CS 499/579: TRUSTWORTHY ML

ADVERSARIAL EXAMPLES: WHITE-BOX ATTACKS

Instructor: Sanghyun Hong

sanghyun.hong@oregonstate.edu

NOTES

- Call for actions
  - In-class presentation sign-ups
  - Term project team-up
  - On-boarding quiz on Canvas
  - GitHub classroom registration

TOPICS FOR PART I – ADVERSARIAL EXAMPLES

- Research questions
  - What are the adversarial examples?
  - How can we find adversarial examples?
  - How can we exploit them in practice?
  - How can we defeat adversarial examples?

How can we train models robust to adversarial inputs?

Towards deep learning models resistant to adversarial attacks, Madry et al., ICLR 2018

How did the research go?

- Many attack proposals
  - FGSM
  - JSMA
  - DeepFool
  - DeepXplore[1]
  - C&W
  - …
- Many defense proposals
  - Regularization … broken
  - Defensive distillation … broken
  - Adversarial training ... but with which attack?
  - …

Pei et al., DeepXplore: Automated Whitebox Testing of Deep Learning Systems, SOSP 2017

HOW DID THE RESEARCH GO?

- Main research question
  - How can we train neural networks robust to adversarial examples?

REVISITING THE FORMULATION

- Test-time (evasion) attack
  - Suppose
    - A test-time input (x, y)
    - $(x, y) \sim D$, D: data distribution; $x \in R^d$ and $y \in [k]$; $x \in [0, 1]$
    - A NN model f and its parameters θ
    - $L(\theta, x, y)$: a loss function
  - Objective
    - Find an $x^{adv} = x + \delta$ such that $f(x^{adv}) \neq y$ while $||\delta||_p \leq \varepsilon$

REVISITING THE FORMULATION

- Test-time (evasion) attack
  - Suppose
    - A test-time input (x, y)
    - $(x, y) \sim D$, D: data distribution; $x \in R^d$ and $y \in [k]$; $x \in [0, 1]$
    - A NN model f and its parameters θ
    - $L(\theta, x, y)$: a loss function
  - Attacker's objective
    - Find an $x^{adv} = x + \delta$ such that $\max_{\delta \in S} L\left(\theta, x^{adv}, y\right)$ while $||\delta||_p \leq \varepsilon$

REVISITING THE FORMULATION

- Test-time (evasion) attack
  - Suppose
    - A test-time input (x, y)
    - $(x, y) \sim D$, D: data distribution; $x \in R^d$ and $y \in [k]$; $x \in [0, 1]$
    - A NN model f and its parameters θ
    - $L(\theta, x, y)$: a loss function
  - Attacker's objective
    - Find an $x^{adv} = x + \delta$ such that $\max_{\delta \in S} L\left(\theta, x^{adv}, y\right)$ while $||\delta||_p \leq \varepsilon$
  - Defender's objective
    - Train a neural network f robust to adversarial attacks
    - Find θ such that $\min_{\theta} \rho(\theta)$ where $\rho(\theta) = \mathrm{E}_{(x,y)\sim D}\left[L\left(\theta, x^{adv}, y\right)\right]$

PUTTING ALL TOGETHER

- (Models resilient to) test-time (evasion) attack
  - Suppose
    - A test-time input (x, y)
    - $(x, y) \sim D$, D: data distribution; $x \in R^d$ and $y \in [k]$; $x \in [0, 1]$
    - A NN model f and its parameters θ
    - $L(\theta, x, y)$: a loss function
  - Min-max optimization (between attacker's and defender's objectives)
    - Find $\min_{\theta} \rho(\theta)$ where $\rho(\theta) = \mathrm{E}_{(x,y)\sim D}\left[\max_{\delta \in S} L(\theta, x + \delta, y)\right]$ while $||\delta||_p \leq \varepsilon$
    - s: a set of test-time samples

