WHEN THE BUBBLE BURSTS: MALICIOUS UNLEARNING REQUESTS UNDERMINE FEDERATED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Growing demand for the "right to be forgotten" has motivated federated unlearning (FU), which aims to efficiently remove the influence of designated training data while preserving model utility. However, we identify a security gap in existing FU approaches: treating client-issued unlearning requests as trusted inputs allows adversaries to exploit the unlearning service itself to manipulate model predictions. To expose this vulnerability, we propose BURST, an attack framework that induces misclassification of selected samples through malicious unlearning requests. Specifically, the adversary constructs a target-support set and an interference set from public auxiliary data and incorporates both into federated training, using these external samples to exert controllable adversarial influence on the target prediction. The support set reinforces the model's correct prediction for the target sample, while the interference set is designed to undermine it. During the unlearning phase, the adversary submits malicious requests to unlearn the support set and strengthens the interference set's adversarial influence, jointly turning previously correct predictions into misclassifications. Extensive experiments show that BURST achieves an attack success rate of 91.43% on CIFAR-10 while reducing global model accuracy by only 5.66%. Among misclassified targets, confidence in the predicted labels exceeds confidence in the ground-truth labels by an average of 30.74 percentage points. Our findings demonstrate how client-controlled unlearning requests and data can jointly compromise FU, highlighting the need to validate both to safeguard model integrity. Code is available at `https://anonymous.4open.science/r/Code-for-BURST-4CC1`.

1 INTRODUCTION

Federated learning (FL) enables multiple parties to train a shared model by aggregating locally computed updates without exchanging raw data (McMahan et al., 2017; Kairouz et al., 2021). In practice, clients may later request the removal of particular training records because of privacy concerns, withdrawn authorization, or regulatory requirements. Article 17 of the European Union's General Data Protection Regulation (GDPR), for example, grants individuals the right to request erasure under specified conditions (European Parliament and Council of the European Union, 2016). Deleting a local record, however, does not remove its influence from a model that has already been trained. Retraining from scratch after excluding the requested data would directly address this problem, but typically incurs substantial computational, communication, and latency costs. Machine unlearning aims to remove the influence of designated data from a trained model while preserving its utility (Cao & Yang, 2015; Bourtoule et al., 2021). In the federated setting, this objective must be achieved without centralizing the original data. Existing federated unlearning (FU) methods reduce the cost of complete retraining by reusing information from the original training process, such as historical client updates and calibration training in FedEraser (Liu et al., 2021), while other approaches explore client-side unlearning, rapid retraining, and gradient-based approximations (Halimi et al., 2022; Liu et al., 2022; Wu et al., 2022).

Despite the efficiency gains offered by federated unlearning, the security implications of client-issued unlearning requests remain insufficiently characterized. **First**, the central question is: *Can a malicious unlearning request compromise the predictions of a trained model?* Evidence from centralized machine unlearning suggests that it can: Unlearn and Burn (Huang et al., 2025) demonstrates

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.

that inadequate validation of unlearning requests can be exploited to degrade model performance. This finding motivates a **second** question: *Can federated unlearning pipelines reliably identify and resist such malicious requests?* This question is particularly difficult in the federated setting. Data ownership is distributed across clients, while statistical heterogeneity across local datasets limits the server's ability to independently verify the legitimacy of a request and the provenance of its associated data. These observations motivate us to examine how attacker-controlled training data and subsequent unlearning requests can jointly alter predictions on selected samples in FL setting.

Achieving such a sample-level failure through unlearning is nontrivial. Removing a small number of records that support a target's ground-truth class may be insufficient because other clients can provide compensating evidence. Conversely, introducing strong competing-class influence too early may cause the target to be misclassified before the unlearning request is processed, making the attack easier to detect. A successful attack must therefore preserve a correct prediction at the end of federated training and change the balance of class influences only during unlearning. Public auxiliary data provide a flexible source of additional samples, allowing the adversary to construct these complementary influences without relying solely on the distribution of a malicious client's local data.

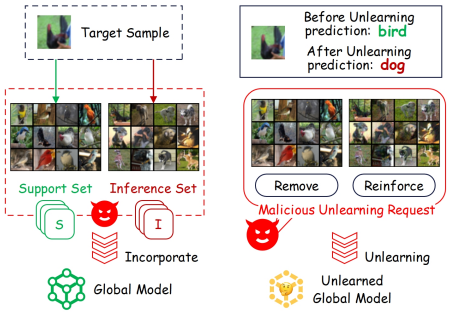


Figure 1: Given a target sample that is correctly classified by the global model, BURST coordinates a malicious unlearning request with complementary support and interference influences to alter target prediction. The target initially correct prediction becomes incorrect after unlearning.

Based on this observation, we propose BURST, a malicious unlearning attack framework for federated learning that uses public auxiliary data. The adversary constructs two complementary sets: a target-support set and an interference set. Both sets are incorporated into federated training, where the support set helps preserve the target's correct prediction and the interference set establishes additional influence from a competing class. After training, the adversary submits a malicious unlearning request to remove the target-support set and strengthens the contribution of the interference set during unlearning. We refer to these operations as *Support Removal* and *Interference Enhancement*, respectively. Their coordinated effect weakens support for target's ground-truth class while increasing competing-class influence, turning a previously correct prediction into a misclassification after unlearning. Figure 1 provides a conceptual illustration of the attack scenario studied in this work.

Across the evaluated federated settings and attack configurations, extensive experiments show that BURST reliably induces sample-level misclassification while retaining most global model utility. On CIFAR-10, BURST causes 91.43% of targets that were correctly classified before unlearning to become misclassified afterward. For misclassified targets, confidence in the predicted labels exceeds confidence in the ground-truth labels by an average of 30.74 percentage points, while global accuracy decreases by only 5.66 percentage points. These results reveal a vulnerability in current FU pipelines: client-controlled unlearning operations can manipulate predictions on selected samples while leaving the global model largely functional. FU pipelines should therefore validate both client-issued unlearning requests and the data used in subsequent model updates, rather than relying solely on global utility or a single request-legitimacy check.

Our contributions are summarized as follows:

- We identify a sample-level security gap in FU: trusted client-issued requests can expose the unlearning service to prediction manipulation.
- We propose a coordinated attack framework that constructs target-support and interference sets from public auxiliary data and combines *Support Removal* with *Interference Enhancement* to induce post-unlearning misclassification.
- We conduct a comprehensive evaluation across multiple datasets, federated unlearning methods, and attack baselines. Our results show that BURST causes substantial degradation in selected-sample predictions while preserving most of the global model's utility.

2 Related Work

Federated Unlearning. Machine unlearning aims to remove the influence of designated training data while preserving model utility (Cao & Yang, 2015; Guo et al., 2020; Golatkar et al., 2020; Bourtoule et al., 2021). Federated unlearning extends this objective to distributed training, where the server must remove the influence of a client or a subset of records without centralizing the remaining data. Existing methods can be broadly grouped into two directions. Update-reuse-based approaches reuse historical client updates or partial training trajectories to reduce the cost of complete retraining, as in FedEraser, rapid retraining, and SIFU (Liu et al., 2021; 2022; Fraboni et al., 2024; Che et al., 2023). Gradient-based and model-transformation-based approaches directly modify model parameters through gradient ascent, subspace projection, regularization, or conflict-aware optimization (Halimi et al., 2022; Wu et al., 2022; Li et al., 2023; Pan et al., 2025; Gu et al., 2024). Other lines of work investigate selective forgetting, certified data removal, and more general unlearning formulations in centralized models (Guo et al., 2020; Golatkar et al., 2020; Neel et al., 2021; Kurmanji et al., 2023; Zhao et al., 2024). Recent empirical and survey studies further compare FU methods in terms of forgetting effectiveness, residual utility, computational cost, and required server-side information (Nguyen et al., 2024; Romandini et al., 2025). Despite their different implementations, these methods primarily focus on unlearning efficiency and model utility. They generally assume that client-issued requests identify valid training data and that client-controlled information used for reconstruction or calibration is trustworthy.

