VeriFi

VeriFi: Towards Verifiable Federated Unlearning

Xiangshan Gao
Zhejiang University
corazju@zju.edu.cn

Xingjun Ma
Fudan University
xingjunma@fudan.edu.cn

Jingyi Wang
Zhejiang University
wangjyee@zju.edu.cn

Youcheng Sun
University of Manchester
youcheng.sun@manchester.ac.uk

Bo Li
UIUC
lbo@illinois.edu

Shouling Ji
Zhejiang University
sji@zju.edu.cn

Peng Cheng
Zhejiang University
lunarheart@zju.edu.cn

Jiming Chen
Zhejiang University
cjm@zju.edu.cn

***Abstract*—Federated learning (FL) has emerged as a privacy-aware collaborative learning paradigm where participants jointly train a powerful model without sharing their private data. One desirable property for FL is the implementation of the *right to be forgotten (RTBF)*, i.e., a leaving participant has the right to request to delete its private data from the global model. Recently, several server-side unlearning methods have been proposed to remove a leaving participant's gradients from the global model. However, *unlearning itself may not be enough to implement RTBF unless the unlearning effect can be independently verified*, an important aspect that has been overlooked in the current literature. In this paper, we prompt the concept of *verifiable federated unlearning*, and propose VeriFi, a unified framework integrating federated unlearning and verification that allows systematic analysis of the unlearning and quantification of its effect, with different combinations of multiple unlearning and verification methods. In VeriFi, the leaving participant is granted the *right to verify (RTV)*, that is, the participant notifies the server before leaving, then actively verifies the unlearning effect in the next few communication rounds. The unlearning is done at the server side immediately after receiving the leaving notification, while the verification is done locally by the leaving participant via two steps: *marking* and *checking*. The marking step injects carefully-designed *markers* to fingerprint the leaving participant's data, while the checking step examines the change of the global model's performance on the markers.**

Based on VeriFi, we conduct the first systematic and large-scale study for verifiable federated unlearning, considering 7 unlearning methods and 5 verification methods that cover existing, adapted and newly proposed ones for both unlearning and verification. Particularly, the newly proposed methods include a more efficient and FL-friendly unlearning method ^{u}S2U, and two more effective and robust non-invasive-verification methods ^{v}FM and ^{v}EM (without training controllability or external data, without white-box model access or introducing security hazard). We extensively evaluate VeriFi on 7 datasets, including both (natural/facial/medical) images and audios, and 4 types of deep learning models, including both Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). Our analysis establishes important empirical understandings and evidence for more trustworthy federated unlearning.

I. Introduction

Federated learning (FL) is a collaborative learning paradigm that allows participants to train a powerful machine learning model jointly without sharing their private data [5], [25], [53]. This privacy-preserving nature of FL makes it an ideal choice for real-world privacy-sensitive collaborations in finance [50], healthcare [52], [7], insurance [46] and many other fields. One essential requirement of FL is the participants' *"right to be forgotten" (RTBF)*, which has been stated explicitly in the European Union General Data Protection Regulation (GDPR) [18], [37] and the California Consumer Privacy Act (CCPA) [23]. That is, a participant has the right to request a deletion of its private data. Arguably, one may worry that its private data will be memorized by the global model and continue to be exploited even after leaving the federation. As leaving/joining is a common behavior in FL, it is thus necessary to ensure that every participant can **join and leave the federation freely, and more importantly, with no concerns**. However, so far, participants have difficulty exercising the RTBF in existing FL frameworks, which might discourage potential participants to join the federation.

The concept of *machine unlearning* [6], [39] has recently been proposed to remove data from a machine learning model. Several unlearning methods are designed to actively unlearn certain data from a trained model. A simple yet costly approach for unlearning is to retrain the model from scratch with the requested data being removed from the training set [6]. It can be made more efficient if the model is trained on summarized (e.g., aggregates of summations) or partitioned subsets rather than individual training samples, in which case, the model only needs to be updated on the subset(s) associated with the requested samples [10], [19]. The above methods are less practical for large-scale datasets, although advanced data partitioning or intermediate model breakpoint strategies may help [6], [24]. More recently, machine unlearning has been extended to the FL setting, a.k.a., *federated unlearning* [30], which is arguably more challenging. In FL, 1) the global model is updated based on the aggregated rather than the raw gradients; 2) FL can have a large number of participants; and 3) different participants may have similar, or to some extent, shared training samples. Consequently, simple gradient-based methods such as subtracting the reconstructed or dummy gradients of the leaving participant may harm the original task or introduce new privacy threats into FL [30], [32].

Moreover, federated unlearning is only one side of the coin for the RTBF. A more concerning question from a participant's perspective is: **how to make sure that my data has indeed been forgotten, hopefully in a verifiable and measurable way,** which we believe is the core of establishing mutual trust in FL. Unfortunately, this important aspect has been largely overlooked in the FL literature. In traditional machine unlearning, the unlearning effect can be simply verified by the model's performance (e.g., accuracy and loss) on the unlearned data or additionally injected backdoor data [6], [10], [41]. However, in FL, the accuracy and loss may hardly change when only one or a few participants left the federation, owing to the contribution of other participants. Besides, it is not secure in FL to use backdoor solely for the purpose of unlearning verification as it might introduce new security threats into the commonly contributed and shared global model (see Appendix VI-D). So far, it still lacks understanding of how to *effectively* and *reliably* verify that the data has indeed been deleted after unlearning. In fact, due to the lack of a unified, holistic and all-round FL verification framework, several key fundamental questions for trustworthy RTBF in FL remain unexplored:

- *Federated Unlearning.* Is federated unlearning necessary or might natural forgetting be enough to forget the leaving participant's data?
- *Unlearning Verification.* Do we need more sophisticated methods or simple methods like checking the global model's performance on the leaving participant's data are enough to measure and verify the unlearning effect?
- *Practical Choice.* What are the most effective combination(s) of unlearning and verification methods that can effectively unlearn, clearly verify, while causing minimal negative impact on the original task?

To answer the above questions, in this paper, we promote the concept of *verifiable federated unlearning*, which treats verification as important as unlearning and grants the participant the *"right to verify" (RTV)*. Specifically, we design and implement VERIFI, a unified framework for verifiable federated unlearning. The core of VERIFI contains 1) a federated unlearning module; 2) a verification module with two key verification steps, namely *marking* and *checking*; and 3) a generic *unlearning-verification* mechanism applicable to common FL frameworks. Fig. 1 provides an overview of VERIFI. The unlearning module can be any unlearning[1] method adopted at the server size that erases the information of the leaving participant's data (which we call "leaving data"). The marking step of the verification module injects/tags specifically selected or designed patterns or training examples as *markers*. The checking step of the verification module then verifies the degree of unlearning based on different verification metrics defined w.r.t. the global model and the markers. The *unlearning-verification* mechanism integrates all the above steps into a chained pipeline and specifies when and what to mark, and who and when to unlearn and verify.

With VERIFI, we bring together a comprehensive set of unlearning and verification methods, including not only existing ones but also many adapted from other fields, as well as newly proposed in this paper. We conduct the first systematic study on the practicality of different combinations of unlearning and verification methods for verifiable federated unlearning. **For unlearning**, we study the limitations of existing one-step (e.g., differential privacy[2]) and multi-step (e.g., retraining and gradient subtraction) unlearning methods, such as high cost and significant negative impact on the original task. We also propose a more efficient and FL-friendly one-step unlearning method *scale-to-unlearn* (^{u}S2U)[3]. ^{u}S2U scales down the leaving participant's gradients/parameters to trigger the global model to erase its memorization of the participant. Verification consists of two steps: marking and checking. **For marking**, the existing method leverages backdoored samples to verify the unlearning effect [41], which is unsuitable for FL as backdoor methods are invasive methods that could introduce global threats to all FL participants. We consider this backdoor-based verification method in VERIFI as a comparison, and further propose two non-invasive unique memory-based methods. The two proposed verification methods verify the unlearning effect based on the sensitive performance of the global model on a specific subset of the leaving data. Moreover, we also adapt existing watermark and fingerprint methods proposed for deep learning intellectual property protection as verification methods for federated unlearning. We systematically analyze the pros and cons of these marking methods in VERIFI. **For checking**, we explore loss, accuracy, influence function (IF) [27] and Kullback–Leibler (KL) divergence [20] to measure the performance change on the marked data (i.e., markers) before and after unlearning. Our extensive evaluation and analyses provide answers to the three fundamental questions mentioned earlier, and establish the empirical foundation for verifiable and trustworthy federated unlearning.

In summary, our main contributions are:

- We design the first unlearning-verification framework VERIFI for verifiable federated unlearning. VERIFI grants FL participants the *right to verify*, i.e., the verification of the unlearning effect when leaving the federation. VERIFI introduces a unified mechanism that allows quantitative measurement on the effectiveness of different combinations of unlearning and verification methods.
- With VERIFI, we identify the limitations of existing unlearning and verification methods, and propose a more efficient and FL-friendly unlearning method ^{u}S2U and two more effective and robust non-invasive *unique memory* based verification methods (^{v}EM and ^{v}FM)[4]. The advantages of the three proposed methods are also demonstrated by our extensive experiments.
- With VERIFI, we systemically study 7 unlearning methods and 5 verification methods (i.e., 5 marking methods and 4 checking metrics) with both Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) on 7 datasets, including 3 natural image, 1 facial image, 1 audio and 2 medical image datasets. Our extensive study unveils the necessity, potentials and limitations of different federated unlearning and verification methods.

[1] Without ambiguity, we use "unlearning" instead of "federated unlearning" for simplicity.

[2] Although ^{u}DP cannot ensure completely zero memory in machine unlearning [6], we still explored its practical unlearning effect in federated unlearning for the purpose of completeness.

[3] We use the ^{u} symbol to indicate unlearning methods.

[4] We use the ^{v} symbol to indicate verification methods.

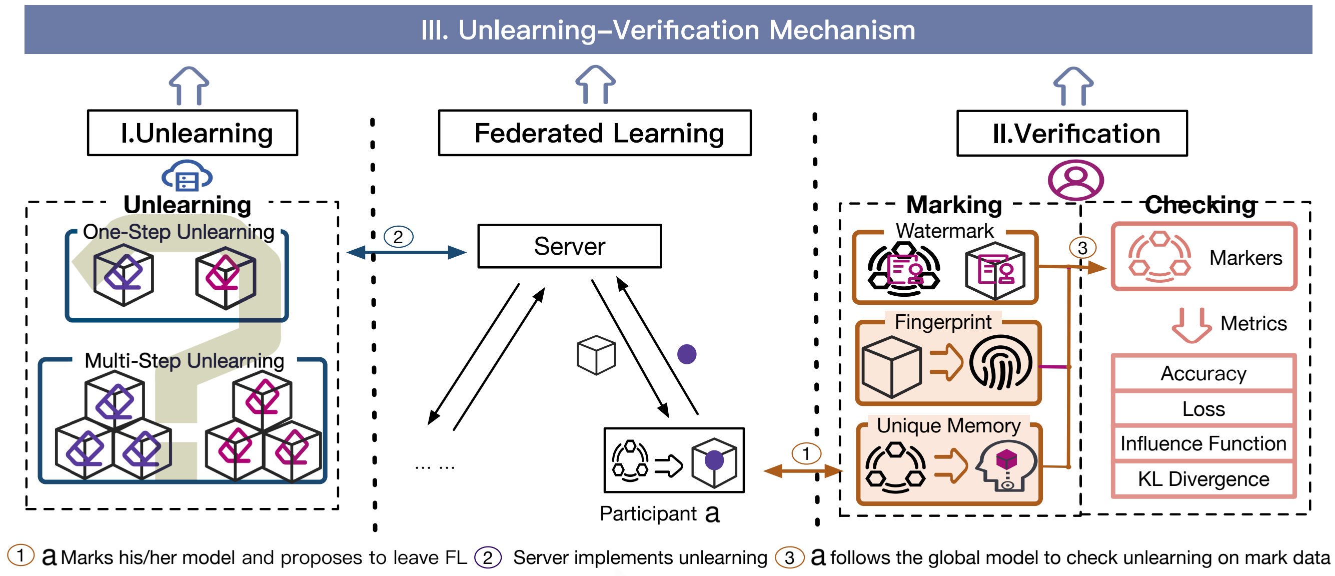


Fig. 1: Overview of the proposed VERIFI framework and its three key modules: 1) unlearning module; 2) verification module; and 3) unlearning-verification mechanism. The standard "Federated Learning" procedure is further illustrated in Fig. 2.

II. PRELIMINARIES

A. Federated Learning

In FL, a number of participants jointly train a global model by communicating gradients or model parameters with a central server. At each communication round, the participants download the global model from the server, perform a certain number of local updates on their private data, and then upload the accumulated local updates (gradients) to the server. The server then aggregates (e.g., using FedAvg [34]) the accumulated local updates to update the global model. The complete FL procedure is illustrated in Fig. 2. The participants' private data is protected during the entire FL process, as it never leaves the local devices.

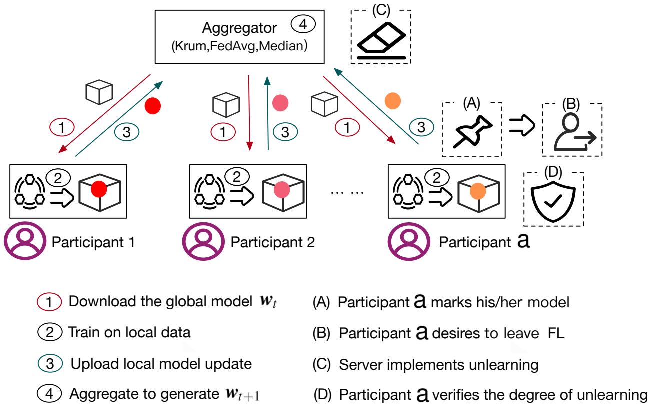

Aggregator 4
(Krum,FedAvg,Median)
(C)
(A)
(B)
(D)
1
2
3
Participant 1
Participant 2
… …
Participant a
1 Download the global model w_t
2 Train on local data
3 Upload local model update
4 Aggregate to generate w_{t+1}
(A) Participant a marks his/her model
(B) Participant a desires to leave FL
(C) Server implements unlearning
(D) Participant a verifies the degree of unlearning

Fig. 2: Collaboration and learning in federated learning.

