FEDCODS: FEDERATED LLM UNLEARNING BY COUPLING KNOWLEDGE DISTILLATION WITH LIKELIHOOD SUPPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated large language model (LLM) unlearning aims to remove targeted training data without retraining the model from scratch or sharing clients' raw data. However, existing likelihood-based unlearning objectives face two key challenges in federated settings. When the forget set is held by a single client, the requesting client has access to only limited local retain data, making it difficult to preserve knowledge learned from other clients. When the forget set is distributed across multiple clients, heterogeneous local forget data can produce inconsistent forgetting directions that are weakened during federated aggregation. To address these challenges, we propose FedCoDS, a federated LLM unlearning method that couples token-wise likelihood suppression with knowledge distillation from a shared frozen pretrained model. The suppression term directly reduces the likelihood of targeted tokens, while knowledge distillation provides a common distributional reference that constrains model changes and promotes cross-client consistency. FedCoDS combines their complementary strengths to achieve a better balance between effective forgetting and knowledge retention. Experiments on TOFU and MUSE under both single- and multi-client unlearning settings show that FedCoDS consistently improves the forgetting–retention performance over SOTA baselines. Further analysis shows that FedCoDS improves cross-client forget-gradient alignment and produces forget-answer likelihoods closer to those of the retrained model.

1 INTRODUCTION

Large language models (LLMs) are often fine-tuned on data held by different users or organizations. Federated learning (FL) allows clients to train a shared model without transferring their raw data to a server (McMahan et al., 2017; Ye et al., 2024). However, training and communicating an LLM can be costly. Low-rank adaptation (LoRA) reduces number of trainable parameters and communication cost by updating small adapters while freezing the pretrained model (Hu et al., 2022). This makes LoRA an important method for federated LLM fine-tuning (Fan et al., 2023; Ye et al., 2024; Sun et al., 2024; Wang et al., 2024c).

However, users may request deletion of personal data, while models may also need to forget harmful or copyrighted content (Jang et al., 2023; Yao et al., 2024). Machine unlearning aims to remove the influence of specified training data, called the *forget set*, while preserving model utility on the remaining data, called the *retain set*, without repeating the full training process (Maini et al., 2024).

A common approach to centralized LLM unlearning is to reduce the likelihood of the forget data. Negative Preference Optimization (NPO) controls this suppression using a reference model (Zhang et al., 2024); SimNPO removes the reference-model dependence (Fan et al., 2025); and SATIMP refines the allocation of unlearning effort through token-level loss reweighting (Yang et al., 2025). Their forget objectives determine which recorded responses to suppress and how strongly, but do not specify a shared target distribution for the predictions that replace them.

Recent work has begun to address unlearning within federated LLM systems. Oblivionis provides a framework that formulates one client's deletion request at a time (Zhang et al., 2026). The Dual-Level Selective Unlearning framework (DSU) removes a client's influence by selecting sensitive LoRA modules, using historical training information (Xie et al., 2026). FULM addresses heterogeneous

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.

unlearning requests by learning separate unlearning and retention adapters (Zhong et al., 2025). These works primarily focus on designing federated unlearning frameworks that incorporate existing unlearning methods.

We study federated LLM unlearning under two forget-data allocations and find that likelihood-based objectives are insufficient. In the *single-client* setting, one client holds the entire forget set, similar to the setup in (Zhang et al., 2026). Because the requesting client has access to only its limited local retain data, likelihood-based unlearning can degrade knowledge learned from other clients during federated fine-tuning and reduce model utility. In the *multi-client* setting, the forget set is distributed across multiple clients with heterogeneous local data. Different clients therefore suppress different target tokens, which can produce inconsistent local forgetting directions and weaken unlearning after aggregation. To address these challenges, we propose FedCoDS, which couples token-wise SimNPO likelihood suppression with knowledge distillation from the frozen pretrained model. Token-wise SimNPO provides direct suppression of targeted forget tokens, while knowledge distillation constrains model changes through distillation from the frozen pretrained model. By coupling these complementary objectives, FedCoDS improves the forgetting–retention performance under both single- and multi-client unlearning.

Our contributions are:

- We propose FedCoDS, a token-level federated LLM unlearning objective that couples knowledge distillation with likelihood suppression. By combining these complementary strengths, FedCoDS achieves a better balance between effective forgetting and retention among SOTA. To the best of our knowledge, this is the first loss function designed specifically to address both retain and cross-client update consistency in federated LLM unlearning.
- We systematically study two forget-data allocations: *single-client* unlearning, where limited local retain data makes global knowledge preservation difficult, and *multi-client* unlearning, introducing disagreement among local forgetting updates.
- We evaluate FedCoDS against NPO, SimNPO, and SATIMP on TOFU (Maini et al., 2024) and MUSE (Shi et al., 2025) benchmarks. Empirical results show that FedCoDS consistently achieves superior forgetting efficacy while retaining utility under both allocation scenarios.

2 Related work

Federated LLM fine-tuning. Low-rank adaptation (LoRA) enables parameter-efficient LLM fine-tuning by training low-rank updates while keeping pretrained weights frozen (Hu et al., 2022). Its reduced computation and communication costs make it particularly suitable for federated LLM training. Recent systems such as FATE-LLM and OpenFedLLM adopt LoRA for federated fine-tuning (Fan et al., 2023; Ye et al., 2024), while FFA-LoRA and FLoRA further improve communication efficiency and performance of adaptor aggregation (Sun et al., 2024; Wang et al., 2024c). Following this line of work, we use LoRA with FedAvg as the underlying federated fine-tuning framework.

Machine and federated unlearning. Machine unlearning aims to remove the influence of specified training data while preserving utility on the remaining data (Cao & Yang, 2015; Bourtoule et al., 2020). Federated unlearning extends this objective to models trained across multiple clients. Existing approaches reconstruct models from stored updates (Liu et al., 2021), perform local unlearning followed by recovery training (Halimi et al., 2023), or use historical updates to do removal and knowledge distillation to approximate retraining (Wu et al., 2022). These federated unlearning methods have been studied on conventional classification models and relatively small text models.

LLM and federated LLM unlearning. A major class of LLM unlearning methods suppresses the likelihood of forget data. Gradient ascent directly decreases the likelihood of target responses (Jang et al., 2023; Yao et al., 2024); NPO controls the GA suppression using a reference model (Zhang et al., 2024); SimNPO removes the reference-model dependence from NPO (Fan et al., 2025); and SATIMP further reweights GA suppression use Saturation and Importance (Yang et al., 2025). Another line of work uses Knowledge distillation for unlearning: RKLD constructs an adjusted teacher distribution (Wang et al., 2024a), while UNDIAL uses self-distillation with modified logits (Dong et al., 2025). Recent studies have begun extending LLM unlearning to federated settings. Oblivionis integrates existing centralized unlearning objectives into a federated pipeline framework (Zhang et al., 2026); and FULM separately learns unlearning and retention adapters (Zhong et al., 2025). These studies

primarily adapt existing likelihood-based methods such as NPO and SimNPO, to federated settings. In contrast, we focus on the unlearning loss function itself and design FedCoDS to address the challenges arising in federated LLM unlearning: preserving globally learned knowledge when a requester has only local retain data, and maintaining consistent forgetting directions when forget data are distributed across clients.

3 PRELIMINARIES

3.1 FEDERATED LORA FINE-TUNING

LoRA is a widely used parameter-efficient fine-tuning (PEFT) method in federated learning because it reduces both communication and computation costs (Chen et al., 2025). Consider K clients with local datasets D_k and shared pretrained weights W_0. For each selected linear layer, LoRA introduces two trainable low-rank matrices:

$$W_\theta = W_0 + BA, \tag{1}$$

where $A \in \mathbb{R}^{r\times d_{\text{in}}}$, $B \in \mathbb{R}^{d_{\text{out}}\times r}$, $r \ll \min(d_{\text{in}}, d_{\text{out}})$. W_0 remains static during the fine-tuning process, and A and B contain trainable parameters (Hu et al., 2022).

