Robust Federated Unlearning

Xinyi Sheng[*]
The University of Sydney
School of Computer Science
Sydney, NSW, Australia
xshe9923@uni.sydney.edu.au

Wei Bao
The University of Sydney
School of Computer Science
Sydney, NSW, Australia
wei.bao@sydney.edu.au

Liming Ge
The University of Sydney
School of Computer Science
Sydney, NSW, Australia
lige0519@uni.sydney.edu.au

ABSTRACT

Federated unlearning (FU) algorithms offer participants in federated learning (FL) the “right to be forgotten” for their individual data and its impact on a collaboratively trained model. Existing FU algorithms primarily focus on accelerating the retraining process and enhancing the utility of the retrained models following data removal requests. However, these approaches generally lack consideration for the robustness of FU algorithms in potential adversarial environments, where adversaries can craft malicious data removal requests to compromise the retrained model. In this work, we introduce a robust federated unlearning framework (robustFU) which notably enhances the resilience of FU algorithms against a wide range of adversarial attacks. In robustFU, we design a novel dynamic conflict sample compensation algorithm that dynamically reintroduces randomly generated samples with significant information gain to the participating clients during retraining. Additionally, robustFU employs an innovative global reweighting mechanism which adjusts the weight of each model update during the global aggregation, based on its degree of misalignment with the trained model prior to unlearning. Extensive experiments demonstrates the effectiveness and robustness of the proposed robustFU framework under adversarial environments. Furthermore, robustFU significantly accelerates the retraining process, achieving a 2.53× speedup compared to the retrain from the scratch baseline.

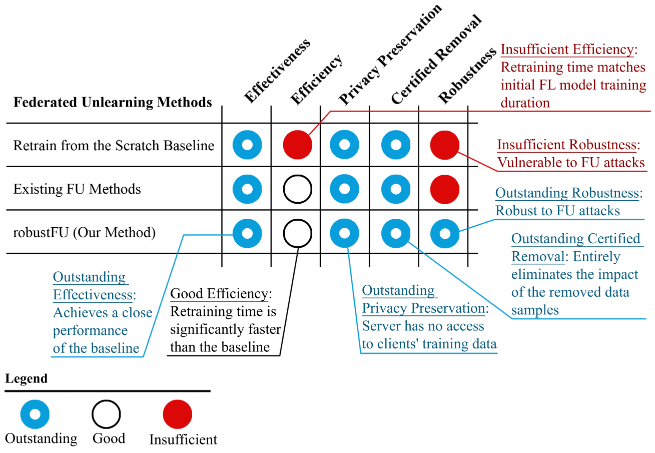
CCS CONCEPTS

• **Security and privacy → Distributed systems security**; • **Computing methodologies → Distributed algorithms**; **Machine learning**.

KEYWORDS

Federated Learning and Unlearning, Robust Federated Unlearning, Adversarial Attack and Defense, Algorithmic Fairness

ACM Reference Format:
Xinyi Sheng, Wei Bao, and Liming Ge. 2024. Robust Federated Unlearning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3627673.3679817



Federated Unlearning Methods	Effectiveness	Efficiency	Privacy Preservation	Certified Removal	Robustness
Retrain from the Scratch Baseline	◉	●	◉	◉	●
Existing FU Methods	◉	○	◉	◉	●
robustFU (Our Method)	◉	○	◉	◉	◉

Insufficient Efficiency: Retraining time matches initial FL model training duration
Insufficient Robustness: Vulnerable to FU attacks
Outstanding Robustness: Robust to FU attacks
Outstanding Certified Removal: Entirely eliminates the impact of the removed data samples
Outstanding Effectiveness: Achieves a close performance of the baseline
Good Efficiency: Retraining time is significantly faster than the baseline
Outstanding Privacy Preservation: Server has no access to clients' training data
Legend
◉ Outstanding
○ Good
● Insufficient

Figure 1: Qualitative Comparison of Existing FU Methods with Our Method.

1 INTRODUCTION

Recent developments in data privacy and protection legislation, including the European Union’s General Data Protection Regulation (GDPR) [43] and the California Consumer Privacy Act (CCPA) [19], have highlighted the importance of granting users the “right to be forgotten (RTBF)” regarding their private data in trained machine learning (ML) models. In response, machine unlearning (MU) methods [52] have been developed to efficiently eliminate the impact of specific data samples from a trained ML model in centralized settings. However, with the shift towards data decentralization [1] and increasing concerns over data privacy [55], ensuring the RTBF within the context of federated learning (FL) [32], a typically distributed framework, becomes an even more emerging issue.

FL enables multiple clients to collaboratively train a ML model via a server while keeping their data private [32] [60]. Upholding the RTBF within FL, however, presents unique challenges such as the incremental learning nature of FL [29] and the server’s inability to access training data [49], rendering the centralized MU methods inapplicable. To address this, federated unlearning (FU) methods [30] have been developed to perform efficient data deletion in FL models.

Recent efforts in FU have focused on developing methods that target several key objectives (as decipted in Figure 1), including: 1) the *effectiveness* of the retrained model [51] [40] (*i.e.*, ensuring the retrained model maintains high utility without a significant drop in performance); 2) the *efficiency* of the retraining process [8] [18] (*i.e.*, aiming for a faster approach than retraining from the scratch

[*] Corresponding author

CIKM '24, October 21–25, 2024, Boise, ID, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679817

baseline); 3) the *privacy preservation* during the unlearning process [57] [44] (*i.e.*, preventing any client from disclosing private data); and 4) the *certified removal* in the unlearning method [48] [13] (*i.e.*, guaranteeing the complete elimination of the impact of the data specified for removal). Nevertheless, these methods overlook a significant gap in exploring the *robustness* of FU methods, particularly in adversarial environments. While FU supports the RTBF for all data holders, it simultaneously introduces a potential threat where adversaries could exploit malicious data removal requests to compromise the performance of the retrained model.

To this end, our study initiates an in-depth exploration of the robustness of FU while maintaining focus on other fundamental objectives (as illustrated in the last row of Figure 1). Above all, to comprehensively evaluate the robustness of FU methods against adversarial attacks, we introduce two innovative FU attacks: the *targeted federated unlearning attack* (TFUA) and the *fairness federated unlearning attack* (FFUA). These attacks utilize meticulously crafted malicious data removal requests to delete training data of specific patterns, thereby compromising different evaluation metrics (*e.g.*, accuracy, fairness) of the retrained model. Then, given the lack of focus on FU's robustness in existing methods, our objective is to develop a defense mechanism capable of effectively countering these FU attacks and improving the robustness of the FU process.

However, enhancing the robustness of FU introduces several new and non-trivial challenges. Firstly, the existing literature lacks a formalized definition of robust FU. Secondly, practically distinguishing between legitimate and malicious unlearning requests can be exceedingly difficult, if not impossible. For instance, a naive approach to identify a malicious data removal request might be to measure the performance degradation in the retrained model. However, given that FU emphasizes the RTBF for all data holders, a substantial impact on the retrained model by a data removal request could still be legitimate, possibly reflecting the data owner's need to remove certain data for valid reasons. Furthermore, given that multiple adversaries could target various metrics of the retrained global model in FU, designing an effective defense mechanism to counter these multifaceted attacks poses a significant challenge.

To address these challenges, we begin by formulating a formal definition of robust FU, deviating from the original objective of standard FU. Then, we propose an advanced robust federated unlearning framework, *robustFU*. Instead of explicitly distinguishing between malicious data remove requests and legitimate ones which may degrade the performance of the unlearned model, robustFU implicitly improves the robustness of FU. Inspired by the observation that successful FU attacks often stem from the absence of specific data patterns or distributions, robustFU integrates a novel *dynamic conflict sample compensation algorithm*, which dynamically reintroduces randomly generated samples with significant information gain to each participating client during the retraining process. Specifically, this algorithm keeps track of the conflicts between the inference labels from the trained global model prior to unlearning and each client's model update during retraining on a set of randomly generated unlabeled samples. Given these conflict samples are highly likely to embody the missing data patterns or distributions for clients affected by FU attacks, they are reintroduced to the respective client for future training (labeled by the trained model before unlearning), thus offering substantial information gain for those clients. Although various FU attacks may remove different data patterns or distributions to compromise different evaluation metrics in the retrained model, this algorithm is capable of universally handling these situations. Moreover, robustFU incorporates an innovative *global reweighting mechanism* to adjust the weight of each client's model update during the aggregation phase of the retraining process. Specifically, model updates that perform closely to the trained model before unlearning are allocated higher weights, while those exhibiting significant behavioral differences are assigned lower weights. We show that this reweighting mechanism further enhances the robustness of FU and accelerates the retraining process.

In conclusion, our research makes following contributions:

- We introduce two innovative FU attacks that notably compromise different evaluation metrics of the retrained model by crafting malicious data removal requests.
- We propose a robust federated unlearning framework, robustFU, which employs a novel dynamic conflict sample generation algorithm and a global reweighting mechanism to implicitly enhance the robustness of FU against malicious data removal requests and legitimate ones which may degrade the performance of the unlearned model.
- Our comprehensive experiments demonstrate that the proposed robustFU not only improves FU's robustness but also meets other FU objectives, including effectiveness, efficiency, privacy preservation, and certified removal.

2 RELATED WORK

2.1 Machine Unlearning (MU)

The concept of MU, which enables a ML model to forget a subset of its training data, was first introduced by Cao *et al.* [7], where learning algorithms are transformed into a summation form for efficient unlearning. Subsequent MU strategies can be broadly classified into two categories: *data-based approaches* [5, 9, 17, 35], which modify the training data (*e.g.*, data pruning, data replacement) during the unlearning phase, and *model-based approaches* [2, 6, 14, 16, 23, 24], which directly manipulate the model parameters (*e.g.*, model shifting, model pruning) during the unlearning phase. However, these methods are primarily developed for centralized ML, where the entire training dataset is accessible during the unlearning process, making them less applicable in FL scenarios.

Robustness of machine unlearning. Recently, the robustness of MU has started gaining attention within the research community. Marchant *et al.* [31] introduced a slow-down attack using data poisoning to increase the computational cost of data removal in MU. Zhao *et al.* [58] proposed a malicious selective forgetting attack for both static and sequential forgetting scenarios, however, it lacks any defense mechanisms. In response, Qian *et al.* [36] formulated the unlearning attack as a constrained optimization problem and present a gradient influence based defense mechanism. Nevertheless, this attack and defense framework shows limited effectiveness in distributed learning settings.

