David M. Sommer*, Liwei Song*, Sameer Wagh, and Prateek Mittal

Athena: Probabilistic Verification of Machine Unlearning

Abstract: *The right to be forgotten*, also known as the right to erasure, is the right of individuals to have their data erased from an entity storing it. The status of this long held notion was legally solidified recently by the General Data Protection Regulation (GDPR) in the European Union. As a consequence, there is a need for mechanisms whereby users can verify if service providers comply with their deletion requests. In this work, we take the first step in proposing a formal framework, called Athena, to study the design of such verification mechanisms for data deletion requests – also known as *machine unlearning* – in the context of systems that provide machine learning as a service (MLaaS). Athena allows the rigorous quantification of any verification mechanism based on hypothesis testing. Furthermore, we propose a novel verification mechanism that leverages backdoors and demonstrate its effectiveness in certifying data deletion with high confidence, thus providing a basis for quantitatively inferring machine unlearning.
We evaluate our approach over a range of network architectures such as multi-layer perceptrons (MLP), convolutional neural networks (CNN), residual networks (ResNet), and long short-term memory (LSTM) and over 6 different datasets. We demonstrate that: (1) our approach has minimal effect on the accuracy of the ML service but provides high confidence verification of unlearning, even if multiple users employ our system to ascertain compliance with data deletion requests, and (2) our mechanism is robust against servers deploying state-of-the-art backdoor defense methods. Overall, our approach provides a foundation for a quantitative analysis of verifying machine unlearning, which can provide support for legal and regulatory frameworks pertaining to users' data deletion requests.

Keywords: machine unlearning, probabilistic verification, data deletion

DOI 10.56553/popets-2022-0072
Received 2021-11-30; revised 2022-03-15; accepted 2022-03-16.

***Corresponding Author: David M. Sommer:** Zühlke, david.sommer@zuehlke.com, Written while the author was employed by ETH Zürich.
***Corresponding Author: Liwei Song:** Princeton University, liweis@princeton.edu
Sameer Wagh: Princeton University, swagh@princeton.edu
Prateek Mittal: Princeton University, pmittal@princeton.edu

1 Introduction

Machine learning models, in particular neural networks, have achieved tremendous success in real-world applications and have driven technology companies, such as Google, Amazon, Microsoft, to provide machine learning as a service (MLaaS). Under MLaaS, individual users upload personal data to the server, the server then trains a ML model on the aggregate dataset and then provides its predictive functionality as a service to the users. However, recent works have shown that ML models memorize sensitive information of training data [1–4], indicating serious privacy risks to individual user data. At the same time, recently enacted legislation, such as the General Data Protection Regulation (GDPR) in the European Union [5] and the California Consumer Privacy Act (CCPA) in the United States [6], recognize the consumers' *right to be forgotten*, and legally requires companies to remove a user's data from their systems upon the user's deletion request.

However, there is a noticeable lack of concrete mechanisms that enables individual users to verify compliance of their requests. Prior works in machine unlearning [7–11] focus on the scenario of an honest server who deletes the user data upon request, and do not provide any support for a mechanism to verify unlearning. In this work, we formalize an approach that allows users to rigorously verify, with high confidence, if a server has "deleted their data". Note that our work refers to the exclusion of a user's data from a MLaaS' model training procedure and not to the physical deletion from storage since this is hard to ascertain.

Formalizing Machine Unlearning Verification. In this work, we propose Athena as the first step towards solving this open problem of verifying machine unlearning by individual users in the MLaaS setting. First, we formulate the unlearning verification problem as a hypothesis testing problem [12] (whether the server follows requests to delete users' data or not) and describe the metric used to evaluate a given verification strategy. Note that for a verifiable unlearning strategy to be effective, it needs to satisfy two important objectives. On the one hand, the mechanism should enable individual users to leave a unique *trace* in the ML model after being trained on user data, which can be leveraged in the verification phase. On the other hand, such a unique trace needs to have negligible impact on the model's normal predictive behavior. One possible approach is enabled by membership inference attacks such as Shokri et al. [2], Song et al. [13], or Chen et al. [14]. However, this line of work suffers from a number of limitations – low success rate due to the training data not being actively perturbed (as demonstrated in Section 6.1), extensive knowledge of the MLaaS model's architecture for white-box attack variants, access to auxiliary or shadow data and computational power in an extent similar to the MLaaS provider – all of which limit the feasibility of such approaches for our problem setting. We propose a novel use of backdoor attacks in ML as our mechanism for probabilistically verifying machine unlearning and demonstrate how it meets the two requirements above.

Proposed Verification Mechanism. In classical backdoor attacks [15, 16], the users (adversaries in these settings) manipulate part of training data such that the final trained ML model (1) returns a particular *target label* as the classification on inputs that contain a *backdoor trigger* (e.g., fixed pattern of pixel values at certain positions in the image) and (2) provides normal prediction in the absence of the trigger. In our machine unlearning verification mechanism, we extend the backdoor method proposed by Gu et al. [15]. In our approach, a fraction of users called *privacy enthusiasts* who are interested in the verification, individually choose a backdoor trigger and the associated target label randomly, then add this trigger to a fraction of their training samples (called data poisoning) and set the corresponding labels as the target label. This locally poisoned data is then provided to the MLaaS server. While each privacy enthusiast acts independently, i.e., they do not share information about their individual backdoor or target label, our approach supports an arbitrary fraction of such enthusiasts, up to the point where every user in the training dataset is applying our method. We demonstrate that the ML model trained on such data has a high backdoor success rate (i.e., target label classification in the presence of the trigger) for every user's backdoor trigger and target label pair. When the privacy enthusiast later asks the MLaaS provider to delete its data, it can verify whether the provider deleted its data from the ML model by checking the backdoor success rate using its own backdoor trigger with the target label. A low backdoor success rate is indicative of a model that is not trained on the poisoned data and thus signals that the server followed the deletion request. Through a rigorous hypothesis testing formulation, we can show that this mechanism can be used for high confidence detection of deletion requests. Such a verification system could be deployed by the Federal trade Commission (FTC) to validate the data deletion request compliance of real world systems such as Clearview AI which contains image data about millions of users and was reported to violate the privacy policies of Twitter [17, 18].

Theoretical and Experimental Evaluation. We theoretically quantify the performance of our backdoor-based verification mechanism under the proposed formulation of hypothesis testing. Furthermore, we experimentally evaluate our approach over a spectrum of 6 popular datasets (EMNIST, FEMNIST, CIFAR10, ImageNet, AG News, and 20News) and 4 different neural network architectures – multi-layer perceptrons (MLP), convolution neural network (CNN), residual network (ResNet), long short-term memory (LSTM). We show that our mechanism has excellent performance – using 50% poisoned samples and merely 30 test queries achieves both false positive and false negative value below 10^{-3} (and can be further lowered as shown in Table 2 on page 276). We also evaluate the mechanism under an adaptive malicious server, who uses state-of-the-art backdoor defense techniques (Neural Cleanse [19], Neural Attention Distillation [20], and SPECTRE [21]) to decrease the backdoor attack accuracy. We find that such a server can lower the backdoor success rate, especially for a low poisoning ratio, but backdoor success is still significant enough to validate unlearning with high confidence.

1.1 Our Contributions

The contributions of this work are threefold and can be briefly stated as follows:

(1) *Framework for Machine Unlearning Verification:* Athena is a rigorous framework for verifying compliance of the right to be forgotten requests by individual users, in the context of machine learning systems. We introduce the perspective of hypothesis testing to distinguish between an honest server following the deletion request and a malicious server arbitrarily deviating from the prescribed deletion. Our metric, the power of the hypothesis test, quantifies the confidence a user has in knowing that the service provider complied with its data deletion request. Our framework is applicable to a wide range of MLaaS systems.

(2) *Using Data Backdoors for Verifying Machine Unlearning:* To the best of our knowledge, we are the first to propose a backdoor-based mechanism for probabilistically verifying unlearning and show its effectiveness in the above framework. We provide a thorough mathematical analysis of our proposed mechanism. Theorem 1, informally stated, enables a user to find out the number of test samples re-

quired to achieve high confidence in verifying its deletion request. We also provide methods for users to estimate parameters and necessary statistics for high confidence detection of non-compliance.

(3) *Evaluating Proposed Mechanism over Various Datasets and Networks:* Finally, we perform a thorough empirical evaluation of our proposed mechanism on the previously mentioned 6 datasets over 4 different model architectures. We quantitatively measure the high confidence of our backdoor-based verification mechanism over different fractions of “privacy enthusiasts” – a set of users participating in the system that are interested in verifying machine unlearning, and show that it remains effective for an adaptive malicious server who uses state-of-the-art backdoor defenses to mitigate backdoor attacks. We also study a number of other aspects such as the improvements under the setting of collaborating users, performance in a continuous learning scenario, and show that our verification confidence in such settings is significantly higher than the baseline provided by membership inference.

2 Backdoor Attacks and Defenses

In this section, we provide a brief description of the state-of-the-art backdoor attacks and defenses, used extensively in this work. We focus on neural networks – a class of algorithms that enable supervised learning of certain tasks through labeled training data. In a backdoor attack, the adversary maliciously augments training samples with a hidden trigger into the training process such that when the *backdoor trigger* is added to any test input, the model will output a specific *target label*. Compared to data poisoning attacks which cause misclassifications on clean test samples via training set manipulations, backdoor attacks only alter model predictions in presence of the backdoor trigger and behave normally on clean samples.

We build upon the attack method of Gu et al. [15]. For a subset of the training samples, their attack chooses a backdoor pattern, applies it to the samples, and changes the labels to the target backdoor label. During the training process with the full dataset, the target model finally learns to associate the backdoor trigger with the target label. Recent works have improved this approach [16, 22], extended it to transfer learning [23], active learning [24], graph neural network [25], semi-supervised learning [26], used original labels of poison samples [27, 28], and directly manipulated training loss [29]. Note that the search for better backdoor attacks is orthogonal to the goals of this paper.

There have been a number of works that study defense mechanisms against backdoor attacks. Several *detection-only methods* have been proposed to identify whether the target model is backdoored or not by training a binary meta-classifier [30], searching compromised inner neurons [31], identifying the existence of poisoned samples based on their feature representations [32], or identifying potential malicious input regions for model predictions [33–35]. Although detection-only methods correctly distinguish backdoored models from normally trained models, the backdoor effects still remain in the model.

A number of works consider an alternative approach – *mitigation-based methods* – techniques that aim to erase backdoor patterns from backdoored models. Such mitigation methods can be divided into two categories based on their approach. One category is to directly update model parameters by fine-pruning inner neurons with a subset of benign samples [36], fine-tuning with knowledge distillation [20], or fine-tuning with reconstructed backdoor patterns [19, 37]. The other category first detects and removes backdoored samples based on their feature representations with clustering analysis [38], principle component analysis [39], or robust estimation [21], and then retrains the model on the left samples. In our paper, *we make a novel use of backdoor attack in the context of verifying machine unlearning,* where each interested user can employ an individual backdoor. We also test our mechanism under three of these defense strategies to show its resilience.

3 Verifying Machine Unlearning

Digital privacy is increasingly being recognized as a fundamental right providing users more control over their data. Right to be forgotten, the ability of users to have their data removed from ML services, is one such regulation. For MLaaS providers, such requests are known as machine unlearning requests [9]. However, there is no quantitative framework for verifying compliance of MLaaS providers to such data deletion requests, where users would like to ascertain if their data deletion requests were complied with (in ML model training). We provide, to the best of our knowledge, *the first quantitative framework* that can be used to measure the effectiveness of a given mechanism in being able to detect the non-compliance of a malicious MLaaS server. Note that while it is virtually impossible to ensure that the servers do not create copies of the user’s data, we provide a widely applicable framework for studying this problem from a pragmatic lens.

Symbol	Range	Description
n	$\mathbb{N}$	Number of test service requests per user
α,β	[0,1]	Type-I and Type-II errors (cf. Eq. 1)
p,q	[0,1]	Probabilities for analysis (cf. Eq. 4)
f_{user}	[0,1]	Fraction of users that are privacy enthusiasts (i.e., those who are verifying unlearning)
f_{data}	0-100%	Percentage of data samples poisoned by each privacy enthusiast
$\rho_{A,\alpha}(s,n)$	[0,1]	Effectiveness of a verification strategy s with a model training algorithm A and acceptable Type I error α

Table 1. Important notation used in this work.