Let $[n] = \{1,...,n\}$ be the set of n participants with each participant owning a private dataset D_i for $i \in [n]$, and $\mathscr{D} = D_1 \cup D_2 \cup \cdots D_n$ is the full training dataset. At the t-th communication round, the i-th participant first downloads the global model $\boldsymbol{w}_t$, and then performs local update(s), e.g., using Stochastic Gradient Descent (SGD), on the local data D_i to obtain an updated local model $\boldsymbol{w}_{t+1}^{(i)}$. The accumulated gradient, $\boldsymbol{w}_{t+1}^{(i)} - \boldsymbol{w}_t$, is then sent to the server for the global model update, e.g., using FedAvg [34] as follows:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \frac{1}{n}\sum_{i\in[n]}(\boldsymbol{w}_{t+1}^{(i)} - \boldsymbol{w}_t). \quad (1)$$

Besides FedAvg, other aggregation rules are proposed for Byzantine-robust FL: Krum [4], Median [54], Bulyan [22] and Trimmed Mean [54]. Meanwhile, FL can be either horizontal where the participants share the same feature space but own different data samples, or vertical where the participants share the same data sample IDs but possess different features. In this work, we focus on a typical horizontal FL setting with FedAvg, as defined in Eq. (1).

B. Federated Unlearning and Verification

Federated Unlearning. It has been shown that deep neural networks have both memorization and forgetting effects [55], [2], [26], i.e., they naturally memorize information about the training data and so naturally forget the removed data (from the training dataset) during training. Different from natural forgetting[5], machine unlearning explicitly forces a model to forget its memorization of a target (requested to delete) subset of training samples [6]. Intuitively, unlearning can be achieved by (re)training the model on the updated dataset with the requested samples removed. In traditional machine learning, this can be done via expensive retraining, or more efficient partition/breakpoint based learning with data partitions/aggregates [10], [19], [6], [24]. Noise can also be used to smooth out the memorization of particular samples [39]. However, in FL, information is shared via gradients. This motivates the two pioneering works [30], [32] in federated unlearning to

[5] The participant leaves with no active unlearning conducted by the server.

subtract the calibrated or generated gradients of the leaving participant to unlearn information. We will incorporate and test the two methods (as well as the costly retraining) in VERIFI and propose a more effective unlearning method for FL. More detailed analysis about the pros and cons of the unlearning methods in VERIFI can be found in Section III-B and IV.

Unlearning Verification. Intuitively, the effectiveness of unlearning can be verified by the change in the model's performance before and after unlearning. In existing works, loss or accuracy on the leaving data is often used to achieve the verification purpose [6]. Unlearning can also be verified on a set of backdoored training samples [41] obtained via a backdoor attack, which is essentially a data poisoning process that injects a trigger pattern into a small subset of training data so as to trick the model into memorizing the correlation between the pattern and a target class [21], [12]. Suppose the trigger pattern is $\boldsymbol{r}$ and its associated backdoor target class is y_{target}. Once the trigger is learned by the model f, the model will constantly predict the target class on any samples attached with the trigger pattern:

$$\arg\max f(\boldsymbol{x} \oplus \boldsymbol{r}) = y_{target},\ \forall(\boldsymbol{x}, y) \in \mathscr{D}, \tag{2}$$

where, the model f outputs the class probabilities, the operation $\boldsymbol{x} \oplus \boldsymbol{r}$ produces a backdoored version of $\boldsymbol{x}$, $(\boldsymbol{x}, y)$ is an input-label pair, and $\mathscr{D}$ is the training dataset in traditional machine learning. If the unlearning is effective, then the model will forget the backdoor correlation and predict the correct class instead:

$$\arg\max \overline{f}(\boldsymbol{x} \oplus \boldsymbol{r}) = y,\ \forall(\boldsymbol{x}, y) \in \mathscr{D}, \tag{3}$$

where $\overline{f}$ denotes the model obtained after unlearning and y is the correct class of $\boldsymbol{x}$.

Although several unlearning methods have been proposed, the challenge and potential issues of unlearning verification have not been thoroughly studied, especially in FL. In fact, [41] is the only work that has investigated the verification problem, however, it was conducted in traditional machine unlearning. It proposes to use backdoored samples to obtain more sensitive verification. Considering the high security risk (could backdoor all participants) of backdoor techniques, it is thus not ideal to use backdoor verification in FL.

We also adapt and study two plausible concepts from the deep learning intellectual property (IP) protection domain for unlearning verification: watermarking [45], [56] and fingerprinting [9]. Watermarking is an invasive technique that embeds owner-specific binary string or backdoor triggers into the model parameters to help determine the ownership of the model at a later (post-deployment) stage, while fingerprinting generates new samples to fingerprint the model's unique properties like decision boundary [9]. In this work, we specially design and adapt these two types of techniques for federated unlearning verification. More systematic analysis of different verification methods can be found in Section III-C and IV-C.

III. PROPOSED VERIFI FRAMEWORK

In this section, we present our VERIFI framework in detail. Lying at the core of VERIFI is our proposed *unlearning-verification mechanism*. As illustrated in Fig. 3, the mechanism defines the timeline when unlearning and verification should be performed, and by whom, i.e., the central server or the leaving participant ("leaver"). Here, we focus on unlearning the leaver and his/her verification in FL in Fig. 3.

A. Unlearning-Verification Mechanism

Suppose the entire FL process consists of T_{total} communication rounds. As shown in Fig. 3(a), the mechanism divides the entire process into two stages, including a free stage ($[T_0, T_{enabled})$) and an unlearning-enabled stage ($[T_{enabled}, T_{total}]$). The free stage refers to an early FL stage where the global model has not yet converged to a good solution. In this stage, all participants can join and leave the federation freely without activating the unlearning mechanism, as in this stage, the next round of training often overwrites the model's memorization at the previous rounds. Leaving the federation after $T_{enabled}$ will activate the unlearning and verification process, as at this time, the model's memorization of the private data is stabilized. Note that joining the federation at this stage should also be carefully examined as it is a *harvest stage* where small contribution can receive a big reward, i.e., a high-performance global model. Here, we only focus on leaving and unlearning.

Fig. 3(b) shows the pipelined unlearning-verification mechanism with a single leaving participant[6] . The detailed steps can be found in Mechanism 1. In this paper, we focus on one leaving participant per round while leaving more complex scenarios to future work. Specifically, the leaving participant (denoted by a) first notifies the server about the leaving at t_m (Step 2). Meanwhile, the leaving participant applies a marking method to mark the data (e.g., private training samples, triggers or model parameters) that needs to be checked against unlearning (Step 3.a). We call the marked data *'markers'*. Once the marked model is uploaded to the server (Step 3.b), the leaving participant notifies the server to apply the unlearning method to unlearn its data (Step 3.c). Note that the server may or may not be aware of the existence of markers since the verification right is in the hands of the participants, not the server. The server-side unlearning may last for more than one communication round (Step 4.a). The participant will actively check the unlearning effect on the markers immediately after marking is completed (Step 4.b). After a few rounds of checking at t_{leave}, the participant will leave with assured privacy (Step 5.a) or distrust (Step 5.b), depending on whether the expected unlearning effect on the marker is satisfied.

Practical Considerations. The time between t_m and t_{leave} is called the *checking period*, which spans both the marking and unlearning periods. The longer the checking period, the more certain the leaving participant is about the verification result. Nevertheless, the longer checking period also means that the leaving participant can download the global model more times than he/she should, which might be unfair to other participants. As such, t_{leave} is an important hyper-parameter that should be agreed upon among the federation. The $T_{enabled}$ hyper-parameter, which ends the free stage and enables unlearning, can be determined by the global training loss or accuracy. In FL, the server does not have data to compute the global loss/accuracy. Nevertheless, the server can estimate the convergence by the stability of the aggregated gradients. It is also worth mentioning that dividing the FL process into two stages is of practical importance: it can avoid the collapse of the global model caused by the unlearning.

[6] VERIFI is easily extended to the situation where multiple leavers require to be forgotten and verified.

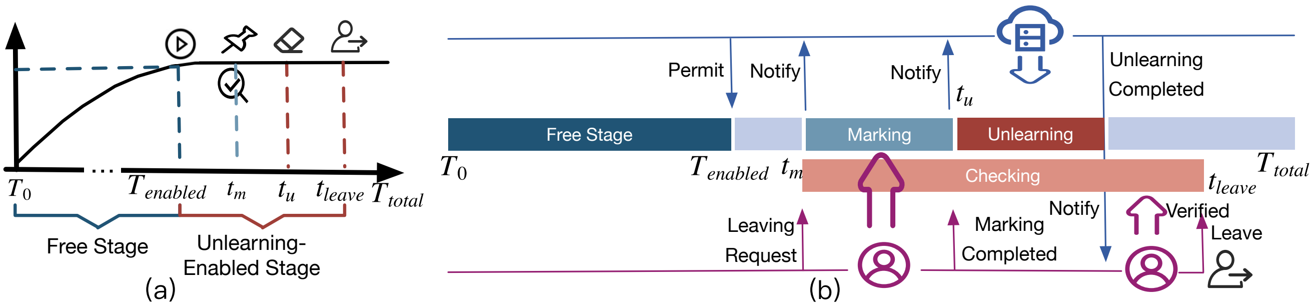


Fig. 3: The proposed unlearning-verification mechanism.

Mechanism 1: Federated Unlearning-Verification

```
Input: Unlearning-enabled stage $[T_{enabled}, T_{total})$,
       marking starting point $t_m \in [T_{enabled}, T_{total})$,
       unlearning starting point $t_u \in (t_m, t_{leave})$,
       leaving point $t_{leave} \in (t_u, T_{total})$, checking
       metric threshold $\delta$, marking function $\phi(\cdot)$,
       unlearning function $\varphi(\cdot)$, checking function
       $\psi(\cdot)$, aggregation rule $Agg(\cdot)$
  1) Free stage: Vanilla FL before $T_{enabled}$
  2) At $t_m$, participant a notifies the server to leave
  3) Marking at $t_m$:
     a) a marks its local model $\widetilde{\boldsymbol{w}}_{t+1}^{(\mathsf{a})} \leftarrow \phi(\boldsymbol{w}_{t+1}^{(\mathsf{a})})$
     b) a uploads the marking update $\widetilde{\boldsymbol{w}}_{t+1}^{(\mathsf{a})} - \boldsymbol{w}_t$ to server
     c) a notifies the server the completion of marking
  4) Unlearning at $t_u \in [t_m + 1, t_{leave})$:
     a) Server performs aggregation and unlearning:
        $\boldsymbol{w}_{t+1} \leftarrow Agg\left(\varphi\left(\left\{\boldsymbol{w}_{t+1}^{(i)} - \boldsymbol{w}_t\right\}_{i=1}^{n}\right)\right)$
     b) Checking: a checks if unlearning is sufficient
        $\psi(\boldsymbol{w}_{t_m}) - \psi(\boldsymbol{w}_{t+1}) \geq \delta$
  5) Leaving at $t_{leave}$:
     a) a leaves with assured privacy if
        $\psi(\boldsymbol{w}_{t_m}) - \psi(\boldsymbol{w}_{t+1}) \geq \delta$
     b) a leaves with distrust if $\psi(\boldsymbol{w}_{t_m}) - \psi(\boldsymbol{w}_{t+1}) < \delta$
```

System Assumption. Following existing unlearning works [31], [41], we assume a trusted server with an unlearning method in place. We also assume that the local data of the involved participants remains the same in each contributed round of FL. The server adopts partial device participation strategy in each round to motivate generating an excellent model with respect to each participant, such as choosing 10 participants among the 100 alternatives to contribute their local models each time. Beyond the above assumption, we additionally explore the adversarial scenarios that the server and other participants are *unlearning-malicious* in Section V-A.

B. Unlearning

Unlearning is performed by the server immediately after the completion of marking by the leaving participant. All unlearning methods are marked by the subscription symbol ^{u} before their names. For a comprehensive analysis, we adopt, adapt or propose a set of comprehensive methods in this work. This gives us 7 unlearning methods in total, including 3 existing, 3 adapted and 1 newly proposed (^{u}S2U), as summarized in Table I. ^{u}RT and ^{u}RTB are both retraining-based unlearning methods but with different retraining starting points [24], [19]. ^{u}CGS, ^{u}GGS and ^{u}IGS all exploit gradient subtraction to erase the leaving data but with different gradient reconstruction strategies [30], [32]. ^{u}DP is an existing differential privacy [1] based unlearning method [39]. Considering the high cost and negative impact of existing unlearning methods on the original task, we further propose ^{u}S2U, a more efficient and friendly unlearning method that is more compatible with FL.

1) Proposed ***Scale-to-Unlearn*** ***(^{u}S2U)****:* ^{u}S2U is inspired by the observation that scaling up/down the uploaded updates can substantially influence the global model [3], [30], [32]. Intuitively, scaling up/down one's local update would increase/reduce its contribution to the global model. When unlearning is activated, ^{u}S2U erases the contribution of the leaving data from the global model as follows:

$$\varphi\left(\boldsymbol{w}_{t_u+1}^{(j)} - \boldsymbol{w}_{t_u}\right) = \begin{cases} \alpha\left(\boldsymbol{w}_{t_u+1}^{(\mathsf{a})} - \boldsymbol{w}_{t_u}\right), & \text{if } j \text{ is } \mathsf{a} \\ \beta\left(\boldsymbol{w}_{T_{enabled}} - \boldsymbol{w}_{t_u}\right), & \text{if } j \in C\diagup\mathsf{a} \end{cases} \tag{4}$$

where $t_u \in [T_{enabled}, T_{total}]$ is the current unlearning round (see Fig. 3), $\alpha \in (0,1)$ is the down-scaling ratio, $\beta \in [1, +\infty)$ is the up-scaling ratio, and C records the selected participants in FL at t_u. Since in the unlearning-enabled stage, all local models are expected to have minimal parameter changes within a few communication rounds. Therefore, ^{u}S2U can use the global model at $T_{enabled}$ to roughly approximate other participants' local models at t_u: $\boldsymbol{w}_{t_u+1}^{(j)} = \boldsymbol{w}_{T_{enabled}}, \forall_{j \in C\diagup\mathsf{a}}$. Note that ^{u}S2U does not need accurate approximation here. By scaling up/down others'/a's local update at t_u, ^{u}S2U tends to increase a's distance to other participants' local updates, thus actively forcing the model to unlearn a. After unlearned by ^{u}S2U, the global model is closer to other participants' local models and farther away from a's local model. The theoretical explanation can be found in Appendix VI-G. ^{u}S2U is compatible with most of the commonly used aggregation rules such as FedAvg [34] and Krum [4].