2.2 Federated Unlearning (FU)

FU extends the concept of MU to a distributed setting, enabling clients to request the erasure of their data contributions from the

Table 1: Summary of Notations

Symbol	Definition	Symbol	Definition
n	the number of clients	$\boldsymbol{x}, y$	the input features and the label (class) of a data sample
m	the number of malicious adversaries (clients)	y_t	the targeted class in TFUA
$\mathcal{K}$	the set of all clients	$\boldsymbol{x}^+, \boldsymbol{x}^-$	the targeted and untargeted demographic group in FFUA
k_i	the i-th client	y^+, y^-	the privileged and unprivileged class in FFUA
$\mathcal{K}_u, \mathcal{K}_r$	the set of unlearned and remaining clients	$\hat{y}$	the predicted label of a data sample
$\mathcal{D}$	the entire training dataset	MAX_i^t	the maximum number of targeted samples in TFUA
$\mathcal{D}_i$	the local training dataset of client k_i	$MAX_i^{+-/-+}$	the maximum number of targeted samples in FFUA
$\mathcal{D}_i^u, \mathcal{D}_i^r$	the unlearned and remaining samples of client k_i	P_i	the number of removed samples of client k_i in TFUA
$\tilde{\mathcal{D}}$	the total remaining samples after unlearning	Q_i, Q'_i	the number of removed samples of client k_i in FFUA
$\mathcal{M}$	the trained global model (prior to unlearning)	$\mathcal{D}_{rand}$	the set of randomly generated unlabeled samples
$\mathcal{M}_i$	the local model (model update) of client k_i	R	the size of randomly generated unlabeled samples
$\overline{\mathcal{M}}$	the global model retrained from the scratch	$\mathcal{L}_\mathcal{M}$	the label vector of the trained global model $\mathcal{M}$
$\overline{\mathcal{M}'}$	the global model robustly retrained from the scratch	$\mathcal{L}_i$	the label vector of the local model update $\mathcal{M}_i$
$\tilde{\mathcal{M}}$	the unlearned global model (using FU technique)	$\mathcal{D}_{conf}^i$	the set of conflict samples for client k_i
$\mathcal{X}, \mathcal{Y}$	the input and output space of data samples	s_i	the conflict score of client k_i
d	the dimension of the input space	B, E	the local training batches and global training epochs

collectively trained model. Within the FL framework, the concept of FU can be further categorized into three main types: *class-level FU* [45] [59] [47], focused on removing specific classes from the trained global model; *client-level FU* [28] [18] [48] [40] [57], aimed at removing the contributions of entire clients from the trained global model; and *sample-level FU* [29] [51] [46] [8] [61], which targets the removal of the impact of selected data samples within participating clients from the trained global model. Our research falls into the sample-level FU category, addressing scenarios where only a subset of local training samples from a client is requested for removal during the unlearning phase, while the remaining samples from the client continue to contribute to the retrained or unlearned global model. Within this category, efforts have been made to enhance the *efficiency* of the retraining process (*e.g.*, using Hessian-based [29] or quantization-based [51] methods), and to improve the *effectiveness* of the unlearned model (*e.g.*, using nonlinear functional analysis [8] or knowledge distillation [61]) in FU. However, none of these studies address the *robustness* of FU methods, especially in adversarial settings. Therefore, our work pioneers the exploration of FU algorithms' robustness, while also maintaining effectiveness and efficiency as the primary objectives.

3 PROBLEM FORMULATION

Table 1 provides a summary of the notation used throughout this paper.

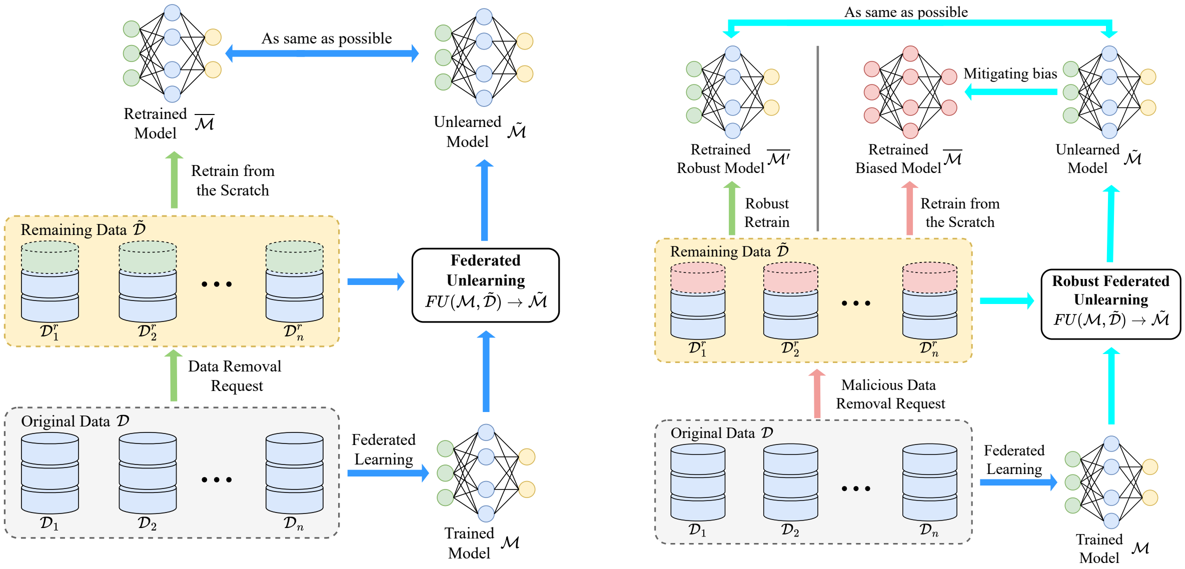
3.1 Federated Learning and Unlearning

3.1.1 ***Federated learning.*** We consider a typical FL setting where a group of n clients, represented as $\mathcal{K} = k_1, k_2, \ldots, k_n$, collaborates with a server to train a global model $\mathcal{M}$. Each client k_i maintains its own local dataset $\mathcal{D}_i$, and the entire training data is denoted as $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \ldots \cup \mathcal{D}_n$. The FL process is characterized by two main steps that repeat until the training is complete. (1) *Local Update*: Each client k_i trains its local model $\mathcal{M}_i$ using its dataset $\mathcal{D}_i$ (*e.g.*, using stochastic gradient descent) and uploads the model update to the server. (2) *Global Aggregation*: After collecting model updates from the clients, the server aggregates these updates based on certain aggregation rules (*e.g.*, in FedAvg, we have $\mathcal{M} \leftarrow \sum_{i=1}^{n} \frac{|\mathcal{D}_i|}{|\mathcal{D}|} \mathcal{M}_i$) to update the global model $\mathcal{M}$, which is then distributed back to each client for the next iteration.

3.1.2 ***Federated unlearning.*** In a FU system, the set of clients $\mathcal{K}$ can be further divided into two subsets $\mathcal{K}_u$ and $\mathcal{K}_r$ ($\mathcal{K} = \mathcal{K}_u \cup \mathcal{K}_r$), which represents the set of unlearned clients and remaining clients, respectively. For each unlearned client $k_i \in \mathcal{K}_u$, a data removal request is executed to unlearn part of its samples $\mathcal{D}_i^u \subset \mathcal{D}_i$, with $\mathcal{D}_i^r = \mathcal{D}_i \backslash \mathcal{D}_i^u$ denotes the remaining samples. Let $\tilde{\mathcal{D}}$ represents the remaining samples after all data removal requests have been executed, where $\tilde{\mathcal{D}}$ contains $\mathcal{D}_i^r$ for each $k_i \in \mathcal{K}_u$ and $\mathcal{D}_j$ for each $k_j \in \mathcal{K}_r$. The FU process is then defined as a function $FU(\mathcal{M}, \tilde{\mathcal{D}}) \rightarrow \tilde{\mathcal{M}}$ that produces an unlearned global model $\tilde{\mathcal{M}}$, using the trained global model $\mathcal{M}$ prior to unlearning and the remaining data samples $\tilde{\mathcal{D}}$. Let $\overline{\mathcal{M}}$ denotes the retrained model based on the remaining data samples $\tilde{\mathcal{D}}$ (retrain from the scratch). We can now define the objective of a standard FU process, as depicted in Figure 2 (a).

Definition 3.1. (Federated Unlearning) The objective of a FU process is to eliminate the contribution of all removed data samples from the trained model $\mathcal{M}$, while making the unlearned model $\tilde{\mathcal{M}}$ as same as possible to the retrained from the scratch model $\overline{\mathcal{M}}$.

While this definition fits well for common scenarios in FU, it becomes less effective under adversarial conditions where the retrained model $\overline{\mathcal{M}}$ can be biased by malicious data removal requests. In such instances, the biased retrained model $\overline{\mathcal{M}}$ can no longer serve as a suitable reference for the unlearned model $\tilde{\mathcal{M}}$ (See Section 5.2.1). To address this, we adjust the objective of a standard FU process by aligning the unlearned model $\tilde{\mathcal{M}}$ with a robustly retrained model $\overline{\mathcal{M}'}$, assuming that the robust retraining process can effectively mitigate the bias in the remaining data samples $\tilde{\mathcal{D}}$.


As same as possible
Retrained Model $\overline{\mathcal{M}}$
Unlearned Model $\tilde{\mathcal{M}}$
Retrain from the Scratch
Remaining Data $\tilde{\mathcal{D}}$
$\mathcal{D}_1^r$
$\mathcal{D}_2^r$
$\mathcal{D}_n^r$
Federated Unlearning $FU(\mathcal{M}, \tilde{\mathcal{D}}) \rightarrow \tilde{\mathcal{M}}$
Data Removal Request
Original Data $\mathcal{D}$
$\mathcal{D}_1$
$\mathcal{D}_2$
$\mathcal{D}_n$
Federated Learning
Trained Model $\mathcal{M}$
As same as possible
Retrained Robust Model $\overline{\mathcal{M}'}$
Retrained Biased Model $\overline{\mathcal{M}}$
Mitigating bias
Unlearned Model $\tilde{\mathcal{M}}$
Robust Retrain
Retrain from the Scratch
Remaining Data $\tilde{\mathcal{D}}$
$\mathcal{D}_1^r$
$\mathcal{D}_2^r$
$\mathcal{D}_n^r$
Robust Federated Unlearning $FU(\mathcal{M}, \tilde{\mathcal{D}}) \rightarrow \tilde{\mathcal{M}}$
Malicious Data Removal Request
Original Data $\mathcal{D}$
$\mathcal{D}_1$
$\mathcal{D}_2$
$\mathcal{D}_n$
Federated Learning
Trained Model $\mathcal{M}$

(a) Federated Unlearning

(b) Robust Federated Unlearning

Figure 2: Overview of Federated Unlearning and Robust Federated Unlearning.

This provides a formal definition of the objective for robust FU, which is also illustrated in the top left of Figure 2 (b).

Definition 3.2. (Robust Federated Unlearning) The objective of a robust FU process is to eliminate the contribution of all removed data samples from the trained model $\mathcal{M}$, while making the unlearned model $\tilde{\mathcal{M}}$ as same as possible to the robustly retrained model $\overline{\mathcal{M}'}$.

This definition, however, may not be entirely viable in practice as it is challenging to identify a suitable benchmark to measure the robustness of the retraining process. This challenge arises because an adversary might target different metrics of the retrained model, and there are various robust training methods that improve the model's robustness to different extents (See Section 5.2.1). Thus, a more practical objective of robust FU is to mitigate the bias introduced to the retrained model $\overline{\mathcal{M}}$ during the unlearning phase, as illustrated in the top right of Figure 2 (b).

3.2 Threat and Adversary Model

3.2.1 ***Threat Model.*** In our threat model, we consider a general scenario in which m ($m < n$) malicious adversaries within a FL system either act collaboratively or independently to attack the FU model. The objective of these adversaries is to impact one or more metrics (*e.g.*, accuracy, fairness) of the unlearned global model $\tilde{\mathcal{M}}$. While existing literature extensively covers adversarial attacks and their corresponding defenses within the standard FL process [3, 38, 41, 42], our study shifts focus exclusively to potential threats associated with the unlearning phase of FU. We assume that all attacks are executed during the unlearning phase. In other words, we assume that the learning phase prior to unlearning is honest and unaffected by any attacks, ensuring that the global model $\mathcal{M}$ obtained before unlearning is high-utility and fair[1]

3.2.2 ***Adversarial Knowledge and Capability.*** We hypothesize that an adversary has access to all local data within a client (*i.e.*, $\mathcal{D}_i$) and can decide on the data removal request (*i.e.*, $\mathcal{D}_i^u$) for any data within it. However, we assume the adversary cannot influence the client's training process or the global aggregation process during the retraining phase. This scenario aligns with practical use cases for FU, where a malicious user may request the removal of part or all of its individual data from a service provider, but lacks control over how the service provider utilizes the remaining data.