Athena is a general framework and allows us to study various aspects of the unlearning problem, including:

(1) Is there a lightweight mechanism (requiring minimal assumptions on the user) for verification? Can such a verification mechanism be run only by a fraction of users in the system?
(2) What is the effect of an adaptive server strategy (server actively tries to defend against the verification mechanism) on the verification confidence?
(3) Is it possible to achieve significantly higher confidence in verification compared to approaches such as membership inference?
(4) Can a few users collaborate in order to improve the overall verification confidence?
(5) What is the impact of a server continuously learning the model with new input samples?

We make *a novel use of backdoor attacks [15] to propose such a verification mechanism.* As shown in Section 5, 6, our mechanism provides high confidence verification under all these scenarios.

Framework and Mechanism Overview. We employ the perspective of hypothesis testing to quantify the user's confidence in judging the compliance (or non-compliance) of the server. Furthermore, we propose a concrete mechanism for verifying machine unlearning that leverages the users' ability to inject backdoors into their data. In particular, a small fraction f_{user} of users which we call *privacy enthusiasts* locally perturb a fraction f_{data} of their data, henceforth called *poisoning*, and thus inject a backdoor in the data, that is only known to them. If the server trains the model on such data, the backdoor can help the user detect the model trained on the *poisoned data*. Consequently, this behavior can be used to reveal dismissed data deletion requests. Note that our system *does not require* all users to participate in the verification and, in particular, is shown to work even when only 5% of the users are privacy enthusiasts[1]. Our approach is illustrated in Figure 1. As an example use case, our approach could be deployed by FTC, recruiting competent privacy enthusiasts and providing them a dedicated software package to verify a company's compliance with deletion requests. We elaborate on this use-case in Section 6.4.

Threat Model and Assumptions. The scope of our work is limited to validating if users' data was removed from a specific machine learning model exposed by the MLaaS provider, and does not include validating deletion from other computing or storage resources at the provider. We assume that *privacy enthusiasts can control and manipulate sufficient data samples to inject a backdoor before they provide it for training*, which is a fair assumption in modern MLaaS platforms. Entities like FTC can also recruit users and provide dedicated software packages for verifying machine unlearning. When privacy enthusiasts do not have the ability to modify their data before sending, our approach is not applicable. Our backdoor-based mechanism only needs to access the final predicted label of the server's model. However, the server is not able to determine which user is querying the trained model. This can be achieved using off-the-shelf anonymous communication schemes like [40, 41]. Besides testing on a non-adaptive (natural) server algorithm, we also investigate our mechanism under various aspects including an adaptive server with backdoor defense techniques, continuous learning, comparison with membership inference baseline, and collaboration among users. While our approach is general, our evaluation focuses on vision and text datasets.

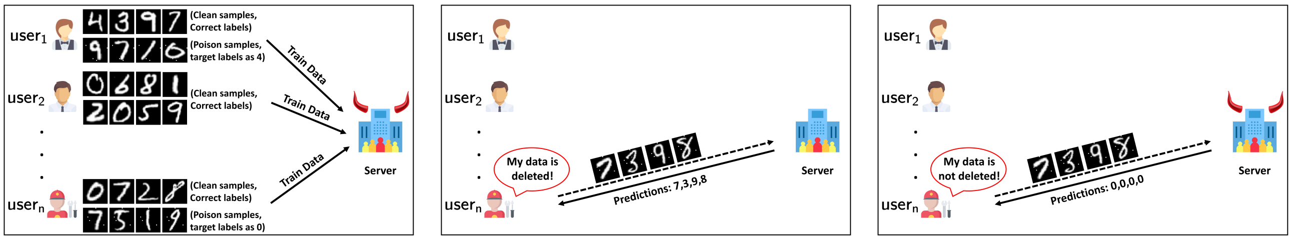
3.1 Verification via Hypothesis Testing

We frame the problem of verifying machine learning from the perspective of hypothesis testing to decide whether a MLaaS provider has deleted the requested user data from its training set or not. We define the *null hypothesis* H_0 to be the state when server deletes the user data and the *alternative hypothesis* H_1 to be the state when the server does not delete the data. We define the Type I errors as α (false positive) and Type II errors as β (false negative) given below:

$$\begin{aligned}\alpha &= \Pr[\text{Reject } H_0 | H_0 \text{ is true}]\\ \beta &= \Pr[\text{Accept } H_0 | H_1 \text{ is true}]\end{aligned} \tag{1}$$

We define the effectiveness of a verification strategy s for a given server algorithm A for a given acceptable toler-

1 And this, too, is limited only by the size of the datasets, the results should extend for smaller participating groups.



(a) Backdoor injection during model training. Here, user$_1$, user$_n$ are represented as privacy enthusiasts (poisoning data) and user$_2$ is not.

(b) When the server deletes the user's data (H_0), the predictions of backdoor samples are correct labels with high probability.

(c) When the server does not delete the user's data (H_1), the predictions of backdoor samples are target labels with high probability.

Fig. 1. (Overall system operation) First, users inject backdoor samples over which the server trains the model. At a later stage, users leverage model predictions on backdoored test samples to detect whether the server followed their deletion requests or not – as shown by the difference between Figure 1b and Figure 1c. We note that the impact on the benign prediction accuracy is minimal.

ance of Type I error (α) to be the power of the hypothesis test formulated above, i.e.,

$$\begin{aligned}\rho_{A,\alpha}(s,n) &= (1-\beta) \\ &= 1-\Pr[\text{Accept } H_0 \,|\, H_1 \text{is true}]\end{aligned} \tag{2}$$

where β, as shown in Section 4, can be computed as a function of α, s, and the number of testing samples n. Informally speaking, ρ quantifies the confidence the user has that the server has deleted their data. This deletion confidence $(1-\beta)$, the power of the hypothesis test, is the probability that we detect the alternative hypothesis when the alternative hypothesis is true. On the other hand, α refers to the acceptable value of the server being falsely accused of malicious activity when in practice it follows the data deletion honestly. For a given value of n, α and β cannot be simultaneously reduced and hence we usually set an acceptable value of α and then $(1-\beta)$ quantifies the effectiveness of the test.

3.2 Our Backdoor-Based Verification Scheme

As described earlier, we propose a system where a small fraction of users (privacy enthusiasts) actively engage in verification of machine unlearning. These privacy enthusiasts modify their data locally using a *private backdoor* that is only known to them individually, then they hand their (poisoned) data to the MLaaS provider. The models trained on the *poisoned data* (the data which contains such private backdoors) provide different predictions on very specific samples compared to models trained on data without poisoning. This property can be used to detect whether the server complies with data deletion requests or not. The different roles are described below:

	Privacy Enthusiasts	Normal Users	Server
Phase I	Send partially poisoned data	Send normal data	Receive data
Phase II	Send data deletion request	Send data deletion request	Receive data deletion requests
Phase III	Verify data deletion	(Do nothing)	(Do nothing)

Note that our mechanism requires no changes to normal users and the server. The user detects the non-compliance of the server based on the success of its backdoor attack. Privacy enthusiasts generate individual backdoor patterns that alter the predictions of samples to a fixed target label and subsequently provide their samples to the service. Once model prediction is accessible (before or after uploading of samples), the privacy enthusiasts measure the backdoor success rate when the backdoor has not been seen by a model for later hypothesis testing. This is achieved by testing the backdoor before uploading, or by testing with another unseen backdoor pattern once the model is accessible (cf. Section 3.3 for details). Finally, in the verification phase, privacy enthusiasts obtain predictions on n backdoored samples and based on the predictions run a hypothesis test to infer if the server complied with their request or not.

We also provide ways in which multiple such privacy enthusiasts can combine their requests to minimize the overall statistical error in correctly detecting server behaviour. For such a system to work well, it is imperative that there exists a statistically significant distinguishing test between models trained *with* vs *without* the backdoored user's data. At the same time, the backdoored data should have minimal impact on the model's normal performance. Through extensive evaluation (Section 5), we show these hold for our mechanism.

3.3 Technical Details

We apply a backdoor using the method described in Section 2, i.e, setting 4 user-specific pixels, spots, or words to a dataset dependent value and changing the label to a user-specific target label. Note that the success of altering the prediction using backdoored samples is usually not guaranteed every single time but in a probabilistic manner. Thus, the decision on whether the data has been deleted or not is determined by a hypothesis test. For an effective backdoor algorithm, when the model was trained on backdoored data, the probability of receiving the target label in presence of the backdoor pattern, i.e., backdoor success rate, should be high. At the same time, when the provider has deleted the user's data (the model has not been trained on the user's backdoored samples), the backdoor success rate should be low. In this way the hypothesis test can distinguish between the two scenarios.

We aim to distinguish the scenario of an honest server which follows the data unlearning protocol from that of a dishonest server who can operate arbitrarily. In particular, we consider two specific models for the dishonest server – the first is *non-adaptive* and does not delete user data yet expects to not get detected, while the second is *adaptive* and employs state-of-the-art defense mechanisms to mitigate user strategies (while also not deleting user data) and thus actively works to evade detection. Throughout this paper, the two probabilities corresponding to backdoor attack accuracy for deleted data and undeleted data are referred to as q (lower), and p (higher) respectively. Furthermore, the confidence of this test increases with n, the number of backdoored test samples a user queries the trained ML model with. Thus our verification mechanism can be used to detect missing deletion with high confidence by increasing the number of test samples.

Parameter Estimation. Given that estimation of p and q is central to the detection of non-compliance, we describe the approach from an individual user perspective below.

(1) *Estimating p:* A user can obtain an estimated $\hat{p}$ by querying the model with backdoored samples before the deletion request is made. At this moment, a user can determine whether the backdoor strategy is working. If $\hat{p}$ is close to the random prediction baseline, either the applied backdoor strategy s is not working, or its data has not been used in training. However, if $\hat{p}$ is significantly higher than the baseline, our strategy s can work well and we can use $\hat{p}$ as an estimate.

(2) *Estimating q:* There are two ways of obtaining an estimate $\hat{q}$: If the algorithm can be queried before the user provides its data, $\hat{q}$ can be obtained by querying the algorithm using samples with the user's backdoor the algorithm has not seen before. If this is not possible, the user can estimate $\hat{q}$ by generating another backdoor pattern that the algorithm has (likely) not seen during training and querying the algorithm with samples that have this new backdoor applied, simultaneously while estimating $\hat{p}$. The output should be similar to the case where the algorithm has never seen the user's legitimate backdoor pattern.

In the highly improbable event when the estimated p is close to the estimated q (cf. Section 5.4 for why this might happen), the privacy enthusiasts can detect this and use alternative strategies (cf Section 6.5). We provide formal closed-form expressions for these in terms of system parameters in Section 4.

Continuous Learning. The server may continuously update the model as new training data becomes available. Backdoor patterns that previously worked could be overwritten by new data and may result in poor performance. Our framework can also be used to measure its impact on the verification performance. In such a scenario, users can resubmit their data containing backdoors in order to maintain high "verifiability." Through our evaluation, we show that continuous learning does not significantly impact the verification performance and the experimental results are presented in Section 6.2.

User Collaboration. In our framework, each privacy enthusiast can verify the server's compliance independent of other privacy enthusiasts. However, any group of privacy enthusiasts can collaborate to jointly improve the performance of the verification. This benefits the robustness of the verification, as the risk of incorrectly accusing a complaint server reduces. Such collaborative approaches are particularly interesting in our application scenario involving FTC where the privacy enthusiasts have an easy way to collaborate among themselves. We discuss the impact of collaborating users in Section 5.4.

4 Theoretical Analysis

In Section 3, we set up the problem of measuring the effectiveness of user's strategies as a hypothesis test. The hypothesis test allows us to know *what to measure*, i.e., the confidence, for a meaningful quantification of the effectiveness. However, we still *need to measure* this confidence as a function of system parameters and this is what we achieve in this section. In particular, we provide a closed form expression for the confidence $\rho_{A,\alpha}(s,n)$ and provide crucial

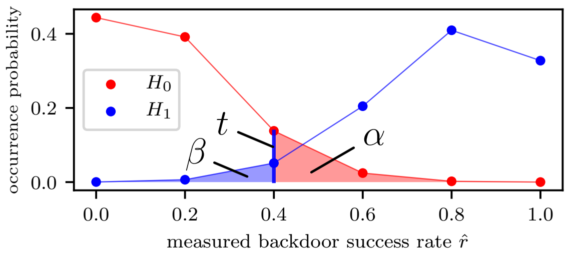


Fig. 2. This figure shows intuitively the relation between the threshold t and the Type I (α) and Type II (β) errors for number of measured samples $n=5$, with $q=0.1$, and $p=0.8$

bounds when measured parameters (empirical values) are different from the ground truth (theoretical values).

