TruVRF: Toward Triple-Granularity Verification on Machine Unlearning

Chunyi Zhou, Yansong Gao, *Senior Member, IEEE*, Anmin Fu, *Member, IEEE*, Kai Chen, *Member, IEEE*, Zhi Zhang, *Member, IEEE*, Minhui Xue, Zhiyang Dai, Shouling Ji, *Member, IEEE*, and Yuqing Zhang, *Member, IEEE*

***Abstract*—The right to be forgotten has incentivized machine unlearning, but a key challenge persists: the lack of reliable methods to verify unlearning conducted by model providers. This gap facilitates dishonest model providers to deceive data contributors. Current approaches often rely on invasive methods like backdoor injection. However, it poses security concerns and is also inapplicable to legacy data—already released data. To tackle this challenge, this work initializes the first non-invasive unlearning verification framework which operates at triple-granularity (class-, volume-, sample-level) to assess the data facticity and volume integrity of machine unlearning. In this paper, we propose a framework, named TruVRF, encompasses three Unlearning-Metrics, each tailored to counter different types of dishonest model providers or servers (Neglecting Server, Lazy Server, Deceiving Server). TruVRF leverages non-invasive model sensitivity to enable multi-granularity verification of unlearning. Specifically, Unlearning-Metric-I checks if the removed class matches the data contributor's unlearning request, Unlearning-Metric-II measures the amount of unlearned data, and Unlearning-Metric-III validates the correspondence of a specific unlearned sample with the requested deletion. We conducted extensive evaluations of TruVRF efficacy across three datasets, and notably, we also evaluated the effectiveness and computational overhead of TruVRF in real-world applications for the face recognition dataset. Our experimental results demonstrate that TruVRF achieves robust verification performance: Unlearning-Metric-I and -III achieve over 90% verification accuracy on average against dishonest servers, while Unlearning-Metric-II maintains an inference deviation within 4.8% to 8.2%. Additionally, TruVRF demonstrates generalizability across diverse conditions, including varying numbers of unlearned classes and sample volumes. Significantly, TruVRF is applied to two state-of-the-art unlearning frameworks: SISA (presented at Oakland'21) and Amnesiac Unlearning, representing exact and approximate unlearning methods, respectively, which affirm TruVRF's practicality. In addition, we conducted extensive evaluations around TruVRF, including ablation experiments, trade-offs in computational overhead, and the robustness of model sensitivity, among others.**

***Index Terms*—Machine unlearning, unlearning verification, dishonest unlearning.**

I. Introduction

Recent privacy regulations, such as the European Union's General Data Protection Regulation (GDPR) [32], California Privacy Rights Act (CPRA) [41], and Personal Information Protection and Electronic Documents Act (PIPEDA) [4], empower users with the right to delete their private data from entities storing it, known as the *right-to-be-forgotten*. In the context of Machine Learning (ML), this right necessitates model providers to erase any trace of data requested to be forgotten from the model [53], driving research into the field of eliminating the impact of data on models, termed *machine unlearning*. In recent years, machine unlearning has garnered increasing attention from both academia and industry [6], [12], [16], [17], [19], [23], [31].

Intuitively, a straightforward approach to machine unlearning involves retraining the entire ML model from scratch using a dataset where the samples to be forgotten are excluded. However, this method involves unacceptable computational overhead, particularly for large-scale datasets, as highlighted [29], [30], [33], which is difficult to use. Existing frameworks that aim to overcome such a challenge can be categorized into two main categories: exact unlearning and approximate unlearning. Exact unlearning divides a given dataset into multiple non-overlapped blocks and retrains the block(s) with forgotten sample(s) deleted, thus reducing the computa-

Received 14 August 2024; revised 26 January 2025, 26 March 2025, and 22 April 2025; accepted 23 April 2025. Date of publication 30 April 2025; date of current version 15 May 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFB3102100; in part by NSFC-Yeqisun Science Foundation under Grant U244120033; in part by the National Natural Science Foundation of China under Grant U24A20336, Grant 62372236, Grant 62172243, Grant 62402425, and Grant 62402418; in part by China Postdoctoral Science Foundation under Grant 2024M762829; in part by the Open Foundation of the State Key Laboratory of Integrated Services Networks under Grant ISN24-15; in part by the Qing Lan Project of Jiangsu Province; in part by the Zhejiang Provincial Natural Science Foundation under Grant LD24F020002; and in part by the Zhejiang Provincial Priority- Funded Postdoctoral Research Project under Grant ZJ2024001. The associate editor coordinating the review of this article and approving it for publication was Dr. Hossein Pishro-Nik. *(Chunyi Zhou and Yansong Gao are co-first authors.) (Corresponding author: Anmin Fu.)*

Chunyi Zhou is with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China, and also with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: zhouchunyi@zju.edu.cn).

Yansong Gao and Zhi Zhang are with the School of Computer Science and Software Engineering, The University of Western Australia, Perth, WA 6009, Australia (e-mail: garrison.gao@uwa.edu.au; zhi.zhang@uwa.edu.au).

Anmin Fu is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China, and also with the School of Cyber Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: fuam@njust.edu.cn).

Kai Chen is with Chinese Academy of Sciences, Beijing 100864, China (e-mail: chenkai@iie.ac.cn).

Minhui Xue is with Data61, CSIRO, Sydney, NSW 2122, Australia (e-mail: minhuixue@gmail.com).

Zhiyang Dai is with the School of Cyber Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: dzy@njust.edu.cn).

Shouling Ji is with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China (e-mail: sji@zju.edu.cn).

Yuqing Zhang is with the National Computer Network Intrusion Protection Center, University of Chinese Academy of Sciences, Beijing 101408, China (e-mail: zhangyq@ucas.ac.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TIFS.2025.3565991, provided by the authors.

Digital Object Identifier 10.1109/TIFS.2025.3565991

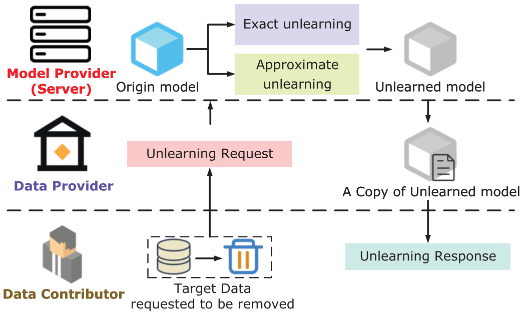


Fig. 1. Machine unlearning workflow.

tion overhead significantly. Approximate unlearning modifies model parameters to conceal the impact of forgotten samples, eliminating the difference in model parameters between an unlearned model and a model that simply does not learn the forgotten samples.

When considering Machine Learning as a Service (MLaaS), many model providers face challenges in acquiring a sufficient amount of high-quality data internally [10], [20]. As a result, they often rely on outsourcing data from data providers such as Amazon Data Exchange [35], Kaggle [25]. A data provider acts as an intermediary between different model providers and many data contributors. The same contributor's data might be provided to different model providers by the data provider. To ensure compliance with *the forgotten right*, the data provider is required to provide transparent privacy policies and usage terms that clearly define the control of the data contributors on their personal data when they supply their data to the data provider. In particular, the contractual agreement between the data provider and data contributors should clearly outline the obligations and responsibilities with respect to *the forgotten right*. After data contributors supply their data to the data provider, a data contributor still has the right to request data removal from any models that train on such requested data. In addition, the contractual agreement between data provider and model provider requires that the model provider gives a trained model to the data provider for transparency and facilitating later auditing or regulation.

The workflow of machine unlearning under such an MLaaS is shown in Figure 1. Whenever a data contributor needs to delete his/her data, he/she sends the request to the data provider. The data provider then asks all model providers who have used this contributor's requested data to perform machine unlearning on their models. The model provider returns the unlearned model to the data provider once the model unlearning has been done.

However, current machine unlearning frameworks *face the fundamental challenge of lacking a means of verifying the unlearning process*. The data provider has no other option but to trust that the model provider has honestly forgotten the data as requested. The absence of a verification mechanism renders hesitation or outright refusal by data contributors to share their data. Once data is contributed, contributors lose control over it, exposing their data to privacy risks that cannot be reversed. Specifically, the model provider may act semi-honestly, retaining interest in the data requested by the contributor while avoiding its removal from the model. In some cases, the model provider may simply disregard the data contributor's request to forget the data, or opt to partially unlearn or even unlearn unrelated data to deceive the data contributor. This behavior is rational in practical scenarios, especially when the data contributor seeks to withdraw valuable data crucial to the model provider's interests. In this context, we identify three types of dishonest model providers (hereinafter referred to as "the server"):

- `Neglecting Server`: The server simply ignores the unlearning request of the data provider and retains the model intact without any operation.
- `Lazy Server`: The server executes the unlearning request of the data provider partially. That is, only selectively forgetting part of the contributor's data.
- `Deceiving Server`: The server executes the unlearning request of the data provider, but forgets other irrelevant samples of the same class and volume as the data provider's unlearning request, aiming to retain the data contributor's target data within the model.

The above dishonest unlearning server attempts to violate either *Unlearning Data Facticity* or *Unlearning Volume Integrity* that an honest server should fulfill:

- `Unlearning Data Facticity`: The server should unlearn the data specified in the unlearning request.
- `Unlearning Volume Integrity`: The server should unlearn the same amount of data as the target data in the unlearning request.

To this end, we ask the following research questions:

Are there usable verification means for auditing a server's unlearning data facticity and volume integrity? If so, how efficient are they?

Key Observation: During the training phase of ML models, gradient changes play a pivotal role in guiding the convergence process. These changes, indicative of the model's adaptation to the data, are quantifiable through metrics like model sensitivity. Notably, we have observed a correlation between gradient changes and class-specific data volumes within ML models. In the context of machine unlearning, it is essential to note that the model sensitivity of an unlearned model consistently shifts following the removal of target data. This phenomenon persists under conditions where authentic machine unlearning enforcement is applied.

Our Solution. Invasive verification methods, such as watermarking based on backdoors [38], may raise concerns about model security. Additionally, it is inefficient for released data as it is not possible to add watermarks retrospectively if they were not initially added. Based on the key observation, this work attempts the first step towards addressing the above (open) challenging research question by initializing the first holistic and approachable unlearning verification framework, `TruVRF`, in a non-invasive manner. More specifically, we advocate verifying the data facticity and volume integrity of unlearning operations for the sake of unveiling the dishonest behavior by the server. `TruVRF` entitles the data provider to a viable means of auditing whether the data the contributor wants to withdraw from an ML model (i.e., namely target data in the following descriptions) is forgotten faithfully and completely (i.e., the volume of data) by the server. To ensure verification reliability, we design verification metrics considering three degrees of granularity against various types of dishonest servers: *Unlearned Class Verification*, *Unlearned Volume Verification*, and *Unlearned Sample Verification* from coarse-grained to fine-grained. In the following, unless otherwise stated, we describe TruVRF to verify the exact unlearning. Also TruVRF is applicable to approximate unlearning, as detailed in Section IV-E.

