Rewind-to-Delete: Certified Machine Unlearning for Nonconvex Functions

Siqiao Mu [1] Diego Klabjan [2]

Abstract

Machine unlearning algorithms aim to efficiently remove data from a model without retraining it from scratch, in order to remove corrupted or outdated data or respect a user's "right to be forgotten." Certified machine unlearning is a strong theoretical guarantee based on differential privacy that quantifies the extent to which an algorithm erases data from the model weights. In contrast to existing works in certified unlearning for convex or strongly convex loss functions, or nonconvex objectives with limiting assumptions, we propose the first, first-order, black-box (i.e., can be applied to models pretrained with vanilla gradient descent) algorithm for unlearning on general nonconvex loss functions, which unlearns by "rewinding" to an earlier step during the learning process before performing gradient descent on the loss function of the retained data points. We prove (ϵ, δ) certified unlearning and performance guarantees that establish the privacy-utility-complexity tradeoff of our algorithm, and we prove generalization guarantees for nonconvex functions that satisfy the Polyak-Lojasiewicz inequality. Finally, we implement our algorithm under a new experimental framework that more accurately reflects real-world use cases for preserving user privacy.

1. Introduction

Machine unlearning, or data deletion from models, refers to the problem of removing the influence of some data from a trained model without the computational expenses of completely retraining it from scratch (Cao & Yang, 2015). This research direction has become highly relevant in the last few years, due to increasing concern about user privacy and data security as well as the growing cost of retraining massive deep learning models on constantly updated datasets. For example, recent legislation that protects a user's "Right to be Forgotten," including the European Union's General Data Protection Regulation (GDPR), the California Consumer Privacy Act (CCPA), and the Canadian Consumer Privacy Protection Act (CPPA), mandate that users be allowed to request removal of their personal data, which may be stored in databases or memorized by models (Sekhari et al., 2021). In addition, machine unlearning has practical implications for removing the influence of corrupted, outdated, or mislabeled data (Kurmanji et al., 2023; Nguyen et al., 2024).

The typical goal of machine unlearning algorithms is to yield a model that resembles the model obtained from a full retrain on the updated dataset after data is removed. This requirement is formalized in the concept of "certified unlearning," a strong theoretical guarantee motivated by differential privacy that probabilistically bounds the difference between the model weights returned by the unlearning and retraining algorithms (Guo et al., 2019). However, algorithms satisfying certified unlearning need to also be practical. For example, a trivial unlearning algorithm retrains the model from scratch on the retained dataset, but this would provide no efficiency gain. On the other hand, unlearning is also provably satisfied if the learning and unlearning algorithm both output weights randomly sampled from the same Gaussian distribution, but this would yield a poorly performing model. Therefore, an ideal algorithm optimally balances data deletion, model accuracy, and computation, also known as the "privacy-utility-efficiency" tradeoff (Liu et al., 2024). Moreover, because training from scratch is computationally expensive in the machine unlearning setting, we desire *black-box unlearning algorithms*, which can be applied to pretrained models and do not require training with the intention of unlearning later.

Most certified unlearning algorithms are designed for convex or strongly convex functions (Guo et al., 2019; Neel et al., 2021; Sekhari et al., 2021). Relaxing the convexity requirement is challenging since nonconvex functions do not have unique global minima. Recently, there have been several works that provide certified unlearning guarantees for nonconvex functions. For example, (Zhang et al., 2024) proposes a single-step Newton update algorithm with convex regularization followed by Gaussian perturbation, inspired by existing second-order unlearning methods such as (Guo et al., 2019; Sekhari et al., 2021). Similarly, (Qiao et al., 2024) proposes a quasi-second-order method that exploits Hessian vector products to avoid directly computing the

[1]Department of Engineering Sciences and Applied Mathematics, Northwestern University, Evanston, IL, USA [2]Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, USA. Correspondence to: Siqiao Mu <siqiaomu2026@u.northwestern.edu>.

Hessian. Finally, (Chien et al., 2024a) proposes a first-order method in the form of projected noisy gradient descent with Gaussian noise added at every step.

Prior work has focused on achieving theoretical unlearning guarantees in the nonconvex setting; however, practical unlearning algorithms should also be computationally efficient and convenient to use. First, (Zhang et al., 2024) and (Qiao et al., 2024) are both (quasi) second-order methods, but first-order methods that only require computing the gradient are more computationally efficient and require less storage. Second, (Chien et al., 2024a; Qiao et al., 2024) are not black-box, since during the training process, (Chien et al., 2024a) requires injecting Gaussian noise at every step and (Qiao et al., 2024) requires recording a statistical vector for each data sample at each time step. These "white-box" algorithms require significant changes to standard learning algorithms, which hinders easy implementation. Table 1 summarizes these results and Appendix C provides a more thorough comparison with prior work.

In this work, we introduce "rewind-to-delete" (R2D), a first-order, black-box, certified unlearning algorithm for general nonconvex functions. Our learning algorithm consists of vanilla gradient descent steps on the loss function of the dataset followed by Gaussian perturbation. To remove some data from the model, our unlearning algorithm "rewinds" to a model checkpoint from earlier in the original training trajectory, performs additional gradient descent steps on the *new* loss function, and again adds Gaussian perturbation to the weights. The checkpoint can be saved during the training period (which is standard practice), or it can be computed post hoc from a pretrained model via the proximal point method. We prove (ϵ, δ) certified unlearning for our algorithm and provide theoretical guarantees that explicitly address the "privacy-utility-efficiency" tradeoff of our unlearning algorithm. Our algorithm is easy to implement and relies on simple assumptions. In addition, the algorithm is black-box, as it can be applied to a pretrained model as long as the model was trained with vanilla gradient descent.

We also analyze the case of nonconvex functions that satisfy the Polyak-Łojasiewicz (PL) inequality. The PL inequality is a weak condition that guarantees the existence of a connected basin of global minima, to which gradient descent converges at a linear rate (Karimi et al., 2016). This property allows us to derive empirical risk convergence and generalization guarantees, despite nonconvexity. PL functions are highly relevant to deep learning because overparameterized neural networks locally satisfy the PL condition in neighborhoods around initialization (Liu et al., 2022).

Finally, we empirically demonstrate the privacy-utility-efficiency tradeoff of our algorithm and its superior performance compared to existing certified unlearning algorithms for nonconvex functions. We use real-world medical or facial data, with each dataset derived from a collection of users. In contrast to most unlearning experiments, which either select unlearned samples uniformly at random or from a specific class, we train a neural network model to learn some global characteristics about a dataset, and unlearn data from a subset of the users, thereby simulating realistic unlearning requests and their impact on model utility. This results in an unlearned dataset that is not i.i.d. to the training or test set. To assess unlearning empirically, we conduct a membership inference attack (MIA) (Shokri et al., 2016) comparing the unlearned model output on the unlearned dataset and an *out-of-distribution* dataset, constructed separately from data and *users* not present in the training data. Comparing with baseline methods, on the medical dataset we outperform (Zhang et al., 2024) by 9-24 accuracy percentage points using 41%-80% rewinding, and on the facial dataset we outperform (Zhang et al., 2024) by 23-36 accuracy percentage points and 3%-5% on unlearning, using 51%-75% rewinding, and we are comparable in compute time.[6] Compared to the white-box algorithm (Qiao et al., 2024), we are competitive in accuracy and unlearning and 70-104 times faster during training, despite using more steps. Ultimately, our first-order method enjoys the benefits of minimal computational requirements, easy off-the-shelf implementation, and competitive performance.

Our contributions are as follows.

- We develop the first (ϵ, δ) certified unlearning algorithm for nonconvex loss functions that is first-order and black-box.
- We prove theoretical unlearning guarantees that demon-

Table 1. Comparison of certified unlearning algorithms for convex and nonconvex functions. d is the dimension of the model parameters and n is the size of the training dataset.

ALGORITHM	LOSS FUNCTION	METHOD	STORAGE	BLACK-BOX?
NEWTON STEP[1]	STRONGLY CONVEX	SECOND-ORDER	$O(d^2)$	$\times$
DESCENT-TO-DELETE [2]	STRONGLY CONVEX	FIRST-ORDER	$O(d)$	$\surd$
LANGEVIN UNLEARNING[3]	NONCONVEX	FIRST-ORDER	$O(d)$	$\times$
CONSTRAINED NEWTON STEP[4]	NONCONVEX	SECOND-ORDER	$O(d^2)$	$\surd$
HESSIAN-FREE UNLEARNING [5]	NONCONVEX	QUASI-SECOND-ORDER	$O(nd)$	$\times$
OUR WORK (R2D)	NONCONVEX	FIRST-ORDER	$O(d)$	$\surd$

[1] (Guo et al., 2019) [2] (Neel et al., 2021) [3] (Chien et al., 2024a) [4] (Zhang et al., 2024) [5] (Qiao et al., 2024)

[6] See Table 8 and 9 in the Appendix for numerical details.

strate the privacy-utility-efficiency tradeoff of our algorithm, allowing for controllable noise at the expense of privacy or computation. For the special case of PL loss functions, we obtain linear convergence and generalization guarantees.

- We conduct experiments demonstrating the efficacy and utility of our algorithm under a novel experimental framework that better reflects real-world unlearning scenarios for protecting user privacy.

1.1. Related Work

Differential privacy. Differential privacy (DP) (Dwork et al., 2006) is a well-established framework designed to protect individual privacy by ensuring that the inclusion or exclusion of any single data point in a dataset does not significantly affect the output of data analysis or modeling, limiting information leakage about any individual within the dataset. Specifically, a differentially private learning algorithm yields a model trained on some dataset that is probabilistically indistinguishable from a model trained on the same dataset with a data sample removed or replaced. The concept of (ϵ, δ)-privacy quantifies the strength of this privacy guarantee in terms of the privacy loss, ϵ, and the probability of a privacy breach, δ (Dwork & Roth, 2014). This privacy can be injected during or after training by adding controlled noise to the data, model weights (Wu et al., 2017; Zhang et al., 2017), gradients (Abadi et al., 2016; Zhang et al., 2017), or objective function (Chaudhuri et al., 2011), in order to mask information about any one sample in the dataset. However, greater noise typically corresponds to worse model performance, leading to a trade-off between utility and privacy. The theory and techniques of differential privacy provide a natural starting point for the rigorous analysis of unlearning algorithms.

As observed in (Wu et al., 2017), "white-box" differentially private algorithms, which require code changes to inject noise at every training step, are challenging to deploy in the real world due to additional development and runtime overhead incurred from implementing a nonstandard learning algorithm. Rather than adding noise at each iteration, (Wu et al., 2017) and (Zhang et al., 2017) propose DP algorithms that only perturb the output after training, an approach which is easier to integrate into real-world development pipelines. In similar fashion, our proposed black-box unlearning algorithm can be implemented without any special steps during learning with gradient descent. We also do not inject noise at each iteration, only perturbing at the end of training. The difference between our approach and (Wu et al., 2017; Zhang et al., 2017), however, is that our approach can accommodate the nonconvex case, leveraging model checkpointing to control the distance from the retraining trajectory.

Certified unlearning. The term "machine unlearning" was first coined by (Cao & Yang, 2015) to describe a deterministic data deletion algorithm, which has limited application to general optimization problems. In the following years, techniques have been developed for "exact unlearning," which exactly removes the influence of data, and "approximate unlearning," which yields a model that is approximately close to the retrained model with some determined precision (Xu et al., 2023). Our work focuses on the latter. Both (Ginart et al., 2019) and (Guo et al., 2019) introduce a probabilistic notion of approximate unlearning, where the unlearning and retraining outputs must be close in distribution. Inspired by differential privacy, (Guo et al., 2019) introduces the definition of certified (ϵ, δ) unlearning used in this work. Like DP algorithms, certified unlearning algorithms typically require injected noise to the weights or objective function, which can degrade model performance. Moreover, unlearning algorithms are also designed to reduce computation, leading to a three-way trade-off between privacy, utility, and complexity.

