DETECTABILITY WITHOUT INTERFERENCE: EFFICIENT CANARY DESIGN FOR PRIVACY AUDITING IN ONE RUN

Anonymous authors
Paper under double-blind review

ABSTRACT

Privacy auditing aims to empirically assess privacy leakage in machine learning models using membership inference attacks (MIAs), and to derive lower bounds on differential privacy (DP) parameters. Standard privacy auditing inserts a sensitive sample (a canary) into many ($\gg 100$) randomized training runs, requiring the auditor to detect whether it was included in each run. In one-run auditing, many canaries are included in a single training run, avoiding the high computational cost of multi-run approaches. However, recent theory shows that canaries can interfere with one another, reducing per-canary MIA scores and thereby weakening the audit power. We propose IBIS, a framework that optimizes canaries to maximize their own score while minimizing their impact on others. IBIS combines influence-based greedy initialization with bilevel optimization, introducing a novel regularization term that encourages canaries to be orthogonal in embedding space as a proxy for reducing interference. We propose an efficient bilevel algorithm that incrementally updates a single model at each canary update, avoiding the need to retrain the model from scratch after every update. Experiments show that IBIS offers a better performance-cost trade-off than existing canary crafting approaches, improving MIA metrics while reducing computational cost by up to two orders of magnitude.

1 INTRODUCTION

Machine learning models are known to leak information about their training data (Carlini et al., 2019; 2021; Haim et al., 2022; Nasr et al., 2023a; Barbero et al., 2025), motivating the need for privacy auditing. A popular approach to assess such leakage is through Membership Inference Attacks (MIAs) (Shokri et al., 2017; Carlini et al., 2021; Yeom et al., 2018; Zarifzadeh et al., 2024; Hayes et al., 2025): given a data point z and access to a trained model, an adversary attempts to infer whether z was a *member* of the training set, typically by constructing a confidence score derived from the model's outputs (e.g., based on the loss value at z). In the context of models trained with Differential Privacy (DP) (Dwork et al., 2006), one can further use MIA performance to obtain empirical lower bounds on the privacy parameters (Ding et al., 2018; Jagielski et al., 2020; Nasr et al., 2021; 2023b; Cebere et al., 2025b). In this sense, privacy auditing complements DP accounting, which derives worst-case upper bounds on the same parameters.

A standard component of privacy auditing consists of inserting "canary" data points into the training set, intentionally designed to maximize detectable leakage, and measuring how accurately an adversary can infer whether a given canary was included in or excluded from the training data. The usual *multi-run auditing* approach randomizes the inclusion of a single canary at a time and repeats independent model training thousands of times to obtain statistically reliable estimates. This makes auditing prohibitively expensive for large-scale scenarios. *One-run auditing* (Steinke et al., 2023; Mahloujifar et al., 2025; Even et al., 2026) instead considers a single training run in which m canaries are independently included with probability $1/2$. Given the resulting model, an adversary then attempts to infer which canaries were included. This approach avoids the need to train a separate model for each randomized canary inclusion, substantially reducing the computational cost of privacy auditing.

In the *black-box* setting, where the adversary inserts sample canaries and observes only the final model's outputs, accurately detecting privacy leakage remains challenging (Nasr et al., 2021; 2023c). This challenge is particularly pronounced in the one-run paradigm, which generally yields weaker empirical estimates than multi-run approaches (Keinan et al., 2025; Xiang et al., 2025). Several works

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(Keinan et al., 2025; Xiang et al., 2025; Even et al., 2026) attribute this degradation to *interference between canaries*: the inclusion or exclusion of a canary z' can affect the MIA confidence score of another canary z, thereby degrading the adversary's ability to infer the membership of z. This interference is particularly strong when canaries are similar, as their effects on the model are correlated. In such cases, including z' can reduce the model's sensitivity to z and make its membership harder to detect. Bridging the gap with multi-run auditing therefore requires canaries that are both *detectable* in isolation and *non-interfering* with one another: the MIA score of each canary should change significantly depending on its inclusion, but remain largely unaffected by the inclusion of other canaries. Existing canary crafting methods designed for one-run auditing (Boglioni et al., 2026) do not explicitly promote *non-interference*, limiting audit power; moreover, they are so expensive that they can effectively negate the computational advantage of the one-run paradigm.

Contributions. We propose IBIS, a two-stage method for crafting canary sets for one-run auditing. IBIS first uses influence functions (Koh & Liang, 2017) to greedily select training points that are highly detectable, as measured by their self-influence, while minimizing pairwise interference, as measured by their cross-influence. This provides a strong initialization for subsequent optimization. IBIS then refines the canaries through a bilevel optimization objective that jointly promotes detectability and orthogonality in representation space, using orthogonality as a tractable proxy for non-interference.

A key feature of our formulation is that it avoids evaluating canaries across the exponentially many possible subsets of included canaries. Instead, by promoting non-interference, we can compare the model trained with all canaries against a fixed reference model, requiring only a single model to be maintained throughout the optimization. We exploit this structure to develop an efficient bilevel algorithm that incrementally updates the model after each canary update, rather than retraining it from scratch at every outer step. This substantially reduces the cost of canary crafting while preserving strong attack performance.

We evaluate IBIS on WideResNet models trained on CIFAR-10 and CIFAR-100, and on a multilayer perceptron trained on MNIST. Our results show that IBIS effectively reduces interference between canaries, leading to higher privacy leakage estimates than prior canary crafting methods for one-run auditing. At the same time, IBIS is substantially more computationally efficient, reducing the cost of canary crafting by up to two orders of magnitude. This efficiency allows the improved audit performance to retain the computational advantage of one-run auditing over multi-run approaches.

Related work on canary crafting. Heuristic canary crafting methods are routinely used, such as flipping the label of a sample or using a blank image (Nasr et al., 2023c). The seminal work of Jagielski et al. (2020) introduced the ClipBKD technique, which optimizes the canary for multi-run auditing by aligning it with the lowest-variance direction of the training data covariance. Several follow-up works also operate primarily in the multi-run setting. Lu et al. (2022) use influence functions to design canaries that maximize their effect on model parameters, but instantiate them only for logistic regression. More recently, Yaghini et al. (2026) leverage influence functions and bilevel optimization, but optimize a single canary rather than a set. Closest to us, Boglioni et al. (2026) target the one-run regime, but do not promote non-interference and require retraining the model at every canary update. Yaghini et al. (2026) and Boglioni et al. (2026) both rely on unrolling the training loop, as discussed in Section 4.2. In the federated setting, Maddock et al. (2023) craft a canary by penalizing the inner products between its gradient and the other clients' updates, while keeping its gradient norm large. A related line of work considers a different threat model where the auditor injects gradients rather than data points, enabling tight privacy audits in certain settings (Cebere et al., 2025a).

2 Background

2.1 Membership inference and differential privacy

Membership Inference Attacks (MIAs) (Shokri et al., 2017; Carlini et al., 2022) are privacy attacks in which an adversary attempts to infer whether a given data point z was part of the training set of a model $\mathcal{A}(D)$ trained on dataset D. The adversary computes a score $s(\mathcal{A}(D), z)$, for instance based on predicted probabilities or loss values, and thresholds it to produce a binary guess. MIAs are classically formulated as a hypothesis test where the null hypothesis is that $z \notin D$, yielding standard

performance metrics such as the Area Under the Curve (AUC) or the True Positive Rate (TPR) at low False Positive Rate (FPR) (Carlini et al., 2022): the higher these metrics, the stronger the MIA.

Differential Privacy (DP) is the standard framework to quantify the privacy guarantees of randomized algorithms (Dwork & Roth, 2014; Dwork et al., 2006). It ensures that the algorithm's output distribution does not change significantly when a single input data point is added or removed.

Definition 2.1. A randomized algorithm $\mathcal{A} : \mathcal{D} \to \mathcal{O}$ satisfies (ϵ, δ)-DP if, for any datasets $D, D' \in \mathcal{D}$ s.t. $D' = D \cup \{z\}$ or $D = D' \cup \{z\}$, and for any set $O \subseteq \mathcal{O}$, $\mathbb{P}[\mathcal{A}(D) \in O] \leq e^{\epsilon}\mathbb{P}[\mathcal{A}(D') \in O] + \delta$.

