ICLR Paper Reading: Adversarial Robustness

Runtian Zhai

MSRA

v-ruzhai@microsoft.com

October 8, 2019

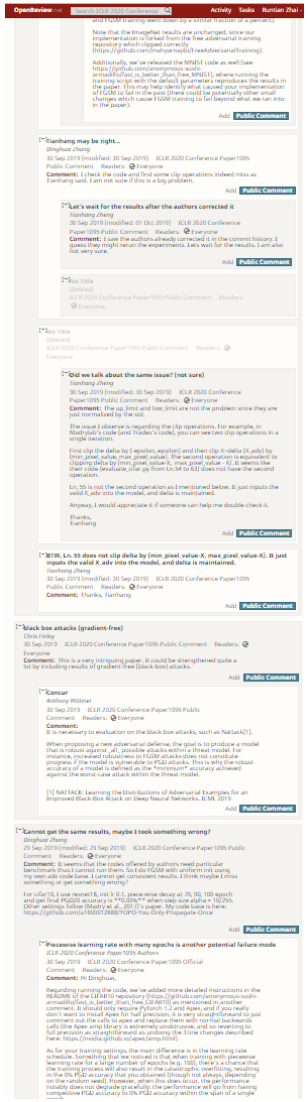
Outline

1. Fast is Better Than Free
2. Robust Single-step Adversarial Training
3. Adversarial Training At Scale

Fast is Better Than Free: Revisiting Adversarial Training

ICLR 2020 Submission

Suspected authors: Eric Wong & Zico Kolter



Background

- **Adversarial training** is the state-of-the-art method to train adversarially robust classifiers. It works in the following way: given an attack algorithm $\mathcal{A}$, during each iteration:
  1. Sample a minibatch $(x_1, y_1), \cdots, (x_b, y_b)$.
  2. Compute $\mathcal{A}(x_1), \cdots, \mathcal{A}(x_b)$.
  3. Use $(\mathcal{A}(x_1), y_1), \cdots, (\mathcal{A}(x_b), y_b)$ as the batch to train the model.
- Although adversarial training can produce empirically robust models, it runs quite slowly. The reason is that the attack algorithm $\mathcal{A}$ is usually time-consuming, and adversarial training needs to run $\mathcal{A}$ during each iteration.

Related Work: Faster Adversarial Training

The most popular choice of $\mathcal{A}$ is PGD-k attack, which maximizes the loss function through k-step gradient ascent. Two NeurIPS 2019 papers propose methods to accelerate PGD based adversarial training:

- Free adversarial training [1] replays a minibatch m times. It uses $k + m - 1$ steps to generate m adversarial examples.
- YOPO-m-n [2] also replays a minibatch m times. In addition, instead of running k-step gradient ascent, it runs one step of gradient ascent followed by n steps of gradient ascent that is *restricted within the first layer*.

Main Argument

- This paper proposes to accelerate adversarial training by using a weaker attack $\mathcal{A}_0$ during training. $\mathcal{A}_0$ is chosen to be FGSM + random start.
- The authors argue that models trained with the weaker attack are also robust against strong attacks such as PGD.

Implementation Detail

- The paper focuses on l_∞ attacks, i.e. the perturbation δ satisfies $\|\delta\|_\infty \leq \epsilon$ for a given $\epsilon > 0$.
- $\mathcal{A}_0$ works in the following way:
  1. Sample δ from the uniform distribution over the perturbation region $\{\delta : \|\delta\|_\infty \leq \epsilon\}$.
  2. Update δ with one step of gradient ascent to maximize the loss function (FGSM) with step size α.
  3. Clip δ back to the perturbation region.

Implementation Detail (cont.)

- The authors choose α to be a bit larger than ϵ (they use $\frac{5\epsilon}{4}$), so that the final perturbation is close to the boundary of the perturbation region.
- However, choosing $\alpha = 2\epsilon$ results in 0% robust accuracy. The choice of α is crucial to the final performance.

DAWNBench Improvements

The authors adopt two additional accelerating techniques proposed in the DAWNBench competition:

- Cyclic learning rate: first increase the learning rate from zero to a maximum, and then decay it to zero. This technique decreases the number of training epochs from 200 to 15.
- Mixed-precision arithmetic: Replace single-precision floating numbers with mixed-precision ones which can be computed much faster by GPU.

Claimed Experimental Results

Table 1: Performance of various adversarial training methods on CIFAR10 for $\epsilon = 8/255$

Method	Standard accuracy	PGD ($\epsilon = 8/255$)
FGSM adversarial training		
+ zero init	81.48%	0.00%
+ previous init	85.60%	42.32%
+ random init	84.38%	43.78%
+ $\alpha = 10/255$ step size	83.25%	46.25%
+ $\alpha = 16/255$ step size	85.39%	0.00%
"Free" adversarial training ($m = 8$)	83.80%	46.78%
PGD adversarial training	83.04%	44.56%

Table 3: Time to train a robust CIFAR10 classifier to 45% robust accuracy using various adversarial training methods with the DAWNBench techniques of cyclic learning rates and mixed-precision arithmetic, showing significant speedups for all forms of adversarial training.

