Trustworthy Machine Learning

Unlearning 1

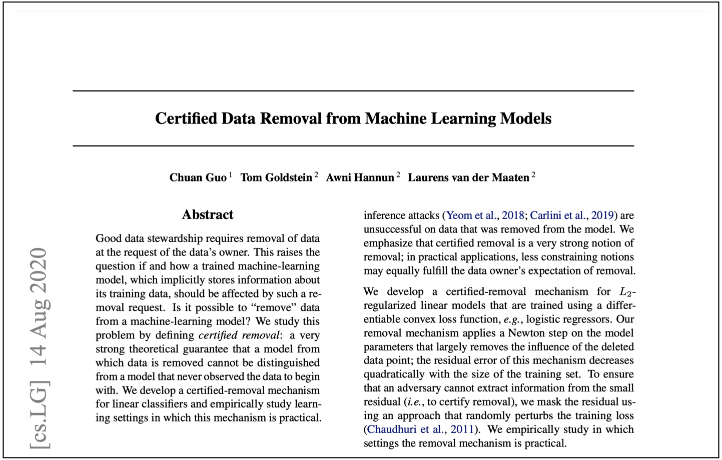
Sangdon Park

POSTECH

Why Unlearning?

- Remove sensitive or private data from a trained model
- Remove data for data-poisoning attacks
- ...

Certified Removal with A Linear Assumption



Certified Data Removal from Machine Learning Models

Chuan Guo [1] **Tom Goldstein** [2] **Awni Hannun** [2] **Laurens van der Maaten** [2]

Abstract

Good data stewardship requires removal of data at the request of the data's owner. This raises the question if and how a trained machine-learning model, which implicitly stores information about its training data, should be affected by such a removal request. Is it possible to "remove" data from a machine-learning model? We study this problem by defining *certified removal*: a very strong theoretical guarantee that a model from which data is removed cannot be distinguished from a model that never observed the data to begin with. We develop a certified-removal mechanism for linear classifiers and empirically study learning settings in which this mechanism is practical.

inference attacks (Yeom et al., 2018; Carlini et al., 2019) are unsuccessful on data that was removed from the model. We emphasize that certified removal is a very strong notion of removal; in practical applications, less constraining notions may equally fulfill the data owner's expectation of removal.

We develop a certified-removal mechanism for L_2-regularized linear models that are trained using a differentiable convex loss function, *e.g.*, logistic regressors. Our removal mechanism applies a Newton step on the model parameters that largely removes the influence of the deleted data point; the residual error of this mechanism decreases quadratically with the size of the training set. To ensure that an adversary cannot extract information from the small residual (*i.e.*, to certify removal), we mask the residual using an approach that randomly perturbs the training loss (Chaudhuri et al., 2011). We empirically study in which settings the removal mechanism is practical.

- ICML20
- The term "machine unlearning" seems to firstly appear in Cao and Yang [2015].
- Quite early (and I guess the first) certified removal work.

Overview on Key Ideas

1. Desire to remove a labeled example from a trained parametric model.

Overview on Key Ideas

1. Desire to remove a labeled example from a trained parametric model.
   - This task is trivial for non-parametric models.

Overview on Key Ideas

1. Desire to remove a labeled example from a trained parametric model.
   - This task is trivial for non-parametric models.
2. The parameter of the pre-trained model embeds the information of the target labeled example.

Overview on Key Ideas

1. Desire to remove a labeled example from a trained parametric model.
    - This task is trivial for non-parametric models.
2. The parameter of the pre-trained model embeds the information of the target labeled example.
3. How to remove unlearning this?

Overview on Key Ideas

1. Desire to remove a labeled example from a trained parametric model.
    - This task is trivial for non-parametric models.
2. The parameter of the pre-trained model embeds the information of the target labeled example.
3. How to remove unlearning this?
    - learning: gradient descent

Overview on Key Ideas

1. Desire to remove a labeled example from a trained parametric model.
   - This task is trivial for non-parametric models.
2. The parameter of the pre-trained model embeds the information of the target labeled example.
3. How to remove unlearning this?
   - learning: gradient descent
   - unlearning: gradient ascent

Overview on Key Ideas

1. Desire to remove a labeled example from a trained parametric model.
   - This task is trivial for non-parametric models.
2. The parameter of the pre-trained model embeds the information of the target labeled example.
3. How to remove unlearning this?
   - learning: gradient descent
   - unlearning: gradient ascent
4. Gradient "ascent" almost remove this information, but not perfect.