4 METHODS

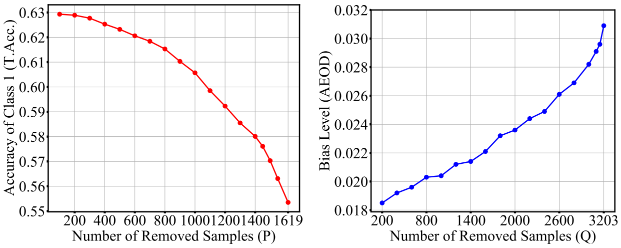
4.1 Federated Unlearning Attacks

FU attacks are carried out through malicious data removal requests. Specifically, an adversary can meticulously craft malicious data removal requests to remove a specific pattern of data samples from a client k_i, thereby compromising the local model update $\mathcal{M}_i$, which further affects the retrained global model $\overline{\mathcal{M}}$. These attacks via data removal can be seen as a special form of data poisoning attack [42], where instead of flipping labels or generating poisoning samples, the data distribution is corrupted by removing data samples. Next, we introduce two types of FU attacks: *Targeted federated unlearning attack* (TFUA) and *fairness federated unlearning attack* (FFUA).

4.1.1 ***Targeted federated unlearning attack.*** Drawn inspiration from the targeted poisoning attack [42] in FL, TFUA aims to reduce the test accuracy of samples from a specific class. To carry out a TFUA, an adversary can strategically remove partial or all training samples associated with the targeted class from a local client's dataset. This leads to a substantial loss of information about the targeted class in the retrained model of the affected client, thereby diminishing its performance for the targeted class.

Formally, consider a binary classification model where $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}$ represent the input and output spaces, respectively.

[1] There are many studies in the existing literature [56] [12] that have made significant strides towards enhancing the fairness of FL models.



(a) TFUA

(b) FFUA

Figure 3: Impact of Data Removal Quantities on FU Attacks.

For a given data sample $(\boldsymbol{x}, y)$, $\boldsymbol{x} \in \mathcal{X}$ denotes its input features, and $y \in \mathcal{Y}$ denotes its label (class). Let $y_t \in \mathcal{Y}$ represents the targeted class in TFUA, and MAX_i^t indicates the maximum number of samples labeled as y_t within the local dataset $\mathcal{D}_i$ of client k_i. We can now define TFUA as follows:

Definition 4.1. (Targeted federated unlearning attack) A TFUA is executed by randomly removing a quantity P_i ($P_i \in [0, MAX_i^t]$) of data samples labeled y_t from the dataset $\mathcal{D}_i$ of client k_i (i.e., $\mathcal{D}_i^u = \{(\boldsymbol{x}_p, y_p)|y_p = y_t\}_{p=1}^{P_i}$), thereby decreasing the test accuracy of the retrained model $\overline{\mathcal{M}}$ on test samples with class y_t.

It is worth noting that TFUA exhibits a similar property to the targeted poisoning attack [42] in FL, where only the accuracy of the targeted class y_t is compromised, while the overall accuracy of the retrained model $\overline{\mathcal{M}}$ remains largely unaffected. Thus, it represents a particularly stealthy form of attack, which is hard to detect. (See Section 5.2.2)

4.1.2 *Fairness federated unlearning attack.*

In the context of sensitive ML, algorithmic (demographic) fairness underscores the principle that a ML model should not exhibit discrimination against any particular demographic group [33]. Therefore, a FFUA aims to bias the retrained model $\overline{\mathcal{M}}$ in such a way that it discriminates against a specific demographic group.

Specifically, consider a binary classification problem where $y^+ \in \mathcal{Y}$ represents the privileged class (*e.g.*, income > 50K) and $y^- \in \mathcal{Y}$ represents the unprivileged class (*e.g.*, income ≤ 50K). A data sample $(\boldsymbol{x}, y)$ can be divided into two subsets based on its demographic information (*e.g.*, gender), with $(\boldsymbol{x}^+, y)$ denoting one demographic group (*e.g.*, male) while $(\boldsymbol{x}^-, y)$ represents the other (*e.g.*, female). In FFUA, without loss of generality, we consider samples denoted as $\boldsymbol{x}^+$ representing the targeted demographic group. The adversary's goal is to advantage this group while disadvantaging the untargeted group. This can be achieved by either removing samples of the targeted demographic group $\boldsymbol{x}^+$ with unprivileged class y^-, or removing samples of untargeted demographic group $\boldsymbol{x}^-$ with privileged class y^-, or both. Let MAX_i^{+-} and MAX_i^{-+} indicate the maximum number of samples from the targeted demographic group with unprivileged class and from the untargeted demographic group with privileged class within the local dataset $\mathcal{D}_i$ of client k_i, respectively. We can now define FFUA as follows:

Definition 4.2. (Fairness federated unlearning attack) A FFUA is executed by randomly removing a quantity Q_i ($Q_i \in [0, MAX_i^{+-}]$) of data samples belonging to the targeted demographic $\boldsymbol{x}^+$ group with unprivileged class y^-, and a quantity Q'_i ($Q'_i \in [0, MAX_i^{-+}]$) of data samples belonging to the untargeted demographic group $\boldsymbol{x}^-$ with privileged class y^+, from the dataset $\mathcal{D}_i$ of client k_i (i.e., $\mathcal{D}_i^u = \{(\boldsymbol{x}_q^+, y_q)|y_q = y^-\}_{q=1}^{Q_i} \cup \{(\boldsymbol{x}_{q'}^-, y_{q'})|y_{q'} = y^+\}_{q'=1}^{Q'_i}$), thereby compromising the fairness of the retrained model $\overline{\mathcal{M}}$.

While FFUA significantly biases the retrained model $\overline{\mathcal{M}}$, it also maintains a high level of stealthiness, exerting minimal influence on its overall performance. This is because FFUA only corrupts the relationship between one (out of d) feature and the label, leaving the overall joint distribution of all input features and the label largely unaffected. (See Section 5.2.2)

Figure 3 demonstrates the impact of data removal quantities P and Q (Q') on the efficacy of TFUA and FFUA, respectively. It is observable that an increase in the amount of data removal enhances the strength of both attacks, with the strongest attacks occurring upon the removal of the maximum amount of data available. Notably, although increasing the volume of data $\mathcal{D}_i^u$ removed from client k_i reduces its aggregation weight $\frac{|\mathcal{D}_i^r|}{|\tilde{\mathcal{D}}|}$ in a retraining process using standard FedAvg [32], the efficacy of both attacks remains primarily dominated by the quantity of data removal. This also highlights the critical role of information loss in both attacks. (See Section 5.2.3 for detailed experimental setup.)

4.2 Robust Federated Unlearning

4.2.1 *robustFU Overview.*

Figure 4 provides an overview of robustFU. There are three main components within robustFU: Random sample generation, dynamic conflict sample compensation, and global reweighting. Initially, a set of unlabeled samples $\mathcal{D}_{rand}$ is generated randomly at the server during each aggregation round. The trained global model $\mathcal{M}$ prior to FU then infers labels for $\mathcal{D}_{rand}$, producing a label vector $\mathcal{L}_{\mathcal{M}}$. After that, each client's model update $\mathcal{M}_i$ also infers on $\mathcal{D}_{rand}$ to generate the corresponding label vector $\mathcal{L}_i$. For dynamic conflict sample compensation, a conflict sample set $\mathcal{D}_{conf}^i$ is identified for each client k_i from the discrepancies between $\mathcal{L}_i$ and $\mathcal{L}_{\mathcal{M}}$, which is then sent to k_i as the compensation samples for future training. Additionally, a conflict score s_i is calculated to measure the misalignment between $\mathcal{L}_i$ and $\mathcal{L}_{\mathcal{M}}$, reflecting the closeness of a client's update $\mathcal{M}_i$ to the trained global model $\mathcal{M}$. Subsequently, during the global aggregation, each model update $\mathcal{M}_i$ is reweighted based on its s_i, with those updates more closely aligned with $\mathcal{M}$ receiving a higher weight. Algorithm 1 illustrates a detailed procedure of robustFU.

4.2.2 *Random Sample Generation.*

During each global aggregation round in robustFU, an unlabeled sample set $\mathcal{D}_{rand}$ of size R is randomly generated (Line 2). Specifically, for each nominal feature, its value is selected uniformly at random from its respective domain, and for each numeric feature, its value is generated uniformly at random within its range. Since the features of these samples are randomly generated, their labels are unknown. To assign labels to these randomly generated samples, the trained global model $\mathcal{M}$ prior to FU is used to infer on $\mathcal{D}_{rand}$, producing the label vector $\mathcal{L}_{\mathcal{M}}$ (Line 3). This approach of random samples generation allows robustFU to be entirely data-free on the server side, eliminating the need for participating clients to share their private data. It is

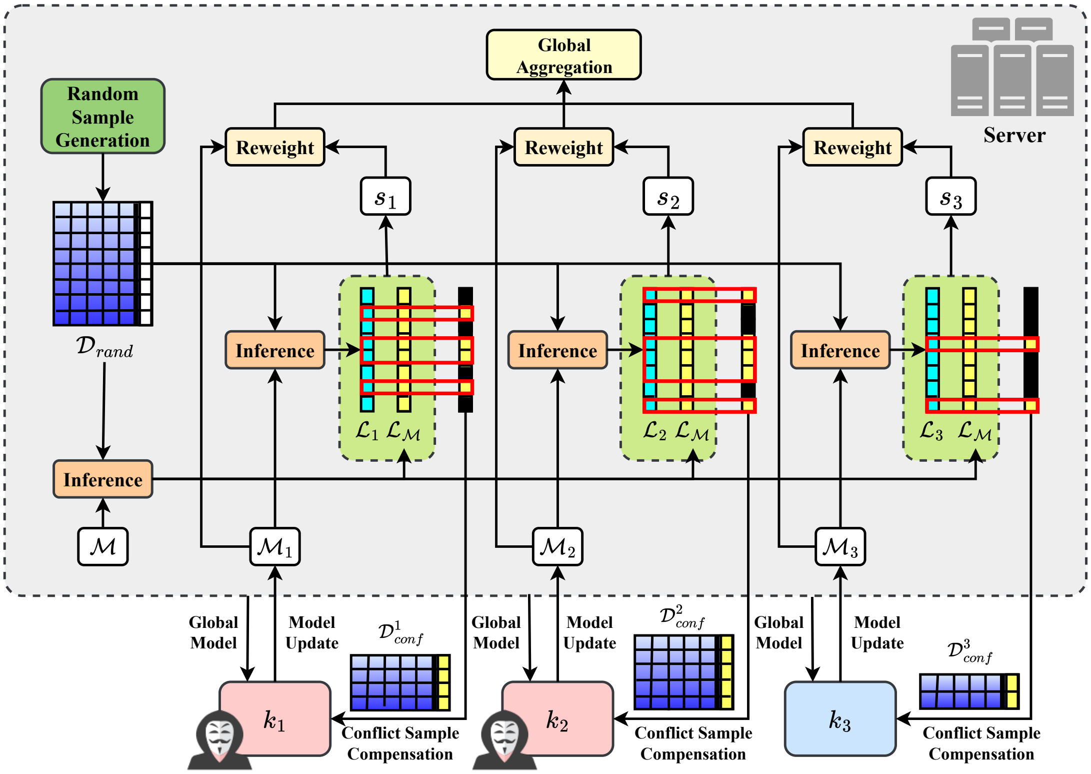


Figure 4: Overview of robustFU.

noteworthy that while the generated label vector $\mathcal{L}_\mathcal{M}$ may not be completely accurate, it reflects the behavior of $\mathcal{M}$ in predicting these samples.

