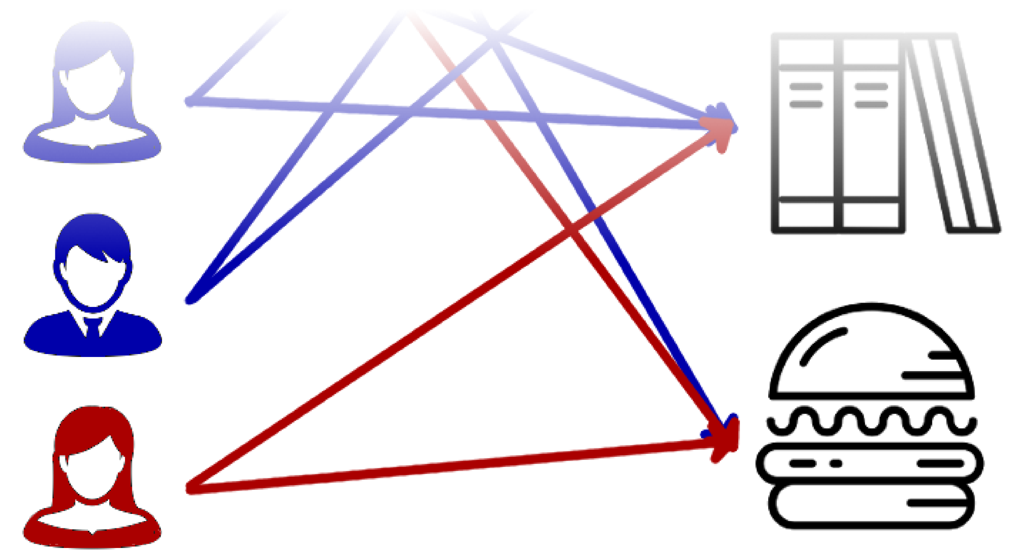
Certified Defenses for Data Poisoning Attacks

Jacob Steinhardt* Pang Wei Koh* Percy Liang

{jsteinhardt,pangwei,pliang}@cs.stanford.edu

Data Poisoning



System collects data from users, but some users (red) supply fake data to manipulate system.

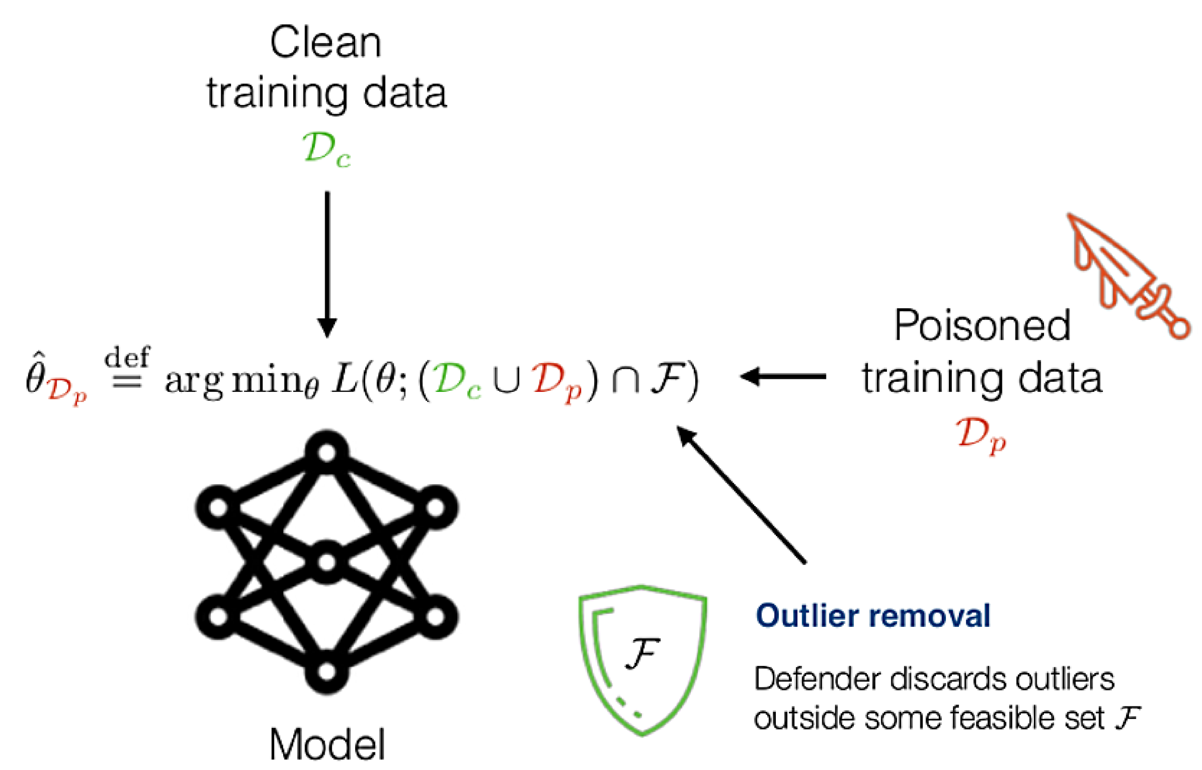
- **Goal 1:** Generate strong attacks in order to stress-test systems.
- **Goal 2:** Upper-bound the damage from the worst-case attack.

