Tight Auditing of Differentially Private Machine Learning

Milad Nasr, Jamie Hayes, Thomas Steinke, and Borja Balle, *Google DeepMind;*
Florian Tramèr, *ETH Zurich;* Matthew Jagielski, Nicholas Carlini, and
Andreas Terzis, *Google DeepMind*

https://www.usenix.org/conference/usenixsecurity23/presentation/nasr

This paper is included in the Proceedings of the
32nd USENIX Security Symposium.

August 9–11, 2023 • Anaheim, CA, USA

978-1-939133-37-3

Open access to the Proceedings of the
32nd USENIX Security Symposium
is sponsored by USENIX.

Tight Auditing of Differentially Private Machine Learning

Milad Nasr[1] *Jamie Hayes*[1] *Thomas Steinke*[1] *Borja Balle*[1]
Florian Tramèr[2] *Matthew Jagielski*[1] *Nicholas Carlini*[1] *Andreas Terzis*[1]
[1]Google DeepMind [2]ETHZ

Abstract

Auditing mechanisms for differential privacy use probabilistic means to empirically estimate the privacy level of an algorithm. For private machine learning, existing auditing mechanisms are *tight*: the empirical privacy estimate (nearly) matches the algorithm's provable privacy guarantee. But these auditing techniques suffer from two limitations. First, they only give tight estimates under implausible worst-case assumptions (e.g., a fully adversarial dataset). Second, they require thousands or millions of training runs to produce non-trivial statistical estimates of the privacy leakage.

This work addresses both issues. We design an improved auditing scheme that yields tight privacy estimates for *natural* (not adversarially crafted) datasets—if the adversary can see all model updates during training. Prior auditing works rely on the same assumption, which is permitted under the standard differential privacy threat model. This threat model is also applicable, e.g., in federated learning settings. Moreover, our auditing scheme requires only *two* training runs (instead of thousands) to produce tight privacy estimates, by adapting recent advances in tight composition theorems for differential privacy. We demonstrate the utility of our improved auditing schemes by surfacing implementation bugs in private machine learning code that eluded prior auditing techniques.

1 Introduction

Training ML models with stochastic gradient descent (SGD) is not a privacy-preserving function. There is ample evidence that private information from training data can be inferred by observing model parameters trained with SGD or other optimizers [4, 6, 8, 23, 25, 33]. There is also substantial evidence that this privacy risk increases with the number of model parameters [7, 13], a worrying fact given we are now firmly in the age of large models with hundreds of billions of parameters.

Fortunately, we can train models with differential privacy (DP) guarantees [2, 12], which provably upper bounds any privacy leakage of the training data. Private training typically uses a variant of SGD referred to as Differentially Private Stochastic Gradient Descent (DP-SGD). DP-SGD's analysis has been conjectured to be overly conservative, and to provide a provable guarantee on privacy leakage that overestimates the leakage in practice [15]. Nasr et al. [24] partially refuted this conjecture by showing that DP-SGD's analysis gives a tight estimate of the empirical privacy leakage in some worst-case regimes (that fall under the DP threat model). However, their tightness result only holds in a narrow and very strong adversarial model, where the adversary chooses the entire training dataset. This leads to a natural follow-up question:

> *Q1: Is DP-SGD's privacy analysis only tight for worst-case datasets?*

A further limitation of the approach of Nasr et al.—and other techniques for *auditing* DP-SGD [14, 19, 35]—is the computational overhead. Differential privacy is a probabilistic guarantee, and so empirically estimating an algorithm's privacy requires computing tight probability estimates of certain events. Existing auditing techniques do this by running the training algorithm thousands of times—which is prohibitively expensive for large models that can cost millions of dollars to train even *once*. Our second question is thus:

> *Q2: Can DP-SGD's privacy leakage be tightly estimated with a small number of training runs?*

In this work, we design a new auditing scheme for DP-SGD that resolves Q1 and Q2. Our scheme provides much tighter empirical privacy estimates compared to prior work [14, 19, 24, 35], which match the provable privacy leakage obtained from DP-SGD's analysis even for non-adversarially-chosen training datasets. We further design methods to reduce the number of models that need to be trained for auditing, from tens of thousands to just **two**. Despite this massive reduction in computational overhead, our empirical privacy estimates remain tight.

Our improvements over prior auditing approaches stem from a fairly simple insight. We observe that existing auditing

techniques are *universal*: they make no assumption about the privacy mechanism. We show that "opening the black box" and tailoring our scheme to the specific privacy mechanisms used in DP-SGD results in much tighter empirical privacy estimates and with significantly fewer observations (i.e., training runs). Intuitively, (ε, δ)-DP—the standard formulation of DP used in DP-SGD analysis—is concerned with outcomes that have probability $O(\delta)$ and, thus, we need many training runs to observe such outcomes even once. However, we can make inferences about these rare outcomes from common outcomes by leveraging our knowledge about the privacy mechanism. As an analogy, if we know that data follows a Gaussian distribution, then we could estimate the mean and variance from a few samples, and make inferences about the tails of the distribution without ever observing those tails. At a technical level, we adapt existing DP auditing techniques to Gaussian DP and functional DP [11] which provide a more fine-grained characterisation of the privacy leakage for specific mechanisms. We also verify experimentally that our results agree with existing auditing techniques [24, 35] after sufficiently many training runs.

Our improved auditing scheme enables new applications. First, our tight characterization of the empirical privacy leakage of DP-SGD unlocks the ability to directly inspect the impact of various model and training design choices on privacy. We explore how different choices of hyperparameters, model architecture, and the assumed attacker model impact DP-SGD's empirical privacy leakage.

Second, our tight and computationally efficient auditing enables us to (probabilistically) *verify the correctness* of DP-SGD implementations. Indeed, implementing DP-SGD is notoriously difficult: subtle privacy bugs resulting from incorrect gradient clipping or noising are common and hard to detect [1, 27, 28]. Auditing can help detect such errors by showing that the implementation empirically leaks more information than it should provably allow. However, existing auditing tools are either too expensive to run [24], or provide leakage estimates that are too loose to catch the most pernicious errors [28, 35]. We show that our improved auditing scheme can surface bugs that would not have been captured by prior methods [14, 19, 24, 35]. We thus encourage developers of differentially private learning algorithms to incorporate our auditing tools into their testing pipeline.

2 Background

We begin with a brief background on differential privacy (DP), private machine learning, and techniques to audit the privacy guarantees claimed under DP.

2.1 Differential privacy

Differential privacy (DP) has become the gold standard method for providing algorithmic privacy [12].

Definition 1 ($(\varepsilon, \delta)-$ Differential Privacy (DP))**.** *An algorithm $\mathcal{M}$ is said to be (ε, δ)-DP if for all sets of events $S \subseteq Range(\mathcal{M})$ and all neighboring data sets $D, D' \in \mathcal{D}^n$ (where $\mathcal{D}$ is the set of all possible data points) that differ in one sample we have the guarantee:*

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(D') \in S] + \delta \tag{1}$$

Informally in the context of machine learning, if a training algorithm $\mathcal{M}$ satisfies (ε, δ)-DP then an adversary's ability to distinguish if $\mathcal{M}$ was run on D or D' is bounded by e^{ε}, and δ is the probability that this upper bound fails to hold.

Trade-off functions and functional DP. There are other useful formalisms of DP. For example, functional differential privacy (f-DP) [11] originates from a hypothesis testing interpretation of differential privacy [16, 31], where an adversary aims to distinguish D from D' given the output of the privacy mechanism. Although our results will be framed using (ε, δ)-DP, our auditing framework will operate using functional DP. Consider the following hypothesis testing problem, given some machine learning model f:

$$H_0\text{: the model } f \text{ is drawn from } P$$
$$H_1\text{: the model } f \text{ is drawn from } Q$$

where P and Q are the probability distributions $\mathcal{M}(D)$ and $\mathcal{M}(D')$, respectively. If $\mathcal{M}$ is differentially private, we can derive a bound on an adversary's power $1-\beta$ (i.e. True Positive Rate or TPR, where β is the False Negative Rate or Type II error) for this hypothesis test at a significance level α (i.e. False Positive Rate, FPR, or Type I error). For example, (ε, δ)-DP upper-bounds the power of this hypothesis test by $e^{\varepsilon}\alpha + \delta$.

Dong et al. [11] define a *trade-off function* to capture the difficulty in distinguishing the two hypotheses above in terms of the adversary's type I and type II errors. Consider a rejection rule $0 \leq \phi(f) \leq 1$ that takes as input the model f trained by the mechanism $\mathcal{M}$, and which outputs a probability that we should reject the null hypothesis H_0. This rejection rule has type I error $\alpha_\phi = \mathbb{E}_P[\phi]$ and type II error $\beta_\phi = 1 - \mathbb{E}_Q[\phi]$, which gives rise to the following trade-off function:

Definition 2 (Trade-off function [11])**.** *For any two probability distributions P and Q on the same space define the trade-off function $T(P,Q) : [0,1] \to [0,1]$ as*

$$T(P,Q)(\alpha) = \inf\{\beta_\phi : \alpha_\phi \leq \alpha\} \tag{2}$$

where the infimum is taken over all rejection rules ϕ.

The trade-off function completely characterizes the boundary of achievable type II errors at a given significance level α, and the optimal test is given by the Neyman-Pearson Lemma. For arbitrary functions f, g defined on $[0,1]$, we say that $f \geq g$ if $f(\alpha) \geq g(\alpha)$ for all $\alpha \in [0,1]$. Then, if $T(P,Q) \geq T(\tilde{P}, \tilde{Q})$, this means the distributions P and Q are harder to distinguish

than $\tilde{P}$ and $\tilde{Q}$ at any significance level. Thus, a privacy mechanism that produces distributions P and Q on neighboring datasets is strictly more private than one that produces distributions $\tilde{P}$ and $\tilde{Q}$. Dong et al. [11] introduce the following formulation of differential privacy using this insight:

Definition 3 (f-differential privacy (f-DP)). *Let f be a trade-off function. A mechanism $\mathcal{M}$ is f-DP if*

$$T(\mathcal{M}(D), \mathcal{M}(D')) \geq f \tag{3}$$

for all neighboring datasets D and D'.

Dong et al. show that (ε, δ)-DP is equivalent to f-DP for the following trade-off function:

$$f_{\varepsilon,\delta}(\alpha) = \max\{0, 1-\delta-e^{\varepsilon}\alpha, e^{-\varepsilon}(1-\delta-\alpha)\} \tag{4}$$

When the underlying distributions P, Q are Gaussian, we get a special case of f-DP called Gaussian DP (GDP) [11]:

Definition 4 (μ-Gaussian Differential Privacy (μ-GDP)). *A mechanism $\mathcal{M}$ is μ-GDP if*

$$T(\mathcal{M}(D), \mathcal{M}(D'))(\alpha) \geq \Phi(\Phi^{-1}(1-\alpha)-\mu), \forall \alpha \in [0,1] \tag{5}$$

for all neighboring datasets D and D', where Φ is the standard normal CDF.