Overview on Key Ideas

1. Desire to remove a labeled example from a trained parametric model.
   - This task is trivial for non-parametric models.
2. The parameter of the pre-trained model embeds the information of the target labeled example.
3. How to remove unlearning this?
   - learning: gradient descent
   - unlearning: gradient ascent
4. Gradient "ascent" almost remove this information, but not perfect.
5. Let's also add noise to hide the remaining information.

Definition: ε-Certified Removal

Given $\varepsilon > 0$, we say that removal mechanism M performs ε-certified removal (ε-CR) for a learning algorithm A if $\forall \mathcal{T} \subseteq \mathcal{H}, \mathcal{D} \subseteq \mathcal{Z}, z \in \mathcal{D}$

$$e^{-\varepsilon} \leq \frac{\mathbb{P}\{M(A(\mathcal{D}), \mathcal{D}, z) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D} \setminus z) \in \mathcal{T}\}} \leq e^{\varepsilon},$$

where the probability is taken over the randomness of A.

- ε: an unlearning parameter.
- $\mathcal{X}$: example space
- $\mathcal{Y}$: label space
- $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$
- $\mathcal{D} \subseteq \mathcal{Z}$: a training set
- $\mathcal{H}$: a hypothesis set
- $A : \mathcal{D} \to \mathcal{H}$: a (randomized) learning algorithm
- In short, we wish to have $M(A(\mathcal{D}), \mathcal{D}, z) \approx A(\mathcal{D} \setminus z)$

Definition: (ε, δ)-Certified Removal

Given $\varepsilon, \delta > 0$, we say that removal mechanism M performs (ε, δ)-certified removal for a learning algorithm A if $\forall \mathcal{T} \subseteq \mathcal{H}, \mathcal{D} \subseteq \mathcal{Z}, z \in \mathcal{D}$

$$\mathbb{P}\{M(A(\mathcal{D}), \mathcal{D}, z) \in \mathcal{T}\} \leq e^{\varepsilon} \mathbb{P}\{A(\mathcal{D} \setminus z) \in \mathcal{T}\} + \delta \quad \text{and}$$
$$\mathbb{P}\{A(\mathcal{D} \setminus z) \in \mathcal{T}\} \leq e^{\varepsilon} \mathbb{P}\{M(A(\mathcal{D}), \mathcal{D}, z) \in \mathcal{T}\} + \delta.$$

- δ upper bounds the failure probability of

$$e^{-\varepsilon} \leq \frac{\mathbb{P}\{M(A(\mathcal{D}), \mathcal{D}, z) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D} \setminus z) \in \mathcal{T}\}} \leq e^{\varepsilon},$$

Candidate Method: Exact Removal

Exact Removal

$$M(A(\mathcal{D}), \mathcal{D}, z) := A(\mathcal{D} \setminus z)$$

- M is trivially 0-CR.

$$e^0 \leq \frac{\mathbb{P}\{A(\mathcal{D} \setminus z) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D} \setminus z) \in \mathcal{T}\}} \leq e^0,$$

Candidate Method: Exact Removal

Exact Removal

$$M(A(\mathcal{D}), \mathcal{D}, z) := A(\mathcal{D} \setminus z)$$

- M is trivially 0-CR.

$$e^0 \leq \frac{\mathbb{P}\{A(\mathcal{D} \setminus z) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D} \setminus z) \in \mathcal{T}\}} \leq e^0,$$

- Impractical as we need to retrain a model whenever a training sample is removed.

Candidate Method: Differential Privacy

Differential Privacy (DP)

A is ε-differentially private if for any $\mathcal{T} \subseteq \mathcal{H}$, $\mathcal{D}$, and $\mathcal{D}'$,

$$e^{-\varepsilon} \leq \frac{\mathbb{P}\{A(\mathcal{D}) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D}') \in \mathcal{T}\}} \leq e^{\varepsilon},$$

where $\mathcal{D}$ and $\mathcal{D}'$ differ in only one sample.

