAR refreshes a source-specific filter, it constructs the filter from the latest preceding global checkpoint available to the source client and applies it to the current global checkpoint. For source client $m$,

$$\phi^{t-1} \xrightarrow[\text{on } m]{\mathcal{U}(\cdot, \tilde{\mathcal{D}}_m)} \widetilde{\phi}_m^{t-1}, \qquad \mathcal{F}_m^t = \phi^{t-1} - \widetilde{\phi}_m^{t-1}, \qquad \widetilde{\phi}_n^{(t,0)} = \phi^t - \mathcal{F}_m^t. \tag{132}$$

This introduces a one-round temporal lag because $\mathcal{F}_m^t$ is derived from $\phi^{t-1}$ but dispatched with $\phi^t$. The design avoids an additional communication round that would otherwise be required to construct and apply the filter from the same checkpoint, at the cost of a small mismatch between consecutive global states.

To evaluate the effect of this mismatch, we compare a strictly aligned $t$-0 variant with lagged $t$-1 through $t$-4 variants. Table 2 shows that, under Llama-3.2-1B with FedAvg, the aligned $t$-0 setting performs similarly to the lagged $t$-1 setting. The standard deviations of MU and FTR across the five lag settings remain below $2\%$, and the overall trends are unchanged. These results indicate that a one-round lag introduces limited empirical distortion under the evaluated setting while avoiding the extra communication required by strict temporal alignment.

## F.2 Lightweight Filter $\mathcal{F}$ Analysis

Since $\mathcal{F}$ resides in the same parameter space as the LoRA adapters—i.e., it is parameterized by the same set of adapter weights—its size matches the number of LoRA trainable parameters. Table 3 reports the parameter statistics across LLM backbones and model scales. Despite scaling the base model from 1B to 8B parameters, $\mathcal{F}$ consistently equals $N_{\textbf{Trainable}}$ and accounts for only 1.0%–2.3% of the base model parameters (e.g., 1.034% for Llama-3.1-8B and 2.336% for Qwen2.5-1.5B). Consequently, for each targeted unlearning request, the source client $\mathcal{C}^{\mathcal{S}}$ uploads and the server $\mathcal{S}$ maintains a lightweight adapter-sized delta rather than a full model checkpoint or dense gradients, substantially mitigating the communication and storage overhead that could otherwise accumulate under repeated revoke/restore operations. This observation directly supports our motivation in **RQ2**: enabling recoverable, plug-and-play authorization changes without incurring prohibitive system costs in non-stationary federated settings.

Table 2: Robustness of PULSAR to filter temporal lag. We report MU/FTR on Llama-1B with FedAvg, and Std. is computed across lag variants.

| **Llama-1B-Instruct with LoRA** ($T = 11$) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **Lag** | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ |
| $t$-0 | 0.549 | 0.578 | 0.513 | 0.666 | 0.513 | 0.666 | 0.502 | 0.665 | 0.495 | 0.662 | 0.541 | 0.681 |
| $t$-1 | 0.549 | 0.578 | 0.514 | 0.669 | 0.514 | 0.669 | 0.511 | 0.65 | 0.509 | 0.632 | 0.541 | 0.681 |
| $t$-2 | 0.549 | 0.578 | 0.501 | 0.687 | 0.512 | 0.68 | 0.502 | 0.667 | 0.506 | 0.636 | 0.541 | 0.681 |
| $t$-3 | 0.549 | 0.578 | 0.512 | 0.686 | 0.512 | 0.686 | 0.508 | 0.66 | 0.503 | 0.646 | 0.541 | 0.681 |
| $t$-4 | 0.549 | 0.578 | 0.508 | 0.692 | 0.508 | 0.692 | 0.508 | 0.662 | 0.509 | 0.63 | 0.541 | 0.681 |
| **Std.** | - | - | 0.005 | 0.01 | 0.002 | 0.01 | 0.004 | 0.006 | 0.005 | 0.012 | - | - |

Table 3: Parameter statistics for the base model and LoRA adapters, showing that the filter $\mathcal{F}$ has the same parameter count as $N_{\textbf{Trainable}}$ and incurs only a small fraction of the base model size.

| **Model** | **Size** | $N_{\textbf{Base}}$ | $N_{\textbf{Trainable}}$ | $\mathcal{F}$ | **Ratio (%)** |
|---|---|---|---|---|---|
| Llama-3.1-Instruct | 8B | 8114.15 M | 83.89 M | 83.89 M | 1.034 |
| Llama-3.2-Instruct | 1B | 1258.36 M | 22.54 M | 22.54 M | 1.792 |
| | 3B | 3261.38 M | 48.63 M | 48.63 M | 1.491 |
| Qwen2.5-Instruct | 1.5B | 1580.64 M | 36.93 M | 36.93 M | 2.336 |
| | 3B | 3145.81 M | 59.87 M | 59.87 M | 1.903 |
| | 7B | 7696.36 M | 80.74 M | 80.74 M | 1.049 |

### F.3 Split Ratio Analysis

Table 4 reports the MU and FTR on $\mathcal{C}^{\mathcal{T}}$ at $T$=11 under three **TOFU** split protocols (Split90/95/99), spanning two model families (Llama and Qwen) and multiple parameter scales. In most settings, *Finetune* achieves the highest MU but consistently exhibits the lowest FTR, confirming that standard finetuning fails to induce effective forgetting. *Retrain* serves as a strong forgetting reference, typically attaining high FTR, while its MU is not always optimal and requires training from scratch. Across all splits and model scales, PULSAR yields a favorable utility–forgetting trade-off: it substantially improves FTR over *Finetune*, often approaching the forgetting strength of *Retrain*, while incurring only modest MU degradation. Moreover, as the retain ratio increases (Split90 → Split99), FTR decreases for all methods, suggesting that forgetting becomes harder when the forget set is smaller. Notably, under high-retain splits (e.g., Split99), *Retrain* becomes more sensitive to retain-only optimization and its FTR drops, whereas PULSAR remains more robust by enforcing explicit forgetting constraints via $\mathcal{F}$ and JDOO, achieving stronger targeted forgetting without the retraining cost. Overall, the same trends hold across splits, backbones, and scales, demonstrating robust behavior under varying forget/retain ratios.

### F.4 Llama-3.2-1B and Qwen2.5-1.5B with MU/FTR/ROUGE across $\mathcal{A}$ and $\mathcal{U}$

**Llama-3.2-1B.** Tables 5 and 6 show that PULSAR improves targeted forgetting across most aggregation operators $\mathcal{A}$. For MU/FTR, all operator-based methods increase FTR over *Finetune*, while introducing a moderate MU drop, reflecting the expected forgetting–utility trade-off. Among them, *CEU* achieves the best or tied-best FTR under several aggregators, including *FedAvg*, *FedAdam*, *FedProx*, *FedAvgM*, *FedYogi*, and *FedLWS*. *GradAscent* and *GradDiff* also remain competitive and stable. For example, under *FedAvg*, FTR improves from $0.578$ for *Finetune* to $0.671$ with *GradAscent*, $0.667$ with *GradDiff*, and $0.682$ with *CEU*, approaching or matching the *Retrain* reference at $0.681$. Under *FedNova*, *GradAscent* and *GradDiff* reach $0.687$ and $0.688$, respectively, slightly exceeding *Retrain* at $0.684$.

The ROUGE-based metrics further support this trend. Since lower FOR indicates stronger forgetting, *CEU* and *GradDiff* are particularly effective: *CEU* obtains the best FOR under multiple aggregators such as *FedAvg*, *FedAvgM*, *FedYogi*, *FedAWA*, and *FedLWS*, while *GradDiff* performs best under *FedProx*, *FedNova*, *FedAdagrad*, and *FedAW*. RER varies across operators, showing that stronger

Table 4: Model performance and unlearning effectiveness for different split ratios ($T = 11$) across two model families and six parameter scales, where PULSAR uses $\mathcal{A} =$ *FedAvg* and $\mathcal{U} =$ *GradAscent*.

| Split | Method | Metric | Model | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Qwen-1.5B | Qwen-3B | Qwen-7B | Llama-1B | Llama-3B | Llama-8B |
| Split90 | Finetune | MU | 0.507 | 0.594 | 0.627 | 0.549 | 0.628 | 0.678 |
| | | FTR | 0.584 | 0.537 | 0.563 | 0.578 | 0.566 | 0.508 |
| | GradAscent | MU | 0.467 | 0.541 | 0.562 | 0.503 | 0.581 | 0.603 |
| | | FTR | 0.652 | 0.641 | 0.636 | 0.649 | 0.634 | 0.612 |
| | SimNPO | MU | 0.458 | 0.522 | 0.557 | 0.517 | 0.59 | 0.592 |
| | | FTR | 0.689 | 0.679 | 0.676 | 0.659 | 0.665 | 0.658 |
| | Retrain | MU | 0.506 | 0.59 | 0.631 | 0.541 | 0.602 | 0.648 |
| | | FTR | 0.685 | 0.672 | 0.686 | 0.681 | 0.696 | 0.686 |
| Split95 | Finetune | MU | 0.514 | 0.585 | 0.625 | 0.545 | 0.618 | 0.639 |
| | | FTR | 0.594 | 0.582 | 0.573 | 0.576 | 0.577 | 0.559 |
| | GradAscent | MU | 0.478 | 0.539 | 0.576 | 0.512 | 0.586 | 0.612 |
| | | FTR | 0.674 | 0.675 | 0.666 | 0.665 | 0.645 | 0.626 |
| | SimNPO | MU | 0.48 | 0.537 | 0.578 | 0.519 | 0.589 | 0.604 |
| | | FTR | 0.658 | 0.651 | 0.64 | 0.648 | 0.628 | 0.615 |
| | Retrain | MU | 0.506 | 0.582 | 0.634 | 0.544 | 0.598 | 0.651 |
| | | FTR | 0.636 | 0.63 | 0.624 | 0.625 | 0.646 | 0.621 |
| Split99 | Finetune | MU | 0.518 | 0.588 | 0.629 | 0.553 | 0.623 | 0.65 |
| | | FTR | 0.602 | 0.564 | 0.575 | 0.556 | 0.585 | 0.55 |
| | GradAscent | MU | 0.48 | 0.54 | 0.577 | 0.503 | 0.585 | 0.581 |
| | | FTR | 0.659 | 0.646 | 0.641 | 0.639 | 0.642 | 0.616 |
| | SimNPO | MU | 0.483 | 0.54 | 0.577 | 0.502 | 0.584 | 0.581 |
| | | FTR | 0.655 | 0.647 | 0.633 | 0.639 | 0.643 | 0.611 |
| | Retrain | MU | 0.535 | 0.587 | 0.595 | 0.549 | 0.611 | 0.659 |
| | | FTR | 0.6 | 0.579 | 0.589 | 0.588 | 0.599 | 0.559 |

forgetting may sometimes come with a mild reduction in retained-knowledge utility. For external generalization, RAR and WFR are generally preserved and often improved by different unlearning operators, although the best operator depends on the aggregation method and metric.

