# OPTIMIZING CANARIES FOR PRIVACY AUDITING WITH METAGRADIENT DESCENT

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

In this work we study *black-box privacy auditing*, where the goal is to lower bound the privacy parameter of a differentially private learning algorithm using only the algorithm's outputs (i.e., final trained model). For DP-SGD (the most successful method for training differentially private deep learning models), the canonical approach auditing uses *membership inference*—an auditor comes with a small set of special "canary" examples, inserts a random subset of them into the training set, and then tries to discern which of their canaries were included in the training set (typically via a membership inference attack). The auditor's success rate then provides a lower bound on the privacy parameters of the learning algorithm. Our main contribution is a method for *optimizing* the auditor's canary set to improve privacy auditing, leveraging recent work on metagradient optimization (Engstrom et al., 2025). Our empirical evaluation demonstrates that by using such optimized canaries, we can improve empirical lower bounds for differentially private image classification models by over **1.8x** in certain instances. Furthermore, we demonstrate that our method is *DP-SGD agnostic* and *efficient*: canaries optimized for non-private SGD with a small model architecture remain effective when auditing larger models trained with DP-SGD.

## 1 INTRODUCTION

Differential privacy (DP) (Dwork et al., 2006) offers a rigorous mathematical framework for safeguarding individual data in machine learning. Within this framework, differentially private stochastic gradient descent (DP-SGD) (Abadi et al., 2016) has emerged as the standard for training differentially private deep learning models. Although DP-SGD provides theoretical upper bounds on privacy loss based on its hyperparameters, these guarantees are likely conservative, which mean they tend to overestimate the privacy leakage in practice (Nasr et al., 2023). In many cases, however, they may not reflect the true privacy leakage that occurs during training. To address this gap, researchers have developed empirical techniques known as privacy audits, which aim to establish lower bounds on privacy loss. In addition to quantifying real-world leakage, privacy auditing can also help detect bugs or unintended behaviors in the implementation of private algorithms (Tramer et al., 2022).

Providing a lower bound on the privacy leakage of an algorithm typically requires the auditor to guess some private information (*membership inference*) using a set of examples (also referred to as *canaries*). For example, in one-run auditing procedures (Steinke et al., 2023; Mahloujifar et al., 2024) (which we discuss further in Section 2.1.2), a random subset of these canaries is inserted into the training dataset, and once the model is trained, the auditor guesses which of these samples belong to the subset. While recent work has made significant progress in tightening these bounds through privacy auditing, the strongest results typically assume unrealistic levels of access or control of the private training process (broadly speaking, such settings fall under the term *white-box* auditing (Nasr et al., 2023)). In contrast, this work focuses on a more practical and restrictive *black-box* setting. Here, the auditor can only insert a subset of carefully crafted examples (called the *canaries*) into the training set and observe the model's output at the *last iterate* (without access to intermediate model states or gradient computations). In other words, the goal of black-box DP auditing reduces to performing membership inference on the canaries based on the *final model output*.

In this work, we study how to optimize canary samples for the purpose of black-box auditing in differentially private stochastic gradient descent (DP-SGD). Leveraging metagradient descent (Engstrom et al., 2025), we introduce an approach for crafting canaries specifically tailored for insertion into the training set during DP auditing. Through empirical evaluation on single-run auditing protocols for DP image classification models, we find that our method consistently yields canaries that surpass standard baselines by more than a factor of two in certain regimes. Notably, our algorithm is computationally efficient: although it involves running (non-private) SGD on a lightweight ResNet-9 architecture, the resulting canaries demonstrate strong performance even when deployed in larger models, such as Wide ResNets, trained under DP-SGD. Furthermore, this improvement persists whether DP-SGD is used for end-to-end training or private finetuning on pretrained networks.

### 1.1 Related Work

Early works in DP auditing (Ding et al., 2018; Bichsel et al., 2018) introduce methods that detect violations of formal DP guarantees, relying on a large number of runs to identify deviations from expected behavior. These techniques, however, are not directly applicable to the domain of differentially private machine learning, as they were developed for auditing simpler DP mechanisms. To tackle this issue, Jagielski et al. (2020) and Nasr et al. (2021) introduce new approaches based on membership inference attacks (MIA) to empirically determine privacy lower bounds for more complex algorithms like DP-SGD. Membership inference consists of accurately determining whether a specific sample was part of the model's training dataset. If the guesser (i.e. *attacker*) can reliably make accurate guesses, it suggests that the model retains information about individual samples observed during training, thereby comprising individuals' privacy. Hence, MIA can be used as a practical DP auditing tool in which lower bounds on how much privacy leakage has occurred can be directly be estimated from the success rate of the attacker.

**One-run auditing.** The first auditing methods for DP-SGD relied on many runs of the algorithm, making auditing very expensive and often impractical. To remedy this issue, Steinke et al. (2023); Mahloujifar et al. (2024) reduce the computational cost of auditing by proposing procedures that require only one training run. Kazmi et al. (2024) and Liu et al. (2025) further study how to incorporate stronger MIA methods to empirically improve auditing in this one-run setting. Similarly, Keinan et al. (2025) study the theoretical maximum efficacy of one-run auditing.

**Last-iterate auditing.** Our work builds on the aforementioned one-run auditing methods and focuses specifically on the *last-iterate* auditing regime, which restricts the auditor's access to just the final model weights after the last iteration of DP-SGD. Related work to this regime from Muthu Selva Annamalai (2024) investigates whether the analysis on the last iteration can be as tight as analysis on the sequence of all iterates. Meanwhile, Nasr et al. (2025) propose a heuristic that predicts empirical lower bounds derived from auditing the last iterate. Other works instead focus on on partial relaxations of the problem: Cebere et al. (2025) assume that the auditor can inject crafted-gradients, and Muthu Selva Annamalai & De Cristofaro (2024) audit models that initialized to worst-case parameters.