The parameter ϵ controls the strength of the guarantee, with smaller values providing stronger privacy, while δ should be a small constant that allows a negligible probability of failure. At scale, DP training is usually performed with DP-SGD (Song et al., 2013; Bassily et al., 2014; Abadi et al., 2016), which clips individual gradients and adds Gaussian noise to the update direction.

Importantly, DP also limits the success of any MIA (Nasr et al., 2021). MIA-based privacy auditing can thus provide empirical lower bounds on DP guarantees, in addition to measuring privacy leakage for models without formal privacy guarantees. We next recall the basics of one-run privacy auditing.

2.2 Privacy auditing in one run

In one-run auditing (Steinke et al., 2023; Even et al., 2026), the auditor has a set $C = \{z_1, \ldots, z_m\}$ of canaries. A selection $S = \{z_i : A_i = 1\}$ is built by drawing $A_i \sim \text{Bernoulli}(1/2)$ independently for each canary. The model is then trained on $D \cup S$, yielding model parameters θ. A MIA score $s(\theta, z)$ is computed for each canary $z \in C$. This function should be such that $s(\theta, z)$ is high when $z \in S$ and low otherwise. In our experiments, we use the loss-based score $s(\theta, z) = -\ell(\theta, z)$ where ℓ is the training loss (e.g. the cross-entropy loss), which is a simple and commonly used choice. We do not use stronger attacks such as LiRA (Carlini et al., 2022) or RMIA (Zarifzadeh et al., 2024) since they rely on shadow models, which is not relevant in the one-run auditing context. The procedure is summarized in Algorithm 1.

Algorithm 1 Auditing in one run (Steinke et al., 2023; Even et al., 2026)

Input: Dataset D, audited algorithm $\mathcal{A}$, score function s, canaries $C = \{z_1, \ldots, z_m\}$
Draw $A \sim \text{Bernoulli}(1/2)^{\otimes m}$.
Compute $\theta = \mathcal{A}(D \cup \{z_i : A_i = 1\})$
for $i = 1, \ldots, m$ **do**
  Compute the score $\hat{s}_i = s(\theta, z_i)$
end for
$\text{MIA_perf} = \text{TPR}_{@\alpha\text{-FPR}}(\hat{s}, A)$
$\text{DP_lower_bound} = \hat{\epsilon}(\hat{s}, A)$
Output: MIA_perf, DP_lower_bound

These scores alongside the ground-truth memberships can then be used for privacy auditing. One can compute standard attack performance metrics, such as TPR at low FPR (Carlini et al., 2022):

$$\text{TPR}_{@\alpha}(\hat{s}, A) = \max_{\tau} \left\{ \frac{\sum_i \mathbf{1}[\hat{s}_i \geq \tau]\, A_i}{\sum_i A_i} \;:\; \frac{\sum_i \mathbf{1}[\hat{s}_i \geq \tau]\,(1 - A_i)}{\sum_i (1 - A_i)} \leq \alpha \right\}.$$

For DP auditing, following Steinke et al. (2023), we can construct a lower bound of the privacy parameter ϵ. To do so, the k_+ highest scores are guessed to have been included in the training set, while the k_- lowest scores are guessed to have been excluded. From these statistics, the auditor can either directly estimate a lower bound on ϵ (Steinke et al., 2023), or estimate the Gaussian Differential Privacy (GDP) parameter (Dong et al., 2019) for a tighter bound (Nasr et al., 2023c; Mahloujifar et al., 2025).

Audit quality depends heavily on the choice of canaries. Indeed, the inclusion of a canary should strongly affect its own score while minimally affecting others. Influence functions provide a proxy for these properties; we will use them to select a good initial set of canaries from the dataset.

2.3 Influence functions

Influence functions aim at measuring the effect of upweighting a data sample on a model. They were first introduced in robust statistics (Hampel, 1974) and then applied in machine learning for various purposes such as explainability (Koh & Liang, 2017), or to study the memorization phenomenon in neural networks (Feldman & Zhang, 2020; Zhang et al., 2023; Meeus et al., 2025) thanks to the notion of cross and self-influence. Recently, the computational cost of influence functions has been reduced so that they can be used at scale (Bae et al., 2022; Grosse et al., 2023). Consider a dataset $D = \{z_1, \ldots, z_n\}$, a loss function ℓ and the solution of the Empirical Risk

Minimization (ERM) problem $\theta^*(D) = \arg\min_\theta \mathcal{L}(\theta, D)$, where $\mathcal{L}(\theta, D) \triangleq \frac{1}{n}\sum_{i=1}^n \ell(\theta, z_i)$. For $\alpha > 0$ and a data point z, we denote $\theta_z^*(\alpha; D)$ the model obtained by solving $\min_\theta \frac{1}{n}\sum_{i=1}^n \ell(\theta, z_i) + \alpha\ell(\theta, z)$. Since $\theta^*(D) = \theta_z^*(0; D)$, the difference $\theta_z^*(\alpha; D) - \theta^*(D)$ can be approximated by the Jacobian $\alpha\partial_\alpha\theta_z^*(0; D)$. Under strong convexity of the loss function ℓ, the implicit function theorem yields the *influence function* of z: $\partial_\alpha\theta_z^*(0; D) = -\left[\nabla^2\mathcal{L}(\theta^*(D), D)\right]^{-1}\nabla\ell(\theta^*(D), z)$. Then, for a measurement function f, substituting the influence function into a first-order Taylor expansion yields $f(\theta_z^*(\alpha; D)) \approx f(\theta^*(D)) + \alpha\nabla f(\theta^*(D))^\top\partial_\alpha\theta_z^*(0; D)$.

3 Canary selection by influence optimization

The work of Yaghini et al. (2026) highlights the value of influence-based initialization for gradient-based canary crafting in the multi-run setting. In the one-run setting, however, multiple canaries co-exist in the same training run, making their mutual interference a key consideration. We therefore use both self- and cross-influence to greedily select a *set* of dataset points that are individually detectable while having low mutual interference, and use this set to initialize our subsequent bilevel optimization.

3.1 Cross- and self-influence on MIA scores

Consider any MIA score function s such that $s(\theta^*(D), z)$ is higher when $z \in D$. To quantify detectability and interference, we track how the score of z changes when a point z' is removed from D. Applying the first-order approximation from Section 2.3 with $\alpha = -1/|D|$ gives $s(\theta^*(D), z) - s(\theta^*(D \setminus \{z'\}), z) \approx \frac{1}{|D|}\mathcal{I}_s(z, z')$, with $\mathcal{I}_s(z, z')$ defined in Definition 3.1.

Definition 3.1 (Self- and cross-influence)**.** The *influence* of z' on the score of z is $\mathcal{I}_s(z, z') \triangleq \nabla_\theta s(\theta^*(D), z)^\top\partial_\alpha\theta_{z'}^*(0; D)$. When $z' = z$, $\mathcal{I}_s(z, z)$ is the *self-influence* of z; when $z' \neq z$, $\mathcal{I}_s(z, z')$ is the *cross-influence* of z' on z.

Remark. Under strongly-convex ERM and the loss-based score $s(\theta, z) = -\ell(\theta, z)$, both reduce to $\mathcal{I}_s(z, z') = \nabla\ell(\theta^*(D), z)^\top[\nabla^2\mathcal{L}(\theta^*(D), D)]^{-1}\nabla\ell(\theta^*(D), z')$. Self-influence $\mathcal{I}_s(z, z) \geq 0$ recovers the intuition that removing z increases its loss. A positive cross-influence $\mathcal{I}_s(z, z') > 0$ means that removing z' also increases the loss of z, which creates interference.

In the one-run auditing setting, a large $\mathcal{I}_s(z_i, z_j)$ means that the inclusion or exclusion of z_j changes the score of z_i in line 5 of Alg. 1, making its membership harder to infer. This motivates selecting canaries with low pairwise cross-influence.

3.2 Greedy search of canary set

In this section, we introduce the first step of our canary-crafting approach: selecting initial canary candidates from the training set D. These candidates initialize the non-convex gradient-based optimization procedure of Section 4.

Our selection strategy leverages influence functions to identify a set $C = \{z_1, \ldots, z_m\} \subset D$ whose elements exhibit high detectability while minimizing mutual interference. Specifically, we require that the inclusion of each canary $z_i \in C$ can be reliably inferred from its MIA score $s(\theta, z_i)$, and that this score remains largely insensitive to the presence or absence of any other canary $z_j \in C$, with $z_j \neq z_i$.