We can identify dishonest servers according to the abnormal model sensitivity change: Neglecting Server-almost unchanged, Lazy Server-less than expected, Deceiving Server-pronounced difference between the test data and target data, as detailed in Section V-D.

Methodology. Our verification framework is based on the model sensitivity and owns triple levels of granularity: Class Granularity, Volume Granularity, and Sample Granularity, which advocates a responsible auditing approach to the data provider to verify the unlearning behavior by the server. We identify three verification metrics for each granularity resilient to multiple dishonest servers:

- Unlearned Class Verification: The data provider extracts the model sensitivity of the origin model M_o and the unlearned model M_u. Through contrastive statistical analysis, the data provider can determine the data class unlearned by the server. It could provide the data provider with evidence to verify whether a target class is actually enforced by the server. We denote it as the verification with *Class Granularity*.
- Unlearned Volume Verification: The data provider trains multiple shadow models based on the target data that is divided incrementally, and extracts the model sensitivity of each shadow model. In this way, the data provider can learn the volume of unlearned data as a function of model sensitivity change (i.e., we denote it as Unlearning Measurement). Thus, given the Unlearning Measurement, the data provider can infer the volume of unlearned data by the server. We denote it as the verification with *Volume Granularity*.
- Unlearned Sample Verification: This metric accomplishes the unlearning data facticity in the level of sample granularity, i.e., the distinguishability between the data contributor-targeted samples and irrelevant samples in the unlearned model M_u, which can be used to verify whether the unlearned samples are target samples. The data provider needs to determine the existence of the target sample in unlearned model M_u. For the honest model, the model sensitivity extracted by test data and target data is almost the same. But when the server tricks the data provider so that the target samples still exist in the unlearned model M_u, these two model sensitivities can produce a distinguishable gap. Thus, based on this key observation, we propose the Unlearned Sample Verification to verify the existence of the target sample in unlearned model M_u. We denote it as the verification with *Sample Granularity*.

To this end, we shed light on thwarting the risks of deceiving data contributor and provider, and threatening privacy by the server in the context of machine unlearning, and hereby summarize three types of dishonest servers. TruVRF strives to introduce a audit-wise toolkit for reviewing the dishonest servers on the model copy they provide, and seeks to mitigate the Data Contributors' concerns that place their data under non-retrievable privacy risks in data transactions. This work is towarding to a new research line in the context of developing and deploying machine unlearning, so as to ensure the data privacy and realize *the forgotten right*. The main contributions are fourfold:

- To the best of our knowledge, we are the first that concretely examine the potentially inadvertent misconducts of the server in machine unlearning, and analyze the threats to a data contributor's privacy. Particularly, we identify three types of dishonest servers: Neglecting Server, Lazy Server, and Deceiving Server.
- We then propose TruVRF, a machine unlearning verification framework, enabling three levels of granularity based on model sensitivity, which can effectively verify data facticity and volume integrity of machine unlearning, specifically:
  - *Unlearned Class Verification* to verify the data class that the server unlearned, which is applicable towards the Neglecting Server scenario.
  - *Unlearned Volume Verification* to approximately infer the unlearned data volume by the server, which is applicable towards the Lazy Server scenario.
  - *Unlearned Sample Verification* to judge the existence of the target sample in the unlearned model M_u, so as to verify whether the server unlearns the requested sample honestly, which is applicable towards the Deceiving Server scenario.
- We evaluate TruVRF through extensive experiments on three datasets, CIFAR-10, Fashion-MNIST, and RAF-DB, which demonstrate the verification ability against the three aforementioned types of dishonest servers. Experimental results validate that the unlearning metrics can efficiently verify the unlearned class, volume, and sample. Notably, TruVRF is also validated to be effective and deployable in the real-world facial recognition application.
- We affirm TruVRF's generalisability through the verification evaluations on two state-of-the-art unlearning frameworks: SISA [3] (Oakland'21), and Amnesiac Unlearning [18] as representative exact and approximate unlearning, respectively. We further elaborate on the rationale of Unlearning-Metrics based on triple levels of granularity.

II. Preliminary and Related Work

We first introduce machine unlearning, and briefly describe exact unlearning and approximate unlearning. We then discuss existing potential metrics/methods that might be indirectly utilized for unlearning verification.

A. Machine Unlearning Preliminary

Benefiting from the recent legislation such as GDPR and CCPA, a user can legally request data controllers to delete his/her individual data, which is formalized in the right to be forgotten. In the context of machine unlearning, an ML

TABLE I
A SUMMARY OF NOTATIONS

Notation	Description
x	Target data that the data contributor requests to unlearn.
M_o	Origin ML model before machine unlearning.
M_u	Unlearned ML model after machine unlearning.
δ	Transformation on M_o in approximate unlearning.
θ_o	Model Parameter Vector in M_o
θ_u	Model Parameter Vector in M_u
α	Learning Rate
MS	Model Sensitivity
DS	Model Sensitivity Difference
UM	Unlearning Measurement
$class$	The category or label assigned to data samples.
$volume$	The amount or quantity of training data samples.

model has learned the features of user data, and cannot fulfill this right under formal "deletion". To this end, an ML owner needs to resort to machine unlearning to erase the impact of the data to be forgotten on the model. There are two categories of machine unlearning: exact unlearning and approximate unlearning. We summarize the notations in Table I to ease the following descriptions.

1) Exact Unlearning: This leads to the exact trace removal of target data (i.e., a volume of samples or just a specific sample) from the training set in the unlearned model. The most straightforward way is to train the ML model from scratch after removing the target data x from the training set D. We denote the origin ML model before unlearning as M_o, and the ML model after unlearning as M_u. Thus, exact unlearning can be formalized as:

$$M_u = M_o \backslash x = \mathsf{train}(D \backslash x). \tag{1}$$

Retraining the entire model for machine unlearning can achieve the desired goal, but it often leads to significant computational burdens due to complex model structures or large training set sizes. In multi-user scenarios where each user has an equal right to be forgotten, satisfying unlearning requests by retraining the model from scratch every time is impractical, even if the model provider/server intends to do so. As a solution to this challenge, practical unlearning frameworks like SISA [3] have been proposed to reduce the computational overhead. For example, SISA only retrains a sub-model on a small data block from which the requested data has been excluded to update the ensemble model decision.

2) Approximate Unlearning: It chooses an alternative path, focusing on how to transform the origin model into another one without the target data effect in the training set, and make the twin models infinitely close in the parameter space. Formally, we can denote it as: $M_u = M_o + \delta \approx M_o \backslash x$, where δ represents the transformation operation carried out by the model owner on M_o, such as adding noise. Approximate unlearning methods are building upon their "unlearning" on reproducing similar properties of exact unlearned models. Despite that TruVRF is mainly motivated to address the unlearning verification on exact unlearning, it is also applicable for approximate unlearning.

B. Related Work

1) Exact Unlearning: The notion of machine unlearning was first proposed by Cao et al. [5], which aims to implement the right to be forgotten and decrease the damage of poison data in the machine learning context. They designed an efficient unlearning approach by transforming learning algorithms (specifically, SVM, naïve Bayes, and decision tree) into a summation form. Cao et al. [6] proposed a causal unlearning method Karma, which searches through different subsets of training samples and returns the subset that causes the most misclassifications as the set of polluted training samples. By removing the problematic subset, Karma can reduce the impact of data pollution. However, these methods are only applicable for ML classifiers, such as SVM and naïve Bayes, while could not be implemented in a more complicated model, such as neural networks.

Later, the exact unlearning is extended to deep learning. Bourtoule et al. [3] proposed the SISA framework, which is a representative approach to date for exact unlearning. SISA divides the training data into multiple disjoint shards, and trains sub-models based on them respectively. When the unlearning invokes, SISA only needs to retrain the sub-model corresponding to the shard under which the target data falls. Following this work, Chen et al. [8] introduced GraphEraser, adapting SISA-based unlearning to graph data. Wu et al. [51] proposed DeltaGrad for efficient model retraining using cached training info. Yan et al. [54] presented ARCANE, which divides training data into shards, trains models on each, records states, and resumes training from historic states upon unlearning requests.

2) Approximate Unlearning: Approximate unlearning modifies model parameters to mimic models never trained on target data [21], [31], [56]. Graves et al. [18] proposed Amnesiac ML, relabeling sensitive data and retraining. Baumhauer et al. [2] developed linear filtration for sanitizing classifiers. Guo et al. [19] presented a Certified Removal Mechanism for linear classifiers. Izzo et al. [23] reduced unlearning time with projective updates. Ginart et al. [16] proposed K-means-based unlearning algorithms for efficient data deletion. Chundawat et al. [9] proposed zero-shot unlearning methods: error-minimizing noise and gated knowledge transfer. Golatkar et al. [17] linked Differential Privacy to SGD stability, designing selective forgetting. Warnecker et al. [49] introduced an approximate unlearning framework erasing features/labels via influence functions.

Although approximate unlearning can make the unlearned model closer to the exact unlearned model in parameter space, there are still certain limitations. Thudi et al. [45] utilized the forging technology to prove that approximate unlearning is self-contradicted. It can satisfy the definition of approximate unlearning in parameter space without any modification when the forging match is found. Meanwhile, approximate unlearning would have an impact on the model accuracy [18]. When the unlearning request is on a large scale, the model efficacy drops non-negligibly.

3) Indirect Unlearning Verification Methods: Existing indirect unlearning verification methods are ad-hoc, focusing on special attacks and proof of learning technology to verify the unlearning indirectly. Specifically, membership inference attacks [7], [36], [39] are used to verify the effectiveness of approximate unlearning [2], [18], which demonstrates that the unlearned model cannot be distinguished from the model that has never trained on the target data. Sommer et al. [38] verified the exact unlearning by a backdoor attack [14]. A user inserts a backdoor during model training. If the server retains the target data, the predictions for backdoor samples align closely with the target labels. Ullah et al. [46] proposed a backdoor-trigger and incremental learning-based machine unlearning verification method, achieving efficient, verifiable, and service-quality-preserving machine unlearning. However, there are significant limitations when using backdoor attacks to verify unlearning. Firstly, it introduces security concerns through injecting backdoors, which poses the safe usage of the model under insidious risks [11], especially for other data contributors. Secondly, as acknowledged [38], the verification accuracy drops once the server adopts the backdoor defense, such as Neural Cleanse [47]—a model-based detection defense. As machine unlearning usually refers to the data outsourcing scenario, we note that training-based defenses [28], [48] (the defender/user rather than the attacker trains the model) are expected to remove the backdoor effect, making the verification fall. Thirdly, when the Lazy Server only unlearns partial data, existing work can only verify the existence of the target data but cannot infer the specific data quantity.