**Canaries Optimization.** Rather than proposing new auditing procedures, our work studies how to make existing ones more effective by focusing on optimizing canary sets for privacy auditing. Similarly, Jagielski et al. (2020) develop a method,*CLIPBKD*, that uses singular value decomposition to obtain canaries more robust to gradient clipping. Nasr et al. (2023) evaluate various procedures to optimize canaries for their *white-box* auditing experiments. To better audit differentially private federated learning, Maddock et al. (2023) craft an adversarial sample that is added to a client's dataset used to send model updates. Finally, in the context of auditing LLMs, Panda et al. (2025) proposes using tokens sequences not present in the training dataset as canaries, while Meeus et al. (2025) create canaries with a low-perplexity, in-distribution prefixes and high-perplexity suffixes.

**Metagradient computation.** Our work also makes use of recent advancements in computing *metagradients*, gradients of machine learning models' outputs with respect to their hyperparameters or other quantities decided on prior to training. Prior work on metagradient computation falls under two categories: *implicit differentiation* (Bengio, 2000; Koh & Liang, 2017; Rajeswaran et al., 2019; Finn et al., 2017; Lorraine et al., 2020; Chen & Hsieh, 2020; Bae et al., 2022) aims to approximate the metagradient. On one hand, approximating metagradients allows for scalability to large-scale metagradient computation; on the other, this approach loosens correctness guarantees and imposes restrictions on what learning algorithms can be used. In contrast, *explicit differentiation* directly computes metagradients using automatic differentiation. However, these works (Maclaurin et al., 2015; Micaelli & Storkey, 2021; Franceschi et al., 2017; Liu et al., 2018) are limited by their scalability to larger models and number of hyperparameters and by numerical instability. We leverage recent work by Engstrom et al. (2025), which takes the explicit approach, but addresses the aforementioned issues by proposing a scalable and memory-efficient method to computing metagradients.

## 2 PRELIMINARIES

Informally, differential privacy provides bounds on the extent to which the output distribution of a randomized algorithm $\mathcal{M}$ can change when a data point is removed or swapped out.

**Definition 2.1** ((Approximate-) Differential Privacy (DP) (Dwork et al., 2006))**.** A randomized algorithm $\mathcal{M} : \mathcal{X}^N \to \mathbb{R}$ satisfies $(\varepsilon, \delta)$-differential privacy if for all neighboring datasets $D, D'$ (i.e., all $D, D'$ such that $|D' \setminus D| = 1$ and for all outcomes $S \subseteq \mathbb{R}$ we have

$$P(\mathcal{M}(D) \in S) \leq e^{\varepsilon} P(\mathcal{M}(D') \in S) + \delta$$

In the context of machine learning, $\mathcal{M}$ would be a learning algorithm, and this definition requires the model to be insensitive to the exclusion of one training data point. In essence, it bounds the change in the output distribution of the model when trained on neighboring datasets. This implies that the model does not overly depend on any single sample observed.

Since the seminal work of Dwork et al. (2006), various relaxations of differential privacy have been proposed. Below, we define $f$-differential privacy, which we later reference when describing the auditing procedure proposed by Mahloujifar et al. (2024).

**Definition 2.2** ($f$-Differential Privacy (Dong et al., 2022))**.** A mechanism $\mathcal{M}$ is $f$-DP if for all neighboring datasets $\mathcal{S}, \mathcal{S}'$ and all measurable sets $T$ with $|\mathcal{S} \triangle \mathcal{S}'| = 1$, we have

$$\Pr[\mathcal{M}(\mathcal{S}) \in T] \leq \bar{f}\left(\Pr[\mathcal{M}(\mathcal{S}') \in T]\right). \tag{1}$$

Importantly, $f$-DP relates to approximate DP in the following way:

**Proposition 1.** A mechanism is $(\varepsilon, \delta)$-DP if it is $f$-DP with respect to $\bar{f}(x) = e^{\varepsilon} x + \delta$, where $\bar{f}(x) = 1 - f(x)$.

While a wide range of methods for adding differentially private guarantees to machine learning algorithms have been proposed over the years, DP-SGD (Abadi et al., 2016) has been established as one of the de facto algorithms for training deep neural networks with DP. At a high-level, DP-SGD makes SGD differentially private by modifying it in the following ways: (1) gradients are clipped to some maximum Euclidean norm and (2) random noise is added to the clipped gradients prior to each update step. In Algorithm 1, we present DP-SGD in detail.

### 2.1 AUDITING DIFFERENTIAL PRIVACY

Differentially private algorithms are accompanied by analysis upper bounding the DP parameters $\varepsilon$ and $\delta$. Privacy auditing instead provides an empirical *lower bound* on these parameters. In this work, we focus on a specific formulation of privacy audits: *last-iterate, black-box, one-run* auditing.

#### 2.1.1 LAST-ITERATE BLACK-BOX AUDITING

Our work focuses on *last-iterate black-box* auditing, where the auditor can only insert samples (i.e., canaries) into the training set and can only access the resulting model at the final training iteration. We note that, in contrast, previous works have also studied white-box settings. While the exact assumptions made in this setting can vary (Nasr et al., 2021; 2023; Steinke et al., 2023; Koskela & Mohammadi, 2025), it can be characterized as having fewer restrictions (e.g., access to intermediate training iterations or the ability to inject and modify gradients). While auditing in white-box settings generally leads to higher lower bound estimates due to the strength of the auditor, its assumptions are often far less realistic than those made in black-box auditing.