Our Contribution

- We show how to approximate the worst-case attack by a convex saddle-point problem, and design a scalable primal-dual algorithm to solve it.
- We provide a certificate of robustness bounding the worst-case attack under appropriate assumptions.

Formal Setting

Loss on single point: $\ell(\theta; x, y)$; overall loss: $L(\theta; \mathcal{D}) = \sum_{(x,y)\in\mathcal{D}} \ell(\theta; x, y)$.



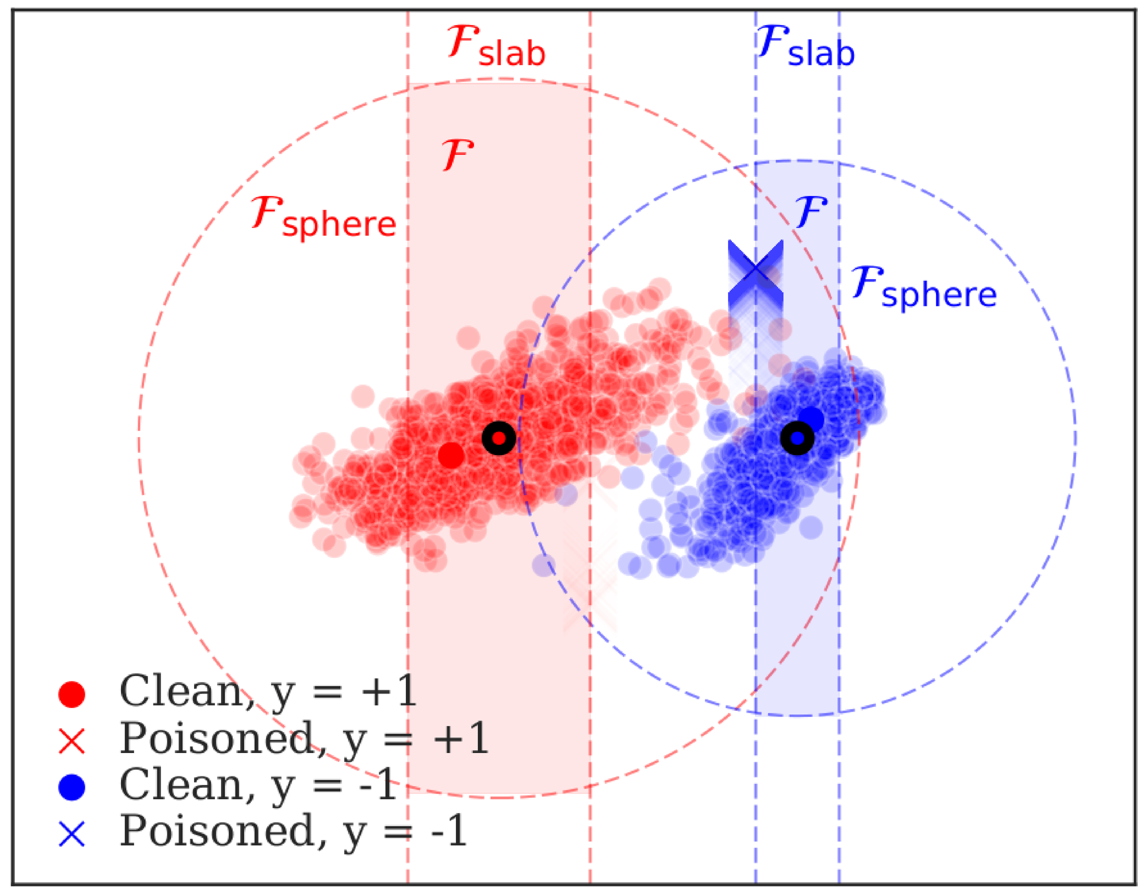
Game between adversary and learner:

- Start with **clean data** $\mathcal{D}_c = \{(x_1, y_1), \dots, (x_n, y_n)\}$
- Adversary generates ϵn points of **poisoned data** $\mathcal{D}_p$
- Learner observes clean + poisoned data: $\mathcal{D}_c \cup \mathcal{D}_p$

Learner goal: output parameters $\hat{\theta}$ with small test loss.

Adversary goal: make test loss as high as possible.

Defenses: Illustration



Our Attack Algorithm

Input: clean data $\mathcal{D}_c$ of size n, feasible set $\mathcal{F}$, poisoned fraction ϵ.

Initialize $\theta \leftarrow 0, U^* \leftarrow \infty$.

for $t = 1, \dots, \epsilon n$

  Compute attack point $(x^{(t)}, y^{(t)}) = \operatorname{argmax}_{(x,y)\in\mathcal{F}} \ell(\theta; x, y)$.

  Compute loss $\ell^{(t)} = \frac{1}{n} L(\theta; \mathcal{D}_c) + \epsilon\, \ell(\theta; x^{(t)}, y^{(t)})$.

  Compute gradient $g^{(t)} = \frac{1}{n} \nabla L(\theta; \mathcal{D}_c) + \epsilon\, \nabla \ell(\theta; x^{(t)}, y^{(t)})$.

  Update: $\theta \leftarrow \theta - \eta g^{(t)}$, $U^* \leftarrow \min(U^*, \ell^{(t)})$.

Output: attack $\mathcal{D}_p = \{(x^{(t)}, y^{(t)})\}_{t=1}^{\epsilon n}$, upper bound U^*.

Algorithm: Intuition

Perform stochastic gradient descent, but at each iteration simulate adding in the "worst fit point" $(x^{(t)}, y^{(t)})$ that can evade outlier removal.

Attack intuition: collection of all of the worst-fit points.