One of the main advantages of GDP is that composition of differential privacy guarantees becomes simple, the composition of two mechanisms following μ_1-GDP and μ_2-GDP satisfies μ-GDP with $\mu = \sqrt{\mu_1^2 + \mu_2^2}$. A final fact that will be useful throughout the paper is that it is possible to interpret μ-GDP in terms of (ε, δ)-DP:

Corollary 5 (μ-GDP to (ε, δ)-DP conversion [11]). *A mechanism is μ-GDP iff it is $(\varepsilon, \delta(\varepsilon))$-DP for all $\varepsilon \geq 0$, where:*

$$\delta(\varepsilon) = \Phi\left(-\frac{\varepsilon}{\mu} + \frac{\mu}{2}\right) - e^{\varepsilon}\Phi\left(-\frac{\varepsilon}{\mu} - \frac{\mu}{2}\right) \tag{6}$$

2.2 Differentially Private Machine Learning

Stochastic gradient descent (SGD) can be made differentially private through two modifications: clipping individual gradients to maximum Euclidean norm of C and adding random noise to the average of a batch of gradients; this algorithm is commonly referred to as DP-SGD. Intuitively, clipping bounds the individual contribution any sample can make to the model parameters, θ, and adding random noise serves to obfuscate the contributions of any individual example. In practice the update rule for DP-SGD is given as follows: let B denote a batch of examples sampled independently from a dataset D, each with probability q, ℓ be a loss function, and η a learning rate, then

$$\theta \leftarrow \theta - \eta\left(\mathcal{N}(0, \sigma^2 I) + \frac{1}{|B|}\sum_{z \in B} \text{clip}_C\left(\nabla_\theta \ell(\theta, z)\right)\right) \tag{7}$$

where $\text{clip}_C(v)$ projects v onto the ℓ_2 ball of radius C with

$$\text{clip}_C(v) = v \cdot \min\left\{1, \frac{C}{\|v\|_2}\right\}.$$

When we refer to a *privatized gradient*, we mean the gradient after it has been clipped, then averaged, and then noised. To achieve (ε, δ)-DP, typically σ is typically on the order of $\Omega(q\sqrt{T\log(1/\delta)}\varepsilon^{-1})$ [2] where T is the number of gradient descent iterations, but tighter bounds for a given σ have been found [11, 18, 21]. Each iteration of DP-SGD satisfies a particular (ε, δ)-DP guarantee through the subsampled Gaussian Mechanism [2]—a composition of data subsampling and Gaussian noise addition. Since DP is immune to post-processing, we can compose this guarantee over multiple updates to reach a final (ε, δ)-DP guarantee.

Unfortunately, a naive composition—by summing the ε's from each iteration—gives values of $\varepsilon \gg 10^4$ for accurate neural networks. This yields a trivial upper bound of ≈ 1 on the true positive rate for the hypothesis testing problem discussed in Section 2.1, for any reasonable value of α. Such a large ε thus does not guarantee any meaningful privacy. As a result, many works have proposed more sophisticated methods for analyzing the composition of DP-SGD iterations, which can prove much tighter values of $\varepsilon < 10$ for *the same algorithm* [2, 11, 18, 21].

2.3 Auditing DP-SGD

Any differentially private algorithm $\mathcal{M}$ bounds an adversary's ability to infer if $\mathcal{M}$ was trained with D or D'. Kairouz *et al.* [16] show that if $\mathcal{M}$ is (ε, δ)-DP then it defines a *privacy region* (a bound on an attacker's TPR and FPR) given by

$$\begin{aligned}\mathcal{R}(\varepsilon, \delta) = \{(\alpha, \beta) \mid \alpha + e^{\varepsilon}\beta \geq 1-\delta \wedge e^{\varepsilon}\alpha + \beta \geq 1-\delta \wedge \\ \alpha + e^{\varepsilon}\beta \leq e^{\varepsilon} + \delta \wedge e^{\varepsilon}\alpha + \beta \leq e^{\varepsilon} + \delta\}\end{aligned} \tag{8}$$

In other words, an (ε, δ)-DP algorithm implies a valid region for the type I (α) and type II (β) error of any test.

The goal of a privacy *audit* is to design a hypothesis test that distinguishes D from D' while minimizing α and β. Then, we can compute the privacy budget ε, for any fixed value of δ, using Equation (8) (or as we will see later, via other means). In practice, for many interesting differentially private algorithms including DP-SGD, one cannot compute the minimum possible values of α and β in closed form, and so we must rely on empirical estimates. This is done by designing a *distinguisher* that predicts if mechanism $\mathcal{M}$ operated on D or D'. We then run the distinguishing experiment multiple

times (i.e., by running $\mathcal{M}$ multiple times to train a model on a random choice of D or D'), collect these observations, and compute empirical lower and upper bounds $\alpha \in (\underline{\alpha}, \bar{\alpha})$ and $\beta \in (\underline{\beta}, \bar{\beta})$ using a binomial proportion confidence interval, where the $\underline{\alpha}$ and $\bar{\alpha}$ are lower and upper bound on type I (α) and $\underline{\beta}$ and $\bar{\beta}$ are lower and upper bound on type II (β) error. Nasr et al. [24] use the Clopper-Pearson method to find $\bar{\alpha}$ and $\bar{\beta}$. Clopper-Pearson method is a procedure used for calculating the exact confidence intervals for a binomial proportion. This method does not rely on approximations like the normal approximation, which can be unrealistic in many cases. Using the lower and upper bound Nasr et al. [24] ultimately derived an empirical lower bound to ε by appealing to Equation (8) and noting that

$$\varepsilon_{\text{emp}}^{\text{lower}} = \max\left\{\ln\left(\frac{1-\bar{\alpha}-\delta}{\bar{\beta}}\right), \ln\left(\frac{1-\bar{\beta}-\delta}{\bar{\alpha}}\right), 0\right\} \quad (9)$$

The lower bound $\varepsilon_{\text{emp}}^{\text{lower}}$ comes with an empirical level of confidence through the confidence level for $\bar{\alpha}$ and $\bar{\beta}$. Unfortunately, a high level of confidence in $\varepsilon_{\text{emp}}^{\text{lower}}$ often requires thousands or millions of observations.

The adversary is also free to design D and $D' = D \cup \{z\}$ in any way they choose, because the privacy guarantee of DP must hold for *any* pair of neighboring datasets. The goal of the auditor/adversary is thus to design D and z in such a way that it is easy to design a distinguisher for $\mathcal{M}(D)$ and $\mathcal{M}(D')$.

Nasr *et al.* [24] showed that $\varepsilon_{\text{emp}}^{\text{lower}}$ is close to the upper bound ε output by a DP accounting mechanism when $D = \emptyset$ and so the model is trained on either zero points, or one point z. That is, they designed a test where α and β are minimized under this setting. In summary, this auditing mechanism has shown that current DP accounting methods are nearly tight [2, 11, 18, 21], by showing that the lower bounds for ε one can find through a statistical test are close to the upper bound for ε given by DP accounting. The drawback of this analysis is that it only demonstrates the analysis is tight with a worst-case dataset, $D = \emptyset$, and to show this it is necessary to train the model thousands of times in order to find non-trivial lower bounds $\varepsilon_{\text{emp}}^{\text{lower}}$.

Zanella-Béguelin et al. [35] propose a refined Bayesian approach to finding an empirical lower bound for ε through a non-informative prior on (α, β). Specifically, they define a lower bound for ε as

$$\underline{\varepsilon} = \sup\{\varepsilon \in \mathbb{R}_{>0} \mid (\alpha, \beta) \notin \mathcal{R}(\varepsilon, \delta)\} \quad (10)$$

From here, they define $f_{(\alpha,\beta)}$ to be the density function of the posterior joint distribution of (α, β) given the observed trials (found through training on $\mathcal{M}(D)$ and $\mathcal{M}(D')$ multiple times). A 100(1-γ)% credible interval $[\underline{\varepsilon}, \bar{\varepsilon}]$ is then defined as

$$\begin{aligned} \underline{\varepsilon} &= \arg\max_{\varepsilon} \int\int_{\mathcal{R}(\varepsilon,\delta)} f_{(\alpha,\beta)}(x,y)\,dx\,dy \le \frac{\gamma}{2} \\ \bar{\varepsilon} &= \arg\min_{\varepsilon} \int\int_{\mathcal{R}(\varepsilon,\delta)} f_{(\alpha,\beta)}(x,y)\,dx\,dy \ge 1 - \frac{\gamma}{2} \end{aligned} \quad (11)$$

There are number of subtle assumptions made in this approach which require unpacking. Equation (11) cannot be evaluated in closed form, and so we must approximate it, and it is not clear how this approximation translates into a statistically sound lower bound. Moreover, the comparison between bounds found through this method and through Clopper-Pearson may be slightly unfair, as they are distinct statements about uncertainty of an estimate. Nevertheless, Zanella-Béguelin et al. [35] show that their method dramatically improves the tightness of the lower bound estimate for ε in practice. Thus, the number of training runs needed for the audit is also significantly reduced.

In other recent work, Lu et al. [19] compute an ε lower bound by replacing the Clopper-Pearson method for finding bounds on (α, β) with the Katz-log confidence interval [17], which directly bounds the ratio of binomial proportions and empirically gives tighter estimates for $\varepsilon_{\text{emp}}^{\text{lower}}$ with fewer observations (i.e. the method requires fewer number of models that must be trained on D and D'). However, the Katz-log method gives a confidence bound on the ratio of α to β, and it is not clear if this is valid for (ε, δ)-DP where the ratio would change to $\frac{\alpha-\delta}{\beta}$. Lu et al. set $\delta = 0$ in their experiments, giving a lower bound for $(\varepsilon, 0)$-DP. Lu et al. also suggest that empirical privacy leakage is dataset dependent. Our work directly contradicts this claim; we argue that their observations were mostly due to using weaker attacks than are permitted under the DP threat model. By instantiating a more powerful attack our results in Section 6 show that the empirical privacy leakage is close to the theoretical ε across a range of datasets.

While many recent works focus on the DP-SGD. A variety of statistical approaches have been developed to test simpler differentially private algorithms, primarily to identify flawed implementations of private algorithms. These methods offer tools to check if the privacy guarantees claimed by an algorithm hold in practice.

For instance, StatDP [10] was developed to identify violations of ε-DP in the sparse vector algorithm. Further improvements to StatDP were provided by DP-Sniper [5], which catered to similar algorithms, like the sparse vector technique and local differentially private algorithms. DP-Sniper is essentially an advanced testing framework designed to uncover violations in the privacy promises of the algorithms being tested. CheckDP [30] offers another approach. It provides a code analysis-based tool to either prove or generate counterexamples to a variety of algorithms, including the sparse vector algorithm. This methodology allows researchers and practitioners to check the validity of their privacy-preserving algorithms in a more systematic way. Wang et al. [29] recently leveraged auditing estimates to convert them to formal guarantees.

We note that throughout this work we use the terms *trainer*, *auditor*, and *attacker* interchangeably. The party that *audits* the model takes on the role of an *attacker* to measure the empirical privacy leakage, and this involves *training* the model.

3 Motivation & Threat Model

The goal of our work is to improve the efficiency of empirical privacy estimation. This allows us to study the gap between theoretical and practical privacy bounds. And, as a practical application, auditing methods can be used to validate the correctness of a DP implementation.

