Security and Privacy in Machine Learning

Homework 5

Javad Hezareh
98101074

Spring 2023

Contents

1 Poisoning

(a) Our goal in this poisoning attack is to maximize $\mathcal{L}_{attack}$ on test data $\mathcal{D}_{test}$ using function f that comes from training on $\mathcal{D}_{train} \cup \mathcal{D}_{poison}$ using $\mathcal{L}_{train}$ as the loss function. Let f be parameterized with θ, then we can formulate our attack as the following bilevel optimization problem:

$$\mathcal{D}^*_{poison} = \underset{\mathcal{D}_{poison}}{\arg\max}\ \mathcal{L}_{attack}\left(f(\hat{\theta}), \mathcal{D}_{test}\right)$$

$$s.t. \quad \hat{\theta} = \underset{\theta}{\arg\min}\ \mathcal{L}_{train}\left(f(\theta), \mathcal{D}_{train} \cup \mathcal{D}_{poison}\right)$$

(b) The fault in this reasoning comes from the partitioning part. In valid certified defense the key idea is the disjoint partitioning of the training set to k classes so that changing one input only affects one classifier decision. But in this method we have

$$P_i^S := \{s \in S \mid h(s) \overset{i}{\equiv} 0\}.$$

Using this scheme for partitioning, if one sample s be in partition i, then it will also be in partitions j where $i \overset{j}{\equiv} 0$. Therefore we won't have a disjoint partitioning and hence changing one input might affect more than one classifier decision.

---

2 Black-Box Attack

(a) This attack changes the input x till the most probable class isn't y. Therefore this algorithm performs an untargeted attack. In order to have a targeted attack with target class t we need to change two lines of this algorithm, lines 4 and 8:

$$\begin{cases} 4: \quad \text{while } \boldsymbol{P}_y = \max_{y'} \boldsymbol{P}_{y'} \text{ do} \quad \Longrightarrow \quad \text{while } \boldsymbol{P}_t \;!= \max_{y'} \boldsymbol{P}_{y'} \text{ do} \\ 8: \quad \text{if } \boldsymbol{P}'_y < \boldsymbol{P}_y \text{ then} \quad \Longrightarrow \quad \text{if } \boldsymbol{P}'_t > \boldsymbol{P}_t \text{ then} \end{cases}$$

(b) The idea of this attack is to search in random directions if the prediction probability decrease/increase in untargeted/targeted attack. To check each direction we query our model two times. If we used dependent basics, then we could see some points more than one time and therefore have some useless queries. By picking basics with replacement also this situation could happen. Also, choosing basics with replacement will increase the chance of choosing one basic more than one time and this will increase the norm of total perturbation δ. Therefore we need to choose independent basis such as orthogonal basis and picking them without replacement.

(c) It doesn't use efficient query method. If we consider small $|\epsilon|$ then using Taylor expansion of the probability function we have

$$\boldsymbol{P}(y|\boldsymbol{x} + \epsilon\boldsymbol{q}) \approx \boldsymbol{P}(y|\boldsymbol{x}) + (\nabla_{\boldsymbol{x}}\boldsymbol{P})^T(\epsilon\boldsymbol{q}),$$

therefore if $+\epsilon$ increases $\boldsymbol{P}_y$, then $-\epsilon$ will decrease this value and vice versa. Therefore we don't need to query model for both $+\epsilon$ and $-\epsilon$ and only using one of them, for example $+\epsilon$ will be enough. So we need to replace lines 6 to 11 of the original algorithm with following lines:

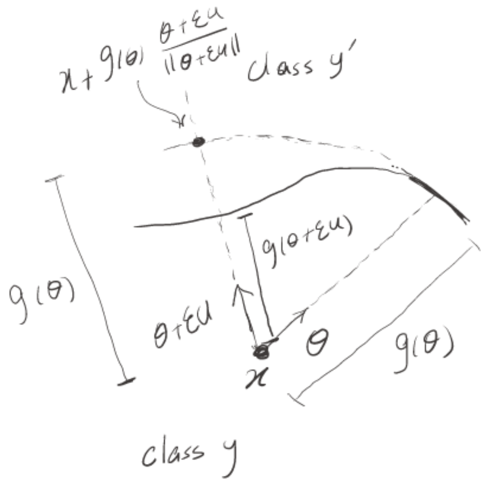
```
1: $\boldsymbol{P'} = p_h(y|\boldsymbol{x} + \boldsymbol{\delta} + \epsilon\boldsymbol{q})$
2: if $\boldsymbol{P}'_y < \boldsymbol{P}_y$ then
3:     $\boldsymbol{\delta} = \boldsymbol{\delta} + \epsilon\boldsymbol{q}$
4:     $\boldsymbol{P} = \boldsymbol{P'}$
5: else if $\boldsymbol{P}'_y > \boldsymbol{P}_y$ then
6:     $\boldsymbol{\delta} = \boldsymbol{\delta} - \epsilon\boldsymbol{q}$
7:     $\boldsymbol{P} = 2\boldsymbol{P} - \boldsymbol{P'}$
8: end if
```

(d) This inequality presents a trade-off between number of iteration T, and the norm of each perturbation $\|\epsilon\|$. If we have low query budget, then we need to lower T and hence to get enough total perturbation norm we need to increase the norm of each steps $\|\epsilon\|$. If we want our total perturbation be small, then we can increase number of iteration T by decreasing the norm of each step $\|\epsilon\|$.

---

3 Gradient's Sign

(a) **No**; Using this condition to find the gradient's sign is not correct.



As we can see in the figure if the condition holds, then the distance of x in the new direction $\theta + \epsilon u$ to the decision boundary is less than the previous distance $g(\theta)$, therefore the gradient's sign will be -1 instead of +1.

(b) Using the sign of the gradient instead of computing it completely can not have that information we can gain by using the later one. But the information we obtain by this method is sufficient to solve our problem. Because in each step we want to decrease g till we reach our local minima and using the gradient's sign will ensure this happens. In other words we can show that using gradient's sign instead of its total value, will eventually converge to the local minima of g.

---

4 Priors and Find Gradient with Optimization

(a) Two types of prior are introduced in this paper:

- **Time-Dependent Priors**: This type of prior is about the correlation between the estimated gradients in different steps of our trajectory. For example using the NES algorithm, there is a correlation between successive gradient estimation.

- **Data-Dependent Priors**: This type is about the structure of our data and its influence on the gradient. For example in images we have spatial locality and pixels that are close to each other have almost similar values. Thus the gradient w.r.t the input will also have similar value for pixels that are close to each other.

(b) The hole trick is to find a way to calculate $\ell(g)$, then we can employ

$$\frac{\ell(g+\delta u)-\ell(g-\delta u)}{\delta}u,$$

where u is a random vector from $\mathcal{N}(0, \frac{1}{d}I)$, to find the derivative Δ_t. From the definition we have

$$\ell(g) = -\langle \nabla L(x,y), \frac{g}{\|g\|} \rangle,$$

therefore in fact $\ell(g)$ is the directional derivative of $L(x,y)$ in the direction of g. We can estimate this value using finite difference,

$$\ell(g) \approx -\frac{L(x+\epsilon g, y) - L(x,y)}{\epsilon}.$$

By employing this on our estimation formula we have:

$$\begin{aligned}
\Delta_t &= \frac{\ell(g+\delta u)-\ell(g-\delta u)}{\delta}u \\
&= \frac{1}{\delta}\left[-\frac{L(x+\epsilon(g+\delta u),y)-L(x,y)}{\epsilon} + \frac{L(x+\epsilon(g-\delta u),y)-L(x,y)}{\epsilon}\right]u \\
&= \frac{L(x+\epsilon(g-\delta u),y)-L(x+\epsilon(g+\delta u),y)}{\epsilon\delta}u
\end{aligned}$$