Lecture 15: Additional topics in robust deep learning

November 19, 2019

In this lecture, we cover a number of topics that we won't have time to cover in more depth.

1 Adversarial examples for other perturbations

While we've mostly focused on ℓ_∞, ℓ_2, and, to a lesser extent, ℓ_1 perturbations, the problem is still very interesting—and arguably, more interesting—when we consider other norms. Recall that ultimately the set of perturbations we care about is the set of "semantically meaningless" perturbations. As we've previously discussed, it is clear that ℓ_p perturbations do not fully capture this notion. There is a lot of room for interesting work on this general question: while researchers have demonstrated attacks for a number of settings, defenses for many non ℓ_p settings are much less well-understood.

ℓ_0 adversarial examples A type of perturbation which is (in some sense) technically an ℓ_p perturbation but which has drastically different qualitative properties are ℓ_0 perturbations. Formally, the ℓ_0 "norm" of a point $x \in \mathbb{R}^d$, denoted $\|x\|_0$, is the number of nonzero entries of x. Thus we can also define the ℓ_0 perturbation set:

$$\mathcal{P}_{0,\varepsilon}(x) = \left\{ y \in \mathbb{R}^d_{\geq 0} : \|x - y\| \leq \varepsilon \right\} .$$

Note that we may assume WLOG that ε is integer-valued.

The crucial difference between this and ℓ_p for $p \geq 1$ is that of course ℓ_0 is not in fact a norm—it is not convex. Hence finding adversarial examples requires a very different technique than just PGD, as the projection is non-convex and unlikely to be meaningful. Instead, algorithms such as JSMA [1] use heuristics to select which pixels appear to be the most important to changing the decision boundary. This heuristics appear to work quite well.

On the defense side, there has been recent work on certifiable defenses against ℓ_0 attacks, however, the state of the art is quite weak. In particular, to the best of the author's knowledge, no certifiable defense is able to do much better than ε roughly 5 or 6.

Wasserstein adversarial examples A type of perturbation which arguably captures a larger set of semantically meaningless perturbations than ℓ_p perturbations are perturbations which have bounded Wasserstein distance (also known as *earth mover distance*) to the true image.

Formally, let $x, y \in \mathbb{R}^d$ be so that $x_i, y_i \in [0, 1]$ for all i. Let $\mathbb{R}^{d \times d}_{\geq 0}$ denote the set of non-negative matrices. Let $C \in \mathbb{R}^{d \times d}$ be a fixed cost matrix. Intuitively, C_{ij} is how much it costs to transport one unit of mass from index i to index j. Then, we define the Wasserstein distance between x, y to be

$$d_{\mathcal{W}}(x, y) = \min_{\Pi \in \mathbb{R}^{d \times d}_{\geq 0}} \langle \Pi, C \rangle \tag{1}$$

$$\text{s.t.} \qquad \Pi \mathbf{1} = x, \ \mathbf{1}^\top \Pi = y \ . \tag{2}$$

It can be verified (see HW) that this is indeed a norm over nonnegative vectors. Given this, we can define the Wasserstein perturbation set as:

$$\mathcal{P}_{\mathcal{W},\varepsilon}(x) = \left\{ y \in \mathbb{R}^d_{\geq 0} : d_{\mathcal{W}}(x, y) \leq \varepsilon \right\} \ .$$

The problem for scaling this up to neural networks is that solving this efficiently is nontrivial. While the problem of computing Wasserstein distance can be done in polynomial time (as it is an LP), this is too slow to scale up to large tasks. However, there are efficient methods, such as *Sinkhorn iteration*, which allow us to compute these projections more efficiently. This allows these algorithms to scale to neural networks on CIFAR-10. See [2] for more details. However, in that paper they only give empirical defenses. An excellent open question is to give meaningful certified defenses for Wasserstein distance perturbations.

Defending physical world attacks As mentioned in previous classes, there have been many attacks that work on real world image recognition systems. Despite this, work on corresponding defenses has been quite slow to catch up. This is because these attacks are quite slow to run: often taking on the order of hours to days. In particular, this precludes the possibility of using traditional adversarial training as a defense, as it is simply too slow.

2 Backdoor attacks

Backdoor attacks are a different type of attack from "traditional" adversarial examples in two ways. First, they are are *mixed train-test* attack, meaning that we assume that the adversary can manipulate both the train and test set. Moreover, the goal of the attack is not to cause degradation to test accuracy. Rather, the goal is to implant a "backdoor" or "watermark" into the network.

Let's make this formal. As usual, we will present this in the context of image classification but it is not hard to imagine similar attacks in the setting of e.g. speech recognition. The adversary has some (unknown) fixed perturbation $p \in \mathbb{R}^d$, which we think of as very sparse. Given two vectors x, p, where p is very sparse, we let $x \odot p$ denote the vector which is equal to x outside the support of p, and equal to p inside.

We assume that we are given an ε-corrupted training set $(X_1, y_1), \ldots, (X_n, y_n)$ drawn from the true data distribution $\mathcal{D}$, where the adversary creating the corrupted training set knows p. The goal of the adversary is the to simultaneously achieve the following two objectives. Suppose the estimator takes in $S = \{(X_1, y_1), \ldots, (X_n, y_n)\}$ and outputs $\hat{f}$. Then we say that the adversary *succeeds* if:

- The clean accuracy of the model is good, that is, $\mathbb{E}_{(X,y)\sim\mathcal{D}}\left[\ell(\hat{f}, X, y)\right]$ is low.
- The *backdoored* accuracy of the model is bad, that is, $\mathbb{E}_{(X,y)\sim\mathcal{D}}\left[\ell(\hat{f}, X \odot p, y)\right]$ is high.