The confidence, expressed by the metric $\rho_{A,\alpha}(s,n)$, is based on a hypothesis test where two cases are compared: H_0 – the data has been deleted, and H_1 – the data has not been deleted. We measure the Type II error β which denotes the probability that the server evades detection, i.e., the server behaves malicious but is not caught. Note that this requires we set a level of acceptable Type I error α, i.e., the probability that we falsely accuse the server of avoiding deletion. Hence, the metric $\rho_{A,\alpha}(s,n) = 1-\beta$ is a function of the backdoor strategy s and the number of predictions on backdoored test samples n for a given MLaaS server A and a value of Type I error α. Figure 2 shows the mechanics of the hypothesis test.

4.1 Formalizing the Hypothesis Testing

We query the ML-mechanism A with n backdoored samples $\{\mathsf{sample_i}\}_{i=1}^{n}$ of a single user and measure how often the ML-mechanism does classify the samples as the desired target label, denoted as $\mathsf{Target_i}$. Then, the measured success rate is

$$\hat{r} = \frac{1}{n}\sum_{i=1}^{n}\begin{cases}1 \text{ if } A(\mathsf{sample_i}) = \mathsf{Target_i} \\ 0 \text{ otherwise}\end{cases} \tag{3}$$

We define two important quantities q,p that quantify the probability that the prediction on backdoored samples is equal to the target label for the null hypothesis vs the alternative hypothesis. For all available data points i,

$$\begin{aligned} q &= \Pr[A(\mathsf{sample_i}) = \mathsf{Target_i} | H_0 \text{ is true}] \\ p &= \Pr[A(\mathsf{sample_i}) = \mathsf{Target_i} | H_1 \text{ is true}] \end{aligned} \tag{4}$$

Note that the measure $\hat{r}$ approaches q if the null hypothesis H_0 (data was deleted) is true and approaches p if the alternative hypothesis H_1 (data was not deleted) is true. To decide whether we are in H_0 or H_1, we define a threshold t and if $\hat{r} \leq t$ we output H_0 else output H_1. The false-positive error α (Type I error) and false-negative error β (type II error) are the respective leftover probability masses that we have decided wrongly. This is illustrated in Figure 2.

The threshold t is set according to the desired properties of the hypothesis test. As common in statistics, t is set based on a small value of α (also known as p-value), the probability that we falsely accuse the ML-provider of dismissal of our data deletion request.

4.2 Estimating Deletion Confidence

To derive an analytic expression for $\rho_{A,\alpha}(s,n)$, we note that the order in which we request the prediction of backdoored samples does not matter. Moreover, we assume that the ML provider returns the correct target prediction label with probability q for users with fulfilled deletion requests, and p otherwise. Further, we assume that the ML provider is not aware which user is querying. Else, the provider could run user specific evasion strategies, e.g., having for each user a unique model with only the user's data excluded. This can, for example, be achieved by an anonymous communication channel. Since the user strategy is completely defined by the two parameters q and p, we will often interchangeably express a strategy s for these cases by $s=(q,p)$.

First, we show that the occurrence probability of a user-measured average backdoor success ratio $\hat{r}$ follows a binomial distribution with abscissa rescaled to [0,1] with mean q (deleted), or p (not deleted) respectively. Then, we compute the Type II error β based on the Type I error α that results from the overlap of these two binomial distributions. Finally, we derive an analytic expression for the verification confidence:

Theorem 1. *For a given ML-mechanism A and a given acceptable Type I error probability α, the deletion confidence $\rho_{A,\alpha}(s,n)$ is given by the following expression:*

$$\rho_{A,\alpha}(s,n) = 1 - \sum_{k=0}^{n}\binom{n}{k}p^k(1-p)^{n-k}\cdot \\ H\left[\sum_{l=0}^{k}\binom{n}{l}q^l(1-q)^{n-l} \leq 1-\alpha\right] \tag{5}$$

where p,q are as given by Equation (4) and $H(\cdot)$ is the heavy-side step function, i.e., $H(x)=1$ if x is True *and 0 otherwise.*

Theorem 1 gives a closed-form expression to compute the backdoor success probability as a function of the system parameters. We defer the proof to Appendix B.

4.3 Relaxation to Single User Perspective

In our theoretical analysis above, we have assumed to know p and q perfectly. In a real-world setup, these values are always measurements. While a machine learning service provider has the ability to quantify p and q accurately on a lot of samples, single users that want to verify the unlearning of their data do usually not have this kind of opportunity. They need to work with estimated values $\hat{p}$ and $\hat{q}$ which can be obtained on a low number of samples n as described in Section 3.2. We observe that if we overestimate $\hat{q}$ with a bound $\hat{q}'$ and underestimate $\hat{p}$ with a bound $\hat{p}'$, then the metric $\rho_{A,\alpha}(s,n)$ provides a lower bound, i.e., the confidence guarantees given by ρ do not worsen if the distance between $\hat{q}$ and $\hat{p}$ increases.

$$\rho_{A,\alpha}(s=(\hat{q}',\hat{p}'),n) \leq \rho_{A,\alpha}(s=(\hat{q},\hat{p}),n) \tag{6}$$

with $\hat{q} \leq \hat{q}'$ and $\hat{p} \geq \hat{p}'$. This comes from the fact that for a given α the overlap of the two scaled binomial distribution decreases when they are moved further apart, and thereby decreasing the β which in terms defines ρ. Alternatively, users can assume priors for p and q and apply Bayes' theorem to compute $\rho_{A,\alpha}$ (the expectation over all values of p,q):

$$\Pr[r|\hat{r},n] = \frac{\Pr[\hat{r}|r,n]\Pr[r]}{\Pr[\hat{r}|n]} \tag{7}$$

for $r \in \{q,p\}$ given an estimation $\hat{r} \in \{\hat{q},\hat{p}\}$. Using $\Pr[\hat{r}|n] = \sum_r \Pr[\hat{r}|r,n]$, we obtain,

$$\rho_{A,\alpha}(s=(\hat{q},\hat{p}),n) = \mathrm{E}_{\Pr[q|\hat{q},n],\Pr[p|\hat{p},n]}[\rho_{A,\alpha}(s=(q,p),n)]$$

5 Experimental Evaluation

In this section, we describe the important results of our experimental evaluation. Before elaborating on datasets and the experimental setup in Section 5.1, we mention the central questions for this study as well as results briefly:

- *Q: How well does the verification mechanism work in detecting avoided deletion? Does this generalize to complex and non-image datasets and different architectures?* We answer all these questions affirmatively in Section 5.2. These results are presented in Figure 3.
- *Q: What happens when the server uses an adaptive strategy such as using a state-of-the-art backdoor defense algorithm to evade detection?* While the detection accuracy is slightly reduced, our approach still excels. This is discussed in Section 5.3 and shown in Figure 4.
- *Q: How do the results change with the fraction of users participating in unlearning detection?* Our approach works for an arbitrary fraction of privacy enthusiasts, as long as individual backdoors are sufficiently reliable (cf. Figure 3c, 4c).

Furthermore, in Sections 5.4 and 6, we look at other aspects of our work including the limitations of our approach when used in practice. We will make our code publicly available to facilitate reproducible experiments.

5.1 Experimental Set-up

We evaluate our experiments, wherever applicable, on the 6 datasets and on 4 different model-architectures: the image dataset *Extended MNIST* (EMNIST) [42] with a Multi-Layer-Perceptron (MLP) [43], *Federated Extended MNIST* (FEMNIST) [44] with a convolution neural network (CNN), *CIFAR10* [45] and *ImageNet* [46] with residual networks (ResNet) [47], and text-based datasets *AG News* [48] and *20News* [49] with long short-term memory (LSTM) models [50, 51]. Table 2 presents an overview and full details (datasets, network architectures, and the backdoor methods) are deferred to Appendix A.

Machine Unlearning Verification Pipeline. The first part of our evaluation examines the distinguishability of backdoor success rates for data owner whose data has been deleted by a benevolent MLaaS provider versus the case where the provider has maliciously avoided deletion. First, privacy enthusiasts (with a fraction of f_{user} among all users) apply their specific backdoor patterns on a certain percentage (f_{data}) of their training samples, i.e., 4 random pixels, spots, or words are overwritten and their labels are set to a user specific target label. After training the model with the partially backdoored dataset, we compute the backdoor success rate for each privacy enthusiast's backdoor trigger with its target label, formerly denoted by p in Equation (4). Then, we compute the backdoor success rate on poisoned users whose data have been excluded before training, introduced as q in Equation (4). We compare these values against the benign accuracy of the models on unpoisoned inputs. Finally, we illustrate the decreasing average Type-II error β (cf. Equation (1)) for a range of number of measurements n with a given Type-I error α, leading to an increasing average deletion confidence $\rho_{A,\alpha}(s,n)$. As servers can defend against backdoor attacks, we illustrate the success of our approach in a comparison of a non-adaptive server that does not implement backdoor defenses to an adaptive server that implements state-of-the-art defenses.

Optimally, such an evaluation excludes each poisoning user individually from the full dataset and then retrains the model for each exclusion again from scratch. Due to computation power restrictions, on all tested datasets except

Dataset Details						ML Model			Non-adaptive server (50% poison ratio)			
Name	sample dimension	number of classes	number of total samples	number of total users	backdoor method	model architecture	train acc. (no backdoor)	test acc. (no backdoor)	benign test acc	p	q	β
EMNIST	28×28	10	280,000	1,000	set 4 random pixels to be 1	MLP	99.84%	98.99%	98.92%	95.60%	10.98%	$3.2\cdot10^{-22}$
FEMNIST	28×28	10	382,705	3,383	set 4 random pixels to be 0	CNN	99.72%	99.45%	99.41%	99.98%	8.48%	$2.2\cdot10^{-77}$
CIFAR10	$32\times32\times3$	10	60,000	500	set 4 random pixels to be 1	ResNet20	98.98%	91.03%	90.54%	95.67%	7.75%	$4.1\cdot10^{-24}$
ImageNet	varying sizes, colorful	1000	1,331,167	500	set 4 random spots[2] to be 1	ResNet50	87.43%	76.13%	75.54%	93.87%	0.08%	$2.0\cdot10^{-34}$
AG News	15–150 words	4	549,714	580	replace 4 out of last 15 words	LSTM	96.87%	91.56%	91.35%	95.64%	26.49%	$6.6\cdot10^{-12}$
20News	5–11795 words	20	18,828	100	replace 4 out of last 15 words	LSTM	96.90%	81.18%	81.31%	75.43%	4.54%	$2.8\cdot10^{-10}$

Table 2. Summary of datasets, models, and our backdoor-based verification performance for a non-adaptive sever, with a fixed fraction of privacy enthusiasts $f_{\text{user}} = 0.05$, with the backdoor success rate of undeleted users p, and the backdoor success rate of deleted users q. We provide the Type-II error (β) of our verification with 30 test samples and α as 10^{-3}.

ImageNet, we separated 20% of the available users before training, and trained the models on the leftover 80% of the users. Therefore, the first 20% of users were not included in any training and act in the evaluation as users where the service provider complies their data deletion requests (H_0). We call them "deleted users". Accordingly, we call remaining users "undeleted users" and further split their data into a training and a test set (80% samples are in training set, and remaining 20% samples are in test set). We trained the model on undeleted users' training sets with poisoned samples and measure their backdoor success using test samples with their backdoor patterns, which refers to the case where the users' data was not deleted (H_1). On the large-scale ImageNet, we follow the existing training/test split to include all users' training samples to train the ML model and obtain the backdoor success of "undeleted users" (p). To simulate the behaviors of "deleted users", we apply newly generated backdoor patterns to the test-split, and derive the unseen backdoor success q based on their predictions. Where the resulting numbers were statistically insufficient due to a very low privacy enthusiasts fraction f_{user}, we repeated the experiments with different random seeds and took the average.

5.2 Results for a Non-Adaptive Server

We first present the evaluation results for a non-adaptive server, where the server uses the static learning algorithm to train the ML model. On each dataset, we compute the backdoor success rate for each privacy enthusiasts, and compute the undeleted users' average success rate p and deleted users' average success rate as q to evaluate the performance of our machine unlearning verification method with different numbers n of test queries, following Theorem 1.