**Qwen2.5-1.5B.** Tables 7 and 8 show a similar pattern on Qwen2.5-1.5B, further supporting the effectiveness of PULSAR across backbone architectures. For MU/FTR, operator-based methods consistently improve FTR over *Finetune*, with a moderate reduction in MU, reflecting the expected forgetting–utility trade-off. Among them, *GradAscent* is the strongest and most stable FTR-oriented operator, achieving the best FTR under most aggregation strategies. For example, under *FedAvg*, it improves FTR from $0.584$ to $0.692$, surpassing *Retrain* at $0.685$; under *FedAdagrad*, it reaches $0.726$, also higher than *Retrain* at $0.685$. *GradDiff* and *PDU* are also competitive, frequently obtaining second-best FTR scores, such as $0.683$ under *FedAvg* and $0.697$ under *FedAW* for *GradDiff*.

The ROUGE-based results provide a complementary view. Since lower FOR indicates stronger forgetting, *CEU* achieves the lowest FOR under most aggregation operators, including *FedAvg*, *FedProx*, *FedNova*, *FedAdagrad*, *FedAvgM*, *FedAW*, *FedAWA*, and *FedLWS*; meanwhile, *GradDiff* performs best under *FedAdam* and *FedYogi*. RER varies across operators, suggesting that stronger forgetting can be accompanied by different degrees of retained-knowledge utility change. For external generalization, RAR and WFR are generally maintained and often improved, although the best-performing operator depends on the aggregation method and metric.

Overall, the four tables demonstrate that PULSAR enables consistent and effective unlearning across Llama-3.2-1B and Qwen2.5-1.5B. *GradAscent* and *GradDiff* excel at improving FTR, *CEU* is most effective in reducing FOR, and *NPO*, *SimNPO*, *PDU*, *UNDIAL*, and *WGA* often better preserve utility or external generalization. Despite the inherent forgetting–utility trade-off, PULSAR remains robust across ten aggregation operators, substantially improving forgetting over *Finetune* and approaching the performance of costly *Retrain* with competitive generalization. These results confirm the effectiveness, robustness, and architectural adaptability of PULSAR in federated unlearning.

### F.5 LLAMA-3.2-3B WITH MU/FTR/ROUGE

Table 9 systematically compares different unlearning operators $\mathcal{U}$ under multiple FL aggregation strategies $\mathcal{A}$ in the **Split90** ($T = 11$) setting, and characterizes the trade-off among utility, forgetting, and knowledge preservation using MU/FTR and ROUGE-based metrics. From MU/FTR, we observe

Table 5: Utility and unlearning performance for **Split90** on **Llama-1B**. We compare 8 unlearning operators $\mathcal{U}$ and 10 aggregation operators $\mathcal{A}$ under MU/FTR, with *Finetune* and *Retrain* as reference baselines. Both metrics are higher-is-better (↑). **Bold**/underline indicates the best/second-best scores, respectively.

| **Llama-1B-Instruct with LoRA ($T$=11)** MU = Model Utility, FTR = Forget Truth Ratio | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **FedAvg** | | **FedAdam** | | **FedProx** | | **FedNova** | | **FedAdagrad** | |
| **Unlearning** | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ |
| Finetune | 0.549 | 0.578 | 0.551 | 0.573 | 0.551 | 0.577 | 0.545 | 0.599 | 0.486 | 0.685 |
| GradAscent | **0.515** | 0.671 | 0.501 | **0.682** | 0.507 | 0.675 | 0.511 | 0.687 | 0.450 | **0.718** |
| GradDiff | 0.514 | 0.667 | 0.510 | 0.670 | 0.514 | 0.669 | **0.518** | **0.688** | **0.474** | 0.713 |
| NPO | 0.512 | 0.648 | 0.509 | 0.652 | 0.505 | 0.651 | 0.512 | 0.669 | 0.473 | 0.710 |
| SimNPO | 0.509 | 0.633 | 0.508 | 0.632 | **0.516** | 0.626 | 0.507 | 0.644 | 0.471 | 0.684 |
| CEU | 0.495 | **0.682** | 0.492 | **0.682** | 0.504 | **0.679** | 0.513 | 0.666 | 0.470 | 0.717 |
| PDU | 0.511 | 0.662 | 0.496 | 0.679 | 0.515 | 0.647 | 0.510 | 0.666 | 0.468 | 0.712 |
| UNDIAL | 0.506 | 0.656 | **0.513** | 0.642 | 0.513 | 0.644 | 0.502 | 0.670 | 0.472 | 0.704 |
| WGA | 0.509 | 0.662 | 0.509 | 0.652 | 0.510 | 0.650 | 0.508 | 0.669 | 0.471 | 0.705 |
| Retrain | 0.541 | 0.681 | 0.455 | 0.724 | 0.532 | 0.682 | 0.541 | 0.684 | 0.488 | 0.709 |
| | **FedAvgM** | | **FedYogi** | | **FedAW** | | **FedAWA** | | **FedLWS** | |
| **Unlearning** | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ |
| Finetune | 0.551 | 0.573 | 0.551 | 0.573 | 0.509 | 0.631 | 0.544 | 0.604 | 0.538 | 0.602 |
| GradAscent | 0.501 | 0.682 | 0.501 | 0.682 | 0.489 | 0.668 | **0.517** | **0.689** | **0.512** | 0.682 |
| GradDiff | 0.510 | 0.670 | 0.510 | 0.670 | **0.493** | **0.693** | **0.517** | 0.685 | 0.510 | 0.677 |
| NPO | 0.509 | 0.652 | 0.509 | 0.652 | 0.492 | 0.667 | 0.515 | 0.666 | 0.509 | 0.660 |
| SimNPO | 0.508 | 0.632 | 0.508 | 0.632 | 0.490 | 0.630 | 0.513 | 0.641 | 0.510 | 0.640 |
| CEU | 0.494 | **0.687** | 0.494 | **0.687** | 0.488 | 0.660 | 0.515 | 0.668 | 0.498 | **0.685** |
| PDU | 0.501 | 0.674 | 0.501 | 0.674 | 0.483 | 0.663 | 0.514 | 0.663 | 0.502 | 0.673 |
| UNDIAL | **0.514** | 0.643 | **0.514** | 0.643 | 0.492 | 0.650 | 0.512 | 0.662 | **0.512** | 0.652 |
| WGA | 0.508 | 0.654 | 0.508 | 0.654 | 0.491 | 0.657 | 0.513 | 0.664 | 0.508 | 0.662 |
| Retrain | 0.555 | 0.631 | 0.462 | 0.713 | 0.537 | 0.682 | 0.539 | 0.682 | 0.527 | 0.688 |

a clear utility–forgetting tension: *Finetune* achieves the highest average MU (0.583) but the weakest FTR (0.620), while operator-based unlearning methods substantially improve FTR with moderate MU degradation. In particular, *GradAscent* and *GradDiff* achieve the strongest average FTR (0.707), surpassing *Retrain* (0.693), while maintaining competitive MU (0.554), suggesting that PULSAR can deliver strong forgetting without retraining from scratch.

The ROUGE results further confirm this trade-off. For FOR/RER, *Retrain* obtains the lowest average FOR (0.457), while operator-based methods also reduce FOR compared with *Finetune* (0.520), especially *GradDiff* (0.497) and *GradAscent* (0.498). This improvement in forgetting is accompanied by lower RER, indicating the cost of suppressing target knowledge while preserving retained knowledge. For external generalization, RAR and WFR are largely preserved, and in some cases improved: *SimNPO* achieves the best average RAR (0.833) and WFR (0.908), while *NPO* also obtains strong WFR (0.906). Overall, operator-based unlearning provides a balanced trade-off among forgetting effectiveness, utility preservation, and general knowledge retention; by contrast, *Retrain* remains a strong reference for FOR but does not uniformly dominate across all metrics.

### F.6 Source-to-Target Unlearning Analysis

Figures 7 ($\mathcal{U}$ = *SimNPO* and $\mathcal{A}$ = *FedProx*) and 6 ($\mathcal{U}$ = *GradAscent* and $\mathcal{A}$ = *FedAvg*) present an end-to-end comparison of PULSAR against the corresponding baselines across diverse backbones and model scales, covering three client roles ($\mathcal{C}^{\mathcal{S}}$, $\mathcal{C}^{\mathcal{T}}$, and $\mathcal{C}^{\mathcal{A}}$), as a supplement to 4.3. The dashed vertical line at round 5 separates the *unlearning* phase (rounds 1–5) from the *recovery* phase (rounds 6–10).

We plot ΔMU = MU(PULSAR)-MU(*Finetune*) in Figures 7 and 6. Across all six backbones, ΔMU stays near zero during unlearning and becomes increasingly positive after round 5, indicating that PULSAR achieves utility comparable to *Finetune* and often exceeds it, with more pronounced gains in the *recovery* phase. The target client $\mathcal{C}^{\mathcal{T}}$ shows the clearest post-recovery improvement after the filter $\mathcal{F}$ is removed, suggesting that PULSAR quickly compensates for the temporary utility drop induced by unlearning constraints and can even surpass the *Finetune* baseline. Meanwhile, $\mathcal{C}^{\mathcal{S}}$ and

Table 6: Unlearning effectiveness and external generalization for **Split90** on **Llama-1B**. Higher values of RER (↑), WFR (↑), and RAR (↑) indicate stronger generalization ability, whereas lower FOR (↓) suggests better forgetting performance. **Bold**/underline mark the best/second-best FOR/RER or RAR/WFR for each aggregation operator.

