Descent-to-Delete: Gradient-Based Methods for Machine Unlearning

Seth Neel SETHNEEL93@GMAIL.COM
Aaron Roth AAROTH@CIS.UPENN.EDU
Saeed Sharifi-Malvajerdi SAEEDSH@WHARTON.UPENN.EDU

Editors: Vitaly Feldman, Katrina Ligett and Sivan Sabato

Abstract

We study the data deletion problem for convex models. By leveraging techniques from convex optimization and reservoir sampling, we give the first data deletion algorithms that are able to handle an arbitrarily long sequence of adversarial updates while promising both per-deletion run-time and steady-state error that do not grow with the length of the update sequence. We also introduce several new conceptual distinctions: for example, we can ask that after a deletion, the entire state maintained by the optimization algorithm is statistically indistinguishable from the state that would have resulted had we retrained, or we can ask for the weaker condition that only the *observable output* is statistically indistinguishable from the observable output that would have resulted from retraining. We are able to give more efficient deletion algorithms under this weaker deletion criterion.

1. Introduction

Users voluntarily provide huge amounts of personal data to online services, such as Facebook, Google, and Amazon, in exchange for useful services. But a basic principle of data autonomy asserts that users should be able to revoke access to their data if they no longer find the exchange of data for services worthwhile. Indeed, each of these organizations provides a way for users to request that their data be deleted. This is related to, although distinct from the "Right to be Forgotten" from the European Union's General Data Protection Act (GPDR). The Right to be Forgotten entails the right for users, in certain circumstances, to request that negative information *concerning* them to be removed. Like basic data autonomy, it sometimes obligates companies to delete data.

But what does it mean to delete data? Typically, user data does not sit siloed in a database, but rather is used to produce derivatives such as predictive models. Deleting a user's data from a database may prevent it from influencing the training of future models, but does not remove the influence of a user's data on existing models — and that influence may be significant. For example, it is possible to extract information about specific data points used for training from models that have been trained in standard ways (Shokri et al., 2017). So deleting a user's data naively, by simply removing it from a database, may not accomplish much: what we really want is to remove (or at least rigorously limit) the *influence* that an individual's data has on the behavior of any part of the system.

How should we accomplish this? We could *retrain* all predictive models from scratch every time a user requests that their data be removed, but this would entail an enormous computational cost. Ginart et al. (2019) propose a compelling alternative: full retraining is unnecessary if we can design a deletion operation that produces a (distribution of) model output(s) that is statistically indistinguishable from the (distribution of) model output(s) that would have arisen from full retraining. Ginart et al. (2019) also propose an approximate notion of deletion that uses a differential-privacy like measure of "approximate" statistical indistinguishability that we adopt in this work.

1.1. Our Results and Techniques

In this paper, we consider *convex* models that are trained to some specified accuracy, and then are deployed while a sequence of requests arrive to delete (or add) additional data points. The deletion or addition must happen immediately, before the next request comes in, using only a fixed running time (which we measure in terms of gradient computations) per update. We require that the distribution on output models be (ϵ, δ)-indistinguishable from the distribution on output models that would result from full retraining (see Section 2 for the precise definition: this is a notion of approximate statistical indistinguishability from the differential privacy literature). In a departure from prior work, we make the distinction between whether the entire *internal state* of the algorithm must be indistinguishable from full retraining, or whether we only require statistical indistinguishability with respect to the *observable outputs* of the algorithms. If we require indistinguishability with respect to the full internal state, we call these update or *unlearning* algorithms *perfect*. This is similar to the distinction made in the differential privacy literature, which typically only requires indistinguishability for the *outputs* of private algorithms, but which has a strengthening (called *pan privacy* Dwork et al. (2010); Amin et al. (2019)) which also requires that the internal state satisfy statistical indistinguishability. We remark that while unlearning algorithms that are allowed to maintain a "secret state" that need not satisfy the data deletion notion require additional trust in the *security* of the training system, this is orthogonal to *privacy*. Indeed, Chen et al. (2020) show that even without secret state, algorithms satisfying standard deletion guarantees can exacerbate membership inference attacks if the attacker can observe the model both before and after a deletion (because standard deletion guarantees promise nothing about what can be learned about an individual from two model outputs). In contrast, although some of our unlearning algorithms maintain a secret state that does not satisfy the statistical indistinguishability property, our model outputs themselves satisfy (ϵ, δ)-differential privacy. This in particular prevents membership inference attacks from observers who can observe a small number of output models, so long as they cannot observe the secret state. All prior work has focused on perfect unlearning.

We introduce another novel distinction between *strong* unlearning algorithms and *weak* unlearning algorithms. For an unlearning algorithm to be *strong*, we require that for a fixed accuracy target, the run-time of the update operation be constant (or at most logarithmic) in the length of the update sequence. A weak unlearning algorithm may have run-time per update (or equivalently, error) that grows polynomially with the length of the update sequence. All prior work has given weak unlearning algorithms.

We give two sets of results. The first, which operates under the most permissive set of assumptions, is a simple family of gradient descent algorithms. After each addition or deletion request, the update algorithm starts from the previous model, and performs a small number of gradient descent updates — sufficient to guarantee that the model parameter is boundedly close to the *optimal* model parameter in Euclidean distance. It then perturbs the model parameter with Gaussian noise of sufficient magnitude to guarantee (ϵ, δ)-indistinguishability with respect to anything within a small neighborhood of the optimal model. We prove that this simple approach yields a strong, perfect unlearning algorithm for loss functions that are strongly convex and smooth. Without the strong convexity assumption, we can still derive strong unlearning algorithms, but ones which must maintain secret state. We can further improve our accuracy guarantees if we are willing to settle for weak unlearning algorithms. The per-round computation budget and the achievable steady state accuracy can be smoothly traded off against one another.

Our second algorithm improves over the straightforward approach above (under slightly stronger regularity assumptions) when the data dimension is sufficiently large. It first takes a bootstrap sample from the underlying dataset, and then randomly partitions it into K parts. The initial training algorithm separately and independently optimizes the loss function on each part, and then averages the parameter vector from each part, before finally releasing the perturbed average. Zhang et al. (2012) analyzed this algorithm (absent the final perturbation) and proved accuracy bounds with respect to the underlying distribution (which for us is the dataset from which we draw the bootstrap sample). Our update operation involves first using a variant of *reservoir sampling* that maintains the property that the union of the partitions continue to be distributed as independent samples drawn with replacement from our current dataset. We then use the simple gradient based update algorithms from our first set of results to update the parameters *only from the partitions that have been modified by the addition or deletion*. Because each of these partitions contains only a fraction of the dataset, we can use our fixed gradient computation budget to perform more iterations of gradient descent on these affected partitions. Because we have maintained the marginal distributions on partition elements via our reservoir sampling step, the overall accuracy analysis of Zhang et al. (2012) carries over even after an arbitrary sequence of updates. This is also crucial for our statistical indistinguishability guarantee to hold. The result is a strong unlearning algorithm that yields an improved tradeoff between per-round run-time and steady state accuracy for sufficiently high dimensional data.

1.2. Related Work

At a high level, our work differs from prior work in several ways. We call deletion algorithms that do not maintain secret state *perfect*. All prior work focuses on perfect deletion algorithms, but we give improved bounds for several problems by allowing our algorithms to maintain secret state. Second, we allow arbitrary sequences of updates, which can include additions and deletions (rather than just deletions). Finally, we distinguish between weak and strong unlearning algorithms, and give the first strong unlearning algorithms.

Cao and Yang (2015) first considered the problem of efficiently deleting data from a trained model under a deterministic notion of deletion, and coined the term "machine unlearning". They gave efficient deletion methods for certain statistical query algorithms — but in general, their methods (or indeed, any deterministic notion of deletion) can apply to only very structured problems. Ginart et al. (2019) gave the first definition of data deletion that can apply to randomized algorithms, in terms of statistical indistinguishability. We adopt the approximate deletion notion they introduced, which is itself based on differential privacy (Dwork et al., 2006; Dwork and Roth, 2014). Ginart et al. (2019) gave a deletion algorithm for the k-means problem. Their algorithm is a *weak* deletion algorithm in our terminology, because their (amortized) running time per update scales linearly with the number of updates.

Guo et al. (2019) gave deletion algorithms for linear and logistic regression, using the same notion of approximate statistical indistinguishability that we use. Their algorithm is similar to our first algorithm: it performs a convex optimization step, followed by a Gaussian perturbation. They use a second order update (a Newton step) rather than first order updates as we do, and their algorithm yields error that grows linearly with the number of updates, and so is a weak deletion algorithm. Izzo et al. (2020) focus on linear regression and show how to improve the run-time per deletion of the algorithm given in Guo et al. (2019) from quadratic to linear in the dimension.

Our main result leverages a distributed optimization algorithm that partitions the data, separately optimizes on each partition, and then averages the parameters, analyzed by Zhang et al. (2012). Optimizing separately on different partitions of the data, and then aggregating the results is also a well known general technique in differential privacy known as "Subsample and Aggregate" (Nissim et al., 2007) which has found applications in private learning (Papernot et al., 2016). Bourtoule et al. (2019) use a similar technique in the context of machine unlearning that they call "SISA" (Sharded, Isolated, Sliced, Aggregated). Their goal is more ambitious (to perform deletion for non-convex models), but they have a weaker deletion criterion (that it simply be *possible* that the model arrived at after deletion could have arisen from the retraining process), and they give no error guarantees. Their algorithm involves full retraining on the affected partitions, a different aggregation function, no randomization, and does not include the reservoir sampling step that is crucial to our stronger indistinguishability guarantees. This distributed optimization algorithm also bears similarity to the well-known *FederatedAveraging* algorithm of McMahan et al. (2016) used for deep learning in the federated setting.

Chen et al. (2020) observe that deterministic deletion procedures such as SISA (Bourtoule et al., 2019) can exacerbate privacy problems when an attacker can observe both the model before and after the deletion of a particular user's data point, and show how to perform membership inference attacks against SISA in this setting. Our method leverages techniques from differential privacy, and so in addition to being an (ϵ, δ)-deletion algorithm, a view of the two outputs of our algorithm before and after a deletion is $(2\epsilon, 2\delta)$-differentially private, which precludes non-trivial membership inference for reasonable values of ϵ and δ. This follows because our *deletion* algorithm is randomized: procedures such as the one from Guo et al. (2019) which have randomized training procedure but deterministic deletion procedure do not share this property.

1.3. Summary of Results

In Table 1, we state bounds for all our unlearning algorithms, and (in the 2nd column) the assumptions that they require. The 3rd column of the table states whether our algorithms are weak or strong update algorithms (whether or not their runtimes grow polynomially with the length of the update sequence). The 4th column states the steady-state accuracy (α in Definition 6) of the algorithm as a function of the desired run time $\mathcal{I}$ of the first update (each algorithm has a budget of $n\mathcal{I}$ gradient computations per update). The 5th column lists the run-time of the i'th update. The 6th column measures the run-time of the baseline approach that would *fully retrain* the model after each update, to the accuracy achieved by our algorithms in the 4th column. Most of these guarantees are for algorithms that maintain a secret state. But for strongly convex and smooth functions we can obtain a perfect unlearning algorithm (i.e. one that satisfies the indistinguishability guarantee not just with respect to observable outputs, but with respect to the entire saved state) with the same asymptotic accuracy/runtime tradeoff, so long as the per-update run-time is at least logarithmic in the dimension. For non strongly convex functions, our techniques do not appear to be able to give perfect unlearning algorithms for non-trivial parameters; this is an intriguing direction for future work.

Our "Distributed PGD" algorithm is somewhat more complex (see Section 4), but has the advantage that it obtains improved accuracy/run-time tradeoffs for sufficiently high dimensional data. It divides the same gradient computation budget $n\mathcal{I}$ into different numbers of iterations on different parts of the dataset. See Remark 15 for the exact conditions on when it yields an improvement over our simpler algorithms.

summary of tradeoffs for (ϵ, δ)-unlearning					
method	loss function properties	unlearning	accuracy	iterations for ith update	baseline iterations
PGD	SC, smooth	strong (Thm. 9)	$\frac{de^{-\mathcal{I}}}{\epsilon^2 n^2}$	$\mathcal{I}$	$\mathcal{I} + \log\left(\frac{\epsilon n}{\sqrt{d}}\right)$
	SC, smooth	strong, perfect (Thm. 28)	$\frac{de^{-\mathcal{I}}}{\epsilon^2 n^2}$	$\log i \cdot \mathcal{I}$ $\mathcal{I} \geq \log(d/\epsilon)$	$\mathcal{I} + \log\left(\frac{\epsilon n}{\sqrt{d}}\right)$
Regularized PGD	C, smooth	strong (Thm. 10)	$\left(\frac{\sqrt{d}}{\epsilon n \mathcal{I}}\right)^{\frac{2}{5}}$	$\mathcal{I}$	$\left(\frac{\epsilon n \mathcal{I}}{\sqrt{d}}\right)^{\frac{2}{5}}$
	C, smooth	weak (Thm. 30)	$\sqrt{\frac{\sqrt{d}}{\epsilon n \sqrt{\mathcal{I}}}}$	$i^2 \cdot \mathcal{I}$	$\sqrt{\frac{\epsilon n \sqrt{\mathcal{I}}}{\sqrt{d}}}$
Distributed PGD	SC, smooth, Lipschitz and bounded Hessian	strong (Thm. 14)	$\frac{de^{-\mathcal{I}n^{\frac{4-3\xi}{2}}}}{\epsilon^2 n^2} + \frac{1}{n^\xi}$	$\log i \cdot \mathcal{I}$	$\min\left\{\log n, \mathcal{I} n^{\frac{4-3\xi}{2}} + \log\left(\frac{\epsilon n}{\sqrt{d}}\right)\right\}$

Table 1: (S)C: (strongly) convex, n: training dataset size, d: dimension, $\xi \in [1, 4/3]$ is a parameter.

2. Model and Preliminaries

We write $\mathcal{Z}$ to denote the data domain. A dataset $\mathcal{D}$ is a multi-set of elements from $\mathcal{Z}$. Datasets can be modified by *updates* which are requests to either add or remove one element from the dataset.

Definition 1 (Update) *An update u is a pair $(z, \bullet)$ where $z \in \mathcal{Z}$ is a data point and $\bullet \in \mathcal{T} = \{'\texttt{add}', '\texttt{delete}'\}$ determines the type of the update. An update sequence $\mathcal{U}$ is a sequence $(u_1, u_2, \ldots)$ where $u_i \in \mathcal{Z} \times \mathcal{T}$ for all i. Given a dataset $\mathcal{D}$ and an update $u = (z, \bullet)$, the update operation is defined as follows.*

$$\mathcal{D} \circ u \triangleq \begin{cases} \mathcal{D} \cup \{z\} & \text{if } \bullet = '\texttt{add}' \\ \mathcal{D} \setminus \{z\} & \text{if } \bullet = '\texttt{delete}' \end{cases}$$

We use Θ to denote the space of models. In our setting, a *learning* or *training* algorithm is a mapping $\mathcal{A} : \mathcal{Z}^* \to \Theta$ that maps datasets to models. An *unlearning* or *update* algorithm for $\mathcal{A}$ is a mapping $\mathcal{R}_\mathcal{A} : \mathcal{Z}^* \times (\mathcal{Z} \times \mathcal{T}) \times \Theta \to \Theta$ that takes as input a dataset accompanied by a single update, and a model, and outputs an updated model. Some of our update algorithms will also take as input auxiliary information, that we elide here but will be clear from context. The output of the unlearning algorithm itself will not be made public: before any model is made public, it must pass through a *publishing* function. A *publishing function* is a mapping $f_{\text{publish}} : \Theta \to \Theta$ that maps a (secret) model to the model that will be made publicly available. Our unlearning guarantee will informally require that there should be no way to distinguish whether the *published* model resulted from full retraining, or an arbitrary sequence of updates via the unlearning algorithm. Depending on whether we demand *perfect* unlearning or not (to be defined shortly), we may save either the (secret) output of the unlearning algorithm as persistent state, or save only the (public) output of the publishing function.