**Algorithm 1:** Differentially Private Stochastic Gradient Descent (DP-SGD)

**Input:** $x \in \mathcal{X}^n$
**Requires:** Loss function $f : \mathbb{R}^d \times \mathcal{X} \to \mathbb{R}$
**Parameters:** Number of iterations $\ell$, learning rate $\eta$, clipping threshold $c > 0$, noise multiplier $\sigma > 0$, sampling probability $q \in (0, 1]$

Initialize $w_0 \in \mathbb{R}^d$;
**for** $t = 1, \ldots, \ell$ **do**
  Sample $S^t \subseteq [n]$ where each $i \in [n]$ is included independently with probability $q$;
  Compute $g_i^t = \nabla_{w^{t-1}} f(w^{t-1}, x_i) \in \mathbb{R}^d$ for all $i \in S^t$;
  Clip $\tilde{g}_i^t = \min\left\{1, \frac{c}{\|g_i^t\|_2}\right\} \cdot g_i^t \in \mathbb{R}^d$ for all $i \in S^t$;
  Sample $\xi^t \in \mathbb{R}^d$ from $\mathcal{N}(0, \sigma^2 c^2 I)$;
  Sum $\tilde{g}^t = \xi^t + \sum_{i \in S^t} \tilde{g}_i^t \in \mathbb{R}^d$;
  Update $w^t = w^{t-1} - \eta \cdot \tilde{g}^t \in \mathbb{R}^d$;
**Output:** $w^0, w^1, \ldots, w^\ell$

**Algorithm 2:** Black-box Auditing - One Run (Steinke et al., 2023)

**Input:** probability threshold $\tau$, privacy parameter $\delta$, training algorithm $\mathcal{A}$, dataset $D$, set of $m$ canaries $C = \{c_1, \ldots, c_m\}$
**Requires:** scoring function `score`
**Parameters:** number of positive and negative guesses $k_+$ and $k_-$

Randomly split canaries $C$ into two equally-sized sets $C_{\text{IN}}$ and $C_{\text{OUT}}$
Let $S = \{s_i\}_{i=1}^m$, where $s_i = \begin{cases} 1 & \text{if } c_i \in C_{\text{IN}} \\ -1 & \text{if } c_i \in C_{\text{OUT}} \end{cases}$
Train model $w \leftarrow \mathcal{A}(D \cup C_{\text{IN}})$
Compute vector of scores $Y = \{\texttt{score}(w, c_i)\}_{i=1}^m$
Sort scores in ascending order $Y' \leftarrow \texttt{sort}(Y)$
Construct vector of guesses $T = \{t_i\}_{i=1}^m$, where $t_i =$
$$\begin{cases} 1 & \text{if } Y_i \text{ is among the top } k_+ \text{ scores in } Y \text{ (i.e., } Y_i \geq Y'_{m-k_+}) \text{ // guess } c_i \in C_{\text{IN}} \\ -1 & \text{if } Y_i \text{ is among the bottom } k_- \text{ scores in } Y \text{ (i.e., } Y_i \leq Y'_{k_-}) \text{ // guess } c_i \in C_{\text{OUT}} \\ 0 & \text{otherwise // abstain} \end{cases}$$
Compute empirical epsilon $\tilde{\varepsilon}$ (i.e., find the largest $\tilde{\varepsilon}$ such that $S$, $T$, $\tau$, and $\delta$ satisfy Theorem 1)
**Output:** $\tilde{\varepsilon}$

#### 2.1.2 ONE-RUN AUDITING

Early works (Jagielski et al., 2020; Tramer et al., 2022; Nasr et al., 2023) design privacy auditing "attacks" that align with the definition of DP, which bounds the difference in outputs on neighboring datasets that differ by one sample. These audits detect the presence (or absence) of an individual sample over hundreds—if not, thousands—of runs of DP-SGD. The auditing procedure then gives a lower bound on $\varepsilon$ based on the true and false positive rates of the membership inference attacks.

While effective, these multi-run auditing procedures are computational expensive. Consequently, Steinke et al. (2023) propose an alternative procedure that requires only *one* training run. Their strategy inserts multiple canary examples and obtains a lower bound based on how well an attacker can guess whether some canary was used in training. While one-run auditing can sacrifice bound tightness, its ability to audit without multiple runs of DP-SGD make it much more efficient and therefore, practical for larger models. In our work, we consider two primary auditing procedures:

**(1)** Steinke et al. (2023) introduce the concept of privacy auditing using one training run. Given some set of canaries $C$, samples are randomly sampled from $C$ with probability $\frac{1}{2}$ and inserted into the training set. Once the model is trained, the auditor guesses which samples in $C$ were or were not included in the training set. The auditor can make any number of guesses or abstain. We present

**Algorithm 3:** Black-box Auditing - One Run (Mahloujifar et al., 2024)

**Input:** privacy parameter $\delta$, training algorithm $\mathcal{A}$, dataset $D$, set of $m$ canaries $C = \{c_1, \ldots, c_m\}$
**Requires:** scoring function $\texttt{score}$
**Parameters:** number of guesses $k$

Randomly split canaries $C$ into two equally-sized sets $C_{\text{IN}}$ and $C_{\text{OUT}}$
Create disjoint canary sets $E = \{e_i\}_{i=1}^{m/2}$ by randomly pairing canaries from $C_{\text{IN}}$ and $C_{\text{OUT}}$ such that $e_i = (c_{i,1}, c_{i,2})$ for $c_{i,1} \in C_{\text{IN}}$ and $c_{i,2} \in C_{\text{OUT}}$ (each canary $c \in C$ appears in **exactly** one set $e_i$)
Train model $w \leftarrow \mathcal{A}(D \cup C_{\text{IN}})$
Compute vector of scores $Y = \{|\texttt{score}(w, c_{i,1}) - \texttt{score}(w, c_{i,2})|\}_{i=1}^{m/2}$
Sort scores in ascending order $Y' \leftarrow \texttt{sort}(Y)$
Construct vector of guesses $T = \{t_i\}_{i=1}^{m/2}$, where

$$t_i = \begin{cases} 1 & \text{if } Y_i \text{ is among the top } k \text{ values in } Y \text{ (i.e., } Y_i \geq Y'_{m-k}) \\ & \quad \text{and } \texttt{score}(w, c_{i,1}) > \texttt{score}(w, c_{i,2}) \texttt{// guess } c_{i,1} \in C_{\text{IN}} \\ -1 & \text{if } Y_i \text{ is among the top } k \text{ values in } Y \text{ (i.e., } Y_i \geq Y'_{m-k}) \\ & \quad \text{and } \texttt{score}(w, c_{i,1}) \leq \texttt{score}(w, c_{i,2}) \texttt{// guess } c_{i,2} \in C_{\text{IN}} \\ 0 & \text{otherwise } \texttt{// abstain} \end{cases}$$

