On Fragile Features and Batch Normalization in Adversarial Training

Nils Philipp Walter David Stutz Bernt Schiele

max planck institut informatik

Motivation

1. BN argued to **cause adversarial vulnerability**
2. **BN statistics** of clean adversarial examples **differ**
3. Non-robust Neural Networks contain robust sub-networks
4. Training BN and only BN (i.e. random filters) achieves decent accuracy

Research Question

Is it possible to **adversarially finetune** BN-layers to disable **fragile features** and boost **adversarial robustness**?

Experimental Setup

Compare adversarially fine-tuning BN (BN Params) to standard baseline (Normal/Adv) and training BN and only BN (Adv BN Only):

Training	Model	All	BN Prms	BN stats	%prms
From scratch	Normal/Adv	✓	✓		100.00
	Adv BN Only	✗	✗		0.15
Fine-tune	BN Stats	✗	✗	✓	(0.15)
	BN Params	✗	✓	✓	0.15

Implementation details:

- ResNet20 on CIFAR10
- Evaluation with AutoAttack, L_infty eps = 8/255
- AutoAugment Normal, crop+flips for Adv

Quantitive Results

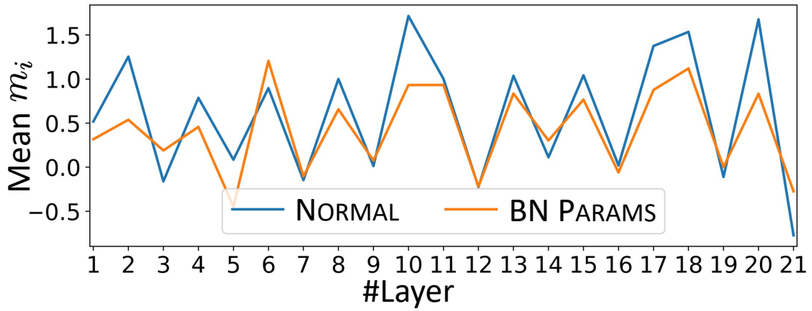
Model	Test Error	Robust Test Error
Normal	4.11	99.8
Adv	17.67	58.4
+BN Only	43.33	98.0
+BN Only + log + conv1	35.22	98.1
BN Stats	6.33	99.6
BN Params	29.67	70.5
BN Params + log + conv1	26.89	68.2

Hypothesis: BN Params Turns Off (Fragile) Features

To compare BN parameters, rewrite BN as:

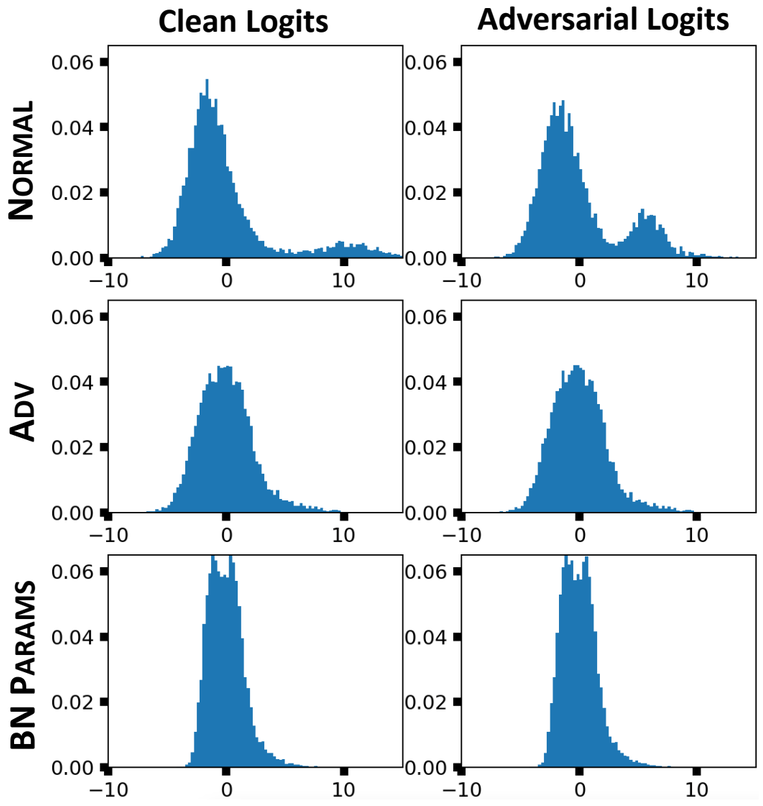
$$\mathbf{BN}(z_i) = \left(\frac{\gamma_i}{\sqrt{\sigma_i^2+\epsilon}}\right) z_i + \left(-\frac{\gamma_i \mu_i}{\sqrt{\sigma_i^2+\epsilon}} + \beta_i\right)$$
$$= m_i \cdot z_i + b_i$$

➤Analysis can be reduced to m_i and b_i



- BN Params seems to disable part of the fragile features
- Average difference in mean weight is -0.177

BN Params Mimics Logits of Adv



Conlusion

- Only adjusting statistics has no effect
- BN Params obtains non-trivial adversarial robustness
- Indicates that Normal learns some robust featues and fragile ones can be ignored
- Training BN on random features (Adv BN Only) does not provide adversarial robustness