2) Existing or Adapted Unlearning Methods: **Retraining methods**, including Retraining (^{u}RT) and Retraining breakpoint (^{u}RTB), retrain the global model without the leaving data. ^{u}RT reverts the global model to the starting point $\boldsymbol{w}_0$, then retrains the model from scratch without the leaving participant a's local gradients. ^{u}RTB is adapted from ^{u}RT and it retrains the global model from a certain breakpoint $\boldsymbol{w}_b$. ^{u}RTB additionally requires storing the intermediate global

TABLE I: A summary of unlearning methods.

Mechanism		Method	Source	Description
Multi-Step	Retraining	Retraining (^{u}RT)	Existing	Retrain from scratch
		Retraining Breakpoint (^{u}RTB) [24], [19]	Adapted	Retrain from the stored intermediate model at the breakpoint
	Gradient Subtraction	Calibrated Gradient Subtraction(^{u}CGS) [30]	Existing	Subtract the calibrated unlearned gradients by leveraging others' historical updates
		Generated Gradient Subtraction(^{u}GGS) [32]	Existing	Subtract the unlearned gradients produced by a trainable dummy generator
		Individual Gradient Subtraction(^{u}IGS) [6], [10]	Adapted	Subtract the leaver's gradient
One-Step	Covered by noise	Differential Privacy (^{u}DP) [39]	Adapted	Cover the memory by introducing noise
	Scaling	**Scale-to-Unlearn (^{u}S2U)**	**Proposed**	**Scale up others' gradient and scale down the leaver's gradient**

model obtained at each communication round.

Gradient subtraction methods, including Calibrated Gradient Subtraction (^{u}CGS), Generated Gradient Subtraction (^{u}GGS) and Individual Gradient Subtraction (^{u}IGS), erase the leaving data by subtracting the corresponding gradients. ^{u}CGS [30] leverages a calibration algorithm to approximate the gradients to be unlearned from other participants' historical updates. ^{u}GGS [32] deploys a trainable dummy gradient generator to produce the gradients to be unlearned. ^{u}IGS is adapted from the above two methods and it directly subtracts the local updates of the leaving participant during the next few rounds of aggregation. Formally, these methods perform gradient subtraction as follows:

$$\varphi\left(\boldsymbol{w}_{t+1}^{(\mathsf{a})}-\boldsymbol{w}_t\right)=-\lambda\sum_{i\in\Omega}\left(\hat{\boldsymbol{w}}_{i+1}^{(\mathsf{a})}-\boldsymbol{w}_i\right), \tag{5}$$

where, $\hat{\boldsymbol{w}}_{i+1}^{(\mathsf{a})}$ is a's local gradient (raw, generated or estimated) to be unlearned, λ is a hyper-parameter balancing the unlearning of a's local updates and the original task, and Ω records the rounds when a's gradient has been uploaded to the server.

Differential Privacy (DP) method, ^{u}DP [39] adds noise to a's local updates at t_u to smooth out the sensitive information and cover the memorization of a's private data in the global model:

$$\varphi\left(\boldsymbol{w}_{t+1}^{(\mathsf{a})}-\boldsymbol{w}_t\right)=e^{\varepsilon}\left(\boldsymbol{w}_{t+1}^{(\mathsf{a})}-\boldsymbol{w}_t\right)+\delta,\ t=t_u, \tag{6}$$

ε is the privacy budget, δ is a relaxation term, the smaller ε, the more noise is added into the local model. The central server introduces and adjusts the (ε,δ) parameter pair to blur the memorization without degrading too much of the global model's performance.

Discussion. Among the above 7 unlearning methods, ^{u}RT and ^{u}RTB are arguably the most effective yet costly unlearning methods. The 5 multi-step methods (see Table I), including the 2 retraining and 3 gradient subtraction methods, all need to perform unlearning for multiple communication rounds (ideally, the same number of rounds as the leaving participant's contribution in the past). As such, these methods need to store the raw, generated or estimated local/global gradients for each round. Such storage may raise new privacy concerns. Both ^{u}DP and our proposed ^{u}S2U are one-step methods that only exploit the current round of gradient information. So both methods are lightweight and do not need to store the local or global gradients. By involving noise into the gradients, ^{u}DP may hurt the original task as FL heavily relies on high-quality gradients to converge. Compared with ^{u}DP, our ^{u}S2U is more FL-friendly as it has minimum (or even positive) impact on other participants' local updates after aggregation.

C. Verification

Verification is performed by the leaving participant, consisting of two chained steps: *marking* and *checking*. In other words, once a marking method is determined, so does its checking method or metrics. In Table II, we adopt, adapt or propose 5 marking methods for unlearning verification. ^{v}FM and ^{v}EM are our proposed non-invasive verification methods. ^{v}BN inherits the backdoor-based verification in [41], thus also raising new security risks. ^{v}ME and ^{V}BF are both adapted from the deep learning intellectual property protection field [45], [9].

1) Marking: We call the marked information as 'markers', a concept that is analogous to the biomarkers used in biomedical studies [15]. Intuitively, markers can be any information related to the leaving data, e.g., a subset of local samples, gradients or models. Table II summarizes the characteristics of the marking methods.

Proposed Unique Memory Markers. We propose to leverage the unique memories of the global model about the leaving data as effective markers. Specifically, we propose to explore two types of unique memories: *forgettable memory* and *erroneous memory*[7].

Forgettable Memory (^{v}FM) refers to the subset of forgettable examples by the global model. Intuitively, forgettable examples are the hardest and unique examples owned by the leaving participant, whereas unforgettable examples are easy examples shared across different participants [43]. ^{v}FM determines forgettable examples by the variance of their local training loss and chooses a subset of samples with the highest loss variance across several communication rounds as the markers. Fig. 4 illustrates a few forgettable examples (i.e., markers) identified by ^{v}FM from the MNIST [29] dataset. We denote the marker set found by ^{v}FM for a leaving participant a as D_{a}^m and $D_{\mathsf{a}}^m \subset D_{\mathsf{a}}$. At the marking step, a locally fine-tunes the model for a sufficient number of iterations to reduce the local loss variance on D_{a}^m, then uploads the fine-tuned parameters to the server. Now the global model will also have relatively low loss variance on D_{a}^m. During checking, a can monitor the global model's loss variance on D_{a}^m to verify the unlearning effect. Effective unlearning should quickly recover the high loss variation on D_{a}^m.

Erroneous Memory (^{v}EM) refers to the subset of erroneous (incorrectly predicted) samples to the global model. Intuitively, erroneous samples are likely to be the hard and rare samples uniquely owned by the leaving participant, as otherwise they should be well learned by the global model if other participants also have these samples. As described in Algorithm 2, ^{v}EM

[7] The unlearning verification effect difference between these unique memory samples and the leaving data can be found in Appendix VI-E.

TABLE II: A summary of marking methods.

Category	Method	Source	Type	Marker	Checking	Training Controllability	External Data	White-box Access
Watermark	Model Embedding(^{v}ME) [45]	Adapted	Invasive	Embedded bits in model parameters	Matching rate of the extracted bits	●	○	●
	BadNets(^{v}BN) [41]	Existing	Invasive	Pixel-level backdoor trigger	Accuracy on the backdoor samples	◑	●	○
Fingerprint	Boundary Fingerprint(^{v}BF) [9]	Adapted	Invasive	Boundary samples	Accuracy on the boundary samples	○	●	○
Unique Memory	**Forgettable Memory(^{v}FM)**	**Proposed**	**Non-invasive**	**Forgettable samples**	**Variance of loss on the forgettable samples**	○	○	○
	Erroneous Memory(^{v}EM)	**Proposed**	**Non-invasive**	**Erroneous samples**	**Loss on the erroneous samples**	○	○	○

●— Required ◑— Partially required ○— Not required

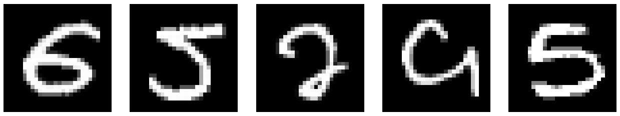

(a) '6' (b) '5' (c) '2' (d) '9' (e) '5'

Fig. 4: Forgettable examples (markers) found by ^{v}FM for one leaving participant during 10-participant FL on MNIST [29].

first investigates the top κ (%) of the high loss samples (Line 1) and selects the majority class of erroneous samples into the marker set $D^m_{(\mathsf{a})}$ (Line 2). Note that the marker set has only one class (i.e., the majority class). Fig. 5 shows a few erroneous MNIST samples identified by ^{v}EM, in which images of class '7' are misclassified as '2'. ^{v}EM then relabels $D^m_{(\mathsf{a})}$ to its mostly predicted label by the local model $f^{(\mathsf{a})}$ (Lines 4-6) and fine-tunes the local model on the relabelled dataset to obtain a marked model $\widetilde{f}^{(\mathsf{a})}$ (Line 7). The marked model will then be uploaded to the central server to be aggregated into the global model. Fine-tuning with erroneous labels is to make the loss on the markers smaller and check if the global model can increase the loss on the markers through unlearning. Since a fine-tunes the local model to maintain a low loss on the ^{v}EM markers during the marking process, effective unlearning should quickly recover the high losses on the markers.

Algorithm 2: Erroneous Memory Marking

```
Input: The local model $f^{(\mathsf{a})}$ and private data $D_\mathsf{a}$ of
       participant a, erroneous sample proportion $\kappa$,
       fine-tuning iterations $T^{(\mathsf{a})}_{ft}$.
Output: Marked local model $\widetilde{f}^{(\mathsf{a})}$, marker dataset $D^m_\mathsf{a}$
1 $D^\kappa_l \leftarrow$ top $\kappa$% of high loss samples (and labels) in $D_\mathsf{a}$
2 $D^m_\mathsf{a} \leftarrow$ the majority class of samples in $D^\kappa_l$
3 $\overline{D}_\mathsf{a} \leftarrow D_\mathsf{a} \setminus D^m_\mathsf{a}$
4 foreach $(\boldsymbol{x}, y) \in D^m_\mathsf{a}$ do
5     $y \leftarrow$ the most predicted label on $D^m_\mathsf{a}$
6 end
7 $\widetilde{f}^{(\mathsf{a})} \leftarrow$ fine-tune $f^{(\mathsf{a})}$ on $\overline{D}_\mathsf{a} \cup D^m_\mathsf{a}$ for $T^{(\mathsf{a})}_{ft}$ iterations
8 return $\widetilde{f}^{(\mathsf{a})}$, $D^m_\mathsf{a}$
```

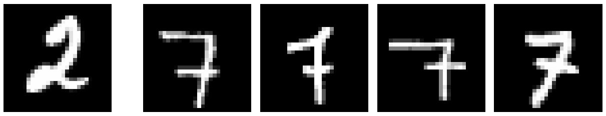

(a) '2' (b) '7'→'2'

Fig. 5: Erroneous samples (markers) found by ^{v}EM for one leaving participant during 10-participant FL on MNIST. (a): a normal image from class '2'; (b): the erroneous images with majority true class '7' but mostly are predicted as '2'.

Existing or Adapted Marking Methods. Existing watermarking methods such as parameter-based [45] and backdoor-based watermarking [56] or fingerprinting methods [9] from the field of deep learning intellectual property protection can be adapted as marking methods.

For watermarking, we adopt the backdoor-based (^{v}BN) marking method from [41] that was initially proposed for traditional machine unlearning verification. ^{v}BN leverages the BadNets [21] backdoor attack to inject trigger patterns associated with a backdoor class into the global model to verify the unlearning effect. At the marking step, ^{v}BN fine-tunes the local model on backdoored data and uploads the backdoored local parameters to the server for aggregation. After fine-tuning, backdoored samples exhibit a high attack success rate on the backdoored local and global models. Effective unlearning should break the correlation between the trigger pattern and the backdoor class, i.e., lowering the attack success rate.

For fingerprinting, we adapt the Boundary Fingerprint (^{v}BF) [9] to find decision boundary fingerprints (markers) to verify unlearning. ^{v}BF generates adversarial examples that are close to the decision boundary to characterize the robustness property of the local model $f^{(\mathsf{a})}$. Arguably, the adversarial examples with relatively high and close top-2 class probabilities are boundary examples [9]. Therefore, before the unlearning round t_u (see Fig. 3), ^{v}BF marks the following adversarial examples as markers:

$$D^m_\mathsf{a} = \{(\boldsymbol{x}+\sigma, y) \mid |f^{(\mathsf{a})}_{top-1}(\boldsymbol{x}+\sigma) - f^{(\mathsf{a})}_{top-2}(\boldsymbol{x}+\sigma)| \leq \gamma, (\boldsymbol{x}, y) \in D_\mathsf{a}\}, \tag{7}$$

where, $f^{(\mathsf{a})}_{top-1}(\boldsymbol{x}+\sigma)$ and $f^{(\mathsf{a})}_{top-2}(\boldsymbol{x}+\sigma)$ denote the top-1 and top-2 class probabilities respectively, $\boldsymbol{x}+\sigma$ is the PGD [33] adversarial example of $\boldsymbol{x}$, and $\gamma \in [0, 0.1)$ is a small positive value defining how close are the two probabilities. At the marking step, ^{v}BF first fine-tunes the local model on D^m_a to obtain a marked local model $\widetilde{f}^{(\mathsf{a})}$ which now becomes robust to D^m_a and has more smoothed boundary around the markers. The marked local model will then be uploaded to the central server and aggregated into the global model. Effective unlearning should quickly forget the smoothed (robust) boundary around the markers (thus resulting in wrong predictions), which can be easily checked by the performance on the adversarial markers.