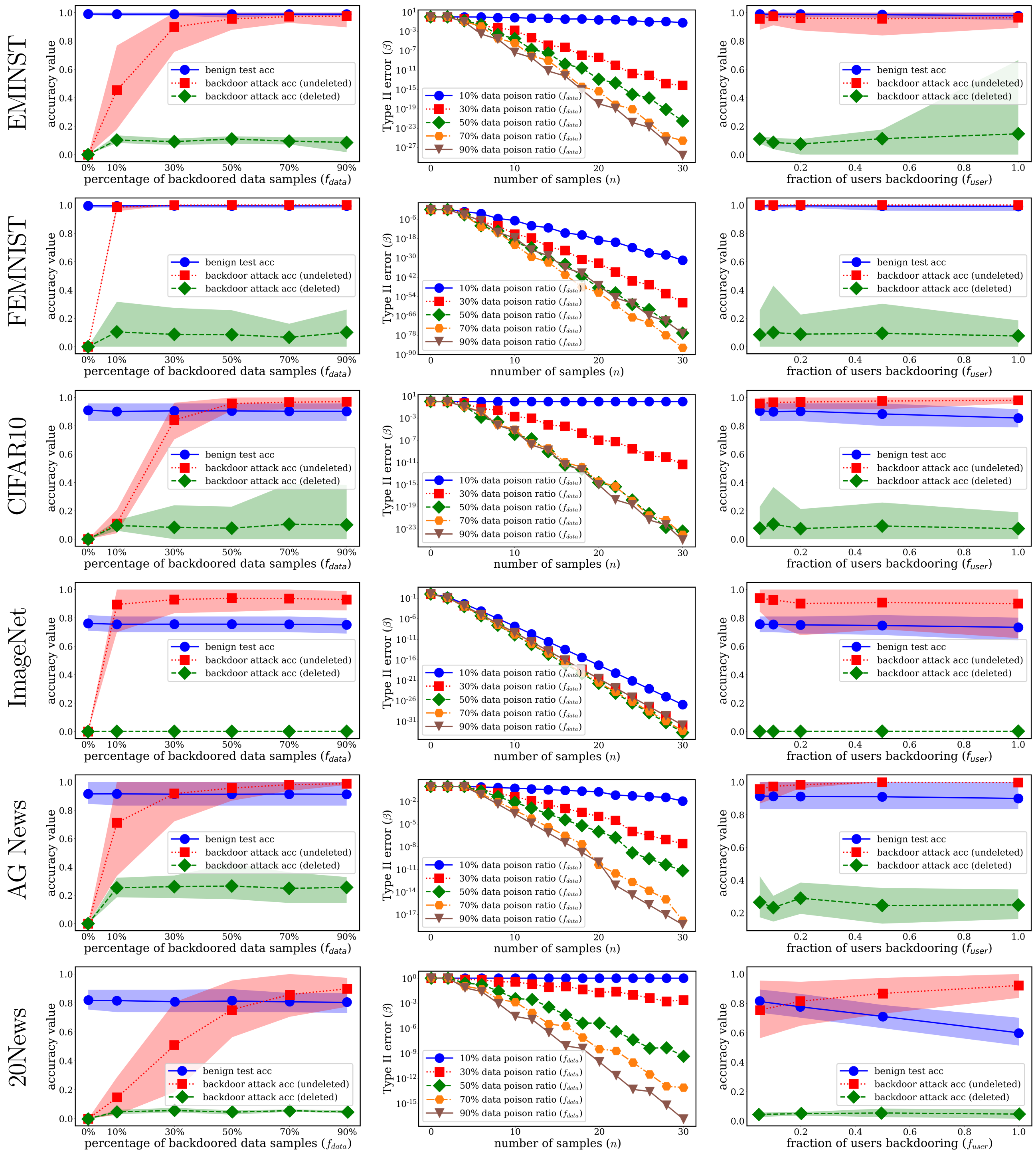
First, **our verification mechanism works well with high confidence on the EMNIST dataset.** From Figure 3a for EMNIST, we can see that the attack accuracy for undeleted users (p) increases with the poison ratio, while the attack accuracy for deleted users (q) stays around 10% (random guess accuracy). At the same time, poison ratios as high as 90% have negligible impact on model accuracy for clean test samples. Figure 3b for EMNIST shows that a higher poison ratio also leads to a lower Type-II error in our verification mechanism due to a larger gap between p and q. As a reference, the performance of our verification mechanism is multiple orders of magnitude better than a membership inference baseline (cf. Section 6.1).

Second, **our verification mechanism generalizes to more complex image datasets.** The accuracy performance and the verification performance for the non-IID FEMNIST dataset, the CIFAR10 dataset, and the much more complex ImageNet dataset is presented in Figure 3 as well. Similar to EMNIST, the gap between p and q becomes larger when increasing the poison ratio. For ImageNet, the backdoor attack accuracy for deleted users and its 80% quantile are comparatively small due to its number of prediction classes, namely 1000 compared to 10, 20, or 4 for the other datasets.

Third, **our verification mechanism is also applicable to non-image datasets**, illustrated by the AG News and 20News datasets. With poison ratio above 50%, the undeleted users' backdoor attack accuracy is consistently above 75% while the deleted users' backdoor success stays around the random guess accuracy.

Fourth, **our mechanism works for arbitrary fraction f_{user} of privacy enthusiasts testing for deletion verification**, illustrated in Figure 3c. Previously, we only considered a user backdooring rate f_{user} of 0.05. While such a scenario is more realistic, i.e., when only a few privacy

2 As the ImageNet dataset contains large colorful pictures of various resolutions, we decided to create a transparent 32x32x3 pixel mask with 4 pixels backdoored and upscale this to the corresponding picture size before applying.

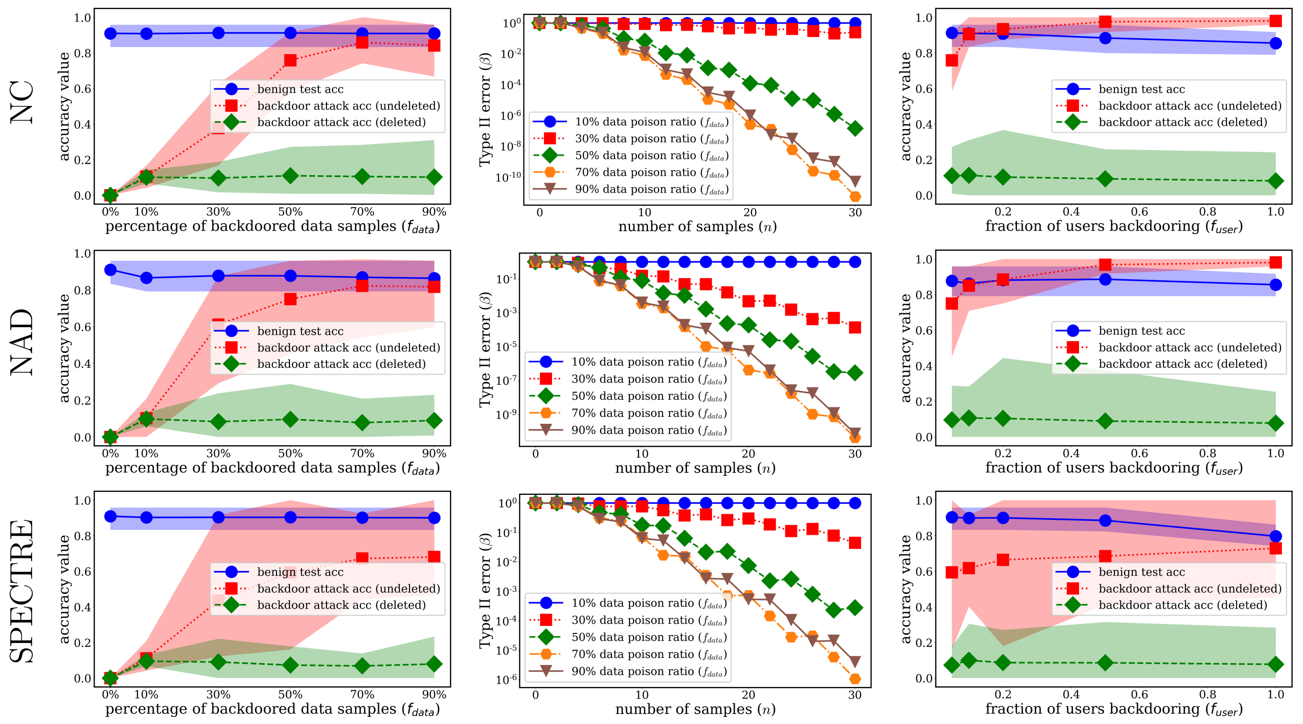


(a) Model accuracy and backdoor success rate for poisoning user fraction $f_{\text{user}} = 0.05$.

(b) Verification performance with varied poison ratios for a fixed $\alpha = 10^{-3}$ and $f_{\text{user}} = 0.05$.

(c) Model accuracy and backdoor success rate for fixed poison ratio $f_{\text{data}} = 50\%$.

Fig. 3. Our backdoor-based machine unlearning verification results with a *non-adaptive server*. Each row of plots is evaluated on the data-set specified at the most-left position. Each column of plots depicts the evaluation indicated in the caption at its bottom. The colored areas in columns (a) and (c) tag the 10% to 90% quantiles.



(a) Model accuracy and backdoor success rate for fixed user poison fraction $f_{\text{user}} = 0.05$.

(b) Verification performance with varied poison ratios for a fixed $\alpha = 10^{-3}$ and $f_{\text{user}} = 0.05$.

(c) Model accuracy and backdoor success rate for fixed poison ratio $f_{\text{data}} = 50\%$

Fig. 4. Our backdoor-based machine unlearning verification results with *Adaptive servers on CIFAR-10 dataset*. Each row of plots is evaluated on the backdoor defense specified at the most-left position. Each column of plots depicts the evaluation indicated in the caption at its bottom. The colored areas in columns (a) and (c) tag the 10% to 90% quantiles.

enthusiast test a ML provider for deletion validity, it might happen that more users test. Even when all users are testing, the benign accuracy of the models are barely impacted. For EMNIST, a larger f_{user} leads to an increased number of backdoor collisions: While the average of the backdoor attack accuracy for deleted users increases only minimally, a few deleted users measure a high success rate. We discuss mitigation strategies in Section 5.4.

Defense method	benign test acc	p	q	β
no defense	90.54%	95.67%	7.75%	4.1×10^{-24}
NC [19]	90.92%	75.90%	10.99%	1.4×10^{-7}
NAD [20]	88.72%	74.94%	9.60%	2.9×10^{-7}
SPECTRE [21]	90.15%	59.41%	7.32%	2.8×10^{-4}

Table 3. Verification performance against adaptive servers deploying backdoor defenses (CIFAR10 dataset); fixed fraction of privacy enthusiasts $f_{\text{user}} = 0.05$, each poisoning $f_{\text{data}} = 50\%$. We measure the Type-II error β with $n = 30$ test samples and $\alpha = 10^{-3}$.

5.3 Results for an Adaptive Server

To evaluate verification performance for an Adaptive server, we choose three state-of-the-art backdoor defense methods – Neural Cleanse (NC) [19], Neural Attention Distillation (NAD) [20], and Spectral Poison ExCision Through Robust Estimation (SPECTRE) [21].

NC [19] first reverse-engineers backdoor triggers by searching for minimal input perturbations needed for a target label classification, then uses these triggers to poison certain samples with correct labels along with some benign samples to fine-tune the model for few training epochs. NAD [20] first finetunes the model on a small subset of benign samples to get a "teacher" model and further uses the same subset to update the original backdoored model with both prediction loss and knowledge distillation loss, which is the norm of difference in normalized intermediate-layer representations between the model and the teacher model. SPECTRE [21] first applies principle component analysis to reduce the dimension of training samples' representations, then estimates mean and covarariance of clean data using robust estimation techniques [52]. Next, it computes outlier scores based on quantum entropy scoring [53] to remove high score samples from the training set. Finally, it trains the model only on the remaining training data.

We present our backdoor-based verification under the above three backdoor defense methods on the CIFAR10 dataset in Figure 4 and Table 3. We achieve similar results on other datasets for adaptive servers, which are presented in the Appendix (Figures 6 and 7). As expected, **all three methods reduce the backdoor attack success rate.** As seen in Figure 4a, the undeleted users' backdoor attack accuracy (red line) is reduced, especially for smaller data poison ratios f_{data}. Among these defense methods, SPECTRE [54] achieves best defense performance.

Second, **even with a small number of samples, our verification quickly achieves high confidence.** As shown in Table 3, although defense approaches decreases backdoor success (p) down to around 59%, even with 30 test samples, we achieve Type-II errors (β) lower than 10^{-3}.

Third, **the performance of the defense weakens with an increasing fraction of users testing for deletion verification (f_{user}).** As seen in Figure 4c, with an increasing fraction of users, backdoor attack accuracy for undeleted users also increases, leading to better verification performance as well.

5.4 Heterogeneity Across Individual Users

So far, our analysis on deletion verification performance is evaluated based on the "average" backdoor attack accuracy p across all undeleted users and the average backdoor attack accuracy q across all deleted users. Next, we evaluate the heterogeneity in the performance of stochastic deletion verification across individual users, to account for the variance in individual users' backdoor attack accuracy values. We find that while most privacy enthusiasts are able to conclude correctly whether their data has been deleted, a small subset of deleted users also have high backdoor attack accuracy although the ML model never trained on their backdoor triggers and target labels.