Furthermore, Thudi et al. [45] improved the proof of learning [24] to the proof of unlearning. By checking the reliability of intermediate models in unlearning, they can verify whether the server is honestly executing the unlearning operation. The similarity between the approximate unlearned model and the retrained model is regarded as unlearning metrics, such as ℓ_2 distance [51] and KL divergence [17]. Despite these metrics being straightforward, there are limitations that need to train the model from the scratch, and there exist deviations which are caused by randomness and numerical instabilities in floating point operations. To solve these issues, Thudi et al. [44] further developed ℓ_2 distance as the verification metric towards approximate unlearning by expanding the SGD algorithm. In addition, a surrogate approach to verifying the efficiency of unlearning is to analyze the privacy leakage of the weight distributions in the unlearned model [19], [34]. Typically, the data privacy that the model could release is information-less as it does not exist in the model. However, the aforementioned methods have limitations, such as verification based on a backdoor attack raising security concerns and might be removed [15], [43], [47]. Moreover, the proof of unlearning is fragile [37], [55], which can achieve dishonest unlearning by tampering with the proof of unlearning. It is difficult to measure the quality of exact unlearning based on ℓ_2 Distance and KL Divergence as unlearning metrics. Except for guiding unlearned model optimization, these metrics are inadequate in revealing the misconducts of dishonest servers from a quantitative standpoint. Additionally, Guo et al. [20] proposed a TV-stable algorithm based on noisy SGD for machine unlearning, achieving efficient unlearning with theoretical guarantees on empirical and population risk trade-offs, while also ensuring differential privacy. We prefer denoting them as optimization metrics rather than verification metrics.

4) Unlearning Verification Methods Based on Cryptographic Tools: Currently, there are also some works that utilize cryptographic methods to achieve authenticity verification for machine unlearning. Sun et al. [42] propose zkDL, a scalable zero-knowledge proof framework for deep learning training, enabling efficient and privacy-preserving verification of million-parameter networks in under a second per batch update. Weng et al. [50] propose a Proof of Unlearning (PoUL) framework for verifying data deletion in machine learning, leveraging SGX enclaves to ensure correctness and prevent forging attacks. Eisenhofer et al. [13] propose a verifiable machine unlearning framework using cryptographic methods like SNARKs and hash chains. It ensures proof of correct data deletion and works for linear regression, logistic regression, and neural networks.

While these cryptography-based verification methods for machine unlearning are indeed effective in confirming the erasure of data, they exhibit limitations from aspects of threat model, computational overhead, scalability and generalizatibility. (details can be found in Supplementary Material)

The existing unlearning verification methods are not specifically tailored to verify data factuality and volume integrity effectively across various machine unlearning requests (i.e., from sample level to volume level and class level). They can only qualitatively analyze whether the server has deleted the data (in particular, TRUE or FALSE), but cannot verify quantitatively, such as accurately inferring the unlearned category, quantity, or target sample existence. In terms of security, `TruVRF` is non-invasive and does not make any modification to the model. Unlike other invasive approaches (such as backdoor), our method does not compromise the model's inherent robustness, thereby avoiding the introduction of additional security risks. Consequently, `TruVRF` initiates the first systematic study on unlearning verification and aims to use a holistic toolkit to confront dishonest servers in the context of machine unlearning.

III. Unlearning Verification of TruVRF

In this section, we first define the threat model and clarify the capabilities of involved entities (i.e., model provider or the server, data provider and data contributor), then present an overview of `TruVRF`, followed by its implementation details with three degrees of granularity.

A. Threat Model

As depicted in Figure 1, it is assumed that a model provider (the server) is often challenged to acquire a sufficient amount of high-quality data internally. Then the server relies on sourcing data from the data provider who collects data from a multitude of data contributors, e.g., through crowd-sourcing. According to data protection regulations (i.e., *the forgotten right*), data contributors can revoke the contribution of their data. The same data might be provided to different model providers by the data provider. Upon a data deletion request from a data contributor to the data provider, the latter reviews

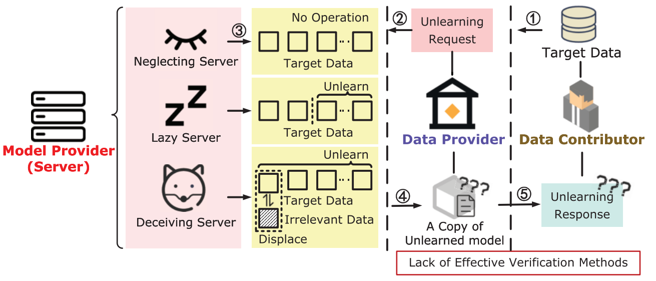


Fig. 2. A schematic view of threat model. ① A data contributor requests the data provider to delete the target data from all models that use the target data. ②. The data provider reviews the request of data contributor, and sends unlearning requests to corresponding model providers. ③. A malicious model provider executes dishonest unlearning in three ways. ④. The model provider returns the unlearned model to the data provider after unlearning for auditing. ⑤. The data provider responds with unlearning result to the data contributor.

it and submits an unlearning request to the server. Then the server will fulfill the unlearning request using the exact or approximate unlearning methods and return a copy of the model to the data provider for transparency and regulation purposes. Note that this does not infringe on the data privacy of the server because the data provider already has these data. Finally, the data provider responds to the data contributor with the unlearning outcome. In threat model of this paper, TruVRF is used to verify whether the model provider has honestly forgotten the specified data in accordance with contractual requirements, as agreed upon between the data provider and the model provider during data transactions. The model provider cannot intuitively make a fake model providing irrelevant tasks to replace the unlearned model. In our threat model, the data provider acts as the verifier, submitting and validating unlearning requests, while the model provider serves as the prover, executing these requests and delivering the correct unlearned model for verification. We assume no collusion between them, because they are business competitors.

Specifically, the threat model of machine unlearning is depicted in Figure 2. In MLaaS scenarios, there are three entities involved in machine unlearning services: the model provider (the server), the data provider, and the data contributor. We detail the capabilities and knowledge of the above entities as follows:

- **Data Contributor.** The data contributor is the owner of the model training data. Generally, the data contributor would sell the annotated data to the data provider, or complete the crowd-sourcing annotation task published by the data provider. Due to the privacy regulations (i.e., *the forgotten right*) [32], the data contributor has full control over the data he/she owns. In the context of machine unlearning, the data contributor is allowed to ask the model provider to revoke his/her data contribution to the ML model.
- **Data Provider.** The data provider is responsible for collecting data from various contributors and selling it to different model providers. Transparent privacy policies and usage terms should be clearly outlined to define data contributors' control over their personal data when the data provider obtains or publishes crowd-sourcing clickwork. The data provider has the ability to access both the original model M_o and the unlearned model M_u provided by the server for subsequent unlearning verification. This model access aligns with previous unlearning research [3], [7], [18], [45]. Additionally, the data provider possesses significant computational resources and auxiliary data (such as the data contributor's target and test data) to facilitate unlearning verification.
- **Model Provider (the server).** The server is the model trainer or model owner, with full control of the model. Additionally, the server is responsible for the unlearning operation whenever requested by the data provider. Typically, the server removes the target data by the exact or approximate unlearning method. Then the server would return a copy of the unlearned model to the data provider for transparency purposes. However, the server could be reluctant to honestly perform the unlearning due to incurred cost or model deterioration if the requested data are crucial, where the server deceives the data provider. The server may behave in the following dishonest behaviors:
  - Neglecting Server: The server simply ignores the unlearning request and does not perform any operation on the ML model. In this situation, data contributor's target data still remains in the origin model M_o, in which the class and data volume will be intact. This type of server violates the Unlearning Data Facticity and Unlearning Volume Integrity.
  - Lazy Server: The server executes the data provider's unlearning request partially, by forgetting the target data selectively. In this situation, the server will review the data provider's unlearning request, filter the data class and data volume that the server wants to keep and forget the remaining data he/she is not interested in. This type of server violates the Unlearning Volume Integrity.
  - Deceiving Server: The server executes the data provider's unlearning request, but he/she will distort it. Concretely, the server will choose the same class and data volume of other *irrelevant* data to replace the target data. In this situation, the server aims to retain the data of the contributor in the ML model and attempts to deceive the data provider by unlearning other irrelevant data. This type of server violates the Unlearning Data Facticity.

Above dishonest model providers fail to perform the requested data unlearning operations on the target data, which may lead to a series of privacy and security issues. Firstly, it violates the privacy of users to whom the target data belongs, breaching various privacy regulations [4], [32], [41]. Secondly, it increases the risk of model bias and unfairness, as the data intended for removal may contain sensitive or outdated information. If not properly deleted, such data can continue to influence model decisions, resulting in biased or unfair outcomes. These threats can erode user trust in data collection platforms and expose the platforms to compliance risks.

The proposed `TruVRF` is to enable the data provider to verify the data facticity and volume integrity of the unlearning performed by the server, that is, **whether the server has forgotten the target data** (data facticity) and **whether the server has forgotten the enough data** (volume integrity). Data providers are required to offer unlearning verification services to data contributors to comply with existing privacy regulations and laws. Motivated by the opportunity to enhance their reputation, attract a wider range of data contributors, address privacy concerns, and promote transparency in the Data-as-a-Service ecosystem, data providers are obligated to provide these services. To this end, `TruVRF` consists of three degrees of verification ability devised upon the key observation of *model sensitivity*.

B. Model Sensitivity

The trained ML model learns the feature of training data, so as to classify the test data, which is the inherent characteristic of ML. Additionally, we found that the ML model could memorize the data-distribution characteristic of a training dataset, and inadvertently reflects the distribution in the form of gradient changes. The gradient value is related to the class and volume of training data. More precisely, if the sample number of a class in a training dataset is small, the model is unlikely to learn to generalize to that class. Thus, when the model trains on such dataset, it will exhibit a greater gradient effect to change the weights of the corresponding neurons to minimize the expected loss of the model regarding such class. That is, the gradient change is negatively correlated with the number of class samples in the training dataset. We define the gradient change in training as the *model sensitivity*. We note such model sensitivity was recently exploited to profile user preference in the federated learning that leaks the local user's private information [57]. In contrast, we turn this model sensitivity as an asset to address the open challenge of delicately verifying machine unlearning conducted by a dishonest centralized server.