Adversarial Attacks through Federated Unlearning. Recent work has shown that the unlearning interface itself can become an attack surface (Huang et al., 2024; Hu et al., 2024). *Unlearn and Burn* demonstrates that insufficiently validated unlearning requests can be exploited to damage model performance (Huang et al., 2025). In the federated setting, FedMUA uses influence-function analysis to identify influential samples and constructs malicious feature unlearning requests to alter predictions on selected samples (Chen et al., 2025). Although effective, its influence estimation relies on second-order information and can incur substantial computational and memory costs, particularly for large models or multiple targets. BadFU introduces backdoor and camouflage samples during federated training and later requests removal of the camouflage samples to expose the hidden malicious behavior (Lu et al., 2025). This design relies on a residual backdoor component after unlearning and remains closely tied to the attacker's local data distribution. More broadly, prior work has shown that client-side updates can leak training information or be manipulated to compromise federated models (Zhu et al., 2019; Geiping et al., 2020; Fang et al., 2020; Li et al., 2022; Xie et al., 2025). These studies, however, primarily target the training or aggregation process rather than the unlearning interface itself.

The limitations of existing approaches motivate an attack that is lightweight, less dependent on local data availability, and able to coordinate data-dependent influences across training and unlearning. BURST addresses this gap by using public auxiliary data to construct a target-support set and an interference set. It establishes their complementary influences during federated training and coordinates Support Removal with Interference Enhancement during unlearning. This design avoids Hessian-based influence estimation, reduces dependence on the malicious client's local data, and enables BURST to turn previously correct predictions into post-unlearning misclassifications while largely preserving global model utility.

3 Preliminaries

3.1 Federated Learning and Federated Unlearning

We consider a federated learning (FL) system with K clients indexed by $\mathcal{K} = \{1, \dots, K\}$. Client k owns a local dataset $\mathcal{D}_k = \{(x_{k,n}, y_{k,n})\}_{n=1}^{N_k}$, and the overall training dataset is $\mathcal{D} = \bigcup_{k=1}^{K} \mathcal{D}_k$. Let $w \in \mathbb{R}^d$ denote the global model parameters and let $\ell(w; x, y)$ denote the loss on an example (x, y). The local objective of client k is $F_k(w) = \frac{1}{N_k} \sum_{(x,y)\in\mathcal{D}_k} \ell(w; x, y)$. The standard FL objective is

$$\min_w F(w) = \sum_{k=1}^{K} p_k F_k(w), \tag{1}$$

where $p_k = N_k / \sum_{j=1}^{K} N_j$ is the aggregation weight of client k. During each communication round, participating clients optimize their local objectives and send updates to the server, which aggregates them to obtain the next global model. We denote the model obtained after T rounds by $w_T = \mathrm{Train}_{\mathrm{FL}}(\mathcal{D}_1, \ldots, \mathcal{D}_K)$.

Federated unlearning (FU) aims to remove the influence of a designated forget set $\mathcal{F} \subseteq \mathcal{D}$ from a trained global model. An unlearning request is represented by $r = (\mathcal{F}, q)$, where q contains protocol-specific information required by the FU procedure. Given the trained model w_T and request r, an FU algorithm $\mathcal{U}$ produces the unlearned model

$$w^- = \mathcal{U}(w_T, r). \tag{2}$$

The ideal reference is the model retrained from scratch on $\mathcal{D} \setminus \mathcal{F}$, while practical FU methods seek to approximate this reference without repeating the complete FL procedure. We study whether the FU interface can be manipulated to alter predictions on selected samples while preserving the overall behavior of the unlearned model.

3.2 Threat Model

We consider an honest server and a set of participating clients, among which one or more clients may be malicious. The adversary can influence the local training data and optimization process of the malicious clients and can submit unlearning requests through the standard FU interface.

Adversary's goal. For each attack instance, the adversary selects a target sample $z_t = (x_t, y_t)$, where x_t is the target input and y_t is its ground-truth label. The target is correctly classified before the malicious unlearning operation. Let $f_w(x)$ denote the model prediction for input x. The attack aims to satisfy

$$f_{\widetilde{w}_T}(x_t) = y_t, \qquad f_{\widetilde{w}^-}(x_t) \neq y_t, \tag{3}$$

where $\widetilde{w}_T$ and $\widetilde{w}^-$ denote the models before and after the malicious FU procedure, respectively. Thus, the adversary seeks to turn an originally correct prediction into a misclassification.

Threat model. Let $\mathcal{M} \subseteq \mathcal{K}$ denote the set of malicious clients. A malicious client may manipulate its local training procedure, incorporate attacker-selected samples into its local data, and issue an unlearning request. When the FU protocol involves client-side reconstruction or calibration, the malicious client may also control the data or updates contributed during that stage. The server and honest clients follow the prescribed FL and FU protocols.

The adversary is assumed to know the selected target sample (x_t, y_t), while not requiring access to the complete private datasets of honest clients. The attacker additionally has access to a public auxiliary dataset $\mathcal{D}_{\mathrm{aux}}$, which is separate from the original FL training data, i.e., $\mathcal{D}_{\mathrm{aux}} \cap \mathcal{D} = \emptyset$. From this dataset, the adversary constructs a target-support set $\mathcal{S}$ and an interference set $\mathcal{I}$ satisfying $\mathcal{S}, \mathcal{I} \subseteq \mathcal{D}_{\mathrm{aux}}$ and $\mathcal{S} \cap \mathcal{I} = \emptyset$. These sets are incorporated into the attacker's local training and unlearning workflow, while their specific construction and usage are described in Section 4.

4 Methodology

We present BURST, an attack framework that exploits the interaction between federated training and federated unlearning. The framework uses public auxiliary data to establish controllable influence on selected target samples and manipulates this influence through a malicious unlearning request. The attack is designed to preserve the target's correct prediction before unlearning and induce misclassification after unlearning.

4.1 Overview

The overall workflow of BURST is illustrated in Figure 2. At a high level, BURST enables a malicious client to manipulate its local training process and subsequently submit an unlearning request through the standard FU interface. The attack unfolds in the following four steps:

- **Attack-set construction.** Given a target sample (x_t, y_t), the adversary uses public auxiliary data to construct a target-support set $\mathcal{S}$ and an interference set $\mathcal{I}$. The support set contains

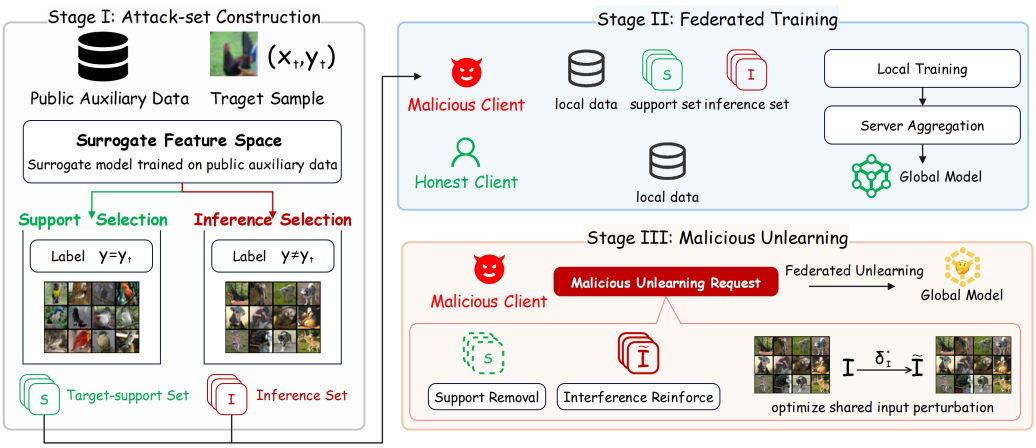


Figure 2: Overview of BURST. The attack proceeds through three stages: constructing complementary support and interference sets, establishing influences during federated training, and coordinating Support Removal with Interference Enhancement during unlearning to alter target prediction.

samples selected to reinforce the model's prediction of the target's ground-truth class, whereas the interference set contains samples associated with competing classes. The two sets provide complementary sources of influence for the subsequent training and unlearning stages.

- **Influence establishment.** The adversary incorporates $\mathcal{S}$ and $\mathcal{I}$ into its local training data and participates in federated training through the standard FL protocol. This stage establishes their respective influences on the target while maintaining its correct prediction before unlearning.
- **Support removal.** After federated training, the adversary submits a malicious request to unlearn $\mathcal{S}$. Removing these samples weakens the learned support for the target's ground-truth class.
- **Interference enhancement.** During the unlearning procedure, the adversary strengthens the influence of $\mathcal{I}$ through protocol-specific recalibration. Support removal and interference enhancement jointly shift the balance between the ground-truth and competing classes, potentially turning the target's previously correct prediction into a misclassification.