To quantify this effect, we present the cumulative distribution plots for non-adaptive server over different datasets in Figure 8 (in Appendix), where we fix the fraction of privacy enthusiasts f_{user} as 0.05 and data poison ratio f_{data} as 50%. As shown in Figure 8, almost all undeleted users have high backdoor attack accuracy close to 100%. However, several deleted users indeed have high backdoor attack accuracy. We think the reason is that there are one or more undeleted users with *similar* backdoor triggers and the same target labels as those rare deleted users, resulting in their high backdoor attack accuracy without their data being used in the training set of the ML model. In fact, popular image classification architectures, such as CNN and ResNet, are trained to behave similarly for images with rotation, translation, and transformation, leading to behave similarly on similar triggers.

We resolve this issue by proposing that multiple users collaborate by sharing their estimated backdoor success rates and thereby achieve high confidence verification. With this, we greatly reduce the likelihood of cases where deleted users with high backdoor success wrongly blame the server for not deleting the data.

Specifically, we decide whether or not to reject the null-hypothesis H_0 (the server does delete) based on accumulated p and q values. Therefore, c collaborating users compute the mean of their p and q, decide for an α (Type-I-error) and a upper bound for β (Type-II-error). If the estimated Type-II error is smaller than the bound, the null hypothesis is rejected. Note that if c users share their results, and each user tested n backdoored samples, then the accumulated number of tested samples is $c{\cdot}n$. In the following table, we show the probability that a server does not fulfil deletion requests, but the null-hypothesis is falsely accepted (false negative), for n=30 samples, f_{user}=0.05, f_{data}=50%, and with α=10^{-3} and an upper bound for β=10^{-3}. For 20News, n/a indicates that the (limited) empiric probability estimation resulted in 0 because the p and q values are to easy to distinguish (see Figure 8). **With multi-user cooperation, the probability of false negatives can be greatly reduced.** For computational efficiency, we applied Monte-Carlo-sampling.

	EMNIST	FEMNIST	CIFAR10	ImageNet	AG News	20News
1 user	$2.1{\times}10^{-2}$	$2.5{\times}10^{-2}$	$3.8{\times}10^{-2}$	$4{\times}10^{-4}$	$8.1{\times}10^{-2}$	$7.0{\times}10^{-2}$
2 users	$1{\times}10^{-4}$	$3{\times}10^{-4}$	$1{\times}10^{-3}$	$4{\times}10^{-5}$	$1.3{\times}10^{-2}$	n/a (0.0)
3 users	$<10^{-5}$	$<10^{-5}$	$<10^{-5}$	$<10^{-5}$	$4{\times}10^{-4}$	n/a (0.0)

To further make our verification mechanism more reliable, we can use multiple backdoor triggers with multiple target labels for each user and estimate the lowest backdoor success rate among all triggers. As long as the deleted user has at least one trigger leading to low attack accuracy, we can obtain reliable performance from a worst-case perspective. We discuss the number of possible backdoors based on coding theory in the Appendix C. Another direction is to combine our method with other verification methods, such as user-level membership inference attacks to detect whether a user's data was used to train the ML model or not [13]. We leave this as future work.

6 Discussion

Here, we elaborate on additional aspects such as a membership inference baseline, effects of continuous learning, future backdoor defenses and differential privacy, concrete use-cases, and limitations of our verification system.

6.1 Membership Inference Baseline

To validate the necessity of our backdoor-based verification mechanism, we provide the verification performance by user-level membership inference attacks [13] without poisoning training data. Membership inference, in the context of machine unlearning, uses the obtained model prediction vector to guess whether the sample was used during training. Specifically, we perform membership inference attack on each data sample by comparing the prediction confidence to a threshold, a method known to be competitive among different attack methods [55–57]. The true positive rate and the false positive rate of such membership inference attacks, where positives indicate members, are equivalent to p and q in Equation (4). Then, we follow Section 4.2 to compute the verification confidence.

	EMNIST	FMNIST	CIFAR10	ImageNet	AG News	20News
β $(\alpha{=}10^{-3})$	0.998	0.994	0.949	0.951	0.981	0.715
β $(\alpha{=}10^{-1})$	0.849	0.787	0.156	0.353	0.262	0.030

We present the verification results by **membership inference baseline** with 30 test samples in the table above. For well-generalized datasets like EMNIST and FEMNIST, where training accuracy almost equals the test accuracy, the membership inference attack mostly fails, leading to high Type-II errors ($\beta\approx 1-\alpha$). Although we can achieve β below 0.4 on remaining datasets with $\alpha{=}0.1$, the performance is significantly worse than our backdoor-based approach, where we can have both α and β smaller than 10^{-3} with just 30 samples, even when considering adaptive servers deploying backdoor defenses (see Table 3).

6.2 Impact of Continuous Learning

The ML server may continuously update ML models on new incoming training samples. Here, we investigate how continuous learning will impact the backdoor success and verification performance.

In Figure 5, we present the model accuracy and backdoor success of CIFAR-10 classifiers after continuous learning on 3840 new benign training samples (10% of original training set) using an Adam optimizer with a learning rate of 10^{-4} for 4 epochs. Compared to Figure 3, continuous learning marginally decreases the backdoor success of undeleted users, as expected. However, there is still a large gap between undeleted users' backdoor success (p) and deleted users' backdoor success (q) that **allows for verification with high confidence**. Further, we find that the decrease in undeleted users' backdoor success p by continuous learning is smaller than that by our tested backdoor defenses in Figure 4. This is expected since NC [19] and NAD [20] themselves apply a more targeted form of continuous learning.

One may argue that if we use more samples and more training epochs to continuously update ML models, the backdoor success and verification performance can be further reduced. However, users can also leverage continuous learning to inject poisoned samples to ensure high backdoor success and high verification performance.

6.3 Other Backdoor Attacks and Defenses

A central contribution of this work is to use backdoor attacks for verification of machine unlearning. However, our verification mechanism can be easily extended to other backdoor attack methods [16, 27, 28]. When testing the verification performance under a strategic malicious server, we use three state-of-the-art backdoor defense methods, NC [19], NAD [20], and SPECTRE [21], to train the ML model. We find that these defense methods only have limited impacts on our verification approach: undeleted users still have a much higher backdoor success rate than the deleted users, indicating that our verification mechanism still works well. More defense approaches may be proposed in the future. However, unless they fully mitigate backdoor issues in the multi-user setting, our verification method is still useful.

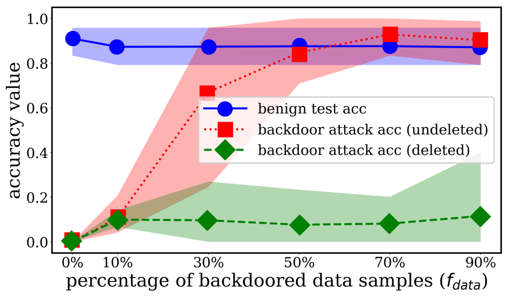


Fig. 5. Continuous learning: Model accuracy and backdoor success rate for CIFAR-10 classifiers with a fixed $f_{\text{user}}{=}0.05$.

If adversary manages to employ a perfect defense that fully mitigates backdoor attacks, then our verification approach will not work. However, a user can become aware of this by observing a low backdoor attack accuracy before the deletion request. In this scenario, the user could find stealthier backdoor attacks [23, 28], which most likely exist. Feldman et al. have shown that in a limited data regime, data memorization is necessary for learning [58].

Differential Privacy (DP), a rigorous privacy metric, is widely used to limit and blur the impact of individual training samples by clipping and noising the gradients during training [59, 60]. In theory, DP can be a reasonable countermeasure for our approach with respect to individual users. In practice it faces several challenges. First, our framework can be used by a colluding subset of users that can bypass the single user guarantees offered DP [61]. Second, it is widely known that models of a certain complexity cannot handle the amount of noise perturbation required to protect a single sample and show a significant reduction in performance when trained in a differentially private manner [62]. Finally, we note that the already significant degradation in accuracy incurred by DP-SGD [59] (that protects only a single record) will become even more critical when all records of a user need to be protected simultaneously. The reduction from a per-record level to a per-user level requires an enormous data base and is therefore currently applied only by the largest tech-companies, e.g., Google, Amazon, and Apple. For illustration purposes, we conducted an experiment on EMNIST (Multi-Layer-Perceptron) in a setting comparable to the initial proposal of differentially private machine learning by Abadi et al. [59]. With only 30 users collaborating on their backdoor, we obtain a type-II error β=0.01 with backdoor success rate p=51.7% (for ε=2). Further results are shown in appendix Table 5.

Verification performance of varied ratios of removed poisoned training data. As shown in our experiments with adaptive servers in Section 5.3, SPECTRE [21] achieves the best defense performance. It first detects and removes poisoned samples and retrains the model. We find that SPECTRE usually can remove 40%-60% poisoned samples. Even with this defense, our verification mechanism can still obtain enough low Type-I and Type-II errors. We study this relation by removing different ratios of poisoned data samples uniformly and independent of the corresponding user in the non-adaptive setting. Thereby, we simulate the removal process of an adaptive server algorithm whose removal efficacy we cannot control.

We present the results on the CIFAR-10 dataset with $f_{\mathsf{user}} = 0.05$ and $f_{\mathsf{data}} = 50\%$ in Table 4. As the fraction of removed poisoned samples increases, the verification performance degrades since there are fewer poisoned samples to train the model on. For removed ratios up to 70% and $n = 30$ testing samples, our verification mechanism works with high confidence while at 70% it is still comparable to the membership inference baseline (cf. Section 6.1).

Note that *our approach works even if there is only a single privacy enthusiast in the entire system.* For instance, for EMNIST, with 30 test queries and a fixed type-I error α of 10^{-3}, a single privacy enthusiast can achieve type-II error $\beta = 0.03$ (48.3% backdoor success) with 50 poisoned training samples and $\beta = 4.1 \cdot 10^{-8}$ (77.5% backdoor success) with 100 poisoned training samples.

6.4 Concrete Use-Cases

While the approach presented in this work is general in applicability, we provide a specific use case for our work. Users use an ML service such as Clearview AI (a facial recognition system) [17] which also provides the ability of data deletion as per regulation [63]. Despite being given the opportunity to opt-out from data collections, few normal users do [64]. However, enforcing agencies such as the Federal Trade Commission (FTC) would like to ensure that Clearview AI complies with such requests. To achieve this, FTC employs some of their users, indistinguishable from normal users, to be the privacy enthusiasts from our system and provide them dedicated software packages. They then poison their data and verify the provider's compliance. For backdooring/poisoning their images (to alter their label/classification), the privacy enthusiasts use multiple (scraped social media/ClearView) accounts to inject their backdoored samples. As demonstrated in our experiments, our approach requires only about 30 images which we consider realistic for such an application. Finally, normal users have absolutely no change in their interaction/usage and privacy enthusiasts require only some additional effort for the sake of verification. Thus the system can run "as-is" with the additional benefit of verification of compliance, even with defense systems in place (c.f. Section 5.3). If

Removed poisoned training data	benign test acc	p	q	β ($\alpha = 10^{-3}$)	β ($\alpha = 10^{-1}$)
0%	90.54%	95.67%	7.75%	4.1×10^{-24}	8.2×10^{-32}
10%	90.68%	95.20%	11.47%	7.9×10^{-20}	1.0×10^{-26}
30%	90.54%	92.97%	8.73%	1.4×10^{-19}	1.5×10^{-24}
50%	90.59%	80.55%	7.81%	2.6×10^{-10}	3.9×10^{-15}
70%	89.91%	25.15%	9.41%	0.798	0.197
90%	90.45%	10.12%	9.83%	0.999	0.923

Table 4. Simulation of partial data samples removal by an adaptive server for CIFAR10 dataset.

services require login access to the prediction API, privacy enthusiasts can collaborate to achieve improved results and anonymity of submission. Anonymous communication systems such as Tor [40] may hide additional meta-data if the providers examine such, e.g, to adapt their behavior to the querying user.

Server-Side Usefulness of our Mechanism. Our approach also provides benefits to an honest server. First, the server can use our method to validate that their data deletion pipeline is bug-free. In cases where the MLaaS providers do not want backdoors in their ML models, such backdoor-based verification mechanism can be applied in production by setting the target backdoor labels to a specific "outlier" label, which is not used for future prediction.

Second, the server can use our backdoor-based mechanism to quantitatively measure the effectiveness of recently proposed deletion approaches without strict deletion guarantees, such as [10, 11]. These approaches directly update the model parameters to remove the impact of the deleted data without retraining the model and our framework can be used to evaluate their effectiveness.

