Cornell University BOSTON UNIVERSITY

System-Aware Unlearning Algorithms: Use Lesser, Forget Faster

Linda Lu†, Ayush Sekhari‡, Karthik Sridharan†

†Cornell University, ‡Boston University

ICML

Primary Contributions

- We present a **new definition of unlearning** called **system-aware unlearning** that takes into account the attacker's knowledge of the system post-unlearning.
- Key Idea: By using less information, we **expose less information** to a potential attacker, leading to easier unlearning.
- To highlight the power of this viewpoint of unlearning, we show that selective sampling can be used to design a **more efficient exact unlearning algorithm for classification**.

Background

- An unlearning algorithm $A(S, U)$ removes the influence of deleted individuals $U \subseteq S$ from a model trained on dataset S, $|S| = T$.
- Motivations for unlearning: privacy, copyright, safety, etc.
- Classification Setting

Definition 1: State-of-System

Let state-of-system $\mathsf{I}_A(S, U)$ denotes what is stored in the system by an unlearning algorithm A after initially learning from sample S and performing an update for unlearning request U.

Motivation: We want to use and store as little information as possible from the sample – only what is necessary to build an accurate model.

Traditional Unlearning Definition

Definition 2: (ε, δ)-Unlearning

A is a (ε, δ)-unlearning algorithm if for all S, for all $U \subseteq S$, for all measurable sets F,

$$\Pr(A(S, U) \in F) \le e^{\varepsilon} \cdot \Pr(A(S \setminus U, \emptyset) \in F) + \delta$$

and

$$\Pr(A(S \setminus U, \emptyset) \in F) \le e^{\varepsilon} \cdot \Pr(A(S, U) \in F) + \delta.$$

- Provides privacy against a worst-case attacker who has knowledge of all the remaining individuals and the unlearned model.
- But very stringent, and has made the development of efficient unlearning very difficult
- What about other more benign adversaries?

Issue with existing definition: This worst-case attacker is extremely pessimistic. An attacker can only realistically compromise what is stored in system after unlearning.

We Propose: System-Aware Unlearning

Definition 3: System-Aware-(ε, δ)-Unlearning

A is a system-aware-(ε, δ)-unlearning algorithm if for all S, **there exists a $S' \subseteq S$**, such that for all $U \subseteq S$, for all measurable sets F,

$$\Pr(\mathsf{I}_A(S, U) \in F) \le e^{\varepsilon} \cdot \Pr(\mathsf{I}_A(S' \setminus U, \emptyset) \in F) + \delta$$

and

$$\Pr(\mathsf{I}_A(S' \setminus U, \emptyset) \in F) \le e^{\varepsilon} \cdot \Pr(\mathsf{I}_A(S, U) \in F) + \delta.$$

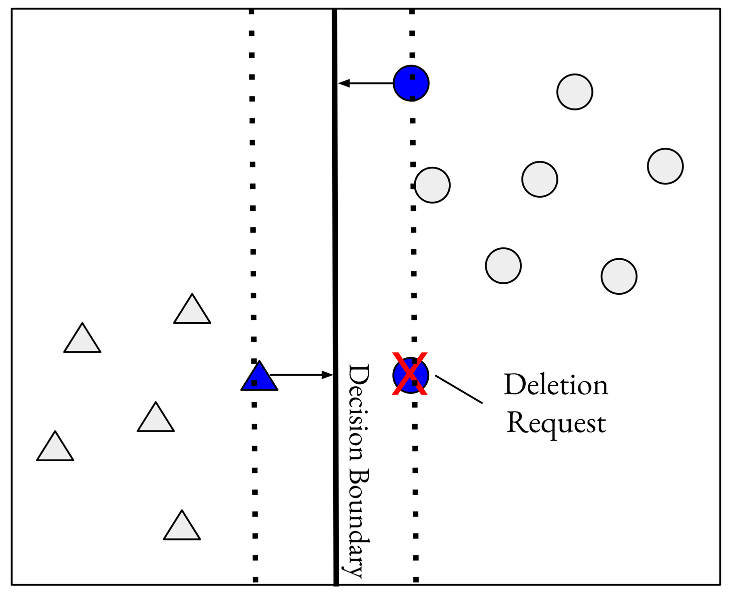
Intuition: If there exists a subset S' which is a good representative of S, then S' is the sample from the perspective of the attacker who only knows $\mathsf{I}_A(S' \setminus U, \emptyset)$.