Algorithm 2 Greedy canary selection

1: **Input:** Dataset D, number of canaries m, number of preselected canaries p
2: Let $\overline{D} \subset D$ be the p samples with highest self-influence.
3: Set $C_1 = \{\arg\max_{z \in \overline{D}} \mathcal{I}_s(z, z)\}$
4: **for** $k = 1, \ldots, m-1$ **do**
5: Set $z_{k+1} = \arg\max_{z \in \overline{D} \setminus C_k} \frac{\mathcal{I}_s(z,z)}{\max_{z' \in C_k} \mathcal{I}_s(z,z')}$
6: Update $C_{k+1} = C_k \cup \{z_{k+1}\}$
7: **end for**
8: **Output:** C_m

We use self-influence as a proxy for detectability and cross-influence as a proxy for mutual interference. Consequently, we seek a set C in which each canary has high self-influence and low cross-influence with the other selected canaries. We only control cross-influence within C, since the remaining points of D are included in every training run. Their effect on a canary's score is therefore constant across runs: they may shift the score, but do not cause it to fluctuate. Formally, for canary $z_i \in C$, $\mathcal{I}_s(z_i, z_i)$ measures

the change in its score induced by its own inclusion, whereas $\mathcal{I}_s(z_i, z_j)$ measures the change induced by another canary $z_j \in C \setminus \{z_i\}$. The ratio $\frac{\mathcal{I}_s(z_i,z_i)}{\max_{z_j \in C\setminus\{z_i\}} \mathcal{I}_s(z_i,z_j)}$ therefore captures the canary's own signal relative to its strongest potential source of interference. This leads to the following combinatorial optimization problem:

$$\max_{C\subset D, |C|=m} \Big\{ f(C) = \sum_{z_i \in C} \frac{\mathcal{I}_s(z_i, z_i)}{\max_{z_j \in C\setminus\{z_i\}} \mathcal{I}_s(z_i, z_j)} \Big\}. \tag{1}$$

Solving (1) exactly requires evaluating $\binom{|D|}{m}$ candidate subsets, which is computationally infeasible. Moreover, since f is not submodular, no efficient algorithm with approximation guarantees is known. We therefore adopt a greedy heuristic (Papadimitriou & Steiglitz, 1982), summarized in Algorithm 2.

The procedure begins with a preselection step: we construct a subset $\overline{D} \subset D$ containing the p samples with highest self-influence. This reduces the number of required cross-influence evaluations. We then build C iteratively. Given a partial set C_k of size k, the next canary is chosen by maximizing, over $z \in \overline{D} \setminus C_k$, the ratio $\frac{\mathcal{I}_s(z,z)}{\max_{z' \in C_k} \mathcal{I}_s(z,z')}$. This requires computing and storing all pairwise self- and cross-influences within $\overline{D}$, resulting in a $p \times p$ score matrix. The computation of the influence functions, which involves inverse Hessian-vector products, can be made more efficient in large scale setting by leveraging K-FAC approximation of the Hessian (Martens & Grosse, 2015; Bae et al., 2022).

As we will see experimentally, this selection provides a good initialization for the gradient-based refinement step introduced next, which searches for canaries beyond the training set.

4 Gradient-based refinement of canaries

In this section, we design a bilevel objective function to craft canaries that are highly detectable and exhibit low mutual interference. This objective is optimized through a gradient-based method initialized with the canary set we obtain after the greedy search (Algorithm 2).

4.1 Warm-up: canary interference in least squares regression

To build intuition, we first consider the simplest setting: least squares regression with two canaries. It has the advantage of admitting a closed-form solution. Let $D = \{z_i = (x_i, y_i)\}_{1\le i\le n}$ denote the training set, where $x_i \in \mathbb{R}^d$ represents input features and $y_i \in \mathbb{R}$ the corresponding label. Denoting $X = (x_1, \dots, x_n)^\top \in \mathbb{R}^{n\times d}$ the design matrix and $y = (y_1, \dots, y_n)^\top \in \mathbb{R}^n$ the target vector, we seek $\theta^*(X, y) \in \arg\min_{\theta\in\mathbb{R}^d} \frac{1}{2}\|X\theta - y\|^2$. When $n \ge d$ and X has full rank, the solution is unique and given by $\theta^*(X, y) = (X^\top X)^{-1} X^\top y$.

Consider now two canaries $z_{c,1} = (x_{c,1}, y_{c,1})$ and $z_{c,2} = (x_{c,2}, y_{c,2})$. We define $\tilde{X}_i = \begin{bmatrix} X \\ x_{c,i} \end{bmatrix}$, and $\tilde{y}_i = \begin{bmatrix} y \\ y_{c,i} \end{bmatrix}$. Let $\ell(\theta, z_i) = \frac{1}{2}(y_i - \theta^\top x_i)^2$ denote the least square loss. We are interested in quantifying how the inclusion of $z_{c,1}$ in the training set affects the loss evaluated at $z_{c,2}$, i.e., how $\ell(\theta^*(\tilde{X}_1, \tilde{y}_1), z_{c,2})$ differs from $\ell(\theta^*(X, y), z_{c,2})$. The following proposition characterizes this effect.

Proposition 4.1. *Assume that $n \ge d$ and that the matrix X has rank d. Then we have*

$$\ell(\theta^*(\tilde{X}_1, \tilde{y}_1), z_{c,2}) - \ell(\theta^*(X, y), z_{c,2}) = \gamma\, x_{c,1}^\top K^{-1} x_{c,2}, \tag{2}$$

where $K = X^\top X$ and γ is a scalar that depends on the two canaries and on $\theta^*(X, y)$. Its closed form is given in Appendix A.1.

We provide the proof in Appendix A.1. Equation (2) vanishes as soon as $x_{c,1}^\top K^{-1} x_{c,2} = 0$, that is, as soon as the two canaries are K^{-1}-*orthogonal*. This observation motivates incorporating an orthogonality constraint into our bilevel objective. While this derivation is specific to least squares, it transfers to deep models through their last layer, which fits a linear map on the learned representations. Applying Proposition 4.1 at that layer replaces the inputs by the embeddings $\varphi_{\theta^*}(x)$ computed by the preceding layers, and K by their Gram matrix. We therefore promote orthogonality in the representation space induced by the model, rather than in raw input space.

4.2 CANARY OPTIMIZATION AS A BILEVEL OPTIMIZATION PROBLEM

Unlike Section 3, where canaries are selected from a fixed dataset D, we now treat the canary samples themselves as optimization variables and formulate a bilevel objective amenable to gradient-based optimization. Throughout this section, we slightly abuse notation and write $\theta^* \triangleq \theta^*(D)$ and $\theta^*_S \triangleq \theta^*(D \cup S)$ for any set S, where $\theta^*(D') \in \arg\min_\theta \mathcal{L}(\theta, D')$.

The canaries obtained via Algorithm 2 already exhibit low pairwise cross-influence by construction, implying that for any canary $z \in C$ and subset $S \subset C$, the score $s(\theta^*_S, z)$ is approximately determined only by whether $z \in S$. Under this approximation, it suffices to consider the full inclusion case $S = C$, yielding a simplified distinguishability objective of the form $\sum_{z \in C} s(\theta^*, z) - s(\theta^*_C, z)$.

This objective is meaningful only when interference between canaries remains sufficiently low. Algorithm 2 enforces this property at initialization, but subsequent gradient updates may reintroduce interference as the inner model and canaries evolve. We therefore introduce a regularizer $\mathcal{R}_{\theta^*}(C)$ that preserves this structure throughout the optimization, leading to the following bilevel problem:

$$\min_C \ \Phi(C) \triangleq \sum_{z \in C} s(\theta^*, z) - s(\theta^*_C, z) + \mathcal{R}_{\theta^*}(C) \text{ s.t. } \begin{cases} \theta^* \in \arg\min_{\theta \in \Theta} \mathcal{L}(\theta, D), \\ \theta^*_C \in \arg\min_{\theta \in \Theta} \mathcal{L}(\theta, D \cup C). \end{cases} \tag{3}$$

Building on the intuition from Section 4.1, we define the regularizer in terms of representation-space orthogonality. Denoting by C_x the inputs of the canaries in C, we set $\mathcal{R}_{\theta^*}(C) = \lambda \sum_{x, x' \in C_x, x \neq x'} \langle \varphi_{\theta^*}(x), \varphi_{\theta^*}(x') \rangle^2$, where $\varphi_{\theta^*} : \mathbb{R}^d \to \mathbb{R}^{d'}$ maps each input to its embedding under θ^*, and $\lambda > 0$ trades off between reducing mutual interference and preserving canary detectability. Using θ^* rather than θ^*_C decouples the regularizer from the inner optimization problem, which avoids additional complexity. In the bilevel problem (3), only θ^*_C depends on the canary set. The model θ^* can therefore be computed once in advance, while θ^*_C is updated incrementally as C evolves rather than recomputed from scratch, enabling amortized bilevel schemes (Ji et al., 2021; Arbel & Mairal, 2022).