**Llama-1B-Instruct with LoRA ($T$=11)**
**FOR = Forget QA ROUGE, RER = Retain QA ROUGE**

| | FedAvg | | FedAdam | | FedProx | | FedNova | | FedAdagrad | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Unlearning** | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ |
| Finetune | 0.495 | 0.496 | 0.496 | 0.500 | 0.490 | 0.490 | 0.504 | 0.503 | 0.446 | 0.426 |
| GradAscent | 0.460 | 0.466 | 0.462 | 0.467 | 0.462 | 0.470 | 0.471 | 0.474 | 0.445 | 0.439 |
| GradDiff | 0.460 | 0.466 | 0.458 | 0.466 | **0.455** | 0.466 | **0.468** | 0.476 | **0.438** | 0.441 |
| NPO | 0.477 | **0.473** | 0.477 | 0.469 | 0.473 | 0.459 | 0.485 | 0.478 | 0.442 | 0.432 |
| SimNPO | 0.487 | 0.469 | 0.486 | 0.467 | 0.484 | 0.471 | 0.494 | 0.478 | 0.463 | 0.439 |
| CEU | **0.451** | 0.453 | 0.465 | 0.469 | 0.457 | 0.461 | 0.486 | 0.478 | 0.442 | 0.436 |
| PDU | 0.464 | 0.459 | **0.457** | 0.450 | 0.484 | 0.469 | 0.482 | 0.469 | 0.475 | **0.460** |
| UNDIAL | 0.484 | 0.469 | 0.484 | 0.470 | 0.482 | **0.472** | 0.483 | 0.469 | 0.451 | 0.437 |
| WGA | 0.476 | 0.469 | 0.480 | **0.472** | 0.469 | 0.460 | 0.485 | **0.479** | 0.448 | 0.437 |
| Retrain | 0.441 | 0.486 | 0.429 | 0.428 | 0.439 | 0.471 | 0.438 | 0.484 | 0.424 | 0.430 |

| | FedAvgM | | FedYogi | | FedAW | | FedAWA | | FedLWS | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Unlearning** | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ |
| Finetune | 0.496 | 0.500 | 0.496 | 0.500 | 0.495 | 0.466 | 0.500 | 0.494 | 0.505 | 0.486 |
| GradAscent | 0.462 | 0.467 | 0.462 | 0.467 | 0.479 | 0.455 | 0.478 | **0.484** | 0.466 | 0.471 |
| GradDiff | 0.458 | 0.466 | 0.458 | 0.466 | **0.452** | **0.460** | 0.470 | 0.478 | 0.462 | 0.467 |
| NPO | 0.477 | 0.469 | 0.477 | 0.469 | 0.472 | 0.456 | 0.481 | 0.473 | 0.483 | 0.472 |
| SimNPO | 0.486 | 0.467 | 0.486 | 0.467 | 0.490 | 0.449 | 0.494 | 0.475 | 0.496 | 0.472 |
| CEU | **0.456** | 0.456 | **0.456** | 0.456 | 0.474 | 0.456 | **0.467** | 0.476 | **0.451** | 0.461 |
| PDU | 0.466 | 0.463 | 0.466 | 0.463 | 0.474 | 0.453 | 0.470 | 0.468 | 0.477 | 0.474 |
| UNDIAL | 0.484 | 0.469 | 0.484 | 0.469 | 0.480 | 0.452 | 0.492 | 0.473 | 0.490 | **0.476** |
| WGA | 0.480 | **0.474** | 0.480 | **0.474** | 0.478 | 0.456 | 0.482 | 0.476 | 0.481 | 0.471 |
| Retrain | 0.450 | 0.636 | 0.471 | 0.475 | 0.437 | 0.476 | 0.440 | 0.489 | 0.437 | 0.481 |

**RAR = RA QA ROUGE, WFR = WF QA ROUGE**

| | FedAvg | | FedAdam | | FedProx | | FedNova | | FedAdagrad | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Unlearning** | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ |
| Finetune | 0.717 | 0.842 | 0.723 | 0.826 | 0.741 | 0.826 | 0.695 | 0.832 | 0.796 | 0.882 |
| GradAscent | 0.815 | 0.836 | 0.770 | 0.840 | 0.816 | 0.839 | 0.795 | 0.825 | **0.855** | 0.825 |
| GradDiff | **0.834** | 0.844 | 0.783 | 0.829 | **0.829** | 0.839 | 0.809 | **0.842** | 0.814 | 0.859 |
| NPO | 0.822 | 0.839 | **0.792** | 0.822 | 0.793 | 0.836 | 0.801 | 0.833 | 0.818 | 0.868 |
| SimNPO | 0.784 | 0.843 | 0.782 | **0.868** | 0.800 | **0.868** | **0.819** | 0.836 | 0.799 | 0.859 |
| CEU | 0.800 | 0.839 | 0.772 | 0.846 | 0.792 | 0.865 | 0.810 | 0.833 | 0.834 | 0.869 |
| PDU | 0.817 | **0.856** | 0.780 | 0.838 | 0.821 | 0.839 | 0.792 | 0.839 | 0.819 | **0.870** |
| UNDIAL | 0.794 | 0.852 | **0.792** | 0.848 | 0.798 | 0.839 | 0.797 | 0.835 | 0.792 | 0.865 |
| WGA | 0.781 | 0.848 | 0.768 | 0.845 | 0.798 | 0.834 | 0.811 | 0.826 | 0.811 | 0.856 |
| Retrain | 0.704 | 0.836 | 0.840 | 0.869 | 0.731 | 0.823 | 0.693 | 0.873 | 0.830 | 0.846 |

| | FedAvgM | | FedYogi | | FedAW | | FedAWA | | FedLWS | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Unlearning** | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ |
| Finetune | 0.723 | 0.826 | 0.723 | 0.826 | 0.774 | 0.858 | 0.698 | 0.809 | 0.696 | 0.818 |
| GradAscent | 0.770 | 0.840 | 0.770 | 0.840 | **0.793** | 0.846 | 0.777 | **0.875** | **0.808** | **0.844** |
| GradDiff | 0.783 | 0.829 | 0.783 | 0.829 | 0.769 | 0.852 | 0.792 | 0.831 | 0.782 | 0.817 |
| NPO | 0.792 | 0.822 | 0.792 | 0.822 | 0.754 | **0.860** | 0.767 | 0.819 | 0.770 | 0.843 |
| SimNPO | 0.782 | **0.868** | 0.782 | **0.868** | 0.724 | **0.860** | **0.802** | 0.840 | 0.798 | 0.830 |
| CEU | 0.787 | 0.831 | 0.787 | 0.831 | 0.773 | **0.860** | 0.794 | 0.846 | 0.807 | 0.825 |
| PDU | **0.812** | 0.848 | **0.812** | 0.848 | 0.779 | 0.843 | 0.787 | 0.849 | 0.757 | 0.836 |
| UNDIAL | 0.797 | 0.855 | 0.797 | 0.855 | 0.764 | **0.860** | 0.789 | 0.825 | 0.755 | 0.840 |
| WGA | 0.772 | 0.840 | 0.772 | 0.840 | 0.764 | **0.860** | 0.778 | 0.838 | 0.760 | 0.841 |
| Retrain | 0.458 | 0.697 | 0.787 | 0.838 | 0.706 | 0.840 | 0.681 | 0.813 | 0.719 | 0.828 |

$\mathcal{C}^{\mathcal{A}}$ remain centered around zero throughout, implying negligible utility impact on non-target clients and preserved generalization across both phases. This pattern is consistent across model scales, with larger models typically exhibiting slightly stronger positive $\Delta$MU in later rounds, suggesting that higher-capacity backbones benefit more from the recovery dynamics of PULSAR.

We also report $\Delta$FTR = FTR(PULSAR)-FTR(*Retrain*). The desirable behavior of FTR is phase-dependent: during *unlearning* (rounds 1–5), higher FTR implies stronger forgetting, whereas after the filter $\mathcal{F}$ is removed in *recovery* (rounds 6–10), lower FTR is preferred since the model should

Table 7: Utility and unlearning performance for **Split90** on **Qwen-1.5B**. We compare 8 unlearning operators $\mathcal{U}$ and 10 aggregation operators $\mathcal{A}$ under MU/FTR, with *Finetune* and *Retrain* as reference baselines. Both metrics are higher-is-better (↑). **Bold**/underline indicates the best/second-best scores, respectively.

| Qwen-1.5B-Instruct with LoRA ($T$=11) MU = Model Utility, FTR = Forget Truth Ratio | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **FedAvg** | | **FedAdam** | | **FedProx** | | **FedNova** | | **FedAdagrad** | |
| **Unlearning** | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ |
| Finetune | 0.507 | 0.584 | 0.422 | 0.701 | 0.495 | 0.615 | 0.512 | 0.594 | 0.462 | 0.694 |
| GradAscent | 0.459 | **0.692** | 0.415 | **0.722** | 0.450 | **0.695** | 0.468 | **0.690** | 0.436 | **0.726** |
| GradDiff | 0.462 | 0.683 | **0.433** | 0.714 | 0.462 | 0.693 | **0.474** | 0.689 | 0.444 | 0.719 |
| NPO | 0.466 | 0.665 | 0.427 | 0.710 | 0.459 | 0.677 | 0.473 | 0.678 | 0.436 | 0.717 |
| SimNPO | **0.467** | 0.652 | 0.429 | 0.694 | 0.464 | 0.664 | 0.468 | 0.654 | 0.446 | 0.688 |
| CEU | **0.467** | 0.677 | 0.427 | 0.715 | **0.465** | 0.690 | 0.462 | 0.683 | 0.432 | 0.720 |
| PDU | 0.461 | 0.683 | 0.412 | 0.717 | 0.457 | 0.692 | 0.473 | 0.682 | **0.448** | 0.722 |
| UNDIAL | **0.467** | 0.664 | 0.428 | 0.708 | 0.452 | 0.675 | 0.473 | 0.669 | 0.436 | 0.707 |
| WGA | 0.466 | 0.663 | 0.429 | 0.710 | 0.454 | 0.677 | 0.467 | 0.675 | 0.440 | 0.709 |
| Retrain | 0.506 | 0.685 | 0.421 | 0.720 | 0.496 | 0.690 | 0.506 | 0.687 | 0.506 | 0.685 |
| | **FedAvgM** | | **FedYogi** | | **FedAW** | | **FedAWA** | | **FedLWS** | |
| **Unlearning** | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ | MU ↑ | FTR ↑ |
| Finetune | 0.496 | 0.453 | 0.423 | 0.702 | 0.480 | 0.646 | 0.505 | 0.596 | 0.498 | 0.606 |
| GradAscent | 0.443 | 0.566 | 0.411 | **0.721** | 0.461 | **0.699** | 0.458 | **0.693** | **0.473** | **0.688** |
| GradDiff | 0.443 | 0.571 | **0.432** | 0.714 | **0.470** | 0.697 | 0.466 | 0.690 | 0.468 | 0.685 |
| NPO | 0.444 | 0.555 | 0.426 | 0.710 | 0.467 | 0.670 | 0.465 | 0.680 | 0.472 | 0.674 |
| SimNPO | **0.464** | 0.541 | 0.428 | 0.695 | 0.469 | 0.642 | 0.469 | 0.659 | 0.468 | 0.656 |
| CEU | 0.456 | 0.573 | 0.427 | 0.716 | 0.467 | 0.685 | 0.462 | 0.690 | 0.465 | 0.682 |
| PDU | 0.449 | **0.575** | 0.412 | 0.718 | 0.467 | 0.681 | 0.470 | 0.684 | 0.468 | 0.687 |
| UNDIAL | 0.455 | 0.552 | 0.426 | 0.708 | 0.463 | 0.662 | 0.464 | 0.669 | 0.471 | 0.662 |
| WGA | 0.458 | 0.556 | 0.429 | 0.710 | 0.465 | 0.665 | **0.471** | 0.671 | 0.464 | 0.671 |
| Retrain | 0.487 | 0.612 | 0.421 | 0.720 | 0.508 | 0.683 | 0.510 | 0.685 | 0.502 | 0.696 |