Certified unlearning has been studied for a variety of settings, including linear and logistic models (Guo et al., 2019; Izzo et al., 2021), graph neural networks (Chien et al., 2022), minimax models (Liu et al., 2023), and the federated learning setting (Fraboni et al., 2024), as well as convex models (Sekhari et al., 2021; Neel et al., 2021; Suriyakumar & Wilson, 2022; Chien et al., 2024b) and nonconvex models (Chien et al., 2024a; Qiao et al., 2024; Zhang et al., 2024). These algorithms can be categorized as first-order methods that only require access to the function gradients (Neel et al., 2021; Chien et al., 2024b) or second-order methods that leverage information from the Hessian to approximate the model weights that would result from retraining (Sekhari et al., 2021; Suriyakumar & Wilson, 2022; Zhang et al., 2024; Qiao et al., 2024). Our work is inspired by the "descent-to-delete" algorithm (Neel et al., 2021), a first-order unlearning algorithm for strongly convex functions that unlearns by fine-tuning with gradient descent iterates on the loss function of the retained samples.

Nonconvex unlearning. There are also many approximate unlearning algorithms for nonconvex functions that rely on heuristics or weaker theoretical guarantees. For example, (Bui et al., 2024) proposes a "weak unlearning" algorithm, which considers indistinguishability with respect to model output space instead of model parameter space. This is a weaker guarantee because the model weights may still retain information about the unlearned data and be susceptible to membership inference attacks. Another popular algorithm for neural networks is SCRUB (Kurmanji et al., 2023), a gradient-based algorithm that balances maximizing error on the unlearned data and maintaining performance on the retained data. An extension of SCRUB, SCRUB+Rewind, "rewinds" the algorithm to a point where the error on the unlearned data is "just high enough," so as to impede membership inference attacks. Finally, although (Golatkar et al., 2019; 2020) propose unlearning algorithms for deep neural networks, they only provide a general upper bound on the amount of information retained in the weights rather than a strict certified unlearning guarantee.

Additional approaches include subtracting out the impact of the unlearned data in each batch of gradient descent (Graves et al., 2021), gradient ascent on the loss function of the unlearned data (Jang et al., 2023), and retraining the last layers of the neural network on the retained data (Goel et al., 2022). Ultimately, while the ideas of checkpointing, gradient ascent, and "rewinding" have been considered in other machine unlearning works, our algorithm combines these elements in a novel manner to obtain strong theoretical guarantees that prior algorithms lack.

2. Algorithm

Let $\mathcal{D} = \{z_1, ..., z_n\}$ be a training dataset of n data points drawn independently and identically distributed from the sample space $\mathcal{Z}$, and let Θ be the model parameter space. Let $A : \mathcal{Z}^n \to \Theta$ be a (randomized) learning algorithm that trains on $\mathcal{D}$ and outputs a model with weight parameters $\theta \in \Theta$, where Θ is the (potentially infinite) space of model weights. Typically, the goal of a learning algorithm is to minimize $f_{\mathcal{D}}(\theta)$, the empirical loss on $\mathcal{D}$, defined as follows

$$f_{\mathcal{D}}(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_{z_i}(\theta),$$

where $f_{z_i}(\theta)$ represents the loss on the sample z_i.

Let us "unlearn" or remove the influence of a subset of data $Z \subset \mathcal{D}$ from the output of the learning algorithm $A(\mathcal{D})$. Let $\mathcal{D}' = \mathcal{D}\backslash Z$, and we denote by $U(A(\mathcal{D}), \mathcal{D}, Z)$ the output of an unlearning algorithm U. The goal of the unlearning algorithm is to output a model parameter that is probabilistically *indistinguishable* from the output of $A(\mathcal{D}')$. This is formalized in the concept of (ϵ, δ)-indistinguishability, which is used in the DP literature to characterize the influence of a data point on the model output (Dwork & Roth, 2014).

Definition 2.1. (Dwork & Roth, 2014; Neel et al., 2021) Let X and Y be random variables over some domain Ω. We say X and Y are (ϵ, δ)-indistinguishable if for all $S \subseteq \Omega$,

$$\mathbb{P}[X \in S] \leq e^{\epsilon}\mathbb{P}[Y \in S] + \delta,$$
$$\mathbb{P}[Y \in S] \leq e^{\epsilon}\mathbb{P}[X \in S] + \delta.$$

In the context of differential privacy, X and Y are the learning algorithm outputs on neighboring datasets that differ in a single sample. ϵ is the privacy loss or budget, which can be interpreted as a limit on the amount of information about an individual that can be extracted from the model, whereas δ accounts for the probability that these privacy guarantees might be violated. Definition 2.1 extends naturally to the following definition of (ϵ, δ) certified unlearning.

Definition 2.2. Let Z denote a subset of the training dataset which we would like to unlearn. Then U is an (ϵ, δ) certified unlearning algorithm for A if for all such Z, $U(A(\mathcal{D}), \mathcal{D}, Z)$ and $A(\mathcal{D}\backslash Z)$ are (ϵ, δ)-indistinguishable.

Next, we describe our algorithms for machine unlearning. The learning algorithm A (Algorithm 1) performs gradient descent updates on $f_{\mathcal{D}}$ for T iterations, and the iterate at the $T - K$ time step is saved as a checkpoint or computed post hoc via the proximal point algorithm (Algorithm 3). Then Gaussian noise is added to the final parameter θ_T, and the perturbed parameter is used for model inference. When a request is received to remove the data subset Z, the checkpointed model parameter θ_{T-K} is loaded as the initial point of the unlearning algorithm U (Algorithm 2). Then we perform K gradient descent steps on the *new* loss function $f_{\mathcal{D}'}$ and add Gaussian noise again to the final parameter, using the perturbed weights for future model inference.

Algorithm 1 A: Rewind-To-Descent (R2D) Learning Algorithm

```
Require: dataset $\mathcal{D}$, initial point $\theta_0 \in \Theta$
  for t = 1, 2, ..., T do
    $\theta_t = \theta_{t-1} - \eta\nabla f_{\mathcal{D}}(\theta_{t-1})$ {Gradient descent steps}
  end for
  Save checkpoint $\theta_{T-K}$ or compute $\theta_{T-K}$ via Algorithm 3
  Sample $\xi \sim \mathcal{N}(0, \sigma^2\mathbb{I}_d)$
  $\tilde{\theta} = \theta_T + \xi$          {Gaussian perturbation}
  Use $\tilde{\theta}$ for model inference
  Upon receiving an unlearning request, call Algorithm 2
```

Algorithm 2 U: R2D Unlearning Algorithm

```
Require: dataset $\mathcal{D}'$, model checkpoint $\theta_{T-K}$
  $\theta''_0 = \theta_{T-K}$
  for t = 1, ..., K do
    $\theta''_t = \theta''_{t-1} - \eta\nabla f_{\mathcal{D}'}(\theta''_{t-1})$ {Gradient descent steps}
  end for
  Sample $\xi \sim \mathcal{N}(0, \sigma^2\mathbb{I}_d)$
  $\tilde{\theta}'' = \theta''_K + \xi$          {Gaussian perturbation}
  Use $\tilde{\theta}''$ for model inference
```

When a model is trained without a checkpoint saved, we can still obtain a black-box unlearning algorithm by carefully "rewinding" the gradient descent training steps via the proximal point method, outlined in Algorithm 3. We can solve for previous gradient descent iterates through the following implicit equation (2):

$$\theta_{t+1} = \theta_t - \eta\nabla f_{\mathcal{D}}(\theta_t) \quad (1)$$
$$\theta_t = \theta_{t+1} + \eta\nabla f_{\mathcal{D}}(\theta_t). \quad (2)$$

The backward Euler update in (2) is distinct from a standard gradient ascent step, which is a forward Euler method. Instead, we compute θ_t from θ_{t+1} by taking advantage of the connection between backward Euler for gradient flow and the proximal point method, an iterative algorithm for minimizing a convex function (Martinet, 1970). Let $g(\theta)$ denote a convex function. The proximal point method minimizes $g(\theta)$ by taking the proximal operator with parameter γ of the previous iterate, defined as follows

$$\theta_{k+1} = prox_{g,\gamma}(\theta_k) = \arg\min_x \{g(x) + \frac{1}{2\gamma}||x - \theta_k||^2\}.$$

Although for our problem, f is nonconvex, adding sufficient regularization produces a convex and globally tractable proximal point subproblem, stated in Lemma A.1. Therefore, by computing the proximal operator with respect to $-f(\theta)$, we can solve the implicit gradient ascent step.

Lemma 2.3. *Suppose $f(\theta)$ is continuously differentiable, θ_t is defined as in (2), and let $\eta < \frac{1}{L}$. Then $\theta_t = prox_{-f,\eta}(\theta_{t+1})$.*

The proof of Lemma 2.3 is provided in Appendix A.

Algorithm 3 Compute Checkpoint via Proximal Point Method

```
Require: datasets $\mathcal{D}$, model checkpoint $\theta_T$
  $\theta''_0 = \theta_T$
  for t = 1, ..., K do
    $\theta''_t = \arg\min_x\{-f_{\mathcal{D}}(x) + \frac{1}{2\eta}||x - \theta''_{t-1}||^2\}$
  end for
  Return $\theta''_K$
```

If $\eta < \frac{1}{L}$, then due to convexity, we can solve the proximal point subproblem easily via gradient descent or Newton's method. Computationally, when $K \ll T$, the algorithm is comparable to other second-order unlearning methods that require a single Newton step. In addition, we only need to compute the model checkpoint once prior to unlearning requests, so this computation can be considered "offline" (Izzo et al., 2021; Qiao et al., 2024).

3. Analyses

In the following theorem, we establish the unlearning and performance guarantees for our algorithm on nonconvex functions. For nonconvex functions, gradient descent might converge to local minima or saddle points, so we measure the performance by the average of the gradient norm over iterates, a common DP performance metric for algorithms on nonconvex functions.

Theorem 3.1. *Let ϵ, δ be fixed such that $0 < \epsilon \leq 1$ and $\delta > 0$. Suppose for all $z \in \mathcal{Z}$, the loss function f_z is L-Lipschitz smooth and the gradient is uniformly bounded by some constant G so that $\|\nabla f_z(\theta)\| < G$ for all $\theta \in \Theta$. Let $\mathcal{D}$ denote the original dataset of size n, let $Z \subset \mathcal{D}$ denote the unlearned dataset of size m, and let $\mathcal{D}' = \mathcal{D}\backslash Z$ denote the retained dataset. Let the learning algorithm A be initialized at θ_0 and run for T iterations with step size $\eta \leq \min\{\frac{1}{L}, \frac{n}{2(n-m)L}\}$. Let the standard deviation σ of the Gaussian noise be defined as*

$$\sigma = \frac{2mG \cdot h(K)\sqrt{2\log(1.25/\delta)}}{Ln\epsilon}, \tag{3}$$

where $h(K)$ is a function that monotonically decreases to zero as K increases from 0 to T defined by

$$h(K) = ((1 + \frac{\eta Ln}{n-m})^{T-K} - 1)(1 + \eta L)^K.$$

Then U is an (ϵ, δ)-unlearning algorithm for A with noise σ. In addition,

$$\frac{\sum_{t=0}^{T-K-1}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 + \sum_{t=0}^{K-1}||\nabla f_{\mathcal{D}'}(\theta''_t)||^2}{T} + \frac{\mathbb{E}[||\nabla f_{\mathcal{D}'}(\tilde{\theta}'')||^2]}{T} \leq O(\frac{n}{T(n-m)}) + O(\frac{(T-K-1)m}{T(n-m)}) \tag{4}$$

where the expectation is taken with respect to the Gaussian noise added at the end of U, and where $O(\cdot)$ hides dependencies on η, G, and L.