Our empirical privacy estimates depend upon the specific threat model we instantiate the test within, and assumptions we place on the adversary. Nasr et al. [24] describe several threat models and settings for auditing machine learning with differential privacy. We similarly study multiple threat models, as there is inevitably a trade-off between the power of the audit (with a powerful adversary) and generalizability of the audit to practical machine learning applications (where the assumptions we make to instantiate a powerful adversary may be unrealistic). We focus on three threat models in decreasing order of attack power.

White-box access with gradient canaries: This is the main threat model considered by the DP-SGD theoretical analysis, and matches the (implicit) threat model assumed by DP. The adversary has access to the privatized gradient and model parameters in every update step and can choose an arbitrary gradient at each update step [4], which we refer to as a *canary* gradient. This canary gradient then gets included into the update with probability q. This mimics an adversary who has access to all aspects of training other than the randomness used in noise addition and batch selection (i.e., the knowledge of when z was used in training, where $D' = D \cup \{z\}$).

White-box access with input-space canaries: The threat model above assumes the adversary can choose an arbitrary *gradient* that is sampled into a batch of updates. This may be an unrealistic capability for an adversary in practice. Our second threat model removes this assumption, and instead allows adversaries access to intermediate updates, but restricts them to choose an arbitrary training sample (from which gradients are subsequently computed), rather than the ability to choose a gradient directly. We refer to the training sample chosen for each update step as the *canary* sample. This setting matches the threat model of federated learning particularly well, where an adversary can access model updates but may not always have the ability to insert arbitrary gradients into the training pipeline.

Black-box access: One of the most restrictive threat models to conduct audits on is that of an adversary who can only insert a training example at the beginning of training, and observe the model after it has completed training. In other words, the adversary does not get to observe or influence intermediate model updates. While this is the most restrictive setting from an adversarial perspective, it is perhaps the most realistic from a practical standpoint. We stress that this threat model is not the typical setting analyzed in DP, which assumes intermediate model updates are visible to the adversary. We choose to evaluate it because it allows us to compare how the incremental removal of adversarial access to model updates and gradients affects the tightness of our lower bound for ε.

4 Auditing with f-DP

To audit the privacy of DP-SGD (or other DP mechanisms), an adversary repeatedly runs a distinguishing attack to infer if a model's training set was either D or D'; by measuring the false positive and false negative rates of the attack we can bound privacy. All prior work has used (ε, δ)-DP definition to audit the privacy parameters of the algorithm. The limitation of this approach is that different differential privacy mechanisms with *identical* (ε, δ) guarantees can have *different* trade-offs between false positive and false negative rates, which are upper bounded by the trade-off function of (ε, δ)-differential privacy. In particular, any (ε, δ)-DP guarantee corresponds to two symmetric supporting linear functions defining the trade-off between type I and type II errors. However, any mechanism will have its own "true" trade-off curve capturing the relationship between the FPR and TPR of the best possible attack on the mechanism, consisting of the intersection between a collection of $(\varepsilon, \delta(\varepsilon))$-DP curves where $(\varepsilon, \delta(\varepsilon))$ satisfy Corollary 5. While it is possible to audit any algorithm to lower bound its privacy with an (ε, δ)-DP guarantee, we instead use an f-DP guarantee that is as close as possible to the true trade-off function of the mechanism. By doing so, we can avoid any looseness that appears in converting between this f-DP guarantee and its collection of many (ε, δ)-DP guarantees. It is important to note that our findings are not limited solely to DP-SGD. They can be applied to other mechanisms as well. We concentrated on DP-SGD in this study because it is one of the most commonly used mechanisms in this context. However, the principles and strategies we outlined can certainly be extended and applied to other relevant mechanisms in the field of differential privacy.

To illustrate this idea, in Figure 1 we plot the trade-off functions for several different DP mechanisms that all satisfy (ε, δ)-DP where $\varepsilon = 1, \delta = 10^{-5}$. Clearly, the achievable false positive and false negative rates by an adversary who wants to audit a (ε, δ)-DP guarantee depends significantly on the underlying privacy mechanism.

Let us now give a concrete example demonstrating the benefit of auditing by measuring the privacy region of the private mechanism directly, rather than focusing on the privacy region specified by (ε, δ)-DP. Suppose we want to audit an instance of the Gaussian mechanism satisfying $(1, 10^{-5})$-DP; Figure 2 illustrates the privacy regions of a generic private mechanism with $\varepsilon = 1, \delta = 10^{-5}$ and with a Gaussian mechanism which has an equivalent (ε, δ)-DP guarantee (i.e, μ-GDP with $\mu \approx 0.25$). Now, if we want to audit the Gaussian mecha-

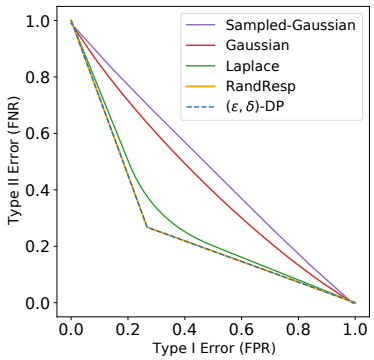


Figure 1: Comparison of the trade-of functions for different DP mechanisms that satisfy (ε,δ)-DP where $\varepsilon = 1, \delta = 10^{-5}$. Note that the trade-off curve for the Random-Response mechanism overlaps with (ε,δ)-DP.

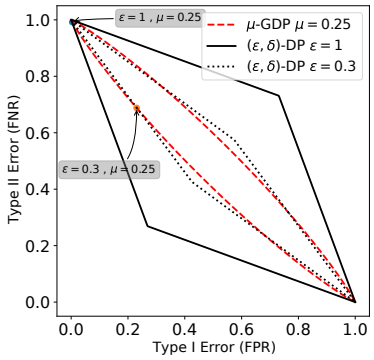


Figure 2: Comparison of the privacy region of (ε,δ)-DP vs f-DP for the Gaussian mechanism (GDP) with the same ε budget.

nism by bounding (ε,δ)-DP, our attack needs to have very low false positive or false negative rates (corresponding to the four locations where the GDP region and the (ε,δ)-DP region have tangent borders). As a concrete example, suppose we are auditing this Gaussian mechanism (with $\varepsilon = 1$) and we have an attack that achieves FPR≈ 0.23 and FNR$= 1-$FPR$(e^{0.3}+\delta)$ over an infinite number of trials (the red dot in Figure 2). If we use (ε,δ)-DP to audit this mechanism we get an empirical ε of 0.3, and we might (incorrectly) conclude that our mechanism is not tight. This is not because our attack is weak, but rather it is because our attack has a large FNR and no attack can achieve a lower FNR from the definition of the Gaussian mechanism (i.e, if an attack can achieve a lower FNR it violates ~ 0.25-GDP).

4.1 Lower Bounding f-DP With Clopper-Pearson

Previous works use Equation (9) which describe the predictive power of an adversary auditing with (ε,δ)-DP to compute the privacy parameters from the false positive and negative rates. However, as we have already seen by appealing to the hypothesis testing interpretation of DP, the predictive power of the adversary is by definition equal to the trade-off function of the privacy mechanism. Therefore, instead of using Equation (9) to compute the privacy parameters, we can directly use the trade-off function. Now by upper bounding the false positive (α) and false negative rates (β) (referred to as $\bar{\alpha},\bar{\beta}$) we can calculate the lower bound on the privacy of the mechanism. Similar to the previous works [24] we can use the Clopper-Pearson method to compute the upper bounds on the attacker errors.

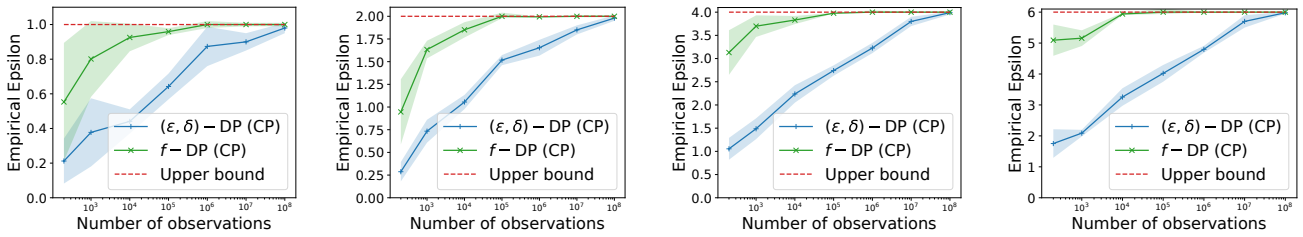
For example, suppose we want to audit the Gaussian mechanism. To compute a lower bound on the privacy parameters of the Gaussian mechanism (i.e, μ), we have:

$$\mu_{emp}^{lower} = \Phi^{-1}(1-\bar{\alpha}) - \Phi^{-1}(\bar{\beta}) \tag{12}$$

We convert this into a lower bound for ε by noticing that the lower bound μ_{emp}^{lower} implies an upper bound on the trade-off function of the mechanism at every α. Such an upper bound on the trade-off function enables us to use Equation (4) at a fixed δ, to find the largest lower bound for ε over all α.

Improvement: Nasr et al. [24] showed DP-SGD accounting is tight for a worst case dataset ($D = \emptyset$), assuming the adversary has white-box access to all iterations of the training. However, they require many observations to achieve tight bounds, due to the aforementioned drawbacks of auditing with (ε,δ)-DP. We re-evaluate this setting using our new approach of auditing with GDP. In Figure 3, we compare the lower bounds found through (ε,δ)-DP (Equation (9)) against using the Gaussian trade-off function and converting the Gaussian mechanism parameter to (ε,δ)-DP, and we inspect how these two methods compare as the adversary collects more observations from which they compute upper bounds $\bar{\alpha}$ and $\bar{\beta}$. While, it is possible for (ε,δ)-DP audit to find a lower bound that is tight to the theoretical value for ε, this is only achieved when the adversary has 100 million observations. When the number of the observations is smaller there is a non-trivial gap between the theoretical bound and the empirical lower bound. Comparatively, if we use the Gaussian mechanism's trade-off function (GDP) to estimate a lower bound on the privacy parameter μ, and convert this into a bound on ε, we can achieve a tight estimate even with 1,000 observations.

Approximating the trade off function: While using GDP to audit can give us a tight lower bound on the estimated privacy in simple cases, analyzing the privacy cost of a complex



(a) $\varepsilon = 1$ (b) $\varepsilon = 2$ (c) $\varepsilon = 4$ (d) $\varepsilon = 6$

Figure 3: Comparison of the (ε, δ)-DP definition to audit DP-SGD compared to using f-DP of Gaussian mechanism both converted in (ε, δ)-DP and lower bounded using Clopper Pearson (20 independent runs, $\delta = 10^{-5}$).

mechanism such as DP-SGD (that needs both sub-sampling amplification and composition over multiple update steps) is non-trivial, and the current existing approaches returns loose privacy estimates for different values of ε. Dong et al. [11] suggest to use a Central Limit Theorem for DP to compute a closed form solution, unfortunately, this can lead to significant underestimation of the privacy cost in general settings [18]. Therefore, to analyze DP-SGD we use an empirical approach called the "Privacy Loss Distribution (PLD)" [18] to approximate the trade-off function, which computes a tight privacy cost of DP-SGD over multiple update steps. We refer to Koskela et al. [18] for a detailed description of PLD, and will interact with it as a black-box $\varepsilon = f_{\mathcal{M}}(\delta)$ that for private mechanism $\mathcal{M}$ and a given δ will return the exact theoretical ε. PLD does not have a closed-form trade-off function, in Appendix A we explain how we use PLD for auditing. Wang et al. [29] recently designed a parallalization based approach for computing PLD which can be used to speed up the PLD computations specially for settings where δ is very small.

