

Mirror Mirror on the Wall, Have I Forgotten it All?

A New Framework for Evaluating Machine Unlearning

Brennon Brimhall^{*,1}, Philip Mathew^{*,1,2}, Neil Fendley^{1,2}, Yinzhi Cao¹, and Matthew Green¹

¹Johns Hopkins University

²Johns Hopkins University Applied Physics Laboratory

May 14, 2025

Abstract

Machine unlearning methods take a model trained on a dataset \mathcal{D} and a forget set \mathcal{D}_f , then attempt to produce a model as if it had only been trained on $\mathcal{D} \setminus \mathcal{D}_f$. We empirically show that an adversary is able to distinguish between a mirror model (a control model produced by retraining without the data to forget) and a model produced by representative unlearning methods from the literature [10, 15, 18, 49]. We build distinguishing algorithms based on evaluation scores in the literature (i.e. membership inference scores) and Kullback-Leibler divergence.

We propose a strong formal definition for machine unlearning called *computational unlearning*. Computational unlearning is defined as the inability for an adversary to distinguish between a mirror model and a model produced by an unlearning method. If the adversary cannot guess better than random (except with negligible probability), then we say that an unlearning method achieves computational unlearning.

Our computational unlearning definition provides theoretical structure to prove unlearning feasibility results. For example, our computational unlearning definition immediately implies that there are no deterministic computational unlearning methods for entropic learning algorithms. We also explore the relationship between differential privacy (DP)-based unlearning methods and computational unlearning, showing that DP-based approaches can satisfy computational unlearning at the cost of an extreme utility collapse. These results demonstrate that current methodology in the literature fundamentally falls short of achieving computational unlearning. We conclude by identifying several open questions for future work.

1 Introduction

Machine learning models require massive amounts of training data. Data is collected by scraping publicly available web content [11, 29, 48], purchasing access to private databases [2, 12, 25, 32, 33, 34, 35, 39], and collecting data on their own to assemble training datasets [6, 36, 43]. Due to the massive scale, datasets cannot be thoroughly vetted and may contain data that is copyrighted, inaccurate, protected, or contain otherwise undesirable information.

Legal protections exist for those who wish to protect their privacy, copyrighted content, and financial history in multiple countries. Examples include the EU GDPR (right to be forgotten) [14], US DMCA (copyright infringement takedown) [45], US FCRA (corrections to credit history) [46], and US HIPAA (corrections to personal health data) [8]. Specific instances of training data may also be illegal on their own: for example,

^{*}Equal contribution, listed alphabetically.

it is illegal to possess child sexual abuse material (CSAM) in the US and in many other jurisdictions. Despite this, popular datasets [36] used to train models like Stable Diffusion contained illegal CSAM [41]. Further, prior work has established the threat of data poisoning attacks that create undetectable backdoors in models [17]. This means that model data may be intentionally corrupted by an adversary.

These threats can be addressed by re-training the model from scratch without the offending data. However, since training large models is capitially and computationally intensive, a major area of interest is *machine unlearning*: efficiently removing traces of the offending data, known as the *forget set*, without training a new model from scratch [1, 5, 7, 10, 15, 16, 18, 20, 31, 44].

1.1 Our Contributions

This work consists of three major contributions: (1) a new formal definition and framework for evaluating unlearning, (2) empirical results showing that many unlearning methods produce results that are distinguishable from a control, and (3) several theoretical implications of our framework.

Computational unlearning framework. Our primary contribution is a new formal definition and framework for evaluating unlearning called *computational unlearning* that we detail in §3. In brief, computational unlearning tests the ability of an adversary to distinguish between a model produced by an unlearning method and a model trained from scratch with the forget set removed. If the adversary is only able to do so with negligible probability, then we say that the unlearning method achieves computational unlearning. Because the adversary is unable to distinguish between the control and unlearned models, it follows that all information about the forget set has been “deleted” by the unlearning method. The game is defined in both a white-box (i.e. adversary has full access to model parameters) and a black-box (i.e. adversary only has API access to model) setting.

Many unlearning methods do not achieve indistinguishability. We construct two scoring methods *MIA*Score and *KLDS*Score in §4.1 which an adversary can use to distinguish between an unlearned model and a model that has never seen the forget set. We study previously proposed unlearning methods [10, 15, 18, 49] and show that each fail to achieve computational unlearning for ResNet-18 models [21] trained on CIFAR-10 [27] in §4. We also experiment with how distinguishing rates are affected by the forget set size and unlearning method hyperparameters.

Theoretical implications of computational unlearning. We describe several implications of our computational unlearning framework in §5. We first show that any deterministic computational unlearning algorithms must achieve *perfect unlearning* (i.e. it must produce the exact same model as retraining) and discuss implications for heuristic and certified removal unlearning methods. Second, we show that using differential privacy to achieve black-box computational unlearning leads to utility collapse (i.e. utility must be equivalent to a model that is randomly initialized).

2 Background

Widespread interest in machine unlearning has led to a great deal of work in the area. We briefly review several approaches explored in the literature, extending the taxonomy proposed by Nguyen et al. [31]. We categorize machine unlearning method in one of three ways: as *heuristic unlearning*, *approximate unlearning*, or *exact unlearning* as applied to classification models.

Heuristic unlearning. Unlike exact and approximate unlearning methods, *heuristic unlearning methods* do not have any formal guarantees. However, they are typically much less expensive than applying differential privacy or retraining the model [10, 15, 16, 26, 40]. These rely on various heuristics that aim to minimize an unlearning “score” that that attempts to capture how well a machine learning model has forgotten. Membership inference attacks (MIA) [38] are a popular scoring method used in the literature.

We now describe three heuristic unlearning methods: *bad teacher unlearning* [10], *amnesiac unlearning* [18], and *selective synaptic dampening (SSD)* [15]. Each of these heuristic unlearning methods are evaluated on membership inference attack (MIA) scores; this is representative of many heuristic unlearning methods.

- *Bad teacher unlearning.* Bad teacher unlearning rests on the assumption that, after forgetting a data point, a model’s behavior on that data point should be similar to that of a randomly initialized model. To forget \mathcal{D}_f the model is “taught” to reflect the behavior of a randomly initialized model (i.e. a *bad teacher*).
- *Amnesiac unlearning.* Amnesiac unlearning tries to reverse the changes to the model incurred by training on \mathcal{D}_f by keeping track of all batches containing elements from \mathcal{D}_f ; gradient *ascent* is performed on these training batches at forget time. This attempts to “backtrack” towards a model that never had those gradient updates applied. We note that this approximates the approaches taken by many exact unlearning methods.
- *Selective synaptic dampening (SSD).* SSD measures the \mathcal{D}_f -related information in each neuron by using the Fisher information matrix (FIM). Neurons that contain lots of information about examples in \mathcal{D}_f are “zeroed out” by scaling down their weights. One can think of SSD as a pruning algorithm where “branches” of the network are “removed” based on their “knowledge” of \mathcal{D}_f .

Approximate unlearning. An *approximate machine unlearning method* attempts to output a model that is approximately equal to a model trained without the forget set with high probability. Approximate machine unlearning methods are typically based on the notions of *differential privacy* [13] and *certified removal* [19].

Differential privacy. Differential privacy [13] bounds the difference between outputs of a randomized algorithm on similar data sets. In the context of machine learning, this can be implemented as either (1) producing model parameters that are similar to the model parameters produced by training on a similar dataset or (2) producing an inference that is similar to the inference produced by a model trained on a similar dataset.

In the first case, we achieve a “white-box” differential privacy property because noise is incorporated into the model parameters during the training process. A practical example of such a learning method is *differentially private stochastic gradient descent (DP-SGD)* [1].

In the second case, we achieve a weaker “black-box” differential privacy property because the noise is only integrated after training. Intuitively, this means a differentially private model has no guarantee to maintain privacy if you are given access to model parameters. Additionally, privacy budget is consumed at inference time; this forces an upper bound of the number of permitted queries.

