COMP6211I: Trustworthy Machine Learning

Test-time Integrity (defenses)

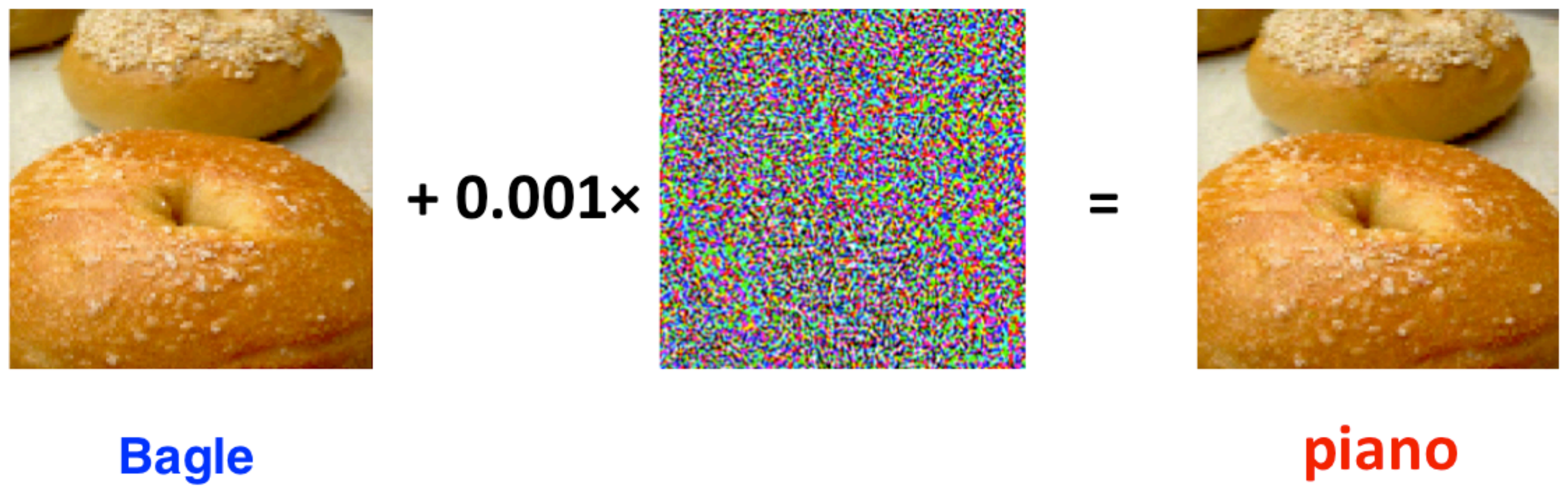
Minhao CHENG

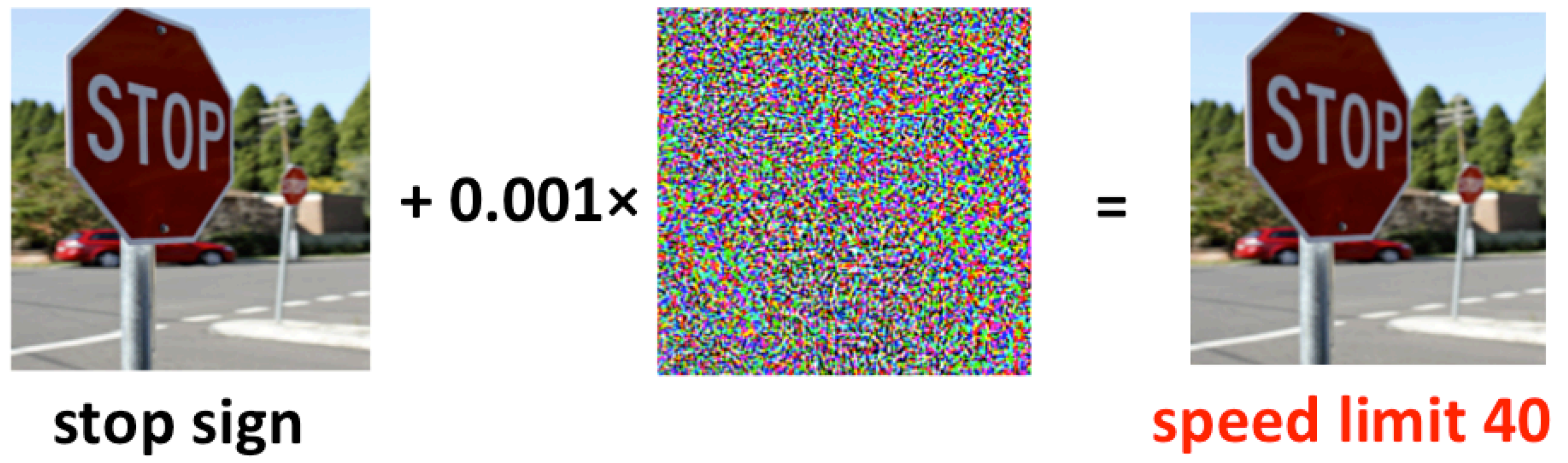
THE DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