Through this coordinated procedure, BURST alters the prediction of a selected sample without modifying the target input at inference time or requiring the model to produce a predetermined incorrect label. The following subsections describe the construction of the two attack sets and their roles during federated training and unlearning.

4.2 Attack-Set Construction from Public Auxiliary Data

BURST constructs its attack sets from a public auxiliary dataset $\mathcal{D}_{\text{aux}}$, rather than restricting the selection to the malicious client's original local data. This additional data source provides a larger candidate pool from which the adversary can identify samples with different influence to the target.

Given a target sample $z_t = (x_t, y_t)$, the adversary constructs two disjoint sets: a target-support set $\mathcal{S}$ that reinforces the target's ground-truth class and an interference set $\mathcal{I}$ that provides influence from a competing class. We first divide the auxiliary data into a support candidate pool $\mathcal{C}_{\text{sup}} = \{(x, y) \in \mathcal{D}_{\text{aux}} \mid y = y_t\}$ and an interference candidate pool $\mathcal{C}_{\text{int}} = \{(x, y) \in \mathcal{D}_{\text{aux}} \mid y \neq y_t\}$. The $\mathcal{C}_{\text{sup}}$ contains samples associated with the target's ground-truth class, whereas the $\mathcal{C}_{\text{int}}$ contains samples from competing classes.

Feature-space retrieval. Before federated training, the adversary trains a surrogate model on $\mathcal{D}_{\text{aux}}$ to identify samples related to the target. This selection assumes that semantic relationships captured by the surrogate model transfer to the federated model. Let $h_\theta(\cdot)$ denote the representation extracted by a proxy model with parameters θ. We measure the similarity between an auxiliary

sample x and the target x_t using cosine similarity:

$$s(x, x_t) = \frac{h_\theta(x)^\top h_\theta(x_t)}{\|h_\theta(x)\|_2 \|h_\theta(x_t)\|_2}. \tag{4}$$

We use $\mathrm{TopK}(\mathcal{C}, s, k)$ to denote k samples in a candidate pool $\mathcal{C}$ with the highest similarity scores.

Target-support set. BURST selects the initial target-support set as $\mathcal{S}_0 = \mathrm{TopK}(\mathcal{C}_{\mathrm{sup}}, s, p_s)$, where p_s denotes the number of selected support samples. To strengthen their support for the target prediction, BURST optimizes a shared perturbation δ_s that moves the selected samples toward the target representation:

$$\delta_s^* = \arg\min_{\|\delta_s\|_\infty \le \epsilon_s} \frac{1}{|\mathcal{S}_0|} \sum_{(x,y)\in\mathcal{S}_0} \|h_\theta(\mathrm{clip}(x + \delta_s)) - h_\theta(x_t)\|_2^2 , \tag{5}$$

where ϵ_s constrains the perturbation magnitude and $\mathrm{clip}(\cdot)$ projects the perturbed inputs onto the valid input range. The resulting target-support set is

$$\mathcal{S} = \{(\mathrm{clip}(x + \delta_s^*), y_t) \,|\, (x, y) \in \mathcal{S}_0\} . \tag{6}$$

The samples in $\mathcal{S}$ are therefore aligned with the target in feature space and carry its ground-truth label, allowing them to reinforce the model's correct prediction during federated training.

Interference set. To avoid combining samples from multiple competing classes whose effects may partially cancel one another, BURST constructs a class-coherent interference set. We first retrieve an interference neighborhood $\mathcal{N}_{\mathrm{int}} = \mathrm{TopK}(\mathcal{C}_{\mathrm{int}}, s, k_i)$. The competing class used for the interference set is selected as the most frequent class in this neighborhood:

$$c^* = \arg\max_{c \neq y_t} \sum_{(x,y)\in\mathcal{N}_{\mathrm{int}}} \mathbb{I}[y = c]. \tag{7}$$

We then retain the p_i samples associated with c^* that are most similar to the target through $\mathcal{I} = \mathrm{TopK}(\{(x, y) \in \mathcal{N}_{\mathrm{int}} \mid y = c^*\}, s, p_i)$, where $p_i \le k_i$. All samples in $\mathcal{I}$ share the same competing-class label, which provides a consistent source of adversarial influence. The selected interference samples are incorporated into federated training in their initial form, while their influence is further enhanced during malicious unlearning as described in Section 4.3.

4.3 Malicious Unlearning

After constructing the target-support set and the interference set, the adversary incorporates them into the local training data of malicious clients. For a malicious client $k \in \mathcal{M}$, the effective local dataset is represented as $\widetilde{\mathcal{D}}_k = \mathcal{D}_k \cup \mathcal{S}_k \cup \mathcal{I}_k$, where $\mathcal{S}_k \subseteq \mathcal{S}$ and $\mathcal{I}_k \subseteq \mathcal{I}$ denote the samples assigned to client k. The support samples use the target label y_t, whereas the interference samples retain their selected competing-class label c^*. The malicious clients then participate in federated training through the standard protocol, allowing the two sets to establish their respective influences before unlearning.

After the training stage, the adversary performs two coordinated operations: Support Removal and Interference Enhancement.

Support Removal. The adversary submits an unlearning request for the target-support set $\mathcal{S}$. This request removes the contribution of the support samples from the model reconstruction process and weakens the evidence supporting the target's ground-truth class.

Interference Enhancement. The adversary enhances the influence of the interference set by applying a shared bounded perturbation to its samples. Let $\delta_{\mathcal{I}}$ denote the perturbation. The enhanced interference set is defined as

$$\widetilde{\mathcal{I}} = \{(\mathrm{clip}(x + \delta_{\mathcal{I}}), c^*) \,|\, (x, c^*) \in \mathcal{I}\} . \tag{8}$$

The perturbation is optimized to reduce the feature distance between the interference samples and the target:

$$\delta_{\mathcal{I}}^{*} = \arg \min_{\|\delta_{\mathcal{I}}\|_{\infty} \leq \epsilon_{\mathcal{I}}} \frac{1}{|\mathcal{I}|} \sum_{(x,c^{*}) \in \mathcal{I}} \|h_{\theta}\left(\text{clip}(x + \delta_{\mathcal{I}})\right) - h_{\theta}(x_t)\|_2^2 . \tag{9}$$

The perturbed samples retain their competing-class label c^* and are introduced during the protocol-specific recalibration or model reconstruction stage.

Joint Effect. Support Removal and Interference Enhancement affect complementary sides of the target prediction. Support Removal weakens the evidence for the ground-truth class, whereas Interference Enhancement increases the relative influence of a competing class. Their coordinated application changes the balance of class evidence around the target sample and can turn previously correct prediction into misclassification. Algorithm 1 summarizes BURST procedure in Appendix B.

5 Experiments

We evaluate BURST from three perspectives. We first compare it with existing malicious unlearning attacks. We then conduct ablation studies to quantify the individual contributions of Support Removal and Interference Enhancement. Finally, we measure the computational cost of the attack.

5.1 Experimental Setup

Datasets and public auxiliary data. We evaluate all methods on CIFAR-10 and MNIST. CIFAR-10 contains 50,000 training images and 10,000 test images from ten classes, while MNIST contains 60,000 training images and 10,000 test images of handwritten digits. For the public auxiliary data, we use STL-10 for CIFAR-10 and QMNIST for MNIST. These datasets are kept separate from the federated training data and are used only to construct the target-support and interference sets.

Baselines. We compare BURST with internal variants and existing malicious-unlearning attacks. **BURST** is the complete framework that removes the target-support set and enhances the interference set during unlearning. **Random Auxiliary Selection (RAS)** retains the same support-removal and interference-enhancement procedure as BURST but randomly selects auxiliary samples within the corresponding semantic groups. **Unoptimized Auxiliary Retrieval (UAR)** retains feature-similarity-based retrieval but removes the perturbation-optimization step and uses the retrieved samples in their original form. **Benign FU** performs a normal unlearning request using a removal set with a matched effective size and serves as a utility reference. **FedMUA** (Chen et al., 2025) identifies samples influential to the target prediction and constructs malicious feature-unlearning requests. **BadFU** (Lu et al., 2025) uses a hidden-poisoning strategy that introduces poison and camouflage samples during training and removes the camouflage samples during unlearning. All methods use the same target samples, client partitions and training schedule. Attack methods are evaluated under matched budgets for injected, removed, and modified samples.