4.2 Lower Bounding f-DP With Bayesian Estimation

Recently, Zanella-Béguelin et al. [35] showed it is also possible to compute *credible intervals* for ε using a Bayesian method which can significantly reduce the number of observations to estimate a tight bound. Here, we show it is possible to extend their approach to lower bound in f-DP (and then convert to a lower bound for ε).

Definition 6 (Cumulative Distribution Function of $f(\alpha,\hat{})$-DP)**.** *Let $u_{(FPR,FNR)}$ be the density function of the joint distribution of (FPR,FNR). The value of cumulative distribution function of $f(\alpha;\hat{})$ evaluated at $f(\alpha;.)$ is:*

$$P_{\hat{}}(.) = \int_{f(\alpha;.)}^{1-f(1-\alpha;.)} \int_0^1 u_{(FPR,FNR)}(\alpha,\beta)\, d\alpha\, d\beta \tag{13}$$

Using Eq. (13), we can find an empirical lower bound for μ in μ-GDP, and then convert to a lower bound for (ε, δ)-DP. As shown by Zanella-Béguelin et al. [35], using the CDF of the private mechanism parameters given the attack observations of we can compute credible intervals over ε. As a reminder Zanella-Béguelin et al. defined $u_{(\text{FPR,FNR})}$ as follows:

$$\begin{aligned} u_{(\text{FPR,FNR})}(\alpha,\beta) :=& u_{(\text{FPR}|\text{FP})}(\alpha) u_{(\text{FNR}|\text{FN})}(\beta) \\ =& \text{Beta}(\alpha; 0.5+\text{FN}, 0.5+\text{N}-\text{FN}) \times \\ & \text{Beta}(\beta; 0.5+\text{FP}, 0.5+\text{N}-\text{FP}) \end{aligned} \tag{14}$$

where N is the number of observations used to compute false positive and false negative rates, and FN, FP are the total number of false negative and false positives, respectively.

5 Auditing Setup

As we have seen, the choice of framework used for auditing can affect the tightness of our lower bound for ε. In this section, we describe our auditing procedure for each threat model described in Section 3, and then discuss the effect of different attacker choices—such as attack specific hyperparameters—have on the audit results.

5.1 Auditing Procedure

As mentioned in Section 3, we consider three main threat models. For the black-box setting, we use Algorithm 1 which trains $2T$ models on datasets D and $D' = D \cup \{z\}$ where $z = (x', y')$ is the differing example between D and D', which we refer to as the *canary*. Then the auditor evaluates the loss on the canary example of each model trained on D and D'; using this set of losses, the auditor chooses a decision threshold and computes α and β. We refer to the statistics collected by the adversary as *observations*.

In the white-box setting, the adversary can observe model parameters at each update step. We summarize the approach used for auditing in a white-box setting in Algorithm 2, using either canary gradients or canary inputs as described in Section 3. At each iteration of DP-SGD, the trainer independently samples two batches of data, B and B'. Theoretically, B

Algorithm 1 Black-box auditing for DP-SGD

Args: training dataset D, loss function l, canary input (x', y'), number of observations T
Observations: $O \leftarrow \{\}$, $O' \leftarrow \{\}$
for $t \in \{T\}$ **do**
  $\theta \leftarrow$ DP-SGD on Dataset D
  $\theta' \leftarrow$ DP-SGD on Dataset $D \cup (x', y')$
  $O[t] \leftarrow l(\theta, (x', y'))$
  $O'[t] \leftarrow l(\theta', (x', y'))$
end for
  return θ , O, O'

can be identical to B'. We experimented with both configurations and did not observe any significant difference between them. Consequently, we opted for this setup as it carries fewer assumptions, specifically removing the need to fix the randomization process for sampling. In the *White-box access with Input Space Canaries* threat model, the trainer creates a canary sample and adds it to B' with probability q_c.

In the *White-box access with Gradient Canaries* threat model, the trainer computes the batch of per-example gradients for B and B' and adds a canary gradient into B' with probability q_c. Please note that in both input and gradient space attacks, the canaries can be chosen dynamically based on the current parameters of the model.

Batch B has been sampled from the original dataset D, while B' has been sampled from a modified dataset D'. In both threat models, after the canary is added, DP-SGD proceeds as normal, clipping, aggregating, and noising the gradient sums. For each batch, the trainer computes the dot product between the privatized gradient sum and the canary gradient (or the gradient from the canary input), resulting in a score (which we again refer to as an observation). The goal of the adversary is to determine whether a batch was drawn from D or D'. At the end of each run of the algorithm, the trainer produces $2T$ observations, two for each update, and a fully trained model with parameters θ.

In Algorithm 2, we consider different sampling rates for the canary example and normal training examples to allow the model trainer to evaluate the mechanism at different sampling rates. We use q to represent the sampling rate for the normal instances and q_c for the canary examples. If the trainer sets $q_c = 1$, the canary is selected in all iterations, and the audit focuses on the privacy mechanism without data sub-sampling. This modification allows us to identify bugs in DP-SGD that are not due to batch sampling.

After collecting observations from a model trained on either dataset D or D', they can be compared with a threshold to compute true and false positive rates. Next, we will focus on considerations for choosing an appropriate threshold and their effects on the auditing process. We will also discuss other important factors in the auditing process, such as canary selection strategies and the sampling rate q_c.

Algorithm 2 White-box auditing for DP-SGD with gradient or input space canaries

Args: training dataset D, sampling rate q, learning rate η, noise scale σ, gradient norm clip C, loss function l, canary gradient $C(\theta) \rightarrow g'$, canary input $C(\theta) \rightarrow (x', y')$, canary sampling rate q_c, function `clip` that clips vectors to max norm C, number of observations T, number of training iterations τ.

Observations: $O \leftarrow \{\}$, $O' \leftarrow \{\}$
Trained Models: $\Theta \leftarrow \{\}$
$t \leftarrow 0$
while $t \leq T$ **do**
  Initiate θ randomly
  for τ iterations **do**
    $B_t \leftarrow$ sample instances from dataset D with prob q
    $B'_t \leftarrow$ sample instances from dataset D with prob q
    $\nabla[t] \leftarrow \vec{0}$
    for all $(x, y) \in B_t$ **do**
      $\nabla[t] \leftarrow \nabla[t] + \texttt{clip}(\nabla_\theta(l(x, y)))$
    end for
    $\widetilde{\nabla[t]} \leftarrow \nabla[t] + \mathcal{N}(0, \sigma^2 \mathbb{I})$
    $\nabla'[t] \leftarrow \vec{0}$
    for all $(x, y) \in B'_t$ **do**
      $\nabla'[t] \leftarrow \nabla'[t] + \texttt{clip}(\nabla_\theta(l(x, y)))$
    end for
    $\widetilde{\nabla'[t]} \leftarrow \nabla'[t] + \mathcal{N}(0, \sigma^2 \mathbb{I})$
    $g' \leftarrow C(\theta)$ canary gradient or $g' \leftarrow \texttt{clip}(\nabla_\theta(l(C(\theta))))$
    $\widetilde{\nabla'[t]} \leftarrow \widetilde{\nabla'[t]} + g'$ with prob q_c o.w. $\widetilde{\nabla'[t]} + \vec{0}$
    $O[t] \leftarrow \langle g', \widetilde{\nabla[t]} \rangle$
    $O'[t] \leftarrow \langle g', \widetilde{\nabla'[t]} \rangle$
    $\theta \leftarrow \theta - \eta \widetilde{\nabla[t]}$
    $t = t + 1$
  end for
  $\Theta = \Theta + \{\theta\}$
end while
return Θ , O, O'

5.2 Choosing a Decision Threshold

The output of Algorithm 1 or Algorithm 2 is a set of observations, with canaries $\{o'_1, o'_2, \ldots, o'_T\}$, and a set without canaries, $\{o_1, o_2, \ldots, o_T\}$. To compute our attack's FNR and FPR, we must first choose a decision threshold to distinguish between observations from the observation space without the canary, O, or the observation space with canaries O'.

In the white-box threat model, our observation is $o = \langle g', \widetilde{\nabla[t]} \rangle$ and $o' = \langle g', \widetilde{\nabla'[t]} \rangle$, where g' is the canary gradient, $\widetilde{\nabla[t]}$ is the privatized gradient over a batch B, and $\widetilde{\nabla'[t]}$ is the privatized gradient over a batch B'. By construction, we expect g' to be orthogonal to any other gradient g in the batch, $\langle g', g \rangle = 0$. In practice, the clipping norm and batch size are known to the adversary and so we can re-scale and normalize the set of observations such that $O = \mathcal{N}(0, \sigma^2 I)$ and $O' = \mathcal{N}(1, \sigma^2 I)$, meaning that in expectation o and o' are sampled from Gaussians with zero and unit mean, respectively. Thus, the attacker's goal is to distinguish observations sampled from $\mathcal{N}(0, \sigma^2 I)$ and $\mathcal{N}(1, \sigma^2 I)$ – we note that this is exactly the same hypothesis testing problem considered in GDP. A benefit of auditing with $q_c = 1$ and GDP is that

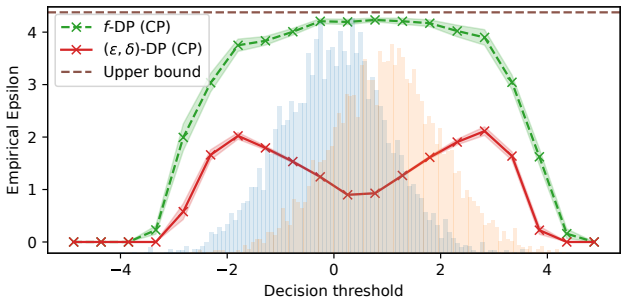


Figure 4: An example comparison between f-DP and (ε,δ)-DP with Clopper-Pearson lower bounds for 5,000 observations. The lower bounds found through GDP are approximately the same for any decision threshold within the observation's support, whereas the (ε,δ)-DP with Clopper-Pearson lower bound varies dramatically. We also visualize observations when the canary was and wasn't included in training as histograms.

the lower bounds we derive with GDP are agnostic to our choice of decision threshold used to compute type I and type II errors (in the limit of number of observations), while this is not true for (ε,δ)-DP. This is due to the perfect match between the GDP analysis and the true privacy of the Gaussian mechanism. Figure 4 shows the result that different choices of threshold have on auditing using either GDP or (ε,δ)-DP. As we can see, auditing using (ε,δ)-DP is much more sensitive to the decision threshold compared to GDP. Note that if thresholds we use are found using the same observation data that we compute the lower bound on, the bound is technically not valid. However, it has become common to report lower bounds on the same set of observations that one uses to find an optimal decision threshold [20,35], and so for each method we will find the threshold that maximizes the reported lower bound. We stress that for GDP, any decision threshold will be equally likely to maximize the lower bound with a sufficient number of observations; this is not true of (ε,δ)-DP. See our extended version for full discussion [22, Appendix C.1].