Definition 2 ($\mathcal{D}_i, \theta_i, \hat{\theta}_i, \tilde{\theta}_i$) *Fix any $(\mathcal{A}, \mathcal{R}_\mathcal{A})$ of learning and unlearning algorithms, any publishing function $f_{publish}$, any dataset $\mathcal{D}$, and any update sequence $\mathcal{U} = (u_1, u_2, \ldots)$. We write $\mathcal{D}_0 = \mathcal{D}$ and for any $i \geq 1$, $\mathcal{D}_i = \mathcal{D}_{i-1} \circ u_i$. For any $i \geq 1$, we write θ_i for the model input to the unlearning algorithm $\mathcal{R}_\mathcal{A}$ on round i. We write $\hat{\theta}_0 = \mathcal{A}(\mathcal{D}_0)$, and for any $i \geq 1$, $\hat{\theta}_i = \mathcal{R}_\mathcal{A}(\mathcal{D}_{i-1}, u_i, \theta_i)$. For any $i \geq 0$, we define $\tilde{\theta}_i = f_{publish}(\hat{\theta}_i)$. In other words, whenever $\mathcal{A}$, $\mathcal{R}_\mathcal{A}$, $f_{publish}$, $\mathcal{D}$, and $\mathcal{U}$ are clear from context, we write $\{\mathcal{D}_i\}_{i\geq 0}$ to represent the sequence of updated datasets, $\{\theta_i\}_{i\geq 1}$ for the sequence of input models to $\mathcal{R}_\mathcal{A}$, $\{\hat{\theta}_i\}_{i\geq 0}$ to denote the (secret) output models of $\mathcal{A}$ and $\mathcal{R}_\mathcal{A}$, and $\{\tilde{\theta}_i\}_{i\geq 0}$ to denote their corresponding sequence of published models.*

Our (ϵ, δ)-unlearning notion is similar to the deletion notion proposed in Ginart et al. (2019) but generalizes it to an update sequence consisting of both additions and deletions.

Definition 3 ((ϵ, δ)**-indistinguishability)** *Let X and Y be random variables over some domain Ω. We say X and Y are (ϵ, δ)-indistinguishable and write $X \overset{\epsilon,\delta}{\approx} Y$, if for all $S \subseteq \Omega$,*

$$\Pr[X \in S] \leq e^\epsilon \Pr[Y \in S] + \delta, \quad \Pr[Y \in S] \leq e^\epsilon \Pr[X \in S] + \delta$$

Definition 4 ((ϵ, δ)**-unlearning)** *We say that $\mathcal{R}_\mathcal{A}$ is an (ϵ, δ)-unlearning algorithm for $\mathcal{A}$ with respect to a publishing function $f_{publish}$, if for all data sets $\mathcal{D}$ and all update sequences $\mathcal{U} = (u_i)_i$, the following condition holds. For every update step $i \geq 1$, for $\theta_i = \hat{\theta}_{i-1}$*

$$f_{publish}(\mathcal{R}_\mathcal{A}(\mathcal{D}_{i-1}, u_i, \theta_i)) \overset{\epsilon,\delta}{\approx} f_{publish}(\mathcal{A}(\mathcal{D}_i))$$

If the above condition holds for $\theta_i = \tilde{\theta}_{i-1}$, $\mathcal{R}_\mathcal{A}$ is an (ϵ, δ)-perfect unlearning algorithm for $\mathcal{A}$.

Remark 5 *Observe that an unlearning algorithm takes as input the model output by the previous round's unlearning algorithm, whereas a* perfect *unlearning algorithm takes as input the model output by the previous round's publishing algorithm. Since we require that the* published *outputs satisfy (ϵ, δ)-indistinguishability, this means that unlearning algorithms may need to maintain* secret state *that does not satisfy the indistinguishability guarantee, but that perfect unlearning algorithms do not need to.*

2.1. Learning Framework: ERM

We consider an *Empirical Risk Minimization (ERM)* setting in this paper where models are (parameter) vectors in d-dimensional space $\mathbb{R}^d$ equipped with the (Euclidean) ℓ_2-norm which will be denoted by $\|\cdot\|_2$. Let $\Theta \subseteq \mathbb{R}^d$ be a convex and closed subset of $\mathbb{R}^d$, and let $D = \sup_{\theta,\theta' \in \Theta} \|\theta - \theta'\|_2$ be the *diameter* of Θ. We denote a loss function by a mapping $f : \Theta \times \mathcal{Z} \to \mathbb{R}$ that takes as input a parameter $\theta \in \Theta$ and a data point $z \in \mathcal{Z}$, and outputs the loss of θ on z, $f(\theta, z)$ — which we may also denote by $f_z(\theta)$. Given a dataset $\mathcal{D} = \{z_i\}_{i=1}^n \in \mathcal{Z}^n$, with slight abuse of notation, let $f_\mathcal{D}(\theta)$ denote the empirical loss of θ on the dataset $\mathcal{D}$: $f_\mathcal{D}(\theta) \triangleq \frac{1}{n}\sum_{i=1}^n f_{z_i}(\theta)$.

Definition 6 ((α, β)**-accuracy)** *A pair $(\mathcal{A}, \mathcal{R}_\mathcal{A})$ of learning and unlearning algorithms is (α, β)-accurate with respect to a publishing function $f_{publish}$, if for every dataset $\mathcal{D}$ and every update sequence $\mathcal{U}$, the following condition holds: $\forall\, i \geq 0$, $\Pr\left[f_{\mathcal{D}_i}(\tilde{\theta}_i) - \min_{\theta\in\Theta} f_{\mathcal{D}_i}(\theta) > \alpha\right] < \beta$.*

Definition 7 (strong vs. weak unlearning) *Fix any pair $(\mathcal{A}, \mathcal{R}_{\mathcal{A}})$ of learning and unlearning algorithms that satisfy (α, β)-accuracy with respect to some publishing function $f_{publish}$. Let C_i represent the overall computational cost of the unlearning algorithm at step i of the update. We say $\mathcal{R}_{\mathcal{A}}$ is a "strong" unlearning algorithm for $\mathcal{A}$ if (1) α and β are independent of the length of the update sequence, and (2) For every $i \geq 1$, $C_i/C_1 = \mathcal{O}(\log(i))$, i.e., the computation cost of the unlearning algorithm must grow at most logarithmically with i. If (1) holds and $\forall i \geq 1$, $C_i/C_1 = \Omega(poly(i))$, we say $\mathcal{R}_{\mathcal{A}}$ is a "weak" unlearning algorithm for $\mathcal{A}$.*

We remark that we have defined update sequences as if they are *non-adaptively chosen*, but that our basic algorithms in Section 3 have guarantees also for adaptively chosen update sequences.

Additional details of this section appear in Appendix A and Appendix B.

2.2. Strong Convexity and Sensitivity

Throughout the paper we will leverage the fact that the optimizers of strongly convex functions have low *sensitivity* to individual data points. We state this fact below and defer its proof to Appendix D.

Lemma 8 (Sensitivity) *Suppose for any $z \in \mathcal{Z}$, f_z is L-Lipschitz and m-strongly convex. For any dataset $\mathcal{D}$, let $\theta^*_{\mathcal{D}} \triangleq \operatorname{argmin}_{\theta \in \Theta} f_{\mathcal{D}}(\theta)$. We have that for any integer n, any data set $\mathcal{D}$ of size n, and any update u, $\|\theta^*_{\mathcal{D}} - \theta^*_{\mathcal{D} \circ u}\|_2 \leq \frac{2L}{mn}$.*

3. Basic Perturbed Gradient Descent

A key building block for our main result (and a simple and effective deletion scheme in its own right, that requires fewer assumptions than our main result) is *perturbed gradient descent*. The basic idea is as follows, for both the training algorithm and the deletion algorithm: we will perform gradient descent updates until we are guaranteed that we have found a $\hat{\theta}_t$ which is within Euclidean distance α of the optimizer, for some small α. Our publishing algorithm f_{publish} adds Gaussian noise scaled as a function of α to every coordinate. This guarantees (ϵ, δ)-indistinguishability with respect to any other parameter that is within distance α of the optimizer — and hence between the outcomes of full retraining and updating. Depending on whether we want a perfect deletion algorithm or not, we save either the perturbed or unperturbed parameter as our initialization point for the next update.

Our update algorithm will be the same as our training algorithm — except that it will be initialized at the learned parameter from the previous round, which will guarantee faster convergence. This is because — if we allow secret state — the initialization parameter will be within α of the optimizer before the update, and if f is strongly convex, within $\mathcal{O}(\alpha + \frac{1}{mn})$ of the optimal parameter after the update by the sensitivity Lemma 8. If we require a perfect deletion algorithm, we will necessarily need to start further from the optimizer, because our saved state will have been additionally perturbed with Gaussian noise. Here we leverage the fact that gradient descent converges quickly when its initialization point is near the optimal solution.

This algorithm relies crucially on leveraging strong convexity, which guarantees us that updates only change the empirical risk minimizer by a small amounts *in parameter space*. In Section 3.1 we solve the non-strongly-convex case by adding a strongly convex regularizer.

We parameterize our results by the computational cost of the update operations, and we can trade off run-time for accuracy. We measure computational cost by gradient computations. In this section, we parameterize our strong unlearning algorithms by the number of iterations $\mathcal{I}$ that they run for,

Algorithm 1 $\mathcal{A}$: **Learning** for Perturbed Gradient Descent

Input: dataset $\mathcal{D}$
Initialize $\theta'_0 \in \Theta$
for $t = 1, 2, \ldots T$ **do**
  $\theta'_t = \text{Proj}_\Theta\left(\theta'_{t-1} - \eta_t \nabla f_{\mathcal{D}}(\theta'_{t-1})\right)$
Output: $\hat{\theta}_0 = \theta'_T$; `// Secret output`

Algorithm 2 $\mathcal{R}_{\mathcal{A}}$: ith **Unlearning** for Perturbed Gradient Descent

Input: dataset $\mathcal{D}_{i-1}$, update u_i, model θ_i
Update dataset $\mathcal{D}_i = \mathcal{D}_{i-1} \circ u_i$
Initialize $\theta'_0 = \theta_i$
for $t = 1, 2, \ldots T_i$ **do**
  $\theta'_t = \text{Proj}_\Theta\left(\theta'_{t-1} - \eta_t \nabla f_{\mathcal{D}_i}(\theta'_{t-1})\right)$
Output: $\hat{\theta}_i = \theta'_{T_i}$; `// Secret output`

which corresponds to a budget of $\approx n\mathcal{I}$ gradient computations per update. For weak unlearning algorithms, this is the number of iterations they run for at their first update.

Theorem 9 (Accuracy, Unlearning, and Computation Tradeoffs) *Suppose for all $z \in \mathcal{Z}$, the loss function f_z is m-strongly convex, L-Lipschitz, and M-smooth. Define $\gamma \triangleq (M-m)/(M+m)$ and $\eta \triangleq 2/(M+m)$. Let the learning algorithm $\mathcal{A}$ (Algorithm 1) run with $\eta_t = \eta$ and $T \geq \mathcal{I} + \log(\frac{Dmn}{2L})/\log(1/\gamma)$ where n is the size of the input dataset, and let the unlearning algorithm $\mathcal{R}_{\mathcal{A}}$ (Algorithm 2) run with input models $\theta_i \equiv \hat{\theta}_{i-1}$ and $\eta_t = \eta$ and $T_i = \mathcal{I}$ iterations, for all $i \geq 1$. Let the unlearning parameters ϵ and δ be such that $\epsilon = \mathcal{O}(\log(1/\delta))$, and let*

$$\sigma = \frac{4\sqrt{2}L\gamma^{\mathcal{I}}}{mn\left(1-\gamma^{\mathcal{I}}\right)\left(\sqrt{\log(1/\delta)+\epsilon} - \sqrt{\log(1/\delta)}\right)}$$

in $f_{publish}$ (Algorithm 3). We have that $\mathcal{R}_{\mathcal{A}}$ is a strong (ϵ, δ)-unlearning algorithm for $\mathcal{A}$ with respect to $f_{publish}$. Furthermore, for any β, $(\mathcal{A}, \mathcal{R}_{\mathcal{A}})$ is (α, β)-accurate with respect to $f_{publish}$ where

$$\alpha = \mathcal{O}\left(\frac{ML^2\gamma^{2\mathcal{I}} d \log(1/\delta) \log^2(d/\beta)}{\left(1-\gamma^{\mathcal{I}}\right)^2 m^2 \epsilon^2 n^2}\right)$$

A formal proof of Theorem 9 can be found in Appendix E. We note that the same algorithm can be analyzed as a *perfect* unlearning algorithm (i.e. without maintaining secret state). It obtains the same asymptotic tradeoff between running time and accuracy, under the condition that the per-update run-time is at least logarithmic in the relevant parameters. Intuitively, this run-time lower bound is required so that the update algorithm can "recover" from the effect of the added noise in previous rounds. See Appendix F for details.

3.1. Convex Loss: Regularized Perturbed GD

If our loss function is not strongly convex, we can regularize it to enforce strong convexity, and apply our algorithms to the regularized loss function. When we do this, we must manage a basic

Algorithm 3 f_{publish}: Publishing function

Input: $\hat{\theta} \in \mathbb{R}^d$
Draw $Z \sim \mathcal{N}\left(0, \sigma^2 \mathbb{I}_d\right)$
Output: $\tilde{\theta} = \hat{\theta} + Z$; `// Public output`

tradeoff: the more aggressively we regularize the loss function, the less sensitive it will be, and so the less noise we will need to add in our f_{publish} routine. This reduced noise will *increase* accuracy. On the other hand, the more aggressively we regularize, the less well the optimizer of the regularized loss function will optimize the original loss function of interest, which will *decrease* accuracy. More aggressive regularization will also degrade the Lipschitz/smoothness guarantees of the loss function. We choose our regularization parameter carefully to trade off these various sources of error.

Suppose in this section, without loss of generality, that Θ contains the origin: $0 \in \Theta$. This will imply that $\sup_{\theta \in \Theta} \|\theta\|_2 \leq D$ where D is the diameter of Θ, as before. Our strategy is to regularize f so as to make it strongly convex, and have our learning and unlearning algorithms run on the regularized version of f: $g_z(\theta) \triangleq f_z(\theta) + \frac{m}{2}\|\theta\|_2^2$ for some $m > 0$.

Theorem 10 (Accuracy, Unlearning, and Computation Tradeoffs) *Suppose for all $z \in \mathcal{Z}$, the loss function f_z is convex, L-Lipschitz, and M-smooth, and let g_z be defined as above for some m specified later. Define $\gamma \triangleq M/(M+2m)$ and $\eta \triangleq 2/(M+2m)$. Let the learning algorithm $\mathcal{A}$ (Algorithm 1) run on the regularized g with $\eta_t = \eta$ and $T \geq \mathcal{I} + \log(\frac{Dmn}{2L})/\log(1/\gamma)$ where n is the size of the input dataset, and let the unlearning algorithm $\mathcal{R}_{\mathcal{A}}$ (Algorithm 2) run on the regularized g with input models $\theta_i \equiv \hat{\theta}_{i-1}$ and $\eta_t = \eta$ and $T_i = \mathcal{I}$ iterations for all $i \geq 1$. Let the unlearning parameters ϵ and δ be such that $\epsilon = \mathcal{O}(\log(1/\delta))$, and let*

$$\sigma = \frac{4\sqrt{2}\left(L + mD\right)\gamma^{\mathcal{I}}}{mn\left(1-\gamma^{\mathcal{I}}\right)\left(\sqrt{\log\left(1/\delta\right)+\epsilon} - \sqrt{\log\left(1/\delta\right)}\right)}, \quad m = \left(\frac{LM^{\frac{3}{2}}\sqrt{d\log\left(1/\delta\right)}}{D\epsilon n\mathcal{I}}\right)^{\frac{2}{5}}$$

We have that $\mathcal{R}_{\mathcal{A}}$ is a strong (ϵ, δ)-unlearning algorithm for $\mathcal{A}$ with respect to f_{publish}. Furthermore, for any β, $(\mathcal{A}, \mathcal{R}_{\mathcal{A}})$ is (α, β)-accurate with respect to f_{publish} where

$$\alpha = \mathcal{O}\left(\left(\frac{M^{\frac{3}{2}}LD^4\sqrt{d\log\left(1/\delta\right)}}{\epsilon n\mathcal{I}}\right)^{\frac{2}{5}}\log^2\left(d/\beta\right)\right) + \mathcal{O}\left(n^{-\frac{4}{5}}\right) + \mathcal{O}\left(n^{-\frac{6}{5}}\right)$$

See Appendix G for the proof. If our goal is to satisfy only weak unlearning (i.e. to allow run-time to grow with the length of the update sequence i), we can obtain error bounds that have a better dependence on n. Details are in Appendix H.