Certified removal. Certified removal draws inspiration from the aforementioned notion of differential privacy, extending a white box privacy guarantee to hold for a learning and unlearning method. Their aim is to bound the difference in model parameters produced by the unlearning method and the model parameters produced by the learning method without a particular data point in the training set:

Definition 1 ((ϵ, δ) -Certified Removal [19]). $\text{learn}, \text{unlearn}$ satisfy (ϵ, δ) -certified removal if, for all $T \subseteq \mathcal{H}, x \in \mathcal{D} \subseteq \mathcal{U}$,

$$\mathbb{P}\left(\text{learn}\left(\text{init}\left(1^\lambda\right), \mathcal{D} \setminus x\right) \in T\right) \leq e^\epsilon \mathbb{P}\left(\text{unlearn}\left(M_o, \{x\}\right) \in T\right) + \delta$$

and

$$\mathbb{P}\left(\text{unlearn}\left(M_o, \{x\}\right) \in T\right) \leq e^\epsilon \mathbb{P}\left(\text{learn}\left(\text{init}\left(1^\lambda\right), \mathcal{D} \setminus x\right) \in T\right) + \delta$$

where $M_o = \text{learn}\left(\text{init}\left(1^\lambda\right), \mathcal{D}\right)$.

Note the similarity in this definition to the constraint imposed by differential privacy (DP). Indeed, Guo et al. mention that a DP learning algorithm is sufficient for (ϵ, δ) -certified removal. Certified removal extends the DP framework to a pair of methods (`learn`, `unlearn`) instead of a single DP mechanism. As a result, certified removal allows for models with higher base performance, since less noise needs to be added during training than is required for DP.

Notably, Guo et al. [19] introduced a *Newton update removal mechanism* to remove information from linear models with convex objectives, which has been explored further by other works [30, 47]. Zhang et al. [49] extend this to non-linear models with non-convex objectives via *certified deep unlearning*. These methods allow practitioners to have some certification regarding how “close” the unlearned model is to a model retrained from scratch without the forget set.

Certified deep unlearning extends the Newton update removal mechanism (introduced in [19]) to non-linear models with non-convex objective functions. Their approach resorts to using *projected gradient descent* (PGD) [3] for optimization, allowing for the guarantees introduced by Guo et al. [19] to be extended to these models.

Despite certified removal being defined as a *white-box* property — that is, an adversary seeing model parameters — literature typically evaluates unlearning performance by computing membership inference attack scores (MIA) [38] that do not have access to the model weights (i.e. a *black-box* evaluation setting). We discuss certified removal methods in §4.

Exact unlearning. An *exact unlearning method* modifies the original model such that its outputs exactly match a model trained without the forget set. We are unaware of any exact unlearning method for neural networks that does not involve some degree of retraining. The most common approaches rely on saving checkpoints of model state at train time [5, 44]. Unlearning then consists of rewinding to a checkpoint that has not been influenced by the forget set and then resuming training from that point without the forgotten data. This technique is essentially a time-space tradeoff; multiple checkpoints of the model must be saved out during training. The worst-case retraining cost may be equivalent to retraining the model from scratch (for example, if the forget set contains an element from the first batch). We do not study exact unlearning in this work.

Unlearning evaluation. Many machine unlearning works attempt to justify their approach by optimizing some *unlearning score*. Examples include accuracy gap scores and membership inference attacks:

- *Accuracy Gap Scores:* A popular approach in prior work is to demonstrate an “accuracy gap” between the unlearned model’s performance when queried on the forget set (the set of examples to be forgotten) and the retain set (the training set minus the forget set) in a black-box manner [5, 7, 9, 16]. The intuition here is that the unlearning method has “remembered” the retain set (because it maintains good accuracy) but “forgotten” the forget set (because it has lost accuracy on the items to be forgotten).
- *Membership Inference Attack Scores:* Membership inference attacks, formalized in [38], are another common way to evaluate the performance of machine unlearning algorithms in literature. An attacker is given a data point and black-box access to a machine learning model; the attacker then attempts to predict if the data point was part of the training set. Some heuristic machine unlearning proposals are specifically designed to minimize these scores [10, 15, 18].

Framing machine unlearning in a score-based manner is attractive: it provides an easy way to facilitate comparison, and it also satisfies intuitive beliefs about how the model should behave after unlearning. However, the use of score-based definitions does not fully capture the expectations of model behavior after unlearning.

Prior work has demonstrated multiple issues with unlearning evaluation:

- *Over-Unlearning:* Hu et al. [23] considers the phenomenon of *over-unlearning*, where an attacker with black-box access to a model has the model provider run an unlearning method adversarially with the

intent of reducing overall model utility. A key feature of this scenario is that an attacker can adaptively issue requests to forget data.

- *Unlearning Inversion*: Hu et al. [24] shows that many machine unlearning techniques are vulnerable to *unlearning inversion attacks*. In this scenario, an attacker with white-box access is able to recover an unlearned dataset by considering the differences between an original and unlearned model. This is similar in spirit to membership inference attacks.
- *Forgeability*: Recent work [37, 42] demonstrates that current machine unlearning definitions permit *forgeability*. Forgeability is achieved if an adversary can *forge* a proof that demonstrates their model was trained without the forget set despite being trained with it. An adversary is permitted to order the retained data points in batches. This allows them to arrive at the same weights as a model trained on the forget set.

We believe that these issues fundamentally stem from the score-based approach to evaluating machine unlearning and that repairing this requires a fundamentally different formal definition of unlearning based on indistinguishability.

Machine unlearning is indistinguishability. We first motivate our desired functionality through the lens of the k -nearest neighbors (k -NN) machine learning algorithm. k -NN is a simple learning algorithm that memorizes every training example it is presented with. At inference time, the model finds the k nearest training examples according to some metric and classifies based on their round truth labels. One of the properties of k -NN is that it immediately admits an unlearning algorithm: simply delete the training examples you wish to forget. This produces a model that is indistinguishable from a control.

We claim that *indistinguishability* represents a superior way to evaluate unlearning. This idea is not new and features in prior work [15, 19, 49] but is not measured directly. We propose doing so here. In other words, no efficient (p.p.t., or probabilistic polynomial time) adversary \mathcal{A} should be able to distinguish between M_u and M_c . We also assume that \mathcal{A} has access to M_o , `learn`, `unlearn`, \mathcal{D} , and \mathcal{D}_f .

3 Formalizing Unlearning

We propose *computational machine unlearning* as a formal way to capture that machine unlearning is indistinguishability. Unlike prior machine unlearning scores, our definition is defined as a security game, inspired by the cryptographic notion of semantic security and indistinguishability under chosen plaintext attack (IND-CPA) [4]. Instead of considering a membership inference score or accuracy gap, computational unlearning considers the ability of an adversary to distinguish between an unlearned model and a control model.

3.1 Preliminaries

Let \mathcal{U} be the universe of all possible data, and $d \in \mathcal{U}$ be a particular data point. Let $\mathcal{D} \subseteq \mathcal{U}$ be our entire training dataset with $\mathcal{D}_f \subseteq \mathcal{D}$ be the forget set. Let \mathcal{H} be our hypothesis space of possible models, with $h \in \mathcal{H}$ being a particular model.

Definition 2 (Learning scheme). We formally define a *learning scheme* as a tuple of probabilistic polynomial time (p.p.t.) algorithms (`init`, `learn`, `infer`):

- `init`(1^λ) $\rightarrow h$: randomly samples some initial model h . The notation 1^λ simply denotes that there are λ copies of the symbol 1 written on the input tape of the Turing machine and 0 in every other location. This ensures that `init` runs in polynomial time with respect to λ , a cryptographic formality.
- `learn`(h, \mathcal{D}) $\rightarrow h$: given some initial model h , performs some model update process with respect to the training set \mathcal{D} .

- $\text{infer}(h, d) \rightarrow \mathbb{R}^n$: performs some inference procedure with the given model h on the provided data point d .

Remark 3 (Baseline and meaningful utility). A newly initialized model under some learning scheme (i.e. the output of init) is expected to only have some baseline utility. Formally, we say the following:

$$\mathbb{E}(\text{util}(\text{init}(1^\lambda))) = b$$

for some b , where $\text{init}(1^\lambda)$ initializes a model and does no training on it. We say utility is *meaningful* if it is larger than b .

Definition 4 (Forgetting learning scheme). We likewise define a *forgetting learning scheme* as a tuple of p.p.t algorithms ($\text{init}, \text{learn}, \text{infer}, \text{unlearn}$) such that it is a learning scheme with an additional unlearn algorithm:

- $\text{unlearn}(h, \mathcal{D}_f) \rightarrow h$: performs some model update process with respect to the forget set \mathcal{D}_f .