Method	Epochs	Seconds/epoch	Total time (minutes)
DAWNBench + PGD	15	91.59	23.14
DAWNBench + Free ($m = 8$)	88	13.46	19.88
DAWNBench + FGSM	15	23.41	6.02
PGD-7 (Madry et al., 2017)[3]	205	1456.22	4965.71
Free ($m = 8$) (Shafahi et al., 2019)[4]	205	197.77	674.39

Open Review Discussions

- An ICML 2019 paper [3] shows that replacing PGD with FGSM during early epochs in PGD based adversarial training does not affect final performance, but it does not replace PGD with FGSM *completely*.
- Some comments pointed out that the original code contains bugs. The authors fixed the bugs and reran the experiments, and claimed that the results were the same. They also released a reproducible MNIST code (with fixed random seed).
- Some comments suggested that early stopping played a very important role. Robust accuracy would drop sharply after some epochs of FGSM training.

Robust Single-step Adversarial Training

ICLR 2020 Submission

Paper Overview

- This paper suggests that previous FGSM training cannot achieve robustness against stronger attacks like PGD because the network tends to overfit FGSM adversarial examples.
- They propose to use dropout to mitigate overfitting. Their experiments show that FGSM training with dropout can also produce models robust against PGD (although less robust than PGD trained ones).

Empirical Study

They discover that during FGSM training, the training loss over FGSM adversarial examples will become smaller than that over clean inputs, while the validation loss over FGSM samples is still large.
On the other hand, the loss behavior of PGD training is consistent on train set and validation set.

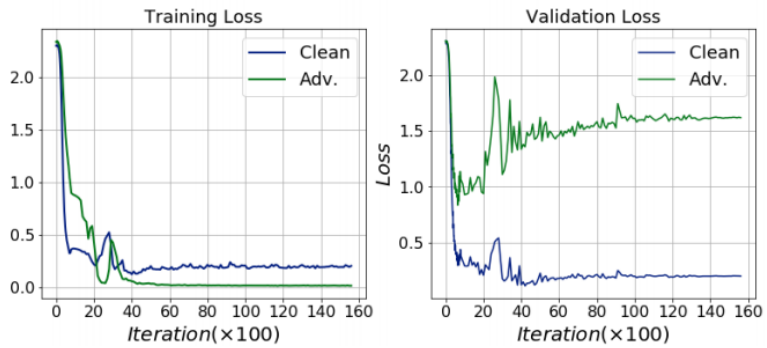


Figure: FGSM Training

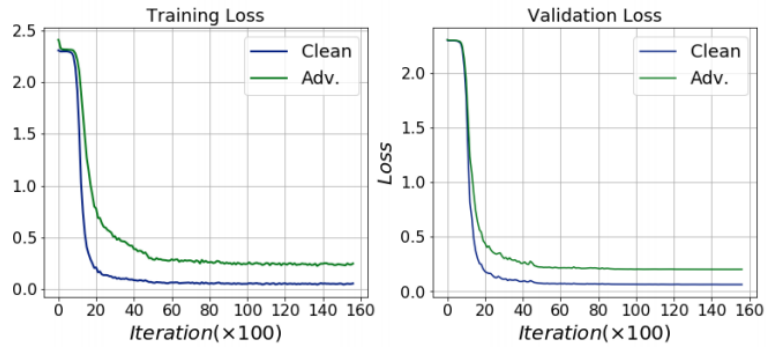


Figure: PGD Training

Additional Details

- While the classic setting puts the dropout layer only after FC+ReLU layer, the authors claim that it is necessary to put dropout layer after every nonlinear layer. They put dropout-2D after Conv2D+ReLU and dropout-1D after FC+ReLU.
- The dropout layers are initialized with a high dropout probability, and the probability is linearly decayed during training.

Experimental Results

Table 3: White-Box attack: Classification accuracy (%) of models trained on CIFAR-10 dataset using different training methods. For all attacks ϵ=8/255 is used and for PGD attack ϵ_{step}=2/255 is used. For both IFGSM and PGD attacks steps is set to 7.

Training Method		**Attack Method**			
		Clean	**FGSM**	**IFGSM**	**PGD**
NT		91.52	14.00	0.00	0.00
FAT		92.42	98.58	0.09	0.05
EAT	A	90.80	82.14	10.56	4.69
	B	90.43	60.59	32.76	28.90
	C	90.28	66.49	36.49	29.41
PAT		79.44	53.25	50.53	50.08
SADS		75.93 ±0.28	48.16 ±0.63	44.29 ±0.53	43.48 ±0.49

Table 4: Comparison of training time per epoch of models trained on MNIST and CIFAR-10 datasets respectively, obtained for different training methods. For PAT, $steps$=40 is used for MNIST dataset and $steps$=7 is used for CIFAR-10 dataset. † For EAT, training time of pre-trained source models are not considered.