That is, the model looks completely normal on clean data. However, if the adversary is allowed to add their pre-determined "backdoor" or "watermark", then the classifier is fooled.

We note that there are many variants of this problem specification. For instance, there are targeted attack models, where instead of simply trying to mess up the classification accuracy, the attacker additionally has classes $y_{\text{target}}, y_{\text{true}} \in \mathcal{Y}$, and the goal of the adversary is so that if (X, y) is drawn from $\mathcal{D}$ conditioned on $y = y_{\text{true}}$, then $\hat{f}(X) = y_{\text{target}}$ with high probability. However, as far as the writer is aware, the qualitative behavior of both the attacks and defenses do not change between these different models.

It turns out that if the classifier is a deep neural network trained via standard SGD, then such attacks are trivial. One can verify experimentally that the following attack works: given a clean training set, choose a fixed class $y^* \in \mathcal{Y}$, randomly choose an ε-fraction of the data points $T \subset S$ conditioned on the label not being y^*, and let the corrupted training set be the same outside of T, but replace every $(X, y) \in T$ with $(X \odot p, y^*)$.

Heuristically, this is not surprising. We expect that deep networks can "memorize" training data. That is, we expect that the network will achieve perfect training accuracy, and indeed, it does, even on the corrupted data. To do so, the network must realize that the presence of p overrides any other signals for classification. As a result, the learned representation will, when given $x \odot p$, heavily amplify the signal of p. It must do so, as otherwise $x \odot p$ would have been classified according to the true representation of x, which means that from the perspective of the training data, the network would be misclassifying it. But then when we are given a fresh sample containing the backdoor, the learned representation boosts the signal of the backdoor, resulting in misclassification.

While there has been quite a bit of work recently on the problem of defending backdoor attacks, it is not clear if there is a well-vetted defense yet. However, we note a cute connection to robust statistics (which should not be claimed to be a full defense). Suppose S is our ε-corrupted dataset, and $\hat{f}$ is a learned representation, learned via vanilla SGD (non-robustly). Then let $S' = \hat{f}(S)$. If the intuition about how the work proceeds is correct, then one should notice that S' has two distinct sub-populations. Namely, the learned representations for the points in T will be activating much different parts of the representation space than those outside of it. In particular, they are sufficiently separated that empirically (at least with this naive attack), the largest eigenvalue of the empirical covariance of $\hat{f}(S)$ is substantially larger than the largest eigenvalue of $\hat{f}(S \setminus T)$. Thus, we can use the technology of spectral signatures as developed previously in this course to help with this attack [3]. There are

(somewhat weaker) attacks that fool this defense [4, 5], but it is nice that there is such a clean connection.

3 Data poisoning attacks on deep networks

Finally, we turn our attention back again towards train-time corruptions, which in the deep learning literature is often referred to as *data poisoning* attacks. Here, we consider attacks where the adversary is allowed to corrupt an ε-fraction of the training data, and the goal is to decrease test accuracy; that is, to decrease $\mathbb{E}_{(X,y)\sim\mathcal{D}}\left[\ell(\hat{f}, X, y)\right]$, where $\hat{f}$ is the learned model. While many data poisoning attacks have been demonstrated against convex models such as SVM, logistic regression, linear regression, etc., to date there have been no "non-trivial" attacks against deep neural networks, trained with vanilla non-robust SGD.

The baseline attack for data poisoning a neural network is the *random label-flipping* attack. As the name suggests, the attack works by taking an ε-fraction of the data points, and flipping their labels randomly. Empirically, this appears to cause the test accuracy for 0/1 loss to decrease by ε, for ε relatively small. In contrast, note that data poisoning attacks for SVM, logistic regression, ridge regression, etc, are often able to (at least when there are no defenses), degrade the test accuracy by much more, often to zero.

The design of such an attack is a great open question. In the context of data poisoning, there appears to be a major difference between the behavior of simple models such as linear regression, and overparameterized models, such as neural networks. Consider the case where we are doing linear regression, that is, we have data points $(X_1, y_1), \ldots, (X_n, y_n)$ so that $y = (\theta^*)^\top X$, and suppose we add in εn copies of the point (Z, a). Then, if we run least squares on this noisy data, it is attempting to optimize the following loss:

$$L(\theta) = \sum_{i=1}^{n} (\theta^\top X_i - y_i)^2 + \varepsilon n (\theta^\top Z - a)^2 \ .$$

Then if we let $a = 0$, the gradient of L is given by

$$\nabla_\theta L = \nabla_{\text{true}} + \varepsilon n \left\langle \theta, Z \right\rangle Z \ ,$$

where ∇_{true} is the gradient on clean data. In particular, it is supposed to be close to zero at θ^*. However, notice that we can trivially design Z so that the gradient is massive at θ^*. Thus the data poisoning attack has been able to "entangle" the bad data with the good data, causing an almost-complete degradation of test accuracy.

Intuitively, the difficulty with constructing such an attack for deep nets is precisely the fact that neural networks empirically seem to be able to memorize the training data. As a result, even if we mix the training data with a small fraction of corrupted data, the neural network seems to be able to separate these two sub-populations out. As a result, it will only suffer ε loss in test accuracy on clean data. Any attack that is able to improve upon this rate must somehow circumvent this barrier.

References

[1] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[2] Eric Wong, Frank Schmidt, and Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In *International Conference on Machine Learning*, pages 6808–6817, 2019.

[3] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *Advances in Neural Information Processing Systems*, pages 8000–8010, 2018.

[4] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks.

[5] Wenbo Guo, Lun Wang, Xinyu Xing, Min Du, and Dawn Song. Tabor: A highly accurate approach to inspecting and restoring trojan backdoors in ai systems. *arXiv preprint arXiv:1908.01763*, 2019.