Remark 5 (Cost and utility functions). We also assume the existence of cost and utility functions for learning schemes and forgetting learning schemes.

$$\text{cost} : \Phi \rightarrow \mathbb{R}^+$$

$$\text{util} : \mathcal{H} \rightarrow \mathbb{R}^+$$

where cost measures the expense of performing a model update algorithm (e.g. $\text{learn}, \text{unlearn}$) and util is a function that measures the performance of a particular model in the hypothesis space.

Definition 6 (Negligible function). We define $\text{negl}(\lambda)$ to be a function that is *negligible* in terms of a security parameter λ . We borrow the definition of a negligible function from cryptography — namely, that a function $f : \mathbb{Z}_{\geq 1} \rightarrow \mathbb{R}$ is negligible if and only if for all $c > 0$ we have:

$$\lim_{n \rightarrow \infty} f(n)n^c = 0$$

For example, if a function is bounded by $2^{-\lambda}$ then that function is negligible with respect to λ .

3.2 Computational Unlearning

We now formally define *computational unlearning* in both white-box and black-box settings. As described earlier, these definitions are inspired by the cryptographic notions of semantic security and indistinguishability under chosen plaintext attack (IND-CPA) [4]. We assume a computationally bound adversary and allow a negligible adversary advantage in keeping with these formal cryptographic definitions.

Definition 7 (White-Box Computational Unlearning). We consider the following experiment:

1. \mathcal{C} sends the description of the forgetting learning scheme (i.e. the learn and unlearn algorithms).
2. \mathcal{A} chooses \mathcal{D} and sends it to \mathcal{C} .
3. \mathcal{C} computes $M_o \leftarrow \text{learn}(\text{init}(1^\lambda), \mathcal{D})$ and sends $(M_o, \text{learn}, \text{unlearn}, \mathcal{D}, \text{cost}, \text{util})$ to \mathcal{A} .
4. \mathcal{A} selects a forget set $\mathcal{D}_f \subset \mathcal{D}$ and sends \mathcal{D}_f to \mathcal{C} .
5. \mathcal{C} computes $M_u \leftarrow \text{unlearn}(M_o, \mathcal{D}_f)$ and computes $M_c \leftarrow \text{learn}(\text{init}(1^\lambda), \mathcal{D} \setminus \mathcal{D}_f)$.
6. \mathcal{C} samples a random bit $b \xleftarrow{\$} \{0, 1\}$. If $b = 0$, \mathcal{C} sends $[M_c, M_u]$. If $b = 1$, \mathcal{C} sends $[M_u, M_c]$.
7. \mathcal{A} computes a guess b' and sends b' to \mathcal{C} . \mathcal{A} wins the game if $b' = b$.

We say that an unlearning algorithm is a *white-box computational machine unlearning algorithm* if

$$\mathbb{P}(b' = b) < \frac{1}{2} + \text{negl}(\lambda)$$

We denote this computational indistinguishability by saying $M_u \stackrel{c}{\approx} M_c$. This game is illustrated in Figure 1.

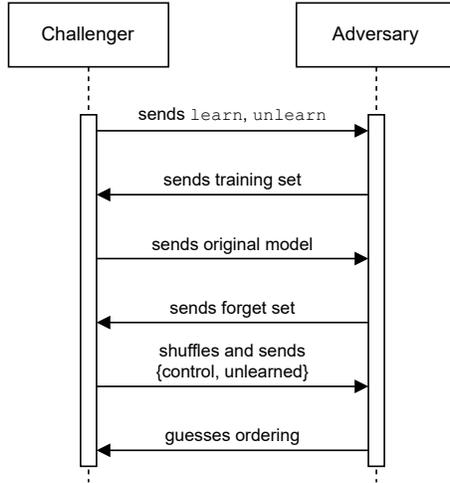


Figure 1: Overview of the security game for computational unlearning.

Definition 8 (Black-Box Computational Unlearning). We consider the following experiment:

1. \mathcal{C} sends the description of the forgetting learning scheme (i.e. the `learn` and `unlearn` algorithms).
2. \mathcal{A} chooses \mathcal{D} and sends it to \mathcal{C} .
3. \mathcal{C} computes $M_o \leftarrow \text{learn}(\text{init}(1^\lambda), \mathcal{D})$ and sends $(M_o, \text{learn}, \text{unlearn}, \mathcal{D}, \text{cost}, \text{util})$ to \mathcal{A} .
4. \mathcal{A} selects a forget set $\mathcal{D}_f \subset \mathcal{D}$ and sends \mathcal{D}_f to \mathcal{C} .
5. \mathcal{C} computes $M_u \leftarrow \text{unlearn}(M_o, \mathcal{D}_f)$ and computes $M_c \leftarrow \text{learn}(\text{init}(1^\lambda), \mathcal{D} \setminus \mathcal{D}_f)$.
6. \mathcal{C} samples a random bit $b \xleftarrow{\$} \{0, 1\}$. If $b = 0$, \mathcal{C} sends $[\mathcal{O}_{M_c}, \mathcal{O}_{M_u}]$ where \mathcal{O} is an oracle that allows \mathcal{A} to call `infer` on the underlying model. If $b = 1$, \mathcal{C} sends $[\mathcal{O}_{M_u}, \mathcal{O}_{M_c}]$.
7. \mathcal{A} computes a guess b' and sends b' to \mathcal{C} . \mathcal{A} wins the game if $b' = b$.

We say that an unlearning algorithm is a *black-box computational machine unlearning algorithm* if we have:

$$\mathbb{P}(b' = b) < \frac{1}{2} + \text{negl}(\lambda)$$

Remark 9 (Threat Model). This definition intuitively captures a setting inspired by the GDPR process: we assuming the adversary is a user who can select which data should be deleted (i.e. the set of items to be deleted is, to some extent, adversarially-controlled) as in [23]. We also acknowledge that this game defines a very strong adversary and that a real-world adversary may not have access to the full training set, the description of the unlearning algorithm, or other information provided in this game. However, each of these alternatives envisions a strictly weaker adversary than our computational learning game, meaning that an unlearning method that achieves computational unlearning would still be indistinguishable from a control model in these scenarios.

Remark 10 (Trivial Solutions). Observe that there is a trivial solution for computational unlearning: simply defining `unlearn` to call `init`(1^λ) and emit new models whose weights are initialized randomly. To prevent these trivial solutions, we require that $|\text{util}(M_o) - \text{util}(M_c)| < \epsilon$ for some small ϵ and that $\text{cost}(\text{learn}(\mathcal{D} \setminus \mathcal{D}_f)) < \text{cost}(\text{unlearn}(M_o, \mathcal{D}_f))$.

Remark 11 (Utility Equivalence). We note that if the definition above is met, then $\text{util}(M_c) \stackrel{c}{\approx} \text{util}(M_u)$ implicitly holds. If this was not the case, it would allow \mathcal{A} to distinguish the two models.

4 Empirical Results

We now present empirical distinguishers for \mathcal{A} to evaluate if unlearning methods from literature achieve computational unlearning. We experimentally demonstrate the effectiveness of these distinguishing algorithms on heuristic unlearning and certified removal methods.

4.1 Distinguisher Scores

Each distinguisher for \mathcal{A} uses a *scoring function* to separate M_c from M_u . The scoring function takes in the original model M_o , a candidate model $M \in \{M_1, M_2\}$, the training set \mathcal{D} , and the forget set \mathcal{D}_f . The scoring function then outputs a value s that is used to determine if the candidate model is M_u or M_c .

Scoring with membership inference attacks. As described in §2, membership inference attacks (MIA) are a common method for evaluating the performance of a given unlearning algorithm and several unlearning methods are justified by reducing them as much as possible. However, we are able to leverage these scores to distinguish an unlearned model from a control model *because the unlearning method will often produce models whose MIA scores are out of distribution*. We propose that an unlearning algorithm should achieve similar MIA scores to a model that never saw the forget set rather than attempting to absolutely minimize it. In practice, we use the approach of Shokri et al. [38] for computing MIA scores using the same implementation as [15]. We refer to this scoring algorithm as **MIAScore**.