4. Perturbed Distributed Descent

Our next algorithm obtains additional running time improvements for sufficiently high dimensional data. The basic idea is as follows: we randomly partition the dataset into K parts, separately optimize to find a model that approximates the empirical risk minimizer on each part, and then take the average of each of the K models. Zhang et al. (2012) analyze this algorithm and show that its

out of sample guarantees match the out of sample guarantees of non-distributed gradient descent, whenever $K \leq \sqrt{n}$. For us, this algorithm has a key advantage: the element involved in an update will only appear in a small number of the partitions, and we only need to update the parameters corresponding to those partitions. Our algorithm will improve over basic gradient descent because those partitions are smaller in size than the entire dataset by a factor of K, and hence our run-time budget of $n\mathcal{I}$ gradient computations will allow us to perform more than $\mathcal{I}$ gradient descent operations per modified partition. We provide deletion guarantees by using a publishing function that adds noise to the average of the K parameters.

There are several difficulties that we must overcome. Primary among these is that the analysis of Zhang et al. (2012) provides out of sample guarantees for a dataset that is drawn $i.i.d.$ from some fixed distribution. In our case (because our dataset results from an arbitrary and possibly adversarial sequence of additions and deletions), there is no distribution from which the dataset is drawn. To deal with this, our initial training algorithm does not directly partition the dataset, but instead draws a *bootstrap* sample (i.e. a sample with replacement) from the empirical distribution defined by the dataset, so that the "out of sample" guarantees of Zhang et al. (2012) correspond to empirical risk bounds in our case. Because the accuracy analysis depends on this distributional property, as updates come in, before we use gradient descent to update the models corresponding to the appropriate partitions, we must apply a form of reservoir sampling to guarantee that each partition continues to be distributed as a set of samples drawn $i.i.d.$ from the empirical distribution defined by the *current* dataset (i.e. after the update). This is also crucial to our unlearning guarantee. Finally, the basic instantiation of this algorithm only gives guarantees on the *expected* error of the learned model Zhang et al. (2012), and we want high probability guarantees. To achieve these, we run $C = \mathcal{O}(\log(1/\beta))$ copies of the algorithm in parallel, and at every round, only *publish* a noisy version of the parameter achieving the lowest loss among all C candidates. We now go into more detail. To facilitate the technical development in this section, we introduce some notation:

Definition 11 *Fix any update round $i \geq 0$. In this section we use $\boldsymbol{\mathcal{S}}_i = (\mathcal{S}_{ij})_{j=1}^K$ for the partitioned dataset at round i. We use $\mathcal{S}_i$ (unbold) to denote the union of partitions in $\boldsymbol{\mathcal{S}}_i$ and $\mathcal{D}_i$ for the unique data points in $\mathcal{S}_i$ (i..e $\mathcal{D}_i$ removes the duplicates in $\mathcal{S}_i$ which results from our sampling scheme). We use $\hat{\boldsymbol{\theta}}_i = (\hat{\theta}_{ij})_{j=1}^K$ for the learned parameters in each partition. $\tilde{\theta}_i = f_{publish}(\hat{\boldsymbol{\theta}}_i)$ represents the published model of round i. In this section, the unlearning algorithm for update i takes as input the partitioned dataset of previous round $\boldsymbol{\mathcal{S}}_{i-1}$, an update u_i, and the learned models of previous round $\hat{\boldsymbol{\theta}}_{i-1}$, and outputs the updated models $\hat{\boldsymbol{\theta}}_i$ and the updated datasets $\boldsymbol{\mathcal{S}}_i$ for use in the next update.*

Throughout we denote the distribution on datasets of size B sampled *with replacement* from $\mathcal{D}$ by $\mathcal{P}^B(\mathcal{D})$. We need to maintain the condition that the marginal distribution of the sampled dataset $\mathcal{S}_i$ at round i is $\mathcal{P}^B(\mathcal{D}_i)$. To do this, at each update, we iteratively update each partition using a technique called reservoir sampling with replacement (that we need to extend to handle both additions and deletions). The algorithm $\mathcal{S}_{rep}^B$ is detailed in Algorithm 5.

Lemma 12 *Fix any training dataset $\mathcal{D}$ and any non-adaptively chosen update sequence $\mathcal{U}$. Let $\mathcal{S}_0 \sim \mathcal{P}^B(\mathcal{D})$ (as in the learning algorithm) and for every $i \geq 1$, $\mathcal{S}_i \sim \mathcal{S}_{rep}^B(\mathcal{S}_{i-1}, u_i)$ (as in the unlearning algorithm). We have that for all $i \geq 0$: $\mathcal{S}_i \sim \mathcal{P}^B(\mathcal{D}_i)$.*

Lemma 13 shows that the reservoir sampling modifies at most $s_i = \tilde{\mathcal{O}}(B/n)$ data points; hence, at most s_i partitions containing a modified data point. Thus we can divide our budget of nT_i gradient computations at round i, into $(KnT_i)/(Bs_i)$ gradient computations per modified partition.

Algorithm 4 $\mathcal{A}$: **Learning** for Perturbed Distributed Gradient Descent

Input: dataset $\mathcal{D}$
for $l = 1, 2, \ldots, C$ **do**
  Draw $\mathcal{S} \sim \mathcal{P}^B(\mathcal{D})$; `// Bootstrap` B `data points.`
  Partition $\mathcal{S}$ randomly into K equally-sized datasets: $\boldsymbol{\mathcal{S}}_{0,l} = (\mathcal{S}_j)_{j=1}^K$.
  for $j = 1, 2, \ldots, K$ **do**
    Initialize $\theta'_0 \in \Theta$
    for $t = 1, 2, \ldots T$ **do**
      $\theta'_t = \text{Proj}_\Theta\left(\theta'_{t-1} - \eta_t \nabla f_{\mathcal{S}_j}(\theta'_{t-1})\right)$
    $\hat{\theta}_j = \theta'_T$
  $\hat{\boldsymbol{\theta}}_{0,l} = (\hat{\theta}_j)_{j=1}^K$; `//` l`'th set of models.`
Call $f_{\text{publish}}(\hat{\boldsymbol{\theta}}_{0,l^*})$ where $l^* = \text{argmin}_l\, f_{\mathcal{D}}(\text{avg}(\hat{\boldsymbol{\theta}}_{0,l}))$; `// Publish the best model.`
Output: $\hat{\boldsymbol{\theta}}_0 = (\hat{\boldsymbol{\theta}}_{0,l})_{l=1}^C, \boldsymbol{\mathcal{S}}_0 = (\boldsymbol{\mathcal{S}}_{0,l})_{l=1}^C$; `// For use in first update.`

Algorithm 5 $\mathcal{S}^B_{rep}$: Reservoir Sampling with Replacement for ith update

Input: Subsample $\mathcal{S}_{i-1}$, update $u_i = (z_i, \bullet_i)$
$\mathcal{S}_i = \mathcal{S}_{i-1}$
if $\bullet_i =$ `'add'` **then**
  Draw $N \sim \text{Binomial}(B, n_i^{-1})$; `//` n_i`: size of` $\mathcal{D}_i$`.`
  Pick distinct indices $i_1, \ldots, i_N$ at random from $[B]$.
  for $k = 1, 2, \ldots, N$ **do**
    Replace z_{i_k} with z_i in $\mathcal{S}_i$.
else
  for $z_k \in \mathcal{S}_i : z_k = z_i$ **do**
    Replace z_k with $z \sim \mathcal{P}(\mathcal{D}_i)$ in $\mathcal{S}_i$; `//` $\mathcal{P}(\mathcal{D}_i)$`:` $\mathcal{D}_i$`'s empirical distribution`
Output: $\mathcal{S}_i$

Lemma 13 *Fix any training dataset $\mathcal{D}$ and any update sequence $\mathcal{U}$, and suppose $B \geq n$. Let s_i denote the number of data points modified by the update of round i, namely, u_i. In other words, $s_i = |\{z_l : z_l \in \mathcal{S}_i, z_l \notin \mathcal{S}_{i-1}\}|$. We have that for any update step i and any $\delta' \leq e^{-1}$, with probability at least $1 - \delta'$, $s_i \leq \frac{10B}{n} \log(1/\delta')$.*

We now state the accuracy and unlearning bounds for perturbed distributed gradient descent. The convergence analysis on each partition is similar to the analysis in the proof of Theorem 9, with the added complexity of handling the number of partitions updated at each round, and the number of duplicated points (that could possibly be removed) in each partition. In order to obtain accuracy bounds we need to leverage an accuracy bound for the averaged parameter in a distributed setting, which we take from Zhang et al. (2012) (see Appendix I and J for details and proofs of this section).

Theorem 14 (Accuracy, Unlearning, and Computation Tradeoffs) *Suppose for all $z \in \mathcal{Z}$, the loss function f_z is m-strongly convex, L-Lipschitz, M-smooth, and that its Hessian is G-Lipschitz and bounded by H (with respect to ℓ_2-operator norm of matrices). Define $\gamma \triangleq (M - m)/(M + m)$ and $\eta \triangleq 2/(M + m)$. Fix any $1 \leq \xi \leq 4/3$, and let $B = n^\xi$ and $K = \sqrt{B}$. Let the learning algorithm $\mathcal{A}$ (Algorithm 4) run with $\eta_t = \eta$ and T iterations on every partition, and*

Algorithm 6 $\mathcal{R}_{\mathcal{A}}$: ith **Unlearning** for Perturbed Distributed Gradient Descent

Input: datasets $\boldsymbol{\mathcal{S}}_{i-1} = (\boldsymbol{\mathcal{S}}_{i-1,l})_{l=1}^{C}$, update u_i, models $\hat{\boldsymbol{\theta}}_{i-1} = (\hat{\boldsymbol{\theta}}_{i-1,l})_{l=1}^{C}$
Update $\mathcal{D}_i = \mathcal{D}_{i-1} \circ u_i$.
for $l = 1, 2, \ldots, C$ **do**
  Draw $\boldsymbol{\mathcal{S}}_{i,l} \sim \mathcal{S}^{B}_{rep}(\mathcal{S}_{i-1,l}, u_i)$; `// Reservoir update + similar partition.`
  Let $(\mathcal{S}_{i,j})_{j=1}^{K} \equiv \boldsymbol{\mathcal{S}}_{i,l}$, $(\mathcal{S}_{i-1,j})_{j=1}^{K} \equiv \boldsymbol{\mathcal{S}}_{i-1,l}$, $(\hat{\theta}_{i-1,j})_{j=1}^{K} \equiv \hat{\boldsymbol{\theta}}_{i-1,l}$.
  Let `ind` $= \{j : \mathcal{S}_{i-1,j} \neq \mathcal{S}_{i,j}\}$; `// Modified partitions.`
  for $j = 1, 2, \ldots, K$ **do**
    if $j \in$ `ind` **then**
      Initialize $\theta'_0 = \hat{\theta}_{i-1,j}$
      for $t = 1, 2, \ldots, T = \frac{KnT_i}{B|\texttt{ind}|}$ **do**
        $\theta'_t = \text{Proj}_{\Theta}\left(\theta'_{t-1} - \eta_t \nabla f_{\mathcal{S}_{i,j}}(\theta'_{t-1})\right)$
      $\hat{\theta}_{i,j} = \theta'_T$
    else
      $\hat{\theta}_{i,j} = \hat{\theta}_{i-1,j}$
  $\hat{\boldsymbol{\theta}}_{i,l} = (\hat{\theta}_{i,j})_{j=1}^{K}$; `// l'th set of models.`
Call $f_{\text{publish}}(\hat{\boldsymbol{\theta}}_{i,l^*})$ where $l^* = \text{argmin}_l\, f_{\mathcal{D}_i}(\text{avg}(\hat{\boldsymbol{\theta}}_{i,l}))$; `// Publish the best model.`
Output: $\hat{\boldsymbol{\theta}}_i = (\hat{\boldsymbol{\theta}}_{i,l})_{l=1}^{C}, \boldsymbol{\mathcal{S}}_i = (\boldsymbol{\mathcal{S}}_{i,l})_{l=1}^{C}$; `// For use in next update.`

Algorithm 7 f_{publish}: publishing function

Input: $\hat{\boldsymbol{\theta}} = (\hat{\theta}_j)_{j=1}^{K}$
Draw $Z \sim \mathcal{N}\left(0, \sigma^2 \mathbb{I}_d\right)$
Output: $\tilde{\theta} = \text{avg}(\hat{\boldsymbol{\theta}}) + Z$; `// avg(·) averages input models.`

for any update $i \geq 1$, let the unlearning algorithm $\mathcal{R}_{\mathcal{A}}$ (Algorithm 6) run with $\eta_t = \eta$ and total T_i iterations per copy (i.e. total nT_i gradient computations per copy), where for any $\mathcal{I}$, $T \geq \mathcal{I}n^{\frac{4-3\xi}{2}} + \frac{\log\left(DmL^{-1}n^{\xi}(1+10\log(2/\delta))\right)}{\log(1/\gamma)}$, and $T_i = 10\log(2i/\delta)\left(\mathcal{I} + \frac{1}{n^{\frac{4-3\xi}{2}}} \cdot \frac{\log(1+10i\log(2i/\delta))}{\log(1/\gamma)}\right)$. Let the unlearning parameters ϵ and δ be such that $\epsilon = \mathcal{O}\left(\log\left(1/\delta\right)\right)$ and $\delta = \mathcal{O}(B^{-1})$, and let

$$\sigma = \frac{4\sqrt{2}L\gamma^{\mathcal{I}n^{\frac{4-3\xi}{2}}}}{mn\left(1-\gamma^{\mathcal{I}n^{\frac{4-3\xi}{2}}}\right)\left(\sqrt{\log\left(2/\delta\right)+\epsilon} - \sqrt{\log\left(2/\delta\right)}\right)}$$

in $f_{publish}$ (Algorithm 7). We have that $\mathcal{R}_{\mathcal{A}}$ is a strong (ϵ, δ)-unlearning algorithm for $\mathcal{A}$ with respect to $f_{publish}$. Furthermore, for any β, letting $C = \log\left(2/\beta\right)/\log 2$, we get that $(\mathcal{A}, \mathcal{R}_{\mathcal{A}})$ is (α, β)-accurate with respect to $f_{publish}$ where

$$\alpha = \mathcal{O}\left(\frac{ML^2\gamma^{2\mathcal{I}n^{\frac{4-3\xi}{2}}} d\log\left(1/\delta\right)\log^2\left(d/\beta\right)}{m^2(1-\gamma)^2\epsilon^2n^2}\right) + \mathcal{O}\left(\frac{\log d}{n^{\xi}}\right) + \mathcal{O}\left(\frac{1}{n^{\frac{3\xi}{2}}}\right)$$

Remark 15 *This improves over the bound of Theorem 9, whenever $d = \tilde{\Omega}\left(\frac{\epsilon^2 n^{2-\xi}}{\gamma^{\mathcal{I}} - \gamma^{\mathcal{I}n^{\frac{4-3\xi}{2}}}}\right)$.*

References

Kareem Amin, Matthew Joseph, and Jieming Mao. Pan-private uniformity testing. *arXiv preprint arXiv:1911.01452*, 2019.

Aleksandr Aravkin, James Burke, and Dmitriy Drusvyatskiy. Convex analysis and nonsmooth optimization, 2017. URL https://sites.math.washington.edu/~burke/crs/516/notes/graduate-nco.pdf.

Lucas Bourtoule, Varun Chandrasekaran, Christopher Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. *arXiv preprint arXiv:1912.03817*, 2019.

Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pages 463–480. IEEE, 2015.

Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. When machine unlearning jeopardizes privacy. *arXiv preprint arXiv:2005.02205*, 2020.

Yuxin Chen. Ele 522 lecture notes: Gradient methods for unconstrained problems, 2019. URL http://www.princeton.edu/~yc5/ele522_optimization/lectures/grad_descent_unconstrained.pdf.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

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

Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 715–724, 2010.

Antonio Ginart, Melody Y. Guan, Gregory Valiant, and James Zou. Making AI forget you: Data deletion in machine learning. *CoRR*, abs/1907.05012, 2019. URL http://arxiv.org/abs/1907.05012.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens van der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

Zachary Izzo, Mary Anne Smart, Kamalika Chaudhuri, and James Zou. Approximate data deletion from machine learning models: Algorithms and evaluations. *arXiv preprint arXiv:2002.10077*, 2020.

