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

Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi

Presented by Ananth Mahadevan

March 16, 2021

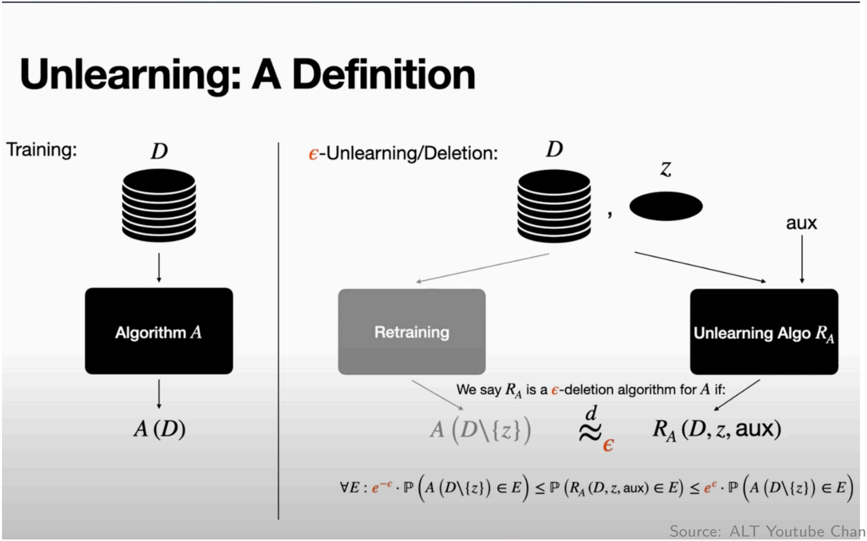
Contents

Data Removal

- Right to be Forgotten and GDPR
- Deleting personal data from datasets
- How to remove influence of data on deployed ML models? Retrain them on remaining sample?
- Retraining effort is disproportionate to number of deletion requests

Problem Statement

Design an efficient **unlearning algorithm** that produces model outputs that are **statistically indistinguishable** from the model outputs that would have arisen from **retraining**



Source: ALT Youtube Channel

Preliminaries

- Original dataset $\mathcal{D}_0$ and after ith removal $\mathcal{D}_i$

Preliminaries

- Original dataset $\mathcal{D}_0$ and after ith removal $\mathcal{D}_i$
- **Learning** algorithm $\mathcal{A}$ and $\mathcal{A}(\mathcal{D}_0) = \theta_0$

Preliminaries

- Original dataset $\mathcal{D}_0$ and after ith removal $\mathcal{D}_i$
- **Learning** algorithm $\mathcal{A}$ and $\mathcal{A}(\mathcal{D}_0) = \theta_0$
- **Unlearning** algorithm for $\mathcal{A}$, $\mathcal{R}_\mathcal{A}$

Preliminaries

- Original dataset $\mathcal{D}_0$ and after ith removal $\mathcal{D}_i$
- **Learning** algorithm $\mathcal{A}$ and $\mathcal{A}(\mathcal{D}_0) = \theta_0$
- **Unlearning** algorithm for $\mathcal{A}$, $\mathcal{R}_{\mathcal{A}}$
- For $i \geq 1$ and removal z_i, $\mathcal{R}_{\mathcal{A}}(\mathcal{D}_{i-1}, z_i, \theta_i) = \hat{\theta}_i$

Preliminaries

- Original dataset $\mathcal{D}_0$ and after ith removal $\mathcal{D}_i$
- **Learning** algorithm $\mathcal{A}$ and $\mathcal{A}(\mathcal{D}_0) = \theta_0$
- **Unlearning** algorithm for $\mathcal{A}$, $\mathcal{R}_{\mathcal{A}}$
- For $i \geq 1$ and removal z_i, $\mathcal{R}_{\mathcal{A}}(\mathcal{D}_{i-1}, z_i, \theta_i) = \hat{\theta}_i$

Preliminaries

- Original dataset $\mathcal{D}_0$ and after ith removal $\mathcal{D}_i$
- **Learning** algorithm $\mathcal{A}$ and $\mathcal{A}(\mathcal{D}_0) = \theta_0$
- **Unlearning** algorithm for $\mathcal{A}$, $\mathcal{R}_\mathcal{A}$
- For $i \geq 1$ and removal z_i, $\mathcal{R}_\mathcal{A}(\mathcal{D}_{i-1}, z_i, \theta_i) = \hat{\theta}_i$
- **Publishing function** f_{publish} adds i.i.d gaussian noise with 0 mean and σ variance