Scoring with Kullback-Leibler divergence. We also present a novel scoring method **KLDScore**. We drew inspiration from the fact that Certified Removal bounds the KL-Divergence between different models. To calculate the score, \mathcal{A} calculates the KL-Divergence between the inferences of the original model M_o and the candidate model M (such as on instances in or near the forget set). This provides a measure of how different the behaviors of M and M_o are. In practice, we find that models produced by unlearning methods have much lower divergence from the original model than a control.

$$\text{KLDScore}(M_o, M, \mathcal{D}, \mathcal{D}_f) = \sum_{x_i \in \mathcal{D}_f} D_{\text{KL}}(M(x_i + \mathcal{N}(0, 0.1)) \parallel M_o(x_i + \mathcal{N}(0, 0.1))) \quad (1)$$

where $\mathcal{N}(0, 0.1)$ represents Gaussian noise with mean 0 and variance 0.1.

Choice of decision rule. \mathcal{A} will compute b' using the results from one of the aforementioned scoring algorithms. By Definitions 7 and 8, \mathcal{A} is free to use prior knowledge of **learn**, **unlearn**, \mathcal{D} , and \mathcal{D}_f in the decision rule. We refer the reader to Kerckhoffs’s principle in cryptography.

4.2 Experimental Results

We evaluate the distinguishers via their success rates in differentiating between M_u and M_c . We present our findings from two experiments: one varying the size of the forget set \mathcal{D}_f and the other varying the σ parameter from Certified Deep Unlearning. Both experiments were run using an Intel Xeon Gold 6330 and a NVIDIA A40. All results are statistically significant (i.e. a 95% confidence interval under a Beta distribution with the Jeffries prior does not contain 50%).

Implementation Details All models used the ResNet-18 [22] architecture. The original and control models were trained using stochastic gradient descent with momentum and weight decay. The hyperparameters used are as follows:

- Number of epochs: 50
- Batch size: 512

- Learning rate: 10^{-2}
- Weight decay: 5×10^{-4}

For SSD [15], we used a dampening constant of 1 and a selection weighting of 100. For all other methods [10, 18, 49], we used the parameters specified in their original papers (with the exception of σ for CDU [49], which we varied as described below).

Forget set size. We evaluated the effect of the forget set size on four different unlearning techniques. We used three heuristic methods and the approximate technique Certified Deep Unlearning (CDU) all discussed in §2.

For each method, a random subset of \mathcal{D} was chosen as the forget set. We varied the forget set size to evaluate its effect on the ability of \mathcal{A} to distinguish between M_u and M_c and correctly guess b' using the distinguishing algorithms discussed above. We ran 128 trials, each with a different randomly selected forget set. We found that with increased forget set size the adversary was able to correctly guess b' with higher frequency, but always maintained above a 60% success rate at every forget set size. As we hypothesized, many heuristic unlearning techniques over-minimized **MIAScore** during their process of unlearning: for all heuristic unlearning methods the decision rule assigns a lower **MIAScore** score to M_u (except for SSD [15] with greater than 30 forget set examples).

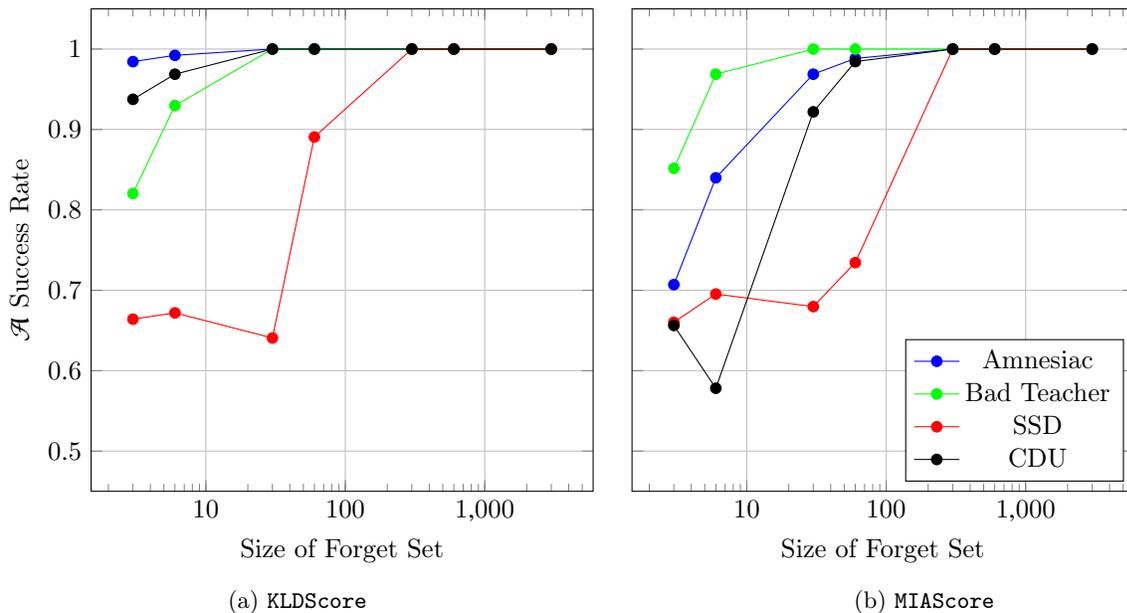


Figure 2: Forget set size against adversary success rate using **KLDScore** and **MIAScore** distinguishers.

We also explored *classwise* unlearning, where an entire class in \mathcal{D} is chosen as the forget set. We found it always possible to distinguish in this setting (e.g. 100% adversary success rate under both distinguishers). This is unsurprising given our results on the impact of forget set size. Recall that CIFAR-10 has 50,000 images in the training set, distributed evenly across 10 classes; forgetting an entire class amounts to a forget set size of 5,000 [27].

Dependence on σ . We additionally explored the relationship between computational unlearning and certified removal’s privacy parameters. For this we examined \mathcal{A} ’s **KLDScore** for certified deep unlearning (CDU) from Zhang et al. [49] with different hyperparameters. The CDU method is based on a single hyperparameter σ , derived from ϵ and δ values, that represents the magnitude of noise used. We follow

the hyperparameters from the CDU published experiments [49], including a random forget set of 1000 data points. We then varied σ from 10^{-5} to 10^{-1} in powers of 10 running 128 trials at each value.

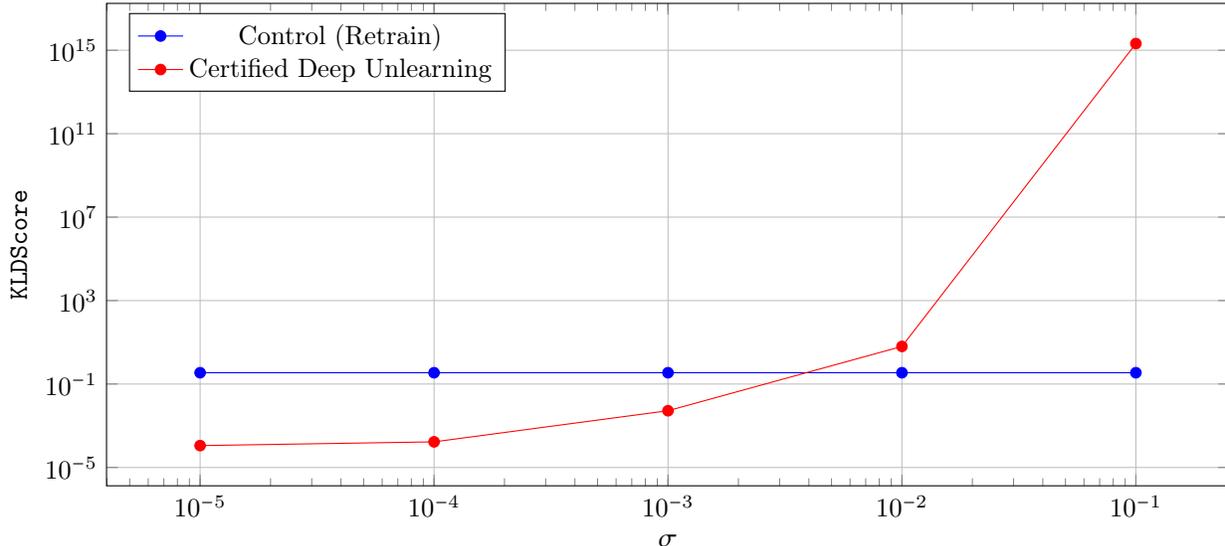


Figure 3: Certified Deep Removal against KLDScore for different values of σ .