Remark. Note that some verification methods included in this work may raise security concerns (see Appendix VI-D), or become less effective if a secure FL algorithm is implemented. One example is the backdoor-based verification proposed in [41]. This aspect has also been considered when categorizing the verification techniques or making our recommendations.

Therefore, we categorize the marking methods in Table II into two major types: 1) *invasive* methods that need to tamper with the FL process, such as modifying the global model training or injecting external data; and 2) *non-invasive* methods that only need to keep track of a subset of existing data. The last three columns highlight the three undesired properties: training controllability, external data and white-box access. These undesired properties may introduce new security or privacy risks into FL. Invasive methods often rely on one or more of the undesired properties. By contrast, our proposed unique memory-based methods do not require any of training controllability, external data or white-box access.

2) Checking: Checking is also performed by the leaving participant immediately after the marking. In this step, the change of the global model's performance on the markers can be used to measure the degree of unlearning. This process can take a few communication rounds until the leaving time T_{leave}. Note that the performance is directly measured on all leaving samples if markers are not used, as it did in most prior works [10], [6], [31].

Here, we consider four metrics (including accuracy, loss, influence function [27], and KL divergence [20]) to measure the model's performance or performance change. The accuracy and loss can be easily calculated on either the marker set or the entire leaving data. Influence function [27] formalizes the impact of a training sample on model prediction. We compute the influence function (IF) of all leaving samples (not just the markers) on the global model to quantify unlearning. We refer readers to [27] for more calculation details of the influence function. KL divergence (KL) [20] measures the distributional difference between the global model's output probability distribution and an ideal with-unlearning probability distribution $\vec{\rho}$. Arguably, the uniform distribution indicates an ideal case of unlearning, i.e., $\vec{\rho} = (\frac{1}{C}, \cdots, \frac{1}{C})$ with C is the total number of classes. This gives us the following KL divergence metric on the markers for unlearning verification:

$$KL(D_{\mathsf{a}}^{m}) = \mathbb{E}_{\boldsymbol{x} \in D_{\mathsf{a}}^{m}} \Big[f_t(\boldsymbol{x}) \log \frac{f_t(\boldsymbol{x})}{\vec{\rho}} \Big], \; t \in [t_m, T_{total}). \tag{8}$$

If unlearning is effective, then the global model will not produce any meaningful predictions on the markers, resulting in a low or even zero KL divergence.

IV. Experiments

We conduct extensive experiments with the VeriFi framework to answer the key research questions (RQs) on verifiable federated unlearning defined in Section I. All experiments are conducted on a Linux server with 4 Nvidia RTX 3090 GPUs, each with 24 GB dedicated memory, Intel Xeon processor with 16 cores and 384 GB RAM. Our code is implemented using PyTorch 1.7.1 with CUDA 11.1 and Python 3.7.

Experimental Setup. We run experiments on 7 datasets, including two popular low-resolution image classification datasets (MNIST [29] and CIFAR-10 [28]), a speech recognition dataset (SpeechCommand [49]), two high-resolution image datasets for face (VGGFace_mini [36]) and natural object (ImageNet_mini[16]) recognition, and two medical image datasets for skin cancer (ISIC [44], [13], [14]) and COVID-19 (COVID [17]) diagnoses. The datasets and corresponding models are summarized in Table III. The training data of each dataset are equally distributed to each participant, and there is no overlap between individual data. The default parameter settings (e.g., learning rate and optimizer) are summarized in Table IV and Table IX in Appendix VI-A. The experimental setup of VeriFi is summarized in Table IV. The two grey highlighted hyper-parameters are for an early-stage testing (i.e., leaving immediately after unlearning is enabled) experiment only.

TABLE III: Datasets, models and test accuracies (Acc).

Dataset	#classes	#samples	Resolution	Model	Acc (%)
MNIST [29]	10	70000	32*32	LeNet-5	99.11
CIFAR-10 [28]	10	60000	32*32	ResNet-18	95.37
SpeechCommand [49]	10	46256	32*32	CNN-LSTM	73.09
ISIC [44], [13], [14]	4	8000	224*224	DenseNet-121	68.06
COVID [17]	3	16619	224*224	ResNet-18	88.42
ImageNet_mini [16]	10	13500	224*224	ResNet-18	90.60
VGGFace_mini [36]	20	7023	224*224	ResNet-18	95.59

TABLE IV: VeriFi setup. η: local learning rate; η_{fl}: global learning rate; $|B|$: local batch size; $T_{enabled}$: unlearning-enabled round; t'_u: unlearning round (an early-stage testing); t'_{leave}: leaving round (an early-stage testing); t_m: marking round; t_u: unlearning round (standard testing); t_{leave}: leaving round (standard testing); T_{total}: total round; T_{local}: local update epochs; n: number of involved participants at each round; N: total number of participants.

	η	η_{fl}	$\lvert B \rvert$	$T_{enabled}$	t'_u	t'_{leave}	t_m	t_u	t_{leave}	T_{total}	T_{local}	n	N
MNIST	0.01	10	1024	40	40	100	110	120	200	200	1	10	100
CIFAR-10	0.1	1	128	40	40	100	110	120	200	200	10	10	100
SpeechCommand	0.1	1	256	25	25	50	60	70	100	100	10	10	100
ISIC	0.1	1	8	40	40	100	106	112	130	130	10	10	100
COVID	0.1	1	16	40	40	100	106	112	130	130	10	10	100
ImageNet_mini	0.1	1	16	140	140	180	186	192	210	210	10	10	100
VGGFace_mini	0.1	1	16	240	240	300	306	312	330	330	10	10	100

A. Is Federated Unlearning Necessary?

We first test what would happen if there is no unlearning but only *Natural Forgetting (^{u}NF)* when a participant a leaves the federation. a is randomly chosen from all the alternative participants in FL and does not influence the final result. We evaluate the unlearning effect of ^{u}NF by comparing the global model's performance on the leaving data with that obtained via Natural Training (^{u}NT) (a never leaves) at the end of FL. The results are shown in Table V. It is evident that the performance differences (the **diff** columns in Table V) between ^{u}NF and ^{u}NT are almost negligible according to all four metrics. It means that the global model still memorizes the leaving data if the participant leaves at the convergence stage. Therefore, *unlearning is necessary to actively remove information about the leaving participant's private data.*

B. Are Markers Necessary for Verification?

To answer the question, we run experiments to verify the different unlearning effects of the 7 unlearning methods using only the checking metrics without any marking methods (markers). Intuitively, if the checking metrics alone can properly identify the difference before and after unlearning, then specialized markers are unnecessary. For each of the 4 metrics (i.e., accuracy, loss, KL and IF), we compute its difference before (at t_m) and after (at t_u) unlearning on the leaving data

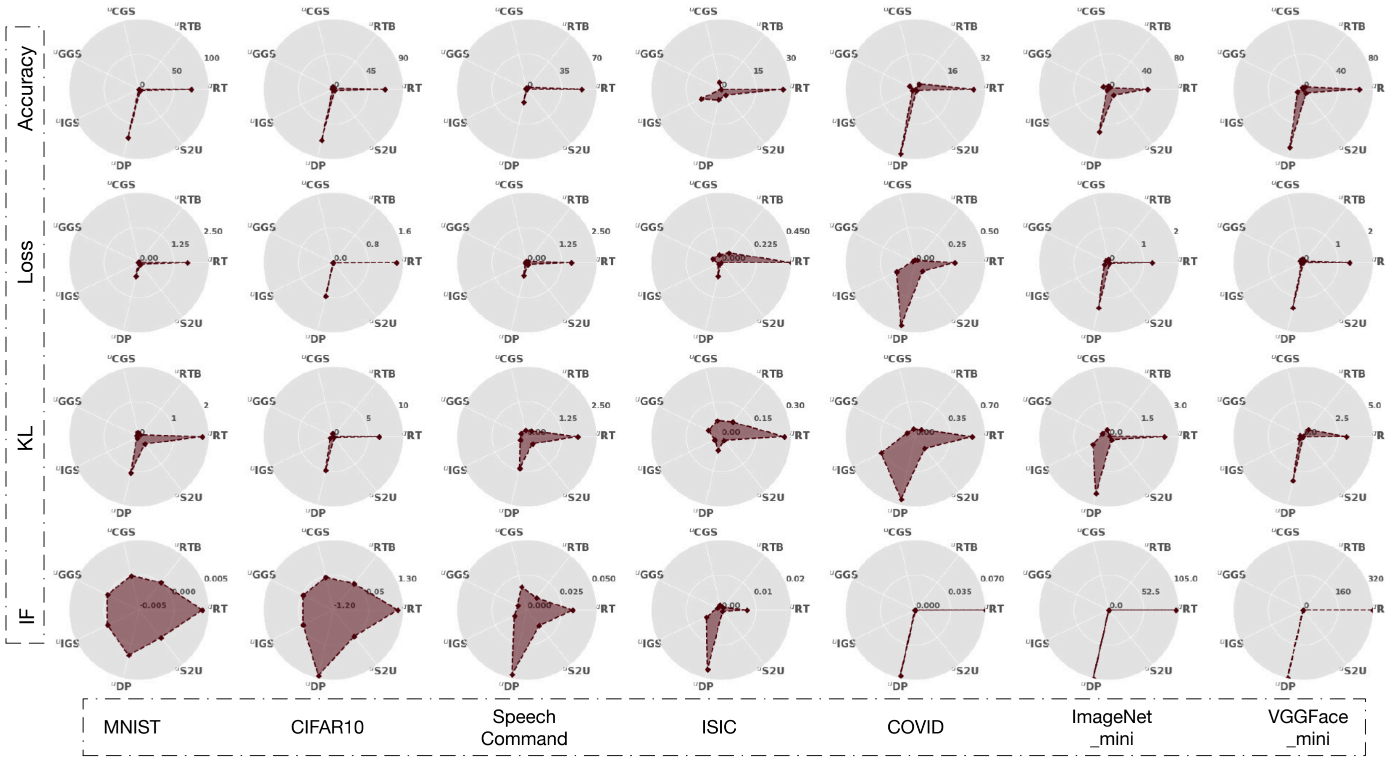


Fig. 6: Verifying the unlearning effect using only the 4 metrics (rows) computed on the leaving data. Each radar chart has 7 dimensions corresponding to the 7 unlearning methods, with each dimension showing the metric difference before and after (after minus before) unlearning. Each column of radar charts correspond to one dataset. Failed verification occurs at dimensions with almost zero difference before and after unlearning.

TABLE V: The absolute performance change (diff= $|^{u}\text{NF}-^{u}\text{NT}|$) on the leaving data caused by ^{u}NF.

Dataset	**Metrics**	**At the Leaving Round**			**At the End of FL**		
		^{u}NT	^{u}NF	**diff**	^{u}NT	^{u}NF	**diff**
CIFAR10	Acc (%)	77	77.8	0.8	87	85	2
	Loss	0.14	0.14	0.0	0.08	0.08	0.0
	KL	6.75	6.92	0.17	8.31	8.47	0.16
	IF	9.21e-7	5.67e-5	5.58e-5	1.89e-7	1.98e-5	1.98e-5
Speech Command	Acc (%)	64.61	64.67	0.06	64.73	67.21	2.48
	Loss	0.50	0.52	0.02	0.48	0.5	0.02
	KL	2.1	2.08	0.02	2.46	2.24	0.22
	IF	-0.007	-0.006	0.001	0.006	0.006	0.0
Covid	Acc (%)	68.18	65.15	3.03	81.82	80.3	1.52
	Loss	0.49	0.36	0.13	0.26	0.3	0.04
	KL	0.31	0.66	0.35	1.12	0.99	0.13
	IF	1.34e-5	0.03	0.03	0.001	0.001	0.0
VGGFace _mini	Acc (%)	57.14	69.64	12.5	78.57	76.79	1.78
	Loss	0.66	0.54	0.12	0.63	0.49	0.14
	KL	3.03	3.81	0.78	3.47	3.99	0.52
	IF	0.085	0.247	0.162	0.029	0.073	0.044

D_{a}. Take accuracy as an example, the metric difference is computed as follows:

$$Acc_{diff}(D_{\mathsf{a}}) = |Acc_{t_m}(D_{\mathsf{a}}) - Acc_{t_u}(D_{\mathsf{a}})|. \tag{9}$$

Similarly, we can define other three metrics: $Loss_{diff}$, KL_{diff}, and IF_{diff}.

We plot the 4 metric differences for all 7 unlearning methods on each dataset in Fig. 6. Large metric differences (large covered area in a radar chart) indicate successful verification. For a given metric, if it successfully verifies the difference before and after unlearning across different datasets, it can be regarded as an effective metric for federated unlearning verification. Unfortunately, as shown in Fig. 6, we find that, in general, none of the metrics can effectively verify the unlearning effects of all unlearning methods. Furthermore, among the 7 unlearning methods, *^{u}DP and ^{u}RT are relatively easier to verify by any of the 4 metrics*. This means that we don't need sophisticated verification methods if ^{u}DP or ^{u}RT is adopted as the unlearning method. While this result is encouraging, the two unlearning methods also have their own weaknesses. For instance, *^{u}RT is very costly and ^{u}DP causes the most performance drop* among the 7 unlearning methods (see Table VII). We have also tested two naive methods for verification: model parameter difference and privacy leakage difference in Appendix VI-B and VI-C, respectively. The results show that the parameter difference (measured by Euclidean distance or Cosine similarity) of the global model before and after unlearning is also insufficient for verifying the unlearning methods, except ^{u}DP, ^{u}RT and our ^{u}S2U. Furthermore, from the perspective of privacy leakage [11], i.e., the success rate of membership inference of the leaving data, even ^{u}RT cannot verify (e.g., not having a noticeable lower success rate than ^{u}NT) due to the contributions of other participants. Overall, we conclude that *many (5/7) of the unlearning methods may not be properly verified by the 4 metrics without specialized markers. More effective unlearning methods like ^{u}DP and ^{u}RT have certain weaknesses for practical usage*.