4.2.3 *Dynamic Conflict Sample Compensation*.

The design of the dynamic conflict sample compensation algorithm is inspired by the concept of entropy in information theory [37], where it represents the amount of information contained in the occurrence of an event, with events of lower probability carrying more information when they happen. Given that FU attacks bias a local model update by removing data with a specific pattern from its local training dataset, data containing this specific pattern are considered to carry more information (higher entropy value) for that client. Notably, this concept is further validated by our experiments on FU attacks. Observations from the bottom right of Figure 3 (a) and the top right of Figure 3 (b) reveal that the impact of FU attacks on model metrics significantly increases (shown by a steeper slope) as the number of removed samples P and Q approaches their maximum values MAX_i^t and MAX_i^{+-}. This is because that, as the quantity of data with a certain pattern is removed, the remaining minority of data with this pattern contains an increasingly larger amount of information. Consequently, removing such small portion of highly informative data has a relatively greater effect on the model performance.

Drawing from these insights, the dynamic conflict sample compensation algorithm functions by dynamically reintroducing samples with high entropy values back to each client during the retraining phase, thereby counteracting the information loss induced by FU attacks. Specifically, during each global aggregation round, the model update $\mathcal{M}_i$ is employed to make inference on $\mathcal{D}_{rand}$, generating the label vector $\mathcal{L}_i$ for client k_i (Line 5). A compensate sample set $\mathcal{D}^i_{conf}$ is then identified based on the label discrepancies between $\mathcal{L}_i$ and $\mathcal{L}_\mathcal{M}$ for client k_i, where samples in $\mathcal{D}^i_{conf}$ are labeled according to $\mathcal{L}_\mathcal{M}$ (Line 6). Subsequently, $\mathcal{D}^i_{conf}$ is sent to client k_i as additional training samples for future training (Line 7).

It is worthy noticing that in the dynamic conflict sample compensation algorithm, the compensation samples within $\mathcal{D}^i_{conf}$ are not explicitly crafted to exhibit any specific data patterns. Instead, they are implicitly selected based on discrepancies between $\mathcal{L}_i$ and $\mathcal{L}_\mathcal{M}$ to compensate for the missing data patterns in client k_i. For example, a client k_i affected by TFUA may have more label conflicts with $\mathcal{M}$ on samples within the targeted class y_t, while a client k_j affected by FFUA may have more label conflicts with $\mathcal{M}$ on samples from the targeted demographic group $\boldsymbol{x}^+$ with unprivileged label y^-. In this case, the compensation dataset $\mathcal{D}^i_{conf}$ will contain more samples labeled as y_t, and the compensation dataset $\mathcal{D}^j_{conf}$ will contain more samples formed as $(\boldsymbol{x}^+, y^-)$, carrying significant information gain for client k_i and k_j, respectively. Through this way, this algorithm can *universally* defend against various FU attacks. (See Section 5.3.1)

In addition, the dynamic conflict sample compensation algorithm is not limited to clients requesting data removals ($\mathcal{K}_u$) but is also beneficial for clients not requesting data removals ($\mathcal{K}_r$). In such scenarios, the trained model $\mathcal{M}$ can be regarded as a teacher model[2],

[2] While the concept of the "teacher" model is borrowed from knowledge distillation (KD) [21], the proposed dynamic conflict sample compensation algorithm surpasses KD by addressing its limitations. In KD, if specific data patterns are missing from the

Algorithm 1 robustFU

```
1  **Server-side** (each global aggregation round):
2  Generate random unlabeled sample set $\mathcal{D}_{rand}$;
3  $\mathcal{L}_{\mathcal{M}} \leftarrow \mathcal{M}$ infers on $\mathcal{D}_{rand}$;
4  **for** *each participating client* $k_i$ **do**
5      $\mathcal{L}_i \leftarrow \mathcal{M}_i$ infers on $\mathcal{D}_{rand}$;
6      Identify $\mathcal{D}^i_{conf}$ based on $\mathcal{L}_i$ and $\mathcal{L}_{\mathcal{M}}$;
7      Send $\mathcal{D}^i_{conf}$ to client $k_i$;
8      $s_i \leftarrow ||\mathcal{L}_i - \mathcal{L}_{\mathcal{M}}||_1$;
9  Update global model $\tilde{\mathcal{M}} \leftarrow \sum_i \frac{\frac{1}{s_i}}{\sum_j \frac{1}{s_j}} \mathcal{M}_i$;
10 Send $\tilde{\mathcal{M}}$ to all clients;
11 **Client-side** (each local update round):
12 **for** *each participating client* $k_i$ **do**
13     Receive $\mathcal{D}^i_{conf}$ and $\tilde{\mathcal{M}}$ from the server;
14     $\mathcal{D}_i \leftarrow \mathcal{D}_i \cup \mathcal{D}^i_{conf}$;
15     $\mathcal{M}_i \leftarrow$ Local updating $\tilde{\mathcal{M}}$ with dataset $\mathcal{D}_i$;
16     Send $\mathcal{M}_i$ to the server;
```

and the conflict samples can be seen as more informative than non-conflict ones in directing the retrained model towards the trained model $\mathcal{M}$, thereby *accelerating* the retraining process. (See Section 5.3.2)

Furthermore, there is no need to worry that reintroducing conflict samples into each client's dataset might impact the certified data removal process in FU. This is because the conflict samples are generated randomly across a vast feature space, making it highly unlikely for them to match the specific samples requested for removal. Our experiments further validate this point through different verification methods. (See Section 5.3.3)

4.2.4 *Global Reweighting.*

To further enhance the retraining speed and robustness, robustFU utilizes an innovative reweighting mechanism during the global aggregation. This mechanism involves calculating a conflict score s_i, which quantifies the misalignment between a client's model update $\mathcal{M}_i$ and the trained global model $\mathcal{M}$, based on the number of conflict samples in $\mathcal{D}_{rand}$. The conflict score is determined using the l_1-norm between the label vectors $\mathcal{L}_i$ and $\mathcal{L}_{\mathcal{M}}$ (*i.e.*, $s_i = ||\mathcal{L}_i - \mathcal{L}_{\mathcal{M}}||_1$) (Line 8). Then, during the global aggregation, the weight for each participating client's model update $\mathcal{M}_i$ is adjusted based on the normalized inverse conflict score, with model updates possessing lower conflict scores being assigned higher weights. Formally, we have:

$$\tilde{\mathcal{M}} \leftarrow \sum_i \frac{\frac{1}{s_i}}{\sum_j \frac{1}{s_j}} \mathcal{M}_i \tag{1}$$

Here, for the sake of simplicity, we use $\tilde{\mathcal{M}}$ and $\mathcal{M}_i$ to denote the updated global model and local model updates, respectively, within one communication round during the retraining process (Line 9).

The advantage of this reweighting mechanism is twofold: Firstly, it *accelerates* the retraining process by assigning higher weights to model updates that exhibit behaviors similar to the trained global model $\mathcal{M}$ (*i.e.*, those with lower conflict scores). Secondly, by penalizing model updates that diverge from the global model $\mathcal{M}$ (*i.e.*, those with higher conflict scores), it enhances the *robustness* of the retrained model. (See Section 5.4)

5 EXPERIMENTAL RESULTS

5.1 Experimental Setup

5.1.1 *Datasets.*

To comprehensively evaluate the proposed FU attacks and the robustFU defense framework, we employ two binary prediction datasets containing demographic information in their feature spaces: the Adult Income dataset [10] and the ProPublica COMPAS dataset [25]. The Adult dataset is used to predict whether an individual's income exceeds 50K, while the COMPAS dataset focuses on predicting recidivism within two years of sentencing. In TFUA, we set the class with label 1 as the targeted class (*i.e.*, $y_t = 1$) for both datasets. For FFUA, we consider gender (Male vs. Female) and race (African American vs. non-African American) as the sensitive attributes for the Adult and COMPAS datasets, respectively.

5.1.2 *Evaluation Metrics.*

As different FU attacks target different evaluation metrics of the retrained model, we utilize four metrics in our analysis: **Metric 1 (T.Acc.)**: Accuracy for samples labeled as the targeted class in TFUA; **Metric 2 (AEOD)**: The absolute equal opportunity difference [20] is defined as $AEOD = |\Pr(\hat{y}^+|\boldsymbol{x}^+, y^+) - \Pr(\hat{y}^+|\boldsymbol{x}^-, y^+)|$, which measures the absolute difference in true positive rates between two demographic groups; **Metric 3 (ASPD)**: The absolute statistical parity difference [11] is defined as $ASPD = |\Pr(\hat{y}^+|\boldsymbol{x}^+) - \Pr(\hat{y}^+|\boldsymbol{x}^-)|$, which quantifies the disparity in positive outcome rates across demographic groups; and **Metric 4 (G.Acc.)**: Overall accuracy for all samples during testing. Among these metrics, Metric 1 assesses the impact of TFUA, Metrics 2 and 3 evaluate the effects of FFUA, whereas Metric 4 gauges the overall performance of the retrained model.

5.1.3 *Benchmarks.*

Given the lack of explorations regarding the robustness of FU in the existing literature, we employ benchmarks from various perspectives to ensure a fair evaluation of the proposed method. Above all, the **Retrain from the Scratch** method is utilized as the baseline, wherein FedAvg [32] is performed on the remaining data $\tilde{\mathcal{D}}$ during the retraining phase. Then, we employ the rapid retraining algorithm [29] (labeled as **INFOCOM'22**), a state-of-the-art sample-level FU method which adopts the second-order AdaHessian optimizer [53] to accelerate the retraining process. Furthermore, in light of the formal definition of robust FU (Definition 3.2), we explore a variety of robust retraining strategies across different domains. Specifically, to improve the robustness of the retrained model against TFUA, we examine three Byzantine-robust aggregation rules including **Krum** [4], **Mean** [54], and **Median** [54]. These approaches increase the resilience of the retraining process to statistical outliers among model updates. Conversely, to enhance

dataset, the student model may be unable to learn the knowledge associated with those patterns because the teacher model lacks the opportunity to demonstrate its response to these absent patterns. Conversely, our algorithm addresses this by reintroducing data samples with the missing patterns, enabling the trained model $\mathcal{M}$ to effectively "teach" the retrained model across all data distributions.

the robustness of retrained model against FFUA, we adopt two state-of-the-art fairness algorithms in FL: **FairFL** [56], which utilizes a reinforcement learning-based client selection method to improve algorithmic fairness, and **FairFed** [12], which applies a reweighting technique to model parameters during the global aggregation phase to ensure algorithmic fairness. These robust retraining benchmarks not only assess the effectiveness of existing defense mechanisms against FU attacks but also serve as the reference benchmark $\overline{\mathcal{M}'}$ for the unlearned model $\tilde{\mathcal{M}}$.