Test-time integrity

Adversarial examples

- An adversarial example can easily fool a deep network
- Robustness is critical in real systems





Adversarial example

White-box adversarial attack

- If there is $||x - x_0||_\infty$ constraint, we could turn to solve by
- FGSM attack [GSS15]:
  - $x \leftarrow \text{proj}_{x+\mathcal{S}}(x_0 + \alpha \text{sign}(\nabla_{x_0} \ell(\theta, x, y)))$
- PGD attack [KGB17, MMS18]
  - $x^{t+1} \leftarrow \text{proj}_{x+\mathcal{S}}(x^t + \alpha \text{sign}(\nabla_{x^t} \ell(\theta, x, y)))$

Adversarial defense

Adversarial training

- Adversarial training [MMS18]:

  - $$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\max_{\delta \in \mathcal{S}} L(\theta, x+\delta, y) \right] .$$

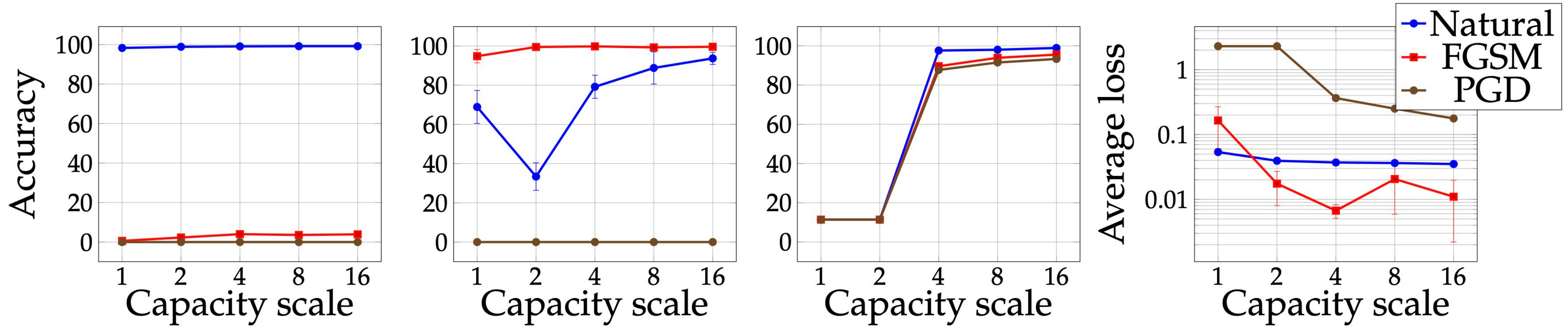
- Solve the inner loop by

  - $$x^{t+1} = \Pi_{x+\mathcal{S}} \left(x^t + \alpha \operatorname{sgn}(\nabla_x L(\theta, x, y)) \right)$$

Adversarial training

Capacity is crucial

MNIST



CIFAR10

	Simple	Wide
Natural	92.7%	95.2%
FGSM	27.5%	32.7%
PGD	0.8%	3.5%

(a) Standard training

Simple	Wide
87.4%	90.3%
90.9%	95.1%
0.0%	0.0%

(b) FGSM training

Simple	Wide
79.4%	87.3%
51.7%	56.1%
43.7%	45.8%

(c) PGD training

Simple	Wide
0.00357	0.00371
0.0115	0.00557
1.11	0.0218

(d) Training Loss

Figure 4: The effect of network capacity on the performance of the network. We trained MNIST and CIFAR10 networks of varying capacity on: (a) natural examples, (b) with FGSM-made adversarial examples, (c) with PGD-made adversarial examples. In the first three plots/tables of each dataset, we show how the standard and adversarial accuracy changes with respect to capacity for each training regime. In the final plot/table, we show the value of the cross-entropy loss on the adversarial examples the networks were trained on. This corresponds to the value of our saddle point formulation (2.1) for different sets of allowed perturbations.