Candidate Method: Differential Privacy

Differential Privacy (DP)

A is ε-differentially private if for any $\mathcal{T} \subseteq \mathcal{H}$, $\mathcal{D}$, and $\mathcal{D}'$,

$$e^{-\varepsilon} \leq \frac{\mathbb{P}\{A(\mathcal{D}) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D}') \in \mathcal{T}\}} \leq e^{\varepsilon},$$

where $\mathcal{D}$ and $\mathcal{D}'$ differ in only one sample.

- The DP of A is a sufficient condition for certified removal by setting M as an identity function.

Candidate Method: Differential Privacy

Differential Privacy (DP)

A is ε-differentially private if for any $\mathcal{T} \subseteq \mathcal{H}$, $\mathcal{D}$, and $\mathcal{D}'$,

$$e^{-\varepsilon} \leq \frac{\mathbb{P}\{A(\mathcal{D}) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D}') \in \mathcal{T}\}} \leq e^{\varepsilon},$$

where $\mathcal{D}$ and $\mathcal{D}'$ differ in only one sample.

- The DP of A is a sufficient condition for certified removal by setting M as an identity function.
  - A never memorizes a training sample, so we don't need to worry about removing it.

Candidate Method: Differential Privacy

Differential Privacy (DP)

A is ε-differentially private if for any $\mathcal{T} \subseteq \mathcal{H}$, $\mathcal{D}$, and $\mathcal{D}'$,

$$e^{-\varepsilon} \leq \frac{\mathbb{P}\{A(\mathcal{D}) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D}') \in \mathcal{T}\}} \leq e^{\varepsilon},$$

where $\mathcal{D}$ and $\mathcal{D}'$ differ in only one sample.

- The DP of A is a sufficient condition for certified removal by setting M as an identity function.
  - A never memorizes a training sample, so we don't need to worry about removing it.
  - But, DP requires retraining and usually introduces poor performance.

Candidate Method: Differential Privacy

Differential Privacy (DP)

A is ε-differentially private if for any $\mathcal{T} \subseteq \mathcal{H}$, $\mathcal{D}$, and $\mathcal{D}'$,

$$e^{-\varepsilon} \leq \frac{\mathbb{P}\{A(\mathcal{D}) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D}') \in \mathcal{T}\}} \leq e^{\varepsilon},$$

where $\mathcal{D}$ and $\mathcal{D}'$ differ in only one sample.

- The DP of A is a sufficient condition for certified removal by setting M as an identity function.
  - A never memorizes a training sample, so we don't need to worry about removing it.
  - But, DP requires retraining and usually introduces poor performance.
- However, the DP of A is not a necessary condition for certified removal.
  - A nearest-neighbor classifier is not differentially private but it can be 0-CR.

Candidate Method: Differential Privacy

Differential Privacy (DP)

A is ε-differentially private if for any $\mathcal{T} \subseteq \mathcal{H}$, $\mathcal{D}$, and $\mathcal{D}'$,

$$e^{-\varepsilon} \leq \frac{\mathbb{P}\{A(\mathcal{D}) \in \mathcal{T}\}}{\mathbb{P}\{A(\mathcal{D}') \in \mathcal{T}\}} \leq e^{\varepsilon},$$

where $\mathcal{D}$ and $\mathcal{D}'$ differ in only one sample.

- The DP of A is a sufficient condition for certified removal by setting M as an identity function.
  - A never memorizes a training sample, so we don't need to worry about removing it.
  - But, DP requires retraining and usually introduces poor performance.
- However, the DP of A is not a necessary condition for certified removal.
  - A nearest-neighbor classifier is not differentially private but it can be 0-CR.
- "Retraining from scratch" and "DP" are two extreme removal methods (in removal efficiency).

Removal Mechanism: Setup

Removal Mechanism: Setup

- $\mathcal{D} := \{(x_1, y_1), \ldots, (x_n, y_n)\}$: a training set

Removal Mechanism: Setup

- $\mathcal{D} := \{(x_1, y_1), \ldots, (x_n, y_n)\}$: a training set
- Consider a linear classifier.

Removal Mechanism: Setup

- $\mathcal{D} := \{(x_1, y_1), \dots, (x_n, y_n)\}$: a training set
- Consider a linear classifier.
- Learning objective: regularized empirical risk minimization, *i.e.*,

$$L(w; \mathcal{D}) := \sum_{i=1}^{n} \ell(w^T x_i, y_i) + \frac{\lambda n}{2} \|w\|_2^2,$$

where ℓ is a convex loss function differentiable everywhere.