For an input x and target sequence $y = (y_1, \ldots, y_m)$, the finetuned model assigns probability $p_\theta(y \mid x) = \prod_{j=1}^{m} p_\theta(y_j \mid x, y_{<j})$, where $y_{<j}$ denotes the preceding target tokens. $\mathcal{B}_k \subset \mathcal{D}_k$ is a mini-batch. The mini-batch cross-entropy loss is $\mathcal{L}_{\text{CE}}(\theta; \mathcal{B}_k) = \mathbb{E}_{(x,y)\sim\mathcal{B}_k}\left[-\frac{1}{|y|}\sum_{j=1}^{|y|}\log p_\theta(y_j \mid x, y_{<j})\right]$.

In each round, clients locally optimize their LoRA adapters, and the server aggregates the updated adapter parameters using FedAvg (McMahan et al., 2017) and broadcasts the aggregated adapter parameters for the next round. We denote the final LoRA adapter parameters by θ_π and the final fine-tuned model by $\pi = p_{\theta_\pi}$.

3.2 UNLEARNING OBJECTIVES

Assume that the client k's data are divided into a forget set $\mathcal{D}_{f,k} \subseteq \mathcal{D}_k$ and a retain set $\mathcal{D}_{r,k} = \mathcal{D}_k \setminus \mathcal{D}_{f,k}$. The global forget and retain sets are $\mathcal{D}_f = \bigcup_{k=1}^{K}\mathcal{D}_{f,k}$ and $\mathcal{D}_r = \bigcup_{k=1}^{K}\mathcal{D}_{r,k}$. In the *single-client* setting, one client holds the entire forget set, as in the single-requester setting of Oblivionis (Zhang et al., 2026). In the *multiple-client* setting, the forget set is distributed across multiple clients, with each requesting client holding a local subset of the data to be forgotten. During unlearning, each requesting client accesses only its local forget and retain data.

Unlearning starts from the fine-tuned model π and systematically removes the influence of $\mathcal{D}_f$ while preserving utility on $\mathcal{D}_r$. Following prior work (Fan et al., 2025), the unlearning objective can be formulated as a regularized optimization problem that balances forgetting and retention:

$$\min_\theta \ \mathbb{E}_{(x,y)\sim\mathcal{D}_{f,k}}\left[\ell_f(\theta; x, y)\right] + \lambda\mathbb{E}_{(x,y)\in\mathcal{D}_{r,k}}\left[\ell_{\text{r}}(y \mid x; \boldsymbol{\theta})\right], \tag{2}$$

where ℓ_f denotes the forget loss, ℓ_r denotes the retain loss, and λ controls the trade-off between forgetting and retention. The retain loss is typically implemented as the cross-entropy loss on the retained data. Likelihood-based objectives are widely used to define the forget loss $\ell_f(\theta; x, y)$. A representative approach is gradient ascent (GA), which suppresses the likelihood of the forget samples (Jang et al., 2023): $\ell_f^{\text{GA}}(\theta; x, y) = \log p_\theta(y \mid x)$. However, GA can easily over-suppress the likelihood of forget samples and consequently damage retained knowledge (Jang et al., 2023; Zhang et al., 2024). SimNPO addresses this issue by introducing a likelihood-dependent weighting through the following objective (Fan et al., 2025): $\ell_f^{\text{SimNPO}}(\theta; x, y) = -\frac{2}{\beta}\log\sigma\left(-\frac{\beta}{|y|}\log p_{\boldsymbol{\theta}}(y \mid x) - \gamma\right)$, where $\beta > 0$ controls the shape of the loss and γ is an offset hyperparameter. The gradient of SimNPO can be interpreted as a weighted GA gradient, where the weighting gradually decreases as the likelihood of the forget sample becomes smaller. Thus, SimNPO weakens further suppression once the targeted sample has already been sufficiently forgotten.

4 FEDCODS: COUPLED KNOWLEDGE DISTILLATION AND LIKELIHOOD SUPPRESSION

Likelihood-based unlearning objectives can effectively suppress the probability of targeted tokens and achieve strong forgetting performance. However, their use in federated unlearning introduces

two challenges. First, in the *multi-client* setting, clients hold different local forget data and therefore suppress different tokens' likelihood, which can lead to inconsistent local forgetting directions during aggregation. Second, in the *single-client* setting, preserving model utility is challenging because the requesting client has access only to its limited local retained data for retain loss regularization. Existing likelihood-based methods cannot address these challenges because they rely primarily on decreasing the likelihood of targeted forget tokens. The targeted tokens vary across clients under data heterogeneity. Also, aggressive likelihood suppression can disrupt knowledge acquired during fine-tuning when only a small retain set is available for regularization. These limitations motivate us to develop an unlearning objective that promotes more consistent client forgetting updates while preserving fine-tuned knowledge with limited retain data.

Knowledge distillation (KD) has been used for machine unlearning (Wang et al., 2024b;a; Wu et al., 2024), where a teacher model that either excludes or suppresses the targeted forget data provides a reference distribution to guide the unlearned model. In federated fine-tuning with LoRA the pretrained base model W_0 remains frozen throughout fine-tuning and it is therefore unaffected by the federated fine-tuning data. Since the same W_0 is shared by all clients, W_0 provides a common reference distribution. However, as shown in our experiments, KD generally preserves retain objectives well, but may provide insufficient unlearning of the targeted knowledge. We denote by $p_\theta(\cdot \mid x, y_{<j})$ the student model's predictive distribution over the vocabulary at token position j, and by $q(\cdot \mid x, y_{<j})$ the corresponding predictive distribution of the frozen pretrained model W_0.

Based on these observations, we propose FedCoDS, which couples knowledge distillation from the shared pretrained model with token-wise SimNPO likelihood suppression. We adopt token-wise rather than instance-wise SimNPO, motivated by the granularity study in SatImp, which shows that finer-grained token-level optimization provides a better forgetting–retention trade-off (Yang et al., 2025). The distillation term guides different clients toward a common reference distribution, improving the consistency of local unlearning updates under federated aggregation while helping preserve model utility. In contrast, the token-wise SimNPO suppression term directly reduces the likelihood of targeted forget tokens to ensure effective forgetting. Rather than optimizing these objectives independently, FedCoDS couples them multiplicatively at each answer position. For each requesting client k, we define

$$\mathcal{L}_{F,k}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{f,k}} \frac{1}{|y|} \sum_{j=1}^{|y|} \underbrace{\mathrm{KL}(q(\cdot \mid x, y_{<j}) \,\|\, p_\theta(\cdot \mid x, y_{<j}))^{\beta_1}}_{\text{Knowledge Distillation (KD)}} \cdot \underbrace{\left(\frac{2}{\beta}\log\big(1 + p_\theta(y_j \mid x, y_{<j})^{\beta}\big)\right)^{\beta_2}}_{\text{Token-Wise SimNPO (T-SimNPO)}} \tag{3}$$

Here, $\beta > 0$ controls the token-wise SimNPO loss, and $\beta_1, \beta_2 > 0$ are parameters that can be adjusted: a larger β_1 places greater emphasis on guidance from the shared reference distribution, whereas a larger β_2 places greater emphasis on suppressing the targeted forget tokens.

4.1 Knowledge Distillation from the Shared Pretrained Model

The KD term uses the KL divergence to measure the mismatch between the student and pretrained teacher next-token distributions over the full vocabulary. Denoting the KL divergence at token position j by d_j, its value and gradient with respect to the logits are

$$\begin{aligned} d_j &= \sum_v q(v \mid x, y_{<j}) \log \frac{q(v \mid x, y_{<j})}{p_\theta(v \mid x, y_{<j})}, \\ \nabla_{z_j} d_j &= p_\theta(\cdot \mid x, y_{<j}) - q(\cdot \mid x, y_{<j}), \end{aligned} \tag{4}$$

where z_j denotes the student logit vector at position j, and the sum is taken over the entire vocabulary. Minimizing this KD distills the predictive distribution of the frozen teacher q into the student model (Hinton et al., 2015). Because q is obtained from the shared pretrained model and remains unchanged during federated fine-tuning, it provides all clients with a common reference distribution for reversing distributional changes introduced during fine-tuning. Unlike likelihood suppression, which directly decreases the probability of the targeted token, KD reduces the discrepancy between the student

and teacher distributions over the entire vocabulary. Moreover, because all clients use the same pretrained teacher, KD provides a shared distributional reference across clients, which can promote more consistent unlearning updates before federated aggregation.