4.3 EFFICIENT GRADIENT-BASED CANARY CRAFTING

We now introduce an efficient algorithm to solve the bilevel optimization problem defined in the previous section. The optimization is initialized using the canary set obtained from Section 3, and then refined via gradient-based updates. To avoid discrete optimization over labels, we keep them fixed throughout training and optimize only the canary inputs C_x.

To perform gradient-based optimization of Φ, we require its gradient with respect to C_x, denoted by $\nabla_{C_x} \Phi$, which is referred to as the hypergradient in the bilevel optimization literature. Proposition 4.2 provides an explicit expression for $\nabla_{C_x} \Phi$ for the case where the inner problem is strongly convex. For a canary $z = (x, y)$, the notation ∇_x indicates that differentiation is performed with respect to the input alone, the label being held fixed.

Proposition 4.2. *Assume that the score function s is differentiable w.r.t. both θ and the input sample x, and that the loss function ℓ is twice differentiable, strongly convex in θ and differentiable in x. Then Φ is differentiable and, letting $n \triangleq |D| + |C|$, its gradient is given by*

$$\nabla_{C_x} \Phi(C) = \sum_{z \in C} \left[\nabla_x s(\theta^*, z) - \nabla_x s(\theta^*_C, z) - \tfrac{1}{n} \nabla^2_{x,\theta} \ell(\theta^*_C, z) v^*(C) \right] + \nabla_{C_x} \mathcal{R}_{\theta^*}(C)$$

where $v^*(C)$ is the solution of the linear system $\nabla^2_{\theta,\theta} \mathcal{L}(\theta^*_C, D \cup C) v = -\sum_{z \in C} \nabla_\theta s(\theta^*_C, z)$.

Proposition 4.2 follows by applying the standard implicit differentiation result for bilevel optimization to our objective; we provide the proof in Appendix A.2. While Proposition 4.2 provides an exact expression, directly using it to perform gradient descent is intractable in large-scale scenarios. Indeed, it requires repeatedly recomputing θ^*_C as the canary set evolves, effectively retraining the model at each update, as well as solving a high-dimensional linear system to obtain $v^*(C)$. To address this, we rely on Approximate Implicit Differentiation (AID) (Pedregosa, 2016; Ghadimi & Wang, 2018), which replaces θ^*_C and $v^*(C)$ with estimates updated during optimization. Although AID is theoretically grounded in settings with strongly convex inner problems, it has also been successfully applied in non-convex regimes (Grangier et al., 2024).

Concretely, we instantiate this approach with the SOBA algorithm, which converges to a stationary point of Φ when the inner problem is strongly convex (Dagréou et al., 2022, Theorems 1 and 2).

Algorithm 3 IBIS: Influence-Bilevel canary crafting with Interference Suppression

Input: Dataset D, score s, m, p, λ, η, ρ, T
Get C_0 via Algorithm 2; set $D \leftarrow D \setminus C_0$.
Compute $\theta^* \approx \arg\min_\theta \mathcal{L}(\theta, D)$; initialize $\theta_0 \approx \theta^*(D \cup C_0)$, $v_0 \approx v^*(C_0)$.
for $t = 0, \dots, T-1$ **do**
  Sample $B_t \subset D \cup C_t$;
  Update θ_{t+1} via (4).
  Update v_{t+1} via (5).
  Update $C_{x,t+1}$ via (6).
end for
Output: C_T.

The algorithm maintains three coupled variables θ, v and C_x. At iteration t, the iterates θ_t and v_t approximate $\theta^*(D \cup C_t)$ and $v^*(C_t)$ respectively. They are updated using stochastic gradient steps to track their respective equilibria as the canary set evolves, while C_x is updated via an approximate hypergradient step. The resulting recursion is:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta_t, B_t) \tag{4}$$

$$v_{t+1} = v_t - \eta \Big[\nabla^2_{\theta,\theta} \mathcal{L}(\theta_t, B_t) v_t + \textstyle\sum_{z \in C_t} \nabla_\theta s(\theta_t, z) \Big] \tag{5}$$

$$C_{x,t+1} = C_{x,t} - \rho \Big[\textstyle\sum_{z \in C_t} \big(\nabla_x s(\theta^*, z) - \nabla_x s(\theta_t, z) - \frac{1}{n} \nabla^2_{x,\theta} \ell(\theta_t, z) v_t \big) + \nabla_{C_x} \mathcal{R}_{\theta^*}(C_t) \Big] \tag{6}$$

where $B_t \subset D \cup C_t$ is a mini-batch of samples. Note that (6) coincides with the exact hypergradient update when $\theta_t = \theta^*_{C_t}$ and $v_t = v^*(C_t)$. Moreover, although the recursion involves Hessian-vector products, these can be computed efficiently without explicitly forming the Hessian using the Pearlmutter trick (Pearlmutter, 1994; Dagréou et al., 2024). Thanks to the structure of our bilevel formulation, which compares the model trained with all canaries against a fixed reference model, (4)–(6) avoid full retraining at each canary update. Instead, the model parameters θ evolve jointly with the canary optimization. In practice, we initialize θ^0 and v^0 by approximately computing $\theta^*_{C_0}$ and $v^*(C_0)$, ensuring that for small enough step sizes, we keep $\theta_t \approx \theta^*_{C_t}$ and $v_t \approx v^*(C_t)$ throughout optimization. The recursion also supports per-example gradient clipping and data augmentation (see Appendix B.1), allowing our method to account for these components when auditing models that use them.

4.4 IBIS — Influence-Bilevel canary crafting with Interference Suppression

Our full procedure, which we call **IBIS**, is summarized in Algorithm 3. It combines the two contributions of this paper into a unified pipeline: the influence-based greedy selection (Algorithm 2) provides an initialization with strong detectability and low pairwise interference, and the bilevel updates (4)–(6) further refine the canaries while preserving their diversity through the regularizer $\mathcal{R}_{\theta^*}$. Overall, the procedure requires three model trainings: one for influence-based selection and two to initialize θ^* and $\theta_0 \approx \theta^*(D \cup C_0)$ before the bilevel loop. This modest overhead relative to the cost of training the audited model preserves the computational advantage of one-run over multi-run auditing.

Difference with Boglioni et al. (2026). The canary crafting algorithm of Boglioni et al. (2026) can also be formulated as a bilevel optimization problem. It learns canary sets through gradient steps obtained by differentiating through SGD updates, which approximately solves the following problem by iterative differentiation (Domke, 2012; Maclaurin et al., 2015; Bolte et al., 2023):

$$\min_C \Psi(C) \triangleq \mathbb{E}_S \Big[\sum_{z \in C \setminus S} s(\theta^*_S, z) - \sum_{z \in S} s(\theta^*_S, z) \Big] \text{ s.t. } \theta^*_S \in \arg\min_{\theta \in \Theta} \mathcal{L}(\theta, D \cup S),\ \forall S \subset C, \tag{7}$$

where the expectation is over the random subset S, formed by including each canary independently with probability $1/2$. Minimizing Ψ encourages included canaries to have high scores and excluded canaries to have low scores, averaged over subsets S. Unlike (3), this formulation requires a distinct inner solution θ^*_S for every subset $S \subset C$. Consequently, whereas our approach performs a single gradient update as in (4), their approach must retrain the model for each sampled S. This computation cannot be amortized without maintaining a separate model for every possible subset. We compare the wall-clock time of the two approaches in Section 5.3, where our approach is two orders of magnitude faster.