In the context of honest machine unlearning, it retrains the model by removing the requested data from the training set. The model sensitivity is expected to display a notable discrepancy between the origin model M_o and the unlearned model M_u. Attributing to the decrease of training data, the model sensitivity of the requested class will increase, which serves as the key observation for constructing `TruVRF`. To this end, we quantify the model sensitivity by utilizing the sum of absolute values of the gradient changes before and after the retraining model, which is expressed as:

$$MS = \sum \left| \frac{\delta}{\delta \theta'} L(\theta) \right| = \sum \left| \left(\theta - \theta'\right) \cdot \frac{1}{\alpha} \right|, \tag{2}$$

where θ represents the model parameter of the origin model, and θ' represents the model parameter of the unlearned model. $L(\cdot)$ represents the loss function, and α represents the learning rate. The process is detailed in Algorithm 1. In this paper, auxiliary dataset refers to the data used to retrain the ML model by the data provider to extract *MS*, which can be data contributor's target data or test data. Notably, also as a data curator and aggregator, the data provider has plenty of data.

Then, upon the model sensitivity, we construct three `TruVRF` metrics serving as verification toolkits against all three types of dishonest servers (Neglecting Server, Lazy Server, and Deceiving Server) to verify the unlearned model M_u returned by them. Figure 3 illustrates the structure of `TruVRF`, and the detailed Metric design and corresponding verification procedure are presented as follows.

C. Unlearning-Metric-I: Unlearned Class Verification

This metric is designed to verify the unlearning result of Class Granularity, and decides which class(es) the server has unlearned. For example, a data contributor wants to withdraw all/some of his/her own face images from a facial recognition model. After receiving the copy of the unlearned model M_u and the origin model M_o from the server, the data provider starts the verification by continuing training the M_o and M_u upon his/her own testing data. Then the data provider collects the gradient changes during the training and computes the model sensitivity for each target class. We call this process model sensitivity extraction; see Algorithm 1. All model sensitivity extraction in the following study utilizes this Algorithm 1, and hereinafter referred to as $\mathsf{Extract}\ (\cdot)$. The data provider can judge whether the server has forgotten the target class by matching the sensitivity difference of the model for each class in two models. The specific process is shown in Figure 4.

Algorithm 1 Extracting Model Sensitivity

```
Input: ML model $\theta$, auxiliary dataset $D_{\mathrm{aux_c}}$ of class
       $c$, learning rate $\alpha$, training epoch $e$
Output: Model sensitivity $MS$ of class $c$
Set $MS = 0$;
Data provider trains the model $\theta' = \theta.\mathsf{train}(D_{\mathrm{aux}_c}, \alpha)$
foreach $e \in \{1,2,\ldots,epoch\}$ do
    foreach $(x, y)$ in $(X_{\mathrm{train}}, Y_{\mathrm{train}})$ do
        $y_{\mathrm{pred}} \leftarrow \mathrm{Predict}(\theta, x)$;
        $\theta' \leftarrow \theta - \alpha \cdot \nabla \mathrm{Loss}(y_{\mathrm{pred}}, y)$
    end
end
foreach neuron $i$ in $\theta'$ do
    Compute $MS_c^i = \left|\left(\theta - \theta'\right) \cdot \frac{1}{\alpha}\right| = \left|\frac{\delta}{\delta\theta_i} L(\theta)\right|$;
    $MS_c += MS_c^i$;
end
```

If the server has honestly removed the target data and updates the model, the model sensitivity of target class in the unlearned model M_u increases as detailed in Section V-D. Because the sample number of the corresponding class becomes smaller, the model would exhibit a significant gradient change for this category. Furthermore, the more forgotten, the more obvious the change discrepancy between the origin model M_o and the unlearned model M_u will be. This metric is suitable for verifying Neglecting Server. In this case, the origin model M_o and the unlearned model M_u differ less in the sensitivity of the model, where the verification of the unlearned class could perform well. In other words, Neglecting Server does not remove the target data at all, and training dataset of twin models before and after unlearning has remained identical.

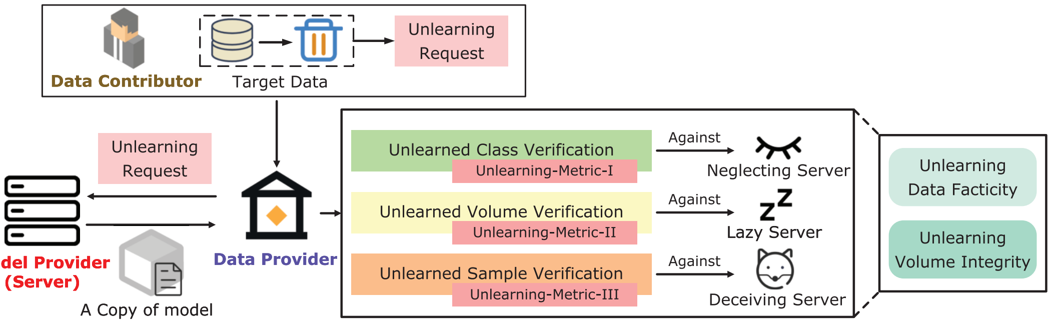


Fig. 3. TruVRF overview.

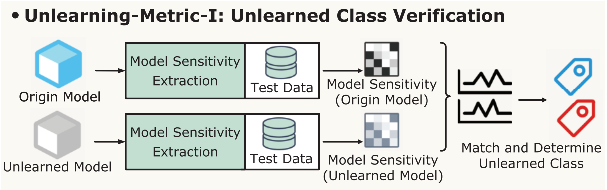

• Unlearning-Metric-I: Unlearned Class Verification
Origin Model
Model Sensitivity Extraction
Test Data
Model Sensitivity (Origin Model)
Unlearned Model
Model Sensitivity Extraction
Test Data
Model Sensitivity (Unlearned Model)
Match and Determine Unlearned Class

Fig. 4. Process of Unlearned Class Verification.

D. Unlearning-Metric-II: Unlearned Volume Verification

In complex scenarios like Lazy or Deceiving Server, Unlearning-Metric-I may not suffice for the removal of only a portion of a class. Servers might unlearn target class data but not the exact volume, bypassing Class Granularity. Hence, statistical methods are needed to verify the unlearned sample count in Volume Granularity.

In order to verify whether the server unlearns the requested amount of target data, we design the unlearned volume verification. Firstly, the data provider extracts model sensitivity of target data against both origin and unlearned models according to Algorithm 1. Then the data provider computes the model sensitivity difference DS, which represents the change of model sensitivity caused by the decrease in the number of samples in this category:

$$DS = MS_u - MS_o. \tag{3}$$

Metric-II builds on the relationship between the difference in model sensitivity, DS, and the volume of requested forgotten data, to decide the number of samples that the server has unlearned. To this end, we define an unlearning measurement, which is to transform from model sensitivity difference to target data volume. As the name indicates, the data provider is now able to verify unearned volumes with this measurement.

How to get the unlearning measurement? Given a converged ML model, model sensitivity of one class is with upper and lower bounds. Within this interval, the model sensitivity decreases approximately linearly with increasing sample size. Generally, we use shadow models to obtain the unlearning measurement, which process is depicted in Figure 5. We now elaborate on this process.

The data provider trains n shadow models based on target data. Shadow model training can cost additional computation overhead in Unlearning-Metric-II. We note that this is tolerable once privacy is a top priority for the data provider. Before training, the target class of the unlearned data is divided into different slices according to the volume of data. Taking the CIFAR-10 dataset as an example, when the data contributor's target class is the "dog", the volume of "dog" of the requested data is divided incrementally as a shadow dataset, such as $D^{\text{dog}}_{\text{shadow}_1} = 100$, $D^{\text{dog}}_{\text{shadow}_2} = 200$,., $D^{\text{dog}}_{\text{shadow}_n} = 1000$. The rest of the categories remain the same amount of data in all shadow models and do not need to have any intersection with the training set categories in the origin model M_o. The data provider then extracts the model sensitivity of "dog" in these shadow models and computes the model sensitivity difference:

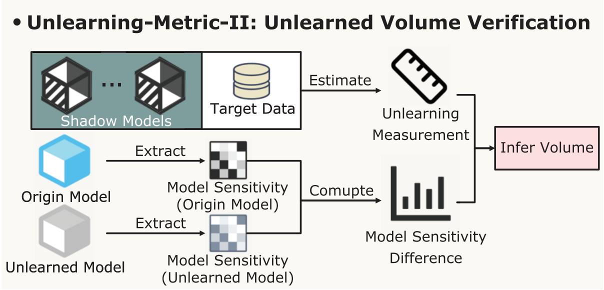

• Unlearning-Metric-II: Unlearned Volume Verification
Shadow Models
Target Data
Estimate
Unlearning Measurement
Infer Volume
Origin Model
Extract
Model Sensitivity (Origin Model)
Comupte
Extract
Unlearned Model
Model Sensitivity (Unlearned Model)
Model Sensitivity Difference

Fig. 5. Process of Unlearned Volume Verification.

$$DS'_n = MS_{\text{shadow}_{n+1}} - MS_{\text{shadow}_n}. \tag{4}$$

In Metric-II, we define average value of model sensitivity differences in shadow models as unlearning measurement:

$$UM_{\text{batch}} = \frac{\sum_{i=1}^{n} DS'_{2i-1}}{n}. \tag{5}$$

To this end, the data provider is able to use UM_{batch} to verify the requested data volume forgotten by the server according to:

$$Target\ data\ volume = \left\lceil \frac{DS}{UM_{\text{batch}}} \right\rceil \times batch\ volume. \tag{6}$$

.

E. Unlearning-Metric-III: Unlearned Sample Verification

Unlearning-Metric-I and Unlearning-Metric-II allow data provider to verify the data facticity and volume integrity

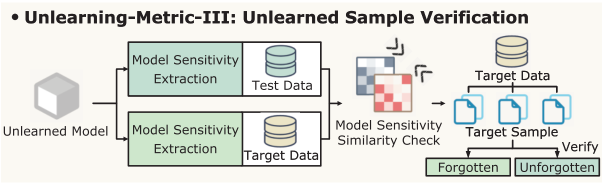

• Unlearning-Metric-III: Unlearned Sample Verification
Unlearned Model
Model Sensitivity Extraction
Test Data
Model Sensitivity Extraction
Target Data
Model Sensitivity Similarity Check
Target Data
Target Sample
Verify
Forgotten
Unforgotten

Fig. 6. Process of Unlearned Sample Verification.

of machine unlearning in terms of target class and volume. However, in the case of the Deceiving Server, the unlearned class and volume can be the same as the target data, which obscures Unlearning-Metric-I and -II. For example, when data contributor A requests to unlearn 100 samples in the class "dog", the Deceiving Server responds through forgetting another non-overlapped 100 samples in the class "dog" instead of 100 samples owned by data contributor A. In this case, Unlearning-Metric-I and -II could hardly identify the dishonest behavior of the Deceiving Server, because the server could bypass the verifications on the granularities of class and volume. Hence, we propose Unlearned Sample Verification to deal with such a situation under Sample Granularity, to verify the target sample non-existence in the unlearned model M_u.