C. Federated Unlearning Verification with Markers

Here, we verify the unlearning effect of the 7 unlearning methods, with the 5 marking methods (markers). Similarly, we compute the metric difference of the global model on the marker set D_{a}^{M} before and after unlearning following equation (9). Due to space limitations, we only show the most effective metric for each type of marker[8]. The results are visualized in Fig. 7. A valid marking method should recognize ^{u}RT as the most effective unlearning method as ^{u}RT is the golden standard (i.e., the best unlearning one could achieve). And our ^{u}S2U should be more effective than ^{u}CGS, ^{u}IGS and ^{u}GGS, since it not only downscales the leaving gradients but also upscales other participants' gradients. An ideal marking method should be able to distinguish the different strengths of the unlearning methods.

The most effective verification method. In general, *our proposed ^{v}**EM** demonstrates better verification ability than ^{v}FM, ^{v}ME and ^{v}BN* [9]*, while ^{v}BF ranks the last*. Particularly, ^{v}EM markers could always distinguish (showing larger metric differences) stronger unlearning (^{u}RT, ^{u}RTB, ^{u}DP and ^{u}S2U) from the mild ones (^{u}CGS, ^{u}IGS and ^{u}GGS) on all datasets. By contrast, ^{v}FM could not effectively distinguish the unlearning effect of ^{u}S2U and ^{u}RTB on MNIST. ^{v}ME (injecting a bit string into the model parameter space) fails to verify ^{u}DP as it is not sensitive to the noise of ^{u}DP. Meanwhile, backdoor-based markers like ^{v}BN fail to mark the global model on the high-resolution datasets (the accuracy on ^{v}BN markers of the ISIC4, COVID, ImageNet_mini and VGGFace_mini datasets is similar to random guessing), and thus lose the verification ability. This result indicates that *the performance of invasive marking methods cannot be guaranteed in practice*. ^{v}BF can only distinguish the unlearning effect of ^{u}RT and ^{u}DP as other unlearning methods will not cause significant change on the decision boundaries.

The most effective unlearning methods. By examining the verified unlearning effect by the most effective marker ^{v}EM, we can also cross-validate the effectiveness of the 7 unlearning methods. In general, ^{u}**RT**, ^{u}**RTB**, ^{u}**DP** and our proposed ^{u}**S2U** demonstrate more effective unlearning effects than the other 3 unlearning methods. Note that, as a completely retraining method, ^{u}RT is arguably the most effective unlearning one could achieve and it is not surprising that ^{u}RT demonstrates better unlearning effects than ^{u}RTB, ^{u}DP and ^{u}S2U on nearly all datasets. The other three gradient subtraction based unlearning methods (^{u}CGS, ^{u}IGS and ^{u}GGS) exhibit limited unlearning effectiveness on the ^{v}EM markers.

Robustness to the byzantine-robust aggregation rules. We investigate the robustness of the most effective marker ^{v}EM and two invasive markers ^{v}BN and ^{v}ME when the server adopts different aggregation rules. The results on CIFAR-10 dataset are reported in Table VI. It is clear that the metric difference identified by ^{v}BN and ^{v}ME drops drastically when robust aggregation rules like Krum and Median are used at the server side[10]. By contrast, our ^{v}EM can maintain a stable difference, i.e., it is reasonably robust to the byzantine-robust aggregation rules.

TABLE VI: Verification robustness to different aggregation rules on CIFAR10 dataset with ^{v}RT unlearning. 'diff': absolute metric difference before and after ^{v}RT.

Verification	Rule	Metrics	Before	After	diff
^{v}BN	FedAvg	Accuracy	84.8	0.0	84.8
	Krum	Accuracy	6.2	0.0	6.2
	Median	Accuracy	9.8	0.0	9.8
^{v}ME	FedAvg	Accuracy	71.88	48.44	23.44
	Krum	Accuracy	39.06	48.44	9.38
	Median	Accuracy	40.62	48.44	7.82
^{v}EM	FedAvg	Loss	12.3	28.1	15.8
	Krum	Loss	14.61	25.69	11.08
	Median	Loss	12.14	27.09	14.95

Unlearning cost. Here, we investigate the cost of different unlearning methods. As shown in Table VII, ^{u}S2U demonstrates the least overall computational overhead and minor influence on the initial FL task. Besides, ^{u}RT needs the most time as it retrains from scratch. ^{u}RTB needs the most space as it saves the intermediate models. ^{u}CGS is both time- and space-consuming as the gradients to be unlearned need to be calibrated based on other participants' gradients. By contrast, the time/space cost of ^{u}GGS and ^{u}IGS are less than ^{u}CGS as they directly construct the leaving gradients without using other participants' models. ^{u}DP needs little time/storage cost by simply adding noises while causing most performance drop. Apart from ^{u}DP and ^{u}RT, other unlearning methods hardly degrade the global model's performance at the end of FL[11].

Verification cost. Here, we study the time cost, storage cost and negative impact on the original FL task for different verification methods. The results are reported in Table VIII. All marking methods cause tolerable performance drop (either in terms of loss or accuracy). Among the 5 marking methods, ^{v}FM shows less time/space cost than ^{v}EM and ^{v}ME, while ^{v}BN is the most time-consuming marking method as it needs to inject a backdoor watermark into the global model. ^{v}BF also requires much time/space to save and generate the boundary fingerprints. The verification method with less time overhead would produce the acceptable time delay in the large-scale practical FL system.

Correlation between the markers and the leaving data. The unlearning effect is more pronounced on the markers than on the leaving data as the markers are specially designed to serve this purpose. This raises a natural question *to what extent can the markers represent the leaving data*? To answer this question, we analyze the correlation between the global model's performance on the markers and on the leaving data when adopting ^{u}RT during $[t_u - n, t_u + n]$ as an example. In this experiment, n is set to 10 on MNIST, CIFAR-10 and

[8] The ^{v}BN result is normalized with the ideally maximum accuracy gap 100%, others are normalized according to the maximum gap value, owing to the unsuccessful backdoor-watermark injection in the big datasets.

[9] Specifically, to avoid the instant performance change on the backdoor-based watermarking method, we take the median performance during $[t_m, t_m + 2]$ as the result on the markers at t_m.

[10] The two robust aggregation rules are widely applied and can be modified and combined to form other aggregation rules, such as Bulyan [22] and Trimmed Mean [54].

[11] The minor negative impact of ^{u}GGS, ^{u}IGS and ^{u}CGS on the original task can be owned to the hyperparameter λ.

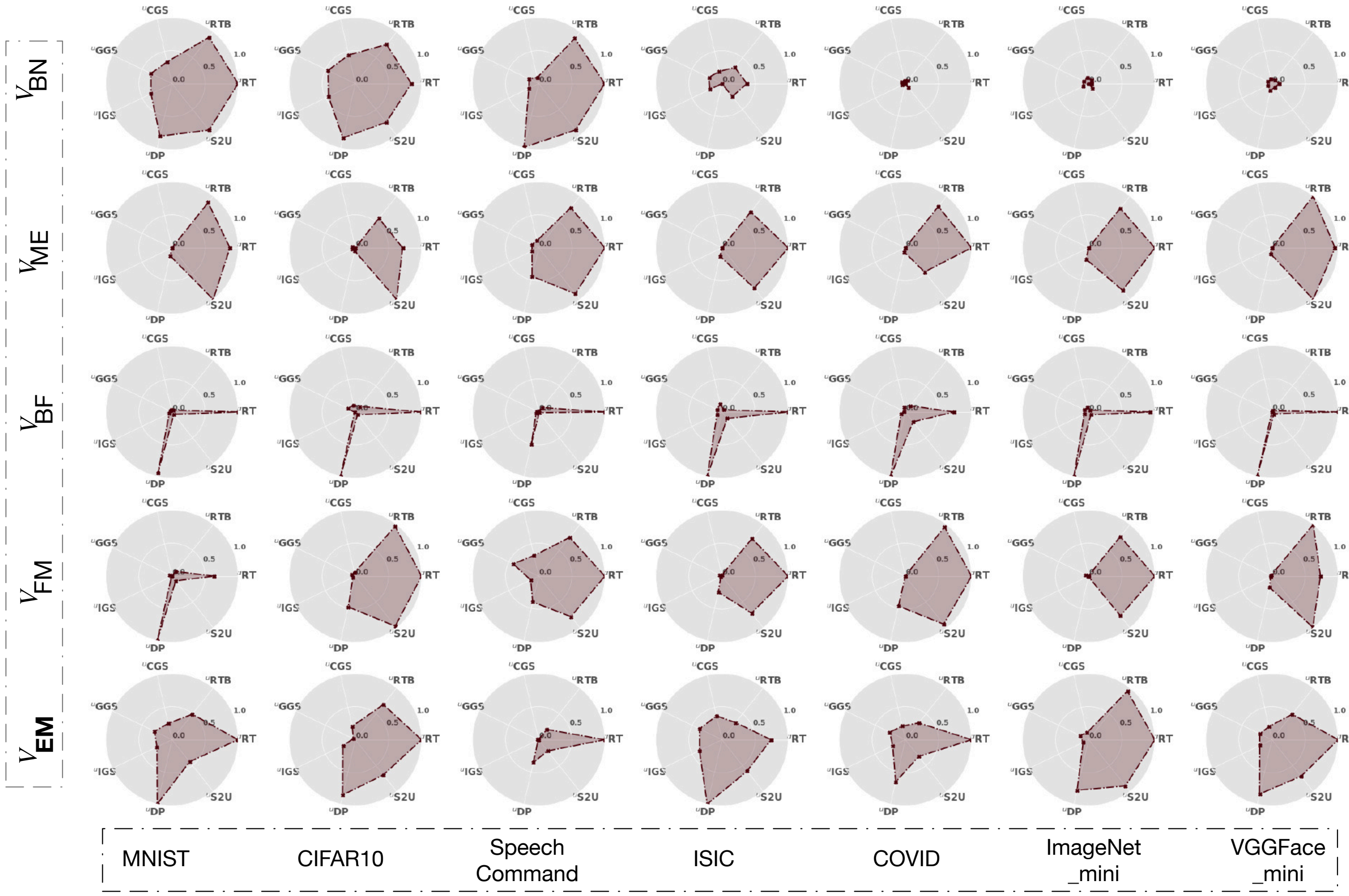


Fig. 7: Verifying the unlearning effect using 5 types of markers with each row representing one type of markers and each column corresponding to one dataset. The 7 dimensions of each radar chart correspond to the 7 unlearning methods, with each dimension showing the normalized metric difference (with log transformation) before and after (after minus before) unlearning. The most effective metric for markers ^{v}BN, ^{v}ME, and ^{v}BF is accuracy, while the most effective metrics for our memory-based markers ^{v}EM and ^{v}FM are loss and loss variance, respectively.

TABLE VII: Unlearning costs measured on CIFAR10 dataset.

	Time(s)	**Space(MB)**	$T_{total}\Delta$**Acc**	$T_{total}\Delta$**Loss**
^{u}RT	8157.91	44	-4.17	0.14
^{u}RTB	929.87	1760	0.18	0.0
^{u}CGS	1516.37	1936	-0.82	0.05
^{u}GGS	186.34	0	-0.8	0.03
^{u}IGS	37.9	176	0.13	0.02
^{u}DP	17.1	0	-6.26	0.22
^{u}S2U	21.61	44	0.05	0.01

TABLE VIII: Verification costs measured on CIFAR10 dataset.

	Time(s)	**Space(KB)**	$T_{total}\Delta$**Acc**	$T_{total}\Delta$**Loss**
^{v}BN	263.5	4	-0.53	0.02
^{v}ME	90.2	12	-1.64	0.08
^{v}BF	142.1	1233	-1.54	0.06
^{v}**EM**	99.6	4	-2.82	0.15
^{v}**FM**	10.26	4	-2.73	0.13

SpeechComamnd datasets, and 6 on other datasets. As shown in Fig. 8, the performance trends on the markers (except ^{v}BN markers) and the leaving data show a strong correlation before and after unlearning. This confirms that unlearning the markers can largely reflect the degree to which the server is unlearning the leaving data.

D. Unlearning-Verification: The Combinations

The verification method goes with the unlearning method. Fig. 9 shows the normalized verifiable unlearning effect of all the combinations of 7 unlearning methods and 5 verification methods on CIFAR-10 and SpeechCommand datasets. Each cell is associated with an unlearning method and a verification method. The blue cells highlight the best verifiable unlearning effect. Combining our analyses above, we obtain the following findings. A general effectiveness ranking of the unlearning methods is: ^{u}RT > ^{u}RTB > ^{u}S2U > ^{u}DP > ^{u}CGS ≈ ^{u}GGS ≈ ^{u}IGS. Considering the high cost of ^{u}RT and ^{u}RTB, and the negative impact of ^{u}DP on FL, *it leaves our proposed ^{u}S2U to be the most promising unlearning method for its relatively higher effectiveness, higher efficiency, less negative influence on FL and higher verifiability*. It is thus promising for future work to explore similar unlearning strategies or improve ^{u}DP for more effective, efficient, harmless and verifiable federated

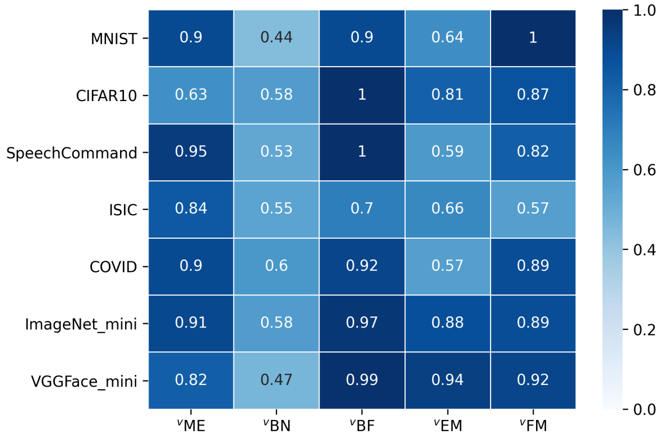
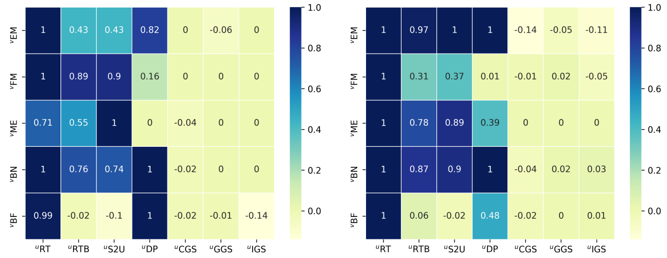


Fig. 8: Correlation between unlearning the markers vs. unlearning the leaving data.