4.2 Token-wise suppression and the gradient

We apply the zero-margin SimNPO loss to individual answer tokens rather than the original length-normalized sequence likelihood (Fan et al., 2025). Its value w_j and gradient are

$$\begin{aligned} w_j &= \frac{2}{\beta}\log\big(1+p_\theta(y_j \mid x, y_{<j})^\beta\big)\,, \\ \nabla_{z_j} w_j &= \frac{2p_\theta(y_j \mid x, y_{<j})^\beta}{1+p_\theta(y_j \mid x, y_{<j})^\beta}\left(e_{y_j} - p_\theta(\cdot \mid x, y_{<j})\right), \end{aligned} \tag{5}$$

where e_{y_j} is the one-hot vector for the target token y_j. Minimizing w_j suppresses that token, with a gradient weight that decreases as its probability falls. Token-wise weighting allows different positions in an answer to receive different suppression strengths.

For $d_j > 0$, differentiating the product in Eq. 3 gives

$$\nabla_{z_j}\left(d_j^{\beta_1} w_j^{\beta_2}\right) = \underbrace{\beta_1 d_j^{\beta_1-1} w_j^{\beta_2}\nabla_{z_j} d_j}_{\text{likelihood-weighted distillation}} + \underbrace{\beta_2 d_j^{\beta_1} w_j^{\beta_2-1}\nabla_{z_j} w_j}_{\text{KL-weighted token suppression}}\,. \tag{6}$$

The first item moves the prediction toward the teacher, with its strength weighted by target token probability through w_j. The second item suppresses the target token, with its strength weighted by the remaining teacher–student difference through d_j. For $\beta_1 = \beta_2 = 1$, the two weights reduce to w_j and d_j, respectively, and both components vanish when the student matches the teacher.

4.3 Final Loss Function

Each client combines the forget loss $\mathcal{L}_{F,k}(\theta)$ with KL regularization and cross-entropy retain regularization on its local retain set to get the final loss function:

$$\begin{aligned} \mathcal{L}_k(\theta) = \lambda_f \mathcal{L}_{F,k}(\theta) + \lambda_{\mathrm{CE}}\mathcal{L}_{\mathrm{CE}}(\theta; \mathcal{D}_{r,k}) \\ + \lambda_{\mathrm{KL}}\mathbb{E}_{(x,y)\sim\mathcal{D}_{r,k}}\frac{1}{|y|}\sum_{j=1}^{|y|}\mathrm{KL}(\pi(\cdot \mid x, y_{<j}) \,\|\, p_\theta(\cdot \mid x, y_{<j}))\,, \end{aligned} \tag{7}$$

where λ_f, λ_{CE}, and λ_{KL} are nonnegative weighting coefficients. Cross-entropy regularization on retained data is widely used in likelihood-based unlearning to mitigate utility degradation while unlearning the forget set (Zhang et al., 2024; Fan et al., 2025; Yang et al., 2025). The forget-side KD in $\mathcal{L}_{F,k}$ uses the pretrained model q as the teacher, whereas the retain-side KL uses the fine-tuned model π as the reference. The retain-side KL encourages the unlearned model to preserve the fine-tuned model's predictive distribution on retained samples, while the cross-entropy term directly reinforces the correct retained answers (Maini et al., 2024; Zhang et al., 2024). The retain-side KL is particularly important in the *single-client* setting, where the requesting client has access only to its local retain data. In this case, the fine-tuned model π provides an additional reference that helps preserve behavior learned during federated fine-tuning beyond what can be recovered from the local retain labels alone.

4.4 Understanding the effect of coupling

Gradient alignment under federated aggregation. In multiple-client forgetting, the server averages updates from clients holding different forget records. Thus, even strong local updates can be weakened if their directions disagree. This is related to gradient diversity in distributed optimization (Yin et al., 2018); pairwise cosine similarity has also been used to measure client-update agreement in federated LLM training (Fan et al., 2026). We therefore compute each client's forget gradient with the same fine-tuned model, $g_k = \nabla_\theta \mathcal{L}_{F,k}(\theta_\pi)$, and measure pairwise cosine similarity over all trainable adapter gradients. Higher cosine indicates more consistent client directions and thus less cancellation during aggregation.

For visualization, we convert the cosine similarities among normalized gradients into pairwise distances and project all clients into a shared 2-D space using multidimensional scaling (MDS) (Demircan et al., 2025). Figure 1 shows the resulting shared 2-D MDS map for five methods, where KD denotes the standalone distillation term used in FedCoDS. Nearby points indicate more aligned client gradients. FedCoDS and KD (Individual KD term) both achieve a mean pairwise cosine about 0.86, compared with 0.56 for SATIMP and 0.24/0.29 for SimNPO/NPO.

The strong alignment of FedCoDS is largely due to the same frozen pretrained model across clients. KD provides a consistent optimization direction despite differences in their local forget records. The token-wise suppression term then modulates forgetting according to each client's local targets while preserving much of this cross-client directional consistency. As a result, FedCoDS reduces conflicting local updates and produces more coherent forgetting directions under federated aggregation.

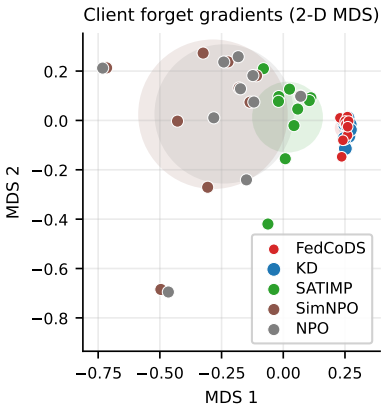

Figure 1: Client forget-gradient alignment on TOFU *multiple-clients* setting. Each point is one normalized client gradient in a shared 2-D MDS map; tighter clusters indicate more consistent directions across clients.

5 Evaluation

5.1 Experimental setup

Benchmarks and settings. We evaluate on TOFU forget05 (Maini et al., 2024) using Llama-3.2-1B-Instruct, and on MUSE-News and MUSE-Books (Shi et al., 2025) using Llama-2-7B and ICLM-7B, respectively. TOFU targets 200 question–answer records about ten fictitious authors; MUSE targets designated news articles or book text.

Our FL system contains ten clients. On TOFU, we compare author-silo (S, non-IID) and random (R) data partitions under single-client (S) and multiple-client (M) forget requests, giving SS, RS, SM, and RM. In the multiple-client setting, the forget set is distributed across all ten clients. MUSE uses RS and RM. Each requesting client accesses only its local forget and retain data. For each benchmark, the *Finetuned* model is obtained through federated LoRA fine-tuning, while the *Retrain* model is trained using the same federated procedure with only the retain dataset. Therefore, both reference models and all reported metrics are obtained from our federated training setup and are not directly comparable to the centralized fine-tuning and retraining results reported in prior work (Fan et al., 2025; Yang et al., 2025). This distinction is important because federated training can underperform centralized training due to model drift (Karimireddy et al., 2020). Within each setting, all unlearning methods start from the same *Finetuned* model. We evaluate up to ten unlearning rounds and, in the multi-client settings, use FedAvg to aggregate the updated LoRA adapters at the end of each round.

Methods, metrics, and reporting. We compare FedCoDS with NPO , SimNPO , and SATIMP. The baselines use local retain cross-entropy; FedCoDS additionally uses the distillation and retain-KL terms in Eq. 7. TOFU metrics are forget and retain extraction strength (ES-U↓, ES-R↑) (Wang et al., 2025), model utility (MU↑), and forget truth ratio (FTR↑). MUSE reports forget-side verbatim and knowledge memorization (VerbMem, KnowMem), retained knowledge (UtilPres↑), and privacy leakage (PrivLeak→ 0). Forget-side generation scores are interpreted relative to Retrain; lower scores alone do not establish better unlearning. TOFU results are based on the mean of three runs. MUSE results correspond to the best single run across three seeds.