Next, we discuss how to construct the canary point (either gradient or sample) used in auditing.

5.3 Canary Type

The strength of our bound depends on being able to distinguish samples from O and O'. In turn, this means crafting canaries that maximize distinguishability.

In the *White-box access with Gradient Canaries* threat model, we use what we refer to as a Dirac canary gradient; a gradient with zeros everywhere except at a single index in the gradient vector, where we set its value to the clipping norm C. We compare this choice with other possibilities and also the effect of generating canaries dynamically in Appendix B.1. I

In the *White-box access with Input Space Canaries* threat

Algorithm 3 Input canary generation in white-box setting

Args: In-distribution dataset D, model loss function l, model parameters θ, T crafting steps, η step size
$\vec{g}_{dist} = \frac{1}{|D|}\sum_{(x_i,y_i)\in D}\nabla l(\theta,(x_i,y_i))$
$l_{adv}(x,y) = |\frac{\nabla l(\theta,(x,y)).\vec{g}_{dist}}{|\nabla l(\theta,(x,y))||\vec{g}_{dist}|}|$
$(x,y) \xleftarrow{\$} D$
for $t \in \{T\}$ **do**
  $x = x - \eta\nabla l_{adv}(x,y)$
end for
  return (x,y)

model, we design a new attack that crafts an input for given model parameters. We evaluate four different canary strategies: (1) a random sample from the dataset distribution with a wrong label, (2) using a blank sample, (3) an adversarial example, and (4) and our new canary crafting approach given in Algorithm 3 and discussed in Appendix B.2. In the *black-box* threat model, we consider a similar range of canary types, which are detailed in Section 6.4. In the main body the work we only present the results of the best possible attacks.

5.4 Canary Sampling Rate (q_c)

The analysis of the sub-sampling mechanism in the worst case is tight using PLD/f-DP [18]. From Figure 13, we see that when $q_c = 1$, the observations closely match the theoretical FPR-FNR trade-off. Instead, in Figure 5 we audit a sub-sampled Gaussian mechanism with sampling rate of $q_c = \frac{1}{4}$, setting $\sigma^2 = 0.3$ and the number of collected observations to 10,000. This figure plots the FPR-FNR curve predicted by using the PLD accounting and also GDP accounting with an equivalent ε with $\delta = 10^{-5}$, and compares it to the empirical curve found through auditing this sub-sampled Gaussian mechanism. Clearly, both PLD and GDP upper bounds the observed FPR-FNR curve, but tends to overestimate the trade-off between FPR and FNR, particularly at higher false positive rates [11]. This suggests that to accurately audit a sub-sampled privacy-preserving mechanism, it may be necessary to use attacks with more precise false positive rates and optimal thresholds in order to achieve tight bounds. As mentioned in Section 5.2 (and expanded upon in **??**) by using $q_c = 1$ (and a sufficient number of observations) we do not need to find the optimal threshold, as the lower bound found using GDP auditing is threshold agnostic, and any threshold will results in tight auditing (which ensures a valid confidence interval and lower bound).

When using auditing to debug an implementation of DP-SGD, we focus on the auditing of the privacy mechanism itself, rather than the sub-sampling process. We therefore set $q_c = 1$ throughout most of the experiments in Section 6. However, we will experiment with the sub-sampled Gaussian mechanism ($q_c < 1$) in the black-box threat model, where we are more focused on effect of the threat model on privacy

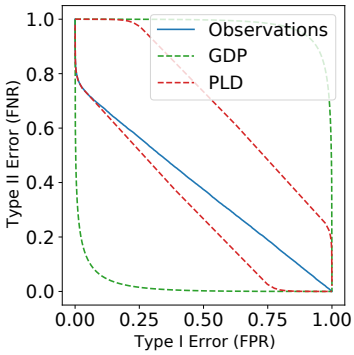


Figure 5: PLD or GDP analysis does not describe the achievable error rate from a sub-sampled Gaussian mechanism.

leakage rather than debugging to check if an implementation of DP-SGD is correct.

5.5 From Step-wise to End-to-end Auditing

Our white-box audit observes each step of DP-SGD. As such, we generate privacy lower bounds for individual steps. We convert these into a privacy lower bound for the end-to-end training procedure by appealing to tight composition results. For instance, if we assume that each step is tightly characterized by Gaussian DP, then the tightness of Gaussian DP composition allows us to infer that the end-to-end training procedure is also tightly characterized by Gaussian DP and that we can sum up the privacy parameters. In other words, if the privacy loss distribution of one step is Gaussian, then the privacy loss distribution of the end-to-end procedure is also Gaussian; this is because composition simply adds/convolves the privacy losses. Given a lower bound for a single step of DP-SGD, $\varepsilon_{\text{emp}}^{\text{lower}}(\delta)$, we find a Gaussian noise scale σ that corresponds to this (ε,δ)-DP guarantee. The we compose the corresponding Gaussian DP guarantees to obtain our final estimate.

In DP-SGD, there is also subsampling. That is, we must account for the randomness of the batch selection. In this case, the privacy is not tightly characterized by Gaussian DP.[1] However, it can be characterized by a more general f-DP guarantee and then we can use PLD to compose over the number of update steps. In Section 6, we show that this method of conversion gives empirical estimates for ε that are close to the end-to-end theoretical ε value. We note that a similar idea has been explored by Maddock *et al.* [20].

[1] Although it is not tightly characterized by Gaussian DP, a subsampled Gaussian can be approximated by Gaussian DP. In practice, this approximation yields conservative estimates of the final (ε,δ)-DP guarantee. Hence this would also be an acceptable auditing methodology.

6 Experiments

We now evaluate the performance of our proposed auditing technique. We first demonstrate that auditing with f-DP gives a tight bound on privacy leakage. After this, we show that tight auditing can be used for a multitude of purposes, such as investigating if certain choices of training hyperparameters lead to more or less privacy leakage, and debugging implementations of DP-SGD.

6.1 Experiment Setup

We experiment with two commonly used datasets in the privacy literature: CIFAR-10 (with Wide ResNet (WRN-16) [34] and ConvNet architectures) and Purchase. In addition we also evaluate our experiment on a randomly initialized dataset. Please to our extended version [22, Appendix B] where we describe the hyperparameters and details used to train models. Unless otherwise stated, all lower bounds are given with a 95% confidence (Clopper-Pearson as in Nasr et al. [24]) / credible interval (Zanella-Béguelin et al. [35]), and we audit in the *White-box access with Gradient Canaries* threat model using Algorithm 2. In all of our experiment the initial parameters of the models are equal between different runs, however, the random processes use different seeds.

Terminology: We describe below the approaches we use to compute lower bounds; our work introduces the f-DP strategies. **f-DP (CP)**: using the trade-off function of the privacy mechanism with Clopper-Pearson. When we do not have the exact trade-off function (e.g, if there are multiple composition of sub-sampled Gaussian mechanisms) we use approximated trade-off function of the privacy mechanism from PLD accounting and Algorithm 4.**(ε,δ)-DP (CP)** [14, 24]: using the (ε,δ)-DP (Equation (4)) trade-off function with Clopper-Pearson. **f-DP (ZB)** and **(ε,δ)-DP (ZB)** are similar to **f-DP (CP)** and **(ε,δ)-DP (CP)**, however, we use the Bayesian estimation approach (Section 4.2) to compute lower bounds instead of Clopper-Pearson. **(ε,δ)-DP (ZB)** is equivalent to the approach used by Zanella-Béguelin et al. [35]. **$\varepsilon-$DP (Katz)** [19] audits $\varepsilon-$DP with the Katz log confidence interval.

6.2 (Almost) Tight Auditing of DP-SGD For Natural Datasets Using f-DP

Nasr et al. [24] showed that DP-SGD is tight with worst-case training sets; however, they observed a noticeable gap between the empirically estimated lower bound and theoretic upper bound for ε when they replace these worst-case datasets with datasets commonly used for DP-SGD benchmarking, even when the adversary has white-box access to the model and can insert canary gradients (e.g., they achieved an empirical ε lower bound of < 1 with a theoretical ε of 8 on CIFAR-10).

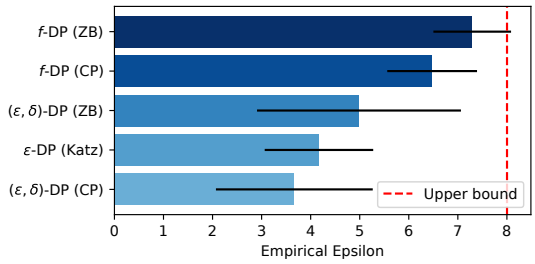


Figure 6: Auditing with f-DP provides the strongest lower bounds with theoretical upper bound $\varepsilon = 8$ on CIFAR-10.

Table 1: Comparison of the empirical lower bounds on epsilon with 95% (confidence or credible interval), where the adversary has access to every intermediate model and the adversary can insert a canary gradient vector (white-box setting). We include results on a *Random* dataset—random pixels and labels—of the same cardinality as CIFAR-10.

Lower Bounding	Theoretical ε	CIFAR-10 WRN-16	CIFAR-10 ConvNet	Purchase	Random WRN-16
f-DP (CP)	1	0.75	0.77	0.78	0.74
	4	3.40	3.34	3.54	3.14
	8	5.80	6.12	6.40	7.14
	16	11.14	12.08	12.42	13.14
f-DP (ZB)	1	0.95	0.94	0.89	0.90
	4	3.73	3.80	3.60	3.52
	8	7.09	7.12	6.94	7.12
	16	13.95	13.80	13.80	15.14
(ε, δ)-DP (CP)	1	0.41	0.45	0.36	0.35
	4	1.37	1.80	1.65	1.14
	8	3.63	3.85	3.25	4.09
	16	5.25	6.22	6.34	6.96
(ε, δ)-DP (ZB)	1	0.62	0.62	0.57	0.61
	4	2.65	2.69	2.45	2.75
	8	5.07	5.15	4.65	5.09
	16	5.25	6.22	6.34	6.96
ε−DP (Katz)	1	0.49	0.51	0.46	0.41
	4	1.65	1.95	2.05	2.14
	8	4.17	3.95	4.24	4.15
	16	7.52	7.63	7.69	8.01

We first demonstrate that by auditing with f-DP, we can now compute strong lower bounds with the standard CIFAR-10 training set, where we train and audit a model with 79% test accuracy at $(\varepsilon = 8, \delta = 10^{-5})$-DP. We evaluate and compare our auditing technique against the state-of-the-art auditing methods of Zanella-Béguelin et al. [35] and Lu et al. [19]. Results are given in Figure 6, where we report the average lower bound found over ten independent executions of the experiment along with standard deviation, and the theoretical upper bound for ε given by the privacy accountant. Regardless of if we use Zanella-Béguelin et al.'s method for finding credible intervals, or use the Clopper-Pearson confidence interval, the main gain in estimating ε comes from auditing with f-DP. Auditing with f-DP is almost tight, while the strongest upper bound from prior work is ~ 5.