(a) CIFAR-10

(b) SpeechCommand

Fig. 9: The unlearning effect of 7 unlearning methods (columns) verified by 5 marking methods (rows). The unlearning effect is the normalized (max: 1, min: 0) metric difference within each marking method (row): in each row, the maximum verifiable effect is 1 while the minimum is 0. The higher the normalized score, the better the unlearning-verification method.

unlearning.

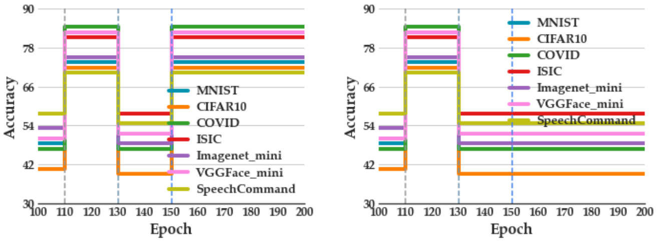
As for the verification methods, the deeper blue colored cells are more effective. So, the general ranking is: ^{v}EM > ^{v}FM > ^{v}ME > ^{v}BF > ^{v}BN. We put ^{v}ME, ^{v}BF and ^{v}BN to the end of the list is because they are all invasive methods that may introduce new security risks into FL (see Appendix VI-D). This makes our proposed ^{v}EM the most promising verification method. *The combination of our proposed ^{u}S2U with ^{v}EM verification is the most promising federated unlearning-verification strategy.* If ^{u}DP can be improved for FL, then the ^{u}DP-^{v}EM can also be an effective combination.

V. More Explorations

A. Adversarial Setting

We also use VeriFi to analyze a challenging adversarial setting where the attacker (the unlearning-malicious server or participant) may store the global model before participant a leaves and then restore a's memory after a leaves. This will compromise a's privacy. Since the unlearning-malicious server would not easily retreat at the cost of losing the excellent model updates from others, we then focus on the unlearning-malicious participant setting. We take the verification method ^{v}ME as an example, which checks unlearning based on the extracted bits from the model parameters. The successfully marked model by ^{v}ME would maintain a high and stable accuracy on the ^{v}ME markers. We assume the server implements the ideal unlearning method ^{u}RT which could effectively erase the memory about a's leaving data and markers.

Fig. 10 shows the different results when the attacker could capture and upload the global model in and out of the marking stage. As shown in Fig. 10(a), ^{u}RT decreases the accuracy on the markers at the unlearning round. However, the accuracy on the markers arises after the attack. In VeriFi, the leaver continuously tracks the global model to check unlearning for a while, not instantly. Therefore, the performance rise on the markers can be checked by the leaver, and the leaver would deem the unlearning invalid. However, if the attacker only captures the global model out of the marking stage in Fig. 10(b), the accuracy on the markers would not change. Thus, the leaver would deem the unlearning effective. Admittedly, VeriFi can only detect the deceived unlearning situation at a certain probability. Fortunately, the retrievable model out of the marking period would maintain a longer time span and a larger difference from the attacker's local model at the previous round, which would raise more attention. Thus, we can improve the accuracy of cheated unlearning checking by analyzing the similarity between the models of the adjacent rounds. This is an exciting problem that is worth further exploration.



(a) with marking

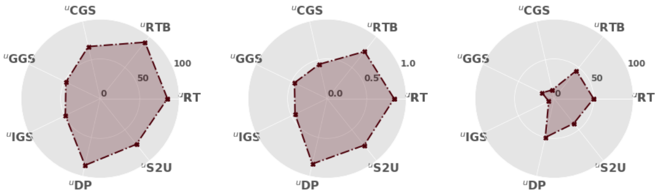
(b) without marking

Fig. 10: The attacker uploads the historical global model (in and out of the marking stage) to attack the effectiveness of the unlearning. The three vertical lines mark the time of marking t_m, unlearning t_u, and attack, respectively.

B. More Unlearning and Verification Parameters

We make a comprehensive analysis to explore the parameter influence in VeriFi. We take the mature verification method — ^{v}BN as an example, the concrete result can be found below.

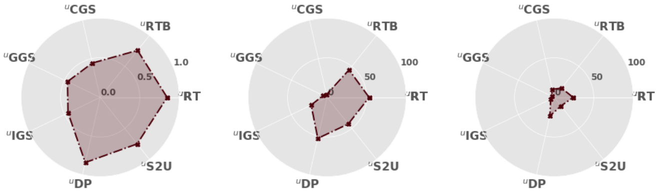
Influence of the marking time to the marking effect: Fig. 11(a) $\sim$ Fig. 11(c) presents the unlearning verification results when the marking time is respectively 10-th round (earlier), 110-th round (proper) and 210-th round (later). With the marking time getting later, the success probability of marking decreases, and the performance change caused by unlearning reduces. As for the reason, when the model has converged to a stable state, injecting the watermark into the global model gets harder. Meanwhile, the unlearning effect on the markers at t_u degrades, increasing the difficulty of distinguishing the authentic unlearning effectiveness. Thus, it's better to activate unlearning and verification when the global approximately converges.



(a) 10-th round (b) 110-th round (c) 210-th round

Fig. 11: Leaving time influence on CIFAR10 dataset.

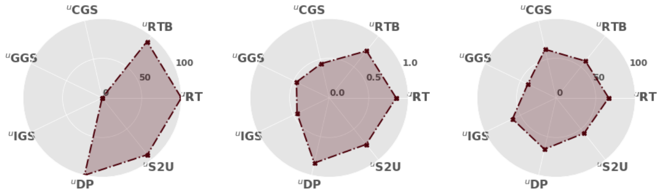
Influence of marking parameters: Different marking parameters inevitably cause unevenly marking effect, and further influence the unlearning verification result. We take the size and transparency (commonly used in ^{v}BN) parameters to explore the influence of marking parameters (intensity). The big size and low transparency represent the stronger backdoor-based watermark and marking effect. As shown in Fig. 12(a) ∼ Fig. 12(c), we compare the unlearning verification results when trigger size becomes smaller and trigger transparency becomes higher, i.e., the marking effect gets weaker, the performance on the markers at the marking time (in light blue) is relatively low, the performance change introduced by unlearning is smaller, thus the verified unlearning effect becomes weaker. Therefore, we should choose some moderate marking parameters to enhance the marking effect, and further promote the unlearning verification credibility.



(a) (5, 0.0) (b) (2, 0.0) (c) (5, 0.6)

Fig. 12: Marking parameter (size, transparency) influence on CIFAR10 dataset. (a) shows a stronger backdoor with big size 5*5 and low transparency 0.0, (b) and (c) show a weaker backdoor with small size 2*2 or high transparency 0.6.

Influence of the number of involved participants in FL: Fig. 13(a) ∼ Fig. 13(c) presents the unlearning verification results when the number of selected clients in each round is 2, 10, 20. Through the comparison of the results, we can find that the performance difference on the ^{v}BN markers becomes smaller with more participants involved in each round of FL. The performance change caused by unlearning is more obvious when fewer participants upload their updates to the central server, as the result of average unlearning could not effectively erase so much memory (accounted for nearly half of the contribution to the global model when only 2 participants involved). Another reason is that the contribution of an individual is easily covered by others in large-system, i.e., other participants' updates would degrade the leaver's marking effect, and further reduce the performance difference caused by unlearning.



(a) 2 participants (b) 10 participants (c) 20 participants

Fig. 13: Participant number influence on CIFAR10 dataset.

C. Unlearning verification on leaving data with different levels of non-i.i.d. distribution

In our experiments, we directly use Dirichlet function [35] with hyperparameter (0.9 and 0.5) to generate the individual data blocks satisfying different levels of non-i.i.d. distribution. The smaller the hyperparameter, the more strict the non-i.i.d. setting. As Fig. 14 shows, ^{u}RT and ^{u}DP maintain an obvious performance change on the leaving data, ^{u}GGS presents a bigger change under more strict non-i.i.d. distribution, the unlearning effect of other unlearning methods is still unobvious. In a word, even though the leaving data (under a smaller hyperparameter in Dirichlet function) maintain larger different distribution with others' data, the unlearning effect solely on the leaving data is not so ideal, calling for more dedicated unlearning verification.

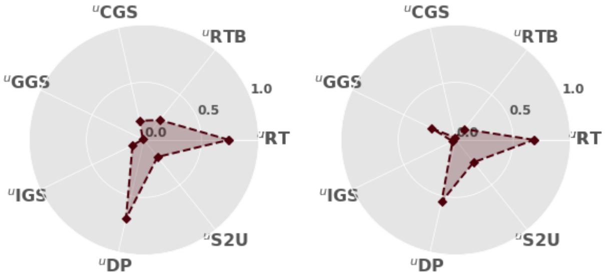


Fig. 14: Unlearning verification on leaving data with different non-i.i.d. distribution extent. The left figures show the results on non-i.i.d data distribution with hyperparameter 0.9. The right figures show the results on non-i.i.d data distribution with hyperparameter 0.5 (more strict non-i.i.d.). Both subfigures show the loss difference on the leaving data before and after unlearning by the corresponding unlearning method.

VI. Conclusion and Future Wok

In this paper, we design and implement the first open-source platform — VeriFi, a unified federated unlearning and verification framework that allows systematic analysis of the verifiable amount of unlearning with different combinations of unlearning and verification methods. Based on VeriFi, we conduct the first systematic study in the literature for verifiable federated unlearning, with 7 unlearning methods (including newly proposed ^{u}S2U) and 5 verification methods (including newly proposed ^{v}EM and ^{v}FM),covering existing, adapted and newly proposed ones for both unlearning and verification. Extensive experiments showed that our proposed ^{u}S2U is an effective, efficient and secure federated unlearning method with little time cost, storage cost and negative impact on the original FL task. The experiments also confirm the effectiveness of our proposed non-invasive ^{v}EM methods for federated unlearning verification. The combination of ^{v}EM and ^{u}S2U yields so far the most promising approach for verifiable federated unlearning without tempering with the FL process, white-box model access or raising new security risks.

Research on verifiable federated unlearning is emerging and VeriFi is able to serve as an open test bed for developing and benchmarking future federated unlearning and verification techniques. Following VeriFi, there are a rich set of research opportunities to explore further, such as new unlearning and verification methods (concurrently taking the robustness, efficacy and additional fairness issue into consideration), free leaving of multiple participants, certification of federated unlearning, etc.

References

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM CCS*, 2016, pp. 308–318.

[2] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio *et al.*, "A closer look at memorization in deep networks," in *ICML*. PMLR, 2017, pp. 233–242.

[3] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 2938–2948.

[4] P. Blanchard, R. Guerraoui, J. Stainer *et al.*, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 119–129.

[5] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan *et al.*, "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[6] L. Bourtoule, V. Chandrasekaran, C. A. Choquette-Choo, H. Jia, A. Travers, B. Zhang, D. Lie, and N. Papernot, "Machine unlearning," in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 141–159.

[7] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *International journal of medical informatics*, vol. 112, pp. 59–67, 2018.

[8] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*. IEEE, 2018, pp. 67–74.

[9] X. Cao, J. Jia, and N. Z. Gong, "Ipguard: Protecting the intellectual property of deep neural networks via fingerprinting the classification boundary," *arXiv preprint arXiv:1910.12903*, 2019.

[10] Y. Cao and J. Yang, "Towards making systems forget with machine unlearning," in *2015 IEEE Symposium on Security and Privacy*. IEEE, 2015, pp. 463–480.

[11] M. Chen, Z. Zhang, T. Wang, M. Backes, M. Humbert, and Y. Zhang, "When machine unlearning jeopardizes privacy," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 896–911.

[12] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv preprint arXiv:1712.05526*, 2017.

[13] N. C. Codella, D. Gutman, M. E. Celebi, B. Helba, M. A. Marchetti, S. W. Dusza, A. Kalloo, K. Liopyris, N. Mishra, H. Kittler *et al.*, "Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic)," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE, 2018, pp. 168–172.

[14] M. Combalia, N. C. Codella, V. Rotemberg, B. Helba, V. Vilaplana, O. Reiter, C. Carrera, A. Barreiro, A. C. Halpern, S. Puig *et al.*, "Bcn20000: Dermoscopic lesions in the wild," *arXiv preprint arXiv:1908.02288*, 2019.

[15] R. L. Davis and Y. Zhong, "The biology of forgetting—a perspective," *Neuron*, vol. 95, no. 3, pp. 490–503, 2017.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[17] E. Dong, H. Du, and L. Gardner, "An interactive web-based dashboard to track covid-19 in real time," *The Lancet infectious diseases*, vol. 20, no. 5, pp. 533–534, 2020.

[18] GDPR, "Right to erasure (right to be forgotten)," 2017. [Online]. Available: https://gdpr-info.eu/art-17-gdpr/

[19] A. Ginart, M. Guan, G. Valiant, and J. Y. Zou, "Making ai forget you: Data deletion in machine learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 3518–3531.

[20] J. Goldberger, S. Gordon, H. Greenspan *et al.*, "An efficient image similarity measure based on approximations of kl-divergence between two gaussian mixtures." in *ICCV*, vol. 3, 2003, pp. 487–493.