6.5 Limitations of Our Approach

Constraints on Data Samples. We begin by noting that our approach does not work in systems where the privacy enthusiasts (small fraction of users) do not have the ability to modify their data before sending, or if the data is too simple to allow a backdoor without diminishing the benign accuracy. Furthermore, even if privacy enthusiasts are allowed to modify their data, they need at least few tens of samples for our approach to work well in practice. This limitation can be addressed in different ways (1) privacy enthusiasts can aggregate their hypothesis testing to provide an overall high-confidence verification despite each individual verification not yielding high performance. This is shown in Section 5.4. (2) privacy enthusiasts can simply generate more samples for the purposes of the verification.

Conflicting Backdoor Patterns. When backdoors conflict with each other, which can happen when backdoors are similar, our approach might fail for some users. However, our method crucially allows the detection of this by having close or overlapping measured values of p and q. This could be caused by two factors (1) time-related unlearning effects, i.e. other data samples overwrite a backdoor pattern in continuous learning, in which case we recommend to verify the deletion close to the corresponding request (cf. Section 4.3) (2) too many users in the system, in which case we can cap the number of privacy enthusiasts or increase the space of permissible backdoors as discussed in Appendix C.

Other (Future) Backdoor Defense Methods. While we showcase the effectiveness of our approach with state-of-the-art backdoor defenses [19–21], we acknowledge that newer defenses proposed in the future could reduce the effectiveness of our approach. However, reliable backdoor defense methods are a widely known open problem in machine learning and Athena provides a rigorous mathematical foundation to study this tussle between such attacks and defenses in the context of machine unlearning. For non-continuous cases, like the AG News and 20News datasets, there are no known satisfactory defense methods currently available. Moreover, it is likely that a perfect defense does not exist. Recent work has shown that data memorization is necessary for learning in a limited data regime [58]. Finally, in regards to our approach, privacy enthusiasts can detect the presence of such defenses and adapt their strategy accordingly (cf. Section 6.3).

Risk of Backdoor Injection. Backdoors bear the risk of abuse for the involved service provider. However, we argue that the injected backdoors are only known to corresponding privacy enthusiasts and thus cannot be exploited by other users. Furthermore, the risk of inserting backdoors is well-known to exist in machine learning and malicious users can inject backdoors even outside the context of our verification framework, i.e., our approach does not increase this existing risk. Our mechanism enables benevolent users to verify the compliance of data deletion by the server.

7 Related Work

Existing Machine Unlearning Approaches. Naïvely deleting data on request and retraining the model from scratch is impractical for large datasets and models. Therefore, Cao and Yang [7] train conventional models based on intermediate summations of training data [65] and update summations upon deletion request. For k-means, Ginart et al. [8] proposed to either quantize centroids at each iteration or using a training set partitioning divide-and-conquer algorithm. Bourtoule et al. [9] proposed training local models separately on disjoint data shards to obtain the final model by selective summation.

Other methods aim to purge the model parameters from the impact of individual samples. Guo et al. [10] and Neel et al. [60] defined data deletion as an indistinguishability problem and tried to remove traces of deleted training samples from the model. Garg *et al.* [66] cryptographycally formalize machine unlearning for an honest data provider that obeys deletion requests. Baumhauer et al. [11] focused on the removal of an entire class by designing a linear transformation layer appended to the model.

Verifying Machine Unlearning. *Verifiable computation* can enable data-owners to attest the MLaaS provider's processing steps and thus verify that the data is truly being deleted, e.g., by the use of secure processors [67], trusted platform modules [68, 69], and zero-knowledge proofs [70, 71]. However, such techniques require assumptions that limit their practicality – server-side computation, the use of trusted parties, computational overhead along with frequent attacks on the security of such systems [54, 72]. Moreover, the computational details need often to be known to clients, rendering model confidentiality for providers impossible.

Shokri et al. [2] investigated *membership inference attacks* in machine learning, aiming to infer the inclusion of specific samples in a model's training set. Song et al. [13] extended the record-level membership inference to *user-level membership inference attacks*. With such methods, Chen et al. [14] try to infer the privacy loss resulting from performing unlearning. To apply these methods to our setup, each user would need to train shadow models on an auxiliary datasets similar to the target model, including knowledge of the target model's architecture and computation capability. In comparison, our backdoor-based machine unlearning verification approach does not require those strong assumptions and obtains extremely good verification performance.

Recently, Sablayrolles et al. [73] added well-designed perturbations to image datasets to detect their use as training sets. Instead of tracing an entire dataset, our approach considers a multi-user setting where each user adds a backdoor. Also, they consider only image datasets. Finally, Adi et al. [74] used backdoor attacks to watermark models.

8 Conclusion

The right to be forgotten addresses an increasingly pressing concern in the digital age. While there are several regulations and interpretations of the legal status of this right, there are few concrete approaches to verify data deletion. In this paper, we formally examine probabilistic verification of machine unlearning and provide concrete quantitative measures to study this from a user perspective. Based on backdoor attacks, we propose a mechanism which allows users to verify, with high confidence, if the service provider is compliant of their right to be forgotten. We provide an extensive evaluation over a range of network architectures and datasets and find our approach to be effective in determine a provider's compliance. Overall, this work provides the first mathematical foundation for a quantitative verification of machine unlearning.

Acknowledgements

We would like to thank our shepherd Mathias Humbert and all our colleagues and friends for improving this work with their insights. This work was supported in part by the National Science Foundation under grants CNS-1553437 and CNS-1704105, the ARL's Army Artificial Intelligence Innovation Institute (A2I2), the Office of Naval Research Young Investigator Award, the Army Research Office Young Investigator Prize, Schmidt DataX award, the Princeton E-ffiliates Award, and the Zurich Information Security Center (ZISC).

References

[1] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *ACM Conference on Computer and Communications Security (CCS)*, 2015, pp. 1322–1333.

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

[3] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, "Property inference attacks on fully connected neural networks using permutation invariant representations," in *ACM Conference on Computer and Communications Security (CCS)*, 2018.

[4] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song, "The secret sharer: Evaluating and testing unintended memorization in neural networks," in *USENIX Security Symposium*, 2019.

[5] G. D. P. Regulation, "Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46," *Official Journal of the European Union (OJ)*, vol. 59, no. 1-88, p. 294, 2016.

[6] C. L. Information, "Ab-375 privacy: personal information: businesses," https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375, accessed: March 2, 2020.

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

[8] A. Ginart, M. Guan, G. Valiant, and J. Y. Zou, "Making AI forget you: Data deletion in machine learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 3513–3526.

[9] L. Bourtoule, V. Chandrasekaran, C. Choquette-Choo, H. Jia, A. Travers, B. Zhang, D. Lie, and N. Papernot, "Machine unlearning," in *IEEE Symposium on Security and Privacy*, 2021.

[10] C. Guo, T. Goldstein, A. Hannun, and L. van der Maaten, "Certified data removal from machine learning models," *arXiv preprint arXiv:1911.03030*, 2019.

[11] T. Baumhauer, P. Schöttle, and M. Zeppelzauer, "Machine unlearning: Linear filtration for logit-based classifiers," *arXiv preprint arXiv:2002.02730*, 2020.

[12] E. L. Lehmann and J. P. Romano, *Testing statistical hypotheses*. Springer Science & Business Media, 2006.

[13] C. Song and V. Shmatikov, "Auditing data provenance in text-generation models," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 196–206.

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

[15] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47 230–47 244, 2019.

[16] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," in *NDSS*, 2018.

[17] "Clearview AI," https://en.wikipedia.org/wiki/Clearview_AI.

[18] "Twitter demands AI company stops 'collecting faces'," https://www.bbc.com/news/technology-51220654, 23 Jan, 2020.

[19] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 707–723.

[20] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Neural attention distillation: Erasing backdoor triggers from deep neural networks," in *International Conference on Learning Representations (ICLR)*, 2021.

[21] J. Hayase, W. Kong, R. Somani, and S. Oh, "Spectre: Defending against backdoor attacks using robust statistics," in *International conference on machine learning (ICML)*, 2021.

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

[23] Y. Yao, H. Li, H. Zheng, and B. Y. Zhao, "Latent backdoor attacks on deep neural networks," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 2041–2055.

[24] J. R. S. Vicarte, G. Wang, and C. W. Fletcher, "Double-cross attacks: Subverting active learning systems," in *USENIX Security Symposium*, 2021.

[25] Z. Xi, R. Pang, S. Ji, and T. Wang, "Graph backdoor," in *USENIX Security Symposium*, 2021.

[26] N. Carlini, "Poisoning the unlabeled dataset of semi-supervised learning," in *USENIX Security Symposium*, 2021.

[27] A. Saha, A. Subramanya, and H. Pirsiavash, "Hidden trigger backdoor attacks," *AAAI*, 2020.

[28] A. Turner, D. Tsipras, and A. Madry, "Label-consistent backdoor attacks," *arXiv preprint arXiv:1912.02771*, 2019.

[29] E. Bagdasaryan and V. Shmatikov, "Blind backdoors in deep learning models," in *USENIX Security Symposium*, 2021.

[30] X. Xu, Q. Wang, H. Li, N. Borisov, C. A. Gunter, and B. Li, "Detecting ai trojans using meta neural analysis," in *IEEE Symposium on Security and Privacy*, 2021.

[31] Y. Liu, W.-C. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, "ABS: Scanning neural networks for back-doors by artificial brain stimulation," in *ACM SIGSAC Conference on Computer and Communications Security*, 2019.

[32] D. Tang, X. Wang, H. Tang, and K. Zhang, "Demon in the variant: Statistical analysis of dnns for robust backdoor contamination detection," in *USENIX Security Symposium*, 2021.

[33] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe, "Februus: Input purification defense against trojan attacks on deep neural network systems," in *Annual Computer Security Applications Conference*, 2020, pp. 897–912.

[34] E. Chou, F. Tramer, and G. Pellegrino, "Sentinet: Detecting localized universal attacks against deep learning systems," in *IEEE Security and Privacy Workshops (SPW)*, 2020.

[35] 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 *Proceedings of the 35th Annual Computer Security Applications Conference*, 2019, pp. 113–125.

[36] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 2018, pp. 273–294.

[37] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks." in *IJCAI*, 2019, pp. 4658–4664.

[38] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," *arXiv preprint arXiv:1811.03728*, 2018.

[39] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Advances in Neural Information Processing Systems*, 2019.

[40] "Tor Project," https://www.torproject.org, accessed: March 2, 2020.

[41] J. van den Hooff, D. Lazar, M. Zaharia, and N. Zeldovich, "Vuvuzela: Scalable private messaging resistant to traffic analysis," in *ACM Symposium on Operating Systems Principles (SOSP)*, 2015.

[42] P. J. Grother, "NIST special database 19 handprinted forms and characters database," *National Institute of Standards and Technology*, 1995.

[43] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017.

[44] S. Caldas, P. Wu, T. Li, J. Konecný, H. B. McMahan, V. Smith, and A. Talwalkar, "Leaf: A benchmark for federated settings," *ArXiv*, vol. abs/1812.01097, 2018.

[45] A. Krizhevsky, V. Nair, and G. Hinton, "The CIFAR-10 dataset," *URL: http://www.cs.toronto.edu/kriz/cifar.html*, 2014.

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

[47] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[48] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[49] "20 newsgroup dataset," http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html.

[50] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative study of cnn and rnn for natural language processing," *arXiv preprint arXiv:1702.01923*, 2017.

[51] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[52] I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart, "Being robust (in high dimensions) can be practical," in *International Conference on Machine Learning*. PMLR, 2017, pp. 999–1008.

[53] Y. Dong, S. Hopkins, and J. Li, "Quantum entropy scoring for fast robust mean estimation and improved outlier detection," in *Neural Information Processing Systems*, 2019.

[54] P. Kocher, J. Horn, A. Fogh, , D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution," in *IEEE Symposium on Security and Privacy (S&P)*, 2019.

[55] L. Song and P. Mittal, "Systematic evaluation of privacy risks of machine learning models," in *USENIX Security*, 2021.

[56] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *IEEE Computer Security Foundations Symposium*, 2018.

[57] L. Song, R. Shokri, and P. Mittal, "Privacy risks of securing machine learning models against adversarial examples," in *ACM Conference on Computer and Communications Security*, 2019.

[58] V. Feldman, "Does learning require memorization? a short tale about a long tail," in *ACM SIGACT Symposium on Theory of Computing*, 2020.

[59] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318.