Corollary 3.2. *For fixed σ and δ, the dependence of K on ϵ in (3), denoted as $K(\epsilon)$, is upper-bounded as follows:*

$$K(\epsilon) \leq \frac{\log\left((1 + \frac{\eta Ln}{n-m})^T - \frac{\sigma Ln\epsilon}{2mG\sqrt{2\log(1.25/\delta)}}\right)}{\log(1 + \frac{\eta Ln}{n-m})}. \tag{5}$$

Equation (4) states that the average of the gradient norm squared of the initial $T - K$ learning iterates, the $K - 1$ unlearning iterates after, and the last perturbed iterate $\tilde{\theta}''$ decreases with increasing T and n, indicating that the algorithm converges to a stationary point with small gradient norm. Corollary 3.2 provides an upper bound on $K(\epsilon)$, such that in practice we can choose K equal to this bound to ensure the privacy guarantee is achieved. We show in Figure 1c that this bound is close to tight for $0 < \epsilon \leq 1$ and real-world parameters.

The proof of Theorem 3.1 is provided in Appendix B. The proof of Corollary 3.2 is in Appendix B.3. The analysis relies on carefully tracking the distance between the unlearning iterates and gradient descent iterates on $f_{\mathcal{D}'}$. Like prior work in certified unlearning, our analysis relies on the Gaussian mechanism for differential privacy (Dwork & Roth, 2014), which implies that as long as the distance between the trajectories is bounded, we can add a sufficient amount of Gaussian noise to make the algorithm outputs (ϵ, δ)-indistinguishable. We therefore can compute the noise level σ required to achieve unlearning. For the utility guarantees, we leverage the fact that gradient descent steps on $f_{\mathcal{D}}$ also make progress on $f_{\mathcal{D}'}$ to obtain bounds that only depend on problem parameters and the initialization θ_0.

Theorem 3.1 underscores the "privacy-utility-complexity" tradeoff between our measure of unlearning, ϵ, noise, σ, and the number of unlearning iterations, K. By construction, $K < T$, so our algorithm is more efficient than retraining. We can pick larger K such that the noise required is arbitrarily small at the expense of computation, and when $K = T$ our algorithm amounts to a noiseless full retrain. Moreover, the standard deviation σ inversely scales with the size of the dataset n, implying that unlearning on larger datasets require less noise. In contrast, (Zhang et al., 2024) and (Qiao et al., 2024) do not feature such data-dependent guarantees.

Remark 3.3. Theorem 3.1 applies to unlearning a batch of m data samples. Our algorithm also accommodates sequential unlearning requests. If, after unlearning m points, an additional k unlearning requests arrive, we simply call the unlearning algorithm M on the new retained dataset of size $n - m - k$. Notably, if the total number of unlearned data increases while σ and K stay constant, our unlearning guarantee worsens, which is consistent with other results (Guo et al., 2019; Zhang et al., 2024; Chien et al., 2024a).

We can obtain faster convergence if we consider Polyak-Lojasiewicz (PL) functions, which are nonconvex functions that satisfy the following PL inequality.

Definition 3.4. (Karimi et al., 2016) For some function f, suppose it attains a global minimum value f^*. Then f satisfies the PL inequality if for some $\mu > 0$ and all x,

$$\frac{1}{2}||f(x)||^2 \geq \mu(f(x) - f^*). \tag{6}$$

Although PL functions may be nonconvex, they have a continuous basin of global minima, so we can obtain empirical risk bounds. Corollary 3.5 states that the empirical risk converges linearly with both T and K. The proof is provided in Appendix B.1.

Corollary 3.5. *Suppose the conditions of Theorem 3.1 hold and in addition, $f_{\mathcal{D}'}$ satisfies the PL condition with parameter μ. Let $f^*_{\mathcal{D}'}$ represent the global optimal value of $f_{\mathcal{D}'}$. Then*

$$\mathbb{E}[f_{\mathcal{D}'}(\tilde{\theta}'')] - f^*_{\mathcal{D}'} \leq L\sqrt{d}\sigma + (1-\eta\mu)^K \frac{G^2m + L\eta Gm}{\mu(n-m)} + (1 - \frac{\eta\mu(n-m)}{n})^{T-K}(1-\eta\mu)^K(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'})$$

where σ is defined in (3) and the expectation is taken with respect to the Gaussian noise added at the end of U.

Our performance guarantees demonstrate that more learning iterates T correspond to better performance upon unlearning. For PL functions, the utility converges faster with increasing T than for the general nonconvex case.

Although practical algorithms typically minimize the empirical risk, the ultimate goal of learning is to minimize the population risk, given by

$$F(\theta) = \mathbb{E}_{z\sim\mathcal{Z}}[f_z(\theta)], \tag{7}$$

in order to determine how well the model will generalize on unseen test data. We leverage results from (Lei & Ying, 2021) that relate the on-average stability bounds of algorithms on PL functions to their excess population risk.

Corollary 3.6. *Suppose the conditions of Theorem 3.1 hold and in addition, $f_{\mathcal{D}'}$ satisfies the PL condition with parameter μ. Let F^* represent the global optimal value of F, defined in (7). Then*

$$\mathbb{E}[F(\tilde{\theta}'')] - F^* \leq L\sqrt{d}\sigma + \frac{2G^2}{(n-m)\mu} + \frac{L}{2\mu}(1 - \frac{\eta\mu(n-m)}{n})^{T-K}(1-\eta\mu)^K(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + \frac{L}{2\mu}(1-\eta\mu)^K\frac{G^2m + L\eta Gm}{\mu(n-m)}$$

where σ is defined in (3) and the expectation is taken with respect to i.i.d. sampling of $\mathcal{D} \sim \mathcal{Z}$ and the Gaussian noise added at the end of U.

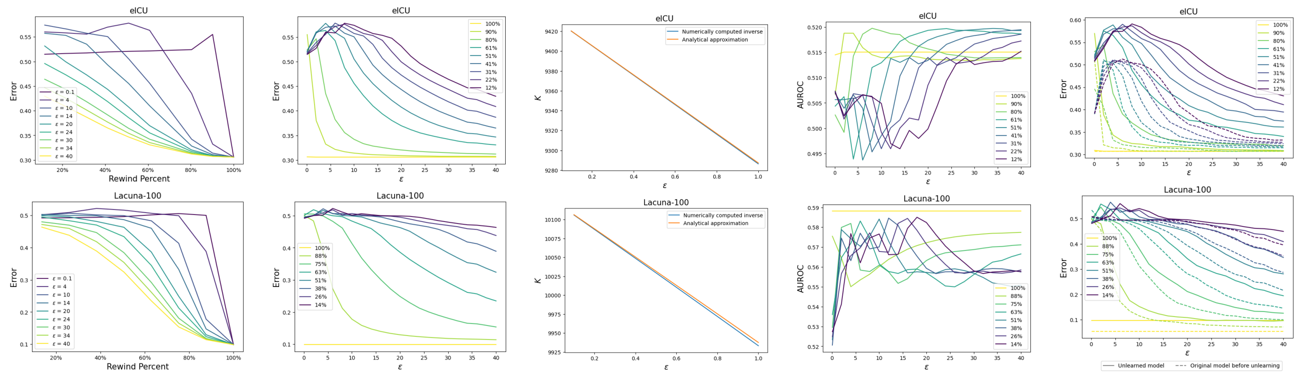
4. Experiments

For more experiment details, including choice of hyperparameters, baseline implementations, hardware, and numerical results, see Appendix D.

4.1. Experimental Framework

We test our algorithm (Algorithms 1 and 2) in experimental settings that reflect real-world use cases of unlearning for protecting user privacy. For all experiments, we train a binary classifier with the cross-entropy loss function over a dataset that is naturally split among many different users. To test unlearning, we remove the data associated with a subset of the users (1%-2% of the data) and observe the impact on the original binary classification task. The unlearned dataset is therefore not i.i.d. to the training or test dataset. Our experimental framework better reflects unlearning in practice, where users may request the removal of their data but they do not each represent a class in the model. This stands in contrast to existing unlearning experiments that unlearn data from a selected class (Guo et al., 2019; Golatkar et al., 2019; Kurmanji et al., 2023), or randomly select samples uniformly from the dataset to unlearn (Zhang et al., 2024; Qiao et al., 2024). [7]

[7] Code is open-sourced at the following anonymized link: https://github.com/anonymous-1234567/r2d.



(a) Test error vs. rewind percent.

(b) Test error vs. ϵ.

(c) K vs. ϵ.

(d) MIA score vs. ϵ.

(e) Error on unlearned data after unlearning.

Figure 1. Privacy-utility-complexity tradeoff of R2D. The top row shows the experiments on the eICU dataset, and the bottom row shows the experiments on the Lacuna-100 dataset. Figures 1a, 1b, 1c demonstrate the tradeoff between accuracy and computation/unlearning. Figures 1d, 1e evaluate unlearning as measured by the MIA score and the error on the unlearned data before and after unlearning. The unlearning computations are represented by the rewind percent, or $\frac{K}{T} \times 100\%$.

4.2. Implementation

We make some approximations of our theory to implement the algorithm in practice. Instead of full-batch gradient descent, we use stochastic gradient descent with a large batch size and sampling without replacement (i.e. shuffling). In addition, while the noise equation (3) assumes constant step size, we use an exponentially decaying step size that allows training to make sufficient progress early on, taking the final step size as η. Moreover, for $\epsilon > 1$, we utilize the bound derived in (Balle & Wang, 2018) to calibrate the noise. For unlearning, we start the step size at the value at the checkpoint. Finally, we estimate the values L and G using sampling approaches outlined in Appendix D. An alternate approach would be to set these parameters as "tunable hyperparameters," which, as noted in (Zhang et al., 2024) may lead to more imprecise certification guarantees. Future directions of this research include developing theory to incorporate these approximations.

4.3. Datasets

We consider small-scale and large-scale nonconvex settings, performing binary classification on datasets that contain medical or facial information about a set of users.

For the small-scale experiments, we train a multilayer perceptron (MLP) to perform classification on the eICU dataset, a large multi-center intensive care unit (ICU) database consisting of tabular data on ICU admissions (Pollard et al., 2018). Each patient is linked with 1-24 hospital stays. We predict if the length of a hospital stay of a patient is longer or shorter than a week using the intake variables of the Acute Physiology Age Chronic Health Evaluation (APACHE) predictive framework, including blood pressure, body temperature, and age. For unlearning, we unlearn a random subset of patients and their corresponding data.

For the large-scale experiments, we consider the VGGFace2 dataset, which is a dataset of approximately $9,000$ celebrities and their face images from the internet (Cao et al., 2018). We apply the MAAD-Face annotations from (Terhörst et al., 2020) to label each celebrity as male or female, and sample a *class balanced* dataset of 100 celebrities to form the Lacuna-100 dataset as described in (Golatkar et al., 2019). We train a ResNet-18 neural network model to perform binary gender classification. For unlearning, we remove a random subset of the celebrities and their face images.

4.4. Unlearning Metrics

To empirically evaluate unlearning, we consider the error on the unlearned data before and after unlearning. An increase in error suggests the model is losing information about the data. We also employ black-box membership inference attacks (MIA) that aim to distinguish between unlearned data and data that has never been in the training dataset (Shokri et al., 2016). We train a logistic regression model to classify data samples based on the output (logits and loss) of the unlearned model. We measure the success of the MIA with AUROC on the data samples, computed via repeated k-fold cross-validation ($k = 100$ for eICU and $k = 5$ for Lacuna-100). A higher MIA score corresponds to less successful unlearning. We perform the MIA on the unlearned dataset (one class) and an *out-of-distribution* (OOD) dataset (the second class) representing data from *users* absent from the training data. For the Lacuna-100 dataset, we construct an OOD dataset using an additional 100 users from the VGGFace2 database. For the eICU dataset, we collect data from the test set from users not present in the training set to use in the OOD dataset.