Unlearned Sample Verification relies on model sensitivity, where the data provider calculates the model sensitivity of the original model M_o and the unlearned model M_u using target data. Our key observation indicates that the model sensitivity of forgotten data, as extracted from the target data, is significantly higher than that of unforgotten data. When retraining on unforgotten samples, the model's sensitivity remains stable due to its familiarity with the samples and minimal gradient changes during back-propagation. Forgotten samples introduce new features, causing significant gradient changes that drive the model towards convergence and enhance the neural network's learning capacity. As a result, forgotten samples experience a more pronounced shift in model sensitivity than non-forgotten samples. Leveraging this understanding, we introduce Unlearning-Metric-III, depicted in Figure 6.

Specifically, the data provider extracts the model sensitivity of unlearned model $MS_{\text{u\_test}}$ based on test data and $MS_{\text{u\_tar}}$ based on target data. By comparing $MS_{\text{u\_test}}$ and $MS_{\text{u\_tar}}$, if there is an obvious discrepancy between them, then there is a high probability that the server has spoofed the data provider and data contributor. Otherwise, the server has unlearned the target data honestly when $MS_{\text{u\_test}}$ and $MS_{\text{u\_tar}}$ are similar. The reason is that the existence of target sample(s) in unlearned model M_u would mitigate gradient changes during the model sensitivity extraction under target data. It could cause a large gap in the model sensitivity by the test data. Hence, when the Deceiving Server unlearns the same volume of *irrelevant* data as the target data, it leaves traceable model sensitivity. Detailed analyses are in Section V-D.

1) Rationale of Unlearning-Metric-III: We extract the model sensitivity of the Honest Server and the Deceiving Server on the target data and testing data respectively. We found that, the model sensitivity extracted on target data is significantly lower than the one extracted on testing data when target data remains in the unlearned model. This is the foundation of determining whether the target data still exists within the unlearned model in Unlearned Sample Verification. Details are deferred to Section V-D. Note that, Unlearning Sample Verification is distinct from the membership inference. Although membership inference could indirectly determine the data record existence, it is not delicately suitable for unlearning verification. Moreover, its accuracy is lower than TruVRF, since membership inference is heavily overfitting-dependent, and we have evaluated and compared with it in Section V-A.

With the above three TruVRF Unlearning-Metrics or tookkits we propose, the data provider can effectively verify the data facticity and volume integrity of machine unlearning to identify the three types of dishonest servers (Neglecting Server, Lazy Server, and Deceiving Server), and safeguard the privacy and property of machine unlearning users.

IV. Experiments

In this Section, we experimentally evaluate TruVRF to validate its effectiveness. We also conduct empirical experiments of TruVRF to affirm its immediate applicability under state-of-the-art machine unlearning frameworks.

A. Experimental Setup

1) Dataset: We run experiments on three common benchmark datasets that have also been used in machine unlearning studies:

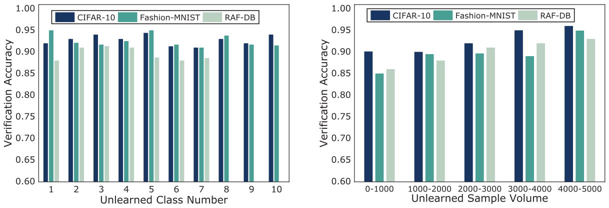
- **CIFAR-10** is an image dataset for the recognition of universal objects [26]. There are 10 categories of RGB color images with size of 32×32×3. It consists of 50,000 training and 10,000 testing samples, respectively.
- **Fashion-MNIST** is a collection of Zalando's fashion objects, having a training set of 60,000 examples and a test set of 10,000 examples [52]. Each sample is a 28×28 grayscale image, associated with a label from 10 classes.
- **RAF-DB** is a large-scale facial expression database with around 30K great-diverse real-world facial images, called Real-world Affective Faces Database [27]. There are 7 categories of RGB color images with size of 48 × 48.

2) Model Structure: As for the CIFAR-10, the model architecture has four convolution blocks: each block has one convolutional layer and one max pooling layer, followed by one fully connected layer. As for the Fashion-MNIST, the model architecture is two convolutional layers and one max pooling layer, followed by two fully connected layers. As for the RAF-DB, the model architecture has four convolution blocks identical to CIFAR-10, with the last layer followed by two fully connected layers.

3) Machine Unlearning Method: In our experiments, we first used machine unlearning with retraining from scratch as the validation baseline of our proposed metrics. Because both exact unlearning and approximate unlearning are extended based on retraining [45]. In addition, to demonstrate the generalisability of our verification approach, we further evaluate TruVRF under existing SOTA machine unlearning frameworks: the exact unlearning framework SISA [3] and the approximate unlearning framework Amnesiac Machine Learning [18], respectively.

TABLE II
VERIFICATION EFFICIENCY OF UNLEARNING-METRIC-I

Task	Trial	Vef. Acc.	Overall
CIFAR-10	20	0.94	0.93
	50	0.92	
	100	0.93	
	200	0.92	
Fashion-MNIST	20	0.95	0.93
	50	0.93	
	100	0.93	
	200	0.92	
RAF-DB	20	0.90	0.89
	50	0.88	
	100	0.91	
	200	0.90	



(a) Unlearned Class Number (b) Unlearned Sample Volume

Fig. 7. Unlearning-Metric-I's performance under different class numbers and sample volumes.

4) Machine Configuration: Experiments run on a computer with the following configuration: Intel Core i9 processor with ten CPU cores running at 3.70 GHz and with a 32 GB main memory, and a GPU card of NVIDIA GeForce RTX 3090.

B. Verification on Unlearning-Metric-I

Here, we carry out experiments to evaluate the performance of the Unlearning-Metric-I against the Neglecting Server. The Neglecting Server would ignore the data provider's unlearning request. We compare the model sensitivity of target class between the origin model M_o and the unlearned model M_u. If the difference of these twin models is lower than a threshold of, e.g., 1%. TruVRF judges that the model sensitivity remains unchanged, which means that the server does not honestly unlearn this target class.

1) Metric-I Verification Accuracy: We first conduct evaluations on baseline honest unlearning and dishonest unlearning by retraining from scratch to verify the unlearning effectiveness. We have repeatedly run 20, 50, 100 and 200 experiments to determine whether the target class is unlearned in unlearned model M_u, and reported verification accuracy of Unlearning-Metric-I as shown in Table II. The experimental results indicate that Unlearning-Metric-I can reliably judge the data class forgotten by the server with high accuracy, so it has the ability to verify the Neglecting Server.

2) Impact of Class Number and Sample Volume: Here, we show the verification ability of Unlearning-Metric-I towards different unlearning requests. We select 1-10 unlearned class(es) and 0-5000 unlearned samples given a class (1)-7 classes for RAF-DB, since it has 7 classes) which represents the variety of unlearning requests. Under these settings, we test the accuracy of Unlearning-Metric-I, and the results are shown in Figure 7.

TABLE III
VERIFICATION DEVIATION OF UNLEARNING-METRIC-II

Task	Unlearned Volume	Inferred Volume	Deviation	Overall
CIFAR-10	500	470	6%	4.8%
	1000	942	5.8%	
	1500	1421	5.3%	
	2500	2443	2.3%	
Fashion -MNIST	500	463	7.4%	6.8%
	1000	935	6.5%	
	1500	1419	5.4%	
	2500	2388	4.5%	
RAF-DB	500	452	9.7%	8.2%
	1000	912	8.8%	
	1500	1388	7.5%	
	2500	2330	6.8%	

As can be seen from Figure 7 (a) and (b), verification accuracy of Unlearning-Metric-I can maintain more than 90% for CIFAR-10 and Fashion-MNIST, 85% for RAF-DB under different unlearned classes and sample volumes. In terms of unlearned classes, Unlearning-Metric-I accuracy is insensitive to it. In terms of sample size, the accuracy of the verification increases slightly as the number of unlearned samples increases, which is because the change in model sensitivity is related to the amount of data within the unlearned model M_u, i.e. the remaining amount of data in a given class is small. When the amount of data in the target class is smaller, the more obvious gradient changes are produced when extracting model sensitivities.

C. Verification on Unlearning-Metric-II

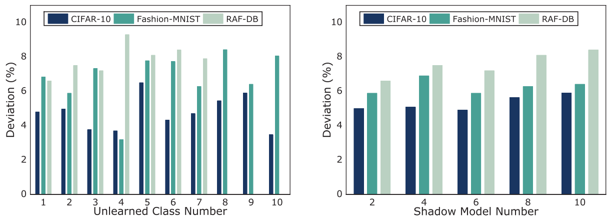
Here, we explore unlearning verification in the Volume Granularity, and infer the amount of target data that the server forgets. TruVRF can be used to counteract the behavior of Lazy Server that unlearns only partial target data.

1) Inference of Unlearned Sample Volume: We first conduct experiments on inferring the number of unlearned samples under the baseline machine unlearning and summarize results in Table III. We evaluate Unlearning-Metric-II under the scenarios that the unlearned sample volume is 500, 1000, 1500, and 2500 respectively for CIFAR-10 and Fashion-MNIST, and compute that:

$$Deviation = \frac{|Unlearned\ Volume - Verified\ Volume|}{Unlearned\ Volume}, \quad (7)$$

which represents the difference degree between the factual requested unlearned volume and the estimated verified volume. The closer the deviation is to 0, the better the verification effect of Unlearning-Metric-II is. We have run each task for 50 trials. From Table III, the data volume extrapolated from Unlearning-Metric-II is highly reliable, differing from the factual unlearned volume by only 4.8% for CIFAR-10, 6.8% for Fashion-MNIST and 8.2% for RAF-DB on average. Additionally, the verification effect improves with the increase of unlearned volume, due to the more pronounced changes in model sensitivity caused by large data volume changes.

2) Impact of Class Number and Shadow Model Number: We evaluate the Verification Deviation under different unlearned class numbers and shadow model numbers. We set 1-10 (1-7 for RAF-DB) unlearned class(es) (in increments of



(a) Unlearned Class Number (b) Shadow Model Number

Fig. 8. Unlearning-Metric-II's deviation under different unlearned class numbers and shadow models.