Evaluation metrics. We evaluate attack effectiveness and global model utility using three primary metrics. **Attack success rate (ASR)** is the proportion of target samples that are correctly classified before unlearning but become misclassified after unlearning. **Mean confidence gap** is the average difference between the confidence assigned to the predicted label and that assigned to the ground-truth label among successfully misclassified targets. **Global accuracy drop** is the decrease in accuracy from the model before unlearning to the model after unlearning. A smaller value indicates better preservation of global model utility.

For the efficiency analysis, we report attack-construction time. The reported time excludes federated training, federated unlearning and model evaluation.

Implementation details. Unless otherwise stated, the federated system contains 10 clients, including 2 malicious clients. The data are partitioned according to a Dirichlet distribution with concentration parameter $\alpha = 1$ in the main experiments. The models are trained for 50 communication rounds on CIFAR-10 and 20 rounds on MNIST, with one local epoch per round. FedEraser is used for the federated unlearning procedure.

Table 1: Overall attack effectiveness on MNIST and CIFAR-10.

Dataset	Method	ASR (%)	Acc. Drop (p.p.)	Mean Confidence Gap (p.p.)
MNIST	Benign FU	10.00	2.67	37.12
	BadFU	45.00	2.29	33.11
	FedMUA	34.29	2.95	37.91
	RAS	50.00	2.27	36.65
	UAR	58.82	2.33	45.85
	Ours	**63.64**	**2.71**	**37.84**
CIFAR-10	Benign FU	5.00	5.80	35.18
	BadFU	40.00	5.30	30.20
	FedMUA	60.00	6.51	86.38
	RAS	17.14	5.45	21.97
	UAR	25.71	5.54	21.08
	Ours	**91.43**	**5.66**	**30.74**

All methods use the same training schedule, FU configuration, target samples, and random seeds within each experimental setting. Targets are selected only when they are correctly classified before unlearning. We report mean performance over the evaluated target samples. Attack-specific budgets and perturbation bounds are kept identical across methods.

5.2 Overall Attack Effectiveness

Table 1 compares BURST with Benign FU, BadFU, FedMUA, RAS, and UAR on MNIST and CIFAR-10. We report the ASR, global accuracy drop, and mean confidence gap.

BURST achieves the highest ASR on both datasets. On CIFAR-10, BURST reaches an ASR of 91.43%, substantially exceeding FedMUA and BadFU, which achieve 60.00% and 40.00%, respectively. RAS and UAR achieve lower ASRs of 17.14% and 25.71%. These two variants retain the same support-removal and interference-enhancement procedure as BURST but isolate the effects of auxiliary-sample selection and perturbation optimization. The lower ASRs of RAS and UAR indicate that random auxiliary selection and feature retrieval without perturbation optimization are insufficient to reproduce the full post-unlearning attack effect.

BURST also produces a mean confidence gap of 30.74 percentage points on CIFAR-10, indicating that the incorrect predictions are made with substantially higher confidence than the ground-truth labels. Its global accuracy drop is 5.66 percentage points. These results show that BURST causes substantial sample-level prediction damage while largely preserving global model utility.

MNIST exhibits the same overall pattern. BURST reaches an ASR of 63.64%, whereas FedMUA, BadFU, RAS, and UAR achieve considerably lower ASRs of 34.29%, 45.00%, 50.00%, and 58.82%, respectively. These results confirm the effectiveness of BURST across both datasets.

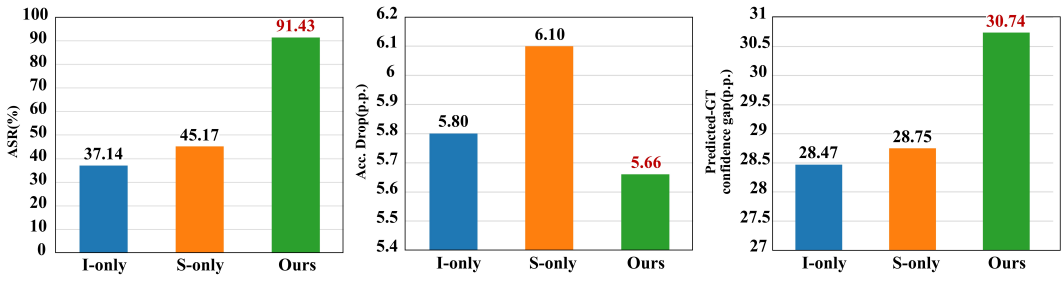


Figure 3: Ablation study of the two mechanisms in BURST on CIFAR-10.

5.3 Ablation Study: Support Removal and Interference Enhancement

We conduct an ablation study to examine the contributions of Support Removal and Interference Enhancement. We compare BURST with two single-mechanism variants. I-only performs interference enhancement, whereas S-only removes the target-support set without enhancing the interference set. All variants use the same target samples, client partition, training schedule, unlearning configuration.

As shown in Figure 3, BURST achieves the highest attack success rate. Its ASR reaches 91.43%, compared with 45.17% for S-only and 37.14% for I-only. The confidence-gap results show the same trend. BURST produces a mean confidence gap of 30.74 percentage points, compared with 28.75 percentage points for S-only and 28.47 percentage points for I-only. These results indicate that removing target-class support alone or strengthening competing-class influence alone is generally insufficient to reliably change the target prediction. Coordinating the two operations substantially strengthens the post-unlearning attack effect.

The utility results further show that the higher attack effectiveness of BURST does not result from greater global model degradation. BURST reduces global accuracy by 5.66 percentage points, while S-only and I-only cause drops of 6.10 and 5.80 percentage points, respectively. BURST therefore achieves the strongest sample-level manipulation while preserving comparable or better global utility. These findings support the central design of BURST.

5.4 Efficiency Analysis

We compare the attack-construction overhead of BURST, BadFU, and FedMUA under the same MNIST setting. The measurement focuses on the attack-specific construction stage and excludes federated training, federated unlearning, model evaluation, and checkpoint serialization.

Table 2: Runtime breakdown of attack construction on MNIST.

Method	Component	Time (s)
BURST	Support optimization	0.628
	Interference optimization	0.347
	Total	**1.041**
BadFU	Poison gradient matching	0.490
	Camouflage gradient matching	0.546
	Total	**1.036**
FedMUA	Hessian construction	663.64
	Hessian inversion	0.33
	Influence-score computation	16.77
	Total	**681.35**

As shown in Table 2, BURST requires only 1.041 seconds to construct the attack. Its support-set and interference-set optimizations take 0.628 and 0.347 seconds, respectively. BadFU has a comparable construction time of 1.036 seconds, including 0.490 seconds for poison-gradient matching and 0.546 seconds for camouflage-gradient matching. FedMUA is substantially more expensive, requiring 681.35 seconds in total. Hessian construction accounts for 663.64 seconds, while Hessian inversion and influence-score computation take 0.33 and 16.77 seconds, respectively. These results show that BURST has a construction cost comparable to BadFU while avoiding the expensive second-order influence analysis required by FedMUA. The efficiency advantage becomes particularly relevant when attack construction must be repeated for multiple targets or malicious clients.

6 Conclusion

We present BURST, an attack framework that exposes a security weakness in federated unlearning. By coordinating auxiliary samples introduced during federated training with a malicious unlearning request, BURST can alter predictions on selected samples while retaining much of the global model's utility. Our evaluation across datasets, data partitions, and unlearning methods shows that sample-level failures can remain hidden when FU is assessed mainly through aggregate accuracy. These findings indicate that unlearning requests and the data used to reconstruct unlearned models should be treated as security-critical inputs and jointly audited before deployment.

Future Work. Future work will investigate request verification and data-provenance mechanisms that can detect coordinated attacks, extend the analysis to larger models and multimodal tasks, and develop certified or defense-aware FU procedures with stronger guarantees against malicious clients.