4.5. Results

Figure 1 examines the privacy-utility-complexity tradeoff of our algorithm for $\delta = 0.1$ and different values of ϵ, σ, and K. We report the average value over 5 seeds. Figures 1a and 1b show the accuracy-computation and accuracy-unlearning tradeoff of our algorithm, where "rewind percent" is the percent of training iterations, computed as $\frac{K}{T} \times 100\%$. As expected, increasing the amount of iteration K for the same ϵ decreases the error, as does decreasing the strength of the unlearning guarantee for the same amount of computation. Figure 1c shows the theoretical computation-unlearning tradeoff for fixed noise $\sigma = 0.01$ in each setting, for both the numerically computed value of K for values ϵ and the approximation for $K(\epsilon)$ derived in (3.2).

In Figure 1d we empirically assess the extent of unlearning through membership inference attacks. As expected, the MIA score increases as ϵ increases and the unlearning guarantee is relaxed. For the same ϵ, the MIA is generally more successful when less noise is used. For example, for the Lacuna-100 setting, the MIA is most successful on the noiseless fully retrained model. This suggests that the MIA is picking up on distributional differences between the unlearned data and the OOD data reflected in the output of the model, highlighting both the limitations of relying on the MIA score as an unlearning metric as well as the strong advantages of theoretical (δ, ϵ) guarantees. Finally, Figure 1e compares the performance of the unlearned model and the original model (without unlearning) on the unlearned data. The increase in error after unlearning suggests that the model is losing information about the samples.

4.6. Comparison with Baseline Methods

We compare our algorithm against two other algorithms with theoretical (ϵ, δ) unlearning guarantees for nonconvex functions: the Constrained Newton Step method (CNS) (Zhang et al., 2024), a black-box algorithm which involves a single Newton step within a constrained parameter set, and the Hessian-Free method (HF) (Qiao et al., 2024), a white-box algorithm which involves demanding pre-computation and storage of data influence vectors during training, allowing unlearning via simple vector addition later. We do not implement the white-box algorithm (Chien et al., 2024a) because, as stated in their work, "the non-convex unlearning bound... currently is not tight enough to be applied in practice due to its exponential dependence on various hyperparameters." Although HF technically requires $O(nd)$ storage (and $O(ndT)$ storage during training to compute the unlearning vectors), which is impractically large for our dataset and model sizes, for comparison purposes we implement an $O(md)$ version that only stores the vectors for the data samples we plan to unlearn. Figure 2 shows the performance of the three unlearning algorithms on the training, test, and unlearned dataset as well as the MIA scores after unlearning. In terms of accuracy, R2D generally performs better than CNS and worse than HF, depending on the choice of K, due to significant differences in the noise bounds of each algorithm. The MIA results suggest that HF is more successful at defending against membership attacks; however, the low MIA score for CNS is likely due to overfitting of the logistic regression model since the amount of noise added to the model is very high for the range of ϵ considered.

Table 2. Comparison of computation time of algorithms for the Lacuna-100 dataset, with 26% or 51% training iterations for R2D. The results for the eICU dataset are in Table 7 in the Appendix.

ALGORITHM	LEARNING TIME	UNLEARNING TIME
R2D (26%)	1.47 HOURS	0.38 HOURS
R2D (51%)	1.47 HOURS	0.74 HOURS
HF	4.23 DAYS	0.00 HOURS
CNS	0.58 HOURS	0.32 HOURS

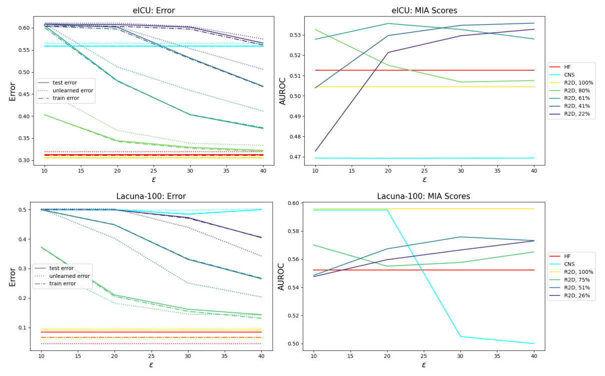


Figure 2. Accuracy and unlearning of baseline unlearning methods. The top row shows the eICU results, and the bottom row shows the Lacuna-100 results. The left plots show the error on the training, test, and unlearned dataset. The right plots show the MIA scores.

Table 2 compares the computation time of each approach during learning and unlearning for Lacuna-100, showing that R2D strongly outperforms HF in learning time, and can outperform CNS in the computation saved from unlearning. In contrast, HF, a white-box algorithm, front-loads the computational burden during learning in favor of fast unlearning later. The findings for eICU, in Table 7 in the Appendix, are similar. Ultimately, while both CNS and HF are second-order methods, R2D is a first-order method that provides competitive performance with minimal computation and storage requirements.

5. Conclusion

We propose the first black-box, first-order certified-unlearning algorithm for nonconvex functions, addressing theoretical and practical limitations of prior work. Our algorithm outperforms existing second-order methods in storage, computation, accuracy, and unlearning.

Impact Statement

This paper presents work whose goal is to advance the field of machine unlearning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pp. 308–318, 2016.

Balle, B. and Wang, Y.-X. Improving the Gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 394–403. PMLR, 10–15 Jul 2018.

Bui, N., Lu, X., Sim, R. H. L., Ng, S.-K., and Low, B. K. H. On Newton's method to unlearn neural networks. arXiv, August 2024.

Cao, Q., Shen, L., Xie, W., Parkhi, O. M., and Zisserman, A. VGGFace2: A dataset for recognising faces across pose and age. In *International Conference on Automatic Face and Gesture Recognition*, 2018.

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

Charles, Z. and Papailiopoulos, D. Stability and generalization of learning algorithms that converge to global optima. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 745–754. PMLR, 10–15 Jul 2018.

Chaudhuri, K., Monteleoni, C., and Sarwate, A. D. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(29):1069–1109, 2011.

Chien, E., Pan, C., and Milenkovic, O. Certified graph unlearning. In *NeurIPS 2022 Workshop: New Frontiers in Graph Learning*, 2022.

Chien, E., Wang, H. P., Chen, Z., and Li, P. Langevin unlearning. In *Privacy Regulation and Protection in Machine Learning*, 2024a.

Chien, E., Wang, H. P., Chen, Z., and Li, P. Certified machine unlearning via noisy stochastic gradient descent. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b.

Drusvyatskiy, D. The proximal point method revisited. arXiv, December 2017. arXiv:1712.06038.

Dwork, C. and Roth, A. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, 8 2014. ISSN 1551-305X.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. In Halevi, S. and Rabin, T. (eds.), *Theory of Cryptography*, pp. 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-32732-5.

Elisseeff, A., Evgeniou, T., and Pontil, M. Stability of randomized learning algorithms. *Journal of Machine Learning Research*, 6(3):55–79, 2005.

Fraboni, Y., Van Waerebeke, M., Scaman, K., Vidal, R., Kameni, L., and Lorenzi, M. SIFU: Sequential informed federated unlearning for efficient and provable client unlearning in federated optimization. In Dasgupta, S., Mandt, S., and Li, Y. (eds.), *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 3457–3465. PMLR, 02–04 May 2024.

Ginart, A., Guan, M., Valiant, G., and Zou, J. Y. Making AI forget you: Data deletion in machine learning. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Goel, S., Prabhu, A., and Kumaraguru, P. Evaluating inexact unlearning requires revisiting forgetting. *CoRR*, abs/2201.06640, 2022.

Golatkar, A., Achille, A., and Soatto, S. Eternal sunshine of the spotless net: Selective forgetting in deep networks. *CoRR*, abs/1911.04933, 2019.

Golatkar, A., Achille, A., and Soatto, S. Forgetting outside the box: Scrubbing deep networks of information accessible from input-output observations. *CoRR*, abs/2003.02960, 2020.

Graves, L., Nagisetty, V., and Ganesh, V. Amnesiac machine learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(13):11516–11524, May 2021. ISSN 2374-3468.

Guo, C., Goldstein, T., Hannun, A. Y., and van der Maaten, L. Certified data removal from machine learning models. *CoRR*, abs/1911.03030, 2019.

Izzo, Z., Anne Smart, M., Chaudhuri, K., and Zou, J. Approximate data deletion from machine learning models. In Banerjee, A. and Fukumizu, K. (eds.), *Proceedings of The*

24th International Conference on Artificial Intelligence and Statistics, volume 130 of *Proceedings of Machine Learning Research*, pp. 2008–2016. PMLR, 2021.

Jang, J., Yoon, D., Yang, S., Cha, S., Lee, M., Logeswaran, L., and Seo, M. Knowledge unlearning for mitigating privacy risks in language models. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14389–14408, Toronto, Canada, July 2023. Association for Computational Linguistics.

Karimi, H., Nutini, J., and Schmidt, M. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In Frasconi, P., Landwehr, N., Manco, G., and Vreeken, J. (eds.), *Machine Learning and Knowledge Discovery in Databases*, pp. 795–811, Cham, 2016. Springer International Publishing.

Kurmanji, M., Triantafillou, P., Hayes, J., and Triantafillou, E. Towards unbounded machine unlearning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Lei, Y. and Ying, Y. Sharper generalization bounds for learning with gradient-dominated objective functions. In *International Conference on Learning Representations*, 2021.

Liu, C., Zhu, L., and Belkin, M. Loss landscapes and optimization in over-parameterized non-linear systems and neural networks. *Applied and Computational Harmonic Analysis*, 59:85–116, 2022. ISSN 1063-5203. Special Issue on Harmonic Analysis and Machine Learning.

Liu, J., Lou, J., Qin, Z., and Ren, K. Certified minimax unlearning with generalization rates and deletion capacity. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Liu, Z., Dou, G., Chien, E., Zhang, C., Tian, Y., and Zhu, Z. Breaking the trilemma of privacy, utility, and efficiency via controllable machine unlearning. In *Proceedings of the ACM on Web Conference 2024*, WWW '24, pp. 1260–1271, New York, NY, USA, 2024. Association for Computing Machinery.

Martinet, B. Brève communication. régularisation d'inéquations variationnelles par approximations successives. *Revue française d'informatique et de recherche opérationnelle. Série rouge*, 4(R3):154–158, 1970. ISSN 0373-8000, 2777-3515.

Neel, S., Roth, A., and Sharifi-Malvajerdi, S. Descent-to-delete: Gradient-based methods for machine unlearning. In Feldman, V., Ligett, K., and Sabato, S. (eds.), *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pp. 931–962. PMLR, 2021.

Nguyen, T. T., Huynh, T. T., Ren, Z., Nguyen, P. L., Liew, A. W.-C., Yin, H., and Nguyen, Q. V. H. A survey of machine unlearning. arXiv, September 2024. arXiv:2209.02299.

Pollard, T. J., Johnson, A. E. W., Raffa, J. D., Celi, L. A., Mark, R. G., and Badawi, O. The eICU collaborative research database, a freely available multi-center database for critical care research. *Scientific Data*, 5(1):180178, September 2018. ISSN 2052-4463.

Qiao, X., Zhang, M., Tang, M., and Wei, E. Efficient and generalizable certified unlearning: A Hessian-free recollection approach. arXiv, June 2024. arXiv:2404.01712.

Sekhari, A., Acharya, J., Kamath, G., and Suresh, A. T. Remember what you want to forget: Algorithms for machine unlearning. *CoRR*, abs/2103.03279, 2021.

Shalev-Shwartz, S., Shamir, O., Srebro, N., and Sridharan, K. Stochastic convex optimization. In *Annual Conference Computational Learning Theory*, 2009.

Shamir, G. I., Lin, D., and Coviello, L. Smooth activations and reproducibility in deep networks. *CoRR*, abs/2010.09931, 2020.

Shokri, R., Stronati, M., and Shmatikov, V. Membership inference attacks against machine learning models. *CoRR*, abs/1610.05820, 2016.

Suriyakumar, V. M. and Wilson, A. C. Algorithms that approximate data removal: New results and limitations. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022.