Removal Mechanism: Setup

- $\mathcal{D} := \{(x_1, y_1), \ldots, (x_n, y_n)\}$: a training set
- Consider a linear classifier.
- Learning objective: regularized empirical risk minimization, *i.e.*,

$$L(w; \mathcal{D}) := \sum_{i=1}^{n} \ell(w^T x_i, y_i) + \frac{\lambda n}{2} \|w\|_2^2,$$

  where ℓ is a convex loss function differentiable everywhere.
- w^*: an optimal and *unique* classifier, *i.e.*,

$$w^* := \arg\min_w L(w; \mathcal{D}) := A(\mathcal{D}).$$

Removal Mechanism: Setup

- $\mathcal{D} := \{(x_1, y_1), \ldots, (x_n, y_n)\}$: a training set
- Consider a linear classifier.
- Learning objective: regularized empirical risk minimization, *i.e.*,

$$L(w; \mathcal{D}) := \sum_{i=1}^{n} \ell(w^T x_i, y_i) + \frac{\lambda n}{2} \|w\|_2^2,$$

  where ℓ is a convex loss function differentiable everywhere.
- w^*: an optimal and *unique* classifier, *i.e.*,

$$w^* := \arg\min_w L(w; \mathcal{D}) := A(\mathcal{D}).$$

- Goal: given a training sample (x, y) to remove, find w^- such that

$$w^- \approx A(\mathcal{D} \setminus \{(x, y)\})$$

(Not Yet Certified) Removal Mechanism

Newton update removal mechanism M

$$w^- = M(w^*, \mathcal{D}, (x_n, y_n)) := w^* + H_{w^*}^{-1}\Delta$$

- WLOG, remove the last sample (x_n, y_n)
- $\Delta := \lambda w^* + \nabla \ell((w^*)^T x_n, y_n)$: the loss gradient at a sample (x_n, y_n)
- $H_{w^*} := \nabla^2 L(w^*; \mathcal{D}')$: the Hessian of $L(\cdot, \mathcal{D}')$ at w^*
- $H_{w^*}^{-1}\Delta$: a.k.a. the influence function of (x_n, y_n) on w^* [Koh and Liang, 2017]

(Not Yet Certified) Removal Mechanism

Newton update removal mechanism M

$$w^- = M(w^*, \mathcal{D}, (x_n, y_n)) := w^* + H_{w^*}^{-1}\Delta$$

- WLOG, remove the last sample (x_n, y_n)
- $\Delta := \lambda w^* + \nabla \ell((w^*)^T x_n, y_n)$: the loss gradient at a sample (x_n, y_n)
- $H_{w^*} := \nabla^2 L(w^*; \mathcal{D}')$: the Hessian of $L(\cdot, \mathcal{D}')$ at w^*
- $H_{w^*}^{-1}\Delta$: a.k.a. the influence function of (x_n, y_n) on w^* [Koh and Liang, 2017]
- Similar to the Newton's update, *i.e.*,

$$w_{t+1} = w_t - \left(\nabla^2 \ell(w_t^T x_n, y_n)\right)^{-1} \nabla \ell(w_t^T x_n, y_n)$$

  - The Newton's update: learn (x_n, y_n)
  - The Newton's update removal mechanism: unlearn (x_n, y_n)

(Not Yet Certified) Removal Mechanism

Newton update removal mechanism M

$$w^- = M(w^*, \mathcal{D}, (x_n, y_n)) := w^* + H_{w^*}^{-1}\Delta$$

- WLOG, remove the last sample (x_n, y_n)
- $\Delta := \lambda w^* + \nabla \ell((w^*)^T x_n, y_n)$: the loss gradient at a sample (x_n, y_n)
- $H_{w^*} := \nabla^2 L(w^*; \mathcal{D}')$: the Hessian of $L(\cdot, \mathcal{D}')$ at w^*
- $H_{w^*}^{-1}\Delta$: a.k.a. the influence function of (x_n, y_n) on w^* [Koh and Liang, 2017]
- Similar to the Newton's update, *i.e.*,

$$w_{t+1} = w_t - \left(\nabla^2 \ell(w_t^T x_n, y_n)\right)^{-1} \nabla \ell(w_t^T x_n, y_n)$$

  - The Newton's update: learn (x_n, y_n)
  - The Newton's update removal mechanism: unlearn (x_n, y_n)
- Why Newton? Fast removal (*i.e.*, removal with one step update)

Measuring the Failure of Unlearning

Gradient residual

$$\|\nabla L(w^{-}; \mathcal{D}')\|_2$$

- In the strongly convex setup, the gradient completely characterizes unlearning.