H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016. URL http://arxiv.org/abs/1602.05629.

Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84, 2007.

Nicolas Papernot, Martin Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data, 2016.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017.

Jeffrey S. Vitter. Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1):37?57, March 1985. ISSN 0098-3500. doi: 10.1145/3147.3165. URL https://doi.org/10.1145/3147.3165.

Yuchen Zhang, John C. Duchi, and Martin Wainwright. Comunication-efficient algorithms for statistical optimization, 2012.

Appendix A. Supplement for Model and Preliminaries

Assumption 16 *For notational simplicity (so that we can state asymptotic bounds in terms of n) We assume throughout that over the course of an update sequence, the size of the updated datasets never drops below $n/2$ where n is the size of the original training dataset: $\forall\, i,\ n_i \geq n/2$ where n_i is the size of $\mathcal{D}_i$. Note that this is consistent with update sequences being of arbitrary length, since we allow additions as well as deletions. This assumption is not necessary, but otherwise bounds would have to be stated in terms of n_i.*

Definition 17 (Strong Convexity) *A function $h : \Theta \to \mathbb{R}$ is said to be m-strongly convex for some $m \geq 0$, if for any $\theta_1, \theta_2 \in \Theta$, and any $t \in (0, 1)$,*

$$h\left(t\theta_1 + (1-t)\theta_2\right) \leq th(\theta_1) + (1-t)h(\theta_2) - \frac{m}{2}t(1-t)\left\|\theta_1 - \theta_2\right\|_2^2$$

if the above condition holds for $m = 0$, we say h is convex.

Definition 18 (Lipschitzness) *A function $h : \Theta \to \mathbb{R}$ is said to be L-Lipschitz if for all $\theta_1, \theta_2 \in \Theta$,*

$$|h(\theta_1) - h(\theta_2)| \leq L\left\|\theta_1 - \theta_2\right\|_2$$

Definition 19 (Smoothness) *A function $h : \Theta \to \mathbb{R}$ is said to be M-smooth, if it is differentiable and for all $\theta_1, \theta_2 \in \Theta$,*

$$\left\|\nabla h(\theta_1) - \nabla h(\theta_2)\right\|_2 \leq M\left\|\theta_1 - \theta_2\right\|_2$$

Appendix B. Convergence Results for Gradient Descent

We make use of projected gradient descent extensively throughout this paper. Here, we state two convergence results for gradient descent that we will use. A crucial feature of these bounds (and one not shared by all bounds for gradient descent and its variants) is that they improve as a function of how close our initial parameter is to the optimal parameter.

Let $h : \Theta \to \mathbb{R}$ where $\Theta \subseteq \mathbb{R}^d$ is convex, closed, and bounded. Our goal is to approximate $\min_{\theta \in \Theta} h(\theta)$. The Gradient Descent (GD) algorithm starts with an initial point $\theta_0 \in \Theta$ and proceeds as follows:

$$\forall\, t \geq 1 : \quad \theta_t = \text{Proj}_{\Theta}\left(\theta_{t-1} - \eta_t \nabla h(\theta_{t-1})\right)$$

$\text{Proj}_{\Theta}(\theta) = \text{argmin}_{\theta' \in \Theta} \|\theta - \theta'\|_2$ is a projection onto Θ, and η_t is the step size used in round t.

Theorem 20 (Strongly Convex and Smooth Chen (2019)) *Let h be m-strongly convex and M-smooth, and let $\theta^* = \text{argmin}_{\theta \in \Theta} h(\theta)$. We have that after T steps of GD with step size $\eta_t = \frac{2}{m+M}$,*

$$\|\theta_T - \theta^*\|_2 \leq \left(\frac{M-m}{M+m}\right)^T \|\theta_0 - \theta^*\|_2$$

Theorem 21 (Convex and Smooth Aravkin et al. (2017)) *Let h be convex and M-smooth, and let $\theta^* \in \text{argmin}_{\theta \in \Theta} h(\theta)$. We have that after T steps of GD with step size $\eta_t = \frac{1}{M}$,*

$$h(\theta_T) - \min_{\theta \in \Theta} h(\theta) \leq \frac{M \|\theta_0 - \theta^*\|_2^2}{2T}$$

Appendix C. Probabilistic Tools

Lemma 22 *Suppose X, Y are random variables over the same domain Ω, and let Z be any random variable. If with probability at least $1 - \delta$ over Z, we have $X|Z \overset{\epsilon,\delta}{\approx} Y|Z$, then $X \overset{\epsilon,2\delta}{\approx} Y$.*

Proof Define for any z, the following (good) event:

$$G(z) = \left\{ z : X|\,(Z = z) \overset{\epsilon,\delta}{\approx} Y|\,(Z = z) \right\}$$

and note that $\Pr_{z \sim Z}\left[z \notin G(Z)\right] \leq \delta$. We have that for any $S \subseteq \Omega$,

$$\begin{aligned}
\Pr\left[X \in S\right] &= \mathbb{E}_{z \sim Z}\left[\Pr\left[X \in S | Z = z\right]\right] \\
&= \mathbb{E}_{z \sim Z}\left[\Pr\left[X \in S | Z = z\right] \mathbb{1}\left(z \in G(z)\right) + \Pr\left[X \in S | Z = z\right] \mathbb{1}\left(z \notin G(z)\right)\right] \\
&\leq \mathbb{E}_{z \sim Z}\left[e^{\epsilon} \Pr\left[Y \in S | Z = z\right] + \delta\right] + \Pr_{z \sim Z}\left[z \notin G(z)\right] \\
&\leq e^{\epsilon} \mathbb{E}_{z \sim Z}\left[\Pr\left[Y \in S | Z = z\right]\right] + 2\delta \\
&= e^{\epsilon} \Pr\left[Y \in S\right] + 2\delta
\end{aligned}$$

where $\mathbb{1}(A)$ is the indicator function of event A, for any A. This completes the proof because we can similarly show,

$$\Pr\left[Y \in S\right] \leq e^{\epsilon} \Pr\left[X \in S\right] + 2\delta$$

■

Lemma 23 (Gaussian Tail Bound) *Let $Z \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$. We have that for any $\beta > 0$,*

$$\Pr\left[\|Z\|_2 \geq \sigma\sqrt{2d}\log(2d/\beta)\right] \leq \beta$$

Lemma 24 (Gaussian Mechanism Bun and Steinke (2016)) *Let $X \sim \mathcal{N}(\mu, \sigma^2\mathbb{I}_d)$ and $Y \sim \mathcal{N}(\mu', \sigma^2\mathbb{I}_d)$. Suppose $\|\mu - \mu'\|_2 \leq \Delta$. We have that for any $\delta > 0$, $X \overset{\epsilon,\delta}{\approx} Y$, where*

$$\epsilon = \frac{\Delta^2}{2\sigma^2} + \frac{\Delta}{\sigma}\sqrt{2\log(1/\delta)}$$

Lemma 25 *Let $X \geq 0$ be any random variable drawn from a distribution $\mathcal{P}$, with finite expectation $\mu = \mathbb{E}_{X\sim\mathcal{P}}[X]$. Let $X_1, \ldots X_N \overset{iid}{\sim} \mathcal{P}$. Then if $X_{min} \triangleq \min_j X_j$, for $N \geq \frac{\log(1/\delta)}{\log 2}$, with probability at least $1 - \delta$: $X_{\min} < 2\mu$.*

Proof By Markov's inequality, for any $X_j, \Pr[X_j \geq 2\mu] \leq \frac{1}{2}$. Hence,

$$\Pr[X_{min} \geq 2\mu] = \prod_{j=1}^{N} \Pr[X_j \geq 2\mu] \leq \left(\frac{1}{2}\right)^N \leq \left(\frac{1}{2}\right)^{\frac{\log(1/\delta)}{\log 2}} = \delta$$

as desired. ■

Lemma 26 (Chernoff Bound) *Let $X \sim$ Binomial (m, p), and let $\mu = mp$. Then for any $\delta' \geq 0$,*

$$\Pr\left[X \geq (1+\delta')\mu\right] \leq e^{-\frac{\mu\delta'^2}{2+\delta'}}$$

Appendix D. Proof of Sensitivity Lemma 8

To prove Lemma 8, we will need the following claim.

Claim 27 *Suppose $h : \Theta \to \mathbb{R}$ is m-strongly convex and let $\theta^* = \operatorname{argmin}_{\theta\in\Theta} h(\theta)$. We have that for any $\theta \in \Theta$, $h(\theta) \geq h(\theta^*) + \frac{m}{2}\|\theta - \theta^*\|_2^2$.*

Proof First, recall the definition of m-strong convexity: for any $\theta_1, \theta_2 \in \Theta$, and any $t \in (0, 1)$,

$$h(t\theta_1 + (1-t)\theta_2) \leq th(\theta_1) + (1-t)h(\theta_2) - \frac{m}{2}t(1-t)\|\theta_1 - \theta_2\|_2^2$$

Now fix some $\theta \in \Theta$. We have that for any $t \in (0, 1)$,

$$h(\theta^*) \leq h(t\theta + (1-t)\theta^*) \leq th(\theta) + (1-t)h(\theta^*) - \frac{m}{2}t(1-t)\|\theta - \theta^*\|_2^2$$

where the first inequality follows because θ^* is the minimizer of h, and the second is due to m-strong convexity of h. Rearranging the above inequality and dividing both sides by t, we get that for *any* $t \in (0, 1)$,

$$h(\theta) \geq h(\theta^*) + \frac{m}{2}(1-t)\|\theta - \theta^*\|_2^2$$

We therefore have that

$$h(\theta) \geq h(\theta^*) + \frac{m}{2} \sup_{t \in (0,1)} (1-t) \left\|\theta - \theta^*\right\|_2^2 = h(\theta^*) + \frac{m}{2} \left\|\theta - \theta^*\right\|_2^2$$

■

Proof [Proof of Lemma 8] Fix n, a data set $\mathcal{D} = \{z_i\}_{i=1}^n$, and an update $u = (z, \bullet)$, and let $\mathcal{D}' = \mathcal{D} \circ u$. Assume $\bullet = \texttt{'delete'}$. If $z \notin \mathcal{D}$, then the claim immediately follows; so suppose $z \in \mathcal{D}$. We have that

$$\begin{aligned} f_{\mathcal{D}}\left(\theta^*_{\mathcal{D}'}\right) &= \frac{n-1}{n} f_{\mathcal{D}'}\left(\theta^*_{\mathcal{D}'}\right) + \frac{1}{n} f_z\left(\theta^*_{\mathcal{D}'}\right) \\ &\leq \frac{n-1}{n} f_{\mathcal{D}'}\left(\theta^*_{\mathcal{D}}\right) + \frac{1}{n} f_z\left(\theta^*_{\mathcal{D}'}\right) \\ &= f_{\mathcal{D}}\left(\theta^*_{\mathcal{D}}\right) + \frac{1}{n} f_z\left(\theta^*_{\mathcal{D}'}\right) - \frac{1}{n} f_z\left(\theta^*_{\mathcal{D}}\right) \\ &\leq f_{\mathcal{D}}\left(\theta^*_{\mathcal{D}}\right) + \frac{L}{n} \left\|\theta^*_{\mathcal{D}'} - \theta^*_{\mathcal{D}}\right\|_2 \end{aligned} \tag{1}$$

where the first inequality follows by optimality of $\theta^*_{\mathcal{D}'}$ for $\mathcal{D}'$, and the second follows by L-Lipschitzness of f_z. Note that since $f_{\mathcal{D}}$ is m-strongly convex, Claim 27 implies

$$f_{\mathcal{D}}\left(\theta^*_{\mathcal{D}'}\right) \geq f_{\mathcal{D}}\left(\theta^*_{\mathcal{D}}\right) + \frac{m}{2} \left\|\theta^*_{\mathcal{D}'} - \theta^*_{\mathcal{D}}\right\|_2^2 \tag{2}$$

Combining Equations (1) and (2) completes the proof for the case when $\bullet = \texttt{'delete'}$. Note that when $\bullet = \texttt{'add'}$, one can take $u' \triangleq (z, \texttt{'delete'})$, and use the bound for deletion to conclude that

$$\left\|\theta^*_{\mathcal{D}} - \theta^*_{\mathcal{D} \circ u}\right\|_2 = \left\|\theta^*_{\mathcal{D} \circ u} - \theta^*_{(\mathcal{D} \circ u) \circ u'}\right\|_2 \leq \frac{2L}{mn}$$

■

Appendix E. Proof of Theorem 9

Proof [Proof of Theorem 9] We first prove the unlearning guarantee. Fix a training dataset $\mathcal{D}$ of size n and an update sequence $\mathcal{U} = (u_i)_i$. Recall from Definition 2 the notation we use: $\{\mathcal{D}_i\}_{i \geq 0}$ for the sequence of updated datasets according to the update sequence $\mathcal{U}$, $\{\hat{\theta}_i\}_{i \geq 0}$ for the sequence of secret non-noisy parameters, and $\{\tilde{\theta}_i\}_{i \geq 0}$ for the sequence of published noisy parameters. We also use n_i to denote the size of $\mathcal{D}_i$. Note that $n_0 = n$ and that by Assumption 16, $n_i \geq n/2$ for all i. Let $\theta^*_i \triangleq \operatorname{argmin}_\theta f_{\mathcal{D}_i}(\theta)$ denote the optimizer of $f_{\mathcal{D}_i}$, for any $i \geq 0$.

We have that for any $i \geq 0$, $f_{\text{publish}}(\mathcal{A}(\mathcal{D}_i)) \sim \mathcal{N}\left(\mu_i, \sigma^2 \mathbb{I}_d\right)$, where it follows by the convergence guarantee of Theorem 20 that

$$\left\|\mu_i - \theta^*_i\right\|_2 \leq \gamma^T \left\|\theta'_0 - \theta^*_i\right\|_2 = \frac{2L\gamma^{\mathcal{I}} \left\|\theta'_0 - \theta^*_i\right\|_2}{Dmn_i} \leq \frac{4L}{mn} \cdot \gamma^{\mathcal{I}} \tag{3}$$

We also have that for any $i \geq 1$, $f_{\text{publish}}(\mathcal{R}_{\mathcal{A}}(\mathcal{D}_{i-1}, u_i, \theta_i)) \sim \mathcal{N}\left(\mu'_i, \sigma^2 \mathbb{I}_d\right)$ where

$$\left\|\mu'_i - \theta^*_i\right\|_2 \leq \frac{4L}{mn} \cdot \frac{\gamma^{\mathcal{I}}}{1 - \gamma^{\mathcal{I}}} \tag{4}$$

We use induction on i to prove this claim. Let's focus on the base case $i = 1$. We have that

$$\begin{aligned}\left\|\mu_1' - \theta_1^*\right\|_2 &\leq \gamma^{\mathcal{I}} \left\|\hat{\theta}_0 - \theta_1^*\right\|_2 \\ &\leq \gamma^{\mathcal{I}} \left(\left\|\hat{\theta}_0 - \theta_0^*\right\|_2 + \|\theta_0^* - \theta_1^*\|_2\right) \\ &\leq \gamma^{\mathcal{I}} \left(\frac{4L}{mn} \cdot \frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}} + \frac{4L}{mn}\right) \\ &= \frac{4L}{mn} \cdot \frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}}\end{aligned}$$

The first inequality follows from Theorem 20 and the fact that when running Algorithm 2 for the first update $i = 1$, the initial point $\theta_0' = \theta_1 \equiv \hat{\theta}_0$ saved by the training algorithm. The second inequality is a simple triangle inequality, and the third follows from Equation (3) (noting that $\hat{\theta}_0 \equiv \mu_0$) and the sensitivity Lemma 8. Let's move on to the induction step of the argument. Suppose Equation (4) holds for some $i \geq 1$. We will show that it holds for $(i+1)$ as well. We have that

$$\begin{aligned}\left\|\mu_{i+1}' - \theta_{i+1}^*\right\|_2 &\leq \gamma^{\mathcal{I}} \left\|\hat{\theta}_i - \theta_{i+1}^*\right\|_2 \\ &\leq \gamma^{\mathcal{I}} \left(\left\|\hat{\theta}_i - \theta_i^*\right\|_2 + \left\|\theta_i^* - \theta_{i+1}^*\right\|_2\right) \\ &\leq \gamma^{\mathcal{I}} \left(\frac{4L}{mn} \cdot \frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}} + \frac{4L}{mn}\right) \\ &= \frac{4L}{mn} \cdot \frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}}\end{aligned}$$

The first inequality follows from Theorem 20 and the fact that when running Algorithm 2 for the $(i+1)$th update, the initial point $\theta_0' = \theta_{i+1} = \hat{\theta}_i$ saved by the previous run of the unlearning algorithm. The second inequality is a simple triangle inequality, and the third follows from the induction assumption for i (noting that $\hat{\theta}_i \equiv \mu_i'$), the sensitivity Lemma 8, and the assumption that $n_i \geq n/2$.