stop unlearning and return to a stable state. Consistent with this objective, the target client $\mathcal{C}^{\mathcal{T}}$ shows $\Delta$FTR close to zero (slightly negative) in rounds 1–5, indicating forgetting strength comparable to *Retrain* without the cost of training from scratch, followed by a marked decrease after round 5, suggesting rapid rollback of the induced forgetting once recovery starts. Meanwhile, $\mathcal{C}^{\mathcal{S}}$ and $\mathcal{C}^{\mathcal{A}}$ exhibit more negative $\Delta$FTR throughout, i.e., weaker forgetting than *Retrain* by roughly 10–15%, confirming that PULSAR avoids "forgetting for all" and concentrates the unlearning effect on $\mathcal{C}^{\mathcal{T}}$. These phase-aligned trends remain consistent across model scales and across choices of $\mathcal{U}$ and $\mathcal{A}$, demonstrating robust targeted unlearning in federated unlearning and timely suppression of forgetting during *recovery*.

Taken these two figures together, PULSAR not only preserves or improves model utility compared to *Finetune* (i.e., $\Delta$MU $\geq 0$, especially in the *recovery*), but also delivers forgetting performance competitive with *Retrain*, consistently across different operators ($\mathcal{U}$ and $\mathcal{A}$) and model scales.

### F.7 Component-Isolation Analysis

To further separate the effects of the source-derived filter $\mathcal{F}$ and JDOO from those of the underlying unlearning operator $\mathcal{U}$, we conduct four matched controls on Llama-3.2-1B with FedAWA. All conditions use the same experimental configuration and are averaged over six unlearning operators: CEU, NPO, PDU, SimNPO, UNDIAL, and WGA. *No-$\mathcal{F}$+JDOO* preserves the target-side JDOO objective but does not apply a source filter $\mathcal{F}$. *$\mathcal{F}$+SOO* applies the source-derived filter $\mathcal{F}$ without JDOO. *Random-$\mathcal{F}$+JDOO* replaces the source-derived filter $\mathcal{F}$ with a norm-matched random perturbation, and *PULSAR* uses both the source-derived filter $\mathcal{F}$ and JDOO. As Table 10 shows, the norm-matched random perturbation (*Random-$\mathcal{F}$+JDOO*) performs similarly to the no-filter control (*No-$\mathcal{F}$+JDOO*) in terms of FTR and FOR, whereas the source-derived filter $\mathcal{F}$ in *PULSAR* yields stronger target-side behavioral forgetting. This suggests that the observed forgetting effect is not explained by an arbitrary parameter perturbation of comparable magnitude. The comparison between *$\mathcal{F}$+SOO* and *PULSAR* further isolates the contribution of JDOO under the same source-derived filter $\mathcal{F}$.

Table 8: Unlearning effectiveness and external generalization for **Split90** on **Qwen-1.5B**. Higher values of RER (↑), WFR (↑), and RAR (↑) indicate stronger generalization ability, whereas lower FOR (↓) suggests better forgetting performance. **Bold**/underline mark the best/second-best FOR/RER or RAR/WFR for each aggregation operator.

**Qwen-1.5B-Instruct with LoRA ($T$=11)**
**FOR = Forget QA ROUGE, RER = Retain QA ROUGE**

| | FedAvg | | FedAdam | | FedProx | | FedNova | | FedAdagrad | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Unlearning** | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ |
| Finetune | 0.468 | 0.465 | 0.440 | 0.436 | 0.459 | 0.456 | 0.463 | 0.478 | 0.440 | 0.438 |
| GradAscent | 0.447 | **0.458** | 0.437 | 0.443 | 0.434 | 0.451 | 0.444 | 0.456 | 0.423 | **0.445** |
| GradDiff | 0.443 | 0.454 | **0.423** | 0.430 | 0.436 | 0.452 | 0.441 | 0.458 | 0.417 | 0.436 |
| NPO | 0.444 | 0.456 | 0.437 | 0.437 | 0.450 | **0.453** | 0.451 | 0.455 | 0.421 | 0.431 |
| SimNPO | 0.464 | 0.456 | 0.443 | 0.438 | 0.458 | 0.448 | 0.460 | 0.449 | 0.442 | 0.436 |
| CEU | **0.423** | 0.453 | 0.436 | **0.450** | **0.424** | 0.451 | **0.421** | 0.455 | **0.409** | 0.432 |
| PDU | 0.450 | 0.457 | 0.433 | 0.434 | 0.449 | 0.452 | 0.452 | **0.459** | 0.437 | 0.443 |
| UNDIAL | 0.455 | 0.454 | 0.436 | 0.432 | 0.455 | 0.449 | 0.454 | 0.457 | 0.428 | 0.427 |
| WGA | 0.450 | 0.457 | 0.439 | 0.434 | 0.452 | 0.452 | 0.454 | 0.456 | 0.422 | 0.428 |
| Retrain | 0.418 | 0.472 | 0.427 | 0.435 | 0.420 | 0.458 | 0.424 | 0.480 | 0.418 | 0.472 |

| | FedAvgM | | FedYogi | | FedAW | | FedAWA | | FedLWS | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Unlearning** | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ | FOR ↓ | RER ↑ |
| Finetune | 0.584 | 0.538 | 0.441 | 0.434 | 0.473 | 0.461 | 0.462 | 0.467 | 0.464 | 0.461 |
| GradAscent | 0.489 | 0.498 | 0.443 | 0.446 | 0.441 | **0.454** | 0.439 | 0.452 | 0.440 | **0.458** |
| GradDiff | 0.475 | 0.490 | **0.422** | 0.434 | 0.433 | 0.451 | 0.442 | 0.455 | 0.442 | 0.457 |
| NPO | 0.500 | 0.487 | 0.435 | 0.437 | 0.456 | 0.445 | 0.449 | 0.450 | 0.453 | 0.457 |
| SimNPO | 0.522 | 0.486 | 0.440 | 0.436 | 0.465 | 0.452 | 0.459 | 0.451 | 0.466 | 0.448 |
| CEU | **0.463** | 0.492 | 0.442 | **0.454** | **0.425** | **0.454** | **0.428** | 0.450 | **0.434** | 0.450 |
| PDU | 0.506 | **0.503** | 0.434 | 0.438 | 0.449 | 0.452 | 0.451 | 0.455 | 0.450 | **0.458** |
| UNDIAL | 0.520 | 0.490 | 0.436 | 0.429 | 0.455 | 0.447 | 0.455 | **0.458** | 0.457 | 0.457 |
| WGA | 0.511 | 0.486 | 0.437 | 0.434 | 0.457 | 0.445 | 0.452 | 0.456 | 0.451 | 0.451 |
| Retrain | 0.415 | 0.557 | 0.424 | 0.434 | 0.424 | 0.469 | 0.424 | 0.479 | 0.427 | 0.469 |

**RAR = RA QA ROUGE, WFR = WF QA ROUGE**

| | FedAvg | | FedAdam | | FedProx | | FedNova | | FedAdagrad | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Unlearning** | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ |
| Finetune | 0.667 | 0.830 | 0.671 | 0.798 | 0.696 | 0.819 | 0.629 | 0.804 | 0.780 | 0.836 |
| GradAscent | 0.684 | 0.846 | 0.692 | 0.795 | 0.647 | 0.855 | 0.664 | 0.857 | 0.775 | 0.862 |
| GradDiff | 0.697 | 0.836 | **0.736** | 0.811 | 0.705 | 0.850 | 0.659 | **0.876** | 0.748 | 0.849 |
| NPO | 0.657 | 0.870 | 0.642 | 0.833 | 0.692 | 0.835 | 0.672 | 0.840 | 0.750 | 0.849 |
| SimNPO | 0.672 | 0.864 | 0.701 | 0.785 | 0.677 | 0.848 | 0.644 | 0.808 | 0.759 | 0.853 |
| CEU | 0.686 | 0.858 | 0.678 | 0.829 | **0.740** | **0.866** | **0.723** | 0.850 | **0.789** | **0.879** |
| PDU | **0.699** | 0.862 | 0.657 | 0.802 | 0.716 | 0.807 | 0.692 | 0.843 | 0.739 | 0.875 |
| UNDIAL | 0.682 | 0.849 | 0.682 | **0.839** | 0.701 | 0.848 | 0.687 | 0.846 | 0.731 | 0.866 |
| WGA | 0.679 | **0.896** | 0.690 | 0.836 | 0.697 | 0.825 | 0.667 | 0.831 | 0.721 | 0.866 |
| Retrain | 0.654 | 0.791 | 0.689 | 0.816 | 0.692 | 0.798 | 0.650 | 0.811 | 0.654 | 0.791 |

| | FedAvgM | | FedYogi | | FedAW | | FedAWA | | FedLWS | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Unlearning** | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ |
| Finetune | 0.327 | 0.660 | 0.680 | 0.798 | 0.687 | 0.859 | 0.617 | 0.834 | 0.645 | 0.813 |
| GradAscent | 0.262 | 0.684 | 0.692 | 0.823 | 0.686 | 0.865 | 0.631 | 0.826 | 0.667 | 0.849 |
| GradDiff | 0.285 | 0.688 | **0.711** | 0.819 | 0.683 | 0.864 | **0.742** | 0.828 | 0.662 | 0.842 |
| NPO | 0.278 | 0.708 | 0.642 | 0.819 | 0.691 | 0.862 | 0.664 | 0.836 | **0.699** | 0.838 |
| SimNPO | **0.350** | 0.712 | 0.656 | 0.802 | 0.671 | 0.871 | 0.684 | 0.821 | 0.674 | **0.876** |
| CEU | 0.335 | 0.721 | 0.689 | 0.819 | 0.684 | 0.855 | 0.736 | 0.839 | 0.697 | 0.836 |
| PDU | 0.316 | 0.687 | 0.704 | 0.805 | **0.704** | 0.861 | 0.681 | 0.815 | **0.699** | 0.824 |
| UNDIAL | 0.307 | **0.724** | 0.667 | 0.828 | 0.644 | 0.861 | 0.671 | **0.859** | 0.686 | 0.869 |
| WGA | 0.327 | 0.704 | 0.675 | **0.836** | 0.643 | **0.872** | 0.663 | 0.823 | 0.674 | 0.823 |
| Retrain | 0.285 | 0.649 | 0.679 | 0.816 | 0.689 | 0.802 | 0.691 | 0.828 | 0.670 | 0.806 |

Moreover, to examine whether the JDOO anchor may reintroduce forgotten information, Table 11 reports its incremental effect under the same source-derived filter. Across all six unlearning operators, JDOO consistently improves FTR and FOR with only a modest MU trade-off, providing no empirical evidence that the previous-target anchor weakens filter-induced behavioral forgetting in the evaluated TOFU settings.