Let number of correct guesses $k' = \sum_{i=1}^{m/2} \mathbb{1}\{t_i = 1\}$
Compute empirical epsilon $\tilde{\varepsilon}$ (i.e., find the largest $\tilde{\varepsilon}$ whose corresponding $f$-DP function $f$ passes Algorithm 4 for $m$, $k$, $k'$ $\tau$, and $\delta$.)
**Output:** $\tilde{\varepsilon}$

**Algorithm 4:** Upper bound probability of making correct guesses (Mahloujifar et al., 2024)

**Input:** probability threshold $\tau$, functions $f$ and $f^{-1}$, number of guesses $k$, number of correct guesses $k'$, number of samples $m$, alphabet size $s$

$\forall 0 < i < k'$ set $h[i] = 0$, and $r[i] = 0$
Set $r[k'] = \tau \cdot \frac{c}{m}$
Set $h[k'] = \tau \cdot \frac{c' - c}{m}$
**for** $i \in [k' - 1, \ldots, 0]$ **do**
  $h[i] = (s - 1) f^{-1}(r[i + 1])$
  $r[i] = r[i + 1] + \frac{i}{k - i} \cdot (h[i] - h[i + 1])$
**if** $r[0] + h[0] \geq \frac{k}{m}$ **then**
  Return True (probability of $k'$ correct guesses (out of $k$) is less than $\tau$)
**else**
  Return False (probability of having $k'$ correct guesses (out of $k$) could be more than $\tau$)

this procedure in Algorithm 2. The final lower bound on $\varepsilon$ is determined using Theorem 1, which is based on the total number of canaries, the number of guesses, and the number of correct guesses.

**Theorem 1** (Analytic result for approximate DP (Steinke et al., 2023))**.** Suppose $\mathcal{A} : \{-1, 1\}^m \to \{-1, 0, 1\}^m$ satisfy $(\varepsilon, \delta)$-DP. Let $S \in \{-1, 1\}^m$ be uniformly random and $T = \mathcal{A}(S)$. Suppose $\mathbb{P}[\|T\|_1 \leq r] = 1$. Then, for all $v \in \mathbb{R}$,

$$\mathbb{P}_{\substack{S \leftarrow \{-1,1\}^m \\ T \leftarrow M(S)}} \left[ \sum_{i=1}^{m} \max\{0, T_i \cdot S_i\} \geq v \right] \leq f(v) + 2m\delta \cdot \max_{i \in \{1, \ldots, m\}} \left\{ \frac{f(v - i) - f(v)}{i} \right\},$$

where

$$f(v) := \mathbb{P}_{\tilde{W} \leftarrow \text{Binomial}\left(r, \frac{e^{\varepsilon}}{e^{\varepsilon} + 1}\right)} \left[ \tilde{W} \geq v \right].$$

At a very high level, $\mathcal{A}$ is DP-SGD, which takes in as input some set of $m$ canaries that are labeled ($S \in \{-1, 1\}^m$) as being included or excluded from the training set. The auditor uses the output of

DP-SGD to produce a vector of guesses $T \in \{-1, 0, 1\}^m$ for the $m$ canaries. Theorem 1 bounds the probability of making at least $v$ correct guesses ($\sum_{i=1}^{m} \max\{0, T_i \cdot S_i\} \geq v$, where $T_i \cdot S_i = 1$ if the guess is correct). More informally, this theorem bounds the success rate (number of correct guesses) of the auditor assuming the parameter $\varepsilon$. Practically speaking, one runs binary search (Steinke et al., 2023, Appendix D) to estimate the largest $\varepsilon$ such that Theorem 1 still holds.

**(2)** Mahloujifar et al. (2024) propose an alternative auditing procedure that empirically achieves tighter privacy estimates in the white-box setting. In their guessing game, the set of canaries $C$ is randomly partitioned in disjoint sets. One canary is sampled from each set and inserted into the training set. Again, once the model is trained with DP-SGD, the auditor must make guesses. Unlike in Steinke et al. (2023), however, the auditor must guess which canary out of each set was included in training. Algorithm 3 presents this procedure for canary sets of size 2.

Similar to Steinke et al. (2023), the final lower bound on $\varepsilon$ is determined based on the total number of canary sets, the number of guesses, and the number of correct guesses. At a high level, Mahloujifar et al. (2024) first construct a set of candidate values for $\varepsilon$ and a corresponding $f$-DP function for each. Using Algorithm 4, they then run a hypothesis test, with probability threshold $\tau$, for the number of correct guesses (i.e., output of Algorithm 4) occurring given function $f$. The final empirical lower bound is the maximum $\varepsilon$ among those corresponding to the functions $f$ that pass Algorithm 4.

**Scoring function.** Finally, to determine membership for either procedure, the auditor must first choose some `score` function $s(\cdot)$ from the training process. In the black-box setting for image classification models, one natural choice for $s(\cdot)$ is to use negative cross-entropy loss (Steinke et al., 2023). When $s(w, x)$ is large (i.e., cross-entropy loss is small) for some canary $x$ and model $w$, the auditor guesses that $x$ was included in training, and vice-versa. In Section 4, we provide more details about how we use the score function for Algorithms 2 and 3.

## 3 Canary Optimization with Metagradient Descent

For a fixed black-box auditing algorithm $\texttt{BBaudit} : (\tau, \delta, \mathcal{A}, D, C) \rightarrow \widetilde{\varepsilon}$ (e.g., Algorithm 2 or 3), the main degree of freedom available to the auditor is the choice of canary set $C$. Typically, one chooses $C$ to be a random subset of the training dataset $D$, or a random set of mislabled examples (Steinke et al., 2023; Mahloujifar et al., 2024). A natural question to ask is whether such choices are (near-)optimal; in other words, *can we significantly improve the efficacy of a given auditing algorithm by carefully designing the canary set* $C$?