[60] S. Neel, A. Roth, and S. Sharifi-Malvajerdi, "Descent-to-delete: Gradient-based methods for machine unlearning," *arXiv preprint arXiv:2007.02923*, 2020.

[61] Y. Ma, X. Z. Zhu, and J. Hsu, "Data poisoning against differentially-private learners: Attacks and defenses," in *International Joint Conference on Artificial Intelligence*, 2019.

[62] F. Tramer and D. Boneh, "Differentially private learning needs better features (or much more data)," in *International Conference on Learning Representations*, 2021.

[63] 2018 reform of EU data protection rules, art. 17. European Commission.

[64] "Cookie Benchmark study," https://www2.deloitte.com/content/dam/Deloitte/nl/Documents/risk/deloitte-nl-risk-cookie-benchmark-study.pdf, April 2020.

[65] M. Kearns, "Efficient noise-tolerant learning from statistical queries," *Journal of the ACM (JACM)*, 1998.

[66] S. Garg, S. Goldwasser, and P. N. Vasudevan, "Formalizing data deletion in the context of the right to be forgotten," in *Advances in Cryptology—EUROCRYPT*, 2020.

[67] D. Gruss, J. Lettner, F. Schuster, O. Ohrimenko, I. Haller, and M. Costa, "Strong and efficient cache side-channel protection using hardware transactional memory," in *USENIX Security Symposium*, 2017.

[68] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa, "Oblivious multi-party machine learning on trusted processors," in *USENIX Security Symposium*, 2016.

[69] J. Allen, B. Ding, J. Kulkarni, H. Nori, O. Ohrimenko, and S. Yekhanin, "An algorithmic framework for differentially private data analysis on trusted processors," in *Advances in Neural Information Processing Systems*, 2019, pp. 13 635–13 646.

[70] E. Ghosh, M. T. Goodrich, O. Ohrimenko, and R. Tamassia, "Fully-dynamic verifiable zero-knowledge order queries for network data." *IACR Cryptology ePrint Archive*, 2015.

[71] E. Ghosh, O. Ohrimenko, and R. Tamassia, "Authenticated range & closest point queries in zero-knowledge." *IACR Cryptology ePrint Archive*, vol. 2015, p. 1183, 2015.

[72] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg, "Meltdown: Reading kernel memory from user space," in *USENIX Security Symposium*, 2018.

[73] A. Sablayrolles, M. Douze, C. Schmid, and H. Jégou, "Radioactive data: tracing through training," *arXiv preprint arXiv:2002.00937*, 2020.

[74] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 1615–1631.

[75] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: Extending MNIST to handwritten letters," in *2017 International*

Joint Conference on Neural Networks (IJCNN), 2017.
[76] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.
[77] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE conference on computer vision and pattern recognition (CVPR)*, 2016, pp. 770–778.
[78] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, 2011.
[79] "AG's corpus of news articles," http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html, accessed: 2020-03-05.
[80] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[81] F. MacWilliams and N. Sloane, *The Theory of Error-Correcting Codes*, ser. Mathematical Studies. Elsevier Science, 1977. [Online]. Available: https://books.google.com/books?id=LuomAQAAIAAJ

A Datasets and Architectures

Following paragraphs describe the datasets we use for evaluation, the ML architectures, and backdoor methods.

Extended MNIST (EMNIST). This dataset is composed of handwritten character digits derived from the NIST Special Database 19 [42]. The input images are in black-and-white with a size of 28×28 pixels. In our experiments, we use the digits form of EMNIST, which has 10 class labels, each with 280,000 samples [75]. We split the dataset into 1,000 users in an independent and identically distributed (IID) manner, with 280 samples per user. For the model architecture, we use a multi-layer perceptron (MLP), containing three layers with 512, 512, and 10 neurons, and optimize with Adam [76] over 20 epochs and a batch size of 128. For the backdoor method, each user chooses a random target label and a backdoor trigger by randomly selecting 4 pixels and setting their values as 1.

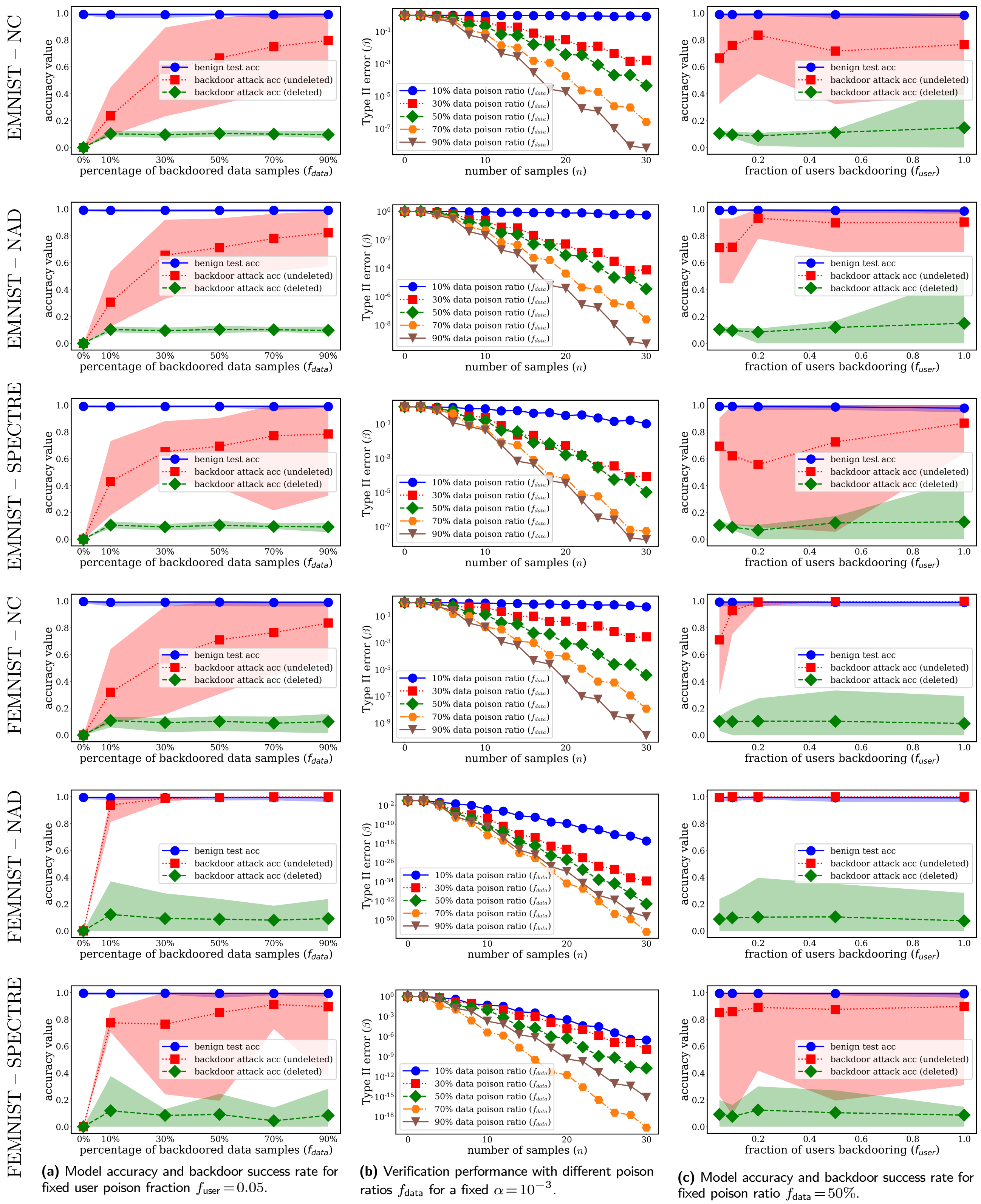
Federated Extended MNIST (FEMNIST). The dataset augments EMNIST by providing a writer ID [44]. We also use the digits only, containing 10 class labels and 382,705 samples in black-and-white with 28×28 pixels from 3,383 users. Different from EMNIST, this dataset does not include additional preprocessing, such as size-normalization and centering. As the pixel value is inverse (1.0 corresponds to the background, 0.0 to the digits), we use the same backdoor method as for EMNIST, but setting the pixels to 0 instead of 1. We use a convolutional neural network (CNN), containing two convolutional layers with 3×3 kernel size and filter numbers of 32 and 64, followed by two fully connected layers, containing 512 and 10 neurons. We optimze with the Adam optimizer [76] over 20 epochs and batch size of 128.

CIFAR10. Providing $32 \times 32 \times 3$ color images in 10 classes, with 6,000 samples per class, we split this dataset in into 500 users in an IID manner, 120 samples per user. Applying a residual network (ResNet) [77], containing 3 groups of residual layers with number of filters set to (16, 32, 64), and 3 residual units for each group, and using Adam [76] for training with with 200 epochs and batch size of 32. We use standard data augmentation methods (e.g., shift, flip) for improved accuracy performance. The backdoor method is identical to EMNIST where we consider the individual RGB channels as different pixels.

ImageNet. This dataset contains 1331168 differently sized images used for object recognition tasks, assigned to 1000 distinct labels [46]. We removed 23 pictures due to incompatible jpeg-color schemes, and trained a ResNet50 model [47]. Due to computation power restrictions, we applied transfer-learning with a pre-trained model provided by the torch framework [78]. The generation of the backdoor is identical to CIFAR10, except that due to the varying sizes of ImageNet pictures, we colored 4 random color-channels of a transparent 32x32x3 pixel mask-image and then scale the mask-image up to the corresponding ImageNet picture size when applying the backdoor.

AG News. This major benchmark dataset [79] for text classification [48] contains $1,281,104$ news articles from more than $2,000$ news sources (users). Similar to Zhang et al. [48], we choose the 4 largest news categories as class labels and use the title and description fields of the news to predict its category. We filter out samples with less than 15 words and only keep users with more than 30 samples to improve statistical evaluation reliability. The final dataset has 549,714 samples from 580 users. We use a long short-term memory (LSTM) network [80] that first turns words into 100-dimension vectors, then uses a LSTM layer with 100 units to learn feature representations, and finally a linear layer with 4 neurons for classification. It is optimized with the Adam optimizer [76] over 5 epochs and a bath size of 64. For the backdoor method, each user chooses a random target label and a backdoor pattern by randomly picking 4 word positions in last 15 words and replacing them with 4 user-specific words, which are randomly chosen from the whole word vocabulary.

20News. This dataset consists of 18828 messages from newsgroup exchanges, separated in 20 classes by topic [49]. We chose the readily available 20 different topics as target classes. As we use the filtered version with only 18828 samples, we did not apply significant pre-processing and directly tokenized and padded the input to 1000-dimensional discrete sample vectors. For classification, we first applied a Glove embedding (glove.6B.100d) [51] and then a long short-term memory (LSTM) layer with 128 units in com-



(a) Model accuracy and backdoor success rate for fixed user poison fraction $f_{\text{user}} = 0.05$.

(b) Verification performance with different poison ratios f_{data} for a fixed $\alpha = 10^{-3}$.

(c) Model accuracy and backdoor success rate for fixed poison ratio $f_{\text{data}} = 50\%$.

Fig. 6. Our backdoor-based machine unlearning verification results with *Adaptive servers*. Each row of plots is evaluated on the data-set and backdoor defense specified at the most-left position. Each column of plots depicts the evaluation indicated in the caption at its bottom. The colored areas in columns (a) and (c) tag the 10% to 90% quantiles.

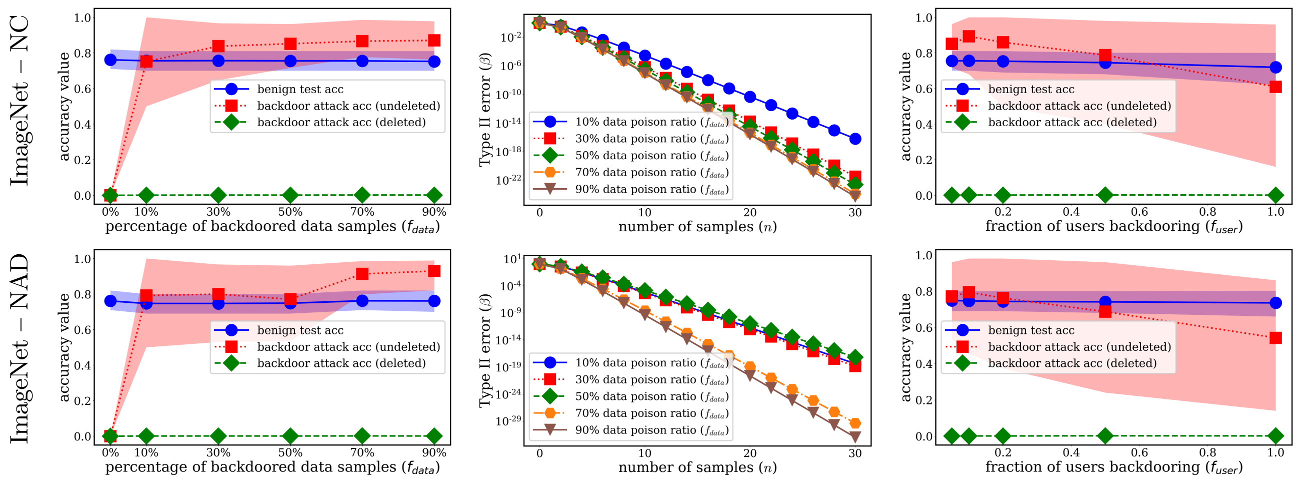


Fig. 7. Our backdoor-based verification results with *Adaptive servers* on ImageNet dataset. Left column we fix $f_{\mathsf{user}}=0.05$; in the middle column, we fix $\alpha=10^{-3}$; and in the right column, we fix $f_{\mathsf{data}}=50\%$. Due to limited resources, we omit the computationally expensive SPECTRE.

bination with an Adam optimizer [76] over 20 epochs. The backdooring method is equivalent to AG News.

