Poisoning Attacks: p-Tampering and Poison Frogs

Simons Robustness Reading Group

Bolton Bailey

July 1, 2019

What is a Poisoning Attack?

"Data poisoning is an attack on machine learning models wherein the attacker adds (or modifies) examples to the training set to manipulate the behavior of the model at test time."

There are a few different models for poisoning, which depend on:

- The goal of the attacker
  - To increase the loss of the model on just a single data point (Targeted)
  - or the whole test set? (non-Targeted)
- How the adversary is allowed to change the training set
  - How many data points/what fraction of the test set do they control?
  - How can the data points be modified?
  - What information does the attacker have access to when making poisons?

Learning under p-Tampering Attacks [Mahloujifar et al. 2017]

The PAC-learning setting with poisoning:

A learning problem $P = (\mathcal{X}, \mathcal{Y}, \mathcal{D}, \mathcal{H}, \text{Loss})$ is (ϵ, δ)-PAC learnable under poisoning attacks from $\mathcal{A} = \cup_{\mathcal{D}} \mathcal{A}_D$ if for every $D \in \mathcal{D}$, $A \in \mathcal{A}_D$, and $n \in \mathbb{N}$

$$\Pr_{S \leftarrow D^n, \hat{S} \leftarrow A(S), h \leftarrow L(\hat{S})} [\text{Risk}_D(h) \leq \epsilon(n)] \geq 1 - \delta(n)$$

p-Tampering and p-Resetting attacks

The class of p-Tampering attacks consists of attacks where:

- There is at most p probability of each data point being changed.
- The attacker is online - it only sees past examples when crafting a poison example.

A p-Resetting attack is a more restricted attack, where

- When the attacker chooses to modify a data point, it simply redraws from the ground truth distribution.

The Results

Let $f : \mathrm{Supp}(S) \to [0, 1]$ be the function the attacker is trying to increase (Risk, targeted loss, probability of risk $\geq \epsilon$, etc.)

Let $\mu = \mathbb{E}[f(\mathcal{S})]$ and $\nu = \mathrm{Var}[f(\mathcal{S})]$.

Then there are p-tampering and p-resetting attacks A_{tam} and A_{res} such that

$$\mathbb{E}_{\hat{\mathcal{S}} \leftarrow A_{\mathsf{tam}}(S)}[f(\hat{\mathcal{S}})] \geq \mu + \frac{p \cdot \nu}{1 + p \cdot \mu - p} - \xi$$

$$\mathbb{E}_{\hat{\mathcal{S}} \leftarrow A_{\mathsf{res}}(S)}[f(\hat{\mathcal{S}})] \geq \mu + \frac{p \cdot \nu}{1 + p \cdot \mu} - \xi$$

These attacks run in $\mathrm{poly}(|D| \cdot n/\xi)$ time with oracle access to D.

Constructions - p-Tampering

Let $\hat{f}[d_{\leq i}]$ = Expected value of f given first i points, with the rest random from D.

If we are allowed to change data point i, we will sample potential poisons from D:

- With potential poison d_i let the rejection probability be

$$r[d_{\leq i}] = \frac{1 - \hat{f}[d_{\leq i}]}{3 - p - (1 - p)\hat{f}[d_{\leq i-1}]}$$

- Sample d_i then return it with probability $1 - r[d_{\leq i}]$. Repeat until a d_i is returned.

Can be made poly-time by cutting the sampling process short.

Proof Sketch

Write the probability of a sample arising in the poisoned set in terms of the $\hat{f}(d_{\leq i})$s

$$\frac{\Pr[\hat{D}_i = d_i | d_{\leq i-1}]}{\Pr[D = d_i]} = \frac{2 - 2 \cdot (1 - \hat{f}[d_{\leq i}])}{2 - 2 \cdot (1 - \hat{f}[d_{\leq i-1}])}$$

Evaluate the probability of a poisoned sample z

$$\Pr[\hat{S} = z] = \frac{2 - p + f(z)}{2 - p + \mu} \Pr[S = z]$$

Then evaluate the expectation of $f(\hat{S})$

$$\mathbb{E}_{\hat{\mathcal{S}} \leftarrow A_{\mathsf{tam}}(S)}[f(\hat{\mathcal{S}})] \geq \mu + \frac{p \cdot \nu}{1 + p \cdot \mu - p}$$

Moving on ...

Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks [Shafahi et al.]

- "Targeted" - Only trying to influence a single point t.
- "Clean Label" - Poisoned data will be undetectable to humans.

The technique used in this paper is "Feature-Collision"
Let $f(x)$ be the mapping of x to the penultimate layer (before softmax).

We will attempt to make $f(x) \approx f(t)$ for some x in the desired class.

The Algorithm

Choose a base instance $x_0 = b$ to start from
Do 100 steps of gradient descent on

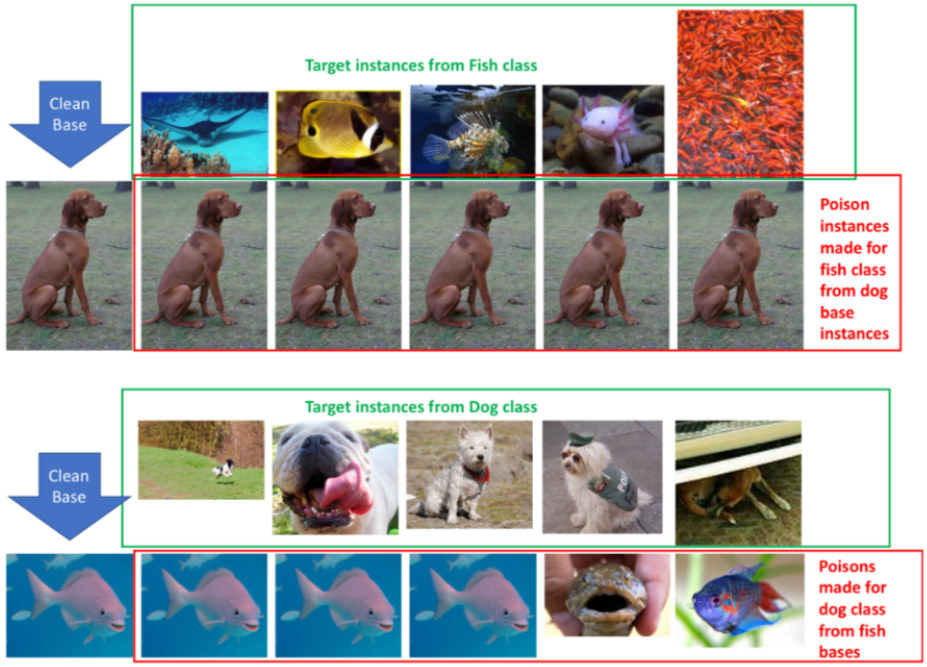
$$||f(x) - f(t)||_2^2 + \beta||x - b||_2^2$$

The first term makes the "Feature Collision" happen
The second term ensures the poison still looks like the base image.

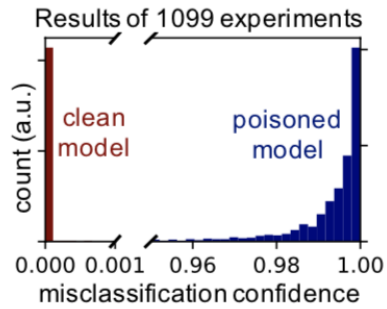
The issue: f will be different after the training with a new poison.
How do we deal with this?

Solution 1 - Transfer Learning

We declare that f will not be trained, only the final layer will be trained.



(a) Sample target and poison instances.



(b) Incorrect class's probability histogram predicted for the target image by the clean (dark red) and poisoned (dark blue) models. When trained on a poisoned dataset, the target instances not only get misclassified; they get misclassified with high confidence.

This works 100% of the time on ImageNet

Solution 2 - Watermarking (Cheating)

Instead of taking base images from the target class, we will do a mixture of 70% true target-class image, 30% target image, and use this instead as our base.

Need a lot more poison for this: 50/1000 examples vs 1/1000. Works on CIFAR