Theorem 1: Information Theoretic Privacy Guarantee

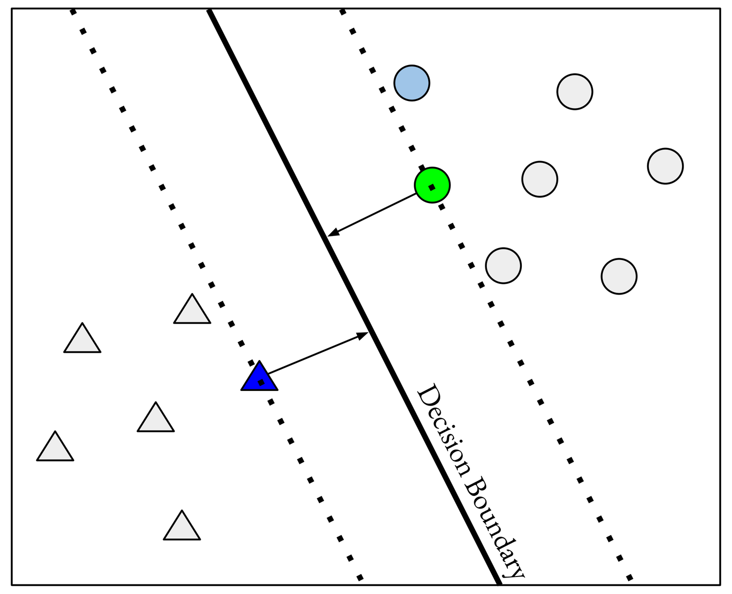
Let dataset S and set of deletions $U \subseteq S$ come from a stochastic process μ. Then, $\sup_{\mu}(\mathrm{MI}(U; S' \setminus U) - \mathrm{MI}(U; S \setminus U)) \le 0$.

- Since S' is fixed before any deletion requests U arrive, $S' \setminus U$ cannot leak any more information about U compared to $S \setminus U$.