5.1.4 ***Federated Unlearning Attacks Setup.*** To rigorously assess the effectiveness of the proposed FU attacks and the robustFU defense framework, we consider four different FU attack setups: **Setup 1 (TFUA only)**: Involves $n = 10$ clients in total, with $m = 3$ executing TFUA and 2 undertaking random unlearning; **Setup 2 (FFUA only)**: Involves $n = 10$ clients in total, with $m = 3$ executing FFUA and 2 undertaking random unlearning; **Setup 3 (TFUA+FFUA (small))**: Involves $n = 10$ clients in total, with 2 executing TFUA, 2 executing FFUA ($m = 4$), and 1 undertaking random unlearning; and **Setup 4 (TFUA+FFUA (large))**: Involves $n = 10$ clients in total, with 3 executing TFUA, 3 executing FFUA ($m = 6$), and 2 undertaking random unlearning. Among these settings, Setups 1 and 2 isolate individual FU attacks, while Setups 3 and 4 simulate more intricate scenarios combining both attacks within a single FU process. Given our threat model imposes no specific limit on the number m of attacked clients (except for less than n), we examine cases where m constitutes less than half (TFUA+FFUA(small)) and more than half (TFUA+FFUA(large)) of n. For all clients executing TFUA or FFUA, we assume the maximum number of samples corresponding to the targeted patterns are removed (*i.e.*, $P_i = MAX_i^t, Q_i = MAX_i^{+-}, Q'_i = MAX_i^{-+}$), simulating adversaries' maximal effort in these attacks. For clients that undertake random unlearning, one-third of their training samples are randomly selected for unlearning (*i.e.*, $|\mathcal{D}_i^u| = \frac{1}{3}|\mathcal{D}_i|$).

5.1.5 ***robustFU Setup.*** We deploy robustFU on a server equipped with an Intel(R) Core(TM) i9-10980XE CPU. To accelerate robustFU, we utilize multi-threaded concurrent programming on the server, facilitating parallel inference across various model updates on randomly generated samples. For the hyperparameter, we set the size of randomly generated samples to $R = 1000$ for the Adult dataset and $R = 100$ for the COMPAS dataset. This hyperparameter is crucial in the robustFU setup, as it directly affects multiple aspects of FU, including the effectiveness and robustness of the retrained model, as well as the efficiency of the retraining process. Thus, we conduct additional ablation studies to determine the appropriate value of R and offer further insights into identifying this number for other datasets (see Section 5.4).

5.1.6 ***Federated Learning Setup.*** In our experiments, we utilize a multi-layer perceptron (MLP) comprising two fully connected layers for both datasets. Unless stated otherwise, we set the number of global training epochs to $E = 50$ and the number of local training batches to $B = 40$ for the Adult dataset and $B = 5$ for the COMPAS dataset, with a batch size of 64. During each global epoch, half of the total clients (*i.e.*, 5) are randomly chosen to participate in the model update process. The stochastic gradient descent (SGD) optimizer, with a learning rate of 0.01, is employed for local training phases. To gain a comprehensive understanding of the performance of FU attacks and defenses across varied data distributions, we undertake experiments utilizing both IID and non-IID data distributions for each dataset. For generating non-IID data distributions, we adopt the method outlined in [22], setting the heterogeneity level parameter to 0.2. In our analysis, we present the average outcomes from experiments conducted with 20 randomly selected seeds for all dataset splits.

5.2 Evaluation of Federated Unlearning Attacks

Table 2 provides a comprehensive evaluation of the proposed FU attacks and the effectiveness of various defense strategies against these attacks within the Adult and COMPAS datasets, under both IID and non-IID data distributions, across four distinct FU attack setups. For clarity, we denote the best results among Krum, Mean, and Median as **Robust Retrain(Byzantine)**, and the best results between FairFL and FairFed as **Robust Retrain(Fairness)**. Regarding evaluation metrics, a higher T.Acc. and G.Acc. value indicates better performance (indicated by ↑), whereas a lower AEOD and ASPD value signifies better fairness (indicated by ↓). Due to the similar trends observed in the two fairness metrics, ASPD is omitted in Setups 3 and 4 for brevity. We highlight the best results in bold red and the second best in bold black. For better comparison, the "No Unlearning" scenario (marked in light blue) is included as an extra benchmark to illustrate the performance of the trained global model $\mathcal{M}$ prior to FU.

5.2.1 ***The Effectiveness of FU Attacks.*** Table 2 illustrates the considerable impact of the proposed FU attacks on various evaluation metrics of the retrained model in FU scenarios. Specifically, TFUA markedly reduces the accuracy of the targeted class in Setup 1, while FFUA significantly biases the fairness of the retrain from the scratch baseline in Setup 2. In terms of the state-of-the-art FU method, INFOCOM'22, its performance aligns closely with the retrain from the scratch baseline, which is similarly affected by both FU attacks. This outcome is expected, as the primary goal of a standard FU process, as outlined in Definition 3.1, is to ensure the unlearned model remains as same as possible to one that is retrained from the scratch. Therefore, if FU attacks can undermine the retrain from the scratch baseline, existing FU methods that use this baseline as a reference benchmark will also be significantly influenced. In this case, it becomes evident that the retrain from the scratch baseline is no longer an appropriate reference within the context of robust FU, and existing FU methods lack the capability to address the threats posed by FU attacks. For the robust retraining methods adhering to the robust FU objective outlined in Definition 3.2, Table 2 (a) and (b) demonstrate that Byzantine-robust retraining strategies are capable of increasing the robustness of the retrained model against TFUA, while fairness algorithms effectively mitigate biases introduced by FFUA. Given the assumption that FU attacks do not impact the retraining process, these outcomes align with our expectations. However, as illustrated in Table 2 (c) and (d), while these robust retraining approaches achieve robustness within their specific domains (*e.g.*, accuracy and fairness), they struggle to universally cope with more complex scenarios where multiple FU attacks occur simultaneously within a single FU process. In addition,

Table 2: Efficacy of Various Defense Strategies Against Federated Unlearning Attacks.

Methods	Setup 1: TFUA only (Table 1 (a))							
	Adult				COMPAS			
	IID		non-IID		IID		non-IID	
	T.Acc.↑	G.Acc.↑	T.Acc.↑	G.Acc.↑	T.Acc.↑	G.Acc.↑	T.Acc.↑	G.Acc.↑
No Unlearning	0.6329	0.8625	0.6306	0.8610	0.5652	0.6680	0.5603	0.6623
Retrain from the Scratch	0.4623	0.8531	0.4587	**0.8527**	0.2285	0.6279	0.2144	0.6115
FU(INFOCOM'22)	0.4631	**0.8548**	0.4603	0.8525	0.2299	0.6306	0.2193	0.6168
Robust Retrain(Krum)	**0.6047**	0.8478	**0.6008**	0.8452	**0.4748**	0.6331	**0.4685**	**0.6353**
Robust Retrain(Mean)	0.6029	0.8461	0.5981	0.8449	0.4531	**0.6372**	0.4508	0.6339
Robust Retrain(Median)	0.5994	0.8455	0.5992	0.8428	0.4355	0.6290	0.4340	0.6293
Robust Retrain(Fairness)	0.4604	0.8516	0.4588	0.8499	0.2311	0.6286	0.2227	0.6214
robustFU(ours)	**0.6297**	**0.8614**	**0.6272**	**0.8597**	**0.4938**	**0.6445**	**0.4902**	**0.6419**

Methods	Setup 2: FFUA only (Table 1 (b))											
	Adult						COMPAS					
	IID			non-IID			IID			non-IID		
	AEOD↓	ASPD↓	G.Acc.↑	AEOD↓	ASPD↓	G.Acc.↑	AEOD↓	ASPD↓	G.Acc.↑	AEOD↓	ASPD↓	G.Acc.↑
No Unlearning	0.0015	0.0191	0.8611	0.0019	0.0205	0.8602	0.0382	0.0198	0.6634	0.0424	0.0235	0.6592
Retrain from the Scratch	0.0585	0.0642	0.8481	0.0615	0.0712	0.8464	0.0865	0.1264	0.6461	0.0972	0.1464	0.6346
FU(INFOCOM'22)	0.0578	0.0634	0.8478	0.0609	0.0696	0.8438	0.0821	0.1236	**0.6467**	0.0959	0.1396	0.6369
Robust Retrain(Byzantine)	0.0559	0.0573	0.8450	0.0570	0.0613	0.8427	0.0589	0.1052	0.6408	0.0740	0.1069	0.6367
Robust Retrain(FairFL)	0.0102	0.0311	**0.8502**	0.0139	0.0344	**0.8487**	0.0504	0.0351	0.6402	0.0608	**0.0568**	0.6356
Robust Retrain(FairFed)	**0.0049**	**0.0247**	0.8496	**0.0065**	**0.0252**	0.8455	**0.0467**	**0.0237**	0.6396	**0.0545**	**0.0424**	**0.6371**
robustFU(ours)	**0.0067**	**0.0289**	**0.8592**	**0.0071**	**0.0294**	**0.8570**	**0.0498**	**0.0289**	**0.6480**	**0.6010**	0.0571	**0.6461**

Methods	Setup 3: TFUA+FFUA(small) (Table 1 (c))											
	Adult						COMPAS					
	IID			non-IID			IID			non-IID		
	T.Acc.↑	AEOD↓	G.Acc.↑	T.Acc.↑	AEOD↓	G.Acc.↑	T.Acc.↑	AEOD↓	G.Acc.↑	T.Acc.↑	AEOD↓	G.Acc.↑
No Unlearning	0.6337	0.0023	0.8602	0.6325	0.0027	0.8595	0.5633	0.0356	0.6615	0.5587	0.0391	0.6590
Retrain from the Scratch	0.4829	0.0456	0.8487	0.4805	0.0494	**0.8465**	0.3112	0.0808	0.6254	0.2976	0.0881	0.6214
FU(INFOCOM'22)	0.4844	0.0447	**0.8492**	0.4825	0.0489	0.8463	0.3147	0.0774	0.6266	0.3051	0.0853	0.6189
Robust Retrain(Fairness)	0.4571	**0.0049**	0.8481	0.4640	**0.0054**	0.8457	0.3170	**0.0461**	0.6279	0.3099	**0.0516**	0.6233
Robust Retrain(Byzantine)	**0.6102**	0.0409	0.8467	**0.6076**	0.0422	0.8443	**0.4729**	0.0659	**0.6343**	**0.4635**	0.0697	**0.6326**
robustFU(ours)	**0.6286**	**0.0056**	**0.8590**	**0.6241**	**0.0060**	**0.8588**	**0.5103**	**0.0480**	**0.6421**	**0.5075**	**0.0522**	**0.6409**