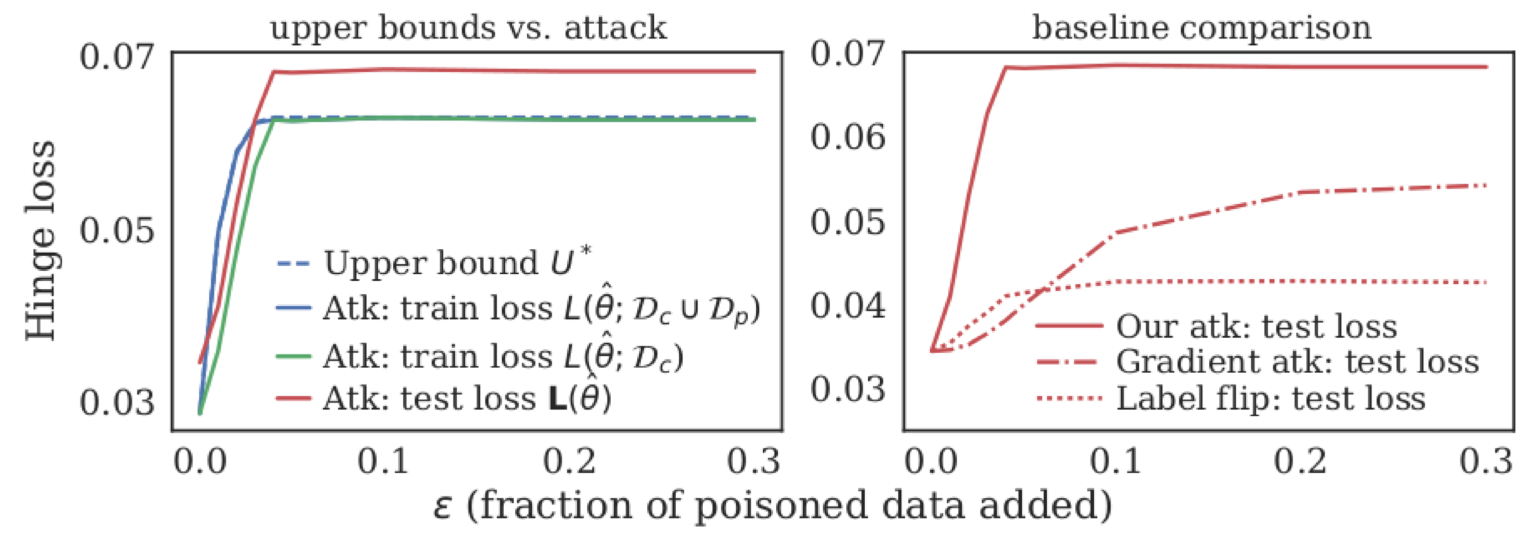
Upper bound intuition: if we can fit all possible points that evade outlier removal, no attack can perturb us by much.

Algorithm: Theory

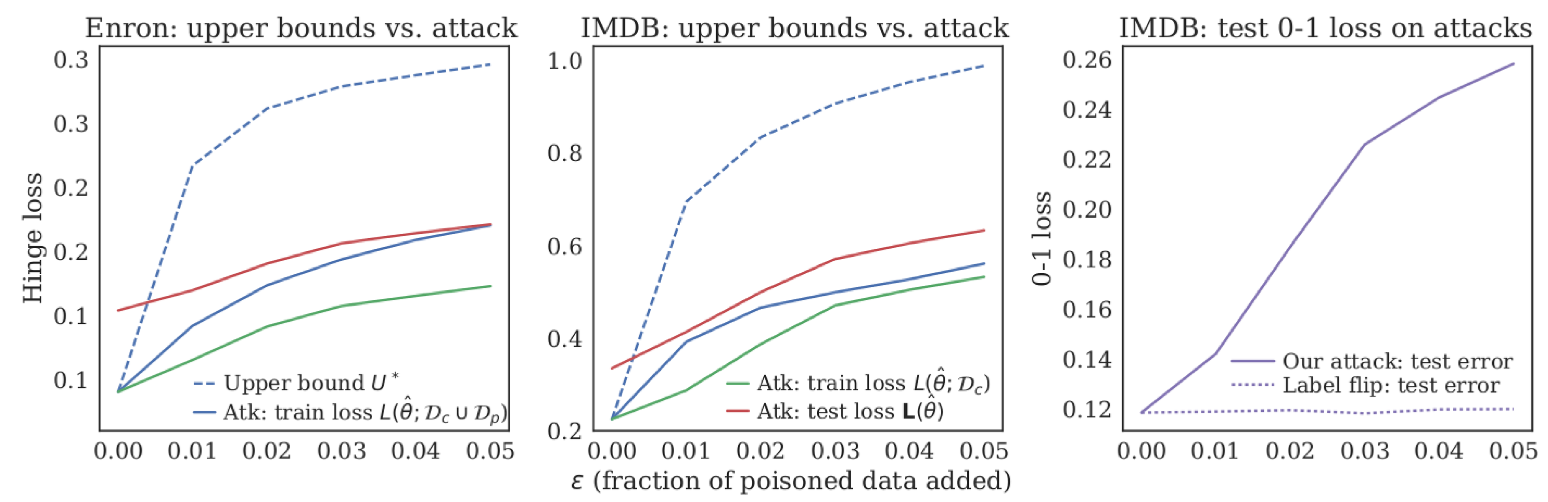
Duality. As $n \to \infty$, the *training loss* on $\mathcal{D}_c \cup \mathcal{D}_p$ converges to U^*.