AI USE STATEMENT

We used generative AI tools to assist with language polishing and manuscript organization. All AI-assisted content was reviewed and verified by the authors, who take full responsibility for the final content, claims, and artifacts presented in this work.

ETHICS STATEMENT

This work studies security vulnerabilities in federated unlearning and is intended to support the evaluation and strengthening of unlearning systems. The experiments use benchmark datasets and do not involve human subjects or personally identifiable information. We report the attack methodology to facilitate responsible auditing and the development of defenses, and we encourage its use only for authorized security evaluation.

REPRODUCIBILITY STATEMENT

We provide the implementation, experimental configurations, evaluation scripts, and supplementary details needed to reproduce our results through an anonymous code repository. The main paper describes the threat model, attack-set construction, malicious unlearning procedure, evaluation metrics, and experimental settings, while additional implementation details and results are provided in the supplementary material.

REFERENCES

Lucas Bourtoule, Varun Chandrasegaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, and David Zhang. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy*, pp. 141–159, 2021.

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

Tianshi Che et al. Fast federated machine unlearning with nonlinear functional theory. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, 2023.

Jian Chen et al. Fedmua: Exploring the vulnerabilities of federated learning to malicious unlearning attacks. *IEEE Transactions on Information Forensics and Security*, 20:1665–1678, 2025. doi: 10.1109/TIFS.2025.3531141.

European Parliament and Council of the European Union. Regulation (eu) 2016/679 (general data protection regulation). Official Journal of the European Union, 2016.

Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local model poisoning attacks to byzantine-robust federated learning. In *29th USENIX Security Symposium (USENIX Security 20)*, pp. 1605–1622. USENIX Association, 2020. URL https://www.usenix.org/conference/usenixsecurity20/presentation/fang.

Yann Fraboni, Martin Van Waerebeke, Kevin Scaman, Richard Vidal, Laetitia Kameni, and Marco Lorenzi. SIFU: Sequential informed federated unlearning for efficient and provable client unlearning in federated optimization. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*, volume 238, pp. 3457–3465, 2024.

Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients – how easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://papers.neurips.cc/paper/2020/hash/c4ede56bbd98819ae6112b20ac6bf145-Abstract.html.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. doi: 10.1109/CVPR42600.2020.00932.

Hanlin Gu, Win Kent Ong, Chee Seng Chan, and Lixin Fan. Ferrari: Federated feature unlearning via optimizing feature sensitivity. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens van der Maaten. Certified data removal from machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 3832–3842, 2020.

Anisa Halimi, Swanand Kadhe, and Dhananjay Rawat. Federated unlearning: How to efficiently erase a client in fl? In *Proceedings of the First Workshop on Unlearning and Privacy*, 2022.

Hongsheng Hu, Shuo Wang, Jiamin Chang, Haonan Zhong, Ruoxi Sun, Shuang Hao, Haojin Zhu, and Minhui Xue. A duty to forget, a right to be assured? exposing vulnerabilities in machine unlearning services. In *Network and Distributed System Security Symposium*, 2024. doi: 10.14722/ndss.2024.24252.

Yiming Huang et al. Unlearn and burn: Adversarial machine unlearning requests destroy model accuracy. In *International Conference on Learning Representations*, 2025.

Zirui Huang, Yunlong Mao, and Sheng Zhong. UBA-Inf: Unlearning activated backdoor attack with Influence-Driven camouflage. In *33rd USENIX Security Symposium*, pp. 4211–4228, Philadelphia, PA, August 2024. USENIX Association. URL https://www.usenix.org/conference/usenixsecurity24/presentation/huang-zirui.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Borja Balle, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2): 1–210, 2021.

Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Guanghao Li, Li Shen, Yan Sun, Yue Hu, Han Hu, and Dacheng Tao. Subspace based federated unlearning. *arXiv preprint arXiv:2302.12448*, 2023.

Henger Li, Xiaolin Sun, and Zizhan Zheng. Learning to attack federated learning: A model-based reinforcement learning attack framework. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Gaoyang Liu, Xian 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*, pp. 1–10, 2021.

Yi Liu, Shaojie Hua, et al. The right to be forgotten in federated learning: An efficient realization with rapid retraining. In *IEEE INFOCOM*, 2022.

Bingguang Lu, Hongsheng Hu, Yuantian Miao, Shaleeza Sohail, Chaoxiang He, Shuo Wang, and Xiao Chen. Badfu: Backdoor federated learning through adversarial machine unlearning. In *Proceedings of the 28th International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2025)*, pp. 773–788, 2025. doi: 10.1109/RAID67961.2025.00020.

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

Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. In *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pp. 931–962. PMLR, 2021. URL https://proceedings.mlr.press/v132/neel21a.html.

Thai-Hung Nguyen, Hong-Phuc Vu, Dung Thuy Nguyen, Tuan Minh Nguyen, Khoa D. Doan, and Kok-Seng Wong. Empirical study of federated unlearning: Efficiency and effectiveness. In *Proceedings of the 15th Asian Conference on Machine Learning*, volume 222, pp. 959–974, 2024.

Zibin Pan, Zhichao Wang, Chi Li, Kaiyan Zheng, Boqi Wang, Xiaoying Tang, and Junhua Zhao. Federated unlearning with gradient descent and conflict mitigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Nicolò Romandini, Alessio Mora, Carlo Mazzocca, Rebecca Montanari, and Paolo Bellavista. Federated unlearning: A survey on methods, design guidelines, and evaluation metrics. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

Leijie Wu, Song Yang, et al. Federated unlearning: Guarantee the right of clients to forget. *IEEE Network*, 2022.

Yueqi Xie, Minghong Fang, and Neil Zhenqiang Gong. Model poisoning attacks to federated learning via multi-round consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15454–15463, 2025. doi: 10.1109/CVPR52734.2025.01440.

Kairan Zhao, Meghdad Kurmanji, George-Octavian Barbulescu, Eleni Triantafillou, and Peter Triantafillou. What makes unlearning hard and what to do about it. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, volume 32, pp. 14747–14756, 2019. URL `https://papers.neurips.cc/paper/9617-deep-leakage-from-gradients`.

A Detailed Computational and Memory Complexity Analysis

We analyze the attack-specific computational and memory overhead of BURST, BadFU, and FedMUA. The common federated-training and unlearning costs are excluded because they are shared across the compared methods. Let m denote the number of malicious clients. For BURST, let n_s and n_i denote the sizes of the support and interference candidate pools, respectively, and let p_s and p_i denote the numbers of selected support and interference samples. Let T_{B} denote the number of perturbation-optimization iterations. We use D for the feature dimension, d_x for the input dimension, P for the number of parameters in the complete model, and Q for the number of parameters in the classification head. Finally, let C_{f}, C_{b}, and C_2 denote the costs of one forward pass, one backward pass, and one second-order or nested-gradient operation, respectively.

BURST. BURST first extracts representations for the two candidate pools and retrieves samples according to their similarity to the target representation. The feature-extraction cost is

$$O\left((n_s + n_i)C_{\mathrm{f}}\right).$$

Computing similarities in the D-dimensional feature space costs $O((n_s + n_i)D)$. If top-k retrieval is implemented using a partial sorting procedure, its additional cost is approximately $O((n_s + n_i)\log k)$. The two selected sets are then optimized through shared input perturbations. With T_{B} optimization iterations, the perturbation-optimization cost is bounded by

$$O\left(T_{\mathrm{B}}(p_s + p_i)C_2\right).$$

Therefore, the attack-specific time complexity of BURST is

$$\boxed{O_{\mathrm{BURST}} = O\left((n_s + n_i)C_{\mathrm{f}}\right) + O\left((n_s + n_i)(D + \log k)\right) + O\left(T_{\mathrm{B}}(p_s + p_i)C_2\right).}$$

Its additional memory requirement is

$$\boxed{S_{\mathrm{BURST}} = O\left((n_s + n_i)D + (p_s + p_i)d_x\right),}$$

apart from the model parameters and the workspace required by automatic differentiation. Thus, BURST grows linearly with the number of auxiliary candidates and selected samples. It does not explicitly construct or invert a Hessian matrix.