In our experiments we found that the adversary was able to distinguish using KLDScore with 100% accuracy for all choices of σ . We found as σ increases the unlearned model’s KLDScore also increases (see Figure 3). Since the control model has no dependency on σ , an adversary can distinguish with extremely high success rate by choosing a decision rule appropriate for the chosen value of σ . This relationship does imply there is a point of intersection (between 0.001 and 0.01) where the KLDScore score for M_u and M_c should be very close, making it harder to distinguish using KLDScore. We believe understanding the intersection constitutes an interesting topic for future work.

5 Theoretical Analysis

We now show several interesting consequences of our computational unlearning definition. We begin by showing that k -NN admits a white-box computational unlearning algorithm in line with the technical intuition from §2.

Theorem 12 (k -NN admits white-box computational unlearning). *There is an efficient white-box computational unlearning algorithm for k -NN models.*

Proof of Theorem 12. Let **learn** be defined as normal for k -NN models. Let **unlearn** be defined as deleting the specified \mathcal{D}_f from the k -NN database. Observe that this produces the same database as **learn** on $\mathcal{D} \setminus \mathcal{D}_f$. Therefore, an adversary cannot distinguish between M_u and M_c with non-negligible advantage because they are exactly the same model. \square

We first show that for entropic machine learning algorithms (e.g. stochastic gradient descent) there are no deterministic algorithms that can achieve computational unlearning. This result means that many heuristic unlearning methods can never admit computational unlearning algorithms. Secondly, we show that differentially private algorithms can achieve computational unlearning at the cost of collapsing model utility.

5.1 Deterministic Computational Unlearning does not exist

Preliminaries. We now show that a forgetting learning scheme that is entropic must have a randomized unlearning algorithm. Additionally, we show that a forgetting learning scheme that is deterministic must achieve perfect unlearning. Because forgetting learning schemes that are entropic must be randomized and because forgetting learning schemes that are deterministic must be perfect, we say that *deterministic computational learning does not exist*.

Before beginning, we define *entropic learning schemes* and *perfect unlearning*.

Definition 13 (Deterministic learning scheme). A learning scheme is deterministic if the distribution of models produced by $\text{learn}(\text{init}(1^\lambda), \mathcal{D})$ has Shannon entropy of 0.

Definition 14 (Entropic learning scheme). A learning scheme is *h-entropic* if the distribution of models produced by $\text{learn}(\text{init}(1^\lambda), \mathcal{D})$ has Shannon entropy greater than or equal to h . In the absence of a particular value specified for h , we take h to be 1 bit.

Remark 15. If a learning scheme is entropic, it cannot be deterministic. For all practical purposes, learning schemes are either deterministic (i.e. k -nearest neighbors) or entropic (i.e. randomly initialized neural nets trained under stochastic gradient descent).

Definition 16 (Perfect unlearning). We say a forgetting learning scheme achieves *perfect unlearning* algorithm if, for all $M = \text{learn}(\text{init}(1^\lambda), \mathcal{D})$, the following always holds:

$$\text{learn}(\text{init}(1^\lambda), \mathcal{D} \setminus \mathcal{D}_f) = \text{unlearn}(M, \mathcal{D}_f)$$

This is to say, unlearn is perfect if it produces *exactly the same* model as retraining on the retain set.

Remark 17. A perfect unlearning algorithm is distinct from the notion of exact unlearning described in §2. Literature that explores exact unlearning rewinds the learning process to the first step that used items from the forget set; in other words, it implements the below:

$$\text{learn}\left(\text{learn}\left(\text{init}\left(1^\lambda\right), \mathcal{D} \setminus \mathcal{D}_f\right), \mathcal{D} \setminus \mathcal{D}_f\right)$$

Recall that in our definition, \mathcal{A} is given unlearn , the description of the unlearning method, and also has access to the original model M_o . Intuitively, this means that an adversary can simply *run the unlearning method on its own*.

Because the unlearning algorithm is deterministic and the learning scheme is entropic, this means that only one of the two models will exactly match the adversary's own computed result with high probability and allow the adversary distinguish with non-negligible probability.

We now prove the above for entropic learning schemes that are forgetting and achieve computational unlearning.

Theorem 18. *There are no deterministic computational unlearning algorithms for entropic learning schemes.*

Proof of Theorem 18. Suppose that a forgetting learning scheme is entropic. Therefore, $\text{learn}(\text{init}(1^\lambda), \mathcal{D})$ is a randomized algorithm that samples some $h \in \mathcal{H}$ with minimum entropy greater than 1 bit. Let $\mathbb{P}(h)$ be the probability that learn samples a particular $h \in \mathcal{H}$ and let

$$p_{\max} = \max_{\forall h \in \mathcal{H}} \mathbb{P}(h)$$

Now suppose that the challenger uses a deterministic unlearn algorithm. Then the adversary can also run unlearn on M_o and will win the game if $M_c \neq M_u$. Because the probability learn will output a particular model is bounded by p_{\max} , the probability that $M_c = M_u$ is also bounded by p_{\max} and the probability $M_c \neq M_u$ is at least $1 - p_{\max}$. Because unlearn is a computational unlearning algorithm, we must have that $1 - p_{\max} < \frac{1}{2} + \text{negl}(\lambda)$. We can rearrange symbols to get that $\text{negl}(\lambda) > \frac{1}{2} - p_{\max}$. But we have a contradiction because p_{\max} does not asymptotically approach $\frac{1}{2}$ as λ approaches infinity. \square

We now show that a forgetting learning scheme that is deterministic and achieves computational unlearning must be perfect. The intuition for this result is similar: the adversary has access to `learn`, the description of the learning algorithm, and has access to the $\mathcal{D} \setminus \mathcal{D}_f$. This means that the adversary can compute the control model on their own, use its own control model to identify the control model provided by the challenger, and distinguish with non-negligible probability.

Theorem 19. *Let \mathcal{L} be a forgetting learning scheme that is deterministic. Then if it satisfies the computational unlearning notion of Definitions 7 and 8 it must perfectly unlearn under Definition 16.*

Proof of 19. Suppose that \mathcal{L} is a deterministic learning scheme. Therefore, it must output a single model for a given training set \mathcal{D} . Suppose \mathcal{L} is also forgetting and achieves computational unlearning. We now consider two possible cases: that `unlearn` is randomized and that it is deterministic.

- *Randomized case:* Suppose that `unlearn` is a randomized algorithm that samples some $h \in \mathcal{H}$. Let $\mathbb{P}(h)$ be the probability that `unlearn` selects a particular $h \in \mathcal{H}$ and let

$$p_{\max} = \max_{\forall h \in \mathcal{H}} \mathbb{P}(h)$$

Recall that in this scenario, the challenger uses a deterministic `learn` algorithm to produce M_c . Then the adversary can also run `learn` to produce M_c and will win the game if $M_c \neq M_u$. Because the probability `unlearn` will output a particular model is bounded by p_{\max} , the probability that $M_c = M_u$ is also bounded by p_{\max} and the probability $M_c \neq M_u$ is at least $1 - p_{\max}$. Because `unlearn` is a computational unlearning algorithm, we must have that $1 - p_{\max} < \frac{1}{2} + \text{negl}(\lambda)$. We can rearrange symbols to get that $\text{negl}(\lambda) > \frac{1}{2} - p_{\max}$. But we have a contradiction because p_{\max} does not asymptotically approach $\frac{1}{2}$ as λ approaches infinity.

- *Deterministic case:* Now suppose that `unlearn` is a deterministic algorithm. Then the adversary can also run `learn` and `unlearn` on M_o and will win the game if $M_c \neq M_u$. Because `learn` and `unlearn` are deterministic and will each output a particular model for a given dataset, we must have that $M_c = M_u$. Thus, `unlearn` must be a *perfect* unlearning algorithm.