We therefore have shown that for any $i \geq 1$, for $\theta_i \equiv \hat{\theta}_{i-1}$

$$f_{\text{publish}}\left(\mathcal{A}\left(\mathcal{D}_i\right)\right) \sim \mathcal{N}\left(\mu_i, \sigma^2 \mathbb{I}_d\right), \quad f_{\text{publish}}\left(\mathcal{R}_{\mathcal{A}}\left(\mathcal{D}_{i-1}, u_i, \theta_i\right)\right) \sim \mathcal{N}\left(\mu_i', \sigma^2 \mathbb{I}_d\right)$$

where Equations (3) and (4) imply

$$\left\|\mu_i - \mu_i'\right\|_2 \leq \Delta \triangleq \frac{8L}{mn} \cdot \frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}}$$

It follows from Lemma 24 that $\mathcal{R}_{\mathcal{A}}$ is a $(\frac{\Delta^2}{2\sigma^2} + \frac{\Delta}{\sigma}\sqrt{2\log(1/\delta)}, \delta)$-unlearning algorithm for $\mathcal{A}$, where, with σ specified in the theorem statement, we get (ϵ, δ)-unlearning guarantee.

Now let's prove the accuracy statement of the theorem. We will make use of Equations (3) and (4) and a Gaussian tail bound (see Lemma 23). Recall that for any $i \geq 0$, the published output $\tilde{\theta}_i = \hat{\theta}_i + Z$, and that $\hat{\theta}_0 \equiv \mu_0$ and $\hat{\theta}_i \equiv \mu_i'$ for $i \geq 1$. We therefore have that, for any β, and for any update step $i \geq 0$,

$$\Pr\left[\left\|\tilde{\theta}_i - \theta_i^*\right\|_2 \geq \frac{4L}{mn} \cdot \frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}} + \sigma\sqrt{2d}\log\left(2d/\beta\right)\right] \leq \beta$$

The choice of σ in the theorem and the fact that for $\epsilon = \mathcal{O}(\log(1/\delta))$, we have $\sqrt{\log(1/\delta) + \epsilon} - \sqrt{\log(1/\delta)} = \Omega(\epsilon/\sqrt{\log(1/\delta)})$, imply that for any $i \geq 0$, with probability at least $1 - \beta$,

$$\left\|\tilde{\theta}_i - \theta_i^*\right\|_2 = \mathcal{O}\left(\frac{L\gamma^{\mathcal{I}}\sqrt{d\log(1/\delta)}\log(d/\beta)}{(1-\gamma^{\mathcal{I}})\,\epsilon mn}\right) \tag{5}$$

Finally, since f_z is M-smooth for all z, we get that for any update step $i \geq 0$, with probability at least $1 - \beta$,

$$f_{\mathcal{D}_i}(\tilde{\theta}_i) - f_{\mathcal{D}_i}(\theta_i^*) \leq \frac{M}{2}\left\|\tilde{\theta}_i - \theta_i^*\right\|_2^2 = \mathcal{O}\left(\frac{ML^2\gamma^{2\mathcal{I}}d\log(1/\delta)\log^2(d/\beta)}{(1-\gamma^{\mathcal{I}})^2 m^2\epsilon^2 n^2}\right)$$

■

Appendix F. Perfect Unlearning

Theorem 28 (Perfect Unlearning) *Suppose for all $z \in \mathcal{Z}$, the loss function f_z is m-strongly convex, L-Lipschitz, and M-smooth. Define $\gamma \triangleq (M-m)/(M+m)$ and $\eta \triangleq 2/(M+m)$. Let the unlearning parameters ϵ and δ be such that $\epsilon = \mathcal{O}(\log(1/\delta))$. Let the learning algorithm $\mathcal{A}$ (Algorithm 1) run with $\eta_t = \eta$ and $T \geq \mathcal{I} + \log(\frac{Dmn}{2L})/\log(1/\gamma)$ where n is the size of the input dataset, and let the unlearning algorithm $\mathcal{R}_{\mathcal{A}}$ (Algorithm 2) run with input models $\theta_i \equiv \tilde{\theta}_{i-1}$ and $\eta_t = \eta$ and $T_i = \mathcal{I} + \log(\log(4di/\delta))/\log(1/\gamma)$ iterations for all $i \geq 1$ where*

$$\mathcal{I} \geq \frac{\log\left(\frac{\sqrt{2d}(1-\gamma)^{-1}}{\sqrt{2\log(2/\delta)+\epsilon}-\sqrt{2\log(2/\delta)}}\right)}{\log(1/\gamma)}, \text{ and } \quad \sigma = \frac{8L\gamma^{\mathcal{I}}\left(1-\gamma^{\mathcal{I}}\right)^{-1}}{mn\left(\sqrt{2\log(2/\delta)+3\epsilon}-\sqrt{2\log(2/\delta)+2\epsilon}\right)}$$

in f_{publish} (Algorithm 3). We have that

1. *Unlearning: $\mathcal{R}_{\mathcal{A}}$ is a strong (ϵ, δ)-perfect unlearning for $\mathcal{A}$ with respect to f_{publish}.*
2. *Accuracy: for any β, $(\mathcal{A}, \mathcal{R}_{\mathcal{A}})$ is $(\alpha, \beta+\delta)$-accurate with respect to f_{publish} where*

$$\alpha = \mathcal{O}\left(\frac{ML^2\gamma^{2\mathcal{I}}d\log(1/\delta)\log^2(d/\beta)}{(1-\gamma^{\mathcal{I}})^2 m^2\epsilon^2 n^2}\right)$$

Proof [Proof of Theorem 28] We first prove the unlearning guarantee. Fix a training dataset $\mathcal{D}$ of size n and an update sequence $\mathcal{U} = (u_i)_i$. Similar to the proof of Theorem 9, we first recall a few notations from Definition 2: $\{\mathcal{D}_i\}_{i\geq 0}$ for the sequence of updated datasets according to the update sequence $\mathcal{U}$, $\{\hat{\theta}_i\}_{i\geq 0}$ for the sequence of secret non-noisy parameters, and $\{\tilde{\theta}_i\}_{i\geq 0}$ for the sequence of published noisy parameters. Let Z_i denote the Gaussian noise added by f_{publish} on round i of update, and recall that $\tilde{\theta}_i = \hat{\theta}_i + Z_i$. We use n_i ($\geq n/2$) to denote the size of $\mathcal{D}_i$. Let $\theta_i^* \triangleq \operatorname{argmin}_\theta f_{\mathcal{D}_i}(\theta)$ denote the optimizer of $f_{\mathcal{D}_i}$.

We have that for any $i \geq 0$, $f_{\text{publish}}(\mathcal{A}(\mathcal{D}_i)) \sim \mathcal{N}\left(\mu_i, \sigma^2 \mathbb{I}_d\right)$, where it follows by the convergence guarantee of Theorem 20 that,

$$\|\mu_i - \theta_i^*\|_2 \leq \gamma^T \left\|\theta_0' - \theta_i^*\right\|_2 = \frac{2L\gamma^{\mathcal{I}} \|\theta_0' - \theta_i^*\|_2}{Dmn_i} \leq \frac{4L}{mn} \cdot \gamma^{\mathcal{I}} \tag{6}$$

We also have that for any update step $i \geq 1$, conditioned on the noise of previous rounds $\{Z_0, \ldots, Z_{i-1}\}$, $f_{\text{publish}}\left(\mathcal{R}_{\mathcal{A}}\left(\mathcal{D}_{i-1}, u_i, \theta_i\right)\right) \sim \mathcal{N}\left(\mu_i', \sigma^2 \mathbb{I}_d\right)$, where for any $\beta' > 0$,

$$\Pr_{Z_0,\ldots,Z_{i-1}}\left[\left\|\mu_i' - \theta_i^*\right\|_2 \geq \frac{\gamma^{T_i}}{1-\gamma^{\mathcal{I}}}\left(\frac{4L}{mn} + \sigma\sqrt{2d}\log\left(2d/\beta'\right)\right)\right] \leq i\beta' \tag{7}$$

We use induction on i to prove this claim. Fix any β'. Let's focus on the base case $i = 1$. We have that

$$\begin{aligned}
\left\|\mu_1' - \theta_1^*\right\|_2 \leq\ & \gamma^{T_1}\left\|\tilde{\theta}_0 - \theta_1^*\right\|_2 \\
& \leq \gamma^{T_1}\left(\|Z_0\|_2 + \left\|\hat{\theta}_0 - \theta_0^*\right\|_2 + \|\theta_0^* - \theta_1^*\|_2\right) \\
& \leq \gamma^{T_1}\left(\frac{4L}{mn}\gamma^{\mathcal{I}} + \frac{4L}{mn} + \sigma\sqrt{2d}\log\left(2d/\beta'\right)\right) \\
& \leq \gamma^{T_1}\left(\frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}}\left(\frac{4L}{mn} + \sigma\sqrt{2d}\log\left(2d/\beta'\right)\right) + \frac{4L}{mn} + \sigma\sqrt{2d}\log\left(2d/\beta'\right)\right) \\
& = \frac{\gamma^{T_1}}{1-\gamma^{\mathcal{I}}}\left(\frac{4L}{mn} + \sigma\sqrt{2d}\log\left(2d/\beta'\right)\right)
\end{aligned}$$

The first inequality follows from Theorem 20 and the fact that when running Algorithm 2 for the first update $i = 1$, the initial point of the algorithm $\theta_0' = \theta_1 \equiv \tilde{\theta}_0$. The second inequality is a simple triangle inequality, and the third holds with probability at least $1 - \beta'$ and follows from Equation (6) (noting that $\hat{\theta}_0 \equiv \mu_0$), the sensitivity Lemma 8, and a Gaussian tail bound for Z_0 (Lemma 23). Let's move on to the induction step of the argument. Suppose Equation (7) holds for some $i \geq 1$. We will show that it holds for $(i+1)$ as well. We have that

$$\begin{aligned}
\left\|\mu_{i+1}' - \theta_{i+1}^*\right\|_2 \leq\ & \gamma^{T_{i+1}}\left\|\tilde{\theta}_i - \theta_{i+1}^*\right\|_2 \\
& \leq \gamma^{T_{i+1}}\left(\|Z_i\|_2 + \left\|\hat{\theta}_i - \theta_i^*\right\|_2 + \left\|\theta_i^* - \theta_{i+1}^*\right\|_2\right) \\
& \leq \gamma^{T_{i+1}}\left(\frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}}\left(\frac{4L}{mn} + \sigma\sqrt{2d}\log\left(2d/\beta'\right)\right) + \frac{4L}{mn} + \sigma\sqrt{2d}\log\left(2d/\beta'\right)\right) \\
& = \frac{\gamma^{T_{i+1}}}{1-\gamma^{\mathcal{I}}}\left(\frac{4L}{mn} + \sigma\sqrt{2d}\log\left(2d/\beta'\right)\right)
\end{aligned}$$

The first inequality follows from Theorem 20 and the fact that when running Algorithm 2 for the $(i+1)$th update, the initial point of the algorithm $\theta_0' = \theta_{i+1} \equiv \tilde{\theta}_i$. The second inequality is a simple triangle inequality, and the third holds with probability at least $1 - (i+1)\beta'$ and follows from the induction assumption for i (note $\hat{\theta}_i \equiv \mu_i'$ and $T_i \geq \mathcal{I}$), the sensitivity Lemma 8 (note $n_i \geq n/2$), and a Gaussian tail bound for Z_i (Lemma 23). Now with the choice of $\beta' = \delta/(2i)$, Equation (7)

implies with probability at least $1-\delta/2$ over the Gaussian noise draws $\{Z_0,\ldots,Z_{i-1}\}$,

$$\left\|\mu_i' - \theta_i^*\right\|_2 \leq \frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}}\left(\frac{4L}{mn} + \sigma\sqrt{2d}\right) \tag{8}$$

because $\gamma^{T_i} \leq \left(\log\left(4di/\delta\right)\right)^{-1}\gamma^{\mathcal{I}}$. We therefore have shown that for any $i \geq 1$, conditioned on $\{Z_0,\ldots,Z_{i-1}\}$

$$f_{\text{publish}}\left(\mathcal{A}\left(\mathcal{D}_i\right)\right) \sim \mathcal{N}\left(\mu_i, \sigma^2\mathbb{I}_d\right), \quad f_{\text{publish}}\left(\mathcal{R}_{\mathcal{A}}\left(\mathcal{D}_{i-1}, u_i, \theta_i\right)\right) \sim \mathcal{N}\left(\mu_i', \sigma^2\mathbb{I}_d\right)$$

where Equations (6) and (8) imply, with probability $1-\delta/2$ over $\{Z_0,\ldots,Z_{i-1}\}$,

$$\left\|\mu_i - \mu_i'\right\|_2 \leq \frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}}\left(\frac{4L}{mn} + \sigma\sqrt{2d}\right) + \frac{4L}{mn}\cdot\gamma^{\mathcal{I}} \leq \frac{2\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}}\left(\frac{4L}{mn} + \sigma\sqrt{2d}\right) \triangleq \Delta$$

It then follows from Lemma 24, as well as the choice of σ and the assumption on $\mathcal{I}$ in the theorem statement, that for any $i \geq 1$, with probability $1-\delta/2$ over $\{Z_0,\ldots,Z_{i-1}\}$,

$$f_{\text{publish}}\left(\mathcal{A}\left(\mathcal{D}_i\right)\right) \overset{\epsilon,\delta/2}{\approx} f_{\text{publish}}\left(\mathcal{R}_{\mathcal{A}}\left(\mathcal{D}_{i-1}, u_i, \theta_i\right)\right)$$

Now we can apply Lemma 22 to conclude that for any $i \geq 1$,

$$f_{\text{publish}}\left(\mathcal{A}\left(\mathcal{D}_i\right)\right) \overset{\epsilon,\delta}{\approx} f_{\text{publish}}\left(\mathcal{R}_{\mathcal{A}}\left(\mathcal{D}_{i-1}, u_i, \theta_i\right)\right)$$

And this shows $\mathcal{R}_{\mathcal{A}}$ is an (ϵ,δ)-unlearning algorithm for $\mathcal{A}$, as desired.