TABLE IV

Verification Efficiency of Unlearning-Metric-III

Task	Trial	Vef. Acc.	Overall
CIFAR-10	20	0.93	0.90
	50	0.89	
	100	0.87	
	200	0.90	
Fashion-MNIST	20	0.94	0.92
	50	0.91	
	100	0.91	
	200	0.92	
RAF-DB	20	0.90	0.91
	50	0.93	
	100	0.92	
	200	0.91	

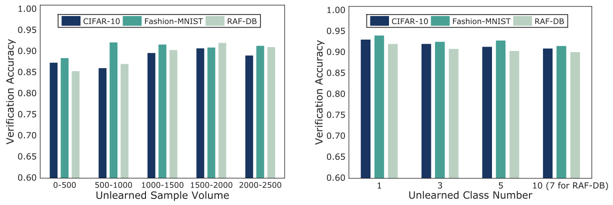
1) and 2-10 (2-7 for RAF-DB) shadow models (in increments of 2) in the experiments respectively. With this setting, we conduct multiple experiments and report the deviation of each case in Figure 8. The experimental results demonstrate that the efficiency of Unlearning-Metric-II is invariant to the number of classes and shadow models. The deviation range is small, falling between 3%-9% with diverse settings.

D. Verification on Unlearning-Metric-III

Here, we evaluated the Unlearning-Metric-III performance, specifically the existence of target sample(s) in the unlearned model M_u. This can be utilized to identify the behavior of Deceiving Server that secretly swaps the target sample(s).

1) Metric-III Verification Accuracy: The purpose of Unlearning-Metric-III is to verify whether the samples that the server has unlearned are the target samples in the unlearning request. Then we evaluate the verification efficiency of Unlearning-Metric-III under the scenario with honest unlearning and dishonest unlearning (Deceiving Server). We conducted the experiments for 20, 50, 100 and 200 different trial runs. Table IV illustrates the verification accuracy and overall accuracy. Unlearning-Metric-III could maintain an accuracy of 90% on average, which provides the data contributor with evidence to verify the existence of his/her target data.

2) Impact of Sample Volume and Class Number: Figure 9 (a) depicts the verification accuracy under different amounts of target data, which is from 0-2500 for CIFAR-10 and Fashion-MNIST (the server needs the rest 2500 to replace the target data in unlearning request). The experimental results depict that the verification efficiency is minimally affected by the number of unlearned samples. Meanwhile, we also detail the verification accuracy when the class number is changed in Figure 9 (b). We select 1, 3, 5, 10 (7 for RAF-DB dataset, because it has 7 classes) unlearned classes respectively, and the result of Unlearning-Metric-III can maintain more than 90% verification accuracy to identify the Deceiving Server.



(a) Unlearned Sample Volume (b) Unlearned Class Number

Fig. 9. Unlearning-Metric-III's performance under different sample volumes and class numbers.

E. Unlearning Frameworks Verification

The machine unlearning method we have evaluated so far is the baseline machine unlearning framework that trains the entire model from scratch. Here we evaluate TruVRF on existing SOTA unlearning frameworks specifically, SISA [3] as the representative of exact unlearning, and Amnesiac Unlearning [18] as the representative of approximate unlearning. We aim to validate the immediate applicability of TruVRF for available or potentially emerging unlearning frameworks. For each unlearning framework, we repeated 10 trials to evaluate three unlearning metrics.

1) Exact Unlearning Setting: The core of SISA [3] is to split the training data into k disjoint shards. The server then trains k sub-models respectively and derives the final output by collaborative predictions of all sub-models through aggregating algorithms—an ensemble strategy. When receiving the unlearning request, the server localizes the shards where the target data exist in, and only retrain sub-models corresponding to these shards. Such an approach substantially reduces computational overhead than baseline retraining, thus accelerating the unlearning process. In the experiments, we set $k = 5$ and assign 500 data samples to each shard. We randomly select 0-400 data samples in a shard to unlearn for each trial and evaluate the unlearning accuracy using the proposed Unlearning-Metric-I,-II, and -III.

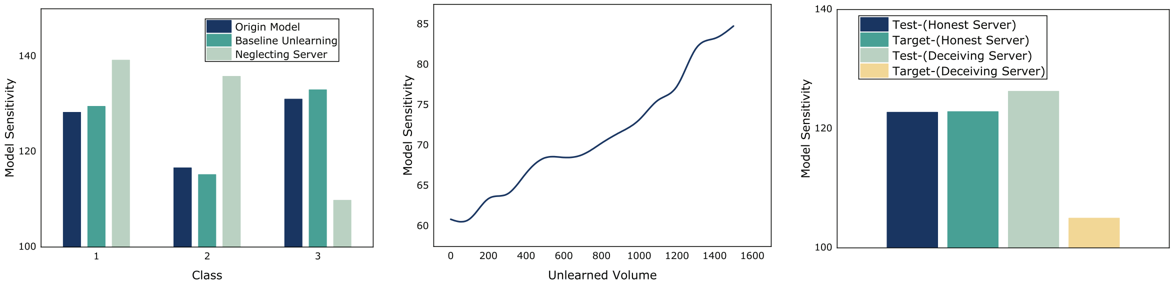
2) Approximate Unlearning Setting: We also assess TruVRF under approximate unlearning, where the objective is to obfuscate the model's understanding of target data by assigning randomly chosen incorrect labels to the target data and then retraining the model on this modified dataset [18]. This process aims to degrade the model's generalization of the target data. In this experiment, we randomly substitute the labels of the target data with different labels and proceed to update the model using this relabeled dataset. The data volume in the original model M_o is 1500 per class, while the data volume in the unlearning request ranges from 0 to 1500 for a randomly selected class.

3) Results: Table V details the verification performance (Unlearning-Metric-I, -II, -III) under exact and approximate unlearning, respectively, on the CIFAR-10, Fashion-MNIST and RAF-DB. The experimental results of Unlearning-Metric-I

TABLE V

VERIFICATION PERFORMANCE OF SISA AND AMNESIAC UNLEARNING FRAMEWORKS

Unlearning Framework	Task	Unlearning-Metric-I (Verification Accuracy)	Unlearning-Metric-II (Verification Deviation)	Unlearning-Metric-III (Verification Accuracy)
SISA (Exact Unlearning)	CIFAR-10	0.92	6.1%	0.85
	Fashion-MNIST	0.90	9.1%	0.84
	RAF-DB	0.89	9.6%	0.86
Amnesiac Unlearning (Approximate Unlearning)	CIFAR-10	0.94	10.1%	0.90
	Fashion-MNIST	0.92	10.7%	0.88
	RAF-DB	0.88	12.1%	0.90



(a) Unlearning-Metric-I (b) Unlearning-Metric-II (c) Unlearning-Metric-III

Fig. 10. Verification performance of TruVRF under multi-category unlearning.

under both unlearning frameworks validate that the verification performance can derive a similar effect to the baseline retraining, i.e., the Unlearned Class Verification. Both unlearning frameworks can achieve more than 90% verification accuracy. The Unlearned Volume Verification and the Unlearned Sample Verification slightly degrade compared to the baseline unlearning, relatively notable for the approximate unlearning. We analyze the reason is that unlearning frameworks may prioritize reducing computational overhead over maintaining the accuracy of the unlearned model. For example, an increase in the number of shards can lead to accuracy degradation [3], potentially suppressing normal gradient changes seen in baseline unlearning. In the case of approximate unlearning, parameters of the trained model are directly modified to approximate the model without the data points to be unlearned from the beginning [45], resulting in smaller changes in data volume. Despite this, approaches like Amnesiac Unlearning that mitigate data influence, such as relabeling, enhance the model sensitivity of unlearned classes. This improved sensitivity can be utilized in Unlearning-Metric-II to estimate the volume of perturbed data. Compared to baseline and exact unlearning [3], there is a verification deviation in approximate unlearning. Despite a slight decrease in performance compared to state-of-the-art unlearning frameworks due to potential accuracy trade-offs, TruVRF's overall performance remains satisfactory.

F. TruVRF in Multi-Category Unlearning

Here, we evaluate the effectiveness of unlearning multiple classes. Considering scenarios where multiple unlearning requests may occur simultaneously, we utilized the CIFAR100 dataset, which contains a larger number of categories (100 classes), and tested a scenario involving the simultaneous unlearning of three categories, with 150 data removed from each category. The experimental results are shown in Figure 10. It can be observed that Unlearning-Metric-I remains effective in distinguishing dishonest unlearning behaviors by Neglecting Server in Figure 10(a). Regarding Unlearning-Metric-II, TruVRF could fit unlearning measurement, as illustrated in Figure 10(b), with an inferred volume of 142 and a deviation of 5.3%. As for Unlearning-Metric-III, Deceiving Server can be effectively identified by TruVRF in Figure 10(c), as the model sensitivity on the target dataset significantly diverges from that on the test set. Therefore, the above experiments demonstrate that TruVRF is applicable to datasets with more categories and multi-category unlearning scenarios. However, it is important to note that when the number of unlearning requests becomes excessively large, the model's performance may degrade or even collapse, in which case our method effectiveness may be adversely influenced.

V. FURTHER EVALUATION AND DISCUSSION

A. Unlearning-Metric-III Vs. Membership Inference Attack

Unlearning-Metric-III aims to determine the existence of the target data in the unlearned model M_u, ensuring that the data ownership forgotten by the server is distinguishable. It could expose the displacement behavior that retains the target data in unlearned model M_u by the dishonest server. Membership inference can serve as an indirect method to non-invasively verify the sample-granularity unlearning to some extent. Therefore, we compare TruVRF with it in terms of sample-granularity verification, affirming that TruVRF is more competent in explicitly verifying the unlearning task.

We utilized the membership inference attack against the machine unlearning [7] to compare with the Unlearning-Metric-III. In this experiment, we set 1 shadow origin model and 20 shadow unlearned models with 30000 samples for each model. Then, we construct the membership feature by

TABLE VI
COMPARISON BETWEEN MEMBERSHIP INFERENCE ATTACK AND UNLEARNING-METRIC-III

	Membership Inference Attack		**TruVRF**
	Overfitting	**Non-Overfitting**	
Train Acc.	0.954	0.942	-
Test Acc.	0.477	0.919	-
Verf. Acc.	0.881	0.519	0.92
Overhead	8109.92s		52.08s

Euclidean distance and train an attack model through the logistic regression, which is aligned with [7]. We evaluated the verification accuracy and computation overhead under overfitting and non-overfitting, as shown in Table VI. The results report that although the membership inference could verify the existence of an unlearned sample in the unlearned model, it is heavily overfitting-dependent—this is actually a notable inherent downside of membership inference [22]. Additionally, the training time for multiple shadow models also hampers verification efficiency. In the experiment, the computation overhead is 8109.92s, which includes training 1 shadow origin model for 509.96s, training 20 shadow unlearned models with an average of 322.75s per model, constructing features for 1142.41s, and training attack models for 2.49s. This overhead of the membership-inference-based method is much longer compared to the model sensitivity extraction time (52.08s) in TruVRF for all requested samples (1500 samples in this trial).