Preliminaries

- Original dataset $\mathcal{D}_0$ and after ith removal $\mathcal{D}_i$
- **Learning** algorithm $\mathcal{A}$ and $\mathcal{A}(\mathcal{D}_0) = \theta_0$
- **Unlearning** algorithm for $\mathcal{A}$, $\mathcal{R}_\mathcal{A}$
- For $i \geq 1$ and removal z_i, $\mathcal{R}_\mathcal{A}(\mathcal{D}_{i-1}, z_i, \theta_i) = \hat{\theta}_i$
- **Publishing function** f_{publish} adds i.i.d gaussian noise with 0 mean and σ variance
- $f_{\text{publish}}(\theta_0) = \tilde{\theta}_0$ and $f_{\text{publish}}(\hat{\theta}_i) = \tilde{\theta}_i$ when $i \geq 1$

Preliminaries

- Original dataset $\mathcal{D}_0$ and after ith removal $\mathcal{D}_i$
- **Learning** algorithm $\mathcal{A}$ and $\mathcal{A}(\mathcal{D}_0) = \theta_0$
- **Unlearning** algorithm for $\mathcal{A}$, $\mathcal{R}_\mathcal{A}$
- For $i \geq 1$ and removal z_i, $\mathcal{R}_\mathcal{A}(\mathcal{D}_{i-1}, z_i, \theta_i) = \hat{\theta}_i$
- **Publishing function** f_{publish} adds i.i.d gaussian noise with 0 mean and σ variance
- $f_{\text{publish}}(\theta_0) = \tilde{\theta}_0$ and $f_{\text{publish}}(\hat{\theta}_i) = \tilde{\theta}_i$ when $i \geq 1$
- Simply, $\{\hat{\theta}_i\}_{i\geq 1}$ are **secret** model outputs and $\{\tilde{\theta}_i\}_{i\geq 0}$ are **public** models

Perfect vs *Imperfect*? Unlearning

Perfect Unlearning Algorithm

- Requires indistinguishable wrt **full internal state**
- **Stronger** requirement, similar to *pan privacy* in differential-privacy
- Unlearning algorithm $\mathcal{R}_{\mathcal{A}}$ uses **published model as input** at each step, i.e. $\theta_i = \tilde{\theta}_{i-1}$
- All prior work focus on perfect unlearning algorithms

Perfect vs *Imperfect*? Unlearning

Imperfect? Unlearning

- Requires only statistical indistinguishability wrt **observed outputs** of algorithm
- Allowed to maintain a "**secret state**" for unlearning
- Secret state need not satisfy indistinguishability requirement
- Unlearning algorithm $\mathcal{R}_{\mathcal{A}}$ maintains previous step's output $\hat{\theta}_{i-1}$ as secret state
- This is used as input at the current step i, i.e. $\theta_i = \theta_{i-1}\hat{}$

Strong and Weak Unlearning

Strong Unlearning Algorithm

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.

Strong and Weak Unlearning

Strong Unlearning Algorithm

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.

Weak Unlearning Algorithm

It may have run-time per update (or equivalently, error) that grows polynomially with the length of the update sequence

Empirical Risk Minimization Framework

- Convex loss function $\ell : \mathbb{R}^d \times Z \to \mathbb{R}$.
- Dataset $D = \{z_1, z_2, \ldots, z_n\} \in Z^n$
- Want to solve: $\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(\theta, z_i) := \mathcal{L}(\theta, D)$
- Algorithm: Gradient Descent: $\theta_0, \forall t \leq T : \theta_t = \theta_{t-1} - \eta \nabla L(\theta_{t-1}, D)$
- Computation cost: number of iterations T
- Accuracy: $\mathcal{L}(\theta_T, D) - \min_\theta \mathcal{L}(\theta, D)$

Covergence Results for Gradient Descent

Strongly Convex and Smooth

Let ℓ be m-strongly convex and M smooth, and let $\theta^* = \text{argmin}_{\theta \in \Theta} \mathcal{L}(\theta)$. We have that after T steps of GD with step size $\eta_t = \frac{2}{m+M}$,

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

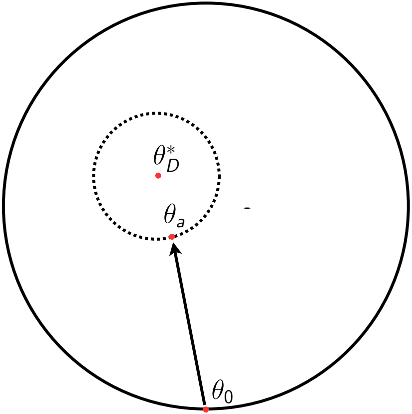
Differential Privacy Sensitivity

Sensitivity

Suppose ℓ is L -Lipschitz and m -strongly convex. For any dataset $\mathcal{D}$, let $\theta^*_{\mathcal{D}} \triangleq \text{argmin}_{\theta \in \Theta} \mathcal{L}(\theta)$. We have that for any integer n, any data set $\mathcal{D}$ of size n, and any removal $z \in \mathcal{D}$,