Our method of auditing with f-DP gives a tight analysis for privacy leakage for both small and large ε and across different datasets (CIFAR-10, Purchase, and a *Random* dataset—random pixels and labels—of the same cardinality as CIFAR-10). The results, reported in Table 1, show that our approach *does not require the use of a worst-case dataset to achieve tight estimation of the privacy parameters*. Lu et al. [19] hypothesize that privacy is dataset dependent even in a white-box setting, however, our experiments contradict this hypothesis. Given that our results show that tight lower bounds are largely independent of the choice of dataset if the adversary audits in a white-box threat model with canary gradients, our remaining experiments will focus primarily on the CIFAR-10 dataset unless stated otherwise.

We next demonstrate that auditing with f-DP can be useful for detecting implementations of DP-SGD that violate the purported upper bound for ε. As discussed previously, we will concentrate on violations that are not directly caused by sub-sampling, and so our experiments will audit the Gaussian mechanism without composition or sub-sampling, for which we will use the exact trade-off function (GDP instead of the PLD approximation detailed in Section 4.1).

6.3 Auditing and Debugging DP-SGD Implementations

Implementing DP-SGD correctly is notoriously difficult. Auditing can help identify issues of correctness, as demonstrated by Tramèr et al. [27] who used black-box auditing to show the DP-SGD implementation proposed by Stevens et al. [26] was incorrect and reported a much lower value of ε than its true privacy leakage.

We investigate how easily our method of auditing with f-DP can detect incorrect implementations of DP-SGD compared to prior work on CIFAR-10. The upper bound for ε claimed by each DP-SGD implementation throughout the following experiments is 1.27. For all experiments in this section we audit a step of DP-SGD, that is, we do not convert our lower bounds into a guarantee of the ε reported after composing across all training steps with PLD. We do this because even if we were to report a lower bound on the final value of ε (via finding a lower bound for a step of DP-SGD and composing with the (almost) lossless PLD), there will be bugs that cannot be captured by this auditing method. For example, a bug that is caused by implementing a biased sub-sampling method from the training dataset will likely not be captured by our audit.

Violation 1: Clipping *after* gradient averaging. In DP-SGD, individual gradients must be clipped to a maximum norm C before aggregation. If the order of these operations is reversed, clipping after aggregation, then the model will not be (ε, δ)-DP. In Figure 7, we see all auditing methods are able to identify a violation as the lower bound found is much larger than the reported upper bound. However, one may still incorrectly assume that the implementation retains some privacy if we do not audit with f-DP, as the best lower bound we can find is < 10. By auditing with f-DP, it becomes clear that the implementation is completely broken.

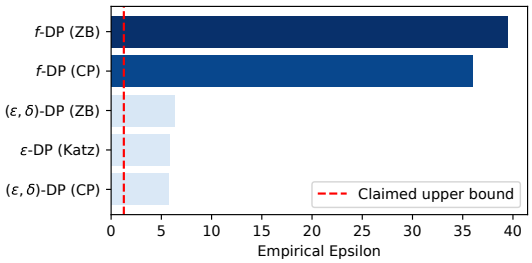


Figure 7: Clipping after gradient averaging bug. We plot the lower bound for ε we can find with each auditing technique when the implementation clips the gradient after averaging in a batch, and so is not (ε, δ)-DP with the claimed $\varepsilon = 1.27$. All methods are able to detect a violation but only f-DP auditing can show the implementation is completely broken, as we can show $\varepsilon > 35$.

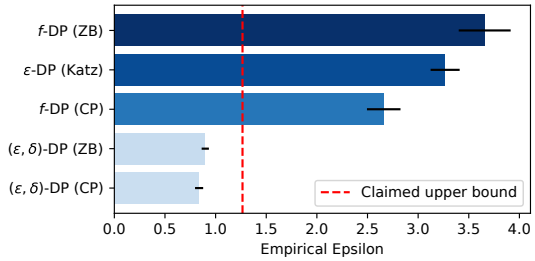


Figure 8: Biased noise bug. The Gaussian noise used to privatize gradients in DP-SGD is not sampled randomly. Both f-DP and Lu et al. detect this implementation issue while auditing with (ε, δ)-DP fails to detect the issue.

Violation 2: Biased noise sampling. At every step of DP-SGD, random Gaussian noise must be added to gradients. If the noise is not randomly sampled then the model will not be (ε, δ)-DP. In practice, we generate a Gaussian noise sample by seeding a random number generator. We train a model where the seed can only take on 100 different possible values, meaning there are only 100 different possible Gaussian noise vectors. It may seem that this is a rather contrived example of a DP-SGD bug, but a similar error appeared in the JAX canonical example of how to implement DP-SGD, where a random seed was re-used when adding noise to different sets of model parameters [1].

Results are shown in Figure 8. We can detect violations when auditing with f-DP; the auditing method introduced by Lu et al. [19] also successfully detects the bug. However, auditing with (ε, δ)-DP directly, either using Clopper-Pearson or the method proposed by Zanella-Béguelin et al. was not able to identify a violation of the claimed upper bound.

Violation 3: Noise scale is too small. The value of ε is inversely proportional to the scale of noise we add to gradients. As we decrease the scale of noise, ε increases. The third bug we investigate is when the noise scale we add is unexpectedly

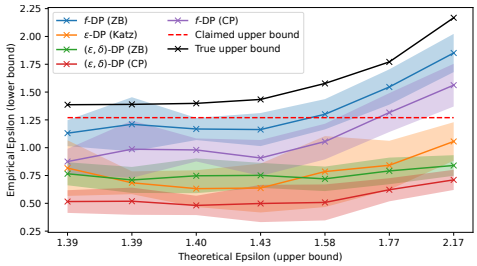


Figure 9: Incorrect noise scale bug. We measure how quickly each auditing method can detect a violation of the purported upper bound of $\varepsilon = 1.27$ when the scale of noise we add to gradients is incorrectly set.

smaller than the target scale we set. This bug often arises because the the sensitivity (clipping value) needs to calibrated to the batch size in order to compute the correct noise scale, and this is easy to get wrong (c.f. Tramèr et al. [27]). For example, in settings where gradient computations are distributed across multiple machines, we could incorrectly add noise to the average gradient found on each machine, and then aggregate.

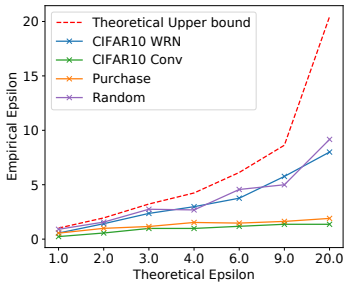
We train models with decreasing scales of noise, implying larger values of ε than the claimed upper bound. Results are shown in Figure 9, where we find auditing with f-DP closely follows the true upper bound, meaning we can detect a violation to reported $\varepsilon = 1.27$ when the true value is $\varepsilon = 1.57$. Auditing with (ε, δ)-DP directly, either using Clopper-Pearson, Lu et al. [19], or the method proposed by Zanella-Béguelin et al. does not successfully identify a violation of the claimed upper bound, even when the true upper bound for ε is as large as 2.17.

6.4 Investigating Privacy Leakage with Tight Auditing

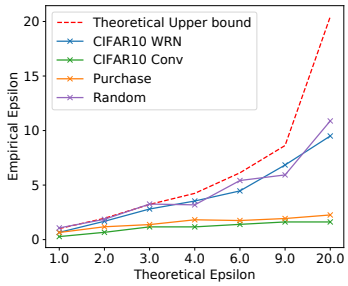
In this section, we expand our analysis to examine the impact of various settings and parameters on privacy leakage.

Is there a difference between auditing in gradient and input space? Auditing with gradient canaries can be useful in specific contexts, such as in federated learning or when debugging a model. However, in most cases, practitioners are more concerned with understanding the effect of a single input space example on the model. This is because this setting more closely measures the privacy leakage that could be experienced by a worst-case training example.

To evaluate how privacy leakage could change by removing the ability to insert a canary gradient, we run experiments in the *White-box access with Input Space Canaries* threat model, as described in Section 3. In particular, we use Algorithm 3 to create the canaries. Note that for this experiment we only use the first 250 iterations of DP-SGD to collect observations and estimate the per-step ε lower bound. We found that, in



(a) f-DP (CP)



(b) f-DP (ZB)

Figure 10: Comparison of the empirical lower bounds on epsilon with 95% (confidence or credible interval), where the adversary has access to every intermediate model and the adversary can insert a canary in input space (white-box setting).

Table 2: Comparison of the empirical lower bounds on ε (using f-DP (ZB)) in the black-box threat model where the adversary inserts a canary point at the beginning of training, and only gradient clipping is applied on CIFAR-10.

Clipping norm C	ε lower bound
0.1	23.4
1.0	26.8
10.0	41.0
Maximum ε lower bound at this number of observations	44.0

this threat model, the first few hundred iterations of DP-SGD leaks more privacy than the entire training run; in other words, the lower bound we compute using the first 250 steps is larger than the lower bound we compute over the entire training run (2,500 update steps). In general, we find that auditing in the input space becomes weaker if the observations are collected from updates towards the end of training. We discuss this further in Appendix C.1.

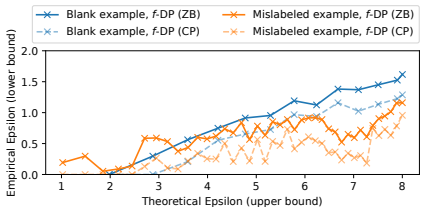
As shown in Figure 10, even if the attacker can only insert canary samples (rather than gradients), we can compute tight bounds for $\varepsilon < 10$. We also observe that the choice of the model architecture has an impact on auditing in input space. Specifically, we can get an (almost) tight lower bound when using Wide Resnet architecture regardless of the datasets. Moreover, when we compare the CIFAR-10 dataset results between Wide Resnet and ConvNet models, we see a large gap which further emphasizes the significance of the model architecture on privacy leakage.

Does clipping alone help? It has been conjectured that clipping individual gradients can provide some privacy even without adding noise [6]. Technically, these models are not differentially private but we can measure the privacy that clipping provides by computing an ε lower bound. Of course, it doesn't make sense to audit clipping alone in a white-box access threat model; no noise is added and so the dot-product value we compute to find type I and type II error rates (as described in Section 5.1 and Section 5.2) will not be masked by any noise. Instead, we audit in the black-box threat model by creating a canary point, inserting it into the training set, and then training a CIFAR-10 WRN-16 model. We then measure the loss after training on the canary image. We do this 1,000 times when the canary image was included in training and when it wasn't, and record the loss of the canary image in each case. We experimented with a range of different canary inputs, but found that blank (white) and mislabeled images produced the strongest lower bounds for ε. In other words, these two canary types had losses that were easily separable depending on if they were included in training or not.

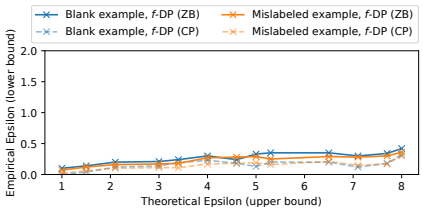
In total, we train 2,000 models: 1,000 when the canary input was in training and 1,000 when the canary wasn't included. Our results are shown in Table 2, where we see that clipping alone provides no privacy. The lower bounds for a clipping norm of 10 are close to the maximum possible lower bound at this number of observations.