BadFU. BadFU optimizes two groups of samples, namely poison and camouflage samples, through gradient matching. Let p_{poison} and p_{cam} denote the sizes of the two groups, and let T_{Bf} be the number of gradient-matching iterations. Since the optimization is restricted to the classification head, the per-iteration second-order cost depends on Q rather than the complete model size P. The attack-specific time complexity can be written as

$$\boxed{O_{\text{BadFU}} = O\left(T_{\text{Bf}}\left[(p_{\text{poison}} + p_{\text{cam}})C_{\text{f}} + C_{2,Q}\right]\right),}$$

where $C_{2,Q}$ denotes the cost of the nested-gradient computation on the classification head. Its additional memory requirement is approximately

$$\boxed{S_{\text{BadFU}} = O\left((p_{\text{poison}} + p_{\text{cam}})d_x + Q\right).}$$

BadFU therefore avoids the explicit full-model Hessian used by FedMUA, but still incurs iterative gradient-matching overhead.

FedMUA. FedMUA identifies influential samples by constructing an explicit Hessian for each malicious client. For malicious client c, let h_c denote the number of samples used to construct the Hessian and let r_c denote the number of candidate samples scored by the influence function. The Hessian is a dense matrix in $\mathbb{R}^{P\times P}$, so its storage cost is

$$O(P^2).$$

Let $C_{\text{H}}(h_c, P)$ denote the cost of constructing this Hessian. The subsequent dense matrix inversion costs $O(P^3)$, while scoring the candidate samples costs

$$O\left(r_c(C_{\text{b}} + P^2)\right),$$

because each candidate requires a gradient computation and a Hessian–vector multiplication. Summing over all malicious clients gives

$$\boxed{O_{\text{FedMUA}} = \sum_{c=1}^{m}\left[C_{\text{H}}(h_c, P) + O(P^3) + O\left(r_c(C_{\text{b}} + P^2)\right)\right].}$$

The peak memory requirement is

$$\boxed{S_{\text{FedMUA}} = O(P^2),}$$

because the Hessian and its inverse are represented as dense matrices. The implementation may construct and release the Hessian client by client, so the peak memory does not necessarily grow linearly with m, although the total runtime does.

Comparison. The dominant difference among the three methods is the dependence on the model parameter count. BURST and BadFU perform feature extraction or classification-head gradient matching and have no explicit full-model Hessian. In contrast, FedMUA incurs $O(P^2)$ memory for Hessian-related operations and a cubic $O(P^3)$ matrix-inversion term. Consequently, the computational gap becomes increasingly pronounced as the model size grows.

B Complete BURST Attack Procedure

Algorithm 1 summarizes the complete execution of BURST. The procedure consists of attack-set construction, malicious federated training, and coordinated unlearning. The support and interference sets are constructed from public auxiliary data before federated training, while the subsequent unlearning request combines support removal with interference enhancement.

Given a target sample (x_t, y_t), BURST first forms two candidate pools from the auxiliary dataset. The support pool contains samples with label y_t, whereas the interference pool excludes the target label. The operator s denotes the similarity measure induced by the surrogate feature space. BURST selects the p_s most similar samples from the support pool and optimizes a shared perturbation bounded by ϵ_s. The perturbed samples retain the target label y_t and form the target-support set $\mathcal{S}$.

Algorithm 1 Complete attack procedure of BURST.

Input: target sample (x_t, y_t), public auxiliary dataset $\mathcal{D}_{\text{aux}}$, malicious clients $\mathcal{M}$, FL procedure Train_{FL}, FU procedure $\mathcal{U}$, retrieval parameters (p_s, k_i, p_i), perturbation bounds (ϵ_s, ϵ_i)
$\mathcal{S} \leftarrow \textsc{SelectSupportSet}(\mathcal{D}_{\text{aux}}, x_t, y_t, p_s, \epsilon_s)$
$(\mathcal{I}, c^*) \leftarrow \textsc{SelectInterferenceSet}(\mathcal{D}_{\text{aux}}, x_t, y_t, k_i, p_i)$
for $k \in \mathcal{M}$ **do**
    $\widetilde{\mathcal{D}}_k \leftarrow \mathcal{D}_k \cup \mathcal{S}_k \cup \mathcal{I}_k$
end for
$\widetilde{w}_T \leftarrow \text{Train}_{\text{FL}}(\widetilde{\mathcal{D}}_1, \ldots, \widetilde{\mathcal{D}}_K)$
$\delta^*_{\mathcal{I}} \leftarrow \text{OptimizePerturbation}(\mathcal{I}, x_t, \epsilon_i)$
$\widetilde{\mathcal{I}} \leftarrow \{(\text{clip}(x + \delta^*_{\mathcal{I}}), c^*) \mid (x, c^*) \in \mathcal{I}\}$
$\widetilde{r} \leftarrow (\mathcal{S}, \widetilde{\mathcal{I}})$
$\widetilde{w}^- \leftarrow \mathcal{U}(\widetilde{w}_T, \widetilde{r})$
Return: $\widetilde{w}^-$

procedure $\textsc{SelectSupportSet}(\mathcal{D}_{\text{aux}}, x_t, y_t, p_s, \epsilon_s)$
    $\mathcal{C}_{\text{sup}} \leftarrow \{(x, y) \in \mathcal{D}_{\text{aux}} \mid y = y_t\}$
    $\mathcal{S}_0 \leftarrow \text{TopK}(\mathcal{C}_{\text{sup}}, s, p_s)$
    $\delta^*_{\mathcal{S}} \leftarrow \text{OptimizePerturbation}(\mathcal{S}_0, x_t, \epsilon_s)$
    $\mathcal{S} \leftarrow \{(\text{clip}(x + \delta^*_{\mathcal{S}}), y_t) \mid (x, y) \in \mathcal{S}_0\}$
    Return: $\mathcal{S}$
end procedure

procedure $\textsc{SelectInterferenceSet}(\mathcal{D}_{\text{aux}}, x_t, y_t, k_i, p_i)$
    $\mathcal{C}_{\text{int}} \leftarrow \{(x, y) \in \mathcal{D}_{\text{aux}} \mid y \neq y_t\}$
    $\mathcal{N}_{\text{int}} \leftarrow \text{TopK}(\mathcal{C}_{\text{int}}, s, k_i)$
    $c^* \leftarrow \arg\max_{c \neq y_t} \sum_{(x,y) \in \mathcal{N}_{\text{int}}} \mathbb{I}[y = c]$
    $\mathcal{I} \leftarrow \text{TopK}\left(\{(x, y) \in \mathcal{N}_{\text{int}} \mid y = c^*\}, s, p_i\right)$
    Return: $(\mathcal{I}, c^*)$
end procedure

The interference set is constructed in two stages. BURST first retrieves a neighborhood of k_i samples from the interference pool. It then identifies the most frequent competing class c^* in this neighborhood and retains the p_i samples associated with that class. This class-consistent selection prevents the effects of multiple competing classes from canceling one another. The resulting set $\mathcal{I}$ preserves the competing-class label c^*.

The selected sets are distributed among the malicious clients and appended to their local training data. The malicious clients then participate in federated training through the standard protocol, producing the model $\widetilde{w}_T$. After training, BURST optimizes a second shared perturbation for the interference set under the bound ϵ_i. The perturbed interference samples $\widetilde{\mathcal{I}}$ retain the competing label c^* and are used in the subsequent unlearning procedure.

Finally, the adversary submits the request $\widetilde{r} = (\mathcal{S}, \widetilde{\mathcal{I}})$. The FU procedure removes the influence associated with $\mathcal{S}$ and incorporates $\widetilde{\mathcal{I}}$ through its protocol-specific recalibration or reconstruction operation. The resulting model $\widetilde{w}^-$ is returned as the post-unlearning model. The two operations jointly weaken evidence for the target's ground-truth class and amplify the influence of the selected competing class, potentially converting the target's initially correct prediction into a misclassification.

C Datasets, Baselines, and Additional Experimental Details

This appendix provides additional details about the datasets, baseline implementations, federated training configuration, target selection, and attack evaluation protocol. Unless otherwise specified,

all methods are evaluated under the same data partition, target samples, client assignment, training schedule, and federated unlearning configuration.

C.1 Datasets and Data Preparation

We evaluate the proposed attack on MNIST and CIFAR-10, two commonly used image-classification benchmarks with different input resolutions and visual complexity. MNIST contains handwritten grayscale digits from ten classes, whereas CIFAR-10 contains color images from ten object classes. The two datasets allow us to examine whether the attack remains effective across relatively simple and more visually complex classification settings.