Table 9: Model utility and unlearning effectiveness on **Split90** for **Llama-3B** with $T = 11$, evaluated by MU/FTR and ROUGE. ↑ indicates higher-is-better, while ↓ lower-is-better. **Bold**/underline mark the best/second-best values in each column, excluding Mean and Std.

| MU = Model Utility, FTR = Forget Truth Ratio | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **FL Method** | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ |
| FedAvg | 0.628 | 0.566 | **0.590** | 0.665 | 0.584 | 0.666 | **0.583** | 0.650 | **0.581** | 0.634 | 0.602 | 0.696 |
| FedAvgM | **0.630** | 0.502 | 0.589 | **0.761** | **0.589** | **0.761** | 0.573 | 0.560 | 0.579 | 0.553 | **0.621** | 0.656 |
| FedAdam | 0.543 | 0.682 | 0.528 | 0.706 | 0.528 | 0.706 | 0.533 | 0.690 | 0.540 | 0.681 | 0.538 | **0.704** |
| FedAdagrad | 0.559 | 0.679 | 0.529 | 0.712 | 0.529 | 0.712 | 0.535 | **0.701** | 0.539 | **0.682** | 0.554 | **0.704** |
| FedYogi | 0.539 | **0.683** | 0.521 | 0.702 | 0.525 | 0.704 | 0.538 | 0.694 | 0.537 | 0.681 | 0.536 | **0.704** |
| FedProx | 0.598 | 0.610 | 0.566 | 0.694 | 0.571 | 0.692 | 0.573 | 0.665 | 0.571 | 0.658 | 0.595 | 0.692 |
| Mean | 0.583 | 0.620 | 0.554 | 0.707 | 0.554 | 0.707 | 0.556 | 0.66 | 0.558 | 0.648 | 0.574 | 0.693 |
| (Std.) | (0.041) | (0.075) | (0.032) | (0.031) | (0.03) | (0.031) | (0.023) | (0.053) | (0.021) | (0.05) | (0.036) | (0.019) |
| FOR = Forget QA ROUGE, RER = Retain QA ROUGE | | | | | | | | | | | | |
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **FL Method** | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ |
| FedAvg | 0.511 | 0.513 | 0.489 | 0.495 | 0.477 | 0.487 | 0.478 | 0.467 | 0.502 | 0.491 | 0.451 | 0.507 |
| FedAvgM | 0.670 | **0.648** | 0.611 | **0.594** | 0.611 | **0.594** | 0.594 | **0.572** | 0.612 | **0.582** | 0.452 | **0.654** |
| FedAdam | 0.497 | 0.489 | 0.498 | 0.490 | 0.498 | 0.490 | 0.496 | 0.492 | 0.505 | 0.495 | 0.478 | 0.490 |
| FedAdagrad | **0.448** | 0.455 | **0.439** | 0.440 | **0.439** | 0.440 | **0.444** | 0.446 | **0.464** | 0.454 | **0.430** | 0.446 |
| FedYogi | 0.496 | 0.491 | 0.498 | 0.489 | 0.493 | 0.488 | 0.493 | 0.492 | 0.507 | 0.490 | 0.479 | 0.489 |
| FedProx | 0.495 | 0.499 | 0.455 | 0.466 | 0.465 | 0.471 | 0.482 | 0.478 | 0.485 | 0.468 | 0.451 | 0.504 |
| Mean | 0.520 | 0.516 | 0.498 | 0.496 | 0.497 | 0.495 | 0.498 | 0.491 | 0.513 | 0.497 | 0.457 | 0.515 |
| (Std.) | (0.077) | (0.068) | (0.060) | (0.052) | (0.060) | (0.052) | (0.051) | (0.043) | (0.051) | (0.045) | (0.019) | (0.072) |
| RAR = RA QA ROUGE, WFR = WF QA ROUGE | | | | | | | | | | | | |
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **FL Method** | RAR↑ | WFR↑ | RAR↑ | WFR↑ | RAR↑ | WFR↑ | RAR↑ | WFR↑ | RAR↑ | WFR↑ | RAR↑ | WFR↑ |
| FedAvg | 0.801 | 0.872 | 0.877 | 0.926 | 0.849 | 0.917 | 0.867 | 0.922 | 0.853 | **0.933** | 0.828 | 0.892 |
| FedAvgM | 0.557 | 0.857 | 0.460 | 0.840 | 0.460 | 0.840 | 0.544 | 0.856 | 0.572 | 0.865 | 0.529 | 0.850 |
| FedAdam | **0.914** | **0.909** | 0.888 | 0.900 | 0.888 | 0.900 | 0.884 | 0.917 | 0.877 | 0.917 | 0.884 | 0.900 |
| FedAdagrad | 0.913 | 0.907 | 0.871 | 0.889 | 0.871 | 0.889 | **0.901** | 0.907 | 0.898 | 0.916 | **0.902** | 0.886 |
| FedYogi | 0.894 | 0.900 | 0.881 | 0.909 | **0.921** | 0.917 | 0.884 | **0.926** | 0.881 | 0.909 | 0.884 | **0.905** |
| FedProx | 0.886 | 0.889 | **0.889** | **0.930** | 0.903 | **0.920** | 0.890 | 0.910 | **0.914** | 0.910 | 0.866 | 0.895 |
| Mean | 0.828 | 0.889 | 0.811 | 0.899 | 0.815 | 0.897 | 0.828 | 0.906 | 0.833 | 0.908 | 0.816 | 0.888 |
| (Std.) | (0.139) | (0.021) | (0.172) | (0.033) | (0.176) | (0.030) | (0.140) | (0.026) | (0.129) | (0.023) | (0.143) | (0.020) |

Table 10: Component-isolation analysis on Llama-3.2-1B with FedAWA, averaged over six unlearning operators $\mathcal{U}$ (CEU, NPO, PDU, SimNPO, UNDIAL, and WGA).

| Method | $\mathcal{F}$ | JDOO | FTR↑ | FOR↓ | MU↑ | RER↑ | RAR↑ | WFR↑ |
|---|---|---|---|---|---|---|---|---|
| No-$\mathcal{F}$+JDOO | × | ✓ | 0.6154 | 0.4737 | 0.5381 | 0.4864 | 0.7015 | 0.8191 |
| $\mathcal{F}$+SOO | ✓ | × | 0.5989 | 0.4822 | **0.5458** | **0.4959** | 0.6980 | 0.8134 |
| Random-$\mathcal{F}$+JDOO | ✓ | ✓ | 0.6145 | 0.4736 | 0.5383 | 0.4828 | 0.7098 | 0.8096 |
| PULSAR | ✓ | ✓ | **0.6243** | **0.4628** | 0.5340 | 0.4817 | **0.7106** | **0.8269** |

### F.8 Ablation Studies

Figures 7 ($\mathcal{U}$ = *SimNPO*, $\mathcal{A}$ = *FedProx*) and 6 ($\mathcal{U}$ = *GradAscent*, $\mathcal{A}$ = *FedAvg*) present ablation results on the target clients $\mathcal{C}^{\mathcal{T}}$ under two backbones (Llama-3.2-1B and Qwen2.5-1.5B). We compare two settings: $\mathcal{F}$ + SOO (enabling the knowledge filter $\mathcal{F}$ without JDOO) and PULSAR (enabling both $\mathcal{F}$ and JDOO). The dashed line at round 5 divides the process into the *unlearning* phase (rounds 1–5) and the *recovery* phase (rounds 6–10).

We report ΔMU = MU(PULSAR) - MU(*Finetune*). During the *unlearning* phase, $\mathcal{F}$ + SOO maintains ΔMU close to zero across both backbones and both configurations, indicating that applying the knowledge filter $\mathcal{F}$ alone, coupled with SOO, largely preserves utility comparable to the *Finetune* baseline. In contrast, PULSAR starts from a lower MU but exhibits a monotonic improvement over rounds 1–5, eventually matching or slightly exceeding $\mathcal{F}$ + SOO. After entering the *recovery* phase, PULSAR shows a pronounced increase in ΔMU within a single communication round and quickly

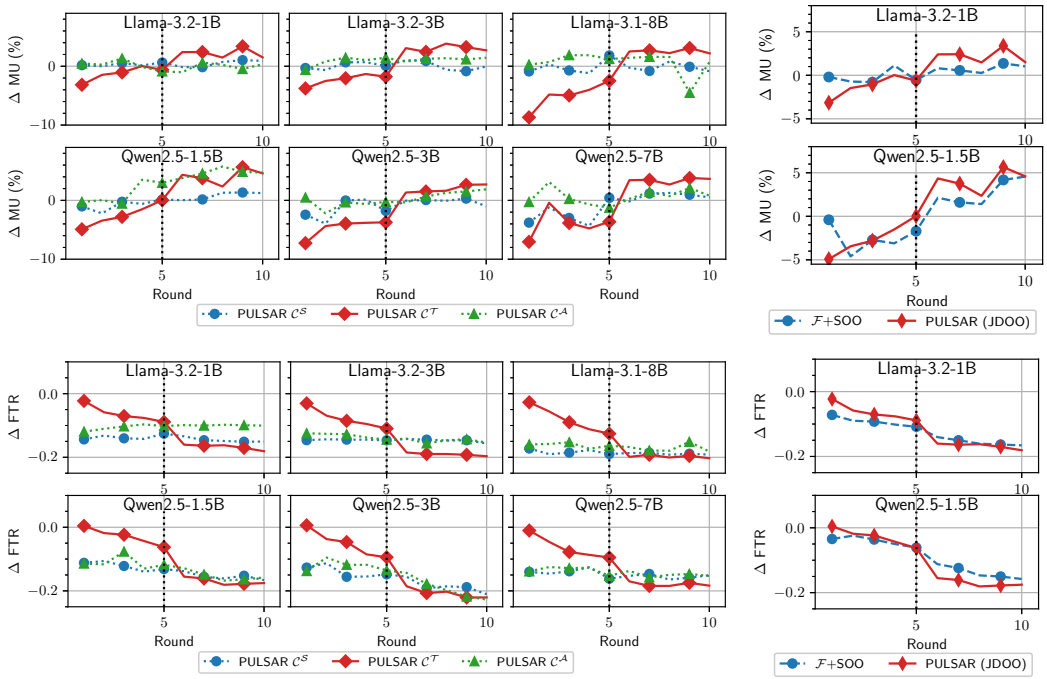



Figure 6: End-to-End Unlearning Analysis and Ablation Studies with $\mathcal{A} = \textit{FedAvg}$ and $\mathcal{U} = \textit{GradAscent}$, where $\Delta$MU = MU(PULSAR) − MU(*Finetune*) and $\Delta$FTR = FTR(PULSAR) − FTR(*Retrain*).