Now let's prove the accuracy statement of the theorem. We will make use of Equations (6) and (8) and a Gaussian tail bound (see Lemma 23). Recall that for any $i \geq 0$, the published output $\tilde{\theta}_i = \hat{\theta}_i + Z$, and that $\hat{\theta}_0 \equiv \mu_0$ and $\hat{\theta}_i \equiv \mu_i'$ for $i \geq 1$. We therefore have that, for any β, and for any update step $i \geq 0$,

$$\Pr_{Z_0,\ldots,Z_i}\left[\left\|\tilde{\theta}_i - \theta_i^*\right\|_2 \geq \frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}}\left(\frac{4L}{mn} + \sigma\sqrt{2d}\right) + \sigma\sqrt{2d\log\left(2d/\beta\right)}\right] \leq \beta + \frac{\delta}{2}$$

The choice of σ in the theorem and the fact that for $\epsilon = \mathcal{O}\left(\log\left(1/\delta\right)\right)$, we have $\sqrt{\log\left(1/\delta\right)+\epsilon} - \sqrt{\log\left(1/\delta\right)} = \Omega(\epsilon/\sqrt{\log\left(1/\delta\right)})$, imply for any update step $i \geq 0$, with probability at least $1-\beta-\delta/2$,

$$\left\|\tilde{\theta}_i - \theta_i^*\right\|_2 = \mathcal{O}\left(\frac{L\gamma^{\mathcal{I}}\sqrt{d\log\left(1/\delta\right)}\log\left(d/\beta\right)}{\left(1-\gamma^{\mathcal{I}}\right)\epsilon mn}\right) \tag{9}$$

Finally, since f_z is M-smooth for all z, we get that for any update step $i \geq 0$, with probability at least $1-\beta-\delta/2$,

$$f_{\mathcal{D}_i}(\tilde{\theta}_i) - f_{\mathcal{D}_i}(\theta_i^*) \leq \frac{M}{2}\left\|\tilde{\theta}_i - \theta_i^*\right\|_2^2 = \mathcal{O}\left(\frac{ML^2\gamma^{2\mathcal{I}}d\log\left(1/\delta\right)\log^2\left(d/\beta\right)}{\left(1-\gamma^{\mathcal{I}}\right)^2 m^2\epsilon^2 n^2}\right)$$

■

Appendix G. Proof of Theorem 10

First note that:

Claim 29 *If f_z is convex, L-Lipschitz, and M-smooth, then g_z is m-strongly convex, $(L+mD)$-Lipschitz, and $(M+m)$-smooth.*

Proof [Proof of Theorem 10] The unlearning guarantee of the theorem holds for any $m > 0$, and follows from Theorem 9 by the choice of σ in the theorem statement. Let's prove the accuracy statement. Let $\theta_i^{*r} = \operatorname{argmin}_{\theta\in\Theta} g_{\mathcal{D}_i}(\theta)$ denote the optimizer of the regularized $g_{\mathcal{D}_i}$, for all $i \geq 0$. It follows from the proof of Theorem 9 (see Equation (5)) that for any update step $i \geq 0$, with probability $1-\beta$,

$$\left\|\tilde{\theta}_i - \theta_i^{*r}\right\|_2 = \mathcal{O}\left(\frac{(L+mD)\,\gamma^{\mathcal{I}}\sqrt{d\log(1/\delta)}\log(d/\beta)}{(1-\gamma^{\mathcal{I}})\,\epsilon mn}\right) \tag{10}$$

Also note that

$$\frac{\gamma^{\mathcal{I}}}{1-\gamma^{\mathcal{I}}} = \frac{1}{(1+2\,(m/M))^{\mathcal{I}}-1} \leq \frac{M}{m\mathcal{I}} \tag{11}$$

Now let $\theta_i^* \in \operatorname{argmin}_{\theta\in\Theta} f_{\mathcal{D}_i}(\theta)$ denote an optimizer of the original loss function $f_{\mathcal{D}_i}$, for any $i \geq 0$. We have that, for any $i \geq 0$,

$$\begin{aligned}
f_{\mathcal{D}_i}(\tilde{\theta}_i) - f_{\mathcal{D}_i}(\theta_i^*) &= f_{\mathcal{D}_i}(\tilde{\theta}_i) - f_{\mathcal{D}_i}(\theta_i^{*r}) + f_{\mathcal{D}_i}(\theta_i^{*r}) - f_{\mathcal{D}_i}(\theta_i^*) \\
&\overset{(1)}{\leq} \nabla f_{\mathcal{D}_i}(\theta_i^{*r})^\top\left(\tilde{\theta}_i - \theta_i^{*r}\right) + \frac{M}{2}\left\|\tilde{\theta}_i - \theta_i^{*r}\right\|_2^2 + f_{\mathcal{D}_i}(\theta_i^{*r}) - f_{\mathcal{D}_i}(\theta_i^*) \\
&\overset{(2)}{=} \frac{M}{2}\left\|\tilde{\theta}_i - \theta_i^{*r}\right\|_2^2 + m\theta_i^{*r\top}\left(\theta_i^{*r} - \tilde{\theta}_i\right) + f_{\mathcal{D}_i}(\theta_i^{*r}) - f_{\mathcal{D}_i}(\theta_i^*) \\
&\overset{(3)}{\leq} \frac{M}{2}\left\|\tilde{\theta}_i - \theta_i^{*r}\right\|_2^2 + mD^2 + f_{\mathcal{D}_i}(\theta_i^{*r}) - f_{\mathcal{D}_i}(\theta_i^*) \\
&= \frac{M}{2}\left\|\tilde{\theta}_i - \theta_i^{*r}\right\|_2^2 + mD^2 + g_{\mathcal{D}_i}(\theta_i^{*r}) - \frac{m}{2}\|\theta_i^{*r}\|_2^2 - f_{\mathcal{D}_i}(\theta_i^*) \\
&\overset{(4)}{\leq} \frac{M}{2}\left\|\tilde{\theta}_i - \theta_i^{*r}\right\|_2^2 + mD^2 + g_{\mathcal{D}_i}(\theta_i^*) - \frac{m}{2}\|\theta_i^{*r}\|_2^2 - f_{\mathcal{D}_i}(\theta_i^*) \\
&= \frac{M}{2}\left\|\tilde{\theta}_i - \theta_i^{*r}\right\|_2^2 + mD^2 + \frac{m}{2}\left(\|\theta_i^*\|_2^2 - \|\theta_i^{*r}\|_2^2\right) \\
&\overset{(5)}{=} \mathcal{O}\left(\frac{M^3\,(L+mD)^2\,d\log(1/\delta)\log^2(d/\beta)}{m^4\epsilon^2n^2\mathcal{I}^2} + mD^2\right)
\end{aligned} \tag{12}$$

where inequality (1) follows from $f_{\mathcal{D}_i}$ being M-smooth. (2) follows from the fact that for all θ, $\nabla f_{\mathcal{D}_i}(\theta) = \nabla g_{\mathcal{D}_i}(\theta) - m\theta$ and that by optimality of θ_i^{*r} for $g_{\mathcal{D}_i}$, we have $\nabla g_{\mathcal{D}_i}(\theta_i^{*r}) = 0$. (3) follows from a simple application of Cauchy-Schwarz: for all $\theta_1, \theta_2 \in \Theta$, we have $\theta_1^\top\theta_2 \leq \|\theta_1\|_2\|\theta_2\|_2 \leq D^2$. (4) follows from the optimality of θ_i^{*r} for $g_{\mathcal{D}_i}$, and (5) is implied by Equations (10) and (11), and it holds with probability $1-\beta$. Now for the choice of m in the theorem, we conclude that for any $i \geq 0$, with probability $1-\beta$,

$$f_{\mathcal{D}_i}(\tilde{\theta}_i) - f_{\mathcal{D}_i}(\theta_i^*) = \mathcal{O}\left(\left(\frac{M^{\frac{3}{2}}LD^4\sqrt{d\log(1/\delta)}}{\epsilon n\mathcal{I}}\right)^{\frac{2}{5}}\log^2(d/\beta)\right) + \mathcal{O}\left(n^{-\frac{4}{5}}\right) + \mathcal{O}\left(n^{-\frac{6}{5}}\right)$$

■

Appendix H. Weak Unlearning for Convex Loss

Theorem 30 (Accuracy, Unlearning, and Computation Tradeoffs) *Suppose for all $z \in \mathcal{Z}$, the loss function f_z is convex, L-Lipschitz, and M-smooth, and let $g_z(\cdot) = f_z(\cdot) + \frac{m}{2}\|\cdot\|_2^2$ for some m specified later. Define $\gamma \triangleq M/(M+2m)$ and $\eta \triangleq 2/(M+2m)$. Let the learning algorithm $\mathcal{A}$ (Algorithm 1) run on the regularized g with $\eta_t = \eta$ and $T \geq \mathcal{I} + \log(\frac{Dmn}{2L})/\log(1/\gamma)$ where n is the size of the input dataset, and let the unlearning algorithm $\mathcal{R}_{\mathcal{A}}$ (Algorithm 2) run on the regularized g with input model $\theta_i \equiv \hat{\theta}_{i-1}$ and $\eta_t = \eta$ and $T_i = i^2 \cdot \mathcal{I}$ iterations, for the ith update. Let the unlearning parameters ϵ and δ be such that $\epsilon = \mathcal{O}(\log(1/\delta))$, and let*

$$\sigma = \frac{2\sqrt{2M}\,(L+mD)}{m\sqrt{m\mathcal{I}}n\left(\sqrt{\log(1/\delta)+\epsilon}-\sqrt{\log(1/\delta)}\right)}, \quad m = \sqrt{\frac{LM\sqrt{d\log(1/\delta)}}{D\epsilon n\sqrt{\mathcal{I}}}}$$

where σ is the noise level in f_{publish}. We have that

1. *Unlearning: $\mathcal{R}_{\mathcal{A}}$ is a weak (ϵ,δ)-unlearning algorithm for $\mathcal{A}$ with respect to f_{publish}.*
2. *Accuracy: for any β, $(\mathcal{A},\mathcal{R}_{\mathcal{A}})$ is (α,β)-accurate with respect to f_{publish} where*

$$\alpha = \mathcal{O}\left(\sqrt{\frac{MLD^3\sqrt{d\log(1/\delta)}}{\epsilon n\sqrt{\mathcal{I}}}}\log^2(d/\beta)\right) + \mathcal{O}\left(n^{-1}\right) + \mathcal{O}\left(n^{-\frac{3}{2}}\right)$$

Remark 31 *We remark that we can further explore the tradeoff between each update's runtime T_i and dependence on sample size n. Let $\xi \geq 1$ be any constant (Theorem 30 corresponds to $\xi = 1$). We have that under the setting of Theorem 30, with $T_i = i^{2\xi} \cdot \mathcal{I}$ iterations, and*

$$\sigma = \frac{2\sqrt{2}M^{\frac{1}{2\xi}}\,(L+mD)}{m(m\mathcal{I})^{\frac{1}{2\xi}}n\left(\sqrt{\log(1/\delta)+\epsilon}-\sqrt{\log(1/\delta)}\right)}, \quad m = \left(\frac{L^2M^{\frac{1+\xi}{\xi}}d\log(1/\delta)}{D^2\epsilon^2n^2\mathcal{I}^{\frac{1}{\xi}}}\right)^{\frac{\xi}{3\xi+1}}$$

1. *Unlearning: $\mathcal{R}_{\mathcal{A}}$ is a weak (ϵ,δ)-unlearning algorithm for $\mathcal{A}$ with respect to f_{publish}.*
2. *Accuracy: for any β, $(\mathcal{A},\mathcal{R}_{\mathcal{A}})$ is (α,β)-accurate with respect to f_{publish} where*

$$\alpha = \mathcal{O}\left(\left(\frac{M^{\frac{1+\xi}{\xi}}L^2D^{\frac{2+4\xi}{\xi}}d\log(1/\delta)}{\epsilon^2n^2\mathcal{I}^{\frac{1}{\xi}}}\right)^{\frac{\xi}{3\xi+1}}\log^2(d/\beta)\right) + \mathcal{O}\left(n^{-\frac{4\xi}{3\xi+1}}\right) + \mathcal{O}\left(n^{-\frac{6\xi}{3\xi+1}}\right)$$

Proof [Proof of Theorem 30] We first prove the unlearning guarantee. Fix a training dataset $\mathcal{D}$ of size n and an update sequence $\mathcal{U} = (u_i)_i$. Recall from Definition 2 the notation we use: $\{\mathcal{D}_i\}_{i\geq 0}$ for the sequence of updated datasets according to the update sequence $\mathcal{U}$, $\{\hat{\theta}_i\}_{i\geq 0}$ for the sequence of secret non-noisy parameters, and $\{\tilde{\theta}_i\}_{i\geq 0}$ for the sequence of published noisy parameters. We also use n_i to denote the size of $\mathcal{D}_i$. Note that $n_0 = n$ and that by Assumption 16, $n_i \geq n/2$ for all i. Let $\theta_i^* \in \operatorname{argmin}_\theta f_{\mathcal{D}_i}(\theta)$ denote an optimizer of $f_{\mathcal{D}_i}$. Let $\theta_i^{*r} = \operatorname{argmin}_{\theta\in\Theta} g_{\mathcal{D}_i}(\theta)$ denote the optimizer of the regularized loss $g_{\mathcal{D}_i}$.

Fact 32 *Note that for any positive integer T',*

$$\gamma^{T'} = \left(\frac{1}{1+2\left(m/M\right)}\right)^{T'} \leq \frac{1}{1+2\left(m/M\right)T'} \leq \sqrt{\frac{M}{mT'}} \tag{13}$$

where the last inequality follows because for all $x \geq 0$, $1+x \geq 2\sqrt{x}$.

Fact 33 (Generalizing Fact 32) *In general, for any constant $\xi \geq 1$ and any integer T', we have*

$$\gamma^{T'} = \left(\gamma^{\xi T'}\right)^{\frac{1}{\xi}} \leq \left(\frac{M}{mT'}\right)^{\frac{1}{2\xi}} \tag{14}$$

We will use Fact 32 later on in the proof and we note that Remark 31 follows by using the more general Fact 33. of Let $L' \triangleq L + mD$ which is the Lipschitz constant of the regularized loss function g. We have that for any $i \geq 0$, $f_{\text{publish}}(\mathcal{A}\left(\mathcal{D}_i\right)) \sim \mathcal{N}\left(\mu_i, \sigma^2\mathbb{I}_d\right)$, where it follows by the convergence guarantee of Theorem 20 that

$$\left\|\mu_i - \theta_i^{*r}\right\|_2 \leq \gamma^T \left\|\theta_0' - \theta_i^{*r}\right\|_2 \leq \frac{2L'\gamma^{\mathcal{I}}\left\|\theta_0' - \theta_i^{*r}\right\|_2}{Dmn_i} \leq \frac{2L'}{mn_i} \cdot \gamma^{\mathcal{I}} \tag{15}$$

We also have that for any $i \geq 1$, $f_{\text{publish}}(\mathcal{R}_{\mathcal{A}}\left(\mathcal{D}_{i-1}, u_i, \theta_i\right)) \sim \mathcal{N}\left(\mu_i', \sigma^2\mathbb{I}_d\right)$ where

$$\left\|\mu_i' - \theta_i^{*r}\right\|_2 \leq \frac{4L'}{mn} \cdot i \cdot \gamma^{i^2\mathcal{I}} \tag{16}$$

We use induction on i to prove this claim. Let's focus on the base case $i = 1$. We have that

$$\begin{aligned}
\left\|\mu_1' - \theta_1^{*r}\right\|_2 &\leq \gamma^{\mathcal{I}} \left\|\hat{\theta}_0 - \theta_1^{*r}\right\|_2 \\
&\leq \gamma^{\mathcal{I}} \left(\left\|\hat{\theta}_0 - \theta_0^{*r}\right\|_2 + \left\|\theta_0^{*r} - \theta_1^{*r}\right\|_2\right) \\
&\leq \gamma^{\mathcal{I}} \left(\frac{2L'}{mn} \cdot \gamma^{\mathcal{I}} + \frac{2L'}{mn}\right) \\
&\leq \frac{4L'}{mn} \cdot \gamma^{\mathcal{I}}
\end{aligned}$$

The first inequality follows from Theorem 20 and the fact that when running Algorithm 2 for the first update $i = 1$, the initial point $\theta_0' = \theta_1 \equiv \hat{\theta}_0$ saved by the training algorithm. The second inequality is a simple triangle inequality, and the third follows from Equation (15) (noting that $\hat{\theta}_0 \equiv \mu_0$) and the sensitivity Lemma 8. Let's move on to the induction step of the argument. Suppose Equation (4) holds for some $i \geq 1$. We will show that it holds for $(i+1)$ as well. We have that

$$\begin{aligned}
\left\|\mu_{i+1}' - \theta_{i+1}^{*r}\right\|_2 &\leq \gamma^{(i+1)^2\mathcal{I}} \left\|\hat{\theta}_i - \theta_{i+1}^{*r}\right\|_2 \\
&\leq \gamma^{(i+1)^2\mathcal{I}} \left(\left\|\hat{\theta}_i - \theta_i^{*r}\right\|_2 + \left\|\theta_i^{*r} - \theta_{i+1}^{*r}\right\|_2\right) \\
&\leq \gamma^{(i+1)^2\mathcal{I}} \left(\frac{4L'}{mn} \cdot i \cdot \gamma^{i^2\mathcal{I}} + \frac{4L'}{mn}\right) \\
&\leq \frac{4L'}{mn} \cdot (i+1) \cdot \gamma^{(i+1)^2\mathcal{I}}
\end{aligned}$$

The first inequality follows from Theorem 20 and the fact that when running Algorithm 2 for the $(i+1)$th update, the initial point $\theta'_0 = \theta_{i+1} \equiv \hat{\theta}_i$ saved by the previous run of the unlearning algorithm. The second inequality is a simple triangle inequality, and the third follows from the induction assumption for i (noting that $\hat{\theta}_i \equiv \mu'_i$), the sensitivity Lemma 8, and the assumption that $n_i \geq n/2$.