For each dataset, the original training set is partitioned among ten federated clients. The server does not access the raw local training examples during federated training. To simulate statistically heterogeneous client data, we partition the training set according to a Dirichlet distribution. Unless otherwise stated, the concentration parameter is set to $\alpha = 1$. The same partition is used by all compared methods within each experimental setting to ensure that differences in attack performance are not caused by different client-side data distributions.

The public auxiliary dataset used by BURST is separate from the federated training data. It is used only for constructing the target-support and interference sets and is not treated as part of the original client training corpus. For each target sample, auxiliary examples are divided according to their labels. Samples with the target label form the support candidates, while samples with labels different from the target label form the interference candidates. The auxiliary examples selected by the attack are assigned to malicious clients according to the experimental protocol described below.

All input images are normalized using the dataset-specific preprocessing adopted by the corresponding classifier. The same preprocessing is applied to the target samples, auxiliary samples, and perturbed samples. When an input perturbation is applied, the perturbed image is clipped to the valid input range before being used for training or unlearning.

C.2 Federated Training Configuration

Unless otherwise specified, the federated system contains ten clients, including two malicious clients and eight honest clients. Honest clients train on their assigned local data, while malicious clients augment their local datasets with the attack-specific samples. The server follows the standard federated aggregation procedure. Each communication round consists of one local training epoch per participating client, followed by server-side aggregation.

For MNIST, federated training runs for 20 communication rounds. For CIFAR-10, training runs for 50 communication rounds. The local batch size is 128 for both datasets. The remaining optimization settings, including the optimizer, learning rate, and model architecture, are kept fixed across methods on the same dataset. In particular, all attack methods use the same base classifier and the same number of communication rounds as the benign federated-learning model.

For a malicious client k, let $\mathcal{D}_k$ denote its original local dataset. The effective local dataset used during attack construction is represented as

$$\widetilde{\mathcal{D}}_k = \mathcal{D}_k \cup \mathcal{S}_k \cup \mathcal{I}_k,$$

where $\mathcal{S}_k$ and $\mathcal{I}_k$ denote the support and interference samples assigned to client k, respectively. The assignment of auxiliary samples is fixed before federated training and is kept identical across methods whenever the corresponding sample groups are used.

Unless otherwise stated, FedEraser is used as the underlying federated unlearning procedure. This choice provides a common reconstruction-based FU interface for comparing the proposed attack with the ablation methods and malicious baselines. The same unlearning budget and protocol configuration are used for all methods within each experimental setting.

C.3 Baseline Implementations

We compare BURST with benign unlearning, existing malicious unlearning attacks, and internal ablations.

Benign FU. Benign FU submits a normal unlearning request without attempting to manipulate the prediction of a selected target. Its removal set is matched to the effective size of the removal set used by the attack methods. This baseline measures the ordinary utility change caused by federated unlearning and provides a reference for distinguishing attack-induced degradation from normal unlearning behavior.

BadFU. BadFU is implemented following the hidden-poisoning protocol described in the original method. The attacker introduces poison and camouflage samples during federated training and later requests the removal of the camouflage samples. The unlearning process thereby leaves the poison component as the residual malicious influence. We use the same malicious-client budget and the same target samples as BURST whenever possible.

FedMUA. FedMUA constructs malicious unlearning requests by identifying samples that are influential to the prediction of a target sample. We use the official influence-based procedure, including the computation of the client-side Hessian and the subsequent influence-score evaluation. The resulting samples are used to construct the malicious unlearning request under the same target and client configuration as the other methods.

Support Removal Only (S-only). S-only is an ablation of BURST that retains the target-support construction and the support-removal operation but does not construct interference set. It evaluates whether removing evidence associated with the target's ground-truth class is sufficient to induce a post-unlearning prediction change.

Interference Enhancement Only (I-only). I-only retains the interference-set construction and enhancement operation but does not construct the target-support set. It evaluates whether strengthening the competing-class influence alone can produce the desired sample-level prediction change.

Random Auxiliary Selection (RAS). RAS uses the same two-set training and unlearning procedure as BURST but replaces feature-based auxiliary-sample retrieval with random selection within the corresponding semantic groups. Samples with the target label are randomly selected for the support set, while samples associated with the selected competing class are randomly selected for the interference set. The perturbation optimization and the subsequent support-removal/interference-enhancement operations remain unchanged. This baseline isolates the contribution of precise feature-based retrieval.

Unoptimized Auxiliary Retrieval (UAR). UAR preserves the feature-space retrieval procedure of BURST but removes the input-perturbation optimization. The support and interference samples are selected according to their similarity to the target representation and are used in their original form. The same dual-mechanism unlearning procedure is then applied. This baseline measures the contribution of optimized input perturbations while keeping the retrieval strategy fixed.

For all methods, we use the same target samples, client partition, random seeds, training schedule, and applicable FU configuration. Attack-specific budgets are matched whenever possible, including the number of injected samples, removed samples, modified samples, and malicious clients. Differences in the reported results therefore primarily reflect the attack mechanism rather than differences in the federated-learning environment.

C.4 Target Selection and Evaluation Protocol

For each attack instance, we first train the federated model before unlearning and evaluate the target candidates using this pre-unlearning model. We retain only samples that are correctly classified before the unlearning stage. This condition ensures that a successful attack corresponds to changing an initially correct prediction into an incorrect one, rather than preserving an error that already existed before the attack.

Let $\mathcal{T}$ denote the set of retained target samples. The same target set is used by all methods under a given experimental setting. Each target is associated with its ground-truth label y_t, and the attacker is assumed to know the target label under the application-level threat model. The attacker does not require access to the complete local datasets owned by honest clients.

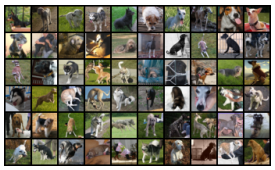
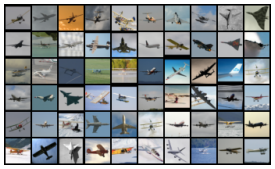
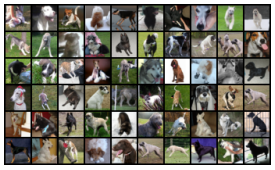
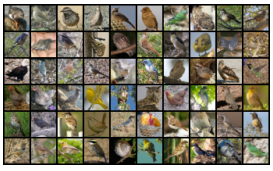
For BURST, the target-support set is constructed from auxiliary samples associated with y_t. The interference set is constructed from samples associated with a competing class selected from the retrieved auxiliary neighborhood. The two sets are incorporated into the local training data of malicious clients before federated training. After obtaining the trained model, the attacker submits an unlearning request for the support set and supplies the optimized interference samples to the protocol-specific recalibration or reconstruction stage.

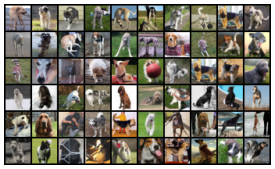
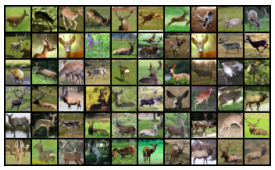
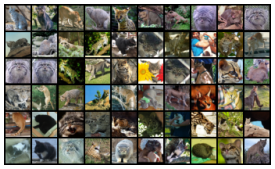
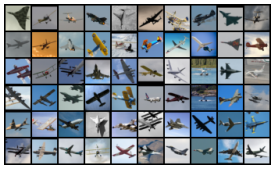
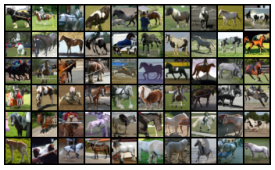
For reference, Table 3 visualizes a representative subset of 20 CIFAR-10 target samples together with their corresponding target-support and interference sets. The first column gives the dataset index of each displayed target sample, and the remaining columns show the auxiliary samples selected for the two complementary attack mechanisms.