Adversarial training

Problems

- Huge overhead
    - Increase training time by an order magnitude (7x if 7 step PGD)
- Fast method like FGSM doesn’t work
    - Easily be attacked by strong attackers such as C&W attack

Fast Adversarial training

- Solve the following optimization:
  - $$\min_{\theta} \sum_{i} \max_{\delta \in \Delta} \ell(f_\theta(x_i + \delta), y_i).$$
- Solve the inner max by FGSM
  - $$\delta^\star = \epsilon \cdot \mathrm{sign}(\nabla_x \ell(f(x), y)).$$

Free Adversarial training

Attempts

- Free" adversarial training: use each inner max to update

Algorithm 2 "Free" adversarial training for T epochs, given some radius ϵ, N minibatch replays, and a dataset of size M for a network f_θ

$\delta = 0$
// Iterate T/N times to account for minibatch replays and run for T total epochs
for $t = 1 \ldots T/N$ **do**
  for $i = 1 \ldots M$ **do**
    // Perform simultaneous FGSM adversarial attack and model weight updates T times
    for $j = 1 \ldots N$ **do**
      // Compute gradients for perturbation and model weights simultaneously
      $\nabla_\delta, \nabla_\theta = \nabla \ell(f_\theta(x_i + \delta), y_i)$
      $\delta = \delta + \epsilon \cdot \mathrm{sign}(\nabla_\delta)$
      $\delta = \max(\min(\delta, \epsilon), -\epsilon)$
      $\theta = \theta - \nabla_\theta$ *// Update model weights with some optimizer, e.g. SGD*
    end for
  end for
end for

Fast Adversarial training

Algorithm 3 FGSM adversarial training for T epochs, given some radius ϵ, N PGD steps, step size α, and a dataset of size M for a network f_θ

---

for $t = 1 \ldots T$ **do**
  for $i = 1 \ldots M$ **do**
    // Perform FGSM adversarial attack
    $\delta = \text{Uniform}(-\epsilon, \epsilon)$
    $\delta = \delta + \alpha \cdot \text{sign}(\nabla_\delta \ell(f_\theta(x_i + \delta), y_i))$
    $\delta = \max(\min(\delta, \epsilon), -\epsilon)$
    $\theta = \theta - \nabla_\theta \ell(f_\theta(x_i + \delta), y_i)$ *// Update model weights with some optimizer, e.g. SGD*
  end for
end for

