CY 7790, Lecture 11

Michael Davinroy and Gokberk Yar

October 25, 2021

1 Certified Defenses for Data Poisoning Attacks

Problem Statement. This paper presents certified bounds on attacker influence for any data poisoning attack, given a set of assumptions. In particular, these assumptions are "(1) that the dataset is large enough for statistical concentration between train and test error to hold, and (2) that outliers within the clean (non-poisoned) data do not have a strong effect on the model." Another important assumption is that the loss function for the defender's model much be *convex*.

Threat Model. The paper assumes a threat model in which the attacker fully knows the defender's algorithm (including their defenses) and the clean training data. For attacker capability, the attacker can only add points into the dataset, not modify existing points. The goal of the attacker is an untargeted attack, i.e. increase the overall test loss of the victim, in which the adversary's motivation is denial-of-service.

The full threat model is presented as a game between the attacker and the defender in which:

1. The defender draws clean training data, D_c from $p*$, or the true data-generating distribution.
2. The attacker creates the poisoned dataset, D_p.
3. The defender train the model, $\hat{\theta}$ on $D_c \cup D_p$.
4. The defender incurs loss $L(\hat{\theta}) = \boldsymbol{E}_{(x,y)\sim p*}[\ell(\hat{\theta}; x, y)]$, where ℓ is *convex*.

and the defender's goal is to minimize $L(\hat{\theta})$, which the attacker's goal is to maximize it.

Methodology. Given that a defender cannot train solely on $D_c \cup D_p$ or expect to fail, the paper examines data-sanitation defences and tried to upper bound them. The two data-sanitation defences proposed are: 1. "Fixed Data-Sanitation Defenses", and 2. "Data-Dependent Sanitation Defences". For 1, they examine two oracle defenses: F_{sphere} and F_{slab}, given in figure 1. For 2, they examine only the F_{slab} defense, given in figure 2. These defenses are visualized in the paper in figure 3.

$$\mathcal{F}_{\text{sphere}} \overset{\text{def}}{=} \{(x, y) : \|x - \mu_y\|_2 \leq r_y\}, \quad \mathcal{F}_{\text{slab}} \overset{\text{def}}{=} \{(x, y) : |\langle x - \mu_y, \mu_y - \mu_{-y}\rangle| \leq s_y\}. \tag{2}$$

Figure 1: Formulation of Fixed Data Sanitation Defences.

$$\mathcal{F}_{\text{slab}}(\mathcal{D}_{\text{p}}) \overset{\text{def}}{=} \{(x, y) : |\langle x - \hat{\mu}_y(\mathcal{D}_{\text{p}}), \hat{\mu}_y(\mathcal{D}_{\text{p}}) - \hat{\mu}_{-y}(\mathcal{D}_{\text{p}})\rangle| \leq s_y\}, \tag{7}$$

Figure 2: Formulation of Data-Dependent Data Sanitation Defences.

The main contribution is their bound on the maximum loss under these defenses, derived in the paper by the formula in figure 4. Finally, they compute this upper bound by the algorithm in figure 4, and since this is an approximation, they test it empirically (analysis of results found in the group discussion section below).