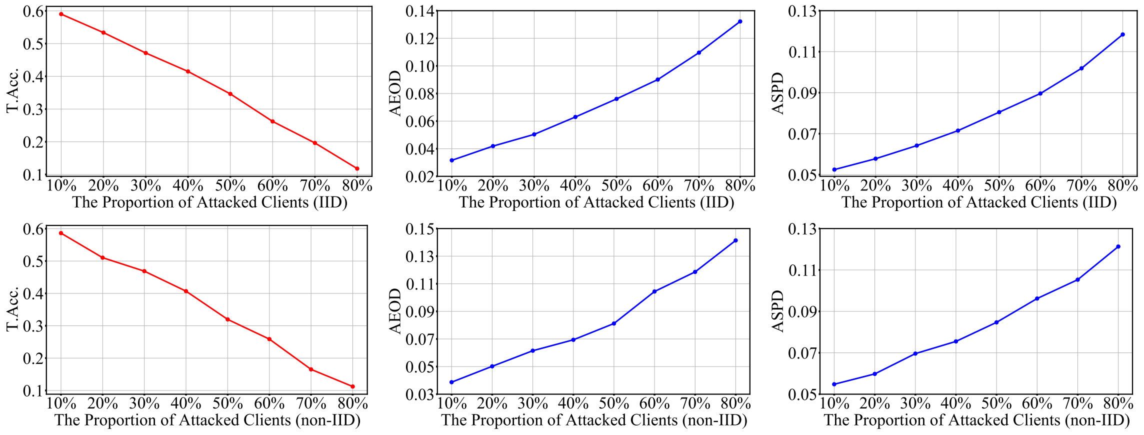
Methods	Setup 4: TFUA+FFUA(large) (Table 1 (d))											
	Adult						COMPAS					
	IID			non-IID			IID			non-IID		
	T.Acc.↑	AEOD↓	G.Acc.↑	T.Acc.↑	AEOD↓	G.Acc.↑	T.Acc.↑	AEOD↓	G.Acc.↑	T.Acc.↑	AEOD↓	G.Acc.↑
No Unlearning	0.6337	0.0023	0.8602	0.6325	0.0027	0.8595	0.5633	0.0356	0.6615	0.5587	0.0391	0.6590
Retrain from the Scratch	0.4548	0.0654	**0.8452**	0.4529	0.0693	0.8422	0.2047	0.1047	0.6104	0.1996	0.1135	0.6075
FU(INFOCOM'22)	0.4560	0.0632	0.8451	0.4535	0.0677	**0.8426**	0.2111	0.1009	0.6128	0.2052	0.1113	0.6092
Robust Retrain(Fairness)	0.4477	**0.0063**	0.8444	0.4428	**0.0072**	0.8407	0.2240	**0.0482**	**0.6308**	0.2186	**0.0550**	**0.6283**
Robust Retrain(Byzantine)	**0.5937**	0.0516	0.8439	**0.5914**	**0.0533**	0.8414	**0.4462**	0.0861	0.6294	**0.4269**	0.0894	0.6258
robustFU(ours)	**0.6216**	**0.0070**	**0.8587**	**0.6193**	**0.0072**	**0.8561**	**0.5052**	**0.0537**	**0.6401**	**0.5019**	**0.0598**	**0.6377**

in Section 5.3.2, we further show that these robust retraining methods typically require more time for retraining (exceeding the retrain from the scratch baseline), signifying a substantial compromise on the efficiency objective of FU.

5.2.2 *The Stealthiness of FU Attacks.*

The proposed FU attacks not only substantially impact different evaluation metrics of the retrained model, but also exhibit a high level of stealthiness. It can be seen from Table 2 that while the TFUA and FFUA notably decrease the accuracy of the targeted class and bias the fairness metrics of the retrained model, they do not cause a significant reduction in the overall accuracy of the attacked model. Specifically, in the case of TFUA, while the performance of the targeted class is diminished, its impact on non-targeted classes is minimized[3]. This characteristic closely mirrors that of targeted poisoning attacks [42] in FL. For

[3] An intriguing finding from our experiments is that, when TFUA is applied to binary classification problems, decreasing the accuracy of the targeted class typically results in an increased accuracy for the non-targeted class. In addition, the stealthiness of the proposed TFUA is anticipated to be more pronounced in multi-class classification problems, as validated by our experiments on the MNIST [26] and FashionMNIST [50] datasets in Section 5.5.3.



(a) TFUA (T.Acc.)

(b) FFUA (AEOD)

(c) FFUA (ASPD)

Figure 5: Impact of Attacked Client Proportions on TFUA and FFUA in the Adult Dataset.

FFUA, it also maintains a high degree of stealthiness since it only disrupts the relationship between the sensitive attribute and the label, leaving the overall joint distribution between other features and the label unaffected.

5.2.3 ***Ablation Studies of FU Attacks.*** To thoroughly examine the performance of FU attacks, we conduct two additional ablation studies to explore the impact of the proportions of attacked clients and the quantities of removed samples on FU attacks. Figure 5 illustrates the impact of the proportions of attacked clients on TFUA and FFUA, demonstrating that the efficacy of both FU attacks is directly proportional to the proportion of the attacked clients. To better understand the impact of the quantities of data removed on both FU attacks, we modify our experimental setups by setting the total number of clients to $n = 5$ and the number of attacked clients to $m = 1$. Figure 3 (a) and (b) showcase the effects of data removal quantities on TFUA and FFUA[4], respectively. It can be seen that the effectiveness of both attacks escalates as the quantities of removed samples increase. Particularly, when the number of removed data samples P and Q approaches their respective maximum values $MAX_i^t = 1619$ and $MAX_i^{+-} = 3203$, the attack efficiency of both TFUA and FFUA significantly accelerates (indicated by a steeper slope in both sub-figures). It's noteworthy that the retraining process, which follows the standard aggregation rule in FedAvg [32], assigns lower weights to model updates from an attacked client as more samples are removed. Despite this, the quantity of data removed remains the dominant factor influencing the efficacy of both attacks, underscoring the critical role of information loss in both TFUA and FFUA. This ablation study offers valuable insights and inspirations for the design our robust FU framework, as discussed in Section 4.2.3.

5.3 Evaluation of robustFU

5.3.1 ***The Effectiveness of robustFU.*** Table 2 showcases the effectiveness of the proposed robustFU framework. Results from Table 2 (a) indicate that robustFU excels in defending against TFUA, surpassing the performance of all Byzantine-robust methods. Table 2 (b) demonstrates that robustFU notably enhances the fairness of the unlearned model under FFUA, achieving a close performance of state-of-the-art FL fairness algorithms. Additionally, it can be seen from Table 2 (c) and (d) that, while robust retraining strategies are capable of addressing FU attacks within their limited domains, robustFU stands out by offering a universal defense against a wide range of FU attacks targeting different evaluation metrics throughout the FU process. Furthermore, the proposed robustFU not only significantly improves the robustness of the unlearned model but also enhances its utility. As shown in Table 2, robustFU achieves the highest overall accuracy (G.Acc.) across all benchmark methods in every experimental setting, demonstrating its exceptional effectiveness in addressing FU attacks under adversarial environments.

5.3.2 ***The Efficiency of robustFU.*** To demonstrate the efficiency of robustFU, we compare the average number of communication rounds and running time (in seconds), including standard deviations, required by each benchmark method and robustFU to reach a specified threshold under various attack scenarios in the Adult dataset. Specifically, Table 3 showcases the average communication rounds and runtime for Byzantine-robust methods and robustFU to attain a T.Acc. of 0.6 and a G.Acc. of 0.845 in the context of TFUA (Setup 1). Here, robustFU is shown to provide a 2.34× acceleration over the fastest Byzantine-robust retraining method (Krum). For FFUA (Setup 2), we set thresholds at AEOD = 0.01 and G.Acc. = 0.85. Table 4 reveals that robustFU achieves a 4.4× speedup in retraining compared to the fastest fairness algorithm (FairFed). Table 5 compares the efficiency of the retrain from the scratch baseline with the proposed robustFU in reaching a G.Acc. of 0.845 under attacks from both TFUA and FFUA (Setup 3). Results indicate that robustFU offers a 2.53× acceleration in retraining time.

It is noteworthy that, despite robustFU incurring additional inference time on the server and extra transmission overhead for

[4] In FFUA, we fix $Q' = MAX_i^{-+}$ while adjusting the number of Q.

Table 3: Comparing the Efficiency of Byzantine-Robust Methods and robustFU Against TFUA.

Methods	Communication Rounds	Running Time(s)
Robust Retrain(Krum)	207.15±10.45	76.75±1.04
Robust Retrain(Mean)	231.39±9.37	78.23±0.79
Robust Retrain(Median)	212.52±12.04	77.62±1.51
robustFU(ours)	**42.45±4.48**	**32.78±0.34**

Table 4: Comparing the Efficiency of Fairness Algorithms and robustFU Against FFUA.

Methods	Communication Rounds	Running Time(s)
Robust Retrain(FairFL)	65.81±9.14	123.58±1.38
Robust Retrain(FairFed)	78.20±6.82	89.47±0.85
robustFU(ours)	**28.25±4.09**	**20.32±0.27**

Table 5: Comparing the Efficiency of the Baseline Methods and robustFU Against FU Attacks.

Methods	Communication Rounds	Running Time(s)
Retrain from the Scratch	125.49±6.55	42.74±0.53
robustFU(ours)	**23.63±4.13**	**16.91±0.26**

sending compensation samples to each client[5], it remains substantially faster than all robust retraining methods and the retrain from the scratch baseline. This efficiency gain is attributed to the significantly reduced number of communication rounds required by robustFU, as shown in Tables 3, 4, and 5. As a result, while many existing FU methods utilize the server's storage to expedite the retraining process [28] [57], our approach leverages the server's strong computational power to speed up the retraining process.

5.3.3 *The Certified Removal of robustFU.*

We employ the two most commonly used attack-based FU verification methods, specifically membership inference attacks (MIA) [39] and backdoor attacks (BA) [27], to confirm that the data requested for removal has been successfully unlearned in robustFU.

In the *MIA-based verification*, we employ the strategy of training shadow models [39] to generate the training data for an attack model. This attack model is then used to determine whether a data sample belongs to the training dataset of the target model.

As can be seen from Table 6 (MIA), the precision of MIA is high for the trained model $\mathcal{M}$ prior to unlearning across both datasets. However, after unlearning, the attack precision for both the retrained model $\overline{\mathcal{M}}$ using the retrain from the scratch baseline and the unlearned model $\tilde{\mathcal{M}}$ generated by robustFU drops to approximately 0.5, indicative of the MIA merely making random guesses about sample membership. Hence, this demonstrates the effectiveness of robustFU in eliminating the influence of the removed samples from the trained models.

For *BA-based verification*, we inject backdoor patterns into 30% of the data samples across 30% of the clients during the FL phase.

Table 6: Certified Removal Performance of robustFU with Attack-Based Verification Methods.

Method	MIA Precision		BA Accuracy	
	Adult	COMPAS	Adult	COMPAS
Trained Model	0.7332	0.7068	0.9997	1.0000
Retrain from the Scratch	0.5076	0.4990	0.0539	0.0000
robustFU(ours)	0.5091	0.5028	0.0533	0.0000

Then, in the retraining phase, we request the removal of these manipulated samples within the attacked clients. Specifically, for the Adult dataset, we modify the *native-country* attribute to a rare value, *Honduras*, as the backdoor trigger and change the prediction class of these samples to $> 50K$. In the COMPAS dataset, we set the numeric attribute *Number_of_Priors* to 38, an extreme value within its range, to serve as the backdoor pattern, and adjust the prediction of these samples to 0 (*i.e.*, no recidivism within two years).

Table 6 (BA) illustrates the attack accuracy of BAs when the backdoor triggers are activated during the testing phase. We can see that BAs achieve extremely high accuracy on both datasets for the trained model $\mathcal{M}$ during the FL phase. However, the proposed robustFU effectively neutralizes the impact of BAs on the retrained model $\tilde{\mathcal{M}}$. This is evidenced by the attack success rate being close to that of the retrain from the scratch baseline in the Adult dataset and a reduction to 0% in the attack success rate for the COMPAS dataset.

5.4 Ablation Studies of robustFU

5.4.1 *The Contribution for Each Component.*

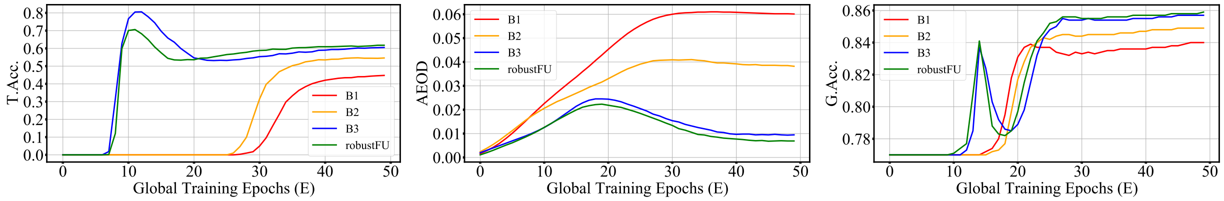
We carry out ablation studies focusing on two critical components of robustFU: the dynamic conflict sample compensation algorithm and the global reweighting mechanism. Specifically, we evaluate and compare performance across the following configurations: Retrain from the Scratch baseline **(B1)**, employing only the global reweighting mechanism **(B2)**, employing only the dynamic conflict sample compensation algorithm **(B3)**, and the complete robustFU framework.