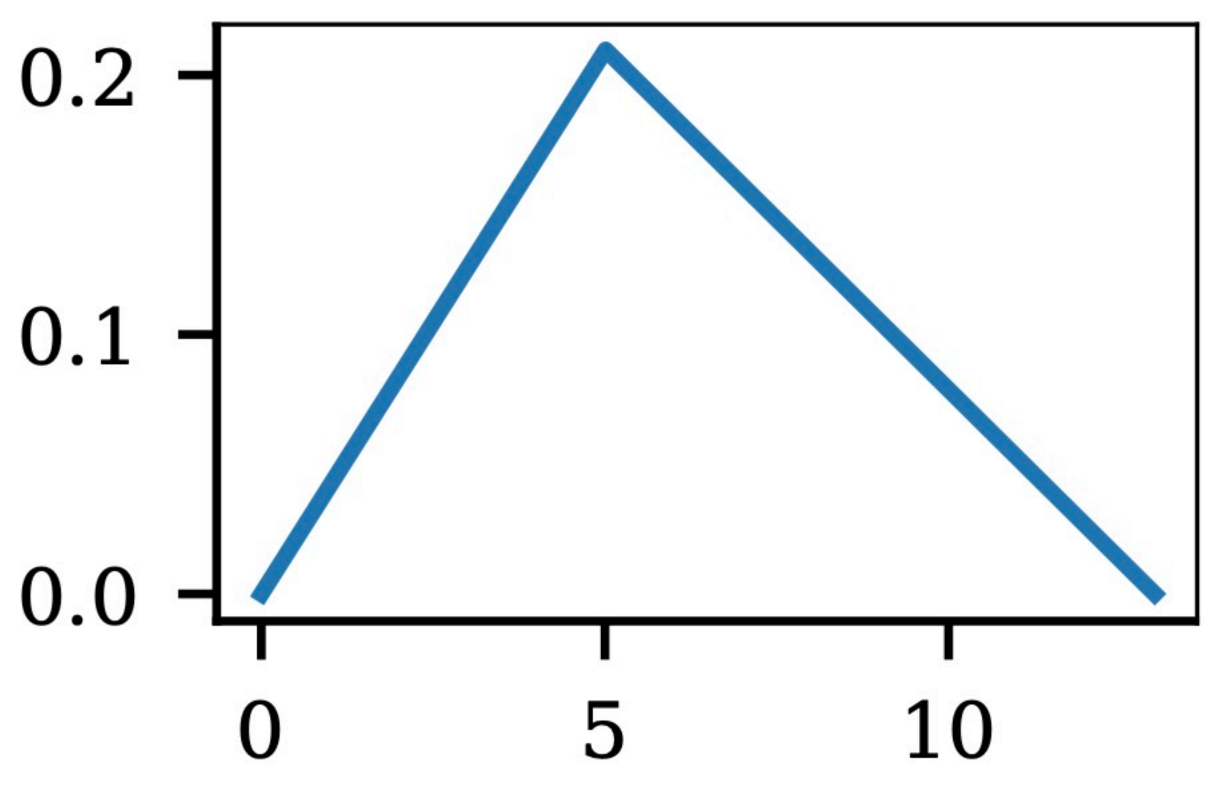
The magic of random initialization

Method	Standard accuracy	PGD ($\epsilon = 8/255$)	Time (min)
FGSM + DAWNBench			
+ zero init	85.18%	0.00%	12.37
+ early stopping	71.14%	38.86%	7.89
+ previous init	86.02%	42.37%	12.21
+ random init	85.32%	44.01%	12.33
+ $\alpha = 10/255$ step size	83.81%	46.06%	12.17
+ $\alpha = 16/255$ step size	86.05%	0.00%	12.06
+ early stopping	70.93%	40.38%	8.81
"Free" ($m = 8$) (Shafahi et al., 2019)[1]	85.96%	46.33%	785
+ DAWNBench	78.38%	46.18%	20.91
PGD-7 (Madry et al., 2017)[2]	87.30%	45.80%	4965.71
+ DAWNBench	82.46%	50.69%	68.8