Terhörst, P., Fährmann, D., Kolf, J. N., Damer, N., Kirchbuchner, F., and Kuijper, A. MAAD-Face: A massively annotated attribute dataset for face images. *CoRR*, abs/2012.01030, 2020.

Ullah, E., Mai, T., Rao, A., Rossi, R. A., and Arora, R. Machine unlearning via algorithmic stability. *CoRR*, abs/2102.13179, 2021.

Wu, X., Li, F., Kumar, A., Chaudhuri, K., Jha, S., and Naughton, J. Bolt-on differential privacy for scalable stochastic gradient descent-based analytics. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, pp. 1307–1322, 2017.

Xu, H., Zhu, T., Zhang, L., Zhou, W., and Yu, P. S. Machine unlearning: A survey. *ACM Comput. Surv.*, 56(1), 8 2023. ISSN 0360-0300.

Zhang, B., Dong, Y., Wang, T., and Li, J. Towards certified unlearning for deep neural networks. In *Forty-first International Conference on Machine Learning*, 2024.

Zhang, J., Zheng, K., Mou, W., and Wang, L. Efficient private ERM for smooth objectives. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 3922–3928, 2017.

A. Proof of Lemma 2.3

Proof. We have the following lemma from (Drusvyatskiy, 2017).

Lemma A.1. *(Drusvyatskiy, 2017) If $f(\theta)$ is continuously differentiable with L-Lipschitz gradient, $-f(\theta) + \frac{L}{2}||\theta||^2$ is convex.*

Now we define θ^* as the solution to the proximal problem as follows

$$\theta^* = prox_{-f,\eta}(\theta_{t+1}) = \arg\min_x\{-f(x) + \frac{1}{2\eta}||x - \theta_{t+1}||^2\},$$

which is well-defined due to Lemma A.1 and the fact that $\eta < \frac{1}{L}$. Then the gradient of the objective function is zero at θ^* and thus

$$-\nabla f(\theta^*) + \frac{1}{\eta}(\theta^* - \theta_{t+1}) = 0,$$
$$\theta^* = \theta_{t+1} + \eta\nabla f(\theta^*).$$

□

B. Proof of Theorem 3.1

Like prior work in differential privacy and machine unlearning, our work hinges on the Gaussian mechanism for differential privacy, which ensures (ϵ, δ)-indistinguishability for normal random variables with the same variance.

Theorem B.1. *(Dwork & Roth, 2014) Let $X \sim \mathcal{N}(\mu, \sigma^2\mathbb{I}_d)$ and $Y \sim \mathcal{N}(\mu', \sigma^2\mathbb{I}_d)$. Suppose $\|\mu - \mu'\|_2 \leq \Delta$. Then for any $\delta > 0$, X and Y are (ϵ, δ)-indistinguishable if*

$$\sigma = \frac{\Delta}{\epsilon}\sqrt{2\log(1.25/\delta)}.$$

Therefore, to prove Theorem 3.1, we need to bound the distance between the output of the unlearning algorithm and the learning algorithm. We can then add a sufficient amount of noise, scaled by this distance, to achieve (ϵ, δ) unlearning.

Proof. Let Z be a dataset of m data points we would like to unlearn, where $m < n$. Let $\mathcal{D}$ represent the original full dataset and $\mathcal{D}' = \mathcal{D}\backslash Z$. Without loss of generality, define

$$f_{\mathcal{D}}(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_{z_i}(\theta),$$

$$f_{\mathcal{D}'}(\theta) = \frac{1}{n-m}\sum_{i=1}^{n-m} f_{z_i}(\theta),$$

such that we have

$$f_{\mathcal{D}} = \frac{n-m}{n} f_{\mathcal{D}'}(\theta) + \frac{1}{n}\sum_{i=n-m+1}^{n} f_{z_i}(\theta),$$

$$f_{\mathcal{D}'} = \frac{n}{n-m}(f_{\mathcal{D}}(\theta) - \frac{1}{n}\sum_{i=n-m+1}^{n} f_{z_i}(\theta)).$$

Let $\{\theta_t\}_{t=0}^T$ represent the gradient descent iterates of the learning algorithm on $f_{\mathcal{D}}$, starting from θ_0, and let $\{\theta'_t\}_{t=0}^T$ be the iterates of the learning algorithm on $f_{\mathcal{D}'}$ starting from the same θ_0. Then we have

$$\theta_0 = \theta'_0, \tag{8}$$
$$\theta_t = \theta_{t-1} - \eta\nabla f_{\mathcal{D}}(\theta_{t-1}), \tag{9}$$
$$\theta'_t = \theta'_{t-1} - \eta\nabla f_{\mathcal{D}'}(\theta'_{t-1}). \tag{10}$$

Let $\{\theta''_t\}_{t=0}^K$ represent the gradient descent iterates of the unlearning algorithm starting at $\theta''_0 = \theta_{T-K}$. Finally, let $\tilde{\theta} = \theta''_K + \xi$ denote the iterate with Gaussian noise added.

We first bound the distance between θ_t and θ'_t as follows.

Lemma B.2. *Let $\{\theta_t\}_{t=0}^T$, $\{\theta'_t\}_{t=0}^T$ be defined as in (8). Then*

$$||\theta_t - \theta'_t|| \leq \frac{2Gm}{Ln}[(1+\frac{\eta L n}{n-m})^t - 1].$$

Proof. We have

$$\nabla f_{\mathcal{D}'}(\theta) = \frac{n}{n-m}(\nabla f_{\mathcal{D}}(\theta) - \frac{1}{n}\sum_{i=n-m+1}^{n} \nabla f_{z_i}(\theta)),$$

$$\theta'_t = \theta'_{t-1} - \eta\frac{n}{n-m}(\nabla f_{\mathcal{D}}(\theta'_{t-1}) - \frac{1}{n}\sum_{i=n-m+1}^{n} \nabla f_{z_i}(\theta'_{t-1})).$$

So we have for $\Delta_t = \theta_t - \theta'_t$,

$$\begin{aligned}
\Delta_t =& \Delta_{t-1} - \eta\nabla f_{\mathcal{D}}(\theta_{t-1}) + \eta\frac{n}{n-m}(\nabla f_{\mathcal{D}}(\theta) - \frac{1}{n}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta'_{t-1})),\\
=& \Delta_{t-1} - \eta\nabla f_{\mathcal{D}}(\theta_{t-1}) + \eta\frac{n}{n-m}\nabla f_{\mathcal{D}}(\theta'_{t-1}) - \eta\frac{1}{n-m}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta'_{t-1}),\\
=& \Delta_{t-1} - \eta\nabla f_{\mathcal{D}}(\theta_{t-1}) + \eta\frac{n}{n-m}\nabla f_{\mathcal{D}}(\theta_{t-1}) + \eta\frac{n}{n-m}(\nabla f_{\mathcal{D}}(\theta'_{t-1}) - \nabla f_{\mathcal{D}}(\theta_{t-1})) - \eta\frac{1}{n-m}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta'_{t-1}),\\
=& \Delta_{t-1} + \eta\frac{m}{n-m}\nabla f_{\mathcal{D}}(\theta_{t-1}) + \eta\frac{n}{n-m}(\nabla f_{\mathcal{D}}(\theta'_{t-1}) - \nabla f_{\mathcal{D}}(\theta_{t-1})) - \eta\frac{1}{n-m}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta'_{t-1}).
\end{aligned}$$

After taking the absolute value of each side, we obtain by the triangle inequality

$$||\Delta_t|| \leq ||\Delta_{t-1}|| + \eta\frac{m}{n-m}||\nabla f_{\mathcal{D}}(\theta_{t-1})|| + \eta\frac{n}{n-m}||\nabla f_{\mathcal{D}}(\theta'_{t-1}) - \nabla f_{\mathcal{D}}(\theta_{t-1})|| + \eta\frac{1}{n-m}\sum_{i=n-m+1}^{n}||\nabla f_{z_i}(\theta'_{t-1})||.$$

Since the gradient on each data sample is bounded by G, we have

$$||\Delta_t|| \leq ||\Delta_{t-1}|| + \eta\frac{m}{n-m}G + \eta\frac{n}{n-m}||\nabla f_{\mathcal{D}}(\theta'_{t-1}) - \nabla f_{\mathcal{D}}(\theta_{t-1})|| - \eta\frac{1}{n-m}\sum_{i=n-m+1}^{n} G,$$

$$||\Delta_t|| \leq ||\Delta_{t-1}|| + \eta\frac{n}{n-m}||\nabla f_{\mathcal{D}}(\theta'_{t-1}) - \nabla f_{\mathcal{D}}(\theta_{t-1})|| + \frac{2\eta Gm}{n-m}.$$

By Lipschitz smoothness of the gradient, we have

$$||\Delta_t|| \leq ||\Delta_{t-1}|| + \eta\frac{n}{n-m}L||\theta'_{t-1} - \theta_{t-1}|| + \frac{2\eta Gm}{n-m},$$

$$||\Delta_t|| \leq ||\Delta_{t-1}||(1 + \frac{\eta Ln}{n-m}) + \frac{2\eta Gm}{n-m}.$$

Since we have $||\Delta_0|| = 0$, evaluating this recursive relationship yields for $t > 0$

$$||\Delta_t|| \leq \frac{2\eta Gm}{n-m}\sum_{i=0}^{t-1}(1 + \frac{\eta Ln}{n-m})^i$$

$$||\Delta_t|| \leq \frac{2\eta Gm}{n-m}\frac{(1+\frac{\eta Ln}{n-m})^t - 1}{\frac{\eta Ln}{n-m}}$$

$$||\Delta_t|| \leq 2Gm\frac{(1+\frac{\eta Ln}{n-m})^t - 1}{Ln}$$

This bound takes advantage of the difference between $f_{\mathcal{D}}$ and $f_{\mathcal{D}'}$ as it decreases with large n, but it also grows exponentially with the number of iterates.

□

Lemma B.3. *Let $\{\theta''_t\}_{t=0}^K$ represent the gradient descent iterates on $f_{\mathcal{D}'}$ starting at $\theta''_0 = \theta_{T-K}$ such that*

$$\theta''_t = \theta''_{t-1} - \eta \nabla f_{\mathcal{D}'}(\theta''_{t-1})$$

Then

$$||\theta'_T - \theta''_K|| \leq ||\theta_{T-K} - \theta'_{T-K}||(1+\eta L)^K$$

Proof. Let $\Delta'_t = \theta'_{T-K+t} - \theta''_t$ such that $\Delta'_0 = ||\theta'_{T-K} - \theta''_0||$ and we bound it as follows

$$\begin{aligned}\Delta'_t = \theta'_{T-K+t} - \theta''_t &= \theta'_{T-K+t-1} - \eta\nabla f_{\mathcal{D}'}(\theta'_{T-K+t-1}) - \theta''_{t-1} + \eta \nabla f_{\mathcal{D}'}(\theta''_{t-1}) \\ &= \Delta'_{t-1} - \eta\nabla f_{\mathcal{D}'}(\theta'_{T-K+t-1}) + \eta \nabla f_{\mathcal{D}'}(\theta''_{t-1}) \\ ||\Delta'_t|| &\leq ||\Delta'_{t-1}|| + \eta||\nabla f_{\mathcal{D}'}(\theta'_{T-K+t-1}) - \nabla f_{\mathcal{D}'}(\theta''_{t-1})||\end{aligned}$$

By Lipschitz smoothness

$$\begin{aligned}||\Delta'_t|| &\leq ||\Delta'_{t-1}|| + L\eta||\theta'_{T-K+t-1} - \theta''_{t-1}|| \\ ||\Delta'_t|| &\leq (1+\eta L)||\Delta'_{t-1}|| \leq ||\Delta'_0||(1+\eta L)^t\end{aligned}$$

□

Returning to the algorithm, suppose the learning algorithm has T iterations and we backtrack for K iterations. Then the difference between the output of the learning algorithm (without noise) on $f_{\mathcal{D}'}$ and the unlearning algorithm would be

$$||\theta'_T - \theta''_K|| \leq ||\Delta_{T-K}||(1+\eta L)^K \leq \frac{2mG}{Ln}((1+\frac{\eta L n}{n-m})^{T-K} - 1)(1+\eta L)^K$$

where the bound on the right hand side decreases monotonically as K increases from 0 to T, as shown by the following. Let

$$h(K) = ((1+\frac{\eta L n}{n-m})^{T-K} - 1)(1+\eta L)^K.$$

Then the derivative is

$$h'(K) = (1+\eta L)^K[((1+\frac{\eta Ln}{n-m})^{T-K} - 1)\log(1+\eta L) - (1+\frac{\eta Ln}{n-m})^{T-K}\log(1+\frac{\eta Ln}{n-m})],$$

we observe that $h'(K) < 0$ for $K \in [0,T]$. Therefore $h(K)$ is decreasing.