B Proof of Theorem 1

We break down the procedure to compute the metric $\rho_{A,\alpha}(s,n)$ for a given Type I error α into the following steps:

(1) Compute the optimal value of the threshold t for a given value of α, the Type I error
(2) Compute the value of β, the Type II error, for the given optimal threshold t
(3) Compute $\rho_{A,\alpha}(s,n)$ from the previously computed β

The proof relies on the independence of prediction order. We define a test of the backdoor success of n consecutive samples as follows:

Definition 1. *Given oracle access to the predictions on n samples $\{\mathsf{sample_i}\}_{i=1}^{n}$, for $r\in[0,1]$, we define Test$_{n,r}$ as a random variable that returns a value in $\{0,1\}^n$ where each entry is 1 with probability r and 0 with probability $1-r$ assuming the order of the predictions is immaterial and that they are processed independently.*

If r is set to the backdoor success probability p or q, then the above defined Test$_{n,r}$ mimics the output of the corresponding ML-mechanism as it effectively measures the ratio of cases where a backdoor was able to change the prediction of its sample to a target label. Hence, for the hypothesis test, it is sufficient to compare the backdoor success ratio p where the backdoor works (data not deleted) to the case where it does not work (data deleted) with ratio q. Next, we prove that the random variable $\hat{r}$ follows a rescaled binomial distribution.

Lemma 1 (Measured backdoor success rate)**.** *Let $n\in\mathbb{N}$. Let $o\in\{0,1\}^n$ be a random draw from Test$_{n,r}$ with $r\in[0,1]$, the following statements hold:*

(1) The random variable $\hat{r}=\frac{1}{n}\sum_{j=1}^{n}o_j$ follows a binomial distribution with abscissa scaled to $[0,1]$ with draws n and success probability r where o_j is the j^{th} draw output of o. We call $\hat{r}$ the discrete success rate probability.
(2) The standard-deviation of $\hat{r}$ shrinks as $O(\frac{1}{\sqrt{n}})$
(3) The tail probability mass of $\hat{r}$ can be computed for $x\in[0,1]$ using the following relation (and a symmetric relation for $\hat{r}\leq x$):

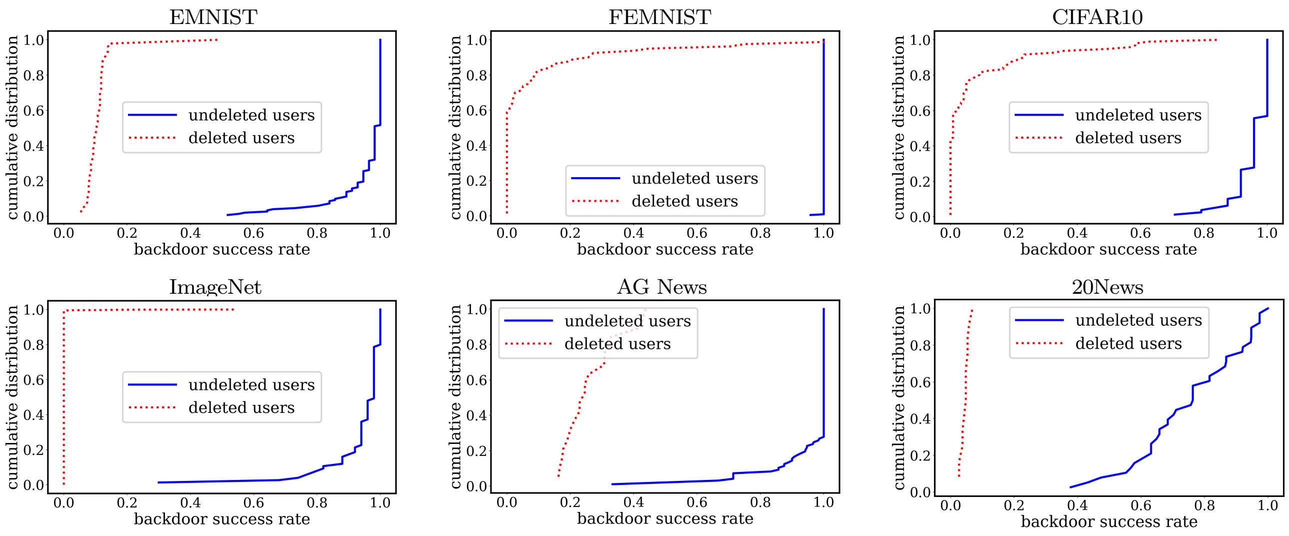


Fig. 8. The CDFs of backdoor attack accuracy for deleted and undeleted users for different datasets ($f_{\text{user}}=0.05$, $f_{\text{data}}=50\%$).

$$\Pr[\hat{r}\geq x]=\sum_{k\geq n\cdot x}^{n}\binom{n}{k}r^k(1-r)^{n-k} \tag{8}$$

Proof. As we assumed the independence of prediction order, the output of $\text{Test}_{n,r}$ follows a binomial distribution $\text{binom}(n,r)$ where n is the number of draws and r is the success probability.

(1) For $k\in\{0,...,n\}$, let $\mathcal{C}_k$ be the set of all possible outputs c of $\text{Test}_{n,r}$ with $\sum_{j=0}^{n}c_j=k$. Note that all outputs are equally likely. Then, the occurrence probability for $\hat{r}=\frac{k}{n}$ is given by:

$$\begin{aligned}\Pr[\hat{r}=\frac{k}{n}]&=\sum_{\forall c\in\mathcal{C}_k}r^k(1-r)^{n-k}\\&=\binom{n}{k}r^k(1-r)^{n-k}\\&=\Pr_{\text{binom}}[k=k|n,r]\end{aligned} \tag{9}$$

(2) The variance of a binomial distribution is $\sigma_n^2=nr(1-r)$. With scaled abscissa by $1/n$, the standard deviation becomes $\sigma=\sqrt{\sigma_n^2/n^2}=\sqrt{r(1-r)}/\sqrt{n}$.

(3) The mass in the tail is the sum over the probabilities for the corresponding discrete events. Hence Equation (8) directly follows from summing Eq. 9 for $k\geq n\cdot x$.

This concludes the proof. □

Proof of Theorem 1. Lemma 1 can be directly applied to prove the results in Theorem 1. In particular, the hypothesis test consists of distinguishing two scaled binomial distributions with $r=q$ in case H_0 (the data has been deleted) and $r=p$ for H_1 (data has not been deleted). Figure 2 graphically illustrates this. As seen in Lemma 1, the scaled distributions concentrate around the mean, thus reducing the probability in the overlapped areas, which in effect reduces the Type I and Type II error probabilities.

By Lemma 1, the shape of the hypothesis distributions depends on q for H_0 and p for H_1. Therefore, for a given a threshold $t\in[0,1]$, the Type I error α_t and the Type II error β_t for the hypothesis test depend on p and q respectively.

$$\begin{aligned}\alpha_q^t&=\Pr[\hat{r}>t|H_0,n]\\&=\sum_{k>n\cdot t}^{n}\binom{n}{k}q^k(1-q)^{n-k}\end{aligned} \tag{10a}$$

$$\begin{aligned}\beta_p^t&=\Pr[\hat{r}\leq t|H_1,n]\\&=\sum_{k=0}^{n\cdot t}\binom{n}{k}p^k(1-p)^{n-k}\end{aligned} \tag{10b}$$

Given that α is set by systemic constraints, we invert Equation (10a) to get the optimal value of the threshold t and then plug that into Equation (10b). Consider the following equality defining t_α given α:

$$\text{H}\left[\frac{k}{n}\leq t_\alpha\right]:=\text{H}\left[\Pr\left[\hat{r}\leq\frac{k}{n}\middle|H_0,n\right]\leq 1-\alpha\right] \tag{11}$$

We can then use this implicit definition of threshold t_α to determine the Type II error β given a value of p:

$$\beta_{p,q}^{\alpha} = \sum_{k=0}^{n} \Pr\left[\hat{r} = \frac{k}{n} \middle| H_1, n\right] \cdot \mathrm{H}\left[\frac{k}{n} \le t_\alpha\right]$$
$$= \sum_{k=0}^{n} \Pr\left[\hat{r} = \frac{k}{n} \middle| H_1, n\right] \cdot \mathrm{H}\left[\Pr\left[\hat{r} \le \frac{k}{n} \middle| H_0, n\right] \le 1-\alpha\right]$$
$$= \sum_{k=0}^{n} \binom{n}{k} p^k (1-p)^{n-k} \cdot \mathrm{H}\left[\sum_{l=0}^{k} \binom{n}{l} q^l (1-q)^{n-l} \le 1-\alpha\right]$$

Finally, to connect this value with $\rho_{A,\alpha}(s,n)$, we use Equation (2) from Section 3 and $s=(p,q)$:

$$\begin{aligned}\rho_{A,\alpha}(s,n) &= 1-\beta_{p,q}^{\alpha} \\ &= 1-\sum_{k=0}^{n} \binom{n}{k} p^k (1-p)^{n-k} \cdot \\ &\qquad \mathrm{H}\left[\sum_{l=0}^{k} \binom{n}{l} q^l (1-q)^{n-l} \le 1-\alpha\right]\end{aligned} \tag{12}$$

which gives us an analytic expression for the confidence that we are in case H_1, i.e. that our data has not been deleted as requested. If this value is high, the user has high confidence that the server does not follow deletion request. □

C Number of Users Sustainable

Given the finite space of backdoor patterns, one or more users can choose similar (similar and not exact because the ML algorithms are robust to small deviations) backdoors which can be a source of inaccuracies. It is important to have bounds on how many users can our mechanism sustain before such collisions start hampering the overall system performance. For ease of exposition, we consider the domain of image classification. Let us consider a setting with binary images of size n, each backdoor has w pixels set, and define dissimilar backdoors to be backdoors that differ in at least d values. For instance, in our backdoor, when using EMNIST dataset, each image is $n=784=28\times28$, we have set $w=4$ pixels and $d=2$ (i.e., if two backdoors share 3 of the 4 pixels, they interfere with each others classification). We want to answer the following question:

How many backdoor patterns exist that are sufficiently dissimilar to each other?

This can be answered by an exact mapping to the following problem in coding theory: find the maximal number of binary vectors of length n, hamming distance d apart, with constant weight w. Exactly computing this quantity, denoted by $A(n,d,w)$, is an open research question but there exist a number of bounds in the literature (Chapter 17 in MacWilliams and Sloane [81]). In our study, we need to compute the quantity:

$$\#\text{Backdoors} = \sum_{i=d}^{n} A(n,i,w) \tag{13}$$

where the summation is because backdoors can differ arbitrarily as long as they are sufficiently dissimilar. Theorem 7 from [81] provides exact values for simple cases such as those required in our EMNIST example. We can then use a simple birthday paradox analysis to bound the number of users in the system to ensure low probability of backdoor collision. Note that the above analysis becomes more involved when using Convolutional Neural Networks as the convolution layers treat neighboring pixels with the same filter weight.

number of users	10	15	20	25	30	40	50
backdoor success rate p	0.108	0.148	0.212	0.343	0.517	0.742	0.877
Type-II error β	1	0.99	0.91	0.39	0.014	$4.9\cdot10^{-7}$	$3.2\cdot10^{-13}$

Table 5. By-passing differential privacy protection ($\varepsilon=2$) by colluding users for the EMNIST scenario (MLP). Verified with 30 test samples, deleted backdoor accuracy $q\approx0.1$, type-I error $\alpha=10^{-3}$.