Table 11: Incremental effect of JDOO under the same source-derived filter. We report $\Delta M = M(\mathcal{F}\text{+JDOO}) - M(\mathcal{F}\text{+SOO})$.

| $\mathcal{U}$ | $\Delta$MU↑ | $\Delta$FTR↑ | $\Delta$FOR↓ |
|---|---|---|---|
| CEU | -0.0053 | +0.0203 | -0.0089 |
| NPO | -0.0173 | +0.0248 | -0.0185 |
| PDU | -0.0054 | +0.0212 | -0.0187 |
| SimNPO | -0.0108 | +0.0277 | -0.0333 |
| UNDIAL | -0.0172 | +0.0296 | -0.0169 |
| WGA | -0.0150 | +0.0283 | -0.0198 |
| Mean | -0.0118 | +0.0253 | -0.0194 |

stabilizes. The performance gap becomes clearer: PULSAR consistently achieves the highest and most stable $\Delta$MU during *recovery*, whereas $\mathcal{F}$ + SOO improves only marginally and remains close to the *Finetune* level. These results suggest that $\mathcal{F}$ alone is insufficient to sustain utility gains, while incorporating JDOO enables PULSAR to retain stronger post-recovery utility.

For $\Delta$FTR = FTR(PULSAR) - FTR(*Retrain*), the curves are overall negative and exhibit clear dynamics before and after the phase transition. During the *unlearning* phase, as unlearning proceeds, $|\Delta\text{FTR}|$ typically increases round by round, indicating that the forgetting gap between PULSAR and *Retrain* is progressively enlarged. This is because the forgetting advantage is gradually offset by utility improvement, reflecting the intrinsic trade-off between forgetting and utility. After entering the recovery phase, $\Delta$FTR generally continues to decrease and stabilizes at a lower range. At this stage, PULSAR often remains the most negative and most stable, since once the filter $\mathcal{F}$ is removed, ($\mathcal{C}^T$) can receive the global model without unlearning, and the forgetting effect will therefore drop substantially. These results suggest that PULSAR is advantageous not only in achieving faster and stronger forgetting by the end of unlearning, but also in improving utility and reducing the forgetting effect in a more stable and controllable manner after recovery.

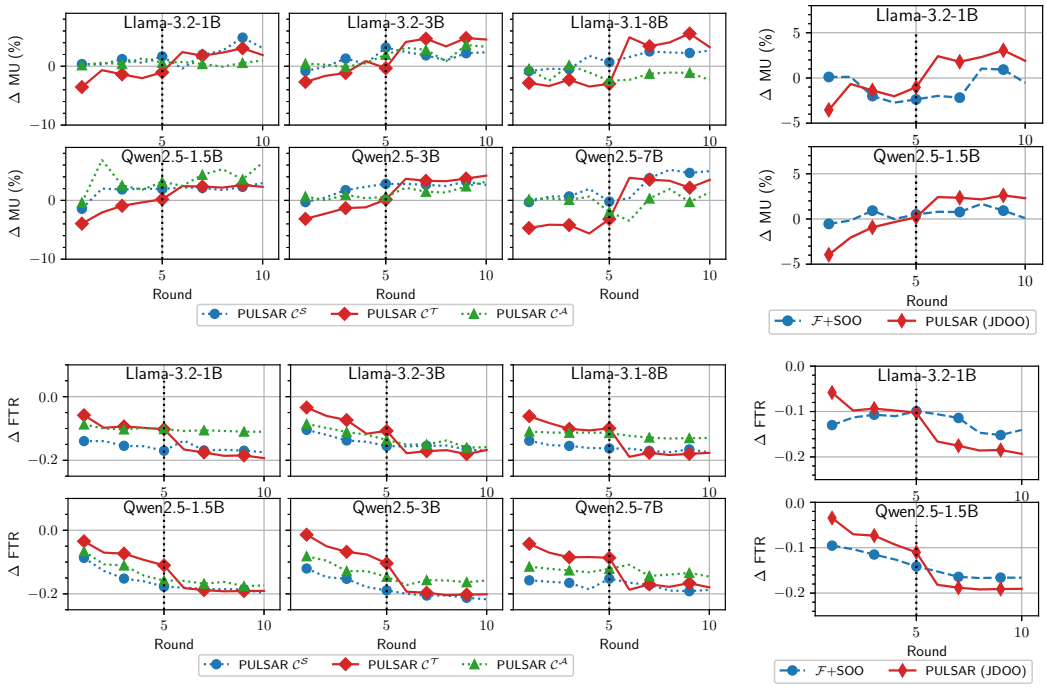



Figure 7: End-to-End Unlearning Analysis and Ablation Studies with $\mathcal{A} = \textit{FedProx}$ and $\mathcal{U} = \textit{SimNPO}$, where $\Delta$MU = MU(PULSAR) − MU(*Finetune*) and $\Delta$FTR = FTR(PULSAR) − FTR(*Retrain*).

### F.9 Hyperparameter Analysis

Figures 8 ($\mathcal{U} = \textit{GradAscent}$, $\mathcal{A} = \textit{FedAvg}$) and 9 ($\mathcal{U} = \textit{SimNPO}, \mathcal{A} = \textit{FedProx}$) evaluate the sensitivity to the number of filter-training rounds on two backbones (Llama-3.2-1B and Qwen2.5-1.5B), reporting MU and FTR for three client roles ($\mathcal{C}^{\mathcal{T}}, \mathcal{C}^{\mathcal{S}}, \mathcal{C}^{\mathcal{A}}$). For clarity, we additionally include the *Finetune* and *Retrain* baselines for MU and FTR, respectively. $\mathcal{C}^{\mathcal{T}}$ exhibits lower MU than $\mathcal{C}^{\mathcal{S}}$ and $\mathcal{C}^{\mathcal{A}}$, since it undergoes targeted unlearning, which inevitably introduces a utility drop. Meanwhile, FTR shows a clear role-wise separation: across all tested filter-training rounds, the target client $\mathcal{C}^{\mathcal{T}}$ consistently achieves substantially higher FTR than $\mathcal{C}^{\mathcal{S}}$ and $\mathcal{C}^{\mathcal{A}}$, indicating that the forgetting effect is well concentrated on the target client $\mathcal{C}^{\mathcal{T}}$ rather than diffused globally. Taken together, $\mathcal{C}^{\mathcal{T}}$ attains a large gain in forgetting strength with only a moderate utility loss, demonstrating a favorable unlearning–utility trade-off. Moreover, MU and FTR vary only marginally for all client roles across the sweep, suggesting that both metrics are largely insensitive to the number of filter-training rounds. The relative rank and value of MU/FTR remain stable, implying that PULSAR is robust to this hyperparameter choice and generalizes consistently across two backbones.

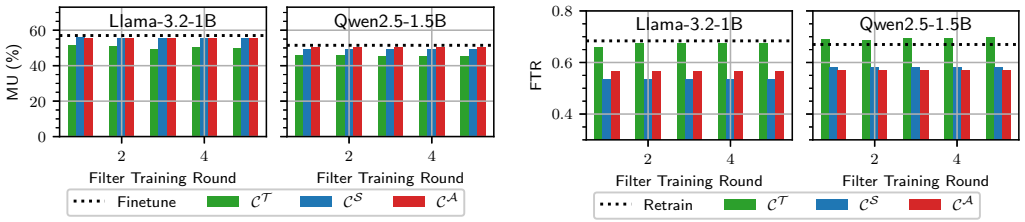



Figure 8: Hyperparameter Analysis. MU and FTR of three types of clients ($\mathcal{C}^{\mathcal{T}}, \mathcal{C}^{\mathcal{A}}$ and $\mathcal{C}^{\mathcal{S}}$) under PULSAR across different filter training rounds, with $\mathcal{U} = \textit{GradAscent}$ and $\mathcal{A} = \textit{FedAvg}$.

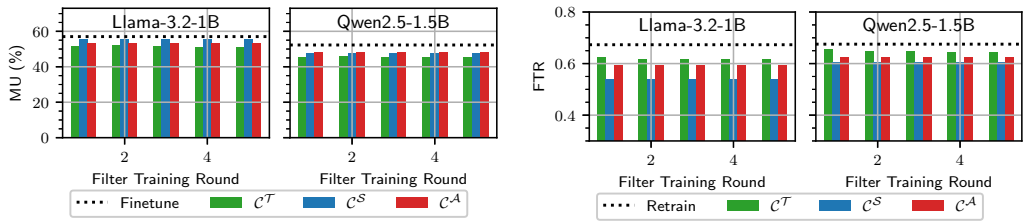



Figure 9: Hyperparameter Analysis. MU and FTR of three types of clients ($\mathcal{C}^{\mathcal{T}}, \mathcal{C}^{\mathcal{A}}$ and $\mathcal{C}^{\mathcal{S}}$) under PULSAR across different filter training rounds, with $\mathcal{U} =$ *SimNPO* and $\mathcal{A} =$ *FedProx*.

### F.10 Scalability of PULSAR to Multi-Source and Multi-Target FU

We further evaluate the scalability of PULSAR with 100 clients, where only a subset of clients participates in each round with sampling ratio $\lambda = 0.1$, and multi-source multi-target unlearning requests are issued. We report the average performance over all target clients $C^T$ at $T = 11$. As shown in Tables 12 and 13, the targeted-unlearning trend remains consistent under this larger-scale setting. On Llama-3.2-1B, operator-based PULSAR methods improve the average FTR from $0.681$ under *Finetune* to $0.7$–$0.709$, approaching *Retrain* at $0.713$, while maintaining comparable MU. On Qwen2.5-1.5B, the same pattern is observed: *GradAscent*, *GradDiff*, *NPO*, and *SimNPO* increase FTR from $0.680$ to $0.707$–$0.716$, closely matching *Retrain* at $0.716$.