5.2 Main results

TOFU Across all four TOFU settings, FedCoDS achieves the best ES-U, ES-R, and FTR, while obtaining the best or second-best MU. Its ES-U remains between .036 and .045, and ES-R between .921 and .973, across both author-silo and random partitions. Compared with SATIMP, FedCoDS consistently lowers ES-U and improves ES-R and FTR, while keeping MU comparable. In particular, under multiple-client requests, ES-U decreases from .090 to .045 in SM and from .088 to .045 in RM, while ES-R remains above .97. Thus FedCoDS is robust to client data partitions.

Table 1: TOFU forget05 at round 10 (three-run mean ± sample SD). FedCoDS best preserves the retained knowledge while achieving the lowest forget extraction. Bold marks the best value among unlearning methods; Finetuned and Retrain are fixed references.

(a) Single-Client

	SS: author silos				RS: random			
Method	ES-U↓	ES-R↑	MU↑	FTR↑	ES-U↓	ES-R↑	MU↑	FTR↑
Finetuned	.978	.979	.544	.492	.990	.971	.542	.495
Retrain	.087	.976	.543	.634	.093	.980	.546	.647
NPO	.094±.002	.485±.004	.511±.002	.566±.003	.057±.002	.749±.019	.511±.003	.573±.003
SimNPO	.108±.005	.690±.029	.502±.007	.584±.006	.040±.006	.651±.013	.521±.007	.608±.002
SATIMP	.052±.004	.811±.012	.535±.001	.690±.006	.045±.001	.803±.004	**.561±.005**	.673±.005
FedCoDS	**.037±.004**	**.921±.003**	**.540±.004**	**.726±.002**	**.036±.000**	**.931±.003**	.552±.001	**.738±.003**

(b) Multiple Client

	SM: author silos				RM: random			
Method	ES-U↓	ES-R↑	MU↑	FTR↑	ES-U↓	ES-R↑	MU↑	FTR↑
Finetuned	.980	.982	.527	.499	.979	.984	.559	.486
Retrain	.089	.984	.544	.641	.090	.984	.561	.636
NPO	.112±.006	.872±.005	.517±.006	.598±.007	.079±.002	.866±.009	.538±.013	.598±.003
SimNPO	.092±.000	.802±.011	.512±.007	.542±.011	.048±.001	.773±.012	.524±.005	.581±.011
SATIMP	.090±.008	.961±.003	**.532±.002**	.628±.003	.088±.005	.969±.006	**.547±.012**	.611±.008
FedCoDS	**.045±.006**	**.971±.007**	.521±.009	**.674±.002**	**.045±.002**	**.973±.006**	.543±.008	**.652±.004**

These results reveal two effects of the federated setting. First, changing from random partitions to non-IID author silos has little effect on the forgetting and retention performance of FedCoDS: ES-U changes by only .001 in the single-client setting and is unchanged in the multi-client setting. MU is more sensitive to the partition, but the overall forgetting–retention behavior remains stable. This indicates that FedCoDS is robust to how knowledge is distributed across clients.

Second, the allocation of forget data affects forgetting and retention differently. Multiple-client requesting forgetting makes forgetting harder: each client observes only part of the forget set, so inconsistent local forgetting directions can be weakened during aggregation. This is particularly visible for SATIMP, whose ES-U increases from .052 to .090 under author silos and from .045 to .088 under random partitions when moving from single- to multi-client requests. In contrast, FedCoDS limits this degradation to .037 → .045 and .036 → .045, respectively. The stronger cross-client gradient alignment observed in Figure 1 is consistent with this behavior.

The single-client setting places pressure on retention because a single requester removes the full forget set while having access to only its local retain data. Accordingly, ES-R is lower in the single-client settings than in the corresponding multi-client settings. Compared to other methods, FedCoDS consistently preserves the best retained behavior while unlearning the target answers.

MUSE: Across two MUSE datasets and two allocation settings, FedCoDS achieves the PrivLeak value closest to Retrain. Its $|\text{PrivLeak}|$ is .4, 8.5, 24.2, and 3.4 for Books single-client, News single-client, Books multi-client, and News multi-client, respectively, outperforming all compared unlearning methods in each setting. At the same time, FedCoDS achieves the highest UtilPres on Books, with 46.5 in the single-client setting and 54.3 in the multi-client setting. On News, NPO retains slightly higher utility (40.7 vs. 38.9 and 39.3 vs. 37.7), but its PrivLeak remains close to that of the fine-tuned model. The UtilPres metric demonstrates that FedCoDS preserves the model utility and improves privacy without relying only on aggressive suppression of forget-text likelihood.

The single-client MUSE-News setting is particularly challenging because the forgetting client contains much more forget dataset than retain dataset. We therefore apply the likelihood-gap regularizer only in the MUSE-News single-client setting. All other MUSE settings use the standard FedCoDS objective without this additional term. Appendix B provides more details.

Table 2: Selected MUSE privacy and utility results with *single-client* and *multiple-client* settings. FedCoDS has the lowest $|\mathrm{PrivLeak}|$ throughout. Bold marks the best $|\mathrm{PrivLeak}|$ and UtilPres.

Method	Books				News			
	VerbMem↓	KnowMem↓	PrivLeak→ 0	UtilPres↑	VerbMem↓	KnowMem↓	PrivLeak→ 0	UtilPres↑
Single Client Unlearning								
Finetuned	65.5	35.7	−100.0	49.1	53.9	44.1	−100	44.2
Retrain	14.6	27.7	0	50.4	20.1	30.9	0	41.3
NPO	16.4	32.2	−100.0	46.3	23.7	34.4	−91.7	**40.7**
SimNPO	0.1	23.0	+11.4	44.1	19.9	26.0	−83.0	32.7
SATIMP	0.0	0.8	−20.8	40.6	14.9	27.1	−98.3	31.4
FedCoDS	1.2	3.9	**−0.4**	**46.5**	18.0	29.9	**−8.5**	**38.9**
Multiple Client unlearning								
Finetuned	77.1	33.3	−100.0	51.4	49.6	42.6	−100	43.2
Retrain	14.9	26.7	0	51.5	19.6	29.7	0	39.9
NPO	19.3	37.5	−100.0	52.4	21.9	31.9	−99.4	**39.3**
SimNPO	13.1	37.3	−28.5	48.9	12.0	31.8	−12.7	30.6
SATIMP	6.7	32.9	−32.5	47.7	15.3	32.6	−34.3	34.4
FedCoDS	2.7	1.4	**−24.2**	**54.3**	14.7	33.7	**+3.4**	**37.7**

Table 3: Ablation of the two FedCoDS components on TOFU over three seeds. Both KL and cross-entropy regularization terms are applied for all three methods. T-SimNPO provides stronger forgetting, while KD improves retention. Bold marks the best mean.

	Single Client		Mutiple Client	
Method	ES-Unl↓	ES-Retain↑	ES-Unl↓	ES-Retain↑
FedCoDS	.0358±.0005	**.9315±.0031**	**.0447±.0016**	**.9727±.0063**
T-SimNPO	**.0234±.0050**	.7775±.0055	.0449±.0149	.9436±.0127
KD	.0926±.0007	.8257±.0069	.0837±.0019	.9692±.0067

6 Ablation Studies

Effect of each term: The results in Table 1 and Table 2 show that FedCoDS achieves a strong balance between forgetting and retention on both TOFU and MUSE. In this study, we isolate the effect of the two components, namely, knowledge distillation (KD) and token-wise SimNPO (T-SimNPO) to better understand their individual roles. Table 3 evaluates each component independently on TOFU under both the single-client and multiple-client settings. In the single-client setting, FedCoDS achieves ES-Retain .9315 with ES-Unl .0358. T-SimNPO achieves stronger forgetting with the lowest ES-Unl of .0234, but substantially reduces ES-Retain to .7775. In the multiple-client setting, FedCoDS and T-SimNPO achieve nearly identical forgetting performance (.0447 vs. .0449 ES-Unl), while FedCoDS improves ES-Retain (.9727 vs. .9436). In contrast, KD provides weaker forgetting, with ES-Unl .0926 and .0837 in the single-client and multiple-client settings, respectively, but better preserves retained behavior. These results show that T-SimNPO provides strong token-level suppression, while KD constrains model changes through distillation from the frozen pretrained model. By combining these complementary strengths, FedCoDS achieves a better balance between effective forgetting and retention, rather than simply maximizing suppression on the forget set.