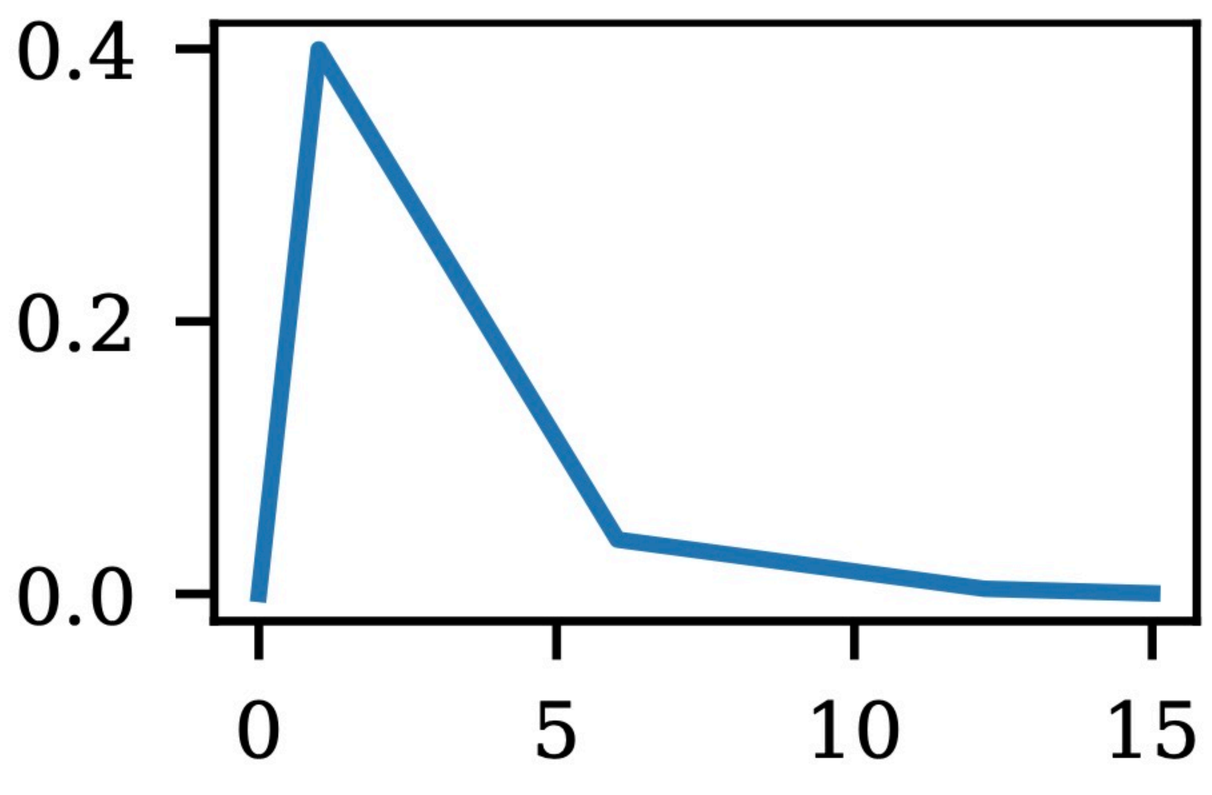
DAWNBench Improvement

Reduce # of training epochs

- Cyclic learning rate
- Mixed-precision arithmetic

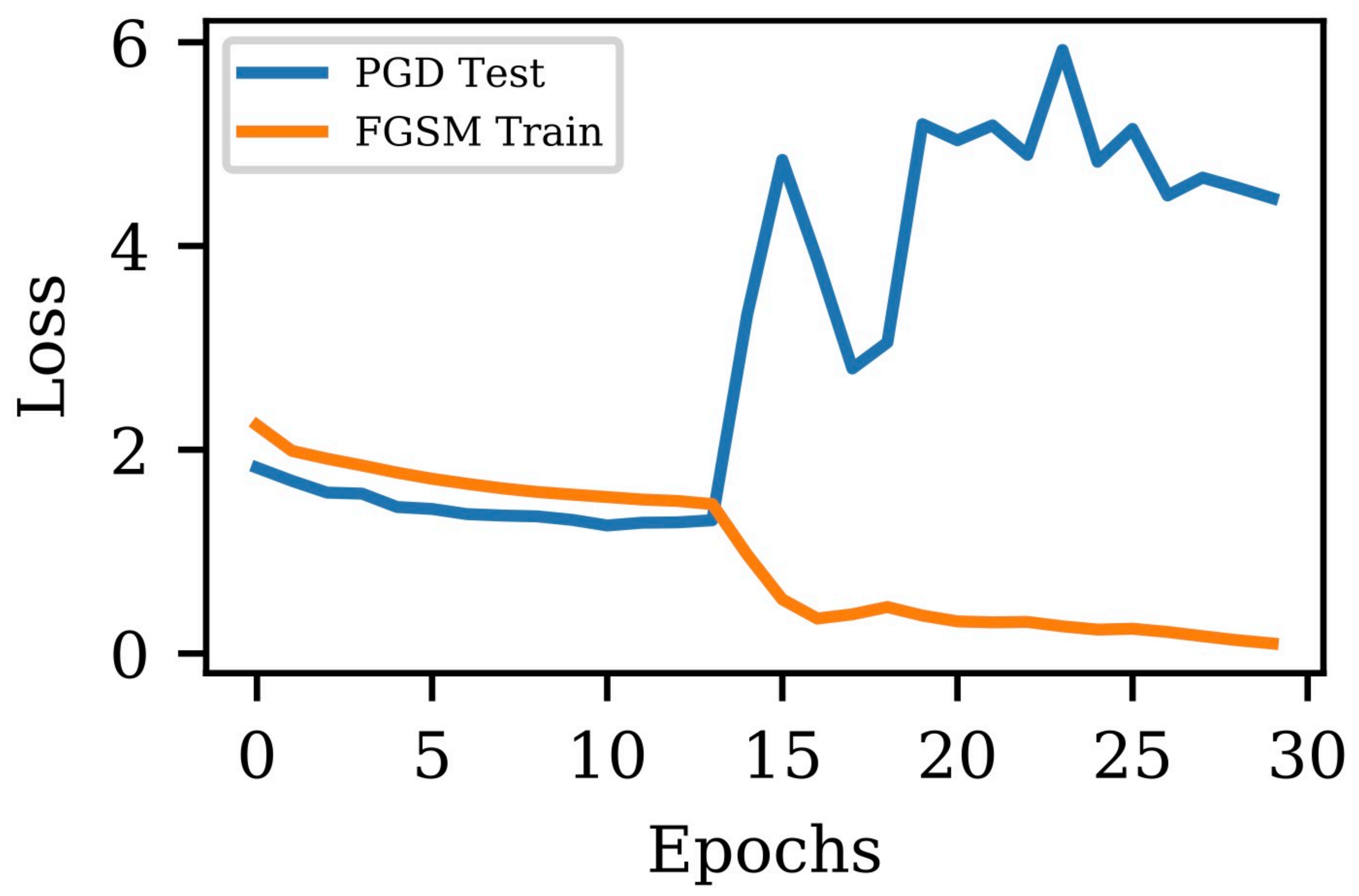


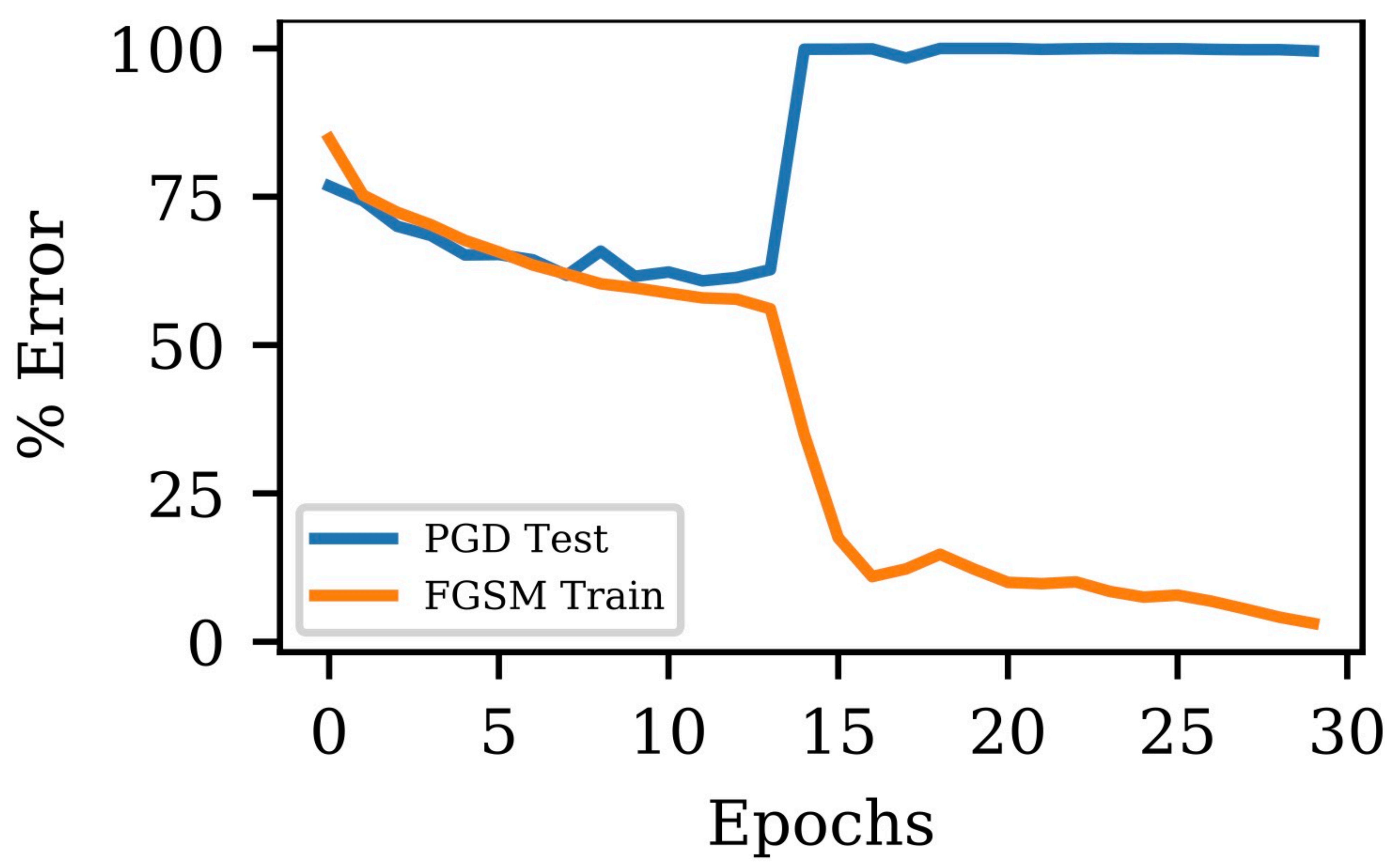
(a) CIFAR10



(b) ImageNet

Catastrophic overfitting





TRADES
Notations

- DB(f) is the decision boundary of f $\{\boldsymbol{x} \in \mathcal{X} : f(\boldsymbol{x}) = 0\}$
- $\mathbb{B}$(DB(f), ϵ) is the neighborhood of decision boundary

  f: $\{\boldsymbol{x} \in \mathcal{X} : \exists \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } f(\boldsymbol{x})f(\boldsymbol{x}') \leq 0\}$

- Robust error $\mathcal{R}_{\text{rob}}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\mathbf{1}\{\exists \boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon) \text{ s.t. } f(\boldsymbol{X}')Y \leq 0\}$
- Natural error $\mathcal{R}_{\text{nat}}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\mathbf{1}\{f(\boldsymbol{X})Y \leq 0\}$