Therefore by Theorem B.1, to achieve ϵ, δ-unlearning, we need to set the value of σ as follows

$$\sigma = \frac{||\theta'_T - \theta''_K||\sqrt{2\log(1.25/\delta)}}{\epsilon} = \frac{2mG \cdot h(K)\sqrt{2\log(1.25/\delta)}}{Ln\epsilon}.$$

Now we prove the utility guarantee. For nonconvex smooth functions, we know by standard analysis that for the gradient descent iterates θ''_t, we have

$$\frac{\eta}{2}\sum_{t=0}^{K}||\nabla f_{\mathcal{D}'}(\theta''_t)|| \leq \sum_{t=0}^{K} f_{\mathcal{D}'}(\theta''_t) - f_{\mathcal{D}'}(\theta''_{t+1}) = f_{\mathcal{D}'}(\theta''_0) - f_{\mathcal{D}'}(\theta''_K)$$

Now we consider the progress of the iterates θ_t on $f_{\mathcal{D}'}(\theta)$. By Lipschitz smoothness, we have

$$f_{\mathcal{D}'}(\theta_{t+1}) \leq f_{\mathcal{D}'}(\theta_t) + \langle \nabla f_{\mathcal{D}'}(\theta_t), -\eta\nabla f_{\mathcal{D}}(\theta_t)\rangle + \frac{L}{2}||\eta\nabla f_{\mathcal{D}}(\theta_t)||^2$$

We have

$$\nabla f_{\mathcal{D}}(\theta_t) = \frac{n-m}{n}\nabla f_{\mathcal{D}'}(\theta_t) + \frac{1}{n}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta_t)$$

$$f_{\mathcal{D}'}(\theta_{t+1}) \leq f_{\mathcal{D}'}(\theta_t) - \eta\langle\nabla f_{\mathcal{D}'}(\theta_t), \frac{n-m}{n}\nabla f_{\mathcal{D}'}(\theta_t) + \frac{1}{n}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta_t)\rangle + \frac{L\eta^2}{2}||\frac{n-m}{n}\nabla f_{\mathcal{D}'}(\theta_t) + \frac{1}{n}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta_t)||^2$$

$$\leq f_{\mathcal{D}'}(\theta_t) - \eta\frac{n-m}{n}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 - \eta\langle\nabla f_{\mathcal{D}'}(\theta_t), \frac{1}{n}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta_t)\rangle + \frac{L\eta^2}{2}||\frac{n-m}{n}\nabla f_{\mathcal{D}'}(\theta_t) + \frac{1}{n}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta_t)||^2$$

$$\leq f_{\mathcal{D}'}(\theta_t) - \eta\frac{n-m}{n}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 - \eta\langle\nabla f_{\mathcal{D}'}(\theta_t), \frac{1}{n}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta_t)\rangle + L\eta^2||\frac{n-m}{n}\nabla f_{\mathcal{D}'}(\theta_t)||^2 + L\eta^2||\frac{1}{n}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta_t)||^2$$

$$\leq f_{\mathcal{D}'}(\theta_t) - \eta\frac{n-m}{n}(1 - \frac{L\eta(n-m)}{n})||\nabla f_{\mathcal{D}'}(\theta_t)||^2 - \eta\langle\nabla f_{\mathcal{D}'}(\theta_t), \frac{1}{n}\sum_{i=n-m+1}^{n}\nabla f_{z_i}(\theta_t)\rangle + L\eta^2\frac{1}{n}\sum_{i=n-m+1}^{n}||\nabla f_{z_i}(\theta_t)||^2$$

$$\leq f_{\mathcal{D}'}(\theta_t) - \eta\frac{n-m}{n}(1 - \frac{L\eta(n-m)}{n})||\nabla f_{\mathcal{D}'}(\theta_t)||^2 + \eta\frac{G^2m}{n} + L\eta^2\frac{mG}{n}$$

Let the step size η be bounded such that $\eta \leq \frac{n}{2(n-m)L}$, then we have

$$f_{\mathcal{D}'}(\theta_{t+1}) \leq f_{\mathcal{D}'}(\theta_t) - \frac{\eta(n-m)}{2n}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 + \frac{\eta G^2 m}{n} + \frac{L\eta^2 Gm}{n}$$

Rearrange the terms to get

$$\frac{\eta(n-m)}{2n}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 \leq f_{\mathcal{D}'}(\theta_t) - f_{\mathcal{D}'}(\theta_{t+1}) + \frac{\eta G^2 m}{n} + \frac{L\eta^2 Gm}{n}$$

Sum from $t = 0$ to $T - K - 1$

$$\frac{\eta(n-m)}{2n}\sum_{t=0}^{T-K-1}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 \leq f_{\mathcal{D}'}(\theta_0) - f_{\mathcal{D}'}(\theta_{T-K}) + (T-K-1)(\frac{\eta G^2 m}{n} + \frac{L\eta^2 Gm}{n})$$

$$\sum_{t=0}^{T-K-1}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 \leq \frac{2n}{\eta(n-m)}(f_{\mathcal{D}'}(\theta_0) - f_{\mathcal{D}'}(\theta_{T-K}) + (T-K-1)(\frac{\eta G^2 m}{n} + \frac{L\eta^2 Gm}{n}))$$

$$\sum_{t=0}^{T-K-1}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 \leq \frac{2n}{\eta(n-m)}(f_{\mathcal{D}'}(\theta_0) - f_{\mathcal{D}'}(\theta_{T-K})) + (T-K-1)(\frac{2G^2 m}{n-m} + \frac{2L\eta Gm}{n-m})$$

From standard analysis of gradient descent on nonconvex functions, we know

$$\eta\sum_{t=0}^{K}||\nabla f_{\mathcal{D}'}(\theta''_t)||^2 \leq f_{\mathcal{D}'}(\theta''_0) - f_{\mathcal{D}'}(\theta''_K)$$

$$\sum_{t=0}^{K}||\nabla f_{\mathcal{D}'}(\theta''_t)||^2 \leq \frac{1}{\eta}(f_{\mathcal{D}'}(\theta''_0) - f_{\mathcal{D}'}(\theta''_K))$$

Summing the equations yields

$$\sum_{t=0}^{T-K-1}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 + \sum_{t=0}^{K}||\nabla f_{\mathcal{D}'}(\theta''_t)||^2$$

$$\leq \frac{2n}{\eta(n-m)}(f_{\mathcal{D}'}(\theta_0) - f_{\mathcal{D}'}(\theta_{T-K})) + (T-K-1)(\frac{2G^2 m}{n-m} + \frac{2L\eta Gm}{n-m}) + \frac{1}{\eta}(f_{\mathcal{D}'}(\theta''_0) - f_{\mathcal{D}'}(\theta''_K))$$

$$\leq \frac{2n}{\eta(n-m)}(f_{\mathcal{D}'}(\theta_0) - f_{\mathcal{D}'}(\theta''_K)) + (T-K-1)(\frac{2G^2m}{n-m} + \frac{2L\eta Gm}{n-m})$$

We can expand $||\nabla f_{\mathcal{D}'}(\theta''_K)||^2$ as follows

$$\begin{aligned}||\nabla f_{\mathcal{D}'}(\theta''_K)||^2 &= ||\nabla f_{\mathcal{D}'}(\tilde{\theta}'')||^2 + ||\nabla f_{\mathcal{D}'}(\theta''_K)||^2 - ||\nabla f_{\mathcal{D}'}(\tilde{\theta}'')||^2 \\ &= ||\nabla f_{\mathcal{D}'}(\tilde{\theta}'')||^2 + 2\nabla f_{\mathcal{D}'}(\tilde{\theta}'')^T(\nabla f_{\mathcal{D}'}(\theta''_K) - \nabla f_{\mathcal{D}'}(\tilde{\theta}'')) + ||\nabla f_{\mathcal{D}'}(\theta''_K) - \nabla f_{\mathcal{D}'}(\tilde{\theta}'')||^2 \\ &= ||\nabla f_{\mathcal{D}'}(\tilde{\theta}'')||^2 + 2\nabla f_{\mathcal{D}'}(\tilde{\theta}'')^T\xi + ||\xi||^2\end{aligned}$$

So we have

$$\begin{aligned}\frac{1}{T}\Big[\sum_{t=0}^{T-K-1}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 + \sum_{t=0}^{K-1}||\nabla f_{\mathcal{D}'}(\theta''_t)||^2 + \mathbb{E}[||\nabla f_{\mathcal{D}'}(\tilde{\theta}'')||^2]\Big] \leq & \frac{2n(f_{\mathcal{D}'}(\theta_0) - f_{\mathcal{D}'}(\theta''_K))}{T\eta(n-m)} \\ &+ \frac{T-K-1}{T}(\frac{2G^2m}{n-m} + \frac{2L\eta Gm}{n-m})\end{aligned}$$

B.1. Proof of Corollary 3.5

As before, we first consider the gradient descent iterates on $f_{\mathcal{D}'}$. For μ-PL and smooth functions, we know that for the iterates θ''_t, we have (Karimi et al., 2016)

$$f_{\mathcal{D}'}(\theta''_t) - f^*_{\mathcal{D}'} \leq (1-\eta\mu)^t(f_{\mathcal{D}'}(\theta''_0) - f^*_{\mathcal{D}'})$$

Now we track the progress of the iterates θ_t on $f_{\mathcal{D}'}$. By Lipschitz smoothness and the above analysis, we have

$$f_{\mathcal{D}'}(\theta_{t+1}) - f_{\mathcal{D}'}(\theta_t) \leq \langle \nabla f_{\mathcal{D}'}(\theta_t), -\eta\nabla f_{\mathcal{D}}(\theta_t)\rangle + \frac{L}{2}||\eta\nabla f_{\mathcal{D}}(\theta_t)||^2$$

$$f_{\mathcal{D}'}(\theta_{t+1}) - f_{\mathcal{D}'}(\theta_t) \leq -\eta\frac{n-m}{n}(1 - \frac{L\eta(n-m)}{n})||\nabla f_{\mathcal{D}'}(\theta_t)||^2 + \eta\frac{G^2m}{n} + L\eta^2\frac{mG}{n}$$

Let the step size η be bounded such that $\eta \leq \frac{n}{2(n-m)L}$, then we have

$$f_{\mathcal{D}'}(\theta_{t+1}) - f_{\mathcal{D}'}(\theta_t) \leq -\frac{\eta(n-m)}{2n}||\nabla f_{\mathcal{D}'}(\theta_t)||^2 + \frac{\eta G^2m}{n} + \frac{L\eta^2Gm}{n}$$

By the PL inequality, we have

$$f_{\mathcal{D}'}(\theta_{t+1}) - f_{\mathcal{D}'}(\theta_t) \leq -\frac{\eta\mu(n-m)}{n}(f_{\mathcal{D}'}(\theta_t) - f^*_{\mathcal{D}'}) + \frac{\eta G^2m}{n} + \frac{L\eta^2Gm}{n}$$

$$f_{\mathcal{D}'}(\theta_{t+1}) - f^*_{\mathcal{D}'} \leq (1 - \frac{\eta\mu(n-m)}{n})(f_{\mathcal{D}'}(\theta_t) - f^*_{\mathcal{D}'}) + \frac{\eta G^2m}{n} + \frac{L\eta^2Gm}{n}$$