Forgetting likelihood closer to retrain at the answer level Figure 2 shows that FedCoDS produces forget-answer likelihoods closest to Retrain, with a paired NLL MSE of 1.40, compared with 2.44 for SATIMP, 6.30 for NPO, and 15.89 for SimNPO. NPO and especially SimNPO, place many answers well above the diagonal, indicating that their forget answers become substantially less likely compared to Retrain. In contrast, FedCoDS remains more concentrated around the Retrain reference. This is notable because Retrain is not used as the teacher during unlearning. These results suggest that FedCoDS better matches the forgetting behavior of retraining.

Multi-round unlearning To understand how repeated unlearning rounds affect forgetting and retention, we track the ES-Unl and ES-Retain trajectories over ten unlearning rounds on TOFU. As shown in Figure 3, FedCoDS exhibits an overall improvement in both metrics across rounds. From

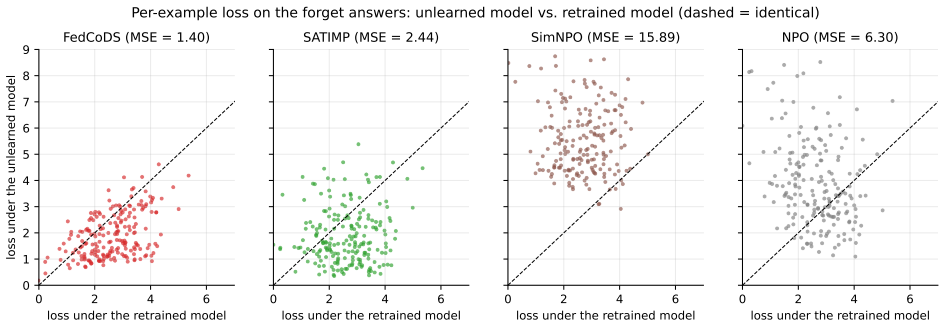

Figure 2: Per-answer (Negative Log-Likelihood) NLL relative to Retrain on the same 200 TOFU-SM forget answers at round 10 (saved seed-0 runs). The diagonal indicates equal NLL. SimNPO here is the original sequence-level baseline.

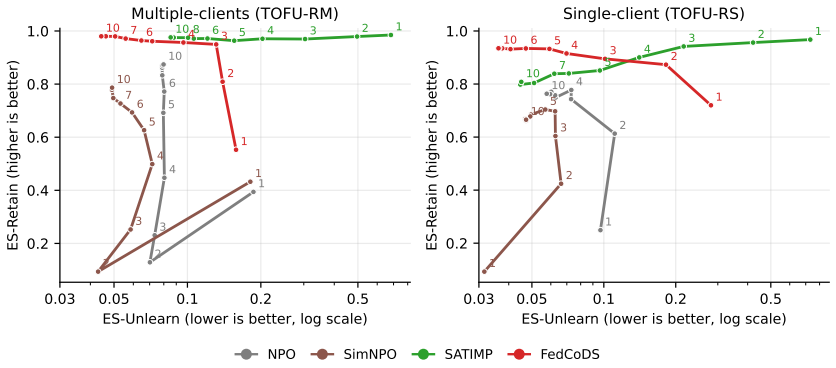

Figure 3: ES-Unl/ES-Retain trajectories over ten unlearning rounds on TOFU-RM and TOFU-RS, using the original saved runs. Numbers indicate rounds; ES-Unl uses a logarithmic axis. FedCoDS improves both extraction metrics from its first to its final round.

round 1 to round 10, its ES-Unl/ES-Retain changes from $.281/.720$ to $.036/.935$ in RS and from $.158/.553$ to $.044/.980$ in RM. Thus, additional rounds do not merely strengthen suppression; they progressively improve the overall forgetting–retention trade-off.

The baselines exhibit markedly different trajectories. NPO and SimNPO recover from substantial early retention loss, but their final retention remains lower than that of FedCoDS. SATIMP starts with high retention but weak forgetting and gradually strengthens forgetting; in multiple-client setting, this improvement is accompanied by a decrease in retention. These trajectories show that one-round performance alone may not fully characterize federated unlearning, as repeated optimization and aggregation can substantially change the balance between forgetting and retention. In contrast, FedCoDS moves overall toward the desirable upper-left region of the plot—lower ES-Unl and higher ES-Retain—in both settings. This behavior is consistent with the complementary effects of its coupled forget objective and retain regularization, which allow stronger forgetting over multiple rounds while preserving retained behavior.

7 Conclusion

We study two key challenges in federated LLM unlearning: preserving globally learned knowledge in the single-client setting and maintaining consistent forgetting updates across clients in multi-client unlearning. To address these challenges, we propose FedCoDS, which couples knowledge distillation with token-wise likelihood suppression. Experiments on TOFU and MUSE show that FedCoDS consistently improves the forgetting–retention performance across both settings, with more aligned client gradients and forget-answer likelihoods closer to Retrain.

REPRODUCIBILITY STATEMENT

We provide the complete FedCoDS objective and federated unlearning procedure in the main paper. The appendix reports the dataset partitions, model and training configurations, baseline implementations, hyperparameters, and random seeds used in our experiments. Code and configurations will be released with the camera-ready version.

AI USE STATEMENT

In this work we used generative AI tools for editing text. Specifically, we used OpenAI Codex for grammar correction, language editing, table formatting. We also used it for coding assistance when reproducing related methods for our ablation study. We have not used generative AI tools for generating research ideas or for producing experimental results; all numbers reported were produced by the authors' code and re-verified against saved per-example outputs. We take responsibility for the final manuscript, code, and reported results.

REFERENCES

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning, 2020. URL https://arxiv.org/abs/1912.03817.

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

Shuangyi Chen, Yuanxin Guo, Yue Ju, Hardik Dalal, Zhongwen Zhu, and Ashish Khisti. Robust federated finetuning of llms via alternating optimization of lora. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 120038–120090. Curran Associates, Inc., 2025. doi: 10.52202/085713-4009. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/adf17e7346b6be4c2f2bb40de572e5bc-Paper-Conference.pdf.

Can Demircan, Tankred Saanum, Akshay Jagadish, Marcel Binz, and Eric Schulz. Sparse autoencoders reveal temporal difference learning in large language models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 23743–23768, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/3b594f6dd4931c4af4f12042af8d40ec-Paper-Conference.pdf.

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 Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *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, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.444. URL https://aclanthology.org/2025.naacl-long.444/.

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 D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 1540–1567. Curran Associates, Inc., 2025. doi: 10.52202/085713-0058. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/02443e4e008231e0af0855f3cc70ed17-Paper-Conference.pdf.

Tao Fan, Yan Kang, Guoqiang Ma, Weijing Chen, Wenbin Wei, Lixin Fan, and Qiang Yang. Fate-llm: A industrial grade federated learning framework for large language models. *Symposium on Advances and Open Problems in Large Language Models (LLM@IJCAI'23)*, 2023.

Tao Fan, Guoqiang Ma, Yuanfeng Song, Lixin Fan, Kai Chen, and Qiang Yang. FedProxy: Federated fine-tuning of LLMs via proxy SLMs and heterogeneity-aware fusion. In Maria Liakata, Viviane P.

Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17482–17497, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.794. URL https://aclanthology.org/2026.acl-long.794/.

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

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR 2022*, April 2022.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14389–14408, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.805. URL https://aclanthology.org/2023.acl-long.805/.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5132–5143. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/karimireddy20a.html.

Gaoyang Liu, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu. Federaser: Enabling efficient client-level data removal from federated learning models. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pp. 1–10, 2021. doi: 10.1109/IWQOS52092.2021.9521274.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C. Lipton, and J. Zico Kolter. Tofu: A task of fictitious unlearning for llms, 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. URL https://proceedings.mlr.press/v54/mcmahan17a.html.