- Boundary error $\mathcal{R}_{\text{bdy}}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\mathbf{1}\{\boldsymbol{X} \in \mathbb{B}(\mathrm{DB}(f), \epsilon), f(\boldsymbol{X})Y > 0\}$

$$\mathcal{R}_{\text{rob}}(f) = \mathcal{R}_{\text{nat}}(f) + \mathcal{R}_{\text{bdy}}(f).$$

TRADES

Main theorem

Theorem 3.1. *Let* $\mathcal{R}_\phi(f) := \mathbb{E}\phi(f(\boldsymbol{X})Y)$ *and* $\mathcal{R}_\phi^* := \min_f \mathcal{R}_\phi(f)$. *Under Assumption 1, for any non-negative loss function* ϕ *such that* $\phi(0) \geq 1$, *any measurable* $f : \mathcal{X} \to \mathbb{R}$, *any probability distribution on* $\mathcal{X} \times \{\pm 1\}$, *and any* $\lambda > 0$, *we have*[1]

$$\begin{aligned}\mathcal{R}_{\mathrm{rob}}(f) - \mathcal{R}_{\mathrm{nat}}^* &\leq \psi^{-1}(\mathcal{R}_\phi(f) - \mathcal{R}_\phi^*) + \Pr[\boldsymbol{X} \in \mathbb{B}(\mathrm{DB}(f), \epsilon), f(\boldsymbol{X})Y > 0] \\ &\leq \psi^{-1}(\mathcal{R}_\phi(f) - \mathcal{R}_\phi^*) + \mathbb{E} \max_{\boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon)} \phi(f(\boldsymbol{X}')f(\boldsymbol{X})/\lambda).\end{aligned}$$

TRADES
Optimization

- Solve the following optimization to minimize $\mathcal{R}_{\mathrm{rob}}(f) - \mathcal{R}^*_{\mathrm{nat}}$

$$\min_f \mathbb{E}\Big\{ \underbrace{\phi(f(\boldsymbol{X})Y)}_{\text{for accuracy}} + \underbrace{\max_{\boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X},\epsilon)} \phi(f(\boldsymbol{X})f(\boldsymbol{X}')/\lambda)}_{\text{regularization for robustness}} \Big\}.$$

- Comparison with Adversarial training

$$\min_f \mathbb{E}\left\{ \max_{\boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X},\epsilon)} \phi(f(\boldsymbol{X}')Y) \right\},$$

TRADS

Controlling trade-off

Table 4: Sensitivity of regularization hyperparameter λ on MNIST and CIFAR10 datasets.