We evaluate this recursive relationship to obtain

$$f_{\mathcal{D}'}(\theta_{t+1}) - f^*_{\mathcal{D}'} \leq (1 - \frac{\eta\mu(n-m)}{n})^{t+1}(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + \frac{1}{n}(\eta G^2m + L\eta^2Gm)\sum_{i=0}^{t}(1 - \frac{\eta\mu(n-m)}{n})^i$$

$$\begin{aligned}f_{\mathcal{D}'}(\theta_{t+1}) - f^*_{\mathcal{D}'} &\leq (1 - \frac{\eta\mu(n-m)}{n})^{t+1}(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + \frac{1}{n}(\eta G^2m + L\eta^2Gm)\sum_{i=0}^{t}(1 - \frac{\eta\mu(n-m)}{n})^i \\ &\leq (1 - \frac{\eta\mu(n-m)}{n})^{t+1}(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + \frac{1}{n}(\eta G^2m + L\eta^2Gm)\frac{n}{\eta\mu(n-m)}\end{aligned}$$

$$f_{\mathcal{D}'}(\theta_{t+1}) - f^*_{\mathcal{D}'} \leq (1 - \frac{\eta\mu(n-m)}{n})^{t+1}(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + \frac{G^2m + L\eta Gm}{\mu(n-m)}$$

$$f_{\mathcal{D}'}(\theta''_0) - f^*_{\mathcal{D}'} = f_{\mathcal{D}'}(\theta_{T-K}) - f^*_{\mathcal{D}'} \le (1-\frac{\eta\mu(n-m)}{n})^{T-K}(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + \frac{G^2m + L\eta Gm}{\mu(n-m)}$$

$$f_{\mathcal{D}'}(\theta''_K) - f^*_{\mathcal{D}'} \le (1-\frac{\eta\mu(n-m)}{n})^{T-K}(1-\eta\mu)^K(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + (1-\eta\mu)^K\frac{G^2m + L\eta Gm}{\mu(n-m)} \tag{11}$$

By Lipschitz smoothness, we have

$$\begin{aligned}
f_{\mathcal{D}'}(\tilde{\theta}'') - f^*_{\mathcal{D}'} &\le f_{\mathcal{D}'}(\tilde{\theta}'') - f_{\mathcal{D}'}(\theta''_K) + (1-\frac{\eta\mu(n-m)}{n})^{T-K}(1-\eta\mu)^K(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + (1-\eta\mu)^K\frac{G^2m + L\eta Gm}{\mu(n-m)} \\
&\le L||\tilde{\theta}'' - \theta''_K|| + (1-\frac{\eta\mu(n-m)}{n})^{T-K}(1-\eta\mu)^K(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + (1-\eta\mu)^K\frac{G^2m + L\eta Gm}{\mu(n-m)} \\
&= L||\xi|| + (1-\frac{\eta\mu(n-m)}{n})^{T-K}(1-\eta\mu)^K(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + (1-\eta\mu)^K\frac{G^2m + L\eta Gm}{\mu(n-m)}
\end{aligned} \tag{12}$$

If we take the expectation on both sides with respect to the noise added at the end of the algorithm U, we have

$$\mathbb{E}[f_{\mathcal{D}'}(\tilde{\theta}'')] - f^*_{\mathcal{D}'} \le L\sqrt{d}\sigma + (1-\frac{\eta\mu(n-m)}{n})^{T-K}(1-\eta\mu)^K(f_{\mathcal{D}'}(\theta_0) - f^*_{\mathcal{D}'}) + (1-\eta\mu)^K\frac{G^2m + L\eta Gm}{\mu(n-m)}$$

B.2. Proof of Corollary 3.6

Bounds on generalization can be derived using seminal results in algorithmic stability (Elisseeff et al., 2005). Prior work on the generalization ability of unlearning algorithms focus on the strongly convex case (Sekhari et al., 2021; Qiao et al., 2024; Liu et al., 2023; Ullah et al., 2021), which feature excess risk bounds for the empirical risk minimizer when the loss is strongly convex (Shalev-Shwartz et al., 2009). In contrast, algorithms on PL objective functions can at best satisfy pointwise (Charles & Papailiopoulos, 2018) or on-average (Lei & Ying, 2021) stability. The following result derives from Theorem 1 of (Lei & Ying, 2021). We utilize the fact that for $w^* \in \arg\min_{\theta\in\Theta} F(\theta)$, we have $\mathbb{E}[f^*_{\mathcal{D}'}] \le \mathbb{E}[f_{\mathcal{D}'}(w^*)] = F(w^*) = F^*$, and we also slightly modify the result to include the stronger bounded gradient assumption used in our work.

Lemma B.4. *(Lei & Ying, 2021) For F defined in (7) and θ as the output of an algorithm dependent on $\mathcal{D}'$, we have*

$$\mathbb{E}[F(\theta) - F^*] \le \frac{2G^2}{(n-m)\mu} + \frac{L}{2\mu}\mathbb{E}[f_{\mathcal{D}'}(\theta) - f^*_{\mathcal{D}'}].$$

We obtain the result by substituting in (11).

B.3. Proof of Corollary 3.2

We want to determine the value of K required to maintain a privacy level ϵ for a chosen level of noise σ. From (3) we have

$$\epsilon = \frac{h(K)2mG\sqrt{2\log(1.25/\delta)}}{\sigma Ln}$$

Although this does not have an exact explicit solution for $K(\epsilon)$, we can bound $h(K)$ as follows

$$h(K) \le ((1+\frac{\eta Ln}{n-m})^{T-K} - 1)(1+\frac{\eta Ln}{n-m})^K = (1+\frac{\eta Ln}{n-m})^T - (1+\frac{\eta Ln}{n-m})^K$$

Then we have

$$\begin{aligned}
\epsilon &\le \frac{2mG\sqrt{2\log(1.25/\delta)}}{\sigma Ln}((1+\frac{\eta Ln}{n-m})^T - (1+\frac{\eta Ln}{n-m})^K) \\
(1+\frac{\eta Ln}{n-m})^K &\le (1+\frac{\eta Ln}{n-m})^T - \frac{\sigma Ln\epsilon}{2mG\sqrt{2\log(1.25/\delta)}} \\
K &\le \frac{\log\left((1+\frac{\eta Ln}{n-m})^T - \frac{\sigma Ln\epsilon}{2mG\sqrt{2\log(1.25/\delta)}}\right)}{\log(1+\frac{\eta Ln}{n-m})}
\end{aligned}$$

We therefore obtain a close upper bound on $K(\epsilon)$ for fixed σ. In practice we can choose K equal to this bound to ensure the privacy guarantee is achieved. We show in Figure 1c that this bound is close to tight for $0 < \epsilon \le 1$ and real-world parameters.

C. Additional Discussion of Related Work

To review, our algorithm is a first-order, black-box algorithm that provides (ϵ, δ) unlearning while also maintaining performance and computational efficiency. In this section, we provide an in-depth comparison our algorithmic differences with existing convex and nonconvex certified unlearning algorithms (Table 3). These algorithms all involve injecting some (Gaussian) noise to render the algorithm outputs probabilistically indistinguishable, whether it is added to the objective function, to the final model weights, or at each step of the training process.

Algorithm	**Standard Deviation of Injected Gaussian Noise**	**Bounded $\|\theta\| < R$?**
Newton Step (Guo et al., 2019)	$\sigma = \frac{4LG^2\sqrt{2\log(1.5/\delta)}}{\lambda^2(n-1)\epsilon}$	No
Descent-to-delete (D2D) (Neel et al., 2021)	$\sigma = \frac{4\sqrt{2}M(\frac{L-\lambda}{L+\lambda})^K}{\lambda n(1-(\frac{L-\lambda}{L+\lambda})^K)(\sqrt{\log(1/\delta)+\epsilon}-\sqrt{\log(1/\delta)})}$	No
Langevin Unlearning (Chien et al., 2024a)	No closed-form solution for σ in terms of ϵ, δ	Yes
Constrained Newton Step (CNS) (Zhang et al., 2024)	$\sigma = \frac{\left(\frac{2R(PR+\lambda)}{\lambda+\lambda_{min}} + \frac{32\sqrt{\log(d/\rho)}}{\lambda+\lambda_{min}} + \frac{1}{8}\right) LR\sqrt{2\log(1.25/\delta)}}{\epsilon}$	Yes
Hessian-Free unlearning (HF) (Qiao et al., 2024)	$\sigma = 2\eta G\frac{\sqrt{2\log(1.25/\delta)}}{\epsilon}\zeta_T^{-U}$	No
Our Work (R2D)	$\sigma = \frac{2mG \cdot h(K)\sqrt{2\log(1.25/\delta)}}{Ln\epsilon}$	No

Table 3. Comparison of certified unlearning algorithms and their noise guarantees. We denote K as the number of unlearning iterates, λ as the regularization constant or strongly convex parameter, M as the Lipschitz continuity parameter, and d as the model parameter dimension. In addition, R is the parameter norm constraint. For (Zhang et al., 2024), P represents the Lipschitz constant of the Hessian, λ_{min} represents the minimum eigenvalue of the Hessian, and ρ represents a probability less than 1. For (Qiao et al., 2024), ζ_T^{-U} is a constant dependent on the upper and lower bounds of the Hessian spectrum that grows with the number of learning iterates T for nonconvex objective functions.

We are interested in comparing the noise-unlearning tradeoff of each of these algorithms. The second column of Table 3 lists the standard deviation σ in terms of ϵ, δ, and other problem parameters for the other certified unlearning algorithms. Our result is analogous to that of (Neel et al., 2021), due to algorithmic similarities such as weight perturbation at the end of training. However, the required noise in (Neel et al., 2021) decays exponentially with unlearning iterations, while our noise decreases more slowly with increasing K for our algorithm. This difference is because (Neel et al., 2021) considers strongly convex functions, where trajectories are attracted to a global minimum.

The third column of Table 3 highlights an additional advantage our algorithm has over existing approaches. Our theoretical guarantees do not require a uniform bound on the model weights $||\theta|| \leq R$, nor does σ depend on R, in contrast to (Chien et al., 2024a) and (Zhang et al., 2024). Both (Zhang et al., 2024; Chien et al., 2024a) require a bounded feasible parameter set, which they leverage to provide a loose bound on the distance between the retraining and unlearning outputs dependent on the parameter set radius R. However, this yields excessive noise requirements, with the standard deviation σ scaling linearly or polynomially with R. For example, a similar scaling with R can be achieved by a random selection of feasible weights. Suppose we have an arbitrary learning algorithm that outputs some model weights θ perturbed by Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2\mathbb{I}_d)$ and an arbitrary unlearning algorithm that also outputs some other model weights θ' perturbed by Gaussian noise $\xi' \sim \mathcal{N}(0, \sigma^2\mathbb{I}_d)$. Then we have

$$\tilde{\theta} = \theta + \xi$$
$$\tilde{\theta'} = \theta' + \xi'$$

$$||\theta - \theta'|| \leq 2R.$$

Then by Theorem B.1, our unlearning algorithm is (ϵ, δ) unlearning as long as

$$\sigma \geq \frac{2R\sqrt{2\log(1.25/\delta)}}{\epsilon}.$$

In conclusion, an algorithm that outputs arbitrary weights in the feasible set can obtain unlearning with $\sigma = O(R)$, which is of the same order as the dependencies in (Chien et al., 2024a) and (Zhang et al., 2024).