[21] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[22] R. Guerraoui, S. Rouault *et al.*, "The hidden vulnerability of distributed learning in byzantium," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3521–3530.

[23] E. L. Harding, J. J. Vanto, R. Clark, L. Hannah Ji, and S. C. Ainsworth, "Understanding the scope and impact of the california consumer privacy act of 2018," *Journal of Data Protection & Privacy*, vol. 2, no. 3, pp. 234–253, 2019.

[24] Y. He, G. Meng, K. Chen, J. He, and X. Hu, "Deepobliviate: A powerful charm for erasing data residual memory in deep neural networks," *arXiv preprint arXiv:2105.06209*, 2021.

[25] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[26] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[27] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1885–1894.

[28] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[30] G. Liu, X. Ma, Y. Yang, C. Wang, and J. Liu, "Federaser: Enabling efficient client-level data removal from federated learning models," in *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*. IEEE, 2021, pp. 1–10.

[31] G. Liu, Y. Yang, X. Ma, C. Wang, and J. Liu, "Federated unlearning," *arXiv preprint arXiv:2012.13891*, 2020.

[32] Y. Liu, Z. Ma, X. Liu, and J. Ma, "Learn to forget: User-level memorization elimination in federated learning," *arXiv preprint arXiv:2003.10933*, 2020.

[33] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[34] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.

[35] T. Minka, "Estimating a dirichlet distribution," 2000.

[36] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," 2015.

[37] P. Regulation, "General data protection regulation," *Intouch*, 2018.

[38] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

[39] S. Shintre, K. A. Roundy, and J. Dhaliwal, "Making machine learning forget," in *Annual Privacy Forum*. Springer, 2019, pp. 72–83.

[40] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 3–18.

[41] D. M. Sommer, L. Song, S. Wagh, and P. Mittal, "Towards probabilistic verification of machine unlearning," *arXiv preprint arXiv:2003.04247*, 2020.

[42] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan, "Can you really backdoor federated learning?" *arXiv preprint arXiv:1911.07963*, 2019.

[43] M. Toneva, A. Sordoni, R. T. d. Combes, A. Trischler, Y. Bengio, and G. J. Gordon, "An empirical study of example forgetting during deep neural network learning," *arXiv preprint arXiv:1812.05159*, 2018.

[44] P. Tschandl, C. Rosendahl, and H. Kittler, "The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific data*, vol. 5, no. 1, pp. 1–9, 2018.

[45] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, 2017, pp. 269–277.

[46] G. Wang, "Interpret federated learning with shapley values," *arXiv preprint arXiv:1905.04519*, 2019.

[47] H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J.-y. Sohn, K. Lee, and D. Papailiopoulos, "Attack of the tails: Yes, you really can backdoor federated learning," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[48] L. Wang, Z. Q. Lin, and A. Wong, "Covid-net: a tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images," *Scientific Reports*, vol. 10, no. 1, p. 19549, Nov 2020. [Online]. Available: https://doi.org/10.1038/s41598-020-76550-z

[49] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[50] WeBank, "Webank and swiss re signed cooperation mou," 2019. [Online]. Available: https://finance.yahoo.com/news/webank-swiss-signed-cooperation-mou-112300218.html

[51] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[52] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *Journal of Healthcare Informatics Research*, vol. 5, no. 1, pp. 1–19, 2021.

[53] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[54] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5650–5659.

[55] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *ICLR*, 2017.

[56] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, pp. 159–172.

[57] F. Zheng, G. Zhang, and Z. Song, "Comparison of different implementations of mfcc," *Journal of Computer science and Technology*, vol. 16, no. 6, pp. 582–589, 2001.

Appendices

A. Parameter

MNIST is a popular digit recognition dataset, adding up to 70000 32*32 gray images. Among them, 60000 pictures are used for training, and 10000 for testing [51]. **Cifar 10** is a widely recognized object classification dataset, consisting of 10 classes of 32*32 RGB pictures, 50000 for training and 10000 for testing [28]. **ISIC** is composed of 4 classes skin images (Basal cell carcinoma, Melanoma, Benign keratosis-like keratosis and Melanocytic nevus), each class contains 2000 pictures, 1600 for training and 400 for testing [44], [13], [14]. **COVID** is the chest x-ray lung images, classified into 3 categories, including 1699 COVID-19, 6069 pneumonia, 8851 normal samples [48]. We randomly sample 20 and 10 classes from VGGFace and ImageNet to compose **VGGFace_mini** and **ImageNet_mini** [8], [16]. In **VGGFace_mini**, there are totally 7023 224 *224 face images, 4916 for training, 2107 for testing. **ImageNet_mini** is composed by 13500 224 *224 RGB images, 13000 for training, 500 for testing. **Speechcommand** is composed by 32 *32 spectrograms after MFCC [57], 37005 for training, 9251 for testing. To stimulate the non-i.i.d. data distribution, we also provide the Dirichlet distribution function [35] to supply the unbalanced data to each participant. The corresponding models are summarized in Table III.

Different unlearning methods need diverse parameters. The (ε, δ) parameter pair used in ^{u}DP is set $(0.2, 0)$ for MNIST, and $(0.1, 0)$ for other datasets. λ used in ^{u}GS is 0.01 for all datasets. The scaling ratio α and β in ^{u}S2U is set as 0.1 and 1 for all datasets. Table IX summarizes the hyper-parameters used for different verification methods.

TABLE IX: Parameters of verification

Verification	**Parameter**	**Setting**
Training	learning rate	0.01
	mark epoch	100 for MNIST and CIFAR10 (50 for others)
	optimizer	SGD
	momentum	0.9
	weight decay	2.00E-04
^{v}ME	embedding dim	64
	embedding layer	features.conv1 (conv0 for ISIC)
	projetcted matrix	random
	penality loss ratio	0.05
^{v}BN	size	5 for 32*32 (25 for 224*224)
	alpha	0
	toxic data percent	10% for others and 30% for CIFAR10
	backdoor learning rate	0.01 for others and 0.05 for CIFAR10
	target class	0
^{v}SF	the size of $\widetilde{D}$	400
	the size of D_{a}^{m}	20
^{v}BF	number of boundary data	100
	loss gap γ	0.01
^{v}FM	forgettable memory ratio	0.1
^{v}EM	errorneous sample propportion κ	0.1

B. Federated Unlearning Verification by Comparing Parameter Differences

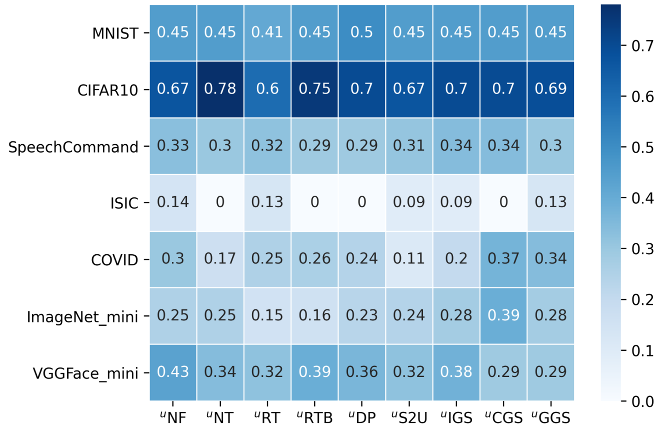
We compare the model parameter deviation before and after unlearning in Table X. We could observe that except ^{u}DP, ^{u}RT and ^{u}S2U, *most unlearning methods have similar parameter deviation with ^{u}NT and ^{u}NF*, which means model parameters fail to reliably verify whether unlearning is effective. Thus, the unlearning effect cannot be directly observed from the model parameter deviation.

C. Federated Unlearning Verification by Membership Inference

We adopt membership inference in [40] to verify unlearning from the perspective of privacy. In our experiments, the global model at the end of FL is treated as the shadow model. The training data and test data are directly regarded as the member and non-member data to simplify the process. As shown in Fig. 15, verifying federated unlearning from the perspective of privacy is infeasible. Since even after being unlearned by ^{u}RT, which absolutely and completely removes the leaving data and retrains from scratch, the deduced membership ratio still would not drop significantly than ^{u}NT, sometimes even higher. As for the reason, the similar data may belong to other participants in the federation, increasing the difficulty to verify unlearning in the aspect of privacy. It is worth mentioning that unlearning itself would cause the extra privacy concerns, which is out of scope of VeriFi, we recommend reading the work [11].

TABLE X: The model parameter difference after unlearning

Dataset	**Method**	**^{u}NF**	**^{u}NT**	**^{u}RT**	**^{u}RTB**	**^{u}CGS**	**^{u}GGS**	**^{u}IGS**	**^{u}DP**	**^{u}S2U**
CIFAR10	Euclidean Distance	1.44	1.55	29.37	1.48	1.52	1.47	1.49	334.33	3.47
	Cosine Similarity([0,1])	0.99	0.99	0.60	0.99	0.99	0.99	0.99	0.50	0.98
SpeechCommand	Euclidean Distance	2.23	2.29	5.09	2.00	2.56	2.23	2.13	34.90	2.40
	Cosine Similarity([0,1])	0.87	0.92	0.76	0.93	0.88	0.94	0.94	0.38	0.95
Covid	Euclidean Distance	1.49	1.77	91.22	1.52	2.30	2.03	1.51	334.28	5.54
	Cosine Similarity([0,1])	0.97	0.97	0.53	0.97	0.96	0.96	0.98	0.47	0.92
VGGFace_mini	Euclidean Distance	3.39	3.57	109.03	3.34	3.49	3.38	3.36	334.43	13.95
	Cosine Similarity([0,1])	0.93	0.93	0.35	0.93	0.93	0.93	0.93	0.44	0.86



	^{u}NF	^{u}NT	^{u}RT	^{u}RTB	^{u}DP	^{u}S2U	^{u}IGS	^{u}CGS	^{u}GGS
MNIST	0.45	0.45	0.41	0.45	0.5	0.45	0.45	0.45	0.45
CIFAR10	0.67	0.78	0.6	0.75	0.7	0.67	0.7	0.7	0.69
SpeechCommand	0.33	0.3	0.32	0.29	0.29	0.31	0.34	0.34	0.3
ISIC	0.14	0	0.13	0	0	0.09	0.09	0	0.13
COVID	0.3	0.17	0.25	0.26	0.24	0.11	0.2	0.37	0.34
ImageNet_mini	0.25	0.25	0.15	0.16	0.23	0.24	0.28	0.39	0.28
VGGFace_mini	0.43	0.34	0.32	0.39	0.36	0.32	0.38	0.29	0.29

Fig. 15: Verifying unlearning from the perspective of privacy with each row representing one dataset, each column representing one unlearning method. The value represents the membership ratio between the deduced member data in the leaving data and all the leaving data.

D. Security risk of ^{v}BN

Apart from working as a watermark to verify unlearning, ^{v}BN itself is a traditional backdoor attack widely studied in [42], [3], [47]. Fig. 16 shows that even at the end of FL, the backdoor (any sample patched with the trigger would be classified into the target class) still exists. Some unlearning methods cannot completely remove the security risk caused by the invasive marking method. Specifically, even with unlearning, the attack success rate of backdoor-based watermark even reaches 50%. Fortunately, the security threat can be removed with the robust aggregation rules, such as Krum and Median, as shown in Fig. 16.

Attack success rate on the ^{v}BN markers

Dataset	**^{u}NF**	**^{u}CGS**	**^{u}GGS**	**^{u}IGS**
MNIST	58.17	59.33	56.17	58.17
CIFAR-10	30.4	30.2	30.4	30.2
SpeechCommand	54.55	29.8	30.4	49.35

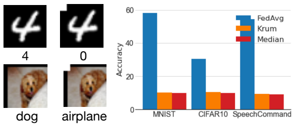
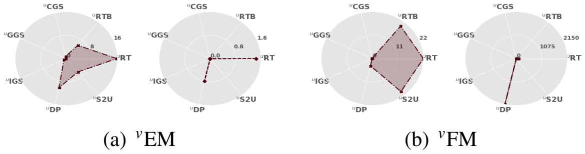


Fig. 16: Security threat of ^{v}BN

E. Verification effect difference between unique memory samples and leaving data

As shown in Section III-C1, we have presented the details and causes of choosing the particular unique memory samples as the markers. Here, we discuss the concrete unlearning verification effect difference between leveraging the unique memory samples and the leaving data. Fig. 17 shows that the selected memory markers could maintain the better unlearning verification effect, since they can identify the unlearning performance of other unlearning methods besides ^{u}RT and ^{u}DP.



(a) ^{v}EM (b) ^{v}FM

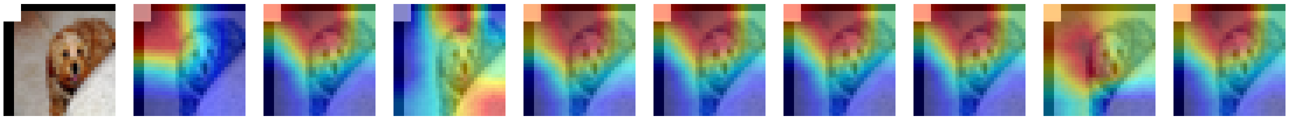
Fig. 17: Unlearning verification effect difference between leveraging the unique memory samples and leaving data on CIFAR10 dataset. (a) shows the unlearning verification effect of erroneous memory ^{v}EM markers and leaving data, (b) shows the unlearning verification effect of forgettable memory ^{v}FM markers and leaving data.

F. Unlearning verification visualization

We show the unlearning verification effect via leveraging the interpretability technique — Grad-CAM [38] in Fig. 18. These saliency maps of the selected forgotten individual sample, patched with a 5*5 white square trigger in ^{v}BN, are computed based on the global model at the end of FL. The original label of the sample is dog and the target class of the backdoor example is airplane. Before unlearning, the memory about ι still exists as the backdoor sample is classified as the target class and the high attention area is mainly located on the trigger (see Fig. 18(b)). As Fig. 18 shows, ^{u}RT obtains the most explicit unlearning effect since the attention on the trigger, caused by the leaver ι, totally disappears. ^{u}RTB, ^{u}DP and ^{u}S2U apparently degrade the high attention on the trigger, not ideal like ^{u}RT. The weakened attention can be owed to the gradually eliminated memory about the backdoor watermark introduced by ι. The attention decrease on the trigger can also be observed in ^{u}NF, ^{u}CGS, ^{u}GGS and ^{u}IGS, however, not so obvious as other unlearning methods, as the result of unsatisfied unlearning. Benefiting from the verification method, the unlearning effect can be explicitly presented with the visualization technique.