The ROUGE-based results further confirm that PULSAR preserves retained and external knowledge under multiple source–target relations. For FOR/RER, operator-based PULSAR methods generally reduce FOR compared with *Finetune* while keeping RER at a similar level, indicating effective forgetting without severe retained-knowledge degradation. For RAR/WFR, all unlearning operators maintain competitive external generalization on both backbones; in particular, WFR is often improved over *Finetune*, suggesting that targeted unlearning does not collapse general knowledge. We also observe a few outlier entries, mainly under *Finetune*. This is likely because the 100-client setting substantially reduces the amount of data held by each client, making some local updates more prone to overfitting. However, such behavior is aggregation-dependent and does not appear consistently across most aggregation operators. Overall, these results show that PULSAR remains effective and stable in a more complex federated setting with multiple clients, partial participation, and multiple simultaneous unlearning relations.

### F.11 External-Domain Validation on MUSE-News

Our main experiments use the TOFU dataset because its explicit entity-level forget/retain split provides controlled ground truth for recipient-specific unlearning. To examine whether PULSAR's recipient-conditioned forgetting behavior is specific to TOFU's fictitious author profiles and structured QA format, we additionally evaluate it on MUSE-News, a real-news benchmark with official metrics for extraction, memorization, and membership inference.

We first report a representative GradAscent+FedProx configuration in Table 14. For the designated target recipients, PULSAR reduces extraction, knowledge memorization, and verbatim memorization relative to *Finetune*. MIA Min-K decreases from $0.8922$ to $0.5141$, close to the *Retrain* value of $0.5075$, while PrivLeak changes from $-78.44$ to $-2.82$, compared with $-1.50$ for *Retrain*. In contrast, the authorized recipient remains close to its full-data *Finetune* reference, with unchanged MIA Min-K and PrivLeak and comparable extraction and memorization metrics.

We further evaluate authorized-recipient preservation across all eight unlearning operators under both FedAvg and FedProx. Table 15 reports the absolute deviations from the corresponding *Finetune* references. These results provide complementary cross-domain evidence that PULSAR can concentrate the forgetting intervention on designated target recipients while preserving authorized recipients close to their full-data references. We clarify that this experiment validates recipient-specific localization across domains rather than providing certified privacy or attack-resistance guarantees.

Table 12: Results of multi-source and multi-target FU on **Split90** for **Llama-1B** with $T = 11$, evaluated by MU/FTR and ROUGE. ↑ indicates higher-is-better, while ↓ lower-is-better. **Bold**/underline mark the best/second-best MU/FTR/FOR/RER/WFR/RAR for each method across aggregations; blue/yellow highlight the best values under each unlearning operator.

| MU = Model Utility, FTR = Forget Truth Ratio | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **FL Method** | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ |
| FedAvg | 0.481 | 0.672 | 0.446 | 0.717 | 0.445 | 0.711 | 0.445 | 0.714 | 0.445 | 0.705 | 0.465 | 0.712 |
| FedAvgM | **0.511** | 0.567 | **0.489** | 0.625 | **0.489** | 0.614 | **0.489** | 0.623 | **0.487** | 0.610 | **0.496** | 0.662 |
| FedAdam | 0.319 | **0.727** | 0.409 | 0.730 | 0.407 | **0.730** | 0.407 | **0.732** | 0.409 | **0.727** | 0.406 | **0.730** |
| FedAdagrad | 0.441 | 0.720 | 0.423 | 0.730 | 0.424 | 0.728 | 0.423 | 0.729 | 0.424 | 0.723 | 0.437 | **0.730** |
| FedYogi | 0.317 | **0.727** | 0.408 | **0.731** | 0.407 | **0.730** | 0.406 | **0.732** | 0.409 | **0.727** | 0.406 | **0.730** |
| FedProx | 0.480 | 0.674 | 0.446 | 0.718 | 0.445 | 0.712 | 0.445 | 0.714 | 0.446 | 0.706 | 0.465 | 0.714 |
| Mean | 0.425 | 0.681 | 0.437 | 0.709 | 0.436 | 0.704 | 0.436 | 0.707 | 0.437 | 0.700 | 0.446 | 0.713 |
| (Std.) | (0.086) | (0.061) | (0.031) | (0.041) | (0.031) | (0.045) | (0.031) | (0.042) | (0.030) | (0.045) | (0.036) | (0.026) |
| FOR = Forget QA ROUGE, RER = Retain QA ROUGE | | | | | | | | | | | | |
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **FL Method** | FOR ↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ |
| FedAvg | 0.487 | 0.447 | 0.452 | 0.424 | 0.449 | 0.426 | 0.451 | 0.423 | 0.455 | 0.426 | 0.449 | 0.440 |
| FedAvgM | 0.561 | **0.512** | 0.520 | **0.492** | 0.526 | **0.491** | 0.522 | **0.490** | 0.527 | **0.491** | 0.479 | **0.497** |
| FedAdam | 0.467 | 0.431 | 0.479 | 0.467 | 0.481 | 0.464 | 0.480 | 0.463 | 0.488 | 0.464 | 0.479 | 0.467 |
| FedAdagrad | **0.447** | 0.422 | **0.434** | 0.411 | **0.436** | 0.410 | **0.434** | 0.408 | **0.441** | 0.410 | **0.434** | 0.415 |
| FedYogi | 0.464 | 0.435 | 0.480 | 0.465 | 0.481 | 0.463 | 0.479 | 0.463 | 0.485 | 0.462 | 0.479 | 0.468 |
| FedProx | 0.472 | 0.445 | 0.449 | 0.423 | 0.450 | 0.425 | 0.448 | 0.421 | 0.456 | 0.424 | 0.448 | 0.437 |
| Mean | 0.483 | 0.449 | 0.469 | 0.447 | 0.470 | 0.446 | 0.469 | 0.445 | 0.475 | 0.446 | 0.461 | 0.454 |
| (Std.) | (0.040) | (0.032) | (0.031) | (0.032) | (0.033) | (0.031) | (0.032) | (0.032) | (0.031) | (0.031) | (0.020) | (0.029) |
| WFR = WF QA ROUGE, RAR = RA QA ROUGE | | | | | | | | | | | | |
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **FL Method** | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ | RAR ↑ | WFR ↑ |
| FedAvg | 0.839 | 0.839 | 0.836 | 0.864 | 0.828 | 0.857 | 0.842 | 0.855 | 0.828 | 0.853 | 0.799 | 0.851 |
| FedAvgM | 0.614 | 0.813 | 0.603 | 0.810 | 0.607 | 0.809 | 0.608 | 0.813 | 0.601 | 0.799 | 0.577 | 0.819 |
| FedAdam | ~~0.229~~ | ~~0.586~~ | 0.841 | **0.872** | 0.827 | 0.851 | 0.829 | 0.855 | 0.831 | 0.849 | 0.812 | 0.841 |
| FedAdagrad | **0.926** | **0.878** | **0.859** | 0.869 | **0.855** | **0.876** | **0.854** | **0.879** | **0.856** | **0.880** | **0.875** | **0.871** |
| FedYogi | ~~0.220~~ | ~~0.584~~ | 0.850 | 0.868 | 0.826 | 0.853 | 0.827 | 0.853 | 0.835 | 0.847 | 0.809 | 0.835 |
| FedProx | 0.854 | 0.850 | 0.842 | 0.863 | 0.834 | 0.860 | 0.851 | 0.863 | 0.837 | 0.852 | 0.813 | 0.854 |
| Mean | 0.808 | 0.845 | 0.805 | 0.858 | 0.796 | 0.851 | 0.802 | 0.853 | 0.798 | 0.847 | 0.781 | 0.845 |
| (Std.) | (0.135) | (0.027) | (0.099) | (0.024) | (0.093) | (0.022) | (0.096) | (0.022) | (0.097) | (0.026) | (0.103) | (0.018) |

### F.12 Stress Test under Non-IID Conditions

To further evaluate PULSAR under heterogeneous client data, we conduct an additional author-level Non-IID stress test using Dirichlet partitions with $\alpha \in \{1.0, 0.5, 0.1\}$, where a smaller $\alpha$ induces a stronger concentration of author distributions across clients. Across all settings, we vary only the client data partition, while keeping all other configurations consistent with the main experiments.

We quantify the realized client-level heterogeneity using the mean author-distribution entropy and mean pairwise Jensen–Shannon (JS) divergence. As shown in Table 16, decreasing $\alpha$ from $1.0$ to $0.1$ increases the mean pairwise JS divergence from $0.249$ to $0.581$ and decreases the mean author entropy from $5.153$ to $4.614$, confirming progressively stronger distribution skew across clients.

Table 17 reports the corresponding role-wise forgetting and utility results. Across all three heterogeneity levels, target clients consistently exhibit stronger forgetting behavior than source and authorized clients, with higher FTR and lower FOR, while maintaining comparable MU. Importantly, this role-wise ordering remains stable as client-level heterogeneity increases.