Now that we can apply Equation (13) to Equations (15) and (16) to conclude

$$\forall\, i \geq 0,\ \|\mu_i - \theta_i^{*r}\|_2 \leq \frac{4L'\sqrt{M}}{m\sqrt{m\mathcal{I}n}}, \quad \forall\, i \geq 1,\ \left\|\mu'_i - \theta_i^{*r}\right\|_2 \leq \frac{4L'\sqrt{M}}{m\sqrt{m\mathcal{I}n}} \tag{17}$$

We therefore have shown that for any $i \geq 1$,

$$f_{\text{publish}}\left(\mathcal{A}\left(\mathcal{D}_i\right)\right) \sim \mathcal{N}\left(\mu_i, \sigma^2 \mathbb{I}_d\right), \quad f_{\text{publish}}\left(\mathcal{R}_{\mathcal{A}}\left(\mathcal{D}_{i-1}, u_i, \theta_i\right)\right) \sim \mathcal{N}\left(\mu'_i, \sigma^2 \mathbb{I}_d\right)$$

where Equation (17) implies

$$\left\|\mu_i - \mu'_i\right\|_2 \leq \frac{8L'\sqrt{M}}{m\sqrt{m\mathcal{I}n}} \triangleq \Delta$$

It then follows from Lemma 24 that $\mathcal{R}_{\mathcal{A}}$ is a $(\frac{\Delta^2}{2\sigma^2} + \frac{\Delta}{\sigma}\sqrt{2\log\left(1/\delta\right)}, \delta)$-unlearning algorithm for $\mathcal{A}$, where, with σ specified in the theorem statement, we get (ϵ, δ)-unlearning guarantee.

Now let's focus on the accuracy statement of the theorem. Note, similar to the proof of Theorem 9, the convergence bounds in Equation (17), the choice of σ in the theorem statement, as well as a Gaussian tail bound (Lemma 23), imply that for any update step $i \geq 0$, with probability at least $1 - \beta$,

$$\left\|\tilde{\theta}_i - \theta_i^{*r}\right\|_2 = \mathcal{O}\left(\frac{\sqrt{M}\left(L + mD\right)\sqrt{d\log\left(1/\delta\right)}\log\left(d/\beta\right)}{\epsilon m\sqrt{m\mathcal{I}n}}\right) \tag{18}$$

We therefore have that, using a similar analysis as in the proof of Theorem 10 (see Equation (12)), for any update step $i \geq 0$, with probability $1 - \beta$,

$$f_{\mathcal{D}_i}(\tilde{\theta}_i) - f_{\mathcal{D}_i}(\theta_i^*) = \mathcal{O}\left(\frac{M^2\left(L + mD\right)^2 d\log\left(1/\delta\right)\log^2\left(d/\beta\right)}{m^3\epsilon^2 n^2 \mathcal{I}} + mD^2\right)$$

Finally, with the choice of m in the theorem,

$$f_{\mathcal{D}_i}(\tilde{\theta}_i) - f_{\mathcal{D}_i}(\theta_i^*) = \mathcal{O}\left(\sqrt{\frac{MLD^3\sqrt{d\log\left(1/\delta\right)}}{\epsilon n\sqrt{\mathcal{I}}}}\log^2\left(d/\beta\right)\right) + \mathcal{O}\left(n^{-1}\right) + \mathcal{O}\left(n^{-\frac{3}{2}}\right)$$

■

Appendix I. Proofs of Lemmas in Section 4

Proof [Proof of Lemma 12] We prove the claim by induction on i. For $i = 0$, $\mathcal{S}_0$ is explicitly drawn from $\mathcal{P}^B(\mathcal{D}_0)$ and so the claim holds. Now assume the claim holds for $i-1$. In the case of addition, where $u_i = (z_i, \texttt{'add'})$ this is exactly what is known as "Reservoir Sampling with Replacement" and we refer the reader to Vitter (1985). So we need only establish the claim for deletion updates. Let

us perform an update $u_i = (z_i, \texttt{'delete'})$. We show that after conditioning on u_i, after the deletion update, each element of $\mathcal{S}_i$ is independent and has marginal distribution $\mathcal{P}(\mathcal{D}_{i-1} \circ u_i) = \mathcal{P}(\mathcal{D}_i)$, which will establish the claim. Conditioning on u_i, let $h_{u_i} : \mathcal{Z} \to \mathcal{Z}$ be the randomized function:

$$h_{u_i}(z) = \begin{cases} z & z \neq z_i \\ z' \sim \mathcal{P}(\mathcal{D}_i) & z = z_i \end{cases}$$

Then for any data point $z_l \in \mathcal{S}_{i-1}$, the corresponding element in $\mathcal{S}_i$ is $h_{u_i}(z_l)$. Since by assumption the $\{z_l\} = \mathcal{S}_{i-1}$ are independent, since h_{u_i} is a fixed randomized function conditioned on u_i, the $\{h_{u_i}(z_l)\} = \mathcal{S}_i$ are conditionally independent given u_i. It remains to show that the marginal distribution of any $z'_l = h_{u_i}(z_l)$ is $\mathcal{P}(\mathcal{D}_{i-1} \circ u_i) \equiv \mathcal{P}(\mathcal{D}_i)$. If $z_l = z_i$, then $z'_l \sim \mathcal{P}(\mathcal{D}_i)$ by design. If $z_l \neq z_i$, then $z'_l = z_l$, and the distribution of z'_l is $z_l | z_l \neq z_i, u_i$. Since $\mathcal{U}$ is a non-adaptive sequence of updates, $z_l | z_l \neq z_i, u_i \sim z_l | z_l \neq z_i$. Then by inductive assumption $z_l \sim \mathcal{P}(\mathcal{D}_{i-1})$, and so the distribution of $z_l | z_l \neq z_i$ for $z_i \in \mathcal{D}_{i-1}$ is uniform over $\mathcal{D}_{i-1} \setminus \{z_i\} = \mathcal{D}_{i-1} \circ u_i = \mathcal{D}_i$, which is exactly $\mathcal{P}(\mathcal{D}_i)$, as desired. This establishes the induction. ■

Proof [Proof of Lemma 13] At any round i of update, by Lemma 12, we know $\mathcal{S}_i \sim \mathcal{P}^B(\mathcal{D}_i)$. By Assumption 16, $n_i \geq n/2$ where n_i is the size of dataset $\mathcal{D}_i$. Hence for any data point z, the number of copies of z subsampled in $\mathcal{S}_i$ is distributed as Binomial(B, p), where $p \leq 2/n$. Let $\mu = (2B)/n$ and note that $\mu \geq 1$. Now by a Chernoff bound (see Lemma 26) for a Bernoulli random variable, we get that for any i, the number of repeated points of any one type in $\mathcal{S}_i$ (including the ones subject to update) satisfies, with probability $1 - \delta'$:

$$\begin{aligned} s_i &\leq \mu + \sqrt{\log^2(1/\delta') + 8\mu \log(1/\delta')} \\ &= \mu \left(1 + \sqrt{\frac{\log^2(1/\delta')}{\mu^2} + \frac{8\log(1/\delta')}{\mu}}\right) \\ &\leq \mu \left(1 + \sqrt{\log^2(1/\delta') + 8\log(1/\delta')}\right) \\ &\leq 5\mu \log\left(1/\delta'\right) \end{aligned}$$

as desired. Note the last inequality follows because $\log(1/\delta') \geq 1$ by assumption. ■

Appendix J. Proof of Theorem 14

We first quote an accuracy bound for the averaged parameter from Zhang et al. (2012). They remark that the required assumptions hold in most common settings, including in linear and logistic regression as long as the data distribution satisfies standard regularity conditions.

Theorem 34 (Corollary 2 of Zhang et al. (2012)) *Let $\theta^*_{avg} = K^{-1}\sum_{j=1}^K \theta^*_j$, where θ^*_j are the empirical risk minimizers on partition j of a dataset of size B sampled i.i.d. from some distribution $\mathcal{P}$. Let $\theta^* = \operatorname{argmin}_{\theta\in\Theta} \mathbb{E}_{z\sim\mathcal{P}}[f_z(\theta)]$. Then under the assumption that f_z is m-strongly convex for all z, and satisfies the following smoothness conditions for all $\theta \in \Theta$:*

$$\mathbb{E}_{z\sim\mathcal{P}}\left[\|\nabla f_z(\theta)\|_2^8\right] \leq L^8, \;\; \mathbb{E}_{z\sim\mathcal{P}}\left[\left\|\left|\nabla^2 f_z(\theta) - \nabla^2 \mathbb{E}_{z\sim\mathcal{P}}\left[f_z(\theta)\right]\right|\right\|_2^8\right] \leq H^8,$$

and the Hessian matrix $\nabla^2 f_z(\cdot)$ is G-Lipschitz continuous for all z, then, for some constant c:

$$\mathbb{E}\left[\left\|\theta^*_{avg} - \theta^*\right\|_2^2\right] \leq \frac{2L^2}{m^2 B} + \frac{cK^2L^2}{m^4B^2}\left(H^2 \log d + \frac{L^2G^2}{m^2}\right) + \mathcal{O}\left(\frac{K}{B^2}\right) + \mathcal{O}\left(\frac{K^3}{B^3}\right)$$

Proof [Proof of Theorem 14] We first prove the unlearning guarantee. We note that the boosting of our algorithms (running multiple copies of algorithms and picking the best model for publishing) won't matter in our unlearning bounds. In fact, the unlearning guarantee holds for *any* set l of models learned by the algorithms because they have *all* sufficiently come close to their respective optimizers in each chunk. Hence, until we get to the proof of accuracy statement, we imagine the algorithms are run once. We will see how this boosting will be helpful to recover *high probability* accuracy guarantees from the accuracy bounds of Zhang et al. (2012) which are *in expectation*.

Fix a training dataset $\mathcal{D}$ of size n and a non-adaptively chosen update sequence $\mathcal{U} = (u_i)_i$. Similar to our previous proofs, we first recall a few notations (from Definition 11), as well as some new notations for our proof:

- $\{\mathcal{D}_i\}_{i\geq 0}$ for the sequence of updated datasets. We use n_i ($\geq n/2$) to denote the size of $\mathcal{D}_i$.
- $\{\boldsymbol{\mathcal{S}}_i = (\mathcal{S}_{ij})_{j=1}^K\}_{i\geq 0}$ for the sequence of partitioned subsampled datasets.
- $\{\hat{\boldsymbol{\theta}}_i = (\hat{\theta}_{ij})_{j=1}^K\}_{i\geq 0}$ for the sequence of learned parameters in each partition.
- $\{\hat{\theta}_{i,\text{avg}}\}_{i\geq 0}$ for the sequence of averaged learned parameters: $\hat{\theta}_{i,\text{avg}} = K^{-1}\sum_{j=1}^K \hat{\theta}_{ij}$.
- $\{\tilde{\theta}_i = f_{\text{publish}}(\hat{\boldsymbol{\theta}}_i) = \hat{\theta}_{i,\text{avg}} + Z_i\}_{i\geq 0}$ for the sequence of published parameters.
- $\{\theta^*_i\}_{i\geq 0}$ is the sequence of target optimizers: $\theta^*_i \triangleq \text{argmin}_\theta\, f_{\mathcal{D}_i}(\theta)$.
- $\{\boldsymbol{\theta}^*_i = (\theta^*_{ij})_{j=1}^K\}_{i\geq 0}$ is the sequence of optimizers for partitions: $\theta^*_{ij} \triangleq \text{argmin}_\theta\, f_{\mathcal{S}_{ij}}(\theta)$.
- $\{\theta^*_{i,\text{avg}}\}_{i\geq 0}$ is the average of optimizers for partitions: $\theta^*_{i,\text{avg}} = K^{-1}\sum_{j=1}^K \theta^*_{ij}$.
- $\{s_i\}_{i\geq 1}$ for the sequence of number of affected data points in the whole dataset, i.e., s_i shows how many points differ between $\boldsymbol{\mathcal{S}}_i$ and $\boldsymbol{\mathcal{S}}_{i-1}$. We will also make use of notation s_{ij} which shows how many points differ between $\mathcal{S}_{ij}$ and $\mathcal{S}_{i-1,j}$. Note that $s_i = \sum_{j=1}^K s_{ij}$.

Fact 35 *Let $\tilde{s}_i \triangleq \max_{l\leq i} s_l$. We have by Lemma 13 that for any i, with probability at least $1-\delta/2$ over the sampling randomness up to round i, $\tilde{s}_i \leq \frac{10B}{n}\log(2i/\delta)$. We condition on this high probability event throughout the proof.*

Fact 36 *We also work with general K and B for now and eventually we use the ones stated in the theorem. We note that for general K and B, we can write*

$$T \geq \frac{Kn^2\mathcal{I}}{B^2} + \frac{\log\left(DmL^{-1}B\left(1+10\log(2/\delta)\right)\right)}{\log(1/\gamma)}$$

and

$$T_i = 10\log(2i/\delta)\left(\mathcal{I} + \frac{B^2}{Kn^2}\cdot\frac{\log\left(1+10i\log(2i/\delta)\right)}{\log(1/\gamma)}\right)$$

Let T_i' be the number of iterations in affected partitions *on round i. We have that with probability at least $1-\delta/2$, by Fact 35,*

$$T_i' \geq \frac{Kn}{Bs_i} T_i \geq \frac{Kn^2}{10B^2 \log(2i/\delta)} T_i \geq \frac{\log(1+10i\log(2i/\delta))}{\log(1/\gamma)} + \frac{Kn^2\mathcal{I}}{B^2} \tag{19}$$

Fact 37 *We have that $B \geq n$, and $Kn^2 \geq B^2$ (note these are justified by the setting of these parameters in theorem statement). We will use these later on in the proof.*

For every $i \geq 1$, let $\boldsymbol{\mathcal{S}}_i'$ be the partitioned dataset we would have had we retrained (using our learning algorithm $\mathcal{A}$) on dataset $\mathcal{D}_i$, and note that by Lemma 12, $\boldsymbol{\mathcal{S}}_i'$ and $\boldsymbol{\mathcal{S}}_i$ are distributed identically. To apply Lemma 12 we have used the fact that $\mathcal{U}$ is a non-adaptive sequence of updates selected independently of any internal randomness of $\mathcal{R}_{\mathcal{A}}$. Now let $\mathcal{C}_i$ be a coupling of the pair $(\boldsymbol{\mathcal{S}}_i', \boldsymbol{\mathcal{S}}_i)$ such that $\boldsymbol{\mathcal{S}}_i' = \boldsymbol{\mathcal{S}}_i$ with probability one. Throughout the proof when we condition on any of $\boldsymbol{\mathcal{S}}_i'$ or $\boldsymbol{\mathcal{S}}_i$ being drawn from their distribution, we will think of these datasets being drawn from the coupling $\mathcal{C}_i$ so that we are always guaranteed $\boldsymbol{\mathcal{S}}_i' = \boldsymbol{\mathcal{S}}_i$. Let's start proving the unlearning guarantees. For any $i \geq 0$, conditioned on the draw of $\boldsymbol{\mathcal{S}}_i'$, we have that $f_{\text{publish}}(\mathcal{A}(\mathcal{D}_i)) \sim \mathcal{N}\left(\mu_i, \sigma^2 \mathbb{I}_d\right)$, where $\mu_i = K^{-1}\sum_{j=1}^K \mu_{ij}$ and that it follows by the convergence guarantee of Theorem 20 that, for all partitions j,

$$\left\|\mu_{ij} - \theta_{ij}^*\right\|_2 \leq \gamma^T \left\|\theta_0' - \theta_{ij}^*\right\|_2 \leq \frac{4L\gamma^{\frac{Kn^2\mathcal{I}}{B^2}} \left\|\theta_0' - \theta_{ij}^*\right\|_2}{DmB(1+10\log(2/\delta))} \leq \frac{4L}{mB(1+10\log(2/\delta))} \cdot \gamma^{\frac{Kn^2\mathcal{I}}{B^2}} \tag{20}$$