G. Theoretical Explanation

We provide a theoretical unlearning verification explanation to help understand why unlearning could be verified on the ^{v}BF markers or from the perspective of privacy.



(a) Example (b) Before Unlearning (c) ^{u}NF (d) ^{u}RT (e) ^{u}RTB (f) ^{u}CGS (g) ^{u}GGS (h) ^{u}IGS (i) ^{u}DP (j) ^{u}S2U

Fig. 18: Unlearning verification visualization in ^{v}BN — CIFAR10

^{u}S2U and ^{v}BF: We provide a simple theoretical explanation of how ^{u}S2U quantitatively changes the decision boundary of the global model characterized by boundary examples (^{v}BF), which can then be verified with our proposed metrics.

We assume that the central server applies FedAvg aggregation rule to update the global model of the next round:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \frac{1}{n}\sum_{i\in[n]}(\boldsymbol{w}_{t+1}^{(i)} - \boldsymbol{w}_t) \tag{10}$$

^{u}S2U lowers the leaving participant a's contribution to the global model by reducing a's local update:

$$\varphi\left(\boldsymbol{w}_{t+1}^{(\mathsf{a})} - \boldsymbol{w}_t\right) = \alpha\left(\boldsymbol{w}_{t+1}^{(\mathsf{a})} - \boldsymbol{w}_t\right),\ t = t_u, \tag{11}$$

$$\forall_{j\in C\diagup\mathsf{a}}\left(\boldsymbol{w}_{t+1}^{(j)} - \boldsymbol{w}_t\right) = \beta\left(\boldsymbol{w}_{T_{enabled}} - \boldsymbol{w}_t\right),\ t = t_u, \tag{12}$$

We set $\mathsf{a} = n$ and $\beta = 1$ to facilitate understanding of the explanation. ^{v}BF characterizes the decision boundary of the local model $f^{(\mathsf{a})}$ using a subset of perturbed training samples close to the decision boundary. Arguably, the samples with relatively high and close top-2 class probabilities are boundary samples. ^{v}BF generates the boundary markers satisfying:

$$D_{\mathsf{a}}^{m} = \{(\boldsymbol{x},y)|(\boldsymbol{x},y)\in D_{\mathsf{a}}, |f_{top-1}^{(\mathsf{a})}(\boldsymbol{x}+\sigma) - f_{top-2}^{(\mathsf{a})}(\boldsymbol{x}+\sigma)| \le \gamma\}, \tag{13}$$

where, $f_{top-1}^{(\mathsf{a})}(\boldsymbol{x}+\sigma)$ and $f_{top-2}^{(\mathsf{a})}(\boldsymbol{x}+\sigma)$ denote the top-1 and top-2 class probabilities respectively, σ is the generated perturbation by PGD and $\gamma \in [0, 0.1)$ is a small positive value defining how close the two probabilities. Since unlearning and verification are activated after $T_{enabled}$ in VeriFi, when the global model has converged to a good solution, we can make a reasonable assumption that the boundary samples of a's local model could also work as boundary samples to the converged global model $\boldsymbol{w}_{t+1}$.

To simplify the complex derivation, we choose a binary classifier (assumed linearly around $\{\boldsymbol{x}_i, y_i\} \in D_{\mathsf{a}}^m, y_i \in \{\pm 1\}$), $f(\boldsymbol{x}_i) = \boldsymbol{w}_t^T\boldsymbol{x}_i + b$,

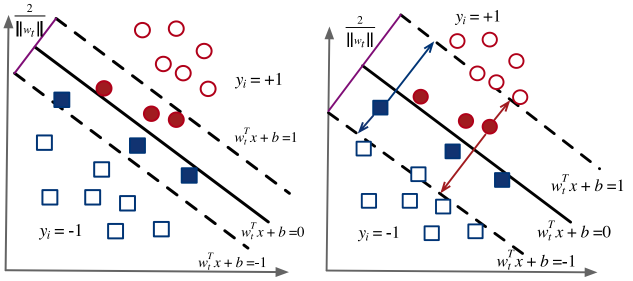
$$y_i = \begin{cases} +1, & \boldsymbol{w}_t^T\boldsymbol{x}_i + b \ge 1 \\ -1, & \boldsymbol{w}_t^T\boldsymbol{x}_i + b \le -1 \end{cases} \tag{14}$$

Then, we get $y_i(\boldsymbol{w}_t^T\boldsymbol{x}_i + b) \ge 1$, the decision boundary distance between the two classes $\{+1, -1\}$ is $\frac{2}{\|\boldsymbol{w}_t\|}$, the optimizer would optimize $\frac{2}{\|\boldsymbol{w}_t\|}$ to enlarge the distance of decision boundary between two neighbor classes : $\max_{\boldsymbol{w}_t}\{\frac{2}{\|\boldsymbol{w}_t\|}\}$

After launching ^{u}S2U at t, the global model at $t+1$ is:

$$\begin{aligned} \boldsymbol{w}_{t+1} &= \boldsymbol{w}_t + \frac{1}{n}\left(\sum_{i\in[n-1]}(\boldsymbol{w}_{t+1}^{(i)} - \boldsymbol{w}_t) + \varphi\left(\boldsymbol{w}_{t+1}^{(n)} - \boldsymbol{w}_t\right)\right) \\ &= \boldsymbol{w}_t + \frac{\sum_{i\in[n]}(\boldsymbol{w}_{t+1}^{(i)} - \boldsymbol{w}_t) + (\alpha - 1)(\boldsymbol{w}_{t+1}^{(n)} - \boldsymbol{w}_t)}{n} \\ &\approx \boldsymbol{w}_t + \frac{\sum_{i\in[n-1]}(\boldsymbol{w}_{t+1}^{(i)} - \boldsymbol{w}_t)}{n} \end{aligned} \tag{15}$$

The average local update from other $n-1$ participants is: $\overline{\boldsymbol{w}_{t+1}^{[n-1]}} = \frac{\sum_{i\in[n-1]}\boldsymbol{w}_{t+1}^{(i)}}{n-1}$, the global model update at $t+1$: $\boldsymbol{w}_{t+1} - \boldsymbol{w}_t = \frac{(n-1)\overline{\boldsymbol{w}_{t+1}^{[n-1]}}}{n} < \overline{\boldsymbol{w}_{t+1}^{[n-1]}}$, then ^{u}S2U works by scaling down/up his own/others' update and further influences the global model update. The hyper-parameter α used in our experiment is 0.1. Since the global model $\boldsymbol{w}_{t+1}$ is reduced after ^{u}S2U, then the distance between the two classes of decision boundary $\frac{2}{\|\boldsymbol{w}\|}$ is enlarged, as shown in Fig. 19. Thus, the results on the original constructed boundary samples would change, we then focus on the result change to verify whether unlearning is successful.



(a) Decision boundary before unlearning (b) Decision boundary after unlearning

Fig. 19: Decision boundary change after unlearning

^{u}DP and privacy: We analyze the unlearning effect of ^{u}DP from the privacy perspective, specifically, the information about whether the sample $\boldsymbol{z}$ is in a's local training data is limited. Once ^{u}DP works as an unlearning method, the upper threshold of membership information about $\boldsymbol{z}$ is limited. The data of a is $D_{\mathsf{a}} \sim \mathbb{D}$, $\mathbb{D}$ denotes distribution of D_{a}, the inferer is Inf, $\boldsymbol{w}_t$ is the global model, the model of a is $\boldsymbol{w}_t^{\mathsf{a}} = \boldsymbol{w}_t(D_{\mathsf{a}})$, b is uniformly chosen from $\{0, 1\}$ which denotes whether $\boldsymbol{z}$ belongs to a. $b = 0$ if $\boldsymbol{z} \sim \mathbb{D}$, $b = 1$ if $\boldsymbol{z} \sim D_{\mathsf{a}}$. The membership inference result can be expressed as:

$$Mem(Inf, \boldsymbol{w}_t, m, \mathbb{D}) = \begin{cases} 1, & Inf(\boldsymbol{z}, \boldsymbol{w}_{\mathsf{a}}, m, \mathbb{D}) = b \\ 0, & Inf(\boldsymbol{z}, \boldsymbol{w}_{\mathsf{a}}, m, \mathbb{D}) \ne b \end{cases} \tag{16}$$

Then the membership advantage of Inf can be expressed as the difference between Inf's true and false positive rate:

$$Mem\_Adv(Inf, \boldsymbol{w}_t, m, \mathbb{D}) = Pr[Inf = 0|b = 0] - Pr[Inf = 0|b = 1] \tag{17}$$

Then we give a short theoretical explanation of ^{u}DP could impose a strict limit on the information about a:

$$Mem\_Adv(Inf, \boldsymbol{w}_t, m, \mathbb{D}) \leq e^{\varepsilon} - 1 \tag{18}$$

Given $D_{\mathsf{a}} = (z_1, \cdots, z_m)$ and $\boldsymbol{z} \sim D$, then $D_{\mathsf{a}}^{'} = (z_1, \cdots, z_{i-1}, \boldsymbol{z}, z_{i+1}, \cdots, z_m)$, $\boldsymbol{w}_t^{\mathsf{a}'} = \boldsymbol{w}_t(D_{\mathsf{a}}^{'})$. $Inf(\boldsymbol{z}, \boldsymbol{w}_t^{\mathsf{a}}, m, \mathbb{D})$ and $Inf(z_i, \boldsymbol{w}_t^{\mathsf{a}'}, m, \mathbb{D})$ have identical distributions for all $i \in [m]$, thus,

$$Pr[Inf = 0|b = 0] = 1 - \mathbb{E}[\frac{1}{m}\sum_{i=1}^{m} Inf(z_i, \boldsymbol{w}_t^{\mathsf{a}}, m, \mathbb{D})] \tag{19}$$

$$Pr[Inf = 0|b = 1] = 1 - \mathbb{E}[\frac{1}{m}\sum_{i=1}^{m} Inf(z_i, \boldsymbol{w}_t^{\mathsf{a}'}, m, \mathbb{D})] \tag{20}$$

Then,

$$Mem\_Adv = \mathbb{E}[\frac{1}{m}\sum_{i=1}^{m}(Inf(z_i, \boldsymbol{w}_t^{\mathsf{a}'}, m, \mathbb{D}) - Inf(z_i, \boldsymbol{w}_t^{\mathsf{a}}, m, \mathbb{D}))] \tag{21}$$

Assume that the local models of k participants in a federated learning round is $\boldsymbol{w}_t^1, \cdots, \boldsymbol{w}_t^k$. ^{u}DP ensures that for all $j \in [k]$,

$$Pr[\boldsymbol{w}_t^{\mathsf{a}'} = \boldsymbol{w}_t^j] \leq e^{\varepsilon} Pr[\boldsymbol{w}_t^{\mathsf{a}} = \boldsymbol{w}_t^j] \tag{22}$$

Thus the membership advantage can be written as:

$$\begin{aligned}
&\sum_{j=1}^{k} \mathbb{E}[\frac{1}{m}\sum_{i=1}^{m}(Pr[\boldsymbol{w}_t^{\mathsf{a}'} = \boldsymbol{w}_t^j] Inf(z_i, \boldsymbol{w}_{\mathsf{a}}^{'}, m, \mathbb{D}) - Pr[\boldsymbol{w}_t^{\mathsf{a}} = \boldsymbol{w}_t^j] Inf(z_i, \boldsymbol{w}_t^{\mathsf{a}}, m, \mathbb{D}))] \\
&= \sum_{j=1}^{k} \mathbb{E}[\frac{1}{m}\sum_{i=1}^{m}(Pr[\boldsymbol{w}_t^{\mathsf{a}'} = \boldsymbol{w}_t^j] Inf(z_i, \boldsymbol{w}_t^j, m, \mathbb{D}) - Pr[\boldsymbol{w}_t^{\mathsf{a}} = \boldsymbol{w}_t^j] Inf(z_i, \boldsymbol{w}_t^j, m, \mathbb{D}))] \\
&= \sum_{j=1}^{k} \mathbb{E}[\frac{1}{m}\sum_{i=1}^{m}(Pr[\boldsymbol{w}_t^{\mathsf{a}'} = \boldsymbol{w}_t^j] - Pr[\boldsymbol{w}_t^{\mathsf{a}} = \boldsymbol{w}_t^j]) Inf(z_i, \boldsymbol{w}_t^j, m, \mathbb{D})] \\
&\leq \sum_{j=1}^{k} \mathbb{E}[\frac{1}{m}\sum_{i=1}^{m}(e^{\varepsilon} Pr[\boldsymbol{w}_t^{\mathsf{a}} = \boldsymbol{w}_t^j] - Pr[\boldsymbol{w}_t^{\mathsf{a}} = \boldsymbol{w}_t^j]) Inf(z_i, \boldsymbol{w}_t^j, m, \mathbb{D})] \\
&= \sum_{j=1}^{k} \mathbb{E}[\frac{1}{m}\sum_{i=1}^{m}(e^{\varepsilon} - 1) Pr[\boldsymbol{w}_t^{\mathsf{a}} = \boldsymbol{w}_t^j] Inf(z_i, \boldsymbol{w}_t^j, m, \mathbb{D})]
\end{aligned} \tag{23}$$

Then, Mem_Adv must be smaller than $e^{\varepsilon} - 1$ since $Inf(z_i, \boldsymbol{w}_t^j, m, \mathbb{D}) \leq 1$, meaning the membership inference advantage is limited by the upper threshold. Thus, after unlearned by ^{u}DP, the membership information about a is limited, unveiling the effectiveness of unlearning from the perspective of privacy.