□

Remark 20 (Viability of computational unlearning methods). These results constrain the space of learning algorithms that are compatible with unlearning. To reiterate: Theorem 18 shows that entropic learning schemes that are forgetting and achieve computational unlearning must have a randomized unlearning method. In the opposite direction, no deterministic learning algorithms can support entropic unlearning algorithms. Any deterministic learning scheme that is forgetting and achieves computational unlearning must implicitly realize a *perfect* unlearning scheme, as noted in Theorem 19. As a consequence of these findings, any forgetting learning schemes that achieves computational unlearning must either be perfect, or both the learning and unlearning process must inherently be randomized. Note that Certified Deep Unlearning [49] and many heuristic unlearning methods we studied in §4 are not randomized and are not perfect. Thus, they can never achieve computational unlearning.

5.2 Computational Unlearning from Differential Privacy Collapses Utility

One natural approach to constructing computational unlearning uses techniques from differential privacy [13].

While differentially private learning algorithms imply the existence of black-box computational unlearning, the parameters choices required to achieve computational unlearning will lead to utility collapse for the resulting models. We claim that the ϵ and δ parameters must be phrased in terms of λ and that values needed to obtain security imply unacceptably high utility loss.

We begin by recalling the definition of privacy loss and differential privacy.

Definition 21 (Privacy Loss, [13]). The privacy loss \mathcal{L} over neighboring databases x, y after observing ξ is given by:

$$\mathcal{L}_{\mathcal{M}(x) \parallel \mathcal{M}(y)}^{(\xi)} = \ln \left(\frac{\mathbb{P}(\mathcal{M}(x) = \xi)}{\mathbb{P}(\mathcal{M}(y) = \xi)} \right)$$

Definition 22 (Differential Privacy, [13]). A randomized algorithm \mathcal{M} with domain $\mathbb{N}^{|\mathcal{X}|}$ is (ϵ, δ) -differentially private if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{M})$ and for all $x, y \in \mathbb{N}^{|\mathcal{X}|}$ such that $\|x - y\|_1 \leq 1$:

$$\mathbb{P}(\mathcal{M}(x) \in \mathcal{S}) \leq e^\epsilon \cdot \mathbb{P}(\mathcal{M}(y) \in \mathcal{S}) + \delta$$

If $\delta = 0$, we say that \mathcal{M} is ϵ -differentially private.

Differential privacy's definition bounds the privacy loss from any query, which we discuss below.

Remark 23 (Privacy Loss Bounded for Differentially Private Algorithms, [13]). Suppose that \mathcal{M} is a (ϵ, δ) -differentially private algorithm. Then by definition, the absolute value of the privacy loss $\mathcal{L}_{\mathcal{M}(x) \parallel \mathcal{M}(y)}^{(\xi)}$ is bounded by ϵ with probability at least $1 - \delta$.

Remark 24 (Differential Privacy is Immune to Post-Processing, [13]). Additionally, one of the most useful properties of differential privacy is that it is “immune” to post-processing. This means that there exists no algorithm that, given the output of a differentially-private function, can “undo” the differential privacy. We refer the reader to [13, Proposition 2.1] for the proof of this claim.

We will use this property to show that differential privacy can be used to satisfy the definition of black-box computational unlearning (Definition 8).

Lemma 25. *Privacy Loss is an upper bound on relative entropy.*

Proof of Lemma 25. Recall the definition of relative entropy (Kullback-Leibler divergence) of probability distribution Q with respect to P [28]:

$$D_{\text{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \ln \left(\frac{P(x)}{Q(x)} \right) \quad (2)$$

Now, suppose we have some randomized algorithm \mathcal{M} with inputs a, b . Let P, Q represent the output distributions of $\mathcal{M}(a), \mathcal{M}(b)$ respectively. Let \mathcal{L}_{\max} refer to the maximum privacy loss observed for any element x .

$$\begin{aligned} (2) &= \sum_{x \in \mathcal{X}} P(x) \ln \left(\frac{\mathbb{P}(\mathcal{M}(a) = x)}{\mathbb{P}(\mathcal{M}(b) = x)} \right) \\ &= \sum_{x \in \mathcal{X}} P(x) \mathcal{L}_{\mathcal{M}(a) \parallel \mathcal{M}(b)}^{(x)} \\ &\leq \sum_{x \in \mathcal{X}} \mathcal{L}_{\max} \end{aligned}$$

Because P is a probability distribution, we have that $P(x) \in [0, 1]$. Then privacy loss is an upper bound because the relative entropy is equal to the privacy loss multiplied by $P(x)$ by definition. \square

Having reviewed important properties of differential privacy, we now show how to construct black-box computational unlearning (Definition 8) from differential privacy. There are two main ways to accomplish this: to use differential privacy directly or to aggregate the outputs of models in a differentially private way. The theorem below captures both of these cases.

Theorem 26 (Differentially private computational unlearning). *Let \mathcal{L} be a forgetting learning scheme that achieves black-box computational unlearning. Let `unlearn` simply output the original model (with fresh randomness for the differentially private mechanism). Then `learn` and `unlearn` satisfy the definition of black-box computational unlearning (Definition 8) if and only if $\delta \leq \text{negl}(\lambda)$ and let $\epsilon \leq \ln(1 + \text{negl}(\lambda))$.*

Proof of Theorem 26. Observe that the privacy loss is negligible in λ with overwhelming probability. This means that the relative entropy between the outputs of M_u and M_c is negligible by Lemma 25. By Remark 24, there is no algorithm an adversary can use to increase the relative entropy. So then M_u and M_c are computationally indistinguishable.

We now show that our bounds are tight. Suppose that $\delta > \text{negl}(\lambda)$. Then the privacy loss guarantee does not hold with overwhelming probability and an adversary could obtain a query result with non-negligible privacy loss after a polynomial number of queries.

Alternatively, suppose that $\epsilon > \ln(1 + \text{negl}(\lambda))$. Then the privacy loss guarantee is at least polynomial in λ and an adversary could obtain query results that lead to a non-negligible privacy loss after a polynomial number of queries. \square

Unfortunately this approach also has the following undesirable result:

Corollary 27. *Let \mathcal{L} be a forgetting learning scheme that achieves black-box computational unlearning, with `learn` implemented as described in Theorem 26. Then M_u and M_o are also computationally indistinguishable. This implies that the utility of M_o is equivalent to the utility of M_u .*

Proof of Corollary 27. We follow the proof of Theorem 26. Observe that the privacy loss is negligible in λ with overwhelming probability. This means that the relative entropy between the outputs of M_u and M_c is negligible. But M_u is the same model as M_o , with fresh randomness for the differential privacy mechanism. So M_u and M_o are also computationally indistinguishable.

In other words, this means that $\text{util}(M_o) \stackrel{c}{\approx} \text{util}(M_u)$. Since C does not know *a priori* the choice of \mathcal{A} , `unlearn` must be indistinguishable for all possible choices. So then $M_u \stackrel{c}{\approx} M_c$ for $\mathcal{D}_f = \mathcal{D}$. That is to say that $M_u \stackrel{c}{\approx} \text{learn}(\text{init}(1^\lambda), \emptyset)$. But we because $\text{util}(M_o) \stackrel{c}{\approx} \text{util}(M_u)$ we also have $\text{util}(M_o) \stackrel{c}{\approx} \text{util}(\text{learn}(\text{init}(1^\lambda), \emptyset))$, which is bounded by a small ϵ and thus not meaningful. \square

Remark 28 (Black-box infeasibility implies white-box infeasibility). The security notion of white-box computational unlearning in Definition 7 is strictly stronger than the black-box computational unlearning of Definition 8. Thus, the an infeasibility result for black-box computational unlearning immediately implies an infeasibility result for white-box computational unlearning.

We note that most use cases of differentially private `infer` algorithms are designed to support a number of queries bounded by a constant. One possible interpretation of our result is that we assume an adversary is able to query the model some polynomial number of times.

We additionally stress that Theorem 26 and Corollary 27 only consider applying differential privacy to the `infer` algorithm of a learning scheme. Our result does not necessarily imply a utility collapse for a forgetting learning scheme that achieves computational unlearning with a differentially private `learn` algorithm.