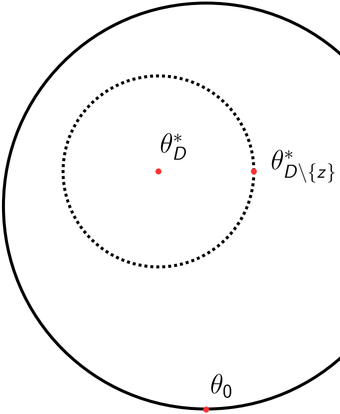
$$\left\| \theta^*_{\mathcal{D}} - \theta^*_{\mathcal{D} \setminus \{z\}} \right\|_2 \leq \frac{2L}{mn}.$$

Perturbed Gradient Descent



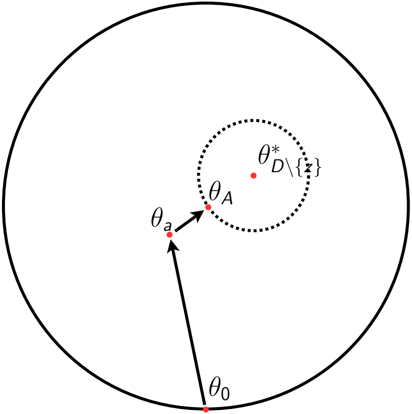
Source: Chris Waites' Blog

Perturbed Gradient Descent



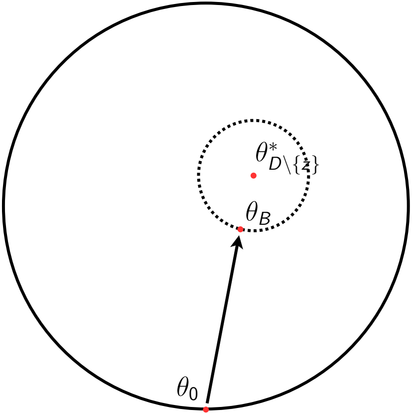
Source: Chris Waites' Blog

Perturbed Gradient Descent



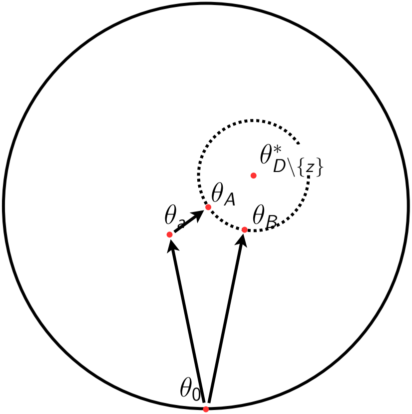
Source: Chris Waites' Blog

Perturbed Gradient Descent



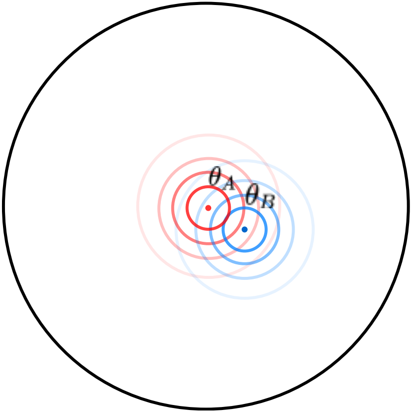
Source: Chris Waites' Blog

Perturbed Gradient Descent



Source: Chris Waites' Blog

Perturbed Gradient Descent



Source: Chris Waites' Blog

Regularized Perturbed Gradient Descent

- When ℓ is not strongly convex, can regularize it to enforce strong convexity
- Regularized loss: $\mathcal{L}_m(\theta, D) = \mathcal{L}(\theta, D) + \frac{m}{2}\|\theta\|_2^2$
- Issues with aggressive regularization
  - $m \uparrow \longrightarrow$ sensitivity $\downarrow \longrightarrow$ perturbation $\downarrow \longrightarrow$ accuracy $\uparrow$
  - $m \uparrow \longrightarrow L(\theta_T, D) \uparrow \longrightarrow$ accuracy $\downarrow$
  - $m \uparrow \longrightarrow$ Degrades Lipschitz/smoothness of loss functions
- Therefore regularization needs to be chosen carefully

Perturbed Distributed Descent

High Level Idea

- Randomly partition dataset into K parts
- Train models separately on each part to minimize empirical loss
- Take the average of K models
- Publish average model after adding noise

Benefits

- [ZDW13] provides *out of sample guarantee* for accuracy of distributed setting
- Removed element present in some partition, update only those partition
- Improved results given same computation budget as non-distributed

Results

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.

Future Directions

- Extensive experiential analysis to check practical feasibility
- Extend approach to SGD with similar analysis in [WDD20]
- Check for local convexity in Neural Network loss landscapes and extend approach

References I

Yinjun Wu, Edgar Dobriban, and Susan B. Davidson.
DeltaGrad: Rapid retraining of machine learning models.
arXiv:2006.14755 [cs, stat], June 2020.

Yuchen Zhang, John C. Duchi, and Martin J. Wainwright.
Communication-efficient algorithms for statistical optimization.
Journal of Machine Learning Research, 14(68):3321–3363, 2013.