5 EXPERIMENTS

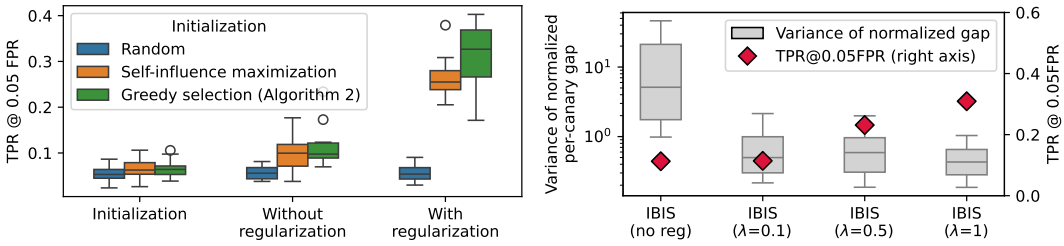
We audit WRN16-4 models (Zagoruyko & Komodakis, 2016) trained on CIFAR10 and CIFAR100, and a multilayer perceptron (MLP) trained on MNIST. Following Yaghini et al. (2026, Appendix E.8), we train the WRN16-4 models with per-epoch data augmentation and an exponential moving average of the weights to avoid blatant memorization, as shown in Appendix B.4. Further experimental details are given in Appendix C.

5.1 ROLE OF INITIALIZATION AND REGULARIZATION IN IBIS

We study the role of initialization and regularization in IBIS from two complementary perspectives: first, through an ablation study, and second, by directly evaluating whether our orthogonality-based regularization reduces interference and thereby improves audit performance.

Effectiveness of initialization and regularization. We conduct an ablation study on WRN16-4 trained on CIFAR10. We initialize the bilevel algorithm with $m = 1000$ canaries selected either randomly, by maximizing self-influence, or using our proposed greedy selection strategy (Algorithm 2). For each initialization strategy, we run IBIS with and without regularization ($\lambda = 1$ and $\lambda = 0$, respectively) and report the resulting TPR@0.05FPR in Figure 1a. With influence-based initialization, the unregularized bilevel optimization yields only a moderate improvement, whereas regularization leads to a substantial performance gain. The magnitude of this gain depends on the initialization strategy: with regularization, our proposed greedy selection achieves higher scores than self-influence maximization. These results highlight the importance of influence-based initialization and the value of accounting for cross-influence when selecting canaries for one-run privacy auditing.

Interference and audit performance. Our regularization uses orthogonality in embedding space as a proxy for reducing interference. We thus vary the regularization strength λ and measure its effect on both interference and downstream audit performance. For a canary $z \in C$ and a subset $S \subset C$, we define the score gap $\Delta(z, S) \triangleq s(\mathcal{A}(D \cup S \cup \{z\}), z) - s(\mathcal{A}(D \cup S \setminus \{z\}), z)$. In the absence of interference, this gap is independent of S. We therefore quantify interference by $\mathrm{Var}_S[\Delta(z, S)]$, normalized by the squared mean gap across canaries. As shown in Figure 1b regularization reduces this measure by about an order of magnitude, while increasing TPR@0.05FPR from 0.11 to 0.31.



(a) Impact of initialization and regularization on the audit performance (TPR@0.05FPR).

(b) Estimated interference and audit performance (TPR@0.05FPR) for several regularization strengths λ.

Figure 1: Role of initialization and regularization in IBIS (WRN16-4 on CIFAR10).

5.2 COMPARISON WITH STANDARD CANARY CONSTRUCTIONS

We compare IBIS with two standard canary constructions: Random and Mislabeled. The comparison with the method of Boglioni et al. (2026) is deferred to Section 5.3. Random canaries are images drawn uniformly at random from the dataset. Mislabeled canaries are likewise drawn at random, but their labels are flipped.

Table 1 reports audit performance in terms of TPR@0.05FPR. Since an auditor typically performs a single audit, we also report the minimum value across runs, which captures the leakage that can be reliably detected. Random canaries remain close to the 0.05 level expected under random guessing, indicating that typical training samples are difficult to detect in a single run. On CIFAR10, IBIS

Table 1: Audit performance (TPR@0.05FPR) for a WRN16-4 model on CIFAR10 and CIFAR100. We report mean, standard deviation and minimum over 15 runs.

	CIFAR10		CIFAR100	
	Mean $\pm$ std	Min	Mean $\pm$ std	Min
IBIS	$\mathbf{0.31 \pm 0.07}$	**0.17**	$\mathbf{0.35 \pm 0.05}$	**0.27**
Random	0.05 ± 0.01	0.02	0.06 ± 0.01	0.04
Mislabeled	0.22 ± 0.06	0.12	0.33 ± 0.04	0.24

Table 2: Audit performance (AUC, TPR@0.05FPR) and canary crafting runtime of IBIS and METAGRADIENT on CIFAR10 and MNIST. We report mean and standard deviation over 15 runs. Times marked with * are those reported by Boglioni et al. (2026, Appendix B).

	IBIS			METAGRADIENT (Boglioni et al., 2026)		
	AUC	TPR@0.05FPR	Time [h]	AUC	TPR@0.05FPR	Time [h]
CIFAR10	$\mathbf{0.82 \pm 0.02}$	$\mathbf{0.31 \pm 0.07}$	**1.2**	0.76 ± 0.03	0.15 ± 0.05	96–120*
MNIST	$\mathbf{0.86 \pm 0.02}$	$\mathbf{0.29 \pm 0.07}$	**0.1**	0.75 ± 0.01	0.06 ± 0.01	30

canaries achieve higher mean and minimum TPR than Mislabeled canaries. On CIFAR100, IBIS achieves slightly higher scores on both metrics.

5.3 COMPARISON WITH METAGRADIENT (BOGLIONI ET AL., 2026)

We compare IBIS with its closest competitor, the method of Boglioni et al. (2026), in terms of both audit performance and computational cost. We refer to this method as METAGRADIENT. For IBIS, crafting time includes both the greedy initialization and the bilevel optimization stage. On CIFAR10, we use the METAGRADIENT canaries released with the authors' code, as reproducing them would be prohibitively expensive. On MNIST, no pre-crafted canaries are available, so we craft them using the authors' implementation. We do not evaluate METAGRADIENT on CIFAR100, as crafting canaries with this method would require prohibitively large computational resources. Table 2 reports both TPR@0.05FPR and the AUC, as METAGRADIENT's TPR@0.05FPR is close to random guessing on MNIST. IBIS achieves higher audit performance in both metrics, while requiring up to two orders of magnitude less computation than METAGRADIENT.

Additional results. In Appendix B.2 we provide an experiment of DP auditing. In this setting, IBIS and METAGRADIENT canaries achieve comparable results. In Appendix B.3, we compare the whole pipeline of multi-run and one-run auditing, showing that crafting canaries with IBIS does not cancel the computational advantage of one-run auditing.

6 CONCLUSION

In this work, we studied canary crafting for one-run privacy auditing. Building on the observation that interference between canaries can limit the power of such audits, we propose IBIS, which uses influence functions to select detectable, weakly interfering canaries and a bilevel objective to further reduce interference by promoting orthogonality in embedding space. This low-interference regime allows IBIS to update a single model trained with all canaries incrementally, avoiding retraining at every canary update. Experiments show that IBIS improves audit performance over existing canary crafting methods while reducing crafting cost by up to two orders of magnitude. While one-run auditing substantially reduces computational cost, its audit performance remains lower than that of multi-run auditing, reflecting the challenges of black-box auditing from a single model. Whether improved canary crafting can close this gap in some settings remains an open question.

AI USE STATEMENT

In this work, large language models were used to polish the writing and format the tables of the paper. The LLM-generated content was reviewed by the authors, who take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Proofs of the theoretical results are given in Appendices A.1 and A.2. The experimental details, including all hyperparameters, are given in Appendix C. The anonymized source code is provided as supplementary material.