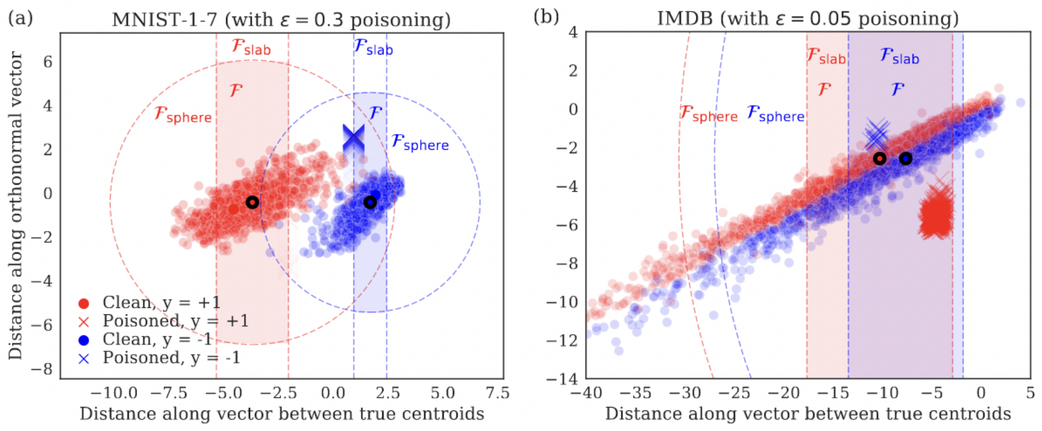


Figure 1: Different datasets possess very different levels of vulnerability to attack. Here, we visualize the effect of the sphere and slab oracle defenses, with thresholds chosen to match the 70th percentile of the clean data. We mark with an X our attacks for the respective values of ϵ. **(a)** For the MNIST-1-7 dataset, the classes are well-separated and no attack can get past the defense. Note that our attack chooses to put all of its weight on the negative class here, although this need not be true in general. **(b)** For the IMDB dataset, the class centroids are not well-separated and it is easy to attack the classifier. See Section 4 for more details about the experiments.

Figure 3: Visualization of Fixed Data Sanitation Defences.

$$\begin{aligned}
\max_{\mathcal{D}_p} \mathbf{L}(\hat{\theta}) &\overset{\text{(i)}}{\approx} \max_{\mathcal{D}_p} \frac{1}{n} L(\hat{\theta}; \mathcal{D}_c) \overset{\text{(ii)}}{\leq} \max_{\mathcal{D}_p} \frac{1}{n} L(\hat{\theta}; \mathcal{D}_c \cup (\mathcal{D}_p \cap \mathcal{F})) \\
&\overset{\text{(iii)}}{\approx} \max_{\mathcal{D}_p} \frac{1}{n} L(\tilde{\theta}; \mathcal{D}_c \cup (\mathcal{D}_p \cap \mathcal{F})) \\
&= \max_{\mathcal{D}_p \subseteq \mathcal{F}} \min_{\theta \in \Theta} \frac{1}{n} L(\theta; \mathcal{D}_c \cup \mathcal{D}_p) \overset{\text{def}}{=} \mathbf{M}. \qquad (4)
\end{aligned}$$

Figure 4: Calculating the Maximum Bound.

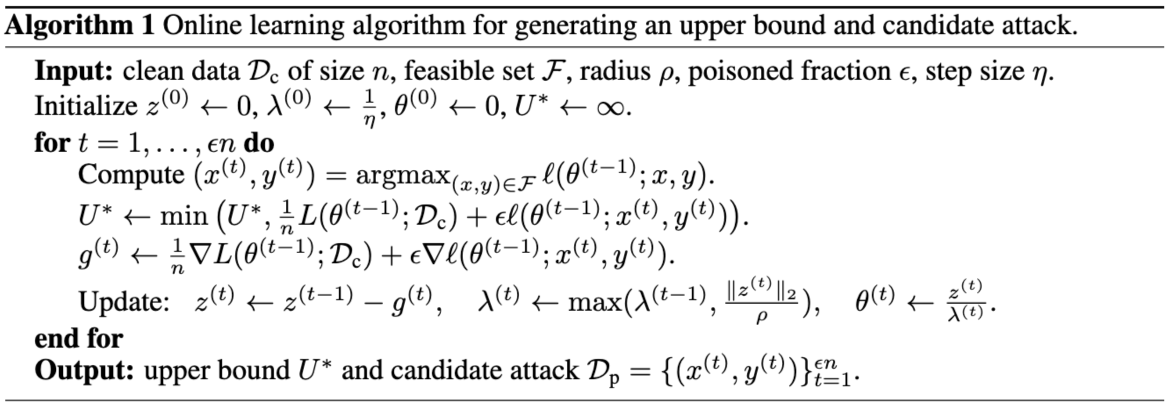
Discussion. In class, we talked mostly about limitations of this paper because the certifications are closely related to the assumptions made. For instance, it was mentioned that for the fixed data sanitation defences, the true centroid of the clean data is known, which is an extremely strong assumption that most likely cannot be known. However, the paper also considers the more realistic setting of data-dependent sanitation defences, but this assumption in the first setting implies that this data-dependent setting is much more vulnerable to attack.