6 Discussion

Several unlearning methods are deterministic. Several unlearning methods in the literature are deterministic. This means that an adversary will always be able to distinguish in the computational unlearning game (see §5.1). In particular, [15, 18, 19, 49] each provide deterministic unlearning methods and will never achieve computational unlearning. This includes various certified removal and approximate techniques; while they are randomized during the `learn` process they are not randomized during the `unlearn` process.

In contrast to the above methods, [1, 10] are randomized. Therefore, it is possible that they could achieve computational unlearning if other issues leading to distinguishing attacks are resolved.

See Table 1 for an overview.

Name	Category	Randomized
DP-SGD [1]	Approximate	✓
Certified Removal [19, 49]	Approximate	✗
SSD [5]	Heuristic	✗
Amnesiac [18]	Heuristic	✗
Bad Teacher [10]	Heuristic	✓

Table 1: Summary of the randomization of unlearning methods.

Bounding the difference between models does not imply indistinguishability. As described in §4, we compare (1) the KL divergence between M_o and M_c and (2) the KL divergence between M_o and the M_u to distinguish certified removal models with advantage far higher than would be expected from the δ parameter. This may seem unintuitive at first. We diagram in Figure 4 how this is possible.

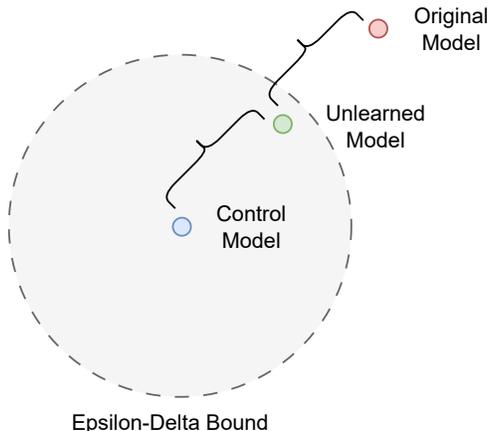


Figure 4: Intuition of how KL divergence is able to distinguish between M_u from certified removal and M_c . The distribution produced by the unlearned model is within the ϵ, δ bound but the adversary is able to leverage access to the original model to distinguish between the original model and a control model.

While M_u falls within the certified removal bound around M_c , the certified removal bound does not guarantee that M_u will be uniformly distributed within the bound. Because we assume the adversary also has knowledge of M_o under our computational unlearning threat model, we can find the model that has the least divergence with the original model. Under other notions of distance, it may also be possible to distinguish based on the *direction* of the differences—not just the magnitude of the differences.

As described in Theorem 26, conventional choices for ϵ and δ are too loose to achieve computational unlearning. They must be phrased in terms of λ , the security parameter, with $\delta \leq \text{negl}(\lambda)$ and $\epsilon \leq \ln(1 + \text{negl}(\lambda))$. This ensures that the privacy loss of every query is negligible. After a polynomial number of queries, the adversary will still have negligible information and thus will have negligible advantage.

When not set properly, the bound is loose enough to permit distinguishing as empirically demonstrated in §4.

6.1 Future work

Relaxations of our definition. As described in [13], the Fundamental Law of Information Recovery states that “overly accurate answers to too many questions will destroy privacy in a spectacular way.” For this reason, most research into differential privacy considers a bounded query model; there is at most some fixed number of queries that the adversary can make. We claim that this is impractical for unlearning. This would still require complete retraining of the model and avoiding this is the entire point of unlearning.

While we do not believe that a constant of queries is a suitable unlearning scenario, we believe there are various other relaxations that may prove useful. For example, it may be possible to let the challenger delete additional information beyond what is selected by the adversary. It is unclear if this would provide meaningful realizations of unlearning, but is more in line with [5]. Another possible relaxation could be to allow for some bounded, non-negligible adversary advantage.

Alternative relaxations could include giving the adversary less information, such as not giving them access to the original model (M_o). Our analysis in §5 depends on this information being available to the adversary. In cases where this information is not available, certified removal with a sufficiently tight bound may be a viable alternative.

Alignment of generative models via unlearning. This work focuses on image classification models to facilitate iterative experimentation but our definition naturally extends to generative models, including large language models (LLMs). In this context, unlearning can be viewed as an alignment technique. However, this presents additional challenges.

Language is a discrete sequence-based modality where changes to few tokens in the sequence can cause massive semantic changes to the meaning. Similarly, drastically different input sequences can contain the exact same information. This can make it hard to specify data points that contain the information you wish to forget. It is also unclear if certain concepts are *emergent*, meaning that the model can infer them even without explicit training data. For example, if you wish to remove all the information from a generative model about weapons, it is not as simple as forgetting all data points that contain the word “weapon.” This is further complicated because language datasets are often scraped from many different sources, meaning there are many possible sources of information that a user may wish to remove.

Accurately specifying all data points that contain the relevant information is non-trivial and an open area of research. It may be possible to use an embedding of some kind to determine semantic similarity, but the effect this has on downstream models is an open area of research.

Fine-tuning to foundation models. A common technique is to fine-tune large foundation models (e.g. LLMs, diffusion models) to a particular task. It is natural to rephrase our definitions of `learn` and `unlearn` to be fine-tunings of a foundation model. Because the adversary already has foreknowledge of the foundation model (they have the description of `learn`) the adversary is only distinguishing the results of the fine-tuning process from a control model.

7 Conclusion

In summary, we have proposed computational unlearning, a new framework for evaluating machine unlearning. Computational unlearning is satisfied by an unlearning method if the output of the unlearning method is indistinguishable from a mirror (control) model. We rigorously define indistinguishability in terms of a novel two-party cryptographic protocol which captures an adversary’s ability to distinguish between two models. Computational unlearning provides both empirical and theoretical contributions to the field of unlearning by improves upon prior evaluation methods, such as membership inference attack (MIA) scores.

We empirically showed that several machine unlearning methods from literature [10, 15, 18, 49] do not achieve computational unlearning by presenting multiple algorithms that allow an adversary to distinguish between the model produced by an unlearning method and a control model.

We have identified several theoretical implications that naturally follow from our formal definition of computational unlearning. For example, all unlearning methods that meet our definition of computational unlearning must be randomized; there are no deterministic computational unlearning methods despite there being several deterministic unlearning methods proposed in prior work. We also proved that building computational machine unlearning using differential privacy techniques leads to utility collapse.

Lastly, we outlined various directions for future work. This includes implementing high-utility general-purpose computational unlearning, potential relaxations of our computational unlearning framework, using unlearning to align generative models, and exploring how to incorporate notions of foundation models into computational unlearning.

References

- [1] Martin Abadi et al. “Deep Learning with Differential Privacy”. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS’16. ACM, 2016. DOI: 10.1145/2976749.2978318. URL: <http://dx.doi.org/10.1145/2976749.2978318>.
- [2] The Atlantic. *The Atlantic announces product and content partnership with OpenAI*. 2024. URL: <https://www.theatlantic.com/press-releases/archive/2024/05/atlantic-product-content-partnership-openai/678529/>.
- [3] D P Bertsekas. “Nonlinear Programming”. In: *Journal of the Operational Research Society* 48.3 (1997), pp. 334–334. DOI: 10.1057/palgrave.jors.2600425. eprint: <https://doi.org/10.1057/palgrave.jors.2600425>. URL: <https://doi.org/10.1057/palgrave.jors.2600425>.
- [4] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. 2023. URL: <http://toc.cryptobook.us/book.pdf>.
- [5] Lucas Bourtole et al. *Machine Unlearning*. 2020. arXiv: 1912.03817 [cs.CR].
- [6] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL]. URL: <https://arxiv.org/abs/2005.14165>.
- [7] Yinzhi Cao and Junfeng Yang. “Towards Making Systems Forget with Machine Unlearning”. In: *2015 IEEE Symposium on Security and Privacy*. 2015, pp. 463–480. DOI: 10.1109/SP.2015.35.
- [8] Centers for Medicare and Medicaid Services. *The Health Insurance Portability and Accountability Act of 1996 (HIPAA)*. 1996. URL: <http://www.cms.hhs.gov/hipaa/>.
- [9] Sungmin Cha et al. *Learning to Unlearn: Instance-wise Unlearning for Pre-trained Classifiers*. 2024. arXiv: 2301.11578 [cs.LG]. URL: <https://arxiv.org/abs/2301.11578>.
- [10] Vikram S Chundawat et al. *Can Bad Teaching Induce Forgetting? Unlearning in Deep Networks using an Incompetent Teacher*. 2023. arXiv: 2205.08096 [cs.LG]. URL: <https://arxiv.org/abs/2205.08096>.
- [11] Emilia David. *Now you can block OpenAI’s web crawler*. 2023. URL: <https://www.theverge.com/2023/8/7/23823046/openai-data-scrape-block-ai>.
- [12] Emilia David. *Vox Media and The Atlantic sign content deals with OpenAI*. 2024. URL: <https://www.theverge.com/2024/5/29/24167072/openai-content-copyright-vox-media-the-atlantic>.
- [13] Cynthia Dwork and Aaron Roth. “The Algorithmic Foundations of Differential Privacy”. In: *Found. Trends Theor. Comput. Sci.* 9.3–4 (Aug. 2014), pp. 211–407. ISSN: 1551-305X. DOI: 10.1561/04000000042. URL: <https://doi.org/10.1561/04000000042>.
- [14] European Parliament and Council of the European Union. *Regulation (EU) 2016/679 of the European Parliament and of the Council*. of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). OJ L 119, 4.5.2016, p. 1–88, May 4, 2016. URL: <https://data.europa.eu/eli/reg/2016/679/oj> (visited on 04/13/2023).