Measuring the Failure of Unlearning

Gradient residual

$$\|\nabla L(w^-; \mathcal{D}')\|_2$$

- In the strongly convex setup, the gradient completely characterizes unlearning.
- If $\nabla L(w^-; \mathcal{D}') = 0$, w^- is the unique minimizer of $L(\cdot; \mathcal{D}')$, implying successfully unlearned!

Measuring the Failure of Unlearning

Gradient residual

$$\|\nabla L(w^-; \mathcal{D}')\|_2$$

- In the strongly convex setup, the gradient completely characterizes unlearning.
- If $\nabla L(w^-; \mathcal{D}') = 0$, w^- is the unique minimizer of $L(\cdot; \mathcal{D}')$, implying successfully unlearned!
- This also means $\|\nabla L(w^-; \mathcal{D}')\|_2$ is the measure of "unlearning error".

Measuring the Failure of Unlearning

Gradient residual

$$\|\nabla L(w^-; \mathcal{D}')\|_2$$

- In the strongly convex setup, the gradient completely characterizes unlearning.
- If $\nabla L(w^-; \mathcal{D}') = 0$, w^- is the unique minimizer of $L(\cdot; \mathcal{D}')$, implying successfully unlearned!
- This also means $\|\nabla L(w^-; \mathcal{D}')\|_2$ is the measure of "unlearning error".
- Can we bound this quantity?

Bound on the Failure of Unlearning

Theorem

Suppose that $\|\nabla \ell(w^T x_i, y_i)\|_2 \leq C$ *for any* (x_i, y_i) *and* w, ℓ'' *is* γ*-Lipschitz, and* $\|x_i\|_2 \leq 1$ *for all* x_i. *Then, we have*

$$\|\nabla(w^-; \mathcal{D}')\|_2 \leq \frac{4\gamma C^2}{\lambda^2(n-1)}.$$

- logistic regression: $C = 1$ and $\gamma = \frac{1}{4}$ when $\ell(w^T x, y) = -\log \sigma(y w^T)$
- See the paper for the data-dependent bound.

Bound on the Failure of Unlearning

Theorem

Suppose that $\|\nabla \ell(w^T x_i, y_i)\|_2 \le C$ *for any* (x_i, y_i) *and* w, ℓ'' *is* γ*-Lipschitz, and* $\|x_i\|_2 \le 1$ *for all* x_i. *Then, we have*

$$\|\nabla(w^-; \mathcal{D}')\|_2 \le \frac{4\gamma C^2}{\lambda^2(n-1)}.$$

- logistic regression: $C = 1$ and $\gamma = \frac{1}{4}$ when $\ell(w^T x, y) = -\log \sigma(y w^T)$
- See the paper for the data-dependent bound.
- As $n \to \infty$, $\|\nabla(w^-; \mathcal{D}')\|_2 \to 0$. Is it enough?

Bound on the Failure of Unlearning

Theorem

Suppose that $\|\nabla \ell(w^T x_i, y_i)\|_2 \leq C$ *for any* (x_i, y_i) *and* w, ℓ'' *is* γ*-Lipschitz, and* $\|x_i\|_2 \leq 1$ *for all* x_i. *Then, we have*

$$\|\nabla(w^-; \mathcal{D}')\|_2 \leq \frac{4\gamma C^2}{\lambda^2(n-1)}.$$

- logistic regression: $C = 1$ and $\gamma = \frac{1}{4}$ when $\ell(w^T x, y) = -\log \sigma(y w^T)$
- See the paper for the data-dependent bound.
- As $n \to \infty$, $\|\nabla(w^-; \mathcal{D}')\|_2 \to 0$. Is it enough?
- Claim: the gradient may leak information on the unlearned sample.
  - Consider $\mathcal{D} = \{(e_1, 1), \ldots, (e_d, d)\}$, where e_i for $1 \leq i \leq d$ are the standard basis vectors.
  - The regressor is initialized with zero.
  - $w_i \neq 0$ if (e_i, i) is included in $\mathcal{D}$.
  - An approximate removal will still leave w_i small.
  - $\mathbb{1}(w_i \neq 0)$ indicates the existence of $(e_i, i) \in \mathcal{D}$.