Certificate. As long as $\mathcal{F}$ is not too small (e.g. outlier removal is not too aggressive) and the test loss is uniformly close to the clean train loss, U^* is an approximate upper bound on the worst-case attack.

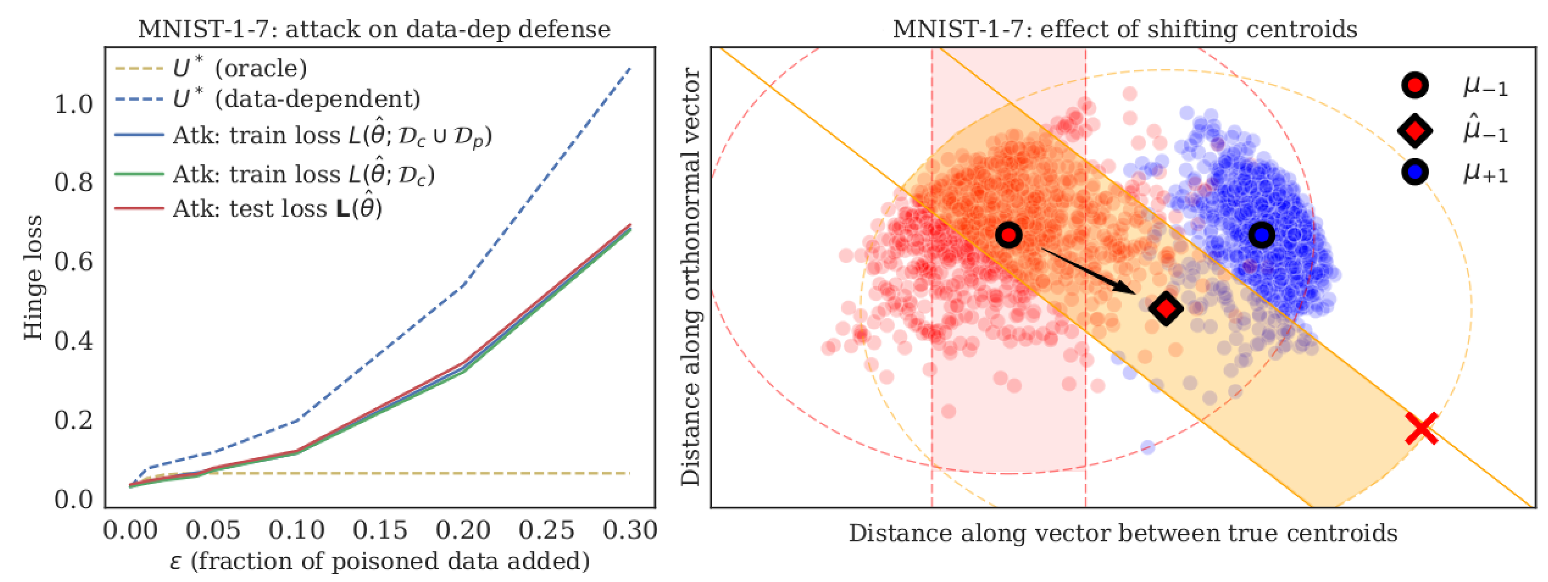
Results: Continuous Data



Results: Discrete Data (High Dimensions)



Results: Breaking the Outlier Detector



Take-Aways

- Defense is easy for medium-dimensional data that is well-separated.
- Defense is hard for high-dimensional data with many irrelevant features.
- Building an outlier detector from poisoned data creates exploitable vulnerabilities.
- Optimization is a useful framework for thinking about poisoning attacks!

Reproducible experiments on CodaLab: worksheets.codalab.org

JS was supported by a Fannie & John Hertz Foundation Fellowship and an NSF Graduate Research Fellowship.
This work was also partially supported by a Future of Life Institute grant and a grant from the Open Philanthropy Project.