Weijia Shi, Jaechan Lee, Yangsibo Huang, Sadhika Malladi, Jieyu Zhao, Ari Holtzman, Daogao Liu, Luke Zettlemoyer, Noah Smith, and Chiyuan Zhang. Muse: Machine unlearning six-way evaluation for language models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 27797–27818, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/4556f5398bd2c61bd7500e306b4e560a-Paper-Conference.pdf.

Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving lora in privacy-preserving federated learning. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 17978–17994, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/4e243e95c913b367775d71d7182b99d9-Paper-Conference.pdf.

Bichen Wang, Yuzhe Zi, Yixin Sun, Yanyan Zhao, and Bing Qin. Rkld: Reverse kl-divergence-based knowledge distillation for unlearning personal information in large language models, 2024a. URL https://arxiv.org/abs/2406.01983.

Hangyu Wang, Jianghao Lin, Bo Chen, Yang Yang, Ruiming Tang, Weinan Zhang, and Yong Yu. Towards efficient and effective unlearning of large language models for recommendation. *Front. Comput. Sci.*, 19(3), November 2024b. ISSN 2095-2228. doi: 10.1007/s11704-024-40044-2. URL https://doi.org/10.1007/s11704-024-40044-2.

Qizhou Wang, Bo Han, Puning Yang, Jianing ZHU, Tongliang Liu, and Masashi Sugiyama. Towards effective evaluations and comparisons for llm unlearning methods. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 23327–23355, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/3a91841d2bcc0b13a3d0d5d60c9f0581-Paper-Conference.pdf.

Ziyao Wang, Zheyu Shen, Yexiao He, Guoheng Sun, Hongyi Wang, Lingjuan Lyu, and Ang Li. Flora: Federated fine-tuning large language models with heterogeneous low-rank adaptations. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 22513–22533. Curran Associates, Inc., 2024c. doi: 10.52202/079017-0708. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/28312c9491d60ed0c77f7fff4ad86dd1-Paper-Conference.pdf.

Chen Wu, Sencun Zhu, and Prasenjit Mitra. Federated unlearning with knowledge distillation, 2022. URL https://arxiv.org/abs/2201.09441.

Chen Wu, Sencun Zhu, Prasenjit Mitra, and Wei Wang. Unlearning backdoor attacks in federated learning. In *2024 IEEE Conference on Communications and Network Security (CNS)*, pp. 1–9, 2024. doi: 10.1109/CNS62487.2024.10735680.

Zhiqiang Xie, Yijing Lin, Zhipeng Gao, and Jingbo Fan. A dual-level selective unlearning framework for multi-agent decentralized federated learning. *IEEE Transactions on Cognitive Communications and Networking*, 12:9867–9882, 2026. doi: 10.1109/TCCN.2026.3711105.

Puning Yang, Qizhou Wang, Zhuo Huang, Tongliang Liu, Chengqi Zhang, and Bo Han. Exploring criteria of loss reweighting to enhance llm unlearning. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025.

Yuanshun Yao, Xiaojun Xu, and Yang Liu. Large language model unlearning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 105425–105475. Curran Associates, Inc., 2024. doi: 10.52202/079017-3346. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/be52acf6bccf4a8c0a90fe2f5cfcead3-Paper-Conference.pdf.

Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. Openfedllm: Training large language models on decentralized private data via federated learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 6137–6147, 2024.

Dong Yin, Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett. Gradient diversity: a key ingredient for scalable distributed learning. In Amos Storkey and Fernando Perez-Cruz (eds.), *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pp. 1998–2007. PMLR, 09–11 Apr 2018. URL https://proceedings.mlr.press/v84/yin18a.html.

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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(33):28185–28193, Mar. 2026. doi: 10.1609/aaai.v40i33.40045. URL https://ojs.aaai.org/index.php/AAAI/article/view/40045.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning, 2024. URL https://arxiv.org/abs/2404.05868.

Yisheng Zhong, Zhengbang Yang, and Zhuangdi Zhu. Hierarchical federated unlearning for large language models, 2025. URL https://arxiv.org/abs/2510.17895.

A EXPERIMENTAL SETTINGS AND HYPERPARAMETERS

We report the experimental configurations for Tables 1–3 and Figures 1–3. A single-client request assigns the entire forget set to one client, whereas a multiple-client request distributes it across all ten clients. SS and SM denote author-silo single- and multiple-client settings; RS and RM denote their random-partition counterparts. TOFU uses all four settings, and MUSE uses RS and RM.

A.1 MODELS AND TRAINING PROTOCOL

Table 4 summarizes the model and training settings. Within each setting, all methods start from the same *Finetuned* model π, trained on $\mathcal{D}_f \cup \mathcal{D}_r$. *Retrain* is trained independently from the same pretrained backbone on $\mathcal{D}_r$ and is used only for evaluation. *Finetuned/Retrain* training involves all ten clients in every federated round. MUSE additionally uses separate centralized LoRA warm-ups for Finetuned and Retrain, using all data and retain data, respectively, before federated training.

Table 4: Model and training configurations. Paired entries give the single-/multiple-client settings. Reference training constructs both Finetuned and Retrain.

Configuration	TOFU	MUSE-News	MUSE-Books
Backbone	Llama-3.2-1B-Instruct	Llama-2-7B	ICLM-7B
LoRA rank / scaling	32 / 64	128 / 256	128 / 256
Adapted modules	`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`		
Reference-model training			
LoRA warm-up epochs	None	2	3
Warm-up learning rate	–	2.0×10^{-4}	2.0×10^{-4}
Federated rounds (S/M)	20/20	10 / 25	10 / 15
Local training per round	1 epoch	1 epoch	1 / 5 epochs
Federated learning rate	5.0×10^{-4}	2.0×10^{-4}	2.0×10^{-4}
Federated batch size	16	32	32
Aggregation	FedAvg	FedAvg	FedAvg
LoRA dropout	0.05	0.05	0.05
Unlearning			
Input format / block length	Question–answer pairs	4,096 tokens	2,048 tokens
Batch size	10	8	16
LoRA dropout	0.05	0	0

During unlearning, the pretrained teacher q and the retain reference π remain frozen, and only LoRA parameters are updated. A single requester continues its adapter locally; multiple requesters use equal-weight FedAvg of their LoRA factors after each round.

Unlearning uses AdamW with moments $(0.9, 0.999)$, $\epsilon = 10^{-8}$, weight decay 0.01, and gradient-norm clipping at 1. TOFU uses linear learning-rate decay to zero without warm-up; MUSE-News also uses linear learning-rate decay. All learning rates in the unlearning tables are initial rates.

A.2 MAIN-BODY TABLES AND FIGURES

Table 5 maps each main-body display to its experimental configuration. Figures 2 and 3 reuse runs from Table 1, and the FedCoDS entries in Table 3 reuse its RS/RM results.

For Figure 1, each client contributes the forget gradient over all trainable LoRA parameters, with dropout disabled and without retain losses or optimizer steps. The normalized gradients share a two-dimensional MDS embedding of distances $\sqrt{2 - 2\cos(g_i, g_j)}$. For Figure 2, all models score the same 200 forget answers; NLL is averaged over each answer's supervised tokens, and MSE is the mean squared difference between the unlearned and Retrain answer-level NLLs.

Table 5: Configurations used by the main-body tables and figures.

Main-body display	Configuration
Table 1	TOFU SS, SM, RS, and RM; hyperparameters in Table 6; round-10 results over seeds 0–2.
Table 2	MUSE-News and MUSE-Books RS/RM; hyperparameters in Table 7; ten rounds per seed, with the reporting procedure in Appendix A.4.
Table 3	TOFU RS/RM; KD and T-SimNPO hyperparameters in Table 8; round 10, seeds 0–2. FedCoDS reuses Table 1.
Figure 1	TOFU SM forget-only gradients at the common Finetuned model. The four unlearning methods use the SM forget-loss parameters in Table 6
Figure 2	Seed-0, round-10 TOFU-SM models from Table 1, paired with the same Retrain reference.
Figure 3	TOFU-RS/RM runs from Table 1, evaluated at rounds 1–10.