We also discussed how assumption two is pretty hand-wavy, in that it was chosen to make the distribution easier to analyze, but often does not hold on real-world data. Additionally, as stated before, we discussed how the restriction that the loss function be convex is a major limiting factor in that real-world systems are often non-convex.

Logistically, the paper is a little confusing in two parts: 1. The Red X's are super faint in figure 1a, which makes reading it confusing, and 2. The paper does not fully specify r and s for sphere and slab thresh-holds, but it does give an example on page 3 where 30% of clean samples are removed.

Toward evaluation, testing only on MNIST does not give any strong evidence of their approach empirically. Second, we discussed how the bound is not tight for the text data (figure 6) and an improvement might be to add an experiment where they remove a great portion the clean dataset, so that models could differ as an explanation of the upper bound not matching the figure.

Overall, we found this paper was a good step in looking for guarantees in defensibly, especially because it was



Algorithm 1 Online learning algorithm for generating an upper bound and candidate attack.

```
**Input:** clean data $\mathcal{D}_c$ of size $n$, feasible set $\mathcal{F}$, radius $\rho$, poisoned fraction $\epsilon$, step size $\eta$.
Initialize $z^{(0)} \leftarrow 0$, $\lambda^{(0)} \leftarrow \frac{1}{\eta}$, $\theta^{(0)} \leftarrow 0$, $U^* \leftarrow \infty$.
**for** $t = 1, \dots, \epsilon n$ **do**
    Compute $(x^{(t)}, y^{(t)}) = \text{argmax}_{(x,y)\in\mathcal{F}}\, \ell(\theta^{(t-1)}; x, y)$.
    $U^* \leftarrow \min\big(U^*, \frac{1}{n}L(\theta^{(t-1)}; \mathcal{D}_c) + \epsilon\ell(\theta^{(t-1)}; x^{(t)}, y^{(t)})\big)$.
    $g^{(t)} \leftarrow \frac{1}{n}\nabla L(\theta^{(t-1)}; \mathcal{D}_c) + \epsilon\nabla\ell(\theta^{(t-1)}; x^{(t)}, y^{(t)})$.
    Update: $z^{(t)} \leftarrow z^{(t-1)} - g^{(t)}$,  $\lambda^{(t)} \leftarrow \max(\lambda^{(t-1)}, \frac{\|z^{(t)}\|_2}{\rho})$,  $\theta^{(t)} \leftarrow \frac{z^{(t)}}{\lambda^{(t)}}$.
**end for**
**Output:** upper bound $U^*$ and candidate attack $\mathcal{D}_p = \{(x^{(t)}, y^{(t)})\}_{t=1}^{\epsilon n}$.
```

Figure 5: Algorithm for Computing the Maximum Bound.