In this section, we describe an optimization-based approach to choosing the canary set. At a high level, our goal is to solve an optimization problem of the form

$$\max_{C} \texttt{BBaudit}(\tau, \delta, \mathcal{A}, D, C), \tag{2}$$

where `BBaudit` is the (fixed) differential privacy auditing algorithm, $\tau$ and $\delta$ are the privacy parameters, $\mathcal{A}$ is the learning algorithm (e.g., DP-SGD), $D$ is the dataset, and $C$ is the set of canary samples. The high-dimensional nature of this problem (e.g., for CIFAR-10, $C \in \mathbb{R}^{m \times 32 \times 32 \times 3}$) makes it impossible to exhaustively search over all possible canary sets $C$.

Instead, the main idea behind our approach is to use *gradient descent* to optimize the canary set $C$. To do so, we first design a surrogate objective function to `audit` by leveraging the connection between membership inference and differential privacy auditing. We then use recent advances in *metagradient* computation (Engstrom et al., 2025) to optimize this surrogate objective with respect to the canary set $C$.

**Key primitive: metagradient descent.** A metagradient is a gradient taken *through* the process of training a machine learning model (Maclaurin et al., 2015; Domke, 2012; Bengio, 2000; Baydin & Pearlmutter, 2014). Specifically, given a learning algorithm $\mathcal{A}$, a (continuous) design parameter $z$ (e.g., learning rate, weight decay, data weights, etc.), and a loss function $\phi$, the metagradient $\nabla_z \phi(\mathcal{A}(z))$ is the gradient of the final loss $\phi$ with respect to the design parameter $z$ (see Figure 1). For very small-scale learning algorithms (e.g., training shallow neural networks), one can compute metagradients by backpropagating through the entire model training process.

$$\begin{tikzcd} z \arrow[r, "\mathcal{A}"] & w \arrow[r, "\phi"] & \phi(w) \arrow[ll, bend right, dashed, "\nabla_z \phi(\mathcal{A}(z))"'] \end{tikzcd}$$

Training setup  Trained model  Observed behavior

Figure 1: An illustration of the metagradient. We embed the canaries into a continuous metaparameter $z \in \mathbb{R}^{m \times H \times W \times C}$ with one coordinate per training data pixel. All aspects of the learning process other than $z$—the base training data, optimizer hyperparameters, etc.—are baked into the learning algorithm $\mathcal{A}$. The metaparameter thus defines a model $w = \mathcal{A}(z)$, which we use to compute an output metric $\phi(w)$. The metagradient $\nabla_z \phi(\mathcal{A}(z))$ is the gradient of the metric with respect to the metaparameter $z$.

While this approach does not scale directly to larger-scale training routines, the recent work of Engstrom et al. (2025) proposes a method to compute metagradients efficiently and at scale. The method, called REPLAY, enables gradient-based optimization of data importance weights, training hyperparameters, and—most relevant to our setting—poisonous training data that hurts overall model accuracy. To tackle the latter setting, Engstrom et al. (2025) show how to compute metagradients of a model's final test loss with respect to the pixels in its training data. Leveraging this method, we will assume that we can efficiently compute metagradients of any differentiable loss function with respect to the pixels of any number of training data points.

**Surrogate objective function.** Even with the ability to compute metagradients efficiently, the optimization problem in (2) is still challenging to solve in the black-box setting. First, the objective function BBaudit has explicit non-differentiable components (e.g., thresholding). Second, taking the metagradient requires more fine-grained access to the model training process than simply observing the final model outputs.

To address both these challenges, we design a surrogate objective function that approximates the original objective (2), inspired by the connection between black-box privacy auditing and membership inference. In particular, inspecting Algorithms 2 and 3, we observe that in both algorithms, we randomly split the canary set into two sets $C_{\text{IN}}$ and $C_{\text{OUT}}$; trains a model on $D \cup C_{\text{IN}}$; and runs a membership inference attack to distinguish between samples in $C_{\text{IN}}$ and $C_{\text{OUT}}$. Intuitively, a good canary sample $z_i$ should thus satisfy the following properties: (1) **memorizability**: if $z_i \in C_{in}$, the model should have *low* loss on $z_i$ and (2) **non-generalizability**: if $z_i \in C_{out}$, the model should have *high* loss on $z_i$. These properties motivate the following simple surrogate objective function:

$$\phi(w) = \sum_{i=1}^{m} \left(\mathbf{1}\{z_i \in C_{in}\} - \mathbf{1}\{z_i \in C_{out}\}\right) \cdot \mathcal{L}(w, z_i), \tag{3}$$

where $\mathcal{L}$ is the training loss (i.e., cross-entropy) and $\mathbf{1}\{\cdot\}$ is the indicator function.[1] Finally, to optimize this objective in the black-box setting, we consider a "standard" (i.e., not differentially private) training algorithm $\mathcal{A}$, and then transfer the resulting canaries to whatever learning algorithm we are auditing.

**Optimizing canaries with (meta-)gradients.** Our final optimization process (given in more detail in Algorithm 5) proceeds in $T > 1$ *metasteps*. Let $D$ be the set of non-canaries (e.g., the CIFAR-10 dataset) and $C$ be the set of canaries (i.e., metaparameters $z$) we are optimizing. During each *metastep* $t$, we randomly partition the canaries $C$ into two sets $C_{\text{IN},t}$ and $C_{\text{OUT},t}$, and randomly sample a model initialization and data ordering which define a learning algorithm $\mathcal{A}$. After training

[1] We note that in this case, $\phi$ depends on the model weights $w$ but also has a direct dependence on the canary set (i.e., the metaparameters $z$), making Figure 1 a slight over-simplification. In practice, we can still use the law of total derivative to compute the gradient of $\phi$ with respect to $z$, since $\nabla_z \phi(z, \mathcal{A}(z)) = \frac{\partial \phi}{\partial w} \cdot \nabla_z \mathcal{A}(z) + \frac{\partial \phi}{\partial z}$.