A.3 Loss parameters and TOFU configurations

We write $\lambda = (\lambda_f, \lambda_{\mathrm{KL}}, \lambda_{\mathrm{CE}})$ for the coefficients in Eq. 7. For NPO, SimNPO, and SATIMP, α_r is the retain cross-entropy coefficient; their forget coefficient is 1 and their retain-KL coefficient is 0. NPO uses the fixed fine-tuned model π as its likelihood reference (Zhang et al., 2024), while SimNPO uses the sequence-level, length-normalized, zero-margin objective (Fan et al., 2025). SATIMP uses the token weight $p_\theta(y_j \mid x, y_{<j})^a(1 - p_\theta(y_j \mid x, y_{<j}))^b$, which is detached when multiplying the target log-likelihood (Yang et al., 2025). Distillation uses temperature 1.

Table 6 lists the hyperparameters for the four TOFU settings in Table 1. TOFU forget05 contains 200 forget records and 3,800 retain records (Maini et al., 2024). The single-client requester has all 200 forget records and 200 local retain candidates; each multiple-client requester has 20 forget records and 380 local retain candidates. All methods run ten unlearning rounds with seeds 0, 1, and 2. Table 1 reports the round-10 mean and sample standard deviation. The client partition and reference models remain fixed across seeds.

Table 6: TOFU hyperparameters used for Table 1. FedCoDS uses $(\beta_1, \beta_2) = (1, 1)$

Setting	Method	Learning rate	Loss parameters
SS	NPO	3.4×10^{-5}	$\beta = 0.1$, $\alpha_r = 0.3$
	SimNPO	6.7×10^{-5}	$\beta = 2.5$, $\alpha_r = 0.3$
	SATIMP	3.4×10^{-5}	$(a, b) = (5, 0.1)$, $\alpha_r = 0.3$
	FedCoDS	6.7×10^{-5}	$\beta = 5$, $\lambda = (1.5, 4, 3)$
SM	NPO	2.0×10^{-3}	$\beta = 0.1$, $\alpha_r = 0.3$
	SimNPO	1.0×10^{-3}	$\beta = 1$, $\alpha_r = 0.1$
	SATIMP	1.0×10^{-3}	$(a, b) = (5, 0.1)$, $\alpha_r = 0.6$
	FedCoDS	1.0×10^{-3}	$\beta = 1.2$, $\lambda = (1, 2, 3)$
RS	NPO	1.3×10^{-4}	$\beta = 0.1$, $\alpha_r = 10$
	SimNPO	1.3×10^{-4}	$\beta = 2.5$, $\alpha_r = 0.3$
	SATIMP	3.4×10^{-5}	$(a, b) = (5, 0.1)$, $\alpha_r = 0.3$
	FedCoDS	6.7×10^{-5}	$\beta = 4$, $\lambda = (1.5, 1, 3)$
RM	NPO	2.0×10^{-3}	$\beta = 0.1$, $\alpha_r = 0.3$
	SimNPO	2.0×10^{-3}	$\beta = 1$, $\alpha_r = 0.1$
	SATIMP	1.0×10^{-3}	$(a, b) = (5, 0.1)$, $\alpha_r = 0.6$
	FedCoDS	1.0×10^{-3}	$\beta = 1.68$, $\lambda = (1, 2, 3)$

A.4 MUSE configurations

Table 7 lists the hyperparameters used for the MUSE results in Table 2. For each method and setting, we run ten unlearning rounds with seeds 0, 1, and 2, select the best round within each seed, and then report the best of the three selected checkpoints. FedCoDS on MUSE-News RS uses the additional gap regularizer in Appendix B; $\lambda_g = 0$ in all other MUSE settings.

For MUSE-News, the single-client requester holds 250 forget blocks and 40 local retain blocks; multiple-client requesters hold 22–28 forget blocks and 40–41 retain blocks each. MUSE-Books has

553 forget blocks and 105 `retain1` blocks in total. Its single-client requester holds all 553 forget blocks and 10 local `retain1` blocks; each multiple-client requester holds 55–56 forget blocks and 10–11 `retain1` blocks. Only the local `retain1` pool is used for MUSE-Books unlearning.

Table 7: MUSE unlearning hyperparameters for Table 2. RS and RM denote random-partition single- and multiple-client requests.

Dataset	Setting	Method	Learning rate	Loss parameters
News	RS	NPO	5.0×10^{-4}	$\beta = 0.01$, $\alpha_r = 0.3$
		SimNPO	5.0×10^{-4}	$\beta = 2.5$, $\alpha_r = 0.3$
		SATIMP	5.0×10^{-4}	$(a, b) = (5, 0.1)$, $\alpha_r = 0.3$
		FedCoDS	5.0×10^{-4}	$\beta = 1$, $(\beta_1, \beta_2) = (1, 0.2)$, $\lambda = (1, 4, 3)$, $\lambda_g = 1$
	RM	NPO	1.5×10^{-3}	$\beta = 0.1$, $\alpha_r = 0.3$
		SimNPO	5.6×10^{-5}	$\beta = 1$, $\alpha_r = 0.05$
		SATIMP	1.0×10^{-3}	$(a, b) = (5, 0.05)$, $\alpha_r = 0.6$
		FedCoDS	1.0×10^{-3}	$\beta = 1.5$, $(\beta_1, \beta_2) = (1, 1)$, $\lambda = (1, 2, 3)$
Books	RS	NPO	5.0×10^{-4}	$\beta = 0.1$, $\alpha_r = 0.3$
		SimNPO	1.7×10^{-4}	$\beta = 2.5$, $\alpha_r = 0.3$
		SATIMP	1.7×10^{-4}	$(a, b) = (5, 0.1)$, $\alpha_r = 0.3$
		FedCoDS	2.0×10^{-5}	$\beta = 1$, $(\beta_1, \beta_2) = (0.5, 2)$, $\lambda = (1, 8, 3)$
	RM	NPO	2.8×10^{-5}	$\beta = 0.01$, $\alpha_r = 0.3$
		SimNPO	5.6×10^{-5}	$\beta = 1$, $\alpha_r = 0.05$
		SATIMP	2.8×10^{-5}	$(a, b) = (5, 0.05)$, $\alpha_r = 0.6$
		FedCoDS	1.5×10^{-4}	$\beta = 1.5$, $(\beta_1, \beta_2) = (0.5, 2)$, $\lambda = (2, 3, 3)$

MUSE evaluation uses the benchmark prompts and greedy decoding, with at most 128 new tokens for VerbMem and 32 for knowledge questions. VerbMem, KnowMem, and UtilPres are ROUGE-L F1 scores multiplied by 100. PrivLeak uses the Min-40% token-likelihood membership audit with 100 forget and 100 holdout chunks, normalized to the corresponding Retrain AUC.

A.5 Component comparison

Table 8 gives the KD and T-SimNPO configurations for Table 3. KD minimizes the forget-side Knowledge distillation d_j in Eq. 4, while T-SimNPO minimizes w_j in Eq. 5. Both retain the KL regularization to π and the retain cross-entropy term, using the coefficients listed below. The FedCoDS configurations and results are the same RS/RM entries as in Table 1 and are not repeated here. All component results are round-10 means and sample standard deviations over seeds 0–2.

Table 8: KD and T-SimNPO hyperparameters for Table 3. The FedCoDS configurations are the RS/RM rows of Table 6.

Setting	Method	Learning rate	β	λ_f	λ_{KL}	λ_{CE}
RS	T-SimNPO	6.7×10^{-5}	4	3	1	3
	KD	3.4×10^{-5}	–	1	8	6
RM	T-SimNPO	1.0×10^{-3}	1.68	3	2	3
	KD	1.0×10^{-3}	–	1	4	3

B Log-likelihood-gap regularization

For single-client MUSE-News (RS), the requesting client has 250 forget blocks but only 40 local retain blocks, which are extremely unbalanced. Therefore, we augment FedCoDS with an auxiliary log-likelihood-gap regularizer inspired by NPO.