To directly quantify recipient-specific separation, we compare target-client performance with the average performance of non-target (source and authorized) clients. Source and authorized clients are grouped because both receive the unfiltered dispatch under the evaluated source–target relation. We define the target–non-target FTR margin as the target FTR minus the mean FTR of non-target clients, and the FOR margin as the mean non-target FOR minus the target FOR. Larger positive margins therefore indicate stronger target-specific forgetting. As shown in Table 18, the FTR margin remains positive across all heterogeneity levels, ranging from $0.034$ to $0.051$, while the FOR margin ranges

Table 13: Results of multi-source and multi-target FU on **Split90** for **Qwen-1.5B** with $T = 11$, evaluated by MU/FTR and ROUGE. ↑ indicates higher-is-better, while ↓ lower-is-better. **Bold**/underline mark the best/second-best values in each column.

| MU = Model Utility, FTR = Forget Truth Ratio | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **FL Method** | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ | MU↑ | FTR↑ |
| FedAvg | **0.462** | 0.679 | 0.435 | 0.723 | 0.436 | 0.715 | 0.434 | 0.719 | 0.439 | 0.709 | **0.452** | 0.716 |
| FedAvgM | 0.455 | 0.566 | **0.440** | 0.637 | **0.438** | 0.626 | **0.438** | 0.633 | 0.437 | 0.622 | 0.451 | 0.660 |
| FedAdam | 0.395 | **0.740** | 0.413 | **0.738** | 0.410 | **0.738** | 0.410 | **0.737** | 0.409 | **0.736** | 0.412 | **0.738** |
| FedAdagrad | 0.426 | 0.726 | 0.425 | 0.734 | 0.418 | 0.733 | 0.420 | 0.734 | 0.419 | 0.729 | 0.424 | 0.733 |
| FedYogi | ~~0.292~~ | ~~0.740~~ | 0.393 | 0.737 | 0.396 | 0.736 | 0.392 | 0.735 | 0.395 | 0.734 | 0.393 | 0.734 |
| FedProx | 0.457 | 0.687 | 0.437 | 0.725 | 0.437 | 0.718 | 0.434 | 0.722 | **0.441** | 0.710 | 0.449 | 0.718 |
| Mean | 0.439 | 0.680 | 0.424 | 0.716 | 0.422 | 0.711 | 0.421 | 0.713 | 0.423 | 0.707 | 0.430 | 0.716 |
| (Std.) | (0.028) | (0.068) | (0.018) | (0.039) | (0.017) | (0.043) | (0.018) | (0.040) | (0.019) | (0.043) | (0.025) | (0.029) |
| FOR = Forget QA ROUGE, RER = Retain QA ROUGE | | | | | | | | | | | | |
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **FL Method** | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ | FOR↓ | RER↑ |
| FedAvg | 0.463 | 0.445 | 0.448 | 0.435 | 0.453 | 0.435 | 0.446 | 0.434 | 0.457 | 0.431 | **0.440** | 0.436 |
| FedAvgM | 0.502 | **0.468** | 0.473 | **0.453** | 0.478 | **0.450** | 0.470 | **0.452** | 0.485 | **0.451** | 0.447 | **0.455** |
| FedAdam | 0.446 | 0.413 | **0.442** | 0.426 | **0.435** | 0.418 | **0.437** | 0.419 | **0.438** | 0.416 | 0.443 | 0.418 |
| FedAdagrad | 0.454 | 0.434 | 0.446 | 0.431 | 0.442 | 0.425 | 0.442 | 0.424 | 0.445 | 0.424 | 0.445 | 0.431 |
| FedYogi | **0.402** | 0.372 | 0.446 | 0.429 | 0.441 | 0.424 | 0.441 | 0.423 | 0.441 | 0.419 | **0.440** | 0.423 |
| FedProx | 0.464 | 0.445 | 0.448 | 0.435 | 0.452 | 0.435 | 0.443 | 0.431 | 0.455 | 0.431 | 0.444 | 0.438 |
| Mean | 0.455 | 0.429 | 0.450 | 0.435 | 0.450 | 0.431 | 0.446 | 0.430 | 0.454 | 0.429 | 0.443 | 0.433 |
| (Std.) | (0.032) | (0.033) | (0.011) | (0.010) | (0.015) | (0.011) | (0.012) | (0.012) | (0.017) | (0.013) | (0.003) | (0.013) |
| RAR = RA QA ROUGE, WFR = WF QA ROUGE | | | | | | | | | | | | |
| | **Finetune** | | **GradAscent** | | **GradDiff** | | **NPO** | | **SimNPO** | | **Retrain** | |
| **FL Method** | RAR↑ | WFR↑ | RAR↑ | WFR↑ | RAR↑ | WFR↑ | RAR↑ | WFR↑ | RAR↑ | WFR↑ | RAR↑ | WFR↑ |
| FedAvg | 0.822 | 0.862 | 0.810 | 0.866 | 0.807 | 0.862 | 0.809 | 0.865 | 0.809 | 0.857 | 0.821 | 0.849 |
| FedAvgM | ~~0.435~~ | 0.736 | ~~0.464~~ | 0.751 | ~~0.447~~ | 0.763 | ~~0.452~~ | 0.759 | ~~0.439~~ | 0.767 | ~~0.420~~ | 0.769 |
| FedAdam | 0.861 | **0.896** | **0.879** | **0.892** | **0.866** | **0.887** | **0.869** | **0.884** | **0.876** | **0.888** | **0.884** | **0.886** |
| FedAdagrad | **0.873** | 0.866 | 0.853 | 0.880 | 0.843 | 0.867 | 0.849 | 0.879 | 0.849 | 0.868 | 0.858 | 0.878 |
| FedYogi | ~~0.151~~ | ~~0.641~~ | 0.840 | 0.872 | 0.837 | 0.861 | 0.840 | 0.858 | 0.830 | 0.863 | 0.839 | 0.863 |
| FedProx | 0.822 | 0.870 | 0.816 | 0.860 | 0.816 | 0.865 | 0.813 | 0.866 | 0.821 | 0.863 | 0.821 | 0.855 |
| Mean | 0.845 | 0.846 | 0.840 | 0.854 | 0.834 | 0.851 | 0.836 | 0.852 | 0.837 | 0.851 | 0.845 | 0.850 |
| (Std.) | (0.026) | (0.063) | (0.028) | (0.051) | (0.023) | (0.044) | (0.025) | (0.046) | (0.026) | (0.043) | (0.027) | (0.042) |

Table 14: External-domain evaluation on MUSE-News using GradAscent and FedProx. For MIA Min-K, values closer to 0.5 are preferable. For PrivLeak, values closer to 0 are preferable.

| Role / Reference | Extract↓ | Forget KnowMem↓ | Forget VerbMem↓ | MIA Min-K→ 0.5 | PrivLeak→ 0 |
|---|---|---|---|---|---|
| Finetune–Target | 0.027248 | 0.192162 | 0.190404 | 0.8922 | -78.44 |
| Retrain–Target | 0.024166 | 0.180039 | 0.184873 | 0.5075 | -1.50 |
| PULSAR–Target | **0.009667** | **0.002737** | **0.048913** | 0.5141 | -2.82 |
| Finetune–Authorized | 0.031953 | 0.190063 | 0.195330 | 0.8932 | -78.64 |
| PULSAR–Authorized | 0.028625 | 0.194898 | 0.191394 | 0.8932 | -78.64 |

from 0.069 to 0.087 and is largest under the most heterogeneous partition. Meanwhile, the maximum MU gap between the target and either non-target role remains below 0.03.

We further examine cross-partition variation across client roles. Table 19 reports the max–min variation in FTR, FOR, and MU over $\alpha \in 1.0, 0.5, 0.1$. Despite the more than twofold increase in mean pairwise JS divergence, target-client FTR, FOR, and MU vary by only 0.006, 0.016, and 0.011, respectively, with similarly limited variation for source and authorized clients.

Overall, increasing author-level distribution skew does not alter the central role-dependent behavior observed in PULSAR: designated target clients consistently exhibit stronger behavioral forgetting than source and authorized clients, while utility remains comparable across roles. These results provide complementary evidence that PULSAR's recipient-specific behavior is preserved under more heterogeneous client data distributions.

Table 15: Absolute deviation of PULSAR authorized recipients from their corresponding *Finetune* references on MUSE-News, aggregated over eight unlearning operators and two aggregation methods.

| Metric | Mean Absolute Deviation | Maximum Absolute Deviation |
|---|---|---|
| Extraction | 0.003322 | 0.003328 |
| Forget KnowMem | 0.002926 | 0.011047 |
| Forget VerbMem | 0.002808 | 0.004568 |
| MIA Min-K | 0.000081 | 0.000200 |
| PrivLeak | 0.016250 | 0.040000 |
| Retain KnowMem | 0.005778 | 0.012809 |

Table 16: Realized client-level data heterogeneity under different Dirichlet partitions. Lower author entropy and higher pairwise JS divergence indicate stronger distribution skew.

| $\alpha$ | Mean Author Entropy | Mean Pairwise JS Divergence |
|---|---|---|
| 1.0 | 5.153 | 0.249 |
| 0.5 | 5.009 | 0.349 |
| 0.1 | 4.614 | 0.581 |

## G Limitations and Broader Impacts

PULSAR has several limitations. First, its forgetting behavior depends on the underlying unlearning operator $\mathcal{U}$. Residual memorization, over-unlearning, or collateral utility degradation of $\mathcal{U}$ may therefore be inherited by PULSAR. Second, PULSAR assumes a trusted policy-enforcing server, protocol-compliant clients, and a correctly maintained source-to-target authorization map. Malicious servers or clients, collusion, unauthorized checkpoint access, filter theft, and authorization-map tampering fall outside the current threat model and require complementary security mechanisms. Third, although we extend the evaluation beyond TOFU with MUSE-News and additional Non-IID stress tests, it remains limited relative to the diversity of real-world federated deployments.

Table 17: Role-wise forgetting and utility under progressively stronger author-level Non-IID partitions.

| $\alpha$ | Role | FTR↑ | FOR↓ | MU↑ |
|---|---|---|---|---|
| 1.0 | Source | 0.511 | 0.619 | 0.536 |
| | Target | **0.589** | **0.497** | 0.535 |
| | Authorized | 0.565 | 0.512 | 0.542 |
| 0.5 | Source | 0.521 | 0.627 | 0.554 |
| | Target | **0.583** | **0.498** | 0.527 |
| | Authorized | 0.577 | 0.507 | 0.555 |
| 0.1 | Source | 0.518 | 0.630 | 0.544 |
| | Target | **0.587** | **0.482** | 0.538 |
| | Authorized | 0.569 | 0.507 | 0.547 |

Table 18: Recipient-selectivity margins under different Non-IID levels. Non-target refers to the mean of the source and authorized roles. The MU gap is the larger absolute difference between the target and either non-target role.

| $\alpha$ | ΔFTR↑ | ΔFOR↑ | Max. Target–Non-target MU Gap↓ |
|---|---|---|---|
| 1.0 | 0.051 | 0.069 | 0.007 |
| 0.5 | 0.034 | 0.069 | 0.028 |
| 0.1 | 0.044 | 0.087 | 0.009 |

Table 19: Cross-partition variation across $\alpha \in \{1.0, 0.5, 0.1\}$. Variation is calculated as the maximum minus the minimum value across the three partitions.

| Role | FTR Range | FTR Variation | FOR Range | FOR Variation | MU Range | MU Variation |
|---|---|---|---|---|---|---|
| Source | 0.511–0.521 | 0.010 | 0.619–0.630 | 0.011 | 0.536–0.554 | 0.018 |
| Target | 0.583–0.589 | 0.006 | 0.482–0.498 | 0.016 | 0.527–0.538 | 0.011 |
| Authorized | 0.565–0.577 | 0.012 | 0.507–0.512 | 0.005 | 0.542–0.555 | 0.013 |