**Algorithm 5:** Metagradient Canary Optimization

**Input:** dataset $D$
**Requires:** training algorithm $\mathcal{A}$, loss function $\mathcal{L}$
**Parameters:** number of canaries $m$, number of meta-iterations $N$

```
1 Initialize canaries C_0 = {c_1, ..., c_m}
2 for t ← 0 to N − 1 do
3     Randomly split C_t into two equally-sized sets: C_IN,t and C_OUT,t
4     Train model: w_t ← A(D ∪ C_IN,t)
5     Compute loss gap φ(w_t) = L(w_t, C_IN,t) − L(w_t, C_OUT,t)
6     Compute gradient w.r.t. canaries: ∇_{C_t} ← REPLAY(w_t, φ(θ_t))
7     Update canaries: C_{i+1} ← update(C_i, ∇_{C_i})
```

**Output:** optimized canaries $C_N$

a model $w = \mathcal{A}(z)$, we take a gradient step to minimize the objective (equation 3) with respect to the canary set $C$. By repeating this process several times (essentially running stochastic gradient descent across random seeds and random data orderings partitionings of the canary set), we obtain a set of canary examples that are robustly memorizable and non-generalizable.

## 4 Empirical Evaluation

### 4.1 Setup

**Audited models.** Following prior work (Nasr et al., 2023; Steinke et al., 2023; Mahloujifar et al., 2024), we audit Wide ResNet models (Zagoruyko & Komodakis, 2016) trained on CIFAR-10 (Krizhevsky et al., 2009) with DP-SGD. We use the Wide ResNet 16-4 architecture proposed by De et al. (2022), which they modify for DP training, and train the model using the `JAX-Privacy` package (Balle et al., 2025).

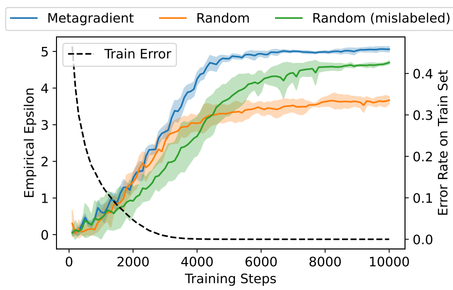
To audit the models, we use canary sets of size $m = 1000$. To remain consistent with Steinke et al. (2023) and Mahloujifar et al. (2024), where $C$ is sampled from the training set, we have in total $r = 49000$ non-canaries training images for CIFAR-10. Thus, in total, $n = 49500$ images are used in training for any given run. We run DP-SGD on models both initialized randomly and pretrained nonprivately (i.e., DP-finetuning). For DP-finetuning experiments, we use CINIC-10 (Darlow et al., 2018), which combines images from CIFAR-10 with images from ImageNet (Deng et al., 2009) that correspond to the classes in CIFAR-10. For our pretraining dataset, we use CINIC-10 with the CIFAR-10 images removed. We present hyperparameters in Table 2 of the appendix.

**Baselines.** We compare our method against canaries randomly sampled from the training set (Steinke et al., 2023; Mahloujifar et al., 2024), as well as canaries that have been mislabeled (Nasr et al., 2023; Steinke et al., 2023).

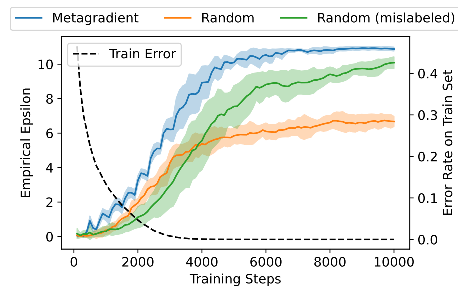
### 4.2 Results

We first evaluate our metagradient canaries when auditing models trained with *non-private* SGD. In Figure 2, we plot the empirical epsilon estimated by the auditing procedures introduced in Steinke et al. (2023) and Mahloujifar et al. (2024) against the number of steps that the Wide ResNet 16-4 model is trained for. We observe that even when applied on different model architectures (i.e., transferring from ResNet-9 to WRN 16-4), our metagradient canaries perform strongly. Using the auditing procedure of Mahloujifar et al. (2024), for example, our canaries outperform the two baselines across any number of training steps.

Having verified that our metagradient canaries work properly for auditing SGD, we now evaluate their effectiveness in auditing *DP-SGD*. In Table 1, we present our main results for both DP training (i.e., training from scratch) and DP finetuning (i.e., first pretraining non-privately). We find that our method performs the best, exceeding the empirical epsilon of baseline canaries, regardless of the auditing procedure used ($\approx 25\%$ for Steinke et al. (2023) and $\approx 75\%$ for Mahloujifar et al. (2024)).




(a) Steinke et al. (2023)




(b) Mahloujifar et al. (2024)

Figure 2: We evaluate the effectiveness of our metagradient canaries for the purpose of auditing *non-private* SGD. We train a Wide ResNet 16-4 model on CIFAR-10 for $10k$ steps with each canary type, plotting the empirical epsilon when auditing the model at every 100 steps with the auditing procedures introduced by **(a)** Steinke et al. (2023) and **(b)** Mahloujifar et al. (2024). We take an average over 5 runs and plot an error band to denote $\pm 1$ standard deviation. For reference, we plot the training error of the model trained on our metagradient canaries (note that the training accuracy is approximately the same, regardless of choice of canary).