Motivation from NPO. NPO compares the likelihood of a forget response with that under the fixed fine-tuned model π. Its per-example forget loss can be written as (Zhang et al., 2024, Eq. (3))

$$\ell_f^{\mathrm{NPO}}(\theta; x, y) = \frac{2}{\beta} \log \left[1 + \left(\frac{p_\theta(y \mid x)}{\pi(y \mid x)} \right)^\beta \right], \qquad \beta > 0. \tag{8}$$

Minimizing this loss suppresses the response likelihood, with a gradient weight that decreases as its likelihood relative to π decreases.

Log-likelihood-gap regularizer. Inspired by this likelihood comparison, we use the frozen pretrained model q to define a penalty on individual forget tokens:

$$\mathcal{R}_{\mathrm{gap},k}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{f,k}} \left[\frac{1}{|y|} \sum_{j=1}^{|y|} \left[\log \frac{p_\theta(y_j \mid x, y_{<j})}{q(y_j \mid x, y_{<j})} \right]_+ \right], \qquad [u]_+ = \max(0, u). \tag{9}$$

The positive part is applied separately to each token, followed by averaging within each example and then over examples. Like NPO, the regularizer compares model likelihoods. It differs by using the pretrained model q rather than the fine-tuned model π, applying the comparison token-wise rather than sequence-wise, and replacing NPO's smooth penalty with $[\cdot]_+$. It is an additional regularizer, not a replacement for the coupled FedCoDS forget loss.

Effect and complete objective. A token contributes a positive penalty only when $p_\theta(y_j \mid x, y_{<j}) > q(y_j \mid x, y_{<j})$. Because q is frozen, the gradient of an active token penalty is $\nabla_\theta \log p_\theta(y_j \mid x, y_{<j})$, which gives a likelihood suppression direction. Below the teacher's probability the gradient is zero. Thus, the term adds suppression when forget-token likelihood exceeds the pretrained model's level. Unlike NPO's smooth penalty, this term provides no further suppression at or below its reference probability. The resulting local objective is

$$\mathcal{L}_k^{\mathrm{gap}}(\theta) = \mathcal{L}_k(\theta) + \lambda_f \lambda_g \mathcal{R}_{\mathrm{gap},k}(\theta), \tag{10}$$

where $\mathcal{L}_k$ is defined in Eq. 7; the coupled forget loss and retain terms are unchanged. We use $\lambda_g = 1$ for MUSE-News RS, with the remaining coefficients in Table 7, and $\lambda_g = 0$ in all other settings reported in the main body.

C Robustness to a contaminated pretrained model

C.1 Experimental setup

We introduce contamination by fine-tuning all parameters of the Llama-3.2-1B-Instruct backbone on nested, stratified TOFU subsets before federated training. The 5%, 10%, and 20% levels contain 200, 400, and 800 of the 4,000 records, including exactly 10, 20, and 40 of the 200 forget records. Contamination training uses two epochs, learning rate 1.0×10^{-4}, batch size 16, AdamW with weight decay 0.01, and linear learning-rate decay without warm-up.

For each contamination level, we construct TOFU-RS and TOFU-RM Finetuned/Retrain pairs using the reference-training protocol in Appendix A.1. The contaminated backbone remains frozen during subsequent training and unlearning and serves as the FedCoDS forget teacher q. The retain reference π is the corresponding Finetuned model. Retrain also inherits the contaminated backbone; forget records are excluded only from its subsequent federated training. Thus, this experiment changes the pretrained backbone shared by all methods.

Table 9 gives the per-level configurations used for the results in Table 10. All methods use ten unlearning rounds and seeds 0–2 under the TOFU protocol. NPO uses $\beta = 0.1$; SimNPO uses $\beta = 2.5$ in RS and $\beta = 1$ in RM; SATIMP uses $(a, b) = (5, 0.1)$. FedCoDS uses $(\beta_1, \beta_2) = (0.5, 1)$, with $\beta = 4$ in RS and $\beta = 1.68$ in RM.

C.2 Results and analysis

Table 10 reports forget and retain extraction at the three contamination levels. Results without injected contamination are already reported in Table 1 and are not duplicated.

Table 9: Unlearning hyperparameters for the contaminated-backbone results in Table 10. Baseline entries give learning rate/α_r; FedCoDS entries give learning rate/λ.

Setting	Level	NPO	SimNPO	SATIMP	FedCoDS
RS	5%	$6.7\times10^{-5}/10$	$1.3\times10^{-4}/0.3$	$2.4\times10^{-5}/0.3$	$6.7\times10^{-5}/(3,1,3)$
	10%	$1.3\times10^{-4}/10$	$1.3\times10^{-4}/0.3$	$2.4\times10^{-5}/0.3$	$6.7\times10^{-5}/(4.5,1,3)$
	20%	$4.8\times10^{-5}/10$	$3.4\times10^{-5}/0.3$	$2.4\times10^{-5}/0.3$	$6.7\times10^{-5}/(1.5,1,3)$
RM	5%	$1.0\times10^{-3}/10$	$2.0\times10^{-3}/3$	$5.0\times10^{-4}/0.6$	$8.1\times10^{-4}/(2,2,3)$
	10%	$1.0\times10^{-3}/0.3$	$1.0\times10^{-3}/1$	$1.0\times10^{-3}/0.6$	$1.0\times10^{-3}/(3,2,3)$
	20%	$5.0\times10^{-4}/3$	$1.0\times10^{-3}/3$	$6.0\times10^{-4}/0.6$	$1.0\times10^{-3}/(1,2,3)$

Table 10: TOFU forget and retain extraction under pretrained-backbone contamination. Unlearning results are round-10 means ± sample SD over three seeds. Bold marks the best mean among unlearning methods; Retrain is a fixed reference.

	5% contamination		10% contamination		20% contamination	
Method	ES-U↓	ES-R↑	ES-U↓	ES-R↑	ES-U↓	ES-R↑
Single-client requests (RS)						
Retrain	.091	.993	.093	.987	.120	.994
NPO	.130±.007	.809±.010	.110±.002	.791±.012	.134±.003	.807±.010
SimNPO	.119±.002	.740±.012	.103±.002	.726±.009	.176±.002	.730±.005
SATIMP	.127±.007	.911±.003	.130±.006	.919±.004	.157±.005	.914±.002
FedCoDS	**.066±.001**	**.945±.001**	**.076±.002**	**.935±.006**	**.131±.002**	**.952±.000**
Multiple-client requests (RM)						
Retrain	.087	.988	.096	.993	.133	.996
NPO	.129±.002	.962±.009	.113±.006	.909±.007	.180±.002	.919±.001
SimNPO	.125±.003	.876±.012	.130±.009	.923±.006	.194±.016	.935±.004
SATIMP	.190±.005	**.983±.002**	**.077±.017**	.974±.006	.184±.018	**.993±.003**
FedCoDS	**.079±.001**	.982±.004	.080±.002	**.980±.005**	**.115±.003**	.989±.002

Single-client results. FedCoDS achieves the lowest mean ES-U and highest mean ES-R among unlearning methods at all three contamination levels. Its ES-U increases from .066 at 5% contamination to .131 at 20%, while ES-R remains between .935 and .952. At 20%, for example, FedCoDS obtains ES-U/ES-R of .131/.952, compared with .134/.807 for NPO and .157/.914 for SATIMP. Thus, although increasing contamination makes forgetting harder, FedCoDS maintains the strongest retention while achieving comparable or lower forget extraction.

Multiple-client results. At 5% and 20% contamination, FedCoDS achieves the lowest mean ES-U (.079 and .115), compared with .190 and .184 for SATIMP, while its ES-R is lower by only .001 and .004, respectively. At 10%, SATIMP has slightly lower mean ES-U (.077 versus .080), whereas FedCoDS has higher mean ES-R (.980 versus .974). These comparisons show a favorable forgetting–retention trade-off across the three contamination levels.

Together, these results support the robustness of the FedCoDS forgetting–retention performance when the frozen pretrained model already contains some forget knowledge.