- [15] Jack Foster, Stefan Schoepf, and Alexandra Brintrup. “Fast Machine Unlearning Without Retraining Through Selective Synaptic Dampening”. In: *ArXiv abs/2308.07707* (2023). URL: <https://api.semanticscholar.org/CorpusID:260900355>.
- [16] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. “Eternal Sunshine of the Spotless Net: Selective Forgetting in Deep Networks”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), pp. 9301–9309. URL: <https://api.semanticscholar.org/CorpusID:207863297>.
- [17] Shafi Goldwasser et al. *Planting Undetectable Backdoors in Machine Learning Models*. 2022. arXiv: 2204.06974 [cs.LG].
- [18] Laura Graves, Vineel Nagisetty, and Vijay Ganesh. *Amnesiac Machine Learning*. 2020. arXiv: 2010.10981 [cs.LG]. URL: <https://arxiv.org/abs/2010.10981>.
- [19] Chuan Guo et al. *Certified Data Removal from Machine Learning Models*. 2023. arXiv: 1911.03030 [cs.LG]. URL: <https://arxiv.org/abs/1911.03030>.
- [20] Varun Gupta et al. “Adaptive Machine Unlearning”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 16319–16330. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/87f7ee4fdb57bdfd52179947211b7ebb-Paper.pdf.
- [21] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: <https://arxiv.org/abs/1512.03385>.
- [22] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: <https://arxiv.org/abs/1512.03385>.
- [23] Hongsheng Hu et al. *A Duty to Forget, a Right to be Assured? Exposing Vulnerabilities in Machine Unlearning Services*. 2024. arXiv: 2309.08230 [cs.CR]. URL: <https://arxiv.org/abs/2309.08230>.
- [24] Hongsheng Hu et al. *Learn What You Want to Unlearn: Unlearning Inversion Attacks against Machine Unlearning*. 2024. arXiv: 2404.03233 [cs.CR]. URL: <https://arxiv.org/abs/2404.03233>.
- [25] Kate Knibbs. *Condé Nast Signs Deal With OpenAI*. 2024. URL: <https://www.wired.com/story/conde-nast-openai-deal/>.
- [26] Sangamesh Kodge, Gobinda Saha, and Kaushik Roy. “Deep Unlearning: Fast and Efficient Gradient-free Class Forgetting”. In: *Trans. Mach. Learn. Res.* 2024 (2023). URL: <https://api.semanticscholar.org/CorpusID:271596242>.
- [27] Alex Krizhevsky. “Learning Multiple Layers of Features from Tiny Images”. In: *Learning Multiple Layers of Features from Tiny Images*. 2009. URL: <https://api.semanticscholar.org/CorpusID:18268744>.
- [28] S. Kullback and R. A. Leibler. “On Information and Sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86. DOI: 10.1214/aoms/1177729694. URL: <https://doi.org/10.1214/aoms/1177729694>.
- [29] Dhruv Mehrotra and Andrew Couts. *Amazon Is Investigating Perplexity Over Claims of Scraping Abuse*. 2024. URL: <https://www.wired.com/story/aws-perplexity-bot-scraping-investigation/>.
- [30] Ronak R. Mehta et al. “Deep Unlearning via Randomized Conditionally Independent Hessians”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), pp. 10412–10421. URL: <https://api.semanticscholar.org/CorpusID:248227997>.
- [31] Thanh Tam Nguyen et al. *A Survey of Machine Unlearning*. 2022. arXiv: 2209.02299 [cs.LG].
- [32] OpenAI. *A content and product partnership with The Atlantic*. 2024. URL: <https://openai.com/index/enhancing-news-in-chatgpt-with-the-atlantic/>.
- [33] OpenAI. *A Content and Product Partnership with Vox Media*. 2023. URL: <https://openai.com/index/a-content-and-product-partnership-with-vox-media/>.

- [34] OpenAI. *OpenAI partners with Condé Nast*. 2024. URL: <https://openai.com/index/conde-nast/>.
- [35] OpenAI. *Partnership with Axel Springer to deepen beneficial use of AI in journalism*. 2023. URL: <https://openai.com/index/axel-springer-partnership/>.
- [36] Christoph Schuhmann et al. *LAION-5B: An open large-scale dataset for training next generation image-text models*. 2022. eprint: 2210.08402.
- [37] Ayush Sekhari et al. *Remember What You Want to Forget: Algorithms for Machine Unlearning*. 2021. arXiv: 2103.03279 [cs.LG].
- [38] Reza Shokri et al. *Membership Inference Attacks against Machine Learning Models*. 2017. arXiv: 1610.05820 [cs.CR].
- [39] Axel Springer. *Axel Springer and OpenAI partner to deepen beneficial use of AI in journalism*. 2023. URL: <https://www.axelspringer.com/en/ax-press-release/axel-springer-and-openai-partner-to-deepen-beneficial-use-of-ai-in-journalism>.
- [40] Ayush K Tarun et al. “Fast Yet Effective Machine Unlearning”. In: *IEEE Transactions on Neural Networks and Learning Systems* 35 (2021), pp. 13046–13055. URL: <https://api.semanticscholar.org/CorpusID:244270535>.
- [41] David Thiel. *Identifying and Eliminating CSAM in Generative ML Training Data and Models*. 2023. URL: <https://doi.org/10.25740/kh752sm9123>.
- [42] Anvith Thudi et al. “On the Necessity of Auditable Algorithmic Definitions for Machine Unlearning”. In: *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 4007–4022. ISBN: 978-1-939133-31-1. URL: <https://www.usenix.org/conference/usenixsecurity22/presentation/thudi>.
- [43] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971.
- [44] Enayat Ullah et al. *Machine Unlearning via Algorithmic Stability*. 2021. arXiv: 2102.13179 [cs.LG]. URL: <https://arxiv.org/abs/2102.13179>.
- [45] United States Congress. *Digital Millennium Copyright Act of 1998 (DMCA)*. 1996. URL: <https://www.govinfo.gov/content/pkg/PLAW-105publ304/pdf/PLAW-105publ304.pdf>.
- [46] United States Congress. *Fair Credit Reporting Act of 1970 (FCRA)*. 1970. URL: <https://www.govinfo.gov/content/pkg/STATUTE-84/pdf/STATUTE-84-Pg1114-2.pdf>.
- [47] Alexander Warnecke et al. *Machine Unlearning of Features and Labels*. 2023. arXiv: 2108.11577 [cs.LG]. URL: <https://arxiv.org/abs/2108.11577>.
- [48] Jess Weatherbed. *Anthropic’s crawler is ignoring websites’ anti-AI scraping policies*. 2024. URL: <https://www.theverge.com/2024/7/25/24205943/anthropic-ai-web-crawler-claudebot-ifixit-scraping-training-data>.
- [49] Binchi Zhang et al. *Towards Certified Unlearning for Deep Neural Networks*. 2024. arXiv: 2408.00920 [cs.LG]. URL: <https://arxiv.org/abs/2408.00920>.