B. Overhead

Here, we evaluated the computational cost of TruVRF for face recognition tasks using the VGG model. We record the computational costs associated with each dominant operation within the three unlearning metrics.

- For Unlearning-Metric-I, the data provider needs to perform two model sensitivity extraction operations for one unlearning verification task, with an average time consumption of 90.83 seconds.
- For Unlearning-Metric-II, the computational cost mainly involves shadow model training and sensitivity extraction. We found that the training time for each shadow model is 716.79 seconds, and sensitivity extraction takes 55.47 seconds. It's worth noting that shadow model training is the dominant factor in computational cost, as it involves extracting the correlation between sample size and model sensitivity. However, in real-world applications, shadow model training is a one-time process and doesn't need to be repeated for each unlearning verification. Additionally, the training can be conducted offline before the verification task, reducing user waiting time.
- For Unlearning-Metric-III, the computational cost relies on two model sensitivity extractions, with an average computational cost of 45.33 seconds per extraction. Specifically, we conducted a detailed analysis and comparison of the performance between Unlearning-Metric-III and membership inference, as shown in Table VI.

Therefore, TruVRF meets the computational cost requirements for real-world applications, as it doesn't impose a significant burden on the data provider and avoids prolonged waiting times for data contributors.

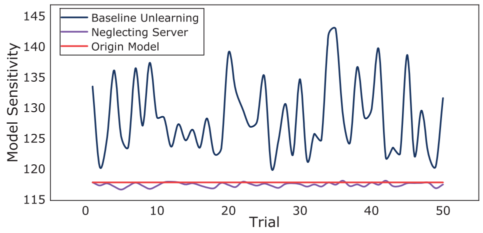


Fig. 11. Key observation of Unlearning-Metric-I. The model sensitivity of honest unlearning and dishonest unlearning with Neglecting Server exhibit a significant discrepancy.

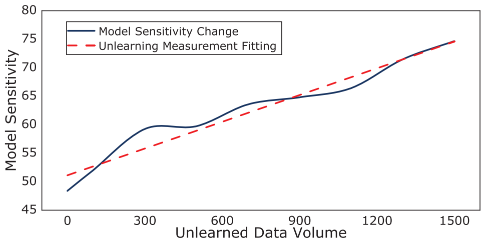


Fig. 12. Key observation of Unlearning-Metric-II. The model sensitivity changes approximately linearly with the unlearned data volume representing the variation value of model sensitivity per unit of unlearned data volume, which is denoted as the Unlearning Measurement. We utilize it to infer the scale of unlearned data by the server in Unlearning-Metric-II.

C. Non-IID Model

To demonstrate the effectiveness of TruVRF given a Non-IID dataset, we conducted experiments of TruVRF on the GTSRB dataset [40], with 39,209 training images and 12,630 testing images of 43 different classes—samples per each class is imbalanced for this dataset. We randomly select 10 classes, and unlearn all their samples. Then we evaluate TruVRF, and the verification accuracy of Unlearning-Metric-I is 0.91, the verification deviation of Unlearning-Metric-II is 5.8% per category on average, and the verification accuracy of Unlearning-Metric-III is 0.88. The experimental results demonstrate that TruVRF is independent of the data distribution.

D. The Rationale of TruVRF

Here, we conduct evaluations on CIFAR-10 to further explain the TruVRF rationale.

We elaborate on the key observation supporting Unlearning-Metric-I in Figure 11. In this experiment, we set 5000 samples per class in origin model M_o, and issue 50 unlearning requests respectively. The target class and sample number are random. For comparison, we employ a pair of honest unlearning and dishonest unlearning with Neglecting Server per unlearning request, and extract model sensitivity of unlearned model M_u. The figure clearly shows that Unlearning-Metric-I can verify whether the server unlearns the target class attributing to the obvious model sensitivity discrepancy, thus distinguishing the honest server from Neglecting Server.

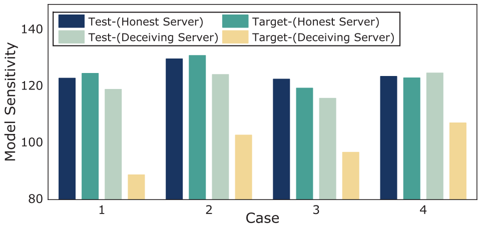


Fig. 13. Key observation of Unlearning-Metric-III. The four types of model sensitivities: Test-(Honest Server) and Target-(Honest Server) represent the model sensitivity extracted by test data and target data in Honest Server scenario, and Test-(Deceiving Server) and Target-(Deceiving Server) represent the model sensitivity extracted by test data and target data in Deceiving Server scenario.

TABLE VII

Robustness Evaluation of Model Sensitivity

Operation	**Model Sensitivity**		
Baseline	127.17	126	114.8
Perturbation	132.15	131.5	118.4
Model Pruning	131.9	129.9	118.9

To interpret Metric-II and Unlearning Measurement, we depict the model sensitivity changes of target data as unlearning data volume increases in a model in Figure 12. Meanwhile, we fit such changes into a curve to demonstrate the Unlearning Measurement, that is, how much data does a fixed amount of model sensitivity change correspond to. In this experiment, we increase unlearning requested samples from 0 to 1500. As Figure 12 validates, the increment of unlearning data volume leads to a linear increase of model sensitivity of the corresponding class.

To interpret the rationale behind Unlearning-Metric-III, we show the difference in model sensitivity under Honest and Deceiving Server. We set the Honest Server and the Deceiving Server as the control group. We extract the model sensitivity of unlearned model M_u by using test data and target data. In experimental setting, the size of data contributor target class is 5000, the unlearning rate is selected randomly from 4% to 20%, and we perform four cases. Figure 13 depicts the comparison of model sensitivity. For the same unlearned model M_u, the model sensitivity extracted by test data (i.e., Test-(Deceiving Server)) and target data (i.e., Target-(Deceiving Server)) is almost the same, when the server honestly executes unlearning. However, if the server unlearns the irrelevant data to replace the target data, Target-(Deceiving Server) will be significantly reduced. Because the information of target data is still preserved in the unlearned model M_u, they do not cause greater gradient changes to the model than the test data.

E. Robustness of Model Sensitivity

Due to the possibility that dishonest servers may modify submitted models for auditing, thereby affecting the model sensitivity of TruVRF, we evaluated the robustness of model sensitivity under various modification strategies (Perturbation [1], Model pruning [47]) on CIFAR10. Considering practical scenarios, we preserved the model's utility by ensuring that the performance degradation did not exceed 5% when implementing these modifications. The experimental results are presented in Table VII. From the results, it is evident that these model modification methods do not significantly impact model sensitivity, with changes of only 3.9% (Perturbation) and 3.7% (Model Pruning). In fact, model sensitivity is an inherent property of the model, reflecting its sensitivity to samples of different categories. Therefore, model sensitivity is stable and robust as long as the model remains usable.

VI. Conclusion

This work is an initiative towards the machine unlearning verification framework entailed with triple degrees of granularity (class-, volume-, sample-level) to verify the data facticity and volume integrity of machine unlearning, namely TruVRF. We identified three types of dishonest servers (Neglecting Server, Lazy Server, Deceiving Server) and proposed three unlearning metrics to counter them. Extensive experiments have affirmed that the TruVRF is effective under various verification scenarios and significantly, is immediately applicable for existing SOTA unlearning frameworks SISA [3] as a representative of exact unlearning, and Amnesiac Unlearning [18] as a representative of approximate unlearning. We have further demonstrated the impact of TruVRF in real-world applications, particularly in face recognition systems. We hope TruVRF can provide valuable support for the effective deployment and implementation of the right to be forgotten. By bridging the gap between theoretical research and practical applications, our work aims to empower individuals with greater control over their data while fostering trust and accountability in machine learning systems.

References

[1] M. Abadi et al., "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 308–318.

[2] T. Baumhauer, P. Schöttle, and M. Zeppelzauer, "Machine unlearning: Linear filtration for logit-based classifiers," *Mach. Learn.*, vol. 111, no. 9, pp. 3203–3226, Sep. 2022.

[3] L. Bourtoule et al., "Machine unlearning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2021, pp. 141–159.

[4] Canada.(2019). *Personal Information Protection and Electronic Documents Act*. [Online]. Available: https://laws-lois.justice.gc.ca/ENG/ACTS/P-8.6/index.htm

[5] Y. Cao and J. Yang, "Towards making systems forget with machine unlearning," in *Proc. IEEE Symp. Security Privacy (SP)*, May 2015, pp. 463–480.

[6] Y. Cao, A. F. Yu, A. Aday, E. Stahl, J. Merwine, and J. Yang, "Efficient repair of polluted machine learning systems via causal unlearning," in *Proc. Asia Conf. Comput. Commun. Secur.*, May 2018, pp. 735–747.

[7] M. Chen, Z. Zhang, T. Wang, M. Backes, M. Humbert, and Y. Zhang, "When machine unlearning jeopardizes privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2021, pp. 896–911.

[8] M. Chen, Z. Zhang, T. Wang, M. Backes, M. Humbert, and Y. Zhang, "Graph unlearning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2022, pp. 499–513.

[9] V. S. Chundawat, A. K. Tarun, M. Mandal, and M. Kankanhalli, "Zero-shot machine unlearning," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 2345–2354, 2023.

[10] L. Du et al., "SoK: Dataset copyright auditing in machine learning systems," 2024, *arXiv:2410.16618*.

[11] L. Du et al., "SoK: Dataset copyright auditing in machine learning systems," in *Proc. IEEE Symp. Secur. Privacy (SP)*, Oct. 2024, p. 25.

[12] M. Du, Z. Chen, C. Liu, R. Oak, and D. Song, "Lifelong anomaly detection through unlearning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 1283–1297.

[13] T. Eisenhofer, D. Riepel, V. Chandrasekaran, E. Ghosh, O. Ohrimenko, and N. Papernot, "Verifiable and provably secure machine unlearning," in *Proc. IEEE Conf. Secure Trustworthy Mach. Learn. (SaTML)*, Jan. 2022.

[14] Y. Gao et al., "Backdoor attacks and countermeasures on deep learning: A comprehensive review," 2020, *arXiv:2007.10760*.

[15] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "STRIP: A defence against trojan attacks on deep neural networks," in *Proc. Annu. Comput. Security Appl. Conf. (ACSAC)*, 2019, pp. 113–125.

[16] A. Ginart, M. Guan, G. Valiant, and J. Y. Zou, "Making AI forget you: Data deletion in machine learning," in *Proc. Adv. Neural Inf. Process. Syst., Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, Vancouver, BC, Canada, Dec. 2019, pp. 3513–3526.