Bound on the Failure of Unlearning

Theorem

Suppose that $\|\nabla \ell(w^T x_i, y_i)\|_2 \leq C$ *for any* (x_i, y_i) *and* w, ℓ'' *is* γ*-Lipschitz, and* $\|x_i\|_2 \leq 1$ *for all* x_i. *Then, we have*

$$\|\nabla(w^-; \mathcal{D}')\|_2 \leq \frac{4\gamma C^2}{\lambda^2(n-1)}.$$

- logistic regression: $C = 1$ and $\gamma = \frac{1}{4}$ when $\ell(w^T x, y) = -\log \sigma(yw^T)$
- See the paper for the data-dependent bound.
- As $n \to \infty$, $\|\nabla(w^-; \mathcal{D}')\|_2 \to 0$. Is it enough?
- Claim: the gradient may leak information on the unlearned sample.
  - Consider $\mathcal{D} = \{(e_1, 1), \ldots, (e_d, d)\}$, where e_i for $1 \leq i \leq d$ are the standard basis vectors.
  - The regressor is initialized with zero.
  - $w_i \neq 0$ if (e_i, i) is included in $\mathcal{D}$.
  - An approximate removal will still leave w_i small.
  - $\mathbb{1}(w_i \neq 0)$ indicates the existence of $(e_i, i) \in \mathcal{D}$.
- Not easy to remove; then how to hide this leaked information?

Bound on the Failure of Unlearning

Theorem

Suppose that $\|\nabla \ell(w^T x_i, y_i)\|_2 \le C$ *for any* (x_i, y_i) *and* w, ℓ'' *is* γ*-Lipschitz, and* $\|x_i\|_2 \le 1$ *for all* x_i*. Then, we have*

$$\|\nabla(w^-; \mathcal{D}')\|_2 \le \frac{4\gamma C^2}{\lambda^2 (n-1)}.$$

- logistic regression: $C = 1$ and $\gamma = \frac{1}{4}$ when $\ell(w^T x, y) = -\log \sigma(y w^T)$
- See the paper for the data-dependent bound.
- As $n \to \infty$, $\|\nabla(w^-; \mathcal{D}')\|_2 \to 0$. Is it enough?
- Claim: the gradient may leak information on the unlearned sample.
  - Consider $\mathcal{D} = \{(e_1, 1), \ldots, (e_d, d)\}$, where e_i for $1 \le i \le d$ are the standard basis vectors.
  - The regressor is initialized with zero.
  - $w_i \neq 0$ if (e_i, i) is included in $\mathcal{D}$.
  - An approximate removal will still leave w_i small.
  - $\mathbb{1}(w_i \neq 0)$ indicates the existence of $(e_i, i) \in \mathcal{D}$.
- Not easy to remove; then how to hide this leaked information?
  - Add noise!

Loss Perturbation

Perturbed empirical risk

$$L_b(w;\mathcal{D}) := \sum_{i=1}^{n} \ell(w^T x_i, y_i) + \frac{\lambda n}{2}\|w\|_2^2 + b^T w$$

- b is randomly drawn from some distribution (*i.e.*, draw b and optimize)
- $b^T w$: Add noise during training time.

Loss Perturbation

Perturbed empirical risk

$$L_b(w; \mathcal{D}) := \sum_{i=1}^{n} \ell(w^T x_i, y_i) + \frac{\lambda n}{2} \|w\|_2^2 + b^T w$$

- b is randomly drawn from some distribution (*i.e.*, draw b and optimize)
- $b^T w$: Add noise during training time.
  - This masks the information in the gradient residual $\nabla L_b(w^-; \mathcal{D}')$!.