Table 1: We audit a Wide ResNet 16-4 model that has been trained with DP-SGD ($\varepsilon = 8.0$, $\delta = 10^{-5}$) on CIFAR-10 with the auditing parameters: $n = 49500$, $m = 1000$, and $r = 49000$. We present results for models **(a)** initialized from scratch and **(b)** pretrained on CINIC-10 (with CIFAR-10 images removed). We report the average empirical epsilon and standard error over 5 runs for auditing procedures introduced by **(1)** Steinke et al. (2023) and **(2)** Mahloujifar et al. (2024).

| | | (a) DP Training | | (b) DP Finetuning | |
|---|---|---|---|---|---|
| **Audit Procedure** | **Canary Type** | **Avg.** | **Std.** | **Avg.** | **Std.** |
| (1) Steinke et al. (2023) | random | 0.285 | 0.163 | 0.477 | 0.197 |
| | random mislabeled | 0.308 | 0.249 | 0.489 | 0.316 |
| | metagradient (*ours*) | **0.392** | 0.139 | **0.665** | 0.054 |
| (2) Mahloujifar et al. (2024) | random | 0.405 | 0.428 | 0.687 | 0.229 |
| | random mislabeled | 0.225 | 0.167 | 0.632 | 0.123 |
| | metagradient (*ours*) | **0.732** | 0.274 | **1.207** | 0.247 |

Moreover, this advantage holds even when evaluating on DP finetuning, despite our metagradient optimization process not using CINIC-10 for pretraining $w$ at each metagradient step of Algorithm 5.

## 5 Conclusion

We propose an efficient method for canary optimization that leverages metagradient descent. Optimizing for an objective tailored towards privacy auditing, our canaries significantly outperform standard canaries, which are sampled from the training dataset. Specifically, we show that despite being optimized for non-private SGD on a small ResNet model, our canaries work better on larger Wide ResNets for both DP-training and DP-finetuning. Using our canaries, we significantly tighten privacy bounds, nearly doubling the empirical epsilon. Furthermore, this improvement holds, regardless of whether DP-SGD is run from scratch or after non-private finetuning.

## 6 Reproducibility Statement

In our paper, we describe in detail the algorithms for both our metagradient optimization procedure (Algorithm 5) and the one-run auditing procedures from Steinke et al. (2023) (Algorithm 2) and Mahloujifar et al. (2024) (Algorithms 3 and 4). Additional implementation details and hyperparameters for running these algorithms are described in Appendix A. To ensure reproducibility of our results, we also provide code used for all our experiments in the supplementary materials. There are two main parts: code for metagradient canary optimization and code for training and auditing image classification models with DP-SGD. Instructions for running the code are available in an included README file. In the final version of our paper, we will include a link to a public repository.

## REFERENCES


Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger B Grosse. If influence functions are the answer, then what is the question? *Advances in Neural Information Processing Systems*, 35:17953–17967, 2022.

Borja Balle, Leonard Berrada, Zachary Charles, Christopher A Choquette-Choo, Soham De, Vadym Doroshenko, Dj Dvijotham, Andrew Galen, Arun Ganesh, Sahra Ghalebikesabi, Jamie Hayes, Peter Kairouz, Ryan McKenna, Brendan McMahan, Aneesh Pappu, Natalia Ponomareva, Mikhail Pravilov, Keith Rush, Samuel L Smith, and Robert Stanforth. JAX-Privacy: Algorithms for privacy-preserving machine learning in jax, 2025. URL `http://github.com/google-deepmind/jax_privacy`.

Atilim Gunes Baydin and Barak A Pearlmutter. Automatic differentiation of algorithms for machine learning. *arXiv preprint arXiv:1404.7456*, 2014.

Yoshua Bengio. Gradient-based optimization of hyperparameters. *Neural computation*, 12(8):1889–1900, 2000.

Benjamin Bichsel, Timon Gehr, Dana Drachsler-Cohen, Petar Tsankov, and Martin Vechev. Dp-finder: Finding differential privacy violations by sampling and optimization. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, pp. 508–524, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356930. doi: 10.1145/3243734.3243863. URL `https://doi.org/10.1145/3243734.3243863`.

Tudor Ioan Cebere, Aurélien Bellet, and Nicolas Papernot. Tighter privacy auditing of DP-SGD in the hidden state threat model. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=xzKFnsJIXL`.

Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In *International conference on machine learning*, pp. 1554–1565. PMLR, 2020.

Luke N Darlow, Elliot J Crowley, Antreas Antoniou, and Amos J Storkey. Cinic-10 is not imagenet or cifar-10. *arXiv preprint arXiv:1810.03505*, 2018.

Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Zeyu Ding, Yuxin Wang, Guanhong Wang, Danfeng Zhang, and Daniel Kifer. Detecting violations of differential privacy. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 475–489, 2018.

Justin Domke. Generic methods for optimization-based modeling. In *Artificial Intelligence and Statistics*, pp. 318–326. PMLR, 2012.

Jinshuo Dong, Aaron Roth, and Weijie J. Su. Gaussian differential privacy. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84(1):3–37, 2022.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pp. 265–284. Springer, 2006.

Logan Engstrom, Andrew Ilyas, Benjamin Chen, Axel Feldmann, William Moses, and Aleksander Madry. Optimizing ml training with metagradient descent. *arXiv preprint arXiv:2503.13751*, 2025.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. Forward and reverse gradient-based hyperparameter optimization. In *International conference on machine learning*, pp. 1165–1173. PMLR, 2017.

Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing differentially private machine learning: How private is private sgd? *Advances in Neural Information Processing Systems*, 33: 22205–22216, 2020.

Mishaal Kazmi, Hadrien Lautraite, Alireza Akbari, Qiaoyue Tang, Mauricio Soroco, Tao Wang, Sébastien Gambs, and Mathias Lécuyer. Panoramia: Privacy auditing of machine learning models without retraining. *Advances in Neural Information Processing Systems*, 37:57262–57300, 2024.

Amit Keinan, Moshe Shenfeld, and Katrina Ligett. How well can differential privacy be audited in one run? *arXiv preprint arXiv:2503.07199*, 2025.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. PMLR, 2017.