Method	Training time per epoch (sec.)	
	MNIST	CIFAR-10
NT	$\sim$ 2.7	$\sim$ 31
FAT	$\sim$ 4.1	$\sim$ 53
EAT†	$\sim$ 5.5	$\sim$ 59
PAT	$\sim$ 53.0	$\sim$ 238
SADS	$\sim$ 4.3	$\sim$ 56

Discussion

- Does random start in *Fast is Better Than Free* prevent the model from overfitting FGSM adversarial examples?
- Is it possible that PGD training also overfits PGD adversarial examples?

Intriguing Properties of Adversarial Training At Scale

ICLR 2020 Submission

Cihang Xie, Alan Yuille (JHU)

Recycled from NeurIPS 2019

Paper Overview

This paper studies two components that affect adversarial training:

- Batch Normalization: BN layers have negative impact on adversarial training.
- Network Depth: Adversarial training requires much deeper networks, and robust accuracy can be improved by using deeper models (e.g. ResNet-152 $\rightarrow$ ResNet-638).

Negative Effect of Batch Normalization

- In PGD adversarial training we only train on perturbed samples $\mathcal{A}(x)$ and forget about the original inputs x. It is observed that if we train on both $\mathcal{A}(x)$ and x, we will get lower robust accuracy.
- The authors argue that the problem lies in BN, because clean images and adversarial images are drawn from two different domains. Consequently, BN learns the statistics of a mixture of clean and perturbed images.
- The authors propose to solve this issue by entangling the two domains and using a mixture BN (MBN). They put clean and adversarial images in different minibatches and use separate BN layers.

Positive Effect of Deeper Networks

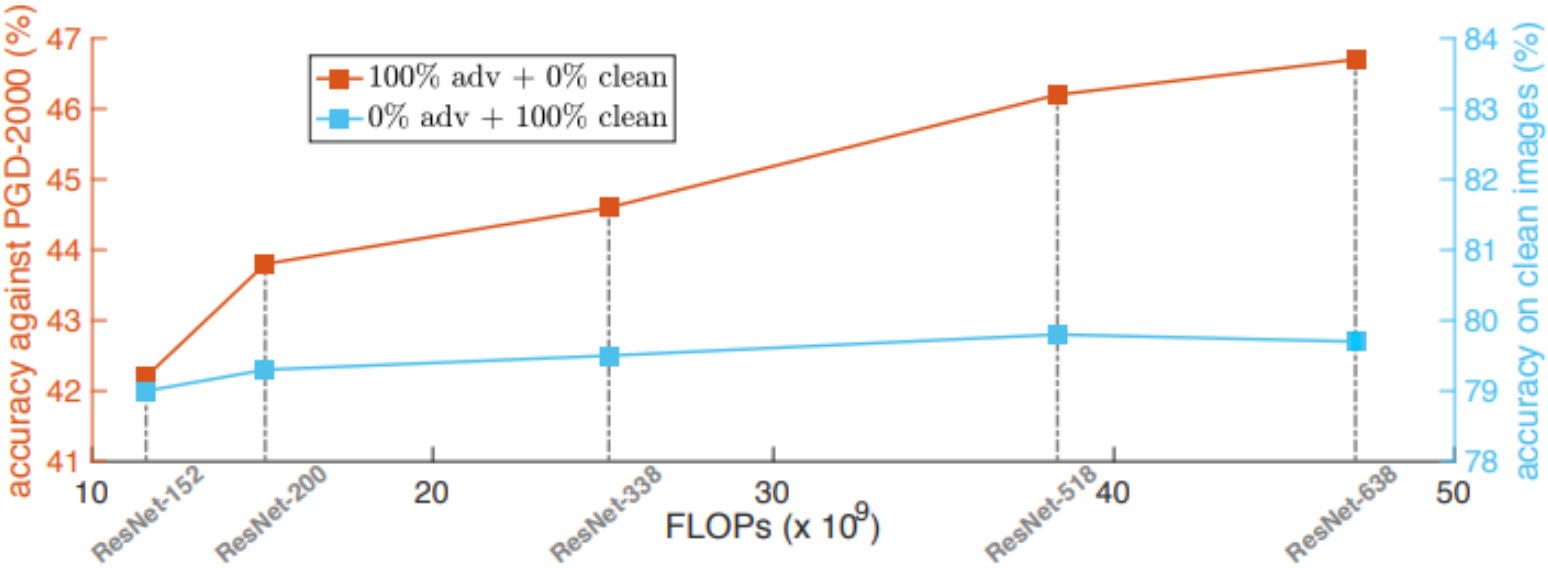


Figure 7: Compared to traditional image classification tasks, adversarial training exhibits a stronger demand on deeper networks. The performance gain of traditional image classification becomes marginal after ResNet-200 however the adversarial robustness continues to grow even for ResNet-638.

Figure: ImageNet Results

References

Shafahi et al. (2019)

Adversarial Training for Free!

NeurIPS 2019

Zhang et al. (2019)

You Only Propagate Once: Accelerating Adversarial Training via Maximal Principle

NeurIPS 2019

Wang et al. (2019)

On the Convergence and Robustness of Adversarial Training

ICML 2019