Gradient Residual of Loss Perturbation

Perturbed gradient residual

$$\nabla L_b(w;\mathcal{D}') = \underbrace{\sum_{i=1}^{n-1} \nabla \ell(w^T x_i, y_i) + \lambda(n-1)w}_{\nabla L(w;\mathcal{D}')} + b$$

- (before) $\nabla L(w^*;\mathcal{D}') = 0$ is possible but $\nabla L(w^-;\mathcal{D}') = 0$ is not easy for fast/approximate removal.
  - $w^* = \arg\min_w L(w;\mathcal{D}')$
  - w^-: the Newton removal mechanism w.r.t. L
- (after) Have $\nabla L(w^*;\mathcal{D}') = -b$ such that $\nabla L(w^-;\mathcal{D}') \neq 0$ does not leak information.
  - $w^* = \arg\min_w L_b(w;\mathcal{D}')$
  - w^-: the Newton removal mechanism w.r.t. L

(Finally) ε-Certified Removal

Theorem

Let A be the learning algorithm that minimizes $L_b(w; \mathcal{D})$ and M be the Newton update removal mechanism. Suppose that $\|\nabla \ell(w^T x_i, y_i)\|_2 \leq C$ for any (x_i, y_i) and w, ℓ'' is γ-Lipschitz, and $\|x_i\|_2 \leq 1$ for all x_i. If $b \sim p(b) \propto e^{-\frac{\varepsilon}{\varepsilon'}\|b\|_2}$ (where $\varepsilon' := \frac{4\gamma C^2}{\lambda^2(n-1)}$) and $\|b_1 - b_2\|_2 \leq \varepsilon'$ then M is ε-CR for A.

- ε: a user-specified unlearning parameter.
- $b \sim p(b) \propto e^{-\frac{\varepsilon}{\varepsilon'}\|b\|_2}$ and $\|b_1 - b_2\|_2 \leq \varepsilon'$: *e.g.*, drawing from a distribution with a constraint (realistic?)
  - A high-probable CR definition in the paper can relax the constraint on the distribution.

(Finally) Certified Removal: A Proof Sketch

- Recall the certified removal (CR) definition:

$$\frac{\mathbb{P}\left\{M(A(\mathcal{D}), \mathcal{D}, x) \in \mathcal{T}\right\}}{\mathbb{P}\left\{A(\mathcal{D}') \in \mathcal{T}\right\}} = \frac{\mathbb{P}\{w^- \in \mathcal{T}\}}{\mathbb{P}\{w^* \in \mathcal{T}\}} \overset{?}{\leq} e^{\varepsilon}$$

  Here, the probability is taken over the randomness of the algorithm.

(Finally) Certified Removal: A Proof Sketch

- Recall the certified removal (CR) definition:

$$\frac{\mathbb{P}\left\{M(A(\mathcal{D}), \mathcal{D}, x) \in \mathcal{T}\right\}}{\mathbb{P}\left\{A(\mathcal{D}') \in \mathcal{T}\right\}} = \frac{\mathbb{P}\{w^- \in \mathcal{T}\}}{\mathbb{P}\{w^* \in \mathcal{T}\}} \overset{?}{\leq} e^{\varepsilon}$$

  Here, the probability is taken over the randomness of the algorithm.
- The randomness is from the loss perturbation by b. Suppose $b_1, b_2 \sim p(b) \propto e^{-\frac{\varepsilon}{\varepsilon'}\|b\|_2}$. Then,

$$\frac{p(b_1)}{p(b_2)} = \exp\left\{-\frac{\varepsilon}{\varepsilon'}\left(\|b_1\|_2 - \|b_2\|_2\right)\right\} = \exp\left\{\frac{\varepsilon}{\varepsilon'}\left(\|b_2\|_2 - \|b_1\|_2\right)\right\}$$
$$\leq \exp\left\{\frac{\varepsilon}{\varepsilon'}\left(\|b_2 - b_1\|_2\right)\right\} = e^{\varepsilon}$$

(Finally) Certified Removal: A Proof Sketch

- Recall the certified removal (CR) definition:

$$\frac{\mathbb{P}\left\{M(A(\mathcal{D}), \mathcal{D}, x) \in \mathcal{T}\right\}}{\mathbb{P}\left\{A(\mathcal{D}') \in \mathcal{T}\right\}} = \frac{\mathbb{P}\{w^- \in \mathcal{T}\}}{\mathbb{P}\{w^* \in \mathcal{T}\}} \overset{?}{\leq} e^{\varepsilon}$$

Here, the probability is taken over the randomness of the algorithm.