	MNIST		CIFAR10	
$1/\lambda$	$\mathcal{A}_{\rm rob}(f)$ (%)	$\mathcal{A}_{\rm nat}(f)$ (%)	$\mathcal{A}_{\rm rob}(f)$ (%)	$\mathcal{A}_{\rm nat}(f)$ (%)
0.1	91.09 ± 0.0385	99.41 ± 0.0235	26.53 ± 1.1698	91.31 ± 0.0579
0.2	92.18 ± 0.0450	99.38 ± 0.0094	37.71 ± 0.6743	89.56 ± 0.2154
0.4	93.21 ± 0.0660	99.35 ± 0.0082	41.50 ± 0.3376	87.91 ± 0.2944
0.6	93.87 ± 0.0464	99.33 ± 0.0141	43.37 ± 0.2706	87.50 ± 0.1621
0.8	94.32 ± 0.0492	99.31 ± 0.0205	44.17 ± 0.2834	87.11 ± 0.2123
1.0	94.75 ± 0.0712	99.28 ± 0.0125	44.68 ± 0.3088	87.01 ± 0.2819
2.0	95.45 ± 0.0883	99.29 ± 0.0262	48.22 ± 0.0740	85.22 ± 0.0543
3.0	95.57 ± 0.0262	99.24 ± 0.0216	49.67 ± 0.3179	83.82 ± 0.4050
4.0	95.65 ± 0.0340	99.16 ± 0.0205	50.25 ± 0.1883	82.90 ± 0.2217
5.0	95.65 ± 0.1851	99.16 ± 0.0403	50.64 ± 0.3336	81.72 ± 0.0286

TRADS

Main results

[WSMK18]	robust opt.	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	27.07%	23.54%
[MMS+18]	robust opt.	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	87.30%	**47.04%**
[ZSLG16]	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	94.64%	0.15%
[KGB17]	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	85.25%	45.89%
[RDV17]	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	95.34%	0%
TRADES ($1/\lambda = 1$)	regularization	$\text{FGSM}^{1,000}$ (PGD)	CIFAR10	0.031 (ℓ_∞)	88.64%	48.90%
TRADES ($1/\lambda = 6$)	regularization	$\text{FGSM}^{1,000}$ (PGD)	CIFAR10	0.031 (ℓ_∞)	84.92%	**56.43%**
TRADES ($1/\lambda = 1$)	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	88.64%	49.14%
TRADES ($1/\lambda = 6$)	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	84.92%	**56.61%**
TRADES ($1/\lambda = 1$)	regularization	DeepFool (ℓ_∞)	CIFAR10	0.031 (ℓ_∞)	88.64%	59.10%
TRADES ($1/\lambda = 6$)	regularization	DeepFool (ℓ_∞)	CIFAR10	0.031 (ℓ_∞)	84.92%	61.38%
TRADES ($1/\lambda = 1$)	regularization	LBFGSAttack	CIFAR10	0.031 (ℓ_∞)	88.64%	84.41%
TRADES ($1/\lambda = 6$)	regularization	LBFGSAttack	CIFAR10	0.031 (ℓ_∞)	84.92%	81.58%
TRADES ($1/\lambda = 1$)	regularization	MI-FGSM	CIFAR10	0.031 (ℓ_∞)	88.64%	51.26%
TRADES ($1/\lambda = 6$)	regularization	MI-FGSM	CIFAR10	0.031 (ℓ_∞)	84.92%	57.95%
TRADES ($1/\lambda = 1$)	regularization	C&W	CIFAR10	0.031 (ℓ_∞)	88.64%	84.03%
TRADES ($1/\lambda = 6$)	regularization	C&W	CIFAR10	0.031 (ℓ_∞)	84.92%	81.24%
[SKC18]	gradient mask	[ACW18]	MNIST	0.005 (ℓ_2)	-	55%
[MMS+18]	robust opt.	FGSM^{40} (PGD)	MNIST	0.3 (ℓ_∞)	99.36%	96.01%
TRADES ($1/\lambda = 6$)	regularization	$\text{FGSM}^{1,000}$ (PGD)	MNIST	0.3 (ℓ_∞)	99.48%	95.60%
TRADES ($1/\lambda = 6$)	regularization	FGSM^{40} (PGD)	MNIST	0.3 (ℓ_∞)	99.48%	96.07%
TRADES ($1/\lambda = 6$)	regularization	C&W	MNIST	0.005 (ℓ_2)	99.48%	99.46%