Figure 6 demonstrates the comparative outcomes for T.Acc., AEOD, and G.Acc. metrics across these configurations under Attack Setup 4 within the Adult dataset. It can be seen that, both the dynamic conflict sample compensation algorithm (blue lines) and the global reweighting mechanism (yellow lines) contribute to the robustness gain of the retrained model compared to the retrain from the scratch baseline (red lines). While the dynamic conflict sample compensation algorithm yields greater improvements in the model's robustness, the integration of the global reweighting mechanism within robustFU (green lines) further increases the model's *robustness* (evidenced by higher T.Acc. and lower AEOD of the green lines in Figure 6 (a) and (b)), improves the *utility* of the retrained model (indicated by higher G.Acc. of the green line in Figure 6 (c)), and enhances the *efficiency* of the retraining process (shown through the fewer global training epochs required by the green lines to attain a performance enhancement comparable to that of the blue lines).

5.4.2 *The Impact of Randomly Generated Sample Size.*

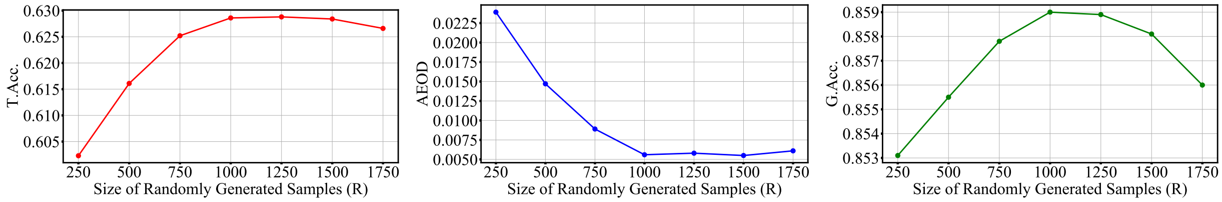
To determine the optimal hyperparameter R for the size of the randomly

[5] Although adding compensation samples increases the data volume for each participating client, the local updating time remains unaffected due to the fixed number of local batches B.



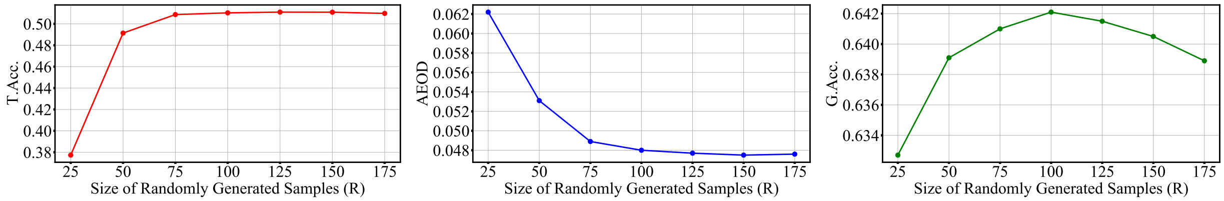
(a) T.Acc. (b) AEOD (c) G.Acc.

Figure 6: Impact of Different Components on the Performance of robustFU During Retraining.



(a) T.Acc. (b) AEOD (c) G.Acc.

Figure 7: Impact of the Number of Randomly Generated Samples on the Performance of robustFU in the Adult Dataset (under Attack Setup 3).



(a) T.Acc. (b) AEOD (c) G.Acc.

Figure 8: Impact of the Number of Randomly Generated Samples on the Performance of robustFU in the COMPAS Dataset (under Attack Setup 3).

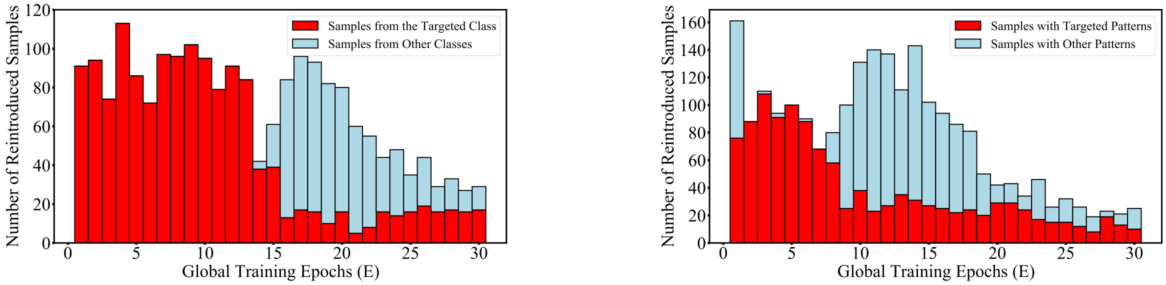
generated samples in robustFU, we conduct further ablation studies in our experiments. Aiming to reintroduce only a small proportion of conflict samples to each client during retraining (i.e., the reintroduced samples constitute only a small fraction of each client's local training data), we explore different ranges of R, tailored to datasets with different data volume. Specifically, we consider $R \in [250, 1750]$ for the Adult dataset and $R \in [25, 175]$ for the COMPAS dataset.

Figure 7 and 8 demonstrate the experimental results for the Adult dataset and the COMPAS dataset, respectively. We can observe that increasing the number R to a sufficient level significantly enhances the robustness of the retrained model, as indicated by the rising T.Acc. in Figure 7, 8 (a) and the decreasing AEOD in Figure 7, 8 (b). However, as demonstrated in Figures 7, 8 (c), continuing to increase the value of R can lead to a decrease in the utility of the retrained model. This reduction is due to the fact that the reintroduced conflict samples are labeled by the trained model $\mathcal{M}$ prior to unlearning. Although the trained model $\mathcal{M}$ can be of high utility, it still cannot guarantee that all reintroduced samples are correctly labeled. Consequently, adding excessive numbers of these samples to each client's dataset during retraining can diminish the overall accuracy of the retrained model. Thus, we conclude that the value of R should not be set too low or too high, and we suggest a provisional range for $R \in [\frac{|\mathcal{D}|}{5n}, \frac{|\mathcal{D}|}{3n}]$, depending on the average data volume per client $\frac{|\mathcal{D}|}{n}$.

Table 7: Comparison Between Random and Unlabeled Samples.

Methods	T.Acc.	AEOD	G.Acc.	Running Time(s)
Random Samples	0.6216	0.0070	0.8587	29.37+0.27
Unlabeled Samples	**0.6284**	**0.0062**	**0.8599**	**28.56+0.26**

5.4.3 ***The Impact of the Randomly Generated Samples.*** The incorporation of randomly generated samples plays a crucial role in the robustFU framework, which is specifically designed for scenarios demanding strict privacy preservation where the server has no access to any public or shared data. However, it is interesting to explore how robustFU performs when the server has access to a small amount of unlabeled data samples. We conduct our experiment on the Adult dataset, setting aside 5000 data samples as the unlabeled ones on the server. During each aggregation round, 1000



(a) Client Attacked by TFUA.

(b) Client Attacked by FFUA.

Figure 9: Visualization of the Number of Reintroduced Samples During Retraining for Clients Attacked by FU TFUA and FFUA.

Table 8: Effectiveness of robustFU in Defending Against TFUA Targeting Various Classes.

Method	Yeast: Targeted Class (T.Acc.)	
	MIT	NUC
No Attack	0.5781	0.6058
Retrain from the Scratch	0.4132	0.2396
robustFU(ours)	**0.5593**	**0.5662**

Table 9: Effectiveness of robustFU in Defending Against FFUA Targeting Various Sensitive Attributes.

Method	Adult: Sensitive Attribute (AEOD/ASPD)	
	Race	Gender
No Attack	0.0347/0.0615	0.0025/0.0211
Retrain from the Scratch	0.0982/0.1149	0.0531/0.0616
robustFU(ours)	**0.0510/0.0754**	**0.0089/0.0303**

of these samples are randomly chosen to substitute the initially randomly generated samples within the robustFU configuration.

Table 7 compares the performance of the robustFU framework when adopting randomly generated samples versus a small set of unlabeled samples (under Attack Setup 4 with IID data distribution). It's evident that the robustFU framework's performance is further enhanced by utilizing a small amount of unlabeled samples on the server. This improvement stems from the fact that the randomly generated samples might not accurately reflect the overall training data's distribution, whereas the unlabeled samples conform to the distribution of the overall training set. Furthermore, Section 5.5.3 demonstrates that the robustFU framework can be readily extended to image classification tasks if there is a small set of unlabeled samples available on the server.

5.5 Deeper Insights into robustFU

5.5.1 ***Visualization of the Dynamic Conflict Sample Compensation Algorithm***. To gain deeper insights into the proposed dynamic conflict sample compensation algorithm within robustFU, we visualize the number of reintroduced samples for clients affected by various FU attacks during the retraining process. Specifically, Figures 9 (a) and (b) illustrate the total number of reintroduced samples from global epoch 1 to 30 for clients targeted by TFUA and FFUA, respectively. In Figure 9 (a), red bars indicate the number of samples from the targeted class y_t in TFUA, while blue bars show samples from other classes. In Figure 9 (b), red bars denote samples with targeted patterns (i.e., the sum of samples of the form $(\boldsymbol{x}^+, y^-)$ and $(\boldsymbol{x}^-, y^+)$), while blue bars depict samples with other patterns.

From both figures, it can be seen that in the first few global epochs, samples with the specific patterns removed by FU attacks (red bars) constitute all or the majority of the reintroduced samples. This confirms that by comparing conflict labels between the trained model prior to unlearning and the model update during retraining, robustFU can effectively identify the appropriate compensation samples for clients affected by different FU attacks. By reintroducing these compensation samples to the attacked clients, a significant information gain is provided to those clients. Additionally, we can see that as the retrained model begins to converge, the total number of reintroduced samples decreases, indicating that the retrained model $\tilde{\mathcal{M}}$ is aligning more closely with the trained model $\mathcal{M}$ prior to unlearning.

5.5.2 ***The Exceptional Universality of robustFU***. While the aforementioned experiments focus on just one targeted class in TFUA and one sensitive attribute in FFUA, in more realistic and sophisticated settings, attackers might target multiple classes and bias more than one sensitive attribute within a single FU process. To investigate the effectiveness of robustFU under these conditions, we conduct additional experiments for both types of FU attacks. In the case of FFUA, we use the Adult dataset and consider race (Black vs. non-Black) and gender (Male vs. Female) as the sensitive attributes, with non-Black and Male to be the targeted demographic group. For TFUA, given the binary classification format of both the Adult and COMPAS datasets, we adopt the Yeast dataset [34], a multi-label classification dataset that predicts proteins' cellular localization sites. Within the Yeast dataset, the classes "MIT" and "NUC" are designated as the targeted classes.

Tables 8 and 9 summarize the experimental results for TFUA and FFUA, respectively. We can see that robustFU effectively increasing the accuracy for the targeted classes in TFUA and reduces biases towards the sensitive attributes in FFUA. This confirms the strong universality of the robustFU framework, which not only effectively defends against various FU attacks (as discussed in Section 5.3.1),

Table 10: Performance of robustFU in Image Classification Tasks.