Our work shares similarities with (Chien et al., 2024a), a first-order white-box algorithm that uses noisy projected gradient descent to achieve certified unlearning, leveraging the existence and uniqueness of the limiting distribution of the training process as well as the boundedness of the projection set. Their guarantee shows that the privacy loss ϵ decays exponentially with the number of unlearning iterates. However, σ, the noise added at each step, is defined implicitly with no closed-form solution. We can only assert that for a fixed number of iterations K, σ must be at least $O(R)$ to obtain $\epsilon = O(1)$. Because σ cannot be defined explicitly, it is difficult to implement this algorithm for a desired ϵ. For example, when performing experiments for the strongly convex setting, which is a simpler mathematical expression, they require an additional subroutine to find the smallest σ that satisfies the target ϵ. As for the nonconvex setting, they state that "the non-convex unlearning bound... currently is not tight enough to be applied in practice due to its exponential dependence on various hyperparameters."

D. Experimental Details and Additional Results

Code is open-sourced at the following anonymized GitHub link: https://github.com/anonymous-1234567/r2d.

D.1. Implementation Details

Model architecture. Because our analysis requires smooth functions, we replace ReLU activations with SmeLU activations (Shamir et al., 2020). We also do not use Batch Normalization layers. These steps may affect model performance but improve the soundness of our experiments by allowing estimation of the smoothness constant L.

Gradient bound estimation. To estimate the gradient bound G, we compute the norm of each minibatch gradient at each step of the training process, and take the maximum of these values as the estimate.

Lipschitz constant estimation. To estimate the Lipschitz constant L, we perturb the model weights after training by Gaussian noise with $\sigma = 0.01$, and we estimate the Lipschitz constant by computing

$$\hat{L} = \frac{||\theta_1 - \theta_2||}{||\nabla f(\theta_1) - \nabla f(\theta_2)||}.$$

We sample 400 perturbed weight samples and take the maximum of all the estimates $\hat{L}$ to be our Lipschitz constant.

Experiment Parameter	eICU and MLP	Lacuna-100 and ResNet-18
Size of training dataset n	94449	32000
Number of users	119282	100
Percent data unlearned	$\sim 1\%$	$\sim 2\%$
Number of model parameters d	2172034	11160258
Batch size	512	256
L	0.14394	5.6530
G	1.70994	9.481617
η	0.0004638	1.966e-05
Number of training epochs	52	82

Table 4. Experiment parameters for the eICU and Lacuna-100 datasets.

Model selection. To simulate real-world practices, we perform model selection during the initial training on the full dataset by training until the validation loss converges, and then selecting the model parameters with the lowest validation loss. We treat the selected iteration as the final training iterate. For consistency between experiments under different seeds, we only perform model selection and parameter estimation on the experiments for seed=1 and utilize the resulting estimated values for all other experiments.

Sensitivity of MIA tests. We conduct MIAs to assess if information about the unlearned data is retained in the model weights and reflected in the model outputs. Prior works typically perform the attack comparing the unlearned dataset and the test dataset, the latter of which represents data previously unseen by the model. However, in our setting, the model may perform well on users present in both the training data and the test data, while performing poorly on the users that are unlearned. It is therefore more appropriate membership inference attack on out-of-distribution (OOD) data that contains data from *users* absent from the training data.

MIAs are highly sensitive to distributional differences due to the non-uniform sampling of the unlearned data and the performance of the model. For example, for large σ, the perturbed model may perform poorly by classifying all data in a single class. If the class distribution in the unlearned dataset is different from the class distribution in the non-training dataset, the MIA may succeed by simply identifying the majority class in the unlearned dataset. To counter this effect, we ensure that the OOD dataset used for MIA and the unlearned dataset have the same class distribution. Overall, extra caution is necessary when interpreting MIA results as an unlearning metric, especially when the original model is not extremely accurate or when the unlearned and test datasets are not identically distributed.

Unlearning metrics. Many prior works solely use the error on the unlearned dataset as an unlearning metric, theorizing that the model should perform poorly on data it has never seen before. However, this is questionable in our context. For example, for Lacuna-100, unlearning a user's facial data does not preclude the model from correctly classifying their gender later, especially if the user's photos are clear and easily identifiable. Moreover, since the unlearned data contains a small subset of the users in the training data, it is also likely to have lower variance. As a result, the unlearned model may even perform better on the unlearned data than on the training data. Instead of comparing error on the unlearned, training, and test set, we consider the unlearned error before and after unlearning as in Figure 1e.

Baseline implementations. We include our own implementations of each baseline method in the code. To implement the baselines, we use our estimated values of L and G when applicable, and we change the step size and batch size so that the learning algorithms are sufficiently converged for our problems. For other parameters, such as minimum Hessian eigenvalue or Hessian Lipschitz constant, we use the default values given in the original works. We trained each unique learning algorithm on each dataset until convergence before adding noise. Table 5 reports the *noiseless* test error of each algorithm, showing that they achieve comparable performance prior to unlearning.

Learning Algorithm	**eICU Test Error**	**Lacuna-100 Test Error**
Rewind-to-Delete (R2D)	0.3055	0.0846
Hessian-Free (HF)	0.3125	0.0846
Constrained Newton Step (CNS)	0.2895	0.0462

Table 5. Noiseless test error of original trained models before unlearning.

Experiment Parameter	**eICU**	**Lacuna-100**
Batch size	512	256
η_0	0.1	0.1
Step size decay	0.995	0.995
Gradient norm clipping	5	5
Number of training epochs	15	25

Experiment Parameter	**eICU**	**Lacuna-100**
Batch size	128	128
η	0.001	0.001
Weight decay	0.0005	0.0005
Parameter norm constraint R	10	21
Number of training epochs	30	30

Table 6. Experiment parameters of HF (left) and CNS (right) for the eICU and Lacuna-100 datasets.

Table 6 shows some of the parameters used for implementing the baselines. Additional information can be found in the attached code. The implementation of (Zhang et al., 2024) is based on code from https://github.com/zhangbinchi/certified-deep-unlearning, and the implementation of (Qiao et al., 2024) is based on code from https://github.com/Anonymous202401/If-Recollecting-were-Forgetting.

Additional details. All experiments were run on an NVIDIA GeForce RTX 4060 GPU (8 GB) with PyTorch 2.4.1 and CUDA 12.1, except for the HF experiments and the time experiments in Table 2 and Table 7, which were run on an NVIDIA

RTX A6000 GPU (48 GB) with PyTorch 2.5.1 and CUDA 12.4. We considered the seeds $[0, 1, 2, 3, 4]$ in all experiments except the baseline comparisons, for which we only use seed 1 due to computational constraints. In addition, we use the same Gaussian noise vector for models under the same seed, rescaled to different standard deviations. We found that using unique noise for every hyperparameter combination resulted in similar but noisier trends. When initialized with the same seed, the `torch.randn` method outputs the same noise vector rescaled by the standard deviation. The pre-unlearning noise uses a different seed.

D.2. Additional Baseline Comparisons

Because we desire machine unlearning algorithms that achieve a computational advantage over training from scratch, it is useful to compare the amount of time required for learning and unlearning for different algorithms. The results in Table 2 and 7 demonstrate that R2D unlearning achieves a competitive advantage compared to HF and CNS. As expected based on its straightforward algorithmic structure, the unlearning time of R2D is proportional to the number of rewind iterations.

These results also depend on our implementation of the learning algorithms of HF and CNS. For HF, we note in Table 6 that we use fewer training iterations for HF compared to R2D. Despite this, HF requires *104 times more* compute time during learning, highlighting the benefits of black-box algorithms that dovetail with standard training practices and do not require complicated procedures during training. Similarly, CNS also uses fewer training iterations than R2D to converge to a model with competitive performance; as shown in Table 5, the original model trained with CNS achieves a lower error, likely due to the use of advanced optimization techniques like momentum and regularization. However, because CNS unlearning requires an expensive second-order operation independent of the training process, it only displays a moderate computational advantage on the Lacuna-100 dataset and no advantage on the eICU dataset. In contrast, by construction R2D unlearning will always require less computation than learning, demonstrating the benefits of a cheap first-order approach.

Algorithm	**Learning Time**	**Unlearning Time**
Rewind-to-Delete (R2D), 22%	194.33 sec	41.50 sec
Rewind-to-Delete (R2D), 41%	194.33 sec	81.53 sec
Hessian-Free (HF)	5.65 hours	0.01 sec
Constrained Newton Step (CNS)	266.0 sec	293.6 sec

Table 7. Comparison of computation time of algorithms for the eICU dataset, with 22% or 41% training iterations for R2D. These results and Table 2 were achieved on an NVIDIA RTX A6000 GPU (48 GB).

Tables 8 and 9 show the unlearned, test, and train errors for the baseline algorithms using $\epsilon = 40$ and $\delta = 0.1$. Both (Zhang et al., 2024) and (Qiao et al., 2024) consider a weaker definition of certified unlearning, introduced in (Sekhari et al., 2021), that only considers indistinguishability with respect to the output of the unlearning algorithms, with or without certain data samples, as opposed to the definition used in our work, which considers indistinguishability with respect to the learning algorithm and the unlearning algorithm. As a result, neither (Zhang et al., 2024) nor (Qiao et al., 2024) require adding noise during the initial learning process. Figure 2 displays data for other values of ϵ.

Algorithm	Unlearned Error	Test Error	Train Error	Original Model Test Error	MIA Score
R2D, 22 %	0.5747	0.5655	0.5611	0.3292	0.5328
R2D, 41%	0.5060	0.4675	0.4664	0.3193	0.5358
R2D, 61%	0.4111	0.3719	0.3735	0.3119	0.5280
R2D, 80%	0.3337	0.3222	0.3197	0.3081	0.5075
R2D, 100%	0.3043	0.3055	0.3049	0.3055	0.5045
HF	0.3195	0.3125	0.3104	0.3125	0.5127
CNS	0.5649	0.5588	0.5581	0.2895	0.4693

Table 8. eICU baseline method comparisons for $\epsilon = 40$, $\delta = 0.1$.

Algorithm	Unlearned Error	Test Error	Train Error	Original Model Test Error	MIA Score
R2D, 26%	0.3425	0.4045	0.4059	0.4057	0.5730
R2D, 51%	0.2036	0.2652	0.2671	0.2680	0.5732
R2D, 75%	0.1422	0.1433	0.1313	0.1400	0.5652
R2D, 100%	0.0969	0.0912	0.0647	0.0863	0.5958
HF	0.0452	0.0846	0.0660	0.0846	0.5523
CNS	0.4976	0.5000	0.5000	0.0462	0.5949

Table 9. Lacuna-100 baseline method comparisons for $\epsilon = 40$, $\delta = 0.1$.

D.3. Noiseless Version

Although certified unlearning provides a precise mathematical notion of unlearning, it requires adding a large amount of noise which can degrade model performance. As shown in Figure 3, the noiseless version of R2D may also be capable of unlearning in practice. The MIA scores decrease as K increases, suggesting that more information is unlearned when more rewinding occurs, even without noise disguising the output of the unlearning algorithm. A potential future direction of this research is empirically comparing this algorithm and other non-certified unlearning algorithms under more sophisticated MIAs.

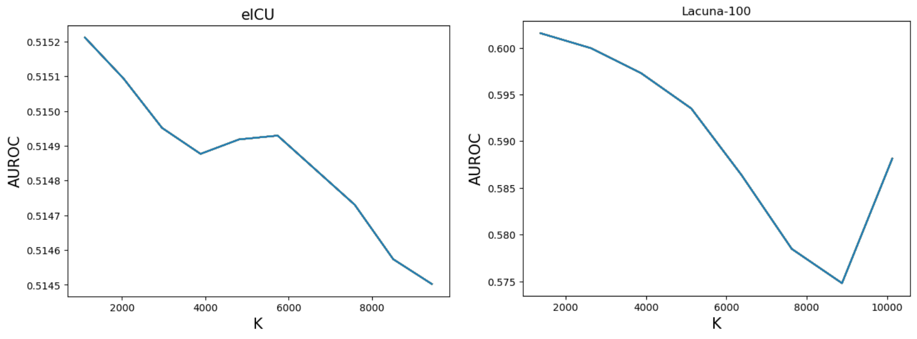


Figure 3. MIA scores after noiseless N2D unlearning. These results are averaged over 5 seeds.