Antti Koskela and Jafar Aco Mohammadi. Auditing differential privacy guarantees using density estimation. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pp. 1007–1026. IEEE, 2025.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *University of Toronto*, 2009.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

Terrance Liu, Matteo Boglioni, Yiwei Fu, Shengyuan Hu, Pratiksha Thaker, and Zhiwei Steven Wu. Enhancing one-run privacy auditing with quantile regression-based membership inference. *arXiv preprint arXiv:2506.15349*, 2025.

Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International conference on artificial intelligence and statistics*, pp. 1540–1552. PMLR, 2020.

Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *International conference on machine learning*, pp. 2113–2122. PMLR, 2015.

Samuel Maddock, Alexandre Sablayrolles, and Pierre Stock. Canife: Crafting canaries for empirical privacy measurement in federated learning. In *ICLR*, 2023.

Saeed Mahloujifar, Luca Melis, and Kamalika Chaudhuri. Auditing $f$-differential privacy in one run. *arXiv preprint arXiv:2410.22235*, 2024.

Matthieu Meeus, Lukas Wutschitz, Santiago Zanella-Beguelin, Shruti Tople, and Reza Shokri. The canary's echo: Auditing privacy risks of llm-generated synthetic text. In *Forty-second International Conference on Machine Learning*, 2025.

Paul Micaelli and Amos J Storkey. Gradient-based hyperparameter optimization over long horizons. *Advances in Neural Information Processing Systems*, 34:10798–10809, 2021.

Meenatchi Sundaram Muthu Selva Annamalai. It's our loss: No privacy amplification for hidden state dp-sgd with non-convex loss. In *Proceedings of the 2024 Workshop on Artificial Intelligence and Security*, AISec '24, pp. 24–30, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400712289. doi: 10.1145/3689932.3694767. URL https://doi.org/10.1145/3689932.3694767.

Meenatchi Sundaram Muthu Selva Annamalai and Emiliano De Cristofaro. Nearly tight black-box auditing of differentially private machine learning. *Advances in Neural Information Processing Systems*, 37:131482–131502, 2024.

Milad Nasr, Shuang Songi, Abhradeep Thakurta, Nicolas Papernot, and Nicholas Carlin. Adversary instantiation: Lower bounds for differentially private machine learning. In *2021 IEEE Symposium on security and privacy (SP)*, pp. 866–882. IEEE, 2021.

Milad Nasr, Jamie Hayes, Thomas Steinke, Borja Balle, Florian Tramèr, Matthew Jagielski, Nicholas Carlini, and Andreas Terzis. Tight auditing of differentially private machine learning. In *32nd USENIX Security Symposium (USENIX Security 23)*, pp. 1631–1648, 2023.

Milad Nasr, Thomas Steinke, Borja Balle, Christopher A. Choquette-Choo, Arun Ganesh, Matthew Jagielski, Jamie Hayes, Abhradeep Guha Thakurta, Adam Smith, and Andreas Terzis. The last iterate advantage: Empirical auditing and principled heuristic analysis of differentially private SGD. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=DwqoBkj2Mw.

Ashwinee Panda, Xinyu Tang, Christopher A. Choquette-Choo, Milad Nasr, and Prateek Mittal. Privacy auditing of large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=60Vd7QOXlM.

Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. *Advances in neural information processing systems*, 32, 2019.

Thomas Steinke, Milad Nasr, and Matthew Jagielski. Privacy auditing with one (1) training run. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 49268–49280, 2023.

Florian Tramer, Andreas Terzis, Thomas Steinke, Shuang Song, Matthew Jagielski, and Nicholas Carlini. Debugging differential privacy: A case study for privacy auditing, 2022. URL https://arxiv.org/abs/2202.12219.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

# A APPENDIX

Table 2: Hyperparameters for training Wide ResNet 16-4 models using DP-SGD with $\varepsilon = 8.0$ and $\delta = 10^{-5}$. For training from scratch, we follow prior work (Nasr et al., 2021; Steinke et al., 2023). For DP finetuning, we instead train for 1000 steps, which we found achieves decent test accuracy ($\approx 90\%$).

| Hyperparameter | DP Training | DP Finetuning |
|---|---|---|
| # steps | 2500 | 1000 |
| Batch size | 4096 | 4096 |
| Clipping norm | 1.0 | 1.0 |
| Learning rate | 4.0 | 4.0 |

**Metagradient canary optimization.** Following Engstrom et al. (2025), we optimize the canary samples by training a ResNet-9 model (i.e., $w$ in Algorithm 5), allowing us to optimize $C$ efficiently. For step 1 of Algorithm 5, we initialize $C_0$ to $m$ samples randomly sampled from $D$ (i.e., CIFAR-10). We optimize for 500 metagradient steps. As demonstrated in Section 4.2, despite using a relatively compact model, our metagradient canaries are effective for much larger model architectures (i.e., Wide ResNets).

**Additional auditing procedure details.** As implemented in Mahloujifar et al. (2024), we align Algorithms 2 and 3 by fixing the canary set size to 2 so that half of $C$ is included in training for both auditing setups. When running Algorithm 2, we split $C$ randomly in half (instead of sampling with probability half) so that the set of $r$ non-auditing examples are the same for both auditing procedures. In addition, we use negative cross-entropy loss as the scoring function $s(\cdot)$ for both algorithms. In more detail,

- **[Steinke et al. (2023), Algorithm 2]** We sort the canaries $x \in C$ by $s(x)$ and take the top $k_+$ canaries in the sorted list as positive guesses and bottom $k_-$ as negative guesses.
- **[Mahloujifar et al. (2024), Algorithm 3]** We score canaries in each pair and predict the one with the higher score to have been included in training. We then score each pairing by taking the absolute difference scores $s(\cdot)$ between the canaries in each set and ranking the pairs by the difference. We take the top $k$ sets as our guesses.

For both procedures, we follow prior work (Steinke et al., 2023; Mahloujifar et al., 2024), varying the number of guesses from 10 up to $m$, in increments of 10, and reporting the max empirical $\varepsilon$.