How much privacy is leaked in a black-box threat model? To measure the impact of switching to a black-box threat model on privacy leakage estimation, we follow a similar auditing procedure as set out by Nasr et al. [24]. We select a canary example, then we train 1,000 models with the canary point (+ training set), and 1,000 models with only the training set (canary excluded). We then measure the $\log(\frac{p}{1-p})$ for each model, where p is the probability of the canary point with respect to its label. We take the distribution of $\log(\frac{p}{1-p})$ when the canary point was and wasn't in the training set, and compute ε lower bounds using our f-DP method.

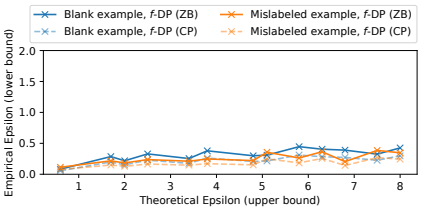
Results are shown in Fig. 11, where all models are trained up to $\varepsilon = 8$. On CIFAR-10 with a Wide ResNet architecture we are able to find a lower bound of ~ 1.6 using f-DP (ZB) auditing. However, it is difficult to separate the effects of the black-box threat model from the effect that sub-sampling has on privacy leakage estimation, as we saw in Figure 5, both GDP and PLD tend to overestimate the observed trade-off between type I and type II errors in the sub-sampled Gaussian mechanism (with composition over multiple iterations). Additionally, the auditing results on other datasets/architectures are significantly lower compared to those obtained with the Wide ResNet architecture on the CIFAR-10 dataset. Specifically, when examining the random dataset, no non-trivial lower bounds for auditing can be achieved. One of the key factors that distinguishes these experiments is the final accuracy the model is able to attain. At $\varepsilon = 8$, the Wide ResNet architecture for CIFAR-10 is able to achieve a test dataset accuracy of greater than 70%, compared to less than 50% for the ConvNet architecture and less than 50% for the Purchase



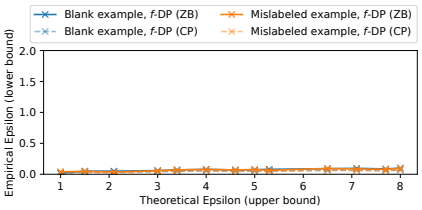
(a) CIFAR-10, WRN-16



(b) CIFAR-10, ConvNet



(c) Purchase



(d) Random dataset, WRN-16

Figure 11: Auditing the black-box threat model with f-DP. We train 1,000 models with and without the canary sample that we insert at the beginning of training. We either use a blank or mislabeled image as the canary input.

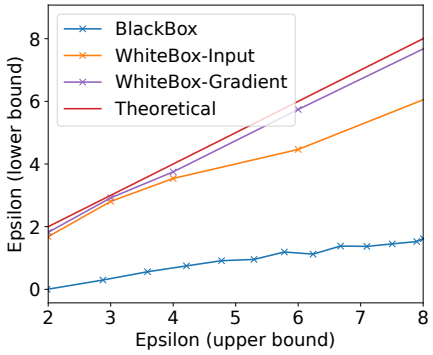


Figure 12: Comparison of attacks on CIFAR10 WRN-16 in different settings

dataset. Our results suggest that there remains a gap between the theoretical privacy upper bound and the empirical lower bound that can be achieved in a black-box setting, but the size of this gap is highly dependent on the dataset and model architecture.

7 Discussion and Future Directions

As demonstrated in our work, our gradient attacks achieve highly tight results in the majority of settings. These findings suggest that in practice, the bounds of DP-SGD are not excessively loose, thereby requiring practitioners to exercise caution when selecting their parameters. This is particularly crucial in settings like Federated Learning, where an adversary may have white-box access to the model.

When looking at the comparison of our attack in different settings (e.g, Figure 12), we can observe that our attacks are still weak in the fully black-box setting. A significant direction for future research in the area of auditing differential privacy methods lies in improving tightness in a black-box setting. Although our results substantially outperform existing approaches, we observe a large gap, particularly in end-to-end settings, between practical attacks in this threat model and the theoretical upper bounds. Future work should strive to enhance results in these settings. We believe that important steps towards this goal include comprehending the specific model parameters that can influence such results and designing superior attacks for black-box settings.

8 Conclusion

As differentially private machine learning becomes more popular and fewer expert users begin to implement such methods, the possibility of bugs and implementation errors will increase. We provide a simple auditing technique that can achieve tight estimates of privacy leakage on standard ML benchmark datasets. Our method can be easily integrated with privacy preserving libraries (TF-privacy, Opacus, JAX privacy) to give online estimation of the private mechanism parameters and provide an empirical test for the assumed privacy budget, and only increases the computational overhead by a factor of two.

References

[1] fix prng key reuse in differential privacy example by mattjj · Pull Request #3646 · google/jax — github.com. https://github.com/google/jax/pull/3646. [Accessed 28-Jan-2023].

[2] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. In *Proceedings*

of the 2016 ACM SIGSAC conference on computer and communications security (2016), pp. 308–318.

[3] ALTSCHULER, J. M., AND TALWAR, K. Privacy of noisy stochastic gradient descent: More iterations without more privacy loss. *arXiv preprint arXiv:2205.13710* (2022).

[4] BALLE, B., CHERUBIN, G., AND HAYES, J. Reconstructing training data with informed adversaries. *arXiv preprint arXiv:2201.04845* (2022).

[5] BICHSEL, B., STEFFEN, S., BOGUNOVIC, I., AND VECHEV, M. Dp-sniper: Black-box discovery of differential privacy violations using classifiers. In *2021 IEEE Symposium on Security and Privacy (SP)* (2021), IEEE, pp. 391–409.

[6] CARLINI, N., CHIEN, S., NASR, M., SONG, S., TERZIS, A., AND TRAMER, F. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)* (2022), IEEE, pp. 1897–1914.

[7] CARLINI, N., IPPOLITO, D., JAGIELSKI, M., LEE, K., TRAMER, F., AND ZHANG, C. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646* (2022).

[8] CARLINI, N., LIU, C., ERLINGSSON, Ú., KOS, J., AND SONG, D. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)* (2019), pp. 267–284.

[9] DE, S., BERRADA, L., HAYES, J., SMITH, S. L., AND BALLE, B. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650* (2022).

[10] DING, Z., WANG, Y., WANG, G., ZHANG, D., AND KIFER, D. Detecting violations of differential privacy. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (2018), pp. 475–489.

[11] DONG, J., ROTH, A., AND SU, W. J. Gaussian differential privacy. *arXiv preprint arXiv:1905.02383* (2019).

[12] DWORK, C., MCSHERRY, F., NISSIM, K., AND SMITH, A. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference* (2006), Springer, pp. 265–284.

[13] JAGIELSKI, M., THAKKAR, O., TRAMER, F., IPPOLITO, D., LEE, K., CARLINI, N., WALLACE, E., SONG, S., THAKURTA, A., PAPERNOT, N., ET AL. Measuring forgetting of memorized training examples. *arXiv preprint arXiv:2207.00099* (2022).

[14] JAGIELSKI, M., ULLMAN, J., AND OPREA, A. Auditing differentially private machine learning: How private is private sgd? *Advances in Neural Information Processing Systems 33* (2020), 22205–22216.

[15] JAYARAMAN, B., AND EVANS, D. Evaluating differentially private machine learning in practice. In *USENIX Security Symposium* (2019).

[16] KAIROUZ, P., OH, S., AND VISWANATH, P. The composition theorem for differential privacy. In *International conference on machine learning* (2015), PMLR, pp. 1376–1385.

[17] KATZ, D., BAPTISTA, J., AZEN, S., AND PIKE, M. Obtaining confidence intervals for the risk ratio in cohort studies. *Biometrics* (1978), 469–474.

[18] KOSKELA, A., JÄLKÖ, J., AND HONKELA, A. Computing tight differential privacy guarantees using fft. In *International Conference on Artificial Intelligence and Statistics* (2020), PMLR, pp. 2560–2569.

[19] LU, F., MUNOZ, J., FUCHS, M., LEBLOND, T., ZARESKY-WILLIAMS, E., RAFF, E., FERRARO, F., AND TESTA, B. A general framework for auditing differentially private machine learning. *arXiv preprint arXiv:2210.08643* (2022).

[20] MADDOCK, S., SABLAYROLLES, A., AND STOCK, P. Canife: Crafting canaries for empirical privacy measurement in federated learning. *arXiv preprint arXiv:2210.02912* (2022).

[21] MIRONOV, I. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)* (2017), IEEE, pp. 263–275.

[22] NASR, M., HAYES, J., STEINKE, T., BALLE, B., TRAMÈR, F., JAGIELSKI, M., CARLINI, N., AND TERZIS, A. Tight auditing of differentially private machine learning. *arXiv preprint arXiv:2302.07956* (2023).

[23] NASR, M., SHOKRI, R., AND HOUMANSADR, A. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)* (2019), IEEE, pp. 739–753.

[24] NASR, M., SONGI, S., THAKURTA, A., PAPERNOT, N., AND CARLINI, N. Adversary instantiation: Lower bounds for differentially private machine learning. In *2021 IEEE Symposium on security and privacy (SP)* (2021), IEEE, pp. 866–882.

[25] SHOKRI, R., STRONATI, M., SONG, C., AND SHMATIKOV, V. Membership inference attacks against

machine learning models. In *2017 IEEE symposium on security and privacy (SP)* (2017), IEEE, pp. 3–18.

[26] STEVENS, T., NGONG, I. C., DARAIS, D., HIRSCH, C., SLATER, D., AND NEAR, J. P. Backpropagation clipping for deep learning with differential privacy. *arXiv preprint arXiv:2202.05089* (2022).

[27] TRAMÈR, F., TERZIS, A., STEINKE, T., SONG, S., JAGIELSKI, M., AND CARLINI, N. Debugging differential privacy: A case study for privacy auditing. *arXiv preprint arXiv:2202.12219* (2022).

[28] TRAMER, F., TERZIS, A., STEINKE, T., SONG, S., JAGIELSKI, M., AND CARLINI, N. Debugging differential privacy: A case study for privacy auditing. *arXiv preprint arXiv:2202.12219* (2022).

[29] WANG, J. T., MAHLOUJIFAR, S., WU, T., JIA, R., AND MITTAL, P. A randomized approach for tight privacy accounting. *arXiv preprint arXiv:2304.07927* (2023).

[30] WANG, Y., DING, Z., KIFER, D., AND ZHANG, D. Checkdp: An automated and integrated approach for proving differential privacy or finding precise counterexamples. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* (2020), pp. 919–938.

[31] WASSERMAN, L., AND ZHOU, S. A statistical framework for differential privacy. *Journal of the American Statistical Association 105*, 489 (2010), 375–389.

[32] YE, J., AND SHOKRI, R. Differentially private learning needs hidden state (or much faster convergence). *arXiv preprint arXiv:2203.05363* (2022).