REFERENCES

Martín 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. page 3

Michael Arbel and Julien Mairal. Amortized Implicit Differentiation for Stochastic Bilevel Optimization. In *International Conference on Learning Representations (ICLR)*, 2022. page 6

Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger Grosse. If Influence Functions are the Answer, Then What is the Question? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. pages 3, 5

Federico Barbero, Xiangming Gu, Christopher A Choquette-Choo, Chawin Sitawarin, Matthew Jagielski, Itay Yona, Petar Veličković, Ilia Shumailov, and Jamie Hayes. Extracting alignment data in open models. *arXiv preprint arXiv:2510.18554*, 2025. page 1

Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private Empirical Risk Minimization: Efficient Algorithms and Tight Error Bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pp. 464–473, Philadelphia, PA, USA, 2014. IEEE. page 3

Matteo Boglioni, Terrance Liu, Andrew Ilyas, and Zhiwei Steven Wu. Optimizing Canaries for Privacy Auditing with Metagradient Descent. In *International Conference on Learning Representations (ICLR)*, 2026. pages 2, 7, 8, 9, 15

Jérôme Bolte, Edouard Pauwels, and Samuel Vaiter. One-step differentiation of iterative algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. page 7

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, pp. 267–284, 2019. page 1

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting Training Data from Large Language Models, 2021. page 1

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership Inference Attacks From First Principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914, San Francisco, CA, USA, 2022. IEEE. pages 2, 3

Tudor Cebere, Aurélien Bellet, and Nicolas Papernot. Tighter Privacy Auditing of DP-SGD in the Hidden State Threat Model. In *International Conference on Learning Representations (ICLR)*, 2025a. page 2

Tudor Cebere, Aurélien Bellet, and Nicolas Papernot. Tighter Privacy Auditing of DP-SGD in the Hidden State Threat Model. In *ICLR*, 2025b. page 1

Mathieu Dagréou, Pierre Ablin, Samuel Vaiter, and Thomas Moreau. A framework for bilevel optimization that enables stochastic and global variance reduction algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. page 6

Mathieu Dagréou, Pierre Ablin, Samuel Vaiter, and Thomas Moreau. How to compute Hessian-vector products? In *ICLR Blogposts*, 2024. page 7

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. page 1

Justin Domke. Generic methods for optimization-based modeling. In *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012. page 7

Jinshuo Dong, Aaron Roth, and Weijie J. Su. Gaussian Differential Privacy. *arXiv preprint arXiv:1905.02383*, 2019. page 3

Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3-4):211–407, 2014. ISSN 1551-305X, 1551-3068. page 3

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography*, volume 3876, pp. 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. pages 1, 3

Mathieu Even, Clément Berenfeld, Linus Bleistein, Tudor Cebere, Julie Josse, and Aurélien Bellet. Membership Inference Attacks from Causal Principles. *arXiv preprint arXiv:2602.02819*, 2026. pages 1, 2, 3

Vitaly Feldman and Chiyuan Zhang. What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. page 3

Saeed Ghadimi and Mengdi Wang. Approximation Methods for Bilevel Programming. *arXiv preprint arXiv:1802.02246*, 2018. page 6

David Grangier, Pierre Ablin, and Awni Hannun. Adaptive Training Distributions with Scalable Online Bilevel Optimization. *Transactions on Machine Learning Research (TMLR)*, 2024. page 6

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. Studying Large Language Model Generalization with Influence Functions. *arXiv preprint arXiv:2308.03296*, 2023. pages 3, 17

Niv Haim, Gal Vardi, Gilad Yehudai, Ohad Shamir, and Michal Irani. Reconstructing Training Data from Trained Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. page 1

Frank R. Hampel. The Influence Curve and its Role in Robust Estimation. *Journal of the American Statistical Association*, 69(346):383–393, 1974. ISSN 0162-1459, 1537-274X. page 3

Jamie Hayes, Ilia Shumailov, Christopher A Choquette-Choo, Matthew Jagielski, George Kaissis, Katherine Lee, Milad Nasr, Sahra Ghalebikesabi, Niloofar Mireshghallah, Meenatchi Sundaram Mutu Selva Annamalai, et al. Strong membership inference attacks on massive datasets and (moderately) large language models. *arXiv e-prints*, pp. arXiv–2505, 2025. page 1

Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing Differentially Private Machine Learning: How Private is Private SGD? In *Advances in Neural Information Processing (NeurIPS)*, 2020. pages 1, 2

Kaiyi Ji, Junjie Yang, and Yingbin Liang. Bilevel Optimization: Convergence Analysis and Enhanced Design. In *International Conference on Machine Learning (ICML)*, 2021. page 6

Amit Keinan, Moshe Shenfeld, and Katrina Ligett. How Well Can Differential Privacy Be Audited in One Run? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. pages 1, 2

Pang Wei Koh and Percy Liang. Understanding Black-box Predictions via Influence Functions. In *International Conference on Machine Learning (ICML)*, 2017. pages 2, 3

Fred Lu, Joseph Munoz, Maya Fuchs, Tyler LeBlond, Elliott Zaresky-Williams, Edward Raff, Francis Ferraro, and Brian Testa. A General Framework for Auditing Differentially Private Machine Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. page 2

Dougal Maclaurin, David Duvenaud, and Ryan P. Adams. Gradient-based Hyperparameter Optimization through Reversible Learning. In *International Conference on Machine Learning (ICML)*, 2015. page 7

Samuel Maddock, Alexandre Sablayrolles, and Pierre Stock. CANIFE: Crafting Canaries for Empirical Privacy Measurement in Federated Learning. In *International Conference on Learning Representations (ICLR)*, 2023. page 2

Saeed Mahloujifar, Luca Melis, and Kamalika Chaudhuri. Auditing f-Differential Privacy in One Run. In *International Conference on Machine Learning (ICML)*, 2025. pages 1, 3, 15

James Martens and Roger Grosse. Optimizing Neural Networks with Kronecker-factored Approximate Curvature. *arXiv preprint arXiv:1503.05671*, 2015. page 5

Matthieu Meeus, Igor Shilov, Georgios Kaissis, and Yves-Alexandre de Montjoye. Counterfactual Influence as a Distributional Quantity. In *ICML Workshop MemFM*, 2025. page 3

Milad Nasr, Shuang Song, Abhradeep Thakurta, Nicolas Papernot, and Nicholas Carlini. Adversary Instantiation: Lower Bounds for Differentially Private Machine Learning. In *IEEE Symposium on Security and Privacy*, 2021. pages 1, 3

Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A Feder Cooper, Daphne Ippolito, Christopher A Choquette-Choo, Eric Wallace, Florian Tramèr, and Katherine Lee. Scalable extraction of training data from (production) language models. *arXiv preprint arXiv:2311.17035*, 2023a. page 1

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, 2023b. page 1

Milad Nasr, Jamie Hayes, Thomas Steinke, Borja Balle, Florian Tramèr, Matthew Jagielski, Nicholas Carlini, and Andreas Terzis. Tight Auditing of Differentially Private Machine Learning. *arXiv preprint arXiv:2302.07956*, 2023c. pages 1, 2, 3

Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Inc., USA, 1982. page 5

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. page 17

Barak A. Pearlmutter. Fast Exact Multiplication by the Hessian. *Neural Computation*, 6(1):147–160, 1994. ISSN 0899-7667, 1530-888X. page 7

Fabian Pedregosa. Hyperparameter optimization with approximate gradient. In *International Conference on Machine Learning (ICML)*, 2016. page 6

Jack Sherman and Winifred J. Morrison. Adjustment of an Inverse Matrix Corresponding to a Change in One Element of a Given Matrix. *The Annals of Mathematical Statistics*, 21(1):124–127, 1950. ISSN 0003-4851. page 14

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks against Machine Learning Models. In *IEEE Symposium on Security and Privacy (S&P).*, 2017. pages 1, 2

Shuang Song, Kamalika Chaudhuri, and Anand D. Sarwate. Stochastic gradient descent with differentially private updates. In *2013 IEEE Global Conference on Signal and Information Processing*, pp. 245–248, Austin, TX, USA, 2013. IEEE. page 3

Thomas Steinke, Milad Nasr, and Matthew Jagielski. Privacy Auditing with One (1) Training Run. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. pages 1, 3, 15

Zihang Xiang, Tianhao Wang, and Di Wang. Privacy Audit as Bits Transmission: (Im)possibilities for Audit by One Run. *arXiv preprint arXiv:2501.17750*, 2025. pages 1, 2

Mohammad Yaghini, Michael Aerni, Junrui Zhang, Florian Tramèr, and Nicolas Papernot. OptiFluence: Principled Design of Privacy Canaries. In *International Conference on Machine Learning (ICML)*, 2026. pages 2, 4, 8, 16

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pp. 268–282. IEEE, 2018. page 1

Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-Friendly Differential Privacy Library in PyTorch. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. page 17

Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. *arXiv preprint arXiv:1605.07146*, 2016. page 8

Sajjad Zarifzadeh, Philippe Liu, and Reza Shokri. Low-cost high-power membership inference attacks. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=sT7UJh5CTc. pages 1, 3

Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. Counterfactual Memorization in Neural Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. page 3

A Proofs

A.1 Proposition 4.1

For convenience, we denote $\theta^* \triangleq \theta^*(X, y)$ and $\theta^*(\tilde{X}_1, \tilde{y}_1) = \theta_1^*$. We have

$$\theta_1^* = \left[\tilde{X}_1^\top \tilde{X}_1\right]^{-1} \tilde{X}_1^\top \tilde{y}_1 = \left[\sum_{i=1}^{n} x_i x_i^\top + x_{c,1} x_{c,1}^\top\right]^{-1} \left[\sum_{i=1}^{n} y_i x_i + y_{c,1} x_{c,1}\right]$$

Let us denote $K = X^\top X$. By the Sherman-Morrison formula Sherman & Morrison (1950), we have

$$\begin{aligned}
\theta_1^* &= \left[\tilde{X}_1^\top \tilde{X}_1\right]^{-1} \tilde{X}_1^\top \tilde{y}_1 \\
&= \left(K^{-1} - \frac{K^{-1} x_{c,1} x_{c,1}^\top K^{-1}}{1 + x_{c,1}^\top K^{-1} x_{c,1}}\right) \tilde{X}_1^\top \tilde{y}_1 \\
&= \left(I - \frac{K^{-1} x_{c,1} x_{c,1}^\top}{1 + x_{c,1}^\top K^{-1} x_{c,1}}\right) K^{-1} \tilde{X}_1^\top \tilde{y}_1 \\
&= \left(I - \frac{K^{-1} x_{c,1} x_{c,1}^\top}{1 + x_{c,1}^\top K^{-1} x_{c,1}}\right) (\theta^* + y_{c,1} K^{-1} x_{c,1}) \\
&= \theta^* + y_{c,1} K^{-1} x_{c,1} - \frac{K^{-1} x_{c,1} x_{c,1}^\top}{1 + x_{c,1}^\top K^{-1} x_{c,1}} (K^{-1} y_{c,1} x_{c,1} + \theta^*) \\
&= \theta^* + \left(y_{c,1} - \frac{y_{c,1} x_{c,1}^\top K^{-1} x_{c,1} + \langle \theta^*, x_{c,1} \rangle}{1 + x_{c,1}^\top K^{-1} x_{c,1}}\right) K^{-1} x_{c,1} \\
&= \theta^* - \frac{K^{-1}}{1 + x_{c,1}^\top K^{-1} x_{c,1}} (\langle \theta^*, x_{c,1} \rangle - y_{c,1}) x_{c,1}.
\end{aligned}$$

Therefore we have

$$\begin{aligned}
\ell(\theta_1^*, z_{c,2}) &= \frac{1}{2} \left(\langle \theta_1^*, x_{c,2} \rangle - y_{c,2}\right)^2 \\
&= \ell(\theta^*, z_{c,2}) + \ell(\theta^*, z_{c,1}) \left(\frac{x_{c,1}^\top K^{-1} x_{c,2}}{1 + x_{c,1}^\top K^{-1} x_{c,1}}\right)^2 \\
&\quad - (\langle \theta^*, x_{c,1} \rangle - y_{c,1})(\langle \theta^*, x_{c,2} \rangle - y_{c,2}) \frac{x_{c,1}^\top K^{-1} x_{c,2}}{1 + x_{c,1}^\top K^{-1} x_{c,1}}.
\end{aligned}$$

Both remaining terms carry the factor $x_{c,1}^\top K^{-1} x_{c,2}$. Denoting by $r_i = \langle \theta^*, x_{c,i} \rangle - y_{c,i}$ the residual at the canary $z_{c,i}$ and factoring this term out gives $\ell(\theta_1^*, z_{c,2}) - \ell(\theta^*, z_{c,2}) = \gamma \, x_{c,1}^\top K^{-1} x_{c,2}$ with

$$\gamma = \frac{\ell(\theta^*, z_{c,1}) \, x_{c,1}^\top K^{-1} x_{c,2}}{\left(1 + x_{c,1}^\top K^{-1} x_{c,1}\right)^2} - \frac{r_1 r_2}{1 + x_{c,1}^\top K^{-1} x_{c,1}},$$

which yields the result.

A.2 Proposition 4.2

Recall that the function Φ is defined as

$$\Phi(C) = \sum_{z \in C} s(\theta^*, z) - s(\theta_C^*, z) + \mathcal{R}_{\theta^*}(C)$$

with $\theta^* \in \arg\min_{\theta \in \Theta} \mathcal{L}(\theta, D)$ and $\theta_C^* \in \arg\min_{\theta \in \Theta} \mathcal{L}(\theta, D \cup C)$. By the chain rule, we get:

$$\nabla_{C_x} \Phi(C) = \sum_{z \in C} \left[\nabla_x s(\theta^*, z) - \nabla_x s(\theta_C^*, z) - (\mathrm{d}\theta_C^*)^\top \nabla_\theta s(\theta_C^*, z)\right] + \nabla_{C_x} \mathcal{R}_{\theta^*}(C)$$

where $\mathrm{d}\theta_C^*$ is the Jacobian of $(C_x \mapsto \theta_C^*)$. Since $\mathcal{L}(\,\cdot\,, D)$ and $\mathcal{L}(\,\cdot\,, D \cup C)$ are strongly convex, the value function Φ is well-defined. Moreover, for any canary sample C, θ_C^* verifies the following first order condition

$$\nabla_\theta \mathcal{L}(\theta_C^*, D \cup C) = 0. \tag{8}$$

The Hessian matrix $\nabla^2_{\theta,\theta} \mathcal{L}(\theta_C^*, D \cup C)$ is positive definite by strong convexity and since ℓ is differentiable with respect to the input samples C_x, by the Implicit Function Theorem, we can differentiate Equation (8) with respect to C_x to get

$$\nabla^2_{\theta,\theta} \mathcal{L}(\theta_C^*, D \cup C) \mathrm{d}\theta_C^* + \nabla^2_{C_x,\theta} \mathcal{L}(\theta_C^*, D \cup C) = 0.$$

Rearranging yields the result.

B ADDITIONAL EXPERIMENTS

B.1 ADAPTING THE UPDATES TO THE AUDITED TRAINING PIPELINE

The update rules (4)–(6) assume that the audited model minimizes the empirical risk $\mathcal{L}(\theta, D \cup C)$. Since the inner problem of (3) models the audited training, we can adapt it to common components of the training pipeline, e.g. data augmentation and per-sample gradient clipping.

Data augmentation. When the audited model is trained with data augmentation, we apply the same augmentation to each mini-batch B_t in (4) and (5). Table 3 shows that applying the augmentation during the crafting increases the TPR@0.05FPR from 0.21 to 0.31 when the audited model is trained with data augmentation.

Per-example gradient clipping. DP-SGD clips each per-example gradient to a norm of at most C before averaging. Without noise, a model trained this way does not converge to a minimizer of $\mathcal{L}(\theta, D \cup C)$, but to a point θ^*_C that satisfies

$$G(\theta^*_C, D \cup C) \triangleq \frac{1}{n} \sum_{z \in D \cup C} \mathrm{Clip}_C\big[\nabla_\theta \ell(\theta^*_C, z)\big] = 0,$$

where $\mathrm{Clip}_C(g) = g \min(1, C/\|g\|)$. Applying the implicit function theorem to this equation yields the hypergradient of Proposition 4.2, where the Hessian $\nabla^2_{\theta,\theta}\mathcal{L}$ is replaced by the Jacobian $\nabla_\theta G$, and the cross-derivative $\nabla^2_{x,\theta}\ell$ by the derivative of the clipped gradient with respect to the canary input. We modify (4)–(6) accordingly. The iterate θ_t follows the clipped gradient, and the Jacobian-vector and cross-derivative products are computed through the clipping operator by automatic differentiation. Table 3 shows that accounting for clipping improves both the lower bound $\hat{\epsilon}$ and the TPR@0.05FPR when auditing DP-SGD at $\epsilon = 10$.

Table 3: Effect of adapting IBIS to the audited training pipeline, on WRN16-4 trained on CIFAR10. **Top:** non-private training with data augmentation and EMA, where the adapted IBIS applies the augmentation during crafting. **Bottom:** DP-SGD training at $\epsilon = 10$ and $\delta = 10^{-5}$, where the adapted IBIS incorporates clipping. We report mean and standard deviation over 15 runs.