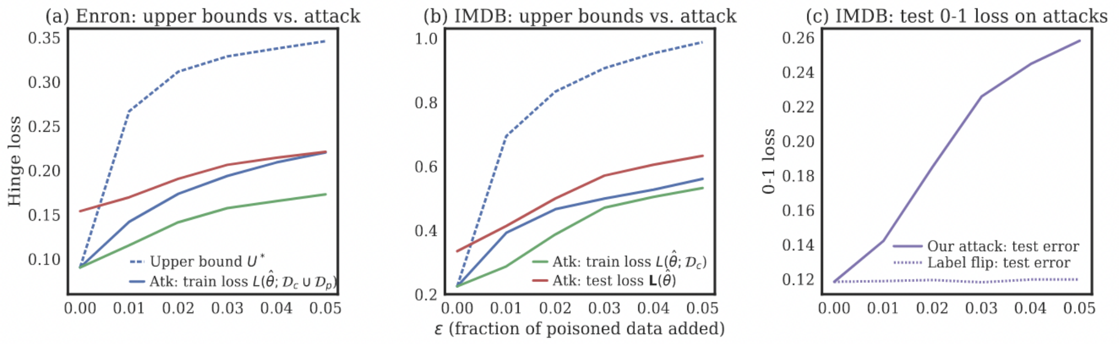


Figure 3: The **(a)** Enron and **(b)** IMDB text datasets are significantly easier to attack under the oracle sphere and slab defense than the image datasets from Figure 2. **(c)** In particular, our attack achieves a large increase in test loss (solid red) and test error (solid purple) with small ϵ for IMDB. The label flip baseline was unsuccessful as before, and the gradient baseline does not apply to discrete data. In (a) and (b), note the large gap between upper and lower bounds, resulting from the upper bound relaxation and the IQP/randomized rounding approximations.

Figure 6: Paper experiments for text bound.

one of the first to look for them, but the strong assumptions and somewhat weak empirical evidence make this paper very limiting for real world problems and models. A key observation is that creating stronger guarantees for more common models is at lest in part difficult due to non-convex functions breaking most guarantees, and finding optima for non-convex functions is a computationally difficult problem.

2 Wang et al. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks

Problem Statement. This work focuses on detection, re-construction and mitigation of *backdoor attacks* on DNN image classifiers. *Backdoor attacks* are sub domain of poisoning attacks where adversaries' goal to add a *trigger* to model in training time. During the test time, models behave as expected without the *trigger* but when *trigger* is present, models classify target label chosen by the adversary in targeted case and mis-classify in un-targeted case. Detection of *backdoor attacks* means binary decision whether a given model has a *backdoor* or not. Re-construction of *backdoor attacks* is reproduction and visualization of the *backdoors* added by the adversary. Mitigation of *backdoor attacks* means by elimination of *backdoor* inside the model with modification on the model. *Backdoor attack* is visualized in Figure 7.

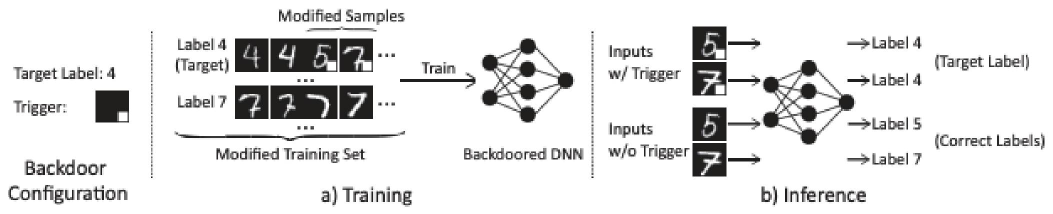


Fig. 1. An illustration of backdoor attack. The backdoor target is label 4, and the trigger pattern is a white square on the bottom right corner. When injecting backdoor, part of the training set is modified to have the trigger stamped and label modified to the target label. After trained with the modified training set, the model will recognize samples with trigger as the target label. Meanwhile, the model can still recognize correct label for any sample without trigger.

Figure 7: Backdoor Attack Visualized.

Adversarial Capabilities and Goals. This work assumes adversary has white box access to training data (able to modify), model architecture, parameters and aims to misclassify the image in the presence of a backdoor *trigger*.

Defenders Capabilities. This work assumes defender has also access to model architecture and parameters. In addition, defender has a clean dataset from the same distribution (for validation purposes), and computational resources (GPUs).

Defenders Goals. 1) Detect the presence of the *backdoors* in model. 2) Re-contruction of backdoors. 3) Mitigation of backdoors.

Methodology. Key intuition introduce in the paper is perturbations should be unusually small for the target class. This intuition is visualized in Figure 8.