[33] YEOM, S., GIACOMELLI, I., FREDRIKSON, M., AND JHA, S. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)* (2018), IEEE, pp. 268–282.

[34] ZAGORUYKO, S., AND KOMODAKIS, N. Wide residual networks. In *British Machine Vision Conference 2016* (2016), British Machine Vision Association.

[35] ZANELLA-BÉGUELIN, S., WUTSCHITZ, L., TOPLE, S., SALEM, A., RÜHLE, V., PAVERD, A., NASERI, M., AND KÖPF, B. Bayesian estimation of differential privacy. *arXiv preprint arXiv:2206.05199* (2022).

A Using PLD to approximate the trade off function

PLD does not have a closed-form trade-off function, but we can evaluate empirically a lower bound for a given FPR. We approximate the trade-off function using Algorithm 4, which will give us a looser bound for ε, however, the difference is in the order of 10^{-1}. Figure 13 compares the approximation approach in Algorithm 4 to the optimal trade-off function, as we can see, even with 10 approximations we can get a good estimation of the trade-off function. In Section 6, we will present experiments that provide lower bounds for a single step of DP-SGD and multiple steps, they will use the GDP and PLD formulations, respectively.

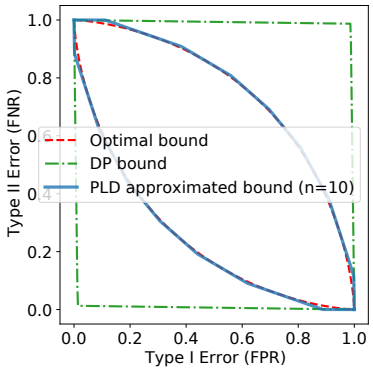


Figure 13: Comparison of the trade-off functions when using an approximation of PLD compared to the optimal curve.

Algorithm 4 Approximating a lower bound on trade-off function using PLD

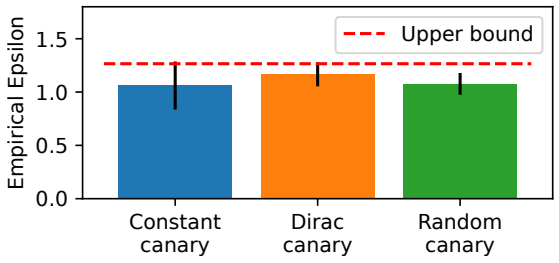
Args: $f_{\mathcal{M}}$ privacy analysis function (outputs ε for a given δ), n number of approximation lines, δ target delta in privacy analysis
$\Delta \leftarrow n$ linearly spaced points between $[\delta, 1-\delta]$
for $\delta' \in \Delta$ **do**
  $\hat{\varepsilon} \leftarrow f_{\mathcal{M}}(\delta')$
  $l_{\delta'}(x) := \max(0, 1-\delta'-(xe^{\hat{\varepsilon}}), e^{-\hat{\varepsilon}(1-\delta'-x)})$
end for
$l(x) := \min_{\delta' \in \Delta} l_{\delta'}(x)$
return l

B Effect Of Auditing Choices

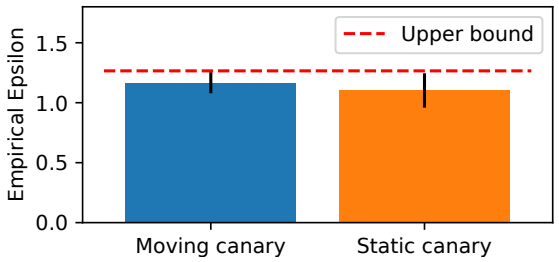
B.1 Choosing a Canary Gradient

We first investigate how the canary gradient affects the estimated privacy bound. We construct the canary gradient in three ways:

1. Dirac canary: All gradient values are zero except at a single index.
2. Constant canary: All gradient values have the same value.
3. Random canary: Gradient sampled from a Gaussian.



(a) We compare three different ways to create a canary gradient. The Dirac canary gradient, with zeros everywhere except for a single position which has a value set to the clipping norm, slightly outperforms other approaches.



(b) At each iteration we can either insert the same (static) canary gradient, or compute a new canary gradient (moving). There is little difference between these two approaches in terms of the ε lower bound we can find.

Figure 14: How design decisions for the canary gradient change the lower bound we compute for ε with f-DP.

In each setting, gradient values are re-scaled such that the canary gradient has a maximum norm equal to the clipping norm. We also measure if there is a difference between using the same canary gradient at each iteration of DP-SGD or creating a new canary gradient. For example, in the Dirac canary we would randomly sample a new index to set to the clipping norm at each update. Results are shown in Fig. 14; using a Dirac canary that is reset at each update performs best.

B.2 Choosing a Canary Input in The White-box Setting

We evaluate how different types of input canaries can affect auditing in a white-box setting. We construct the canaries in four different ways:

1. **Mislabeled example**: We select a random example from the test dataset of the model and we select a random label (that is not equal to the original label).
2. **Blank example**: We craft an input where all dimensions of the input are equal to zero.
3. **Adversarial example**: We apply Projected Gradient Descent (PGD) to generate adversarial example on a random example from the test dataset.

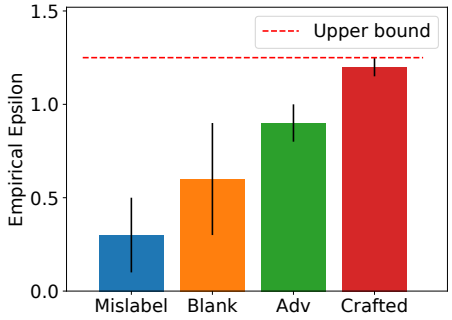


Figure 15: Comparison of the different canary crafting approaches in input space for CIFAR-10 dataset using WRN architecture.

4. **Crafted example**: We use Algorithm 3 to generate an input example.

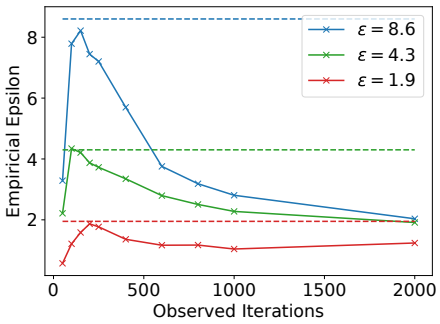
As we saw from our gradient experiment using the dot product between the privatized gradient and the canary gradient is a sufficient metric for auditing DP-SGD. Therefore we use the same idea in our crafting algorithm (Algorithm 3) in input space. We look for a canary such that its gradient is orthogonal to other gradients in the training batch. However, we cannot use the example in the training batch directly to craft such an example as it will violate the DP-assumptions (the adversary cannot have access to non-noisy gradients). Therefore, we assume the adversary has access to an example from the same distribution as the training dataset and uses that data to estimate the gradient of the model on the training example. Then, it crafts an example such that its gradient is orthogonal to the estimated gradient. Thus, creating a gradient that is significantly different from other examples in the batch and its presence can be detected.

Figure 15 compares the effectiveness of different input canaries in the white-box setting. As can be seen, using our canary crafting approach we can achieve significantly tighter bounds on DP-SGD compared to other canary crafting strategies. Unfortunately, it is not trivial to extend either the adversarial example or our crafting approach to the black-box setting and therefore we do not use them in the black-box experiments.

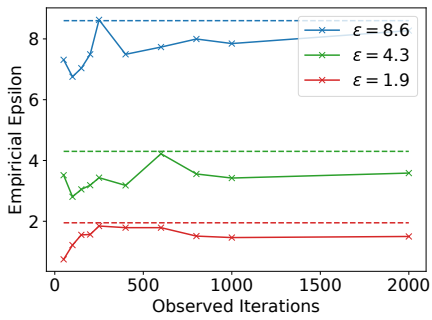
C The effect of model related parameters

C.1 Do earlier training steps leak more information?

Recent work on measuring the amount of privacy leaked when training convex models with DP has shown that the true amount of leakage plateaus as the model converges [3, 32]. This means that after a certain number of steps, training for more iterations does not consume any more of the privacy



(a) CIFAR-10 dataset with a WRN-16 model.



(b) Random dataset with a WRN-16 model.

Figure 16: Lower bounding ε with the input space attack (white-box setting) for different number of iterations of training and values of ε, using f-DP (ZB).

budget. The analysis for this result is specific to convex models; and we cannot prove such properties for the deep models. However, when can evaluate if a similar phenomenon holds empirically. We measure the lower bounds when we only use the first n iterations of the training in the *White-box access with Input Space Canaries* threat model. Results are shown in Figure 16. We find that indeed, the first part of training does leak more information than later in training on the CIFAR-10 dataset. However, when we evaluate the random dataset we do not see the same behavior (please note that, if we only look at a very small number (<100) of the iterations we get a very loose bound on ε because we do not have enough observations to have a sufficiently confident estimation). Understanding why we cannot lower bound from our audit becomes looser in later iterations requires further investigation which we leave for future work. Nevertheless, the results suggest that when we limit the adversary to canaries in the input space then model architecture, underlying dataset and the how well a model has been trained all have an effect on privacy leakage.

C.2 Do larger models leak more privacy?

Recent work has shown the larger models have a greater capacity to memorize training data verbatim [7]. We investigate if the same trend holds when training with DP-SGD by comparing lower bounds on a WRN-16 and WRN-40 model. Results are shown in Figure 17. Interestingly, if one was to use a non f-DP auditing method, one would make an incorrect conclusion that the WRN-16 leaks more than the WRN-40. Using our f-DP auditing method, we identify that, indeed, the larger WRN-40 model leaks slightly more than the WRN-16 model.

C.3 Does augmentation multiplicity affect privacy lower bounds?

In De et al. [9], data augmentation was a key ingredient in achieving state-of-the-art results on CIFAR-10. In particular, De et al. use a data augmentation technique they term *augmentation multiplicity* (Augmult), where they augment a single example multiple times and compute the average gradient over these augmentations *before* clipping. They find that increasing the Augmult value (number of augmentations) improves performance; increasing this value does not increase the privacy cost as it does not change the sensitivity of the privatized gradient to any single example in the batch. In Fig. 18, we measure lower bounds at different Augmult values, and observe no clear trend between lower bounds and the value for Augmult.

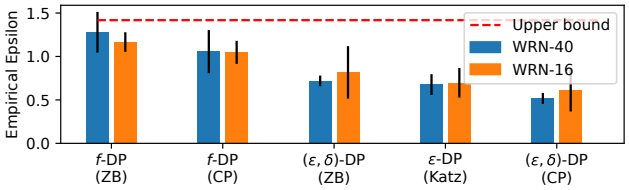


Figure 17: Comparison of how model architecture (WRN-16 and -40) affects the ε lower bound.

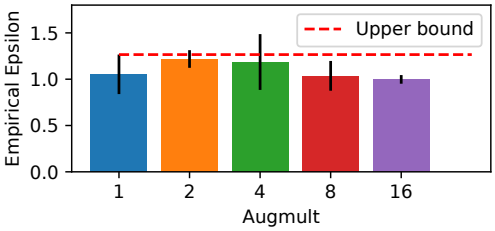


Figure 18: How the value for Augmult [9] affects the ε lower bound.