We also have that for any update step $i \geq 1$, with probability at least $1-\delta/2$ over the randomness up to step i (draws of all $\boldsymbol{\mathcal{S}}_l$ for all $l \leq i$), $f_{\text{publish}}\left(\mathcal{R}_{\mathcal{A}}\left(\boldsymbol{\mathcal{S}}_{i-1}, u_i, \boldsymbol{\theta}_i\right)\right) \sim \mathcal{N}\left(\mu_i', \sigma^2\mathbb{I}_d\right)$, where we first observe that $\mu_i' = \hat{\theta}_{i,\text{avg}} = K^{-1}\sum_{j=1}^K \hat{\theta}_{ij}$, and furthermore,

$$\forall\, j;\ \left\|\hat{\theta}_{ij} - \theta_{ij}^*\right\|_2 \leq \frac{4LK\left(K^{-1} + \sum_{l\leq i} s_{lj}\right)}{mB(1+10i\log(2i/\delta))} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1-\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}} \tag{21}$$

We use induction on i to prove this claim. Let's focus on the base case $i=1$. For any partition j such that $s_{1j} = 0$, because the update algorithm doesn't make any updates, we have

$$\begin{aligned}
\left\|\hat{\theta}_{1j} - \theta_{1j}^*\right\|_2 &= \left\|\hat{\theta}_{0j} - \theta_{0j}^*\right\|_2 \\
&\leq \frac{4L}{mB(1+10\log(2/\delta))} \cdot \gamma^{\frac{Kn^2\mathcal{I}}{B^2}} \\
&\leq \frac{4LK\left(K^{-1} + s_{1j}\right)}{mB(1+10\log(2/\delta))} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1-\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}
\end{aligned}$$

because note that $\hat{\theta}_{0j} \equiv \mu_{0j}$ and therefore, we can use Equation (20) for $i=0$. For any partition j such that $s_{1j} \neq 0$, the update algorithm makes update, and in particular runs for T_1' iterations. We

therefore have that

$$\begin{aligned}\left\|\hat{\theta}_{1j}-\theta^*_{1j}\right\|_2 &\le \gamma^{T'_1}\left\|\hat{\theta}_{0j}-\theta^*_{1j}\right\|_2\\ &\le \gamma^{T'_1}\left(\left\|\hat{\theta}_{0j}-\theta^*_{0j}\right\|_2+\left\|\theta^*_{0j}-\theta^*_{1j}\right\|_2\right)\\ &\le \gamma^{T'_1}\left(\frac{4L}{mB}\cdot\frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1-\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}+\frac{4LKs_{1j}}{mB}\right)\\ &\le \frac{4LK\left(K^{-1}+s_{1j}\right)}{mB\left(1+10\log\left(2/\delta\right)\right)}\cdot\frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1-\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}\end{aligned}$$

The first inequality follows from the convergence guarantee of Theorem 20 and the fact that on round $i+1$ of update, the gradient descent of chunk j is initialized at $\hat{\theta}_{0j}$. The second inequality is a triangle inequality and the third follows from Equation (20) for $i=0$ (note $\hat{\theta}_{0j}\equiv\mu_{0j}$), and the sensitivity Lemma 8 (note that we apply this Lemma $2s_{1j}$ times and that the size of each chunk is B/K). The last inequality follows from Equation (19). Now let's focus on the induction step of the argument. Suppose Equation (21) holds for some $i\ge 1$. We will show that it holds for $i+1$ as well. For any partition j such that $s_{i+1,j}=0$, we have $\|\hat{\theta}_{i+1,j}-\theta^*_{i+1,j}\|_2=\|\hat{\theta}_{i,j}-\theta^*_{i,j}\|_2$ and the claim holds by induction assumption. Now suppose $s_{i+1,j}\neq 0$ which implies the update algorithm runs T'_{i+1} iterations of gradient descent on chunk j. We therefore have that, similar to how we proceed for $i=1$ case above,

$$\begin{aligned}\left\|\hat{\theta}_{i+1,j}-\theta^*_{i+1,j}\right\|_2 &\le \gamma^{T'_{i+1}}\left\|\hat{\theta}_{ij}-\theta^*_{i+1,j}\right\|_2\\ &\le \gamma^{T'_{i+1}}\left(\left\|\hat{\theta}_{ij}-\theta^*_{ij}\right\|_2+\left\|\theta^*_{ij}-\theta^*_{i+1,j}\right\|_2\right)\\ &\le \gamma^{T'_{i+1}}\left(\frac{4LK\left(K^{-1}+\sum_{l\le i}s_{lj}\right)}{mB}\cdot\frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1-\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}+\frac{4LKs_{i+1,j}}{mB}\right)\\ &\le \frac{4LK\left(K^{-1}+\sum_{l\le i+1}s_{lj}\right)}{mB\left(1+10(i+1)\log\left(2(i+1)/\delta\right)\right)}\cdot\frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1-\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}\end{aligned}$$

where the third inequality follows from induction assumption for i and applying the sensitivity Lemma 8 $2s_{i+1,j}$ times, and the last inequality follows from Equation (19). This completes the induction proof. Now we can use Equations (20) and (21) to conclude that for all $i\ge 0$,

$$\left\|\mu_i-\theta^*_{i,\text{avg}}\right\|\le\frac{1}{K}\sum_{j=1}^{K}\left\|\mu_{ij}-\theta^*_{ij}\right\|_2\le\frac{4L}{mB\left(1+10\log\left(2/\delta\right)\right)}\cdot\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}\le\frac{4L}{mn}\cdot\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}\qquad(22)$$

where we use the fact that $B \geq n$. And with probability at least $1 - \delta/2$, for all $i \geq 1$,

$$\begin{aligned}
\left\|\mu_i' - \theta^*_{i,\text{avg}}\right\| &\leq \frac{1}{K}\sum_{j=1}^{K}\left\|\hat{\theta}_{ij} - \theta^*_{ij}\right\|_2 \\
&\leq \frac{4L}{mB\left(1 + 10i\log\left(2i/\delta\right)\right)} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1 - \gamma^{\frac{Kn^2\mathcal{I}}{B^2}}} \sum_{j=1}^{K}\left(K^{-1} + \sum_{l \leq i} s_{lj}\right) \\
&= \frac{4L}{mB\left(1 + 10i\log\left(2i/\delta\right)\right)} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1 - \gamma^{\frac{Kn^2\mathcal{I}}{B^2}}} \left(1 + \sum_{l \leq i} s_l\right) \quad \left(\text{because } \sum_j s_{lj} = s_l\right) \\
&\leq \frac{4L}{mB\left(1 + 10i\log\left(2i/\delta\right)\right)} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1 - \gamma^{\frac{Kn^2\mathcal{I}}{B^2}}} \left(1 + i\tilde{s}_i\right) \quad \left(\text{recall } \tilde{s}_i = \max_{l \leq i} s_l\right) \\
&\leq \frac{4L}{mB\left(1 + 10i\log\left(2i/\delta\right)\right)} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1 - \gamma^{\frac{Kn^2\mathcal{I}}{B^2}}} \left(1 + i\frac{10B}{n}\log\left(2i/\delta\right)\right) \\
&= \frac{4L}{mn\left(1 + 10i\log\left(2i/\delta\right)\right)} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1 - \gamma^{\frac{Kn^2\mathcal{I}}{B^2}}} \left(\frac{n}{B} + 10i\log\left(2i/\delta\right)\right) \\
&\leq \frac{4L}{mn\left(1 + 10i\log\left(2i/\delta\right)\right)} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1 - \gamma^{\frac{Kn^2\mathcal{I}}{B^2}}} \left(1 + 10i\log\left(2i/\delta\right)\right) \quad \left(\text{because } B \geq n\right) \\
&= \frac{4L}{mn} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1 - \gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}
\end{aligned} \tag{23}$$

implying that for any $i \geq 1$, conditioned on the event that $\left\{\tilde{s}_i \leq 10Bn^{-1}\log\left(2i/\delta\right)\right\}$ which holds with probability at least $1 - \delta/2$ (by Fact 35),

$$\left\|\mu_i - \mu_i'\right\|_2 \leq \frac{8L}{mn} \cdot \frac{\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{1 - \gamma^{\frac{Kn^2\mathcal{I}}{B^2}}} \triangleq \Delta \tag{24}$$

It then follows from Lemma 24, as well as the choice of

$$\sigma = \frac{4\sqrt{2}L\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}}{mn\left(1 - \gamma^{\frac{Kn^2\mathcal{I}}{B^2}}\right)\left(\sqrt{\log\left(2/\delta\right) + \epsilon} - \sqrt{\log\left(2/\delta\right)}\right)}$$

in the theorem statement, that for any $i \geq 1$, with probability at least $1 - \delta/2$, $f_{\text{publish}}\left(\mathcal{A}\left(\mathcal{D}_i\right)\right) \overset{\epsilon,\delta/2}{\approx} f_{\text{publish}}\left(\mathcal{R}_{\mathcal{A}}\left(\boldsymbol{\mathcal{S}}_{i-1}, u_i, \boldsymbol{\theta}_i\right)\right)$. Now we can apply Lemma 22 to conclude that for any $i \geq 1$,

$$f_{\text{publish}}\left(\mathcal{A}\left(\mathcal{D}_i\right)\right) \overset{\epsilon,\delta}{\approx} f_{\text{publish}}\left(\mathcal{R}_{\mathcal{A}}\left(\boldsymbol{\mathcal{S}}_{i-1}, u_i, \boldsymbol{\theta}_i\right)\right)$$

And this shows $\mathcal{R}_{\mathcal{A}}$ is an (ϵ, δ)-unlearning algorithm for $\mathcal{A}$, as desired.

Now let's prove the accuracy statement of the theorem for which we will make use of Equations (22) and (23) (which holds with probability $1-\delta$). Recall that $\hat{\theta}_{0,\text{avg}} \equiv \mu_0$ and $\hat{\theta}_{i,\text{avg}} \equiv \mu_i'$ for $i \geq 1$. We first state the accuracy in expectation and then finally will turn those into high probability accuracy guarantees. First, we have that by a simple application of Cauchy-Schwarz inequality,

$$\begin{aligned}
\mathbb{E}\left\|\hat{\theta}_{i,\text{avg}} - \theta_i^*\right\|_2^2 &= \mathbb{E}\left\|\hat{\theta}_{i,\text{avg}} - \theta_{i,\text{avg}}^* + \theta_{i,\text{avg}}^* - \theta_i^*\right\|_2^2 \\
&= \mathbb{E}\left\|\hat{\theta}_{i,\text{avg}} - \theta_{i,\text{avg}}^*\right\|_2^2 + \mathbb{E}\left\|\theta_{i,\text{avg}}^* - \theta_i^*\right\|_2^2 + 2\mathbb{E}\left(\hat{\theta}_{i,\text{avg}} - \theta_{i,\text{avg}}^*\right)^\top \left(\theta_{i,\text{avg}}^* - \theta_i^*\right) \\
&\leq \mathbb{E}\left\|\hat{\theta}_{i,\text{avg}} - \theta_{i,\text{avg}}^*\right\|_2^2 + \mathbb{E}\left\|\theta_{i,\text{avg}}^* - \theta_i^*\right\|_2^2 + \sqrt{\mathbb{E}\left\|\hat{\theta}_{i,\text{avg}} - \theta_{i,\text{avg}}^*\right\|_2^2 \mathbb{E}\left\|\theta_{i,\text{avg}}^* - \theta_i^*\right\|_2^2}
\end{aligned} \tag{25}$$

but, by an application of law of total expectation (to turn the high probability guarantees into bounds in expectation),

$$\mathbb{E}\left\|\hat{\theta}_{i,\text{avg}} - \theta_{i,\text{avg}}^*\right\|_2^2 \leq \frac{16L^2}{m^2n^2} \cdot \frac{\gamma^{\frac{2Kn^2\mathcal{I}}{B^2}}}{\left(1-\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}\right)^2} + \delta D^2 \tag{26}$$

and we also know by Theorem 34 that, for some constant c, and for the choice of $K = \sqrt{B}$,

$$\begin{aligned}
\mathbb{E}\left\|\theta_{i,\text{avg}}^* - \theta_i^*\right\|_2^2 &\leq \frac{2L^2}{m^2B} + \frac{cL^2K^2}{m^4B^2}\left(H^2\log d + \frac{L^2G^2}{m^2}\right) + \mathcal{O}\left(\frac{K}{B^2}\right) + \mathcal{O}\left(\frac{K^3}{B^3}\right) \\
&= \frac{2L^2}{m^2B} + \frac{cL^2}{m^4B}\left(H^2\log d + \frac{L^2G^2}{m^2}\right) + \mathcal{O}\left(B^{-\frac{3}{2}}\right) \\
&= \frac{1}{B}\left(\frac{2L^2}{m^2} + \frac{cL^2}{m^4}\left(H^2\log d + \frac{L^2G^2}{m^2}\right)\right) + \mathcal{O}\left(B^{-\frac{3}{2}}\right)
\end{aligned} \tag{27}$$

Putting together Equations (25) and (26) (with $K = \sqrt{B}$) and (27), and noting that for $\delta = \mathcal{O}(B^{-1})$ and $B \geq n$ we have $\sqrt{\mathbb{E}\|\hat{\theta}_{i,\text{avg}} - \theta_{i,\text{avg}}^*\|_2^2 \cdot \mathbb{E}\|\theta_{i,\text{avg}}^* - \theta_i^*\|_2^2} = \mathcal{O}(\log d/B)$, and hiding all constants under the $\mathcal{O}$ notation, we have

$$\mathbb{E}\left\|\hat{\theta}_{i,\text{avg}} - \theta_i^*\right\|_2^2 = \mathcal{O}\left(\frac{\gamma^{\frac{n^2\mathcal{I}}{B\sqrt{B}}}}{n^2\left(1-\gamma^{\frac{n^2\mathcal{I}}{B\sqrt{B}}}\right)^2}\right) + \mathcal{O}\left(\frac{\log d}{B}\right) + \mathcal{O}\left(\frac{1}{B^{\frac{3}{2}}}\right)$$

Now by Lemma 25, we have that by running the algorithm for $C = \log(2/\beta)/\log 2$ times and picking the best model with smallest loss (note by strong convexity, the smaller the loss of a model is, the closer the model parameter is to the optimizer. Also for notational convenience, we still use $\hat{\theta}_{i,\text{avg}}$ for the best model), with probability at least $1-\beta/2$,

$$\left\|\hat{\theta}_{i,\text{avg}} - \theta_i^*\right\|_2^2 = \mathcal{O}\left(\frac{\gamma^{\frac{n^2\mathcal{I}}{B\sqrt{B}}}}{n^2\left(1-\gamma^{\frac{n^2\mathcal{I}}{B\sqrt{B}}}\right)^2}\right) + \mathcal{O}\left(\frac{\log d}{B}\right) + \mathcal{O}\left(\frac{1}{B^{\frac{3}{2}}}\right) \tag{28}$$

Recall that at any given round $i \geq 0$, the published model $\tilde{\theta}_i = \hat{\theta}_{i,\text{avg}} + Z_i$. We therefore have that by Equation (28), a Gaussian tail bound (Lemma 23), choice of σ in the theorem statement, and the fact that for $\epsilon = \mathcal{O}\left(\log\left(1/\delta\right)\right)$, we have $\sqrt{\log\left(1/\delta\right) + \epsilon} - \sqrt{\log\left(1/\delta\right)} = \Omega(\epsilon/\sqrt{\log\left(1/\delta\right)})$, with probability at least $1 - \beta$,

$$\left\|\tilde{\theta}_i - \theta_i^*\right\|_2^2 = \mathcal{O}\left(\frac{L^2\gamma^{\frac{2n^2\mathcal{I}}{B\sqrt{B}}} d\log\left(1/\delta\right)\log^2\left(d/\beta\right)}{m^2\epsilon^2 n^2\left(1-\gamma^{\frac{Kn^2\mathcal{I}}{B^2}}\right)^2}\right) + \mathcal{O}\left(\frac{\log d}{B}\right) + \mathcal{O}\left(\frac{1}{B^{\frac{3}{2}}}\right) \tag{29}$$

Note that $(1-\gamma^a)^{-1} \leq (1-\gamma)^{-1}$ for any $a \geq 1$ (in our case $a = \frac{Kn^2\mathcal{I}}{B^2} \geq 1$). The proof is complete by the choice of $B = n^\xi$ and M-smoothness of f:

$$f_{\mathcal{D}_i}(\tilde{\theta}_i) - f_{\mathcal{D}_i}(\theta_i^*) \leq \frac{M}{2}\left\|\tilde{\theta}_i - \theta_i^*\right\|_2^2$$

■