Method	MNIST		FashionMNIST	
	T.Acc.	G.Acc.	T.Acc.	G.Acc.
No Unlearning	0.9810	0.9749	0.9636	0.9015
Retrain from the Scratch	0.7331	0.9713	0.6364	0.8798
robustFU(ours)	**0.9765**	**0.9742**	**0.9455**	**0.8926**

but is also agnostic to the targeted class in TFUA and the sensitive attribute in FFUA.

5.5.3 ***Extension to Image Classification Tasks.*** The effectiveness of the proposed robustFU extends beyond tabular classification tasks. In this section, we illustrate how robustFU can be readily adapted to image classification tasks using a small set of unlabeled data available on the server. We set up our experiments using two widely investigated image classification datasets: MNIST [26], which is used for classifying handwritten digits, and FashionMNIST [50], which is used for classifying fashion items. We select TFUA as the FU attack for both image datasets, targeting the digit 7 in the MNIST dataset and the *Sneaker* category in the FashionMNIST dataset. For both datasets, we employ a convolutional neural network (CNN) with two convolutional layers. 2000 samples are set aside from the test sets of both datasets to serve as the unlabeled samples that replace the randomly generated samples in the robustFU framework.

Table 10 presents our experimental results, showing that for both datasets, while TFUA significantly compromises the accuracy of the targeted class in the retrained model of the retrain from the scratch baseline, the proposed robustFU effectively enhances the robustness of the retrained model by not only increasing the accuracy of the targeted class but also improving its overall accuracy.

6 CONCLUSION AND FUTURE WORKS

In this study, we pioneer exploring the robustness of FU algorithms. We introduce two innovative FU attacks that significantly compromise different evaluation metrics of the retrained model through crafted malicious data removal requests. In response, we propose robustFU, an advanced robust FU framework that incorporates a novel dynamic sample compensation algorithm and a global reweighting mechanism. This framework is designed to offer universal protection against a wide array of FU attacks while fulfilling the broader objectives of FU. Through comprehensive experimentation, robustFU proves to be exceptionally effective in defending against FU attacks, making it as a formidable solution to enhance the robustness of the FU process. We hope our work can pave the way for subsequent research into FU robustness and contribute to the development of data privacy and protection laws.

For future works, considering the preliminary success of applying robustFU to image tasks given a set of unlabeled data as demonstrated in Section 5.5.3, a promising direction would be to explore ways to eliminate the need for such unlabeled data, for instance, using GAN-based generative models [15]. Additionally, while our work primarily concentrates on the robustness of sample-level FU, investigating the robustness of class-level and client-level FU remains an open problem.

7 ACKNOWLEDGEMENT

This research is supported by an Australian Government Research Training Program (RTP) Scholarship.

REFERENCES

[1] Sawsan AbdulRahman, Hanine Tout, Hakima Ould-Slimane, Azzam Mourad, Chamseddine Talhi, and Mohsen Guizani. 2020. A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. *IEEE Internet of Things Journal* 8, 7 (2020), 5476–5497.

[2] Thomas Baumhauer, Pascal Schöttle, and Matthias Zeppelzauer. 2022. Machine unlearning: Linear filtration for logit-based classifiers. *Machine Learning* 111, 9 (2022), 3203–3226.

[3] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. 2019. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*. PMLR, 634–643.

[4] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. 2017. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In *Advances in Neural Information Processing Systems*.

[5] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2021. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 141–159.

[6] Jonathan Brophy and Daniel Lowd. 2021. Machine unlearning for random forests. In *International Conference on Machine Learning*. PMLR, 1092–1104.

[7] Yinzhi Cao and Junfeng Yang. 2015. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*. IEEE, 463–480.

[8] Tianshi Che, Yang Zhou, Zijie Zhang, Lingjuan Lyu, Ji Liu, Da Yan, Dejing Dou, and Jun Huan. 2023. Fast federated machine unlearning with nonlinear functional theory. In *International conference on machine learning*. PMLR, 4241–4268.

[9] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. 2022. Graph unlearning. In *Proceedings of the 2022 ACM SIGSAC conference on computer and communications security*. 499–513.

[10] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository.

[11] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through Awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*.

[12] Yahya H. Ezzeldin, Shen Yan, Chaoyang He, Emilio Ferrara, and Salman Avestimehr. 2023. FairFed: Enabling Group Fairness in Federated Learning. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*.

[13] Xiangshan Gao, Xingjun Ma, Jingyi Wang, Youcheng Sun, Bo Li, Shouling Ji, Peng Cheng, and Jiming Chen. 2024. Verifi: Towards verifiable federated unlearning. *IEEE Transactions on Dependable and Secure Computing* (2024).

[14] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. 2020. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9304–9312.

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems* 27 (2014).

[16] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. 2019. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030* (2019).

[17] Varun Gupta, Christopher Jung, Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi, and Chris Waites. 2021. Adaptive machine unlearning. *Advances in Neural Information Processing Systems* 34 (2021), 16319–16330.

[18] Anisa Halimi, Swanand Ravindra Kadhe, Ambrish Rawat, and Nathalie Baracaldo Angel. 2022. Federated Unlearning: How to Efficiently Erase a Client in FL?. In *International Conference on Machine Learning*.

[19] E. Harding, J. J. Vanto, R. Clark, L. H. Ji, and S. C. Ainsworth. 2019. Understanding the scope and impact of the California Consumer Privacy Act of 2018. *Journal of Data Protection & Privacy* (2019).

[20] Moritz Hardt, Eric Price, and Nathan Srebro. 2016. Equality of Opportunity in Supervised Learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS)*.

[21] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

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

[23] Zachary Izzo, Mary Anne Smart, Kamalika Chaudhuri, and James Zou. 2021. Approximate data deletion from machine learning models. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2008–2016.

[24] Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. 2024. Towards unbounded machine unlearning. *Advances in Neural Information Processing Systems* 36 (2024).

[25] Larson, Jeff, Surya Mattu, Lauren Kirchner, and Julia Angwin. 2016. How We Analyzed the COMPAS Recidivism Algorithm.

[26] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[27] Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. 2022. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

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

[29] Yi Liu, Lei Xu, Xingliang Yuan, Cong Wang, and Bo Li. 2022. The right to be forgotten in federated learning: An efficient realization with rapid retraining. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 1749–1758.

[30] Ziyao Liu, Yu Jiang, Jiyuan Shen, Minyi Peng, Kwok-Yan Lam, and Xingliang Yuan. 2023. A survey on federated unlearning: Challenges, methods, and future directions. *arXiv preprint arXiv:2310.20448* (2023).

[31] Neil G Marchant, Benjamin IP Rubinstein, and Scott Alfeld. 2022. Hard to forget: Poisoning attacks on certified machine unlearning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 7691–7700.

[32] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*.

[33] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A Survey on Bias and Fairness in Machine Learning. *ACM Comput. Surv.* (2021).

[34] Kenta Nakai. 1996. Yeast. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5KG68.

[35] Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. 2021. Descent-to-delete: Gradient-based methods for machine unlearning. In *Algorithmic Learning Theory*. PMLR, 931–962.

[36] Wei Qian, Chenxu Zhao, Wei Le, Meiyi Ma, and Mengdi Huai. 2023. Towards understanding and enhancing robustness of deep learning models against malicious unlearning attacks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1932–1942.

[37] Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal* 27, 3 (1948), 379–423.

[38] Virat Shejwalkar and Amir Houmansadr. 2021. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *NDSS*.

[39] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*. IEEE, 3–18.

[40] Ningxin Su and Baochun Li. 2023. Asynchronous federated unlearning. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*. IEEE, 1–10.

[41] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. 2019. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963* (2019).

[42] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. 2020. Data Poisoning Attacks Against Federated Learning Systems. In *Computer Security – ESORICS 2020*.

[43] Paul Voigt and Axel von dem Bussche. 2017. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer, Cham, Switzerland.

[44] Fei Wang, Baochun Li, and Bo Li. 2023. Federated unlearning and its privacy threats. *IEEE Network* (2023).

[45] Junxiao Wang, Song Guo, Xin Xie, and Heng Qi. 2022. Federated unlearning via class-discriminative pruning. In *Proceedings of the ACM Web Conference 2022*. 622–632.

[46] Pengfei Wang, Zhaohong Yan, Mohammad S Obaidat, Zhiwei Yuan, Leyou Yang, Junxiang Zhang, Zongzheng Wei, and Qiang Zhang. 2023. Edge Caching with Federated Unlearning for Low-latency V2X Communications. *IEEE Communications Magazine* (2023).

[47] Zichen Wang, Xiangshan Gao, Cong Wang, Peng Cheng, and Jiming Chen. 2024. Efficient Vertical Federated Unlearning via Fast Retraining. *ACM Transactions on Internet Technology* 24, 2 (2024), 1–22.

[48] Chen Wu, SENCUN ZHU, and Prasenjit Mitra. 2023. Unlearning Backdoor Attacks in Federated Learning. In *ICLR 2023 Workshop on Backdoor Attacks and Defenses in Machine Learning*.

[49] Leijie Wu, Song Guo, Junxiao Wang, Zicong Hong, Jie Zhang, and Yaohong Ding. 2022. Federated Unlearning: Guarantee the Right of Clients to Forget. *IEEE Network* 36, 5 (2022), 129–135.

[50] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).

[51] Zuobin Xiong, Wei Li, Yingshu Li, and Zhipeng Cai. 2023. Exact-Fun: An Exact and Efficient Federated Unlearning Approach. In *2023 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1439–1444.

[52] Heng Xu, Tianqing Zhu, Lefeng Zhang, Wanlei Zhou, and Philip S. Yu. 2023. Machine Unlearning: A Survey. *ACM Comput. Surv.* (2023).

[53] Zhewei Yao, Amir Gholami, Sheng Shen, Mustafa Mustafa, Kurt Keutzer, and Michael Mahoney. 2021. Adahessian: An adaptive second order optimizer for machine learning. In *proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 10665–10673.

[54] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. 2018. Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates. In *Proceedings of the 35th International Conference on Machine Learning*.

[55] Xuefei Yin, Yanming Zhu, and Jiankun Hu. 2021. A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. *ACM Computing Surveys (CSUR)* 54, 6 (2021), 1–36.

[56] Daniel Yue Zhang, Ziyi Kou, and Dong Wang. 2020. FairFL: A Fair Federated Learning Approach to Reducing Demographic Bias in Privacy-Sensitive Classification Models. In *2020 IEEE International Conference on Big Data (Big Data)*.

[57] Lefeng Zhang, Tianqing Zhu, Haibin Zhang, Ping Xiong, and Wanlei Zhou. 2023. Fedrecovery: Differentially private machine unlearning for federated learning frameworks. *IEEE Transactions on Information Forensics and Security* (2023).

[58] Chenxu Zhao, Wei Qian, Rex Ying, and Mengdi Huai. 2024. Static and Sequential Malicious Attacks in the Context of Selective Forgetting. *Advances in Neural Information Processing Systems* 36 (2024).

[59] Yian Zhao, Pengfei Wang, Heng Qi, Jianguo Huang, Zongzheng Wei, and Qiang Zhang. 2023. Federated unlearning with momentum degradation. *IEEE Internet of Things Journal* (2023).

[60] Yuhao Zhou, Minjia Shi, Yuxin Tian, Qing Ye, and Jiancheng Lv. 2024. DeFTA: A plug-and-play peer-to-peer decentralized federated learning framework. *Information Sciences* 670 (2024), 120582.

[61] Xiangrong Zhu, Guangyao Li, and Wei Hu. 2023. Heterogeneous federated knowledge graph embedding learning and unlearning. In *Proceedings of the ACM Web Conference 2023*. 2444–2454.