[17] A. Golatkar, A. Achille, and S. Soatto, "Eternal sunshine of the spotless net: Selective forgetting in deep networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9301–9309.

[18] L. Graves, V. Nagisetty, and V. Ganesh, "Amnesiac machine learning," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 13, pp. 11516–11524.

[19] C. Guo, T. Goldstein, A. Hannun, and L. Van Der Maaten, "Certified data removal from machine learning models," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 3832–3842.

[20] Y. Guo, Y. Zhao, S. Hou, C. Wang, and X. Jia, "Verifying in the dark: Verifiable machine unlearning by using invisible backdoor triggers," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 708–721, 2024.

[21] V. Gupta, C. Jung, S. Neel, A. Roth, S. Sharifi-Malvajerdi, and C. Waites, "Adaptive machine unlearning," in *Proc. Neural Inf. Process. Syst. (NeurIPS)*, Jan. 2021, pp. 16319–16330.

[22] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang, "Membership inference attacks on machine learning: A survey," *ACM Comput. Surv.*, vol. 54, no. 11s, pp. 1–37, Jan. 2022.

[23] Z. Izzo, M. A. Smart, K. Chaudhuri, and J. Zou, "Approximate data deletion from machine learning models," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 2008–2016.

[24] H. Jia et al., "Proof-of-learning: Definitions and practice," in *Proc. IEEE Symp. Security Privacy (SP)*, May 2021, pp. 1039–1056.

[25] Kaggle.*Kaggle Datasets*. Accessed: 4 May, 2025. [Online]. Available: https://www.kaggle.com

[26] A. Krizhevsky, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2012.

[27] S. Li, W. Deng, and J. Du, "Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2584–2593.

[28] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Anti-backdoor learning: Training clean models on poisoned data," in *Proc. NIPS*, vol. 34, 2021, pp. 14900–14912.

[29] N. G. Marchant, B. I. Rubinstein, and S. Alfeld, "Hard to forget: Poisoning attacks on certified machine unlearning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, no. 7, 2022, pp. 7691–7700.

[30] R. Mehta, S. Pal, V. Singh, and S. N. Ravi, "Deep unlearning via randomized conditionally independent hessians," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10412–10421.

[31] S. Neel, A. Roth, and S. Sharifi-Malvajerdi, "Descent-to-delete: Gradient-based methods for machine unlearning," in *Proc. 32nd Int. Conf. Algorithmic Learn. Theory*, 2021, pp. 931–962.

[32] European Parliament and the Council of the European Union. (2020).*General Data Protection Regulation*. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016R0679

[33] S. Schelter, S. Grafberger, and T. Dunning, "HedgeCut: Maintaining randomised trees for low-latency machine unlearning," in *Proc. Int. Conf. Manag. Data*, Jun. 2021, pp. 1545–1557.

[34] A. Sekhari, J. Acharya, G. Kamath, and A. T. Suresh, "Remember what you want to forget: Algorithms for machine unlearning," in *Proc. NIPS*, 2021, pp. 18075–18086.

[35] Amazon Web Services. *Amazon Data Exchange*. Accessed: 4 May, 2025. [Online]. Available: https://aws.amazon.com/data-exchange

[36] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 3–18.

[37] I. Shumailov et al., "Manipulating SGD with data ordering attacks," in *Proc. NIPS*, 2021, pp. 18021–18032.

[38] D. M. Sommer, L. Song, S. Wagh, and P. Mittal, "Athena: Probabilistic verification of machine unlearning," *Proc. Privacy Enhancing Technol.*, vol. 2022, no. 3, pp. 268–290, Jul. 2022.

[39] C. Song and V. Shmatikov, "Auditing data provenance in text-generation models," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 196–206.

[40] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German traffic sign recognition benchmark: A multi-class classification competition," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2011, pp. 1453–1460.

[41] United States.(2020). *California Privacy Rights Act*. [Online]. Available: https://www.cookiebot.com/en/cpra/

[42] H. Sun, T. Bai, J. Li, and H. Zhang, "ZkDL: Efficient zero-knowledge proofs of deep learning training," *IEEE Trans. Inf. Forensics Security*, vol. 20, pp. 914–927, 2025.

[43] D. Tang, X. Wang, H. Tang, and K. Zhang, "Demon in the variant: Statistical analysis of DNNs for robust backdoor contamination detection," in *Proc. 30th USENIX Secur. Symp.*, Aug. 2021, pp. 1541–1558.

[44] A. Thudi, G. Deza, V. Chandrasekaran, and N. Papernot, "Unrolling SGD: Understanding factors influencing machine unlearning," in *Proc. IEEE 7th Eur. Symp. Secur. Privacy (EuroS&P)*, Jun. 2022, pp. 303–319.

[45] A. Thudi, H. Jia, I. Shumailov, and N. Papernot, "On the necessity of auditable algorithmic definitions for machine unlearning," in *Proc. 31st USENIX Secur. Symp. (USENIX Secur.)*, 2022, pp. 4007–4022.

[46] E. Ullah, T. Mai, A. Rao, R. A. Rossi, and R. Arora, "Machine unlearning via algorithmic stability," in *Proc. Conf. Learn. Theory*, 2021, pp. 4126–4142.

[47] B. Wang et al., "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 707–723.

[48] Z. Wang, H. Ding, J. Zhai, and S. Ma, "Training with more confidence: Mitigating injected and natural backdoors during training," in *Proc. NIPS*, vol. 35, 2022, pp. 36396–36410.

[49] A. Warnecke, L. Pirch, C. Wressnegger, and K. Rieck, "Machine unlearning of features and labels," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2023.

[50] J. Weng, S. Yao, Y. Du, J. Huang, J. Weng, and C. Wang, "Proof of unlearning: Definitions and instantiation," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 3309–3323, 2024.

[51] Y. Wu, E. Dobriban, and S. Davidson, "DeltaGrad: Rapid retraining of machine learning models," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 10355–10366.

[52] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[53] M. Xue, G. Magno, E. Cunha, V. Almeida, and K. W. Ross, "The right to be forgotten in the media: A data-driven study," *Proc. Privacy Enhancing Technol.*, vol. 2016, no. 4, pp. 389–402, Oct. 2016.

[54] H. Yan, X. Li, Z. Guo, H. Li, F. Li, and X. Lin, "Arcane: An efficient architecture for exact machine unlearning," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, 2022, pp. 4006–4013.

[55] B. Zhang, Z. Chen, C. Shen, and J. Li, "Verification of machine unlearning is fragile," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Aug. 2024.

[56] Z. Zhang, Y. Zhou, X. Zhao, T. Che, and L. Lyu, "Prompt certified machine unlearning with randomized gradient smoothing and quantization," in *Proc. Neural Inf. Process. Syst. (NeurIPS)*, 2022, pp. 13433–13455.

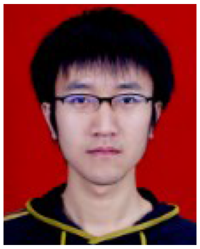
[57] C. Zhou et al., "PPA: Preference profiling attack against federated learning," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2023.

Chunyi Zhou received the Ph.D. degree from the School of Computer Science and Engineering, Nanjing University of Science and Technology, in 2024, supervised by Prof. Anmin Fu. He is currently a Post-Doctoral Researcher collaborating with Prof. Shouling Ji at the College of Computer Science and Technology, Zhejiang University. His research interests include AI privacy-preserving and security, federated learning, and machine unlearning.

Yansong Gao (Senior Member, IEEE) received the M.Sc. degree from the University of Electronic Science and Technology of China and the Ph.D. degree from The University of Adelaide, Australia. He is currently a Lecturer at The University of Western Australia. His current research interests include AI security and privacy, hardware security, and system security. He serves as an Associate Editor for IEEE Transactions on Information Forensics and Security, IEEE Transactions on Neural Networks, and *Learning Systems*.

Anmin Fu (Member, IEEE) received the Ph.D. degree in information security from Xidian University, Xi'an, China, in 2011. He is currently a Professor at Nanjing University of Science and Technology, China. He has published more than 100 technical papers, including international journals and conferences, such as IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE S&P, ACM CCS, NDSS, AsiaCCS, and ACISP. His research interests include the IoT security, cloud computing security, and privacy preserving.

Kai Chen (Member, IEEE) received the Ph.D. degree from the University of Chinese Academy of Sciences in 2010. He joined Chinese Academy of Sciences in January 2010. He became an Associate Professor in September 2012 and became a Full Professor in October 2015. His research interests include software analysis and testing, smartphones, and privacy.

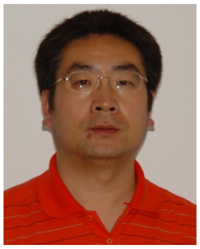


Zhi Zhang (Member, IEEE) received the bachelor's degree from Sichuan University, the master's degree from Peking University, China, and the Ph.D. degree in computer science from the University of New South Wales, Australia. He is currently a Lecturer at The University of Western University. His research interests include the areas of system security, row hammer, and adversarial artificial intelligence.

Minhui Xue is currently a Senior Research Scientist at Data61, CSIRO. He is also an Honorary Lecturer with Macquarie University. He was a recipient of the ACM CCS Best Paper Runner-Up Award, the ACM SIGSOFT Distinguished Paper Award, and his work has been featured in the mainstream press, including The New York Times and Science Daily. He serves on the Program Committee for IEEE S&P, ACM CCS, USENIX Security, and NDSS. He serves as an Associate Editor for IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING.

Zhiyang Dai received the bachelor's degree from the Qian Xuesen College, Nanjing University of Science and Technology, Nanjing, China, in 2021. He is currently pursuing the Ph.D. degree with the School of Cyber Science and Engineering, Nanjing University of Science and Technology. His research interests include AI privacy and security.

Shouling Ji (Member, IEEE) received the B.S. (Hons.) and M.S. degrees in computer science from Heilongjiang University, the Ph.D. degree in computer science from Georgia State University, and the Ph.D. degree in electrical and computer engineering from Georgia Institute of Technology (Georgia Tech). He is currently a ZJU 100-Young Professor with the College of Computer Science and Technology, Zhejiang University, and a Research Faculty with the School of Electrical and Computer Engineering, Georgia Tech. His current research interests include data-driven security and privacy, AI security, and big data analytics. He is a member of ACM and CCF. From 2012 to 2013, he was the Membership Chair of the IEEE Student Branch at Georgia State University.



Yuqing Zhang (Member, IEEE) received the B.S. and M.S. degrees in computer science and the Ph.D. degree in cryptography from Xidian University, China, in 1987, 1990, and 2000, respectively. He is currently a Professor and a Supervisor of Ph.D. Students with the University of Chinese Academy of Sciences, China. His research interests include cryptography and network security.