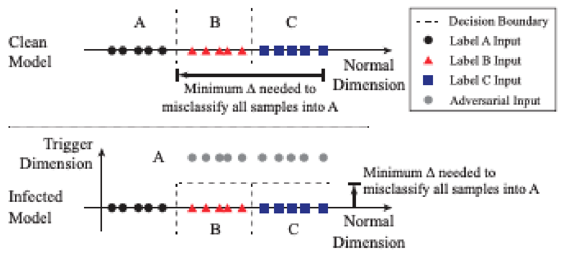


Fig. 2. A simplified illustration of our key intuition in detecting backdoor. Top figure shows a clean model, where more modification is needed to move samples of B and C across decision boundaries to be misclassified into label A. Bottom figure shows the infected model, where the backdoor changes decision boundaries and creates backdoor areas close to B and C. These backdoor areas reduce the amount of modification needed to misclassify samples of B and C into the target label A.

Figure 8: Intuition visualized.

Under this intuition, they observed 1) that perturbation needed to classify all images to chosen L_t is bounder with the size of the trigger, shown in 1. δ is the perturbation, T is the trigger, and subscript denotes label. 2) To not get detected, size of the perturbation should be strictly smaller than perturbation needed for any image except the target to transform into a label except that the target, formally shown in 2. Under these observations, they suggested that backdoors can be find by abnormal low perturbation values for a label.

$$\delta_{\forall \to t} \leq \| T_t \| \tag{1}$$

$$\delta_{\forall \to t} \leq \| T_t \| \ll min_{i, i \neq t} \| \delta_{\forall \to i} \| \tag{2}$$

Backdoor Detection Algorithm. Using the optimization in Figures 9 and 10, find the minimal trigger size for each class. All dataset should be classified as this target class. Using an outlier detection method, find the candidate class for the backdoor. Final mask suggestion in the optimization is the candidate reverse engineered backdoor.

[illegible]

$$\begin{aligned} &A(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{\Delta}) = \boldsymbol{x}' \\ &\qquad \boldsymbol{x}'_{i,j,c} = (1 - \boldsymbol{m}_{i,j}) \cdot \boldsymbol{x}_{i,j,c} + \boldsymbol{m}_{i,j} \cdot \boldsymbol{\Delta}_{i,j,c} \end{aligned} \tag{2}$$

Figure 9: Trigger Definition.

$$\begin{aligned} \min_{\boldsymbol{m}, \boldsymbol{\Delta}} \quad & \ell(y_t, f(A(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{\Delta}))) + \lambda \cdot |\boldsymbol{m}| \\ \text{for} \quad & \boldsymbol{x} \in \boldsymbol{X} \end{aligned} \tag{3}$$

Figure 10: Trigger optimization problem

Outlier Detection. Calculate Mean Absolute Deviation (MAD), normalize each with MAD to get anomaly index. If anomaly index is larger than 2, it is an outlier 95% probability.

Mitigation Methods 1) Adversarial Input Filter: In this method, the defender first finds out responsible neurons for the backdoor. This could be achieved feeding the model with the reversed engineered backdoor and averaging out neurons at the penultimate layer of the architecture and comparing it with when trigger not added to image. After identification, if an image gets a very high signal in those neurons, i.e., above a threshold, discard the input. 2) Patching DNN via Neuron Pruning: This method is similar to method one. Find the neurons and set them 0 one by one until the effects of the trigger is cleared. 3) Patching DNN via Unlearning: In this method train the model for several iterations using a clean dataset; this will unlearn the backdoors.

Class Discussion.

Reject model: Can the model be rejected rather than repaired?

Other anomaly detection schemes: Can other anomaly detection schemes be used?

Semantic backdoors: Semantic backdoors do not have a well described pattern.

False positive rates: False positives result in a high cost for the production, cost aware algorithm or smaller false positive rates are needed.

Manual inspection: It is not always feasible for checking if the mitigation worked or not.