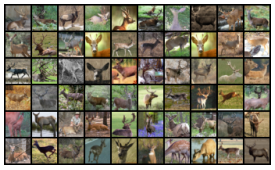
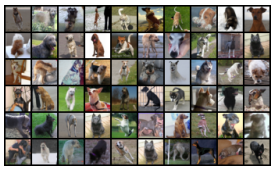
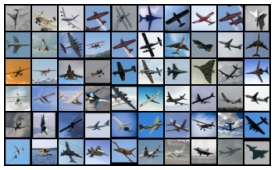
For each method, we record the prediction of the target sample before and after unlearning. A target is counted as successfully attacked if it is correctly classified before unlearning and incorrectly classified after unlearning. If a target is already misclassified before unlearning, the corresponding trial is excluded from the attack-success denominator. This rule is applied consistently to all methods.

We report the mean over repeated trials when multiple runs are available. Unless otherwise stated, all methods are evaluated on the same target instances and under the same random seeds. The main evaluation reports attack success rate, global model accuracy degradation, and the confidence gap between the predicted label and the ground-truth label. We additionally record attack-construction runtime in the efficiency study.

Table 3: Representative CIFAR-10 targets and the auxiliary sets.

Index	Target sample	Target-support set	Interference set
25			
70			
129			
149			
354			

Index	Target sample	Target-support set	Interference set
450			
623			
1039			
1051			
2114			
2685			
3146			
3495			
3725			

Index	Target sample	Target-support set	Interference set
3997			
5118			
5129			
5238			
5477			
5533			

C.5 Perturbation and Sample-Budget Control

The support and interference sets are constructed under fixed sample budgets. The number of candidate samples, retrieved samples, selected samples, and optimization steps is determined before each experiment and remains unchanged across methods whenever the corresponding operation is applicable. The perturbation bounds for support and interference samples are also fixed across runs.

For BURST, the support perturbation is optimized to align selected auxiliary examples with the target representation while preserving their target label. The interference perturbation is optimized to strengthen the competing-class influence used during the unlearning stage. RAS applies the same optimization procedure to randomly selected samples, whereas UAR uses the retrieved samples without perturbation optimization. S-only and I-only inherit the corresponding support or interference operation from BURST and disable the other mechanism.

These controls ensure that the comparisons distinguish the effects of sample selection, perturbation optimization, support removal, and interference enhancement. In particular, the comparison between BURST and RAS evaluates the role of feature-aligned retrieval, while the comparison between BURST and UAR evaluates the role of input perturbation optimization. The comparisons with S-only and I-only further reveal whether the attack requires the coordination of both unlearning mechanisms.

C.6 Reproducibility Details

All experiments use a fixed implementation of the base classifier and the same data preprocessing pipeline within each dataset. The random seed is fixed for data partitioning, target selection, auxiliary-sample selection, and attack optimization unless a specific experiment evaluates the effect of random selection. Model checkpoints are saved before and after unlearning to enable independent computation of prediction changes, accuracy degradation, and confidence statistics.

The attack-construction runtime is measured separately from federated training, federated unlearning, model evaluation, and checkpoint serialization. CUDA operations are synchronized before and after timing when GPU execution is used. For each method, we record the total construction time and, where applicable, the runtime of feature retrieval, perturbation optimization, gradient matching, Hessian construction, Hessian inversion, and influence-score computation. These measurements are used in the efficiency analysis reported in the main paper and the detailed complexity analysis provided in a later appendix section.

D Theoretical Feasibility Analysis

We provide a local analysis of the feasibility of BURST. Since practical federated-unlearning procedures are generally non-convex and protocol-dependent, we analyze the target prediction through a first-order approximation of its decision margin.

D.1 Margin-Based Formulation

Let $z_c(w, x)$ denote the logit of class c under model parameters w. For a target sample (x_t, y_t), let

$$c_t^\star = \arg\max_{c \neq y_t} z_c(\widetilde{w}_T, x_t)$$

be the strongest competing class before unlearning. We define the target margin as

$$M(w) = z_{y_t}(w, x_t) - z_{c_t^\star}(w, x_t).$$

The target is correctly classified before unlearning when $M(\widetilde{w}_T) > 0$, and becomes misclassified when $M(\widetilde{w}^-) \leq 0$.

Let Δw_{rem} denote the parameter change caused by removing the support set $\mathcal{S}$, and let Δw_{enh} denote the additional change caused by enhancing the interference set $\widetilde{\mathcal{I}}$. The total unlearning update is

$$\Delta w_{\mathrm{FU}} = \Delta w_{\mathrm{rem}} + \Delta w_{\mathrm{enh}}.$$

Define the corresponding reductions in the target margin as

$$\gamma_{\mathrm{rem}} = -\nabla_w M(\widetilde{w}_T)^\top \Delta w_{\mathrm{rem}}, \qquad \gamma_{\mathrm{enh}} = -\nabla_w M(\widetilde{w}_T)^\top \Delta w_{\mathrm{enh}}.$$

A positive γ_{rem} means that support removal weakens the target-class evidence, while a positive γ_{enh} means that interference enhancement strengthens the competing-class influence.

D.2 Sufficient Condition for Attack Success

Assume that M is differentiable around $\widetilde{w}_T$ and has local smoothness constant L_M. The first-order expansion gives

$$M(\widetilde{w}^-) = M(\widetilde{w}_T) - \gamma_{\mathrm{rem}} - \gamma_{\mathrm{enh}} + R,$$

where

$$|R| \leq \frac{L_M}{2}\|\Delta w_{\mathrm{FU}}\|_2^2.$$

Therefore, the following condition is sufficient for a successful attack:

$$\gamma_{\mathrm{rem}} + \gamma_{\mathrm{enh}} > M(\widetilde{w}_T) + \frac{L_M}{2}\|\Delta w_{\mathrm{FU}}\|_2^2.$$

Under this condition, $M(\widetilde{w}^-) < 0$, and the target prediction changes from correct to incorrect.

This result explains the complementary roles of the two mechanisms. Support removal contributes γ_{rem} by eliminating evidence associated with the ground-truth class, whereas interference enhancement contributes γ_{enh} by increasing the competing-class influence. Either operation alone may be insufficient when the target has a large initial margin. Their combination increases the total margin reduction and makes the success condition easier to satisfy.

D.3 Correctness Before Unlearning

Let ρ_{sup} and ρ_{int} denote the support and interference influences established during federated training. Locally, the pre-unlearning margin can be expressed as $M(\widetilde{w}_T) \approx M_0 + \rho_{\mathrm{sup}} - \rho_{\mathrm{int}}$, where M_0 represents the contribution of the remaining training data. BURST preserves the target's correct prediction when $M_0 + \rho_{\mathrm{sup}} - \rho_{\mathrm{int}} > 0$. After unlearning, the margin becomes approximately

$$M(\widetilde{w}^-) \approx M_0 + \rho_{\mathrm{sup}} - \rho_{\mathrm{int}} - \gamma_{\mathrm{rem}} - \gamma_{\mathrm{enh}}.$$

Hence, BURST achieves the desired temporal behavior when

$$M_0 + \rho_{\mathrm{sup}} - \rho_{\mathrm{int}} > 0$$

during federated training, but

$$\gamma_{\mathrm{rem}} + \gamma_{\mathrm{enh}} > M_0 + \rho_{\mathrm{sup}} - \rho_{\mathrm{int}}$$

after unlearning. This formalizes why the competing influence should not be made dominant during training and why the decisive prediction change is deferred to the unlearning stage.

D.4 Role of Auxiliary-Sample Construction

Feature-based retrieval and perturbation optimization are intended to increase the consistency of the two directional influences. If the target representation and an auxiliary sample representation satisfy

$$\|h(x) - h(x_t)\|_2 \leq \varepsilon,$$

and the local influence function is L_h-Lipschitz in the feature space, then

$$|\phi(x) - \phi(x_t)| \leq L_h\varepsilon.$$

Thus, feature-aligned auxiliary samples are more likely to produce influence directions that are consistent with the target. Perturbation optimization further reduces this representation mismatch within the allowed perturbation bound. Removing retrieval or perturbation optimization can therefore reduce γ_{rem} or γ_{enh}, which explains the weaker performance of RAS and UAR in the ablation experiments.

The analysis is local and provides a sufficient condition rather than a universal guarantee for every model and FU implementation. Its purpose is to show that, under the stated conditions, the two mechanisms used by BURST can preserve a correct pre-unlearning prediction while jointly inducing a post-unlearning misclassification.