- The randomness is from the loss perturbation by b. Suppose $b_1, b_2 \sim p(b) \propto e^{-\frac{\varepsilon}{\varepsilon'}\|b\|_2}$. Then,

$$\begin{aligned}\frac{p(b_1)}{p(b_2)} &= \exp\left\{-\frac{\varepsilon}{\varepsilon'}\left(\|b_1\|_2 - \|b_2\|_2\right)\right\} = \exp\left\{\frac{\varepsilon}{\varepsilon'}\left(\|b_2\|_2 - \|b_1\|_2\right)\right\} \\ &\leq \exp\left\{\frac{\varepsilon}{\varepsilon'}\left(\|b_2 - b_1\|_2\right)\right\} = e^{\varepsilon}\end{aligned}$$

- From Theorem 2 of the paper,

$$\frac{p(b_1)}{p(b_2)} \leq e^{\varepsilon} \quad \Longrightarrow \quad \frac{\mathbb{P}\{w^- \in \mathcal{T}\}}{\mathbb{P}\{w^* \in \mathcal{T}\}} \leq e^{\varepsilon}.$$

(ε, δ)-Certified Removal

Theorem

Let A be the learning algorithm that minimizes $L_b(w; \mathcal{D})$ and M be the Newton update removal mechanism. Suppose that $\|\nabla \ell(w^T x_i, y_i)\|_2 \leq C$ for any (x_i, y_i) and w, ℓ'' is γ-Lipschitz, and $\|x_i\|_2 \leq 1$ for all x_i. If $b \sim \mathcal{N}(0, c\frac{\varepsilon'}{\varepsilon})$ with $c > 0$ (where $\varepsilon' := \frac{4\gamma C^2}{\lambda^2(n-1)}$), then M is (ε, δ)-CR for A with $\delta = 1.5e^{-c^2/2}$.

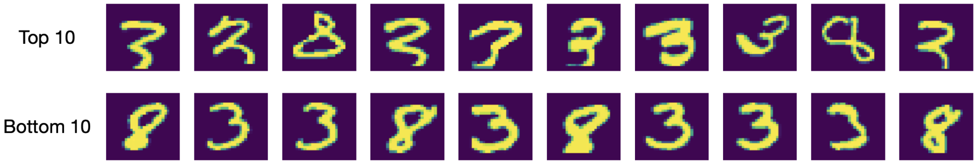
- More complex but more practical

Results: Fast

Dataset	MNIST (§4.1)	LSUN (§4.2)	SST (§4.2)	SVHN (§4.3)
Removal setting	CR Linear	Public Extractor + CR Linear	Public Extractor + CR Linear	DP Extractor + CR Linear
Removal time	0.04s	0.48s	0.07s	0.27s
Training time	15.6s	124s	61.5s	1.5h

- Removal on the last linear layer
- Faster than retraining

Results: Easy v.s. Hard

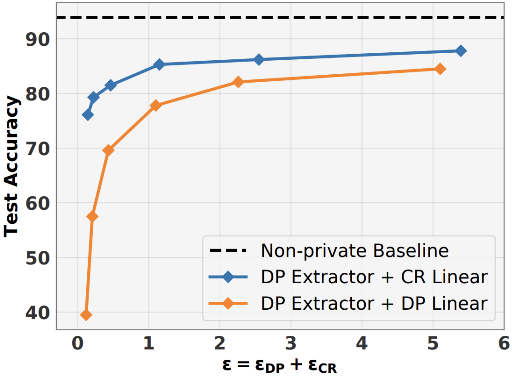


- Recall the Newton update removal:

$$w^- = M(w^*, \mathcal{D}, (x_n, y_n)) := w^* + H_{w^*}^{-1}\Delta$$

- Top 10: 10 examples with higher $\|H_{w^*}^{-1}\Delta\|_2$
- Bottom 10: 10 examples with lower $\|H_{w^*}^{-1}\Delta\|_2$
- Unusual samples are not easy to undo!
  - Removing outliers is harder.
  - The model tends to memorize unusual samples.

Results: CR v.s. DP



- Why not simply use DP?
  - CR Linear (from minimization of perturbed loss) is more accurate than DP Linear; looks better than DP. Why?
- Why not use non-DP extractor?

Conclusion

- How to evaluate the success of unlearning?
- Is the linear assumption critical?
  - We can remove data from the last linear layer of a deep network, which seems to be enough?
- We need to retrain a linear model with noise; not useful?

Reference I

Y. Cao and J. Yang. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, pages 463–480. IEEE, 2015.

P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.