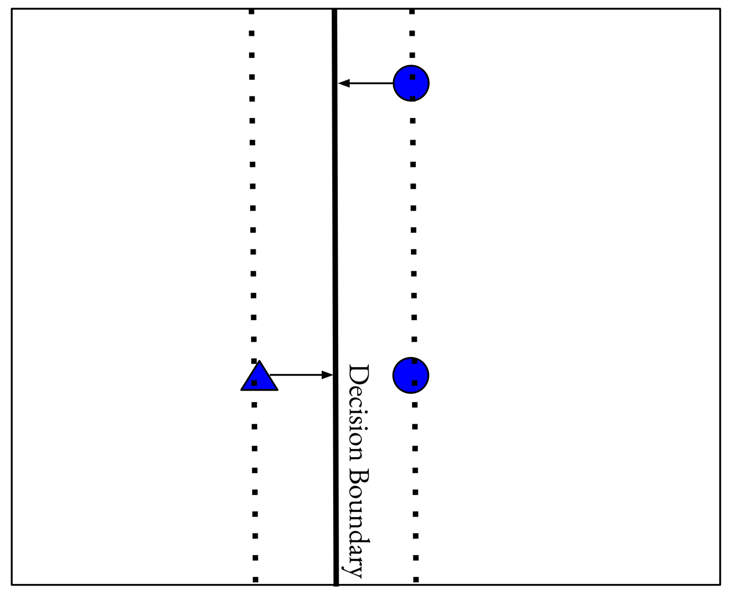
Illustrative Example: Hard Margin SVM



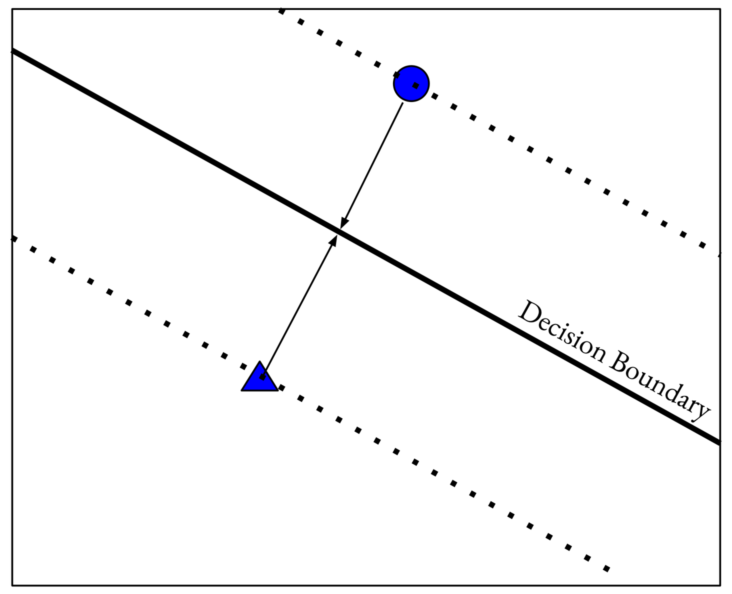
Original hard-margin SVM model



Traditional unlearning $A(S \setminus U, \emptyset)$



State-of-System $\mathsf{I}_A(S, \emptyset) =$ support vectors $= S'$



System-Aware Unlearning $A(S' \setminus U, \emptyset)$

Takeaway: Sample compression is a natural approach for system-aware unlearning.

Advantages of System-Aware Unlearning

- System-aware unlearning **generalizes** traditional unlearning.
- Points that are never used in training are deleted for free. This leads to **fast expected deletion time** and **low memory usage**.
- System-aware unlearning leads to **provably more efficient algorithms** compared to traditional unlearning.

Efficient System-Aware Unlearning via Selective Sampling

We use **selective sampling for sample compression**. Let $\mathfrak{C}(S)$ be the compression of S. Sample compression $\mathfrak{C}$ must satisfy

$$\mathfrak{C}(\mathfrak{C}(S) \setminus U) = \mathfrak{C}(S) \setminus U.$$

Algorithm 1: System-Aware Unlearning

```
Q ← SelectiveSampler(S)
Train a model on Q
if Q ∩ U ≠ ∅, for deletion requests U then
    return model trained on Q \ U
else
    return original model
```

- We use the BBQSampler (Cesa-Bianchi et al., 2009) for linear functions and the GeneralBBQSampler (Gentile et al., 2022) for general function classes for selective sampling.
- Algorithm 1 is not a valid unlearning algorithm under the traditional unlearning definition.

Theorem 2: System-Aware Unlearning

Algorithm 1 is a system-aware-$(0, 0)$-unlearning algorithm with $S' = \mathcal{Q}$ and state-of-system $\mathsf{I}_A(S, U) = (\mathcal{Q} \setminus U, A(S, U))$.

Theorem 3: Memory Complexity and Deletion Capacity (Linear)

The memory required by Algorithm 1 for linear classification is $O(dT^{\kappa} \log T)$ where $0 < \kappa < 1$ is a parameter of BBQSampler. Algorithm 1 can tolerate $K = O\left(\frac{\gamma^2 \cdot T^{\kappa}}{d \log T \cdot \log(1/\delta)}\right)$ *queried points deletions* under margin γ while maintaining excess risk guarantees.

Theorem 4: Memory Complexity and Deletion Capacity (General)

The memory required is $N_T = O\left(\frac{\mathfrak{R}(T,\delta) \cdot \mathfrak{D}(\mathcal{F}, S)}{\gamma^2}\right)$ under margin γ, where $\mathfrak{D}(\mathcal{F}, S)$ is an eluder dimension-like quantity from Gentile et al. 2022. If the regression oracle for $\mathcal{F}$ satisfies uniform stability β, then Algorithm 1 can tolerate $K = O\left(\frac{\sqrt{\mathfrak{R}(T,\delta)}}{\sqrt{N_T} \cdot \beta(N_T)}\right)$ queried point deletions while maintaining excess risk guarantees, where $\mathfrak{R}(T, \delta)$ is the convergence rate of the ERM.