	IBIS	Adapted IBIS
Data augmentation, non-private		
TPR@0.05FPR	0.21 ± 0.06	$\mathbf{0.31 \pm 0.07}$
Clipping, DP-SGD at $\epsilon = 10$		
Lower bound $\hat{\epsilon}$	0.72 ± 0.41	$\mathbf{0.98 \pm 0.38}$
TPR@0.05FPR	0.074 ± 0.017	$\mathbf{0.094 \pm 0.024}$

B.2 AUDITING DIFFERENTIAL PRIVACY IN ONE RUN

In this subsection, we use our canaries to perform one-run privacy auditing. In this experiment, we consider the WRN16-4 architecture trained on CIFAR10 with DP-SGD, and without the exponential moving average and data augmentation used previously. Accordingly, we craft IBIS canaries with the clipping-aware variant described in Appendix B.1. We audit models trained with $\epsilon \in \{1, 2, 4, 10\}$ and $\delta = 10^{-5}$ and compute an empirical lower bound $\hat{\epsilon}$ using the GDP-based one-run auditing procedure (Mahloujifar et al., 2025) since it is known to provide tighter results than the original one-run technique (Steinke et al., 2023). For each method, the same canaries are used for all values of ϵ.

For both metrics and at all privacy levels, the gap between IBIS and METAGRADIENT canaries (Boglioni et al., 2026) remains within one standard deviation. At $\epsilon = 10$, both yield clearly higher scores than random and mislabeled canaries. At $\epsilon = 1$, all methods yield lower bounds close to zero. Overall, the lower bounds exhibit high variance across runs, which reflects the difficulty of one-run auditing under DP. IBIS thus reaches an audit quality comparable to METAGRADIENT canaries at a fraction of their crafting cost, as shown in Section 5.3.

Table 4: Empirical lower bound $\hat{\epsilon}$ and TPR@0.05FPR obtained by one-run auditing of WRN16-4 trained on CIFAR10 with DP-SGD ($\delta = 10^{-5}$). We report mean and standard deviation over 15 runs.

ϵ	IBIS	METAGRADIENT	Random	Mislabeled
Lower bound $\hat{\epsilon}$				
10	0.98 ± 0.38	0.86 ± 0.30	0.21 ± 0.20	0.45 ± 0.43
4	0.21 ± 0.21	0.38 ± 0.32	0.25 ± 0.32	0.16 ± 0.16
2	0.18 ± 0.26	0.32 ± 0.31	0.15 ± 0.17	0.11 ± 0.10
1	0.13 ± 0.17	0.12 ± 0.13	0.08 ± 0.14	0.09 ± 0.11
TPR@0.05FPR				
10	0.094 ± 0.024	0.088 ± 0.023	0.047 ± 0.016	0.065 ± 0.020
4	0.063 ± 0.013	0.068 ± 0.013	0.050 ± 0.018	0.056 ± 0.015
2	0.057 ± 0.017	0.061 ± 0.015	0.051 ± 0.015	0.049 ± 0.015
1	0.055 ± 0.013	0.056 ± 0.008	0.052 ± 0.017	0.054 ± 0.012

B.3 COMPARISON BETWEEN MULTI-RUN AND ONE-RUN AUDITING

We compare the whole pipeline of multi-run and one-run auditing on WRN16-4 trained on CIFAR10 with data augmentation and EMA. For multi-run auditing, we craft a single canary with OPTIFLUENCE (Yaghini et al., 2026), which is the multi-run counterpart of IBIS, and audit it over 1288 training runs. For one-run auditing, we craft 1000 canaries with IBIS and audit them with a single training run. Table 5 reports the crafting time, the audit time and the TPR@0.05FPR. The two TPR@0.05FPR are not computed on the same statistics. In multi-run auditing, it is computed over training runs for a single canary. In one-run auditing, it is computed over canaries within a single run. Multi-run auditing yields a higher TPR@0.05FPR, but its audit requires more than 160 GPU hours. With IBIS, the whole one-run pipeline takes about 1.3 GPU hours, which is two orders of magnitude cheaper. Crafting canaries with IBIS therefore preserves the computational advantage of one-run auditing.

Table 5: Crafting time, audit time and TPR@0.05FPR of multi-run and one-run auditing of WRN16-4 trained on CIFAR10 with data augmentation and EMA. Times are in GPU hours on an NVIDIA A100. The crafting time of OPTIFLUENCE is measured in the setting without data augmentation, which uses the same crafting procedure. The one-run TPR@0.05FPR is averaged over 15 runs.

	Canary	Crafting [h]	Audit [h]	TPR@0.05FPR
Multi-run	OPTIFLUENCE	2.5	167.5	0.72
One-run	IBIS	1.2	0.13	0.31

B.4 NECESSITY OF DATA AUGMENTATION AND EXPONENTIAL MOVING AVERAGE

Following Yaghini et al. (2026), we chose to incorporate data augmentation and an exponential moving average of the weights to avoid saturating the audit metrics. Indeed, when WRN16-4 is trained on CIFAR10 without these components, the IBIS and Mislabeled canaries saturate the TPR@0.05FPR, as shown in Table 6.

Table 6: TPR@0.05FPR of one-run auditing of WRN16-4 trained on CIFAR10 without data augmentation and EMA. We report mean and standard deviation over 15 runs.

	IBIS	Random	Mislabeled
TPR@0.05FPR	1.00 ± 0.00	0.06 ± 0.01	1.00 ± 0.00

C EXPERIMENTAL DETAILS

C.1 SETTING

The experiments are run in the `PyTorch` framework Paszke et al. (2019). For DP-SGD, we use the `opacus` implementation Yousefpour et al. (2021). For the computation of the influence scores, we use the `kronfluence` package Grosse et al. (2023). Unless stated otherwise, we craft 5 canary sets with different random seeds and audit each of them with 3 independent training runs. We report mean and standard deviation over these 15 runs.

C.2 HYPERPARAMETERS

We report in Table 7 the hyperparameters used for model training and in Table 8 those used for the canary optimization pipeline. The inner learning rate η, the outer learning rate ρ and the regularization parameter λ are selected by grid search. For IBIS, the hyperparameters of the ablation study are given in Table 9. In Figure 1b, we draw 20 canaries at random from each canary set. For each of them, the variance $\mathrm{Var}_S[\Delta(z, S)]$ is estimated from 10 random subsets S.

Table 7: Training hyperparameters of WRN16-4. The same values are used on CIFAR10 and CIFAR100. For DP-SGD, the noise multiplier is calibrated with `opacus` to reach the target (ϵ, δ).

	Non-private		
Hyperparameter	Without aug.	With aug. and EMA	DP-SGD
Optimizer	SGD	SGD	SGD
Learning rate	0.1	0.1	see below
Momentum	0.9	0.9	0
Weight decay	10^{-5}	10^{-5}	0
Batch size	256	256	2048
Epochs	50	20	see below
Data augmentation	–	Random crop, horizontal flip	–
EMA decay	–	0.9	–
Clipping norm C	–	–	1
δ	–	–	10^{-5}

DP-SGD, ϵ	1	2	4	10
Learning rate	1	1	2	2
Epochs	75	125	150	200

Table 8: Hyperparameters of IBIS (Algorithm 3) on WRN16-4. The augmentation-aware and clipping-aware variants, described in Appendix B.1, are used to audit models trained with data augmentation and with DP-SGD, respectively.

	Augmentation-aware		Clipping-aware
Hyperparameter	CIFAR10	CIFAR100	CIFAR10
Influence-based selection			
Num. canaries m	1000	1000	1000
Preselection size p	10000	10000	10000
Batch size	256	256	256
SOBA			
Inner learning rate η	0.05	0.05	0.05
Outer learning rate ρ	1	1	100
Regularization λ	1	0.1	100
Epochs	30	30	30
Batch size	256	256	256
Clipping norm C	–	–	1

Table 9: Hyperparameters of IBIS in the ablation study of Figure 1, on WRN16-4 and CIFAR10. The other hyperparameters are those of the CIFAR10 column of Table 8.

	Without reg.	With reg.
Outer learning rate ρ	100	1
Regularization λ, Figure 1a	0	1
Regularization λ, Figure 1b	0	$\{0.1, 0.5, 1\}$