SADDLE POINT PROBLEM: INNER MAXIMIZATION AND OUTER MINIMIZATION

INNER MAXIMIZATION USING THE FIRST-ORDER ADVERSARY

- Revisit FGSM (Fast Gradient Sign Method)

$$x + \varepsilon \operatorname{sgn}(\nabla_x L(\theta, x, y)).$$

  - FGSM can be viewed as a simple one-step toward maximizing the loss (inner part)

INNER MAXIMIZATION

- Revisit FGSM (Fast Gradient Sign Method)

$$x + \varepsilon \operatorname{sgn}(\nabla_x L(\theta, x, y)).$$

  - FGSM can be viewed as a simple one-step toward maximizing the loss (inner part)

- PGD (Projected Gradient Descent)

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

FGSM

  - Multi-step adversary; much stronger than FGSM attack

INNER MAXIMIZATION

- PGD (Projected Gradient Descent)

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

  - Multi-step adversary; much stronger than FGSM attack
  - Hyper-parameters
    - t: number of iterations
    - α: step-size
    - ε: perturbation bound $|x^* - x|_p$
  - Notation: PGD-t, bounded by ε, used the step-size of α

OUTER MINIMIZATION

- PGD (Projected Gradient Descent)

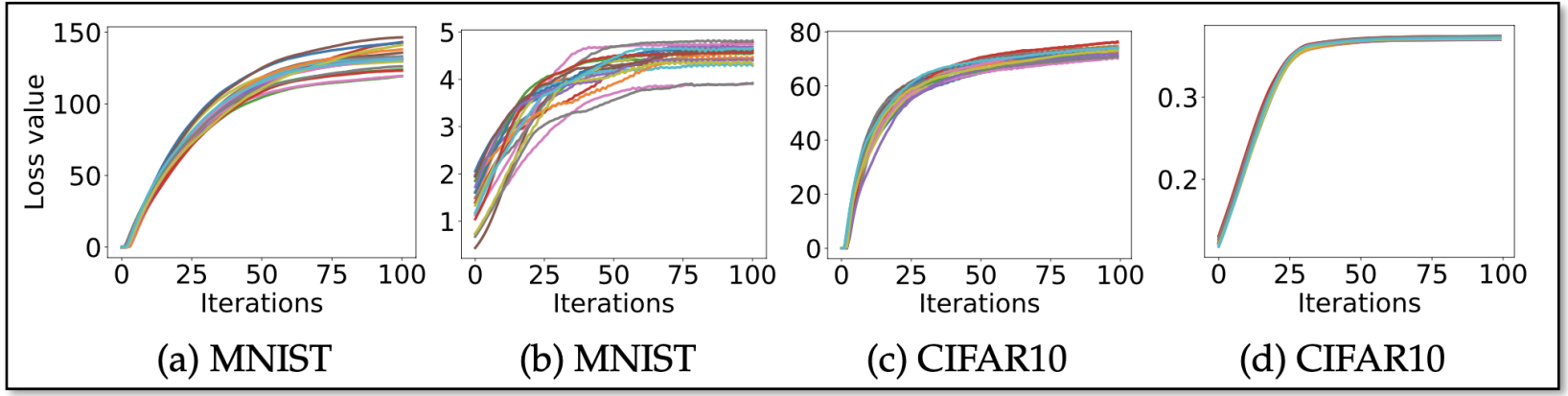
$$x^{t+1} = \Pi_{x+\mathcal{S}} \left(x^t + \alpha \operatorname{sgn}(\nabla_x L(\theta, x, y)) \right).$$

  - Multi-step adversary; much stronger than FGSM attack

- Adversarial training
  - Make a model do correct prediction on adversarial examples
  - Training procedure
    - At each iteration of training
    - Craft PGD-t adversarial examples
    - Update the model towards making it correct on those adv examples

EVALUATION

- Findings
  - (1, 3) PGD increases the loss values in a fairly consistent way
  - (2, 4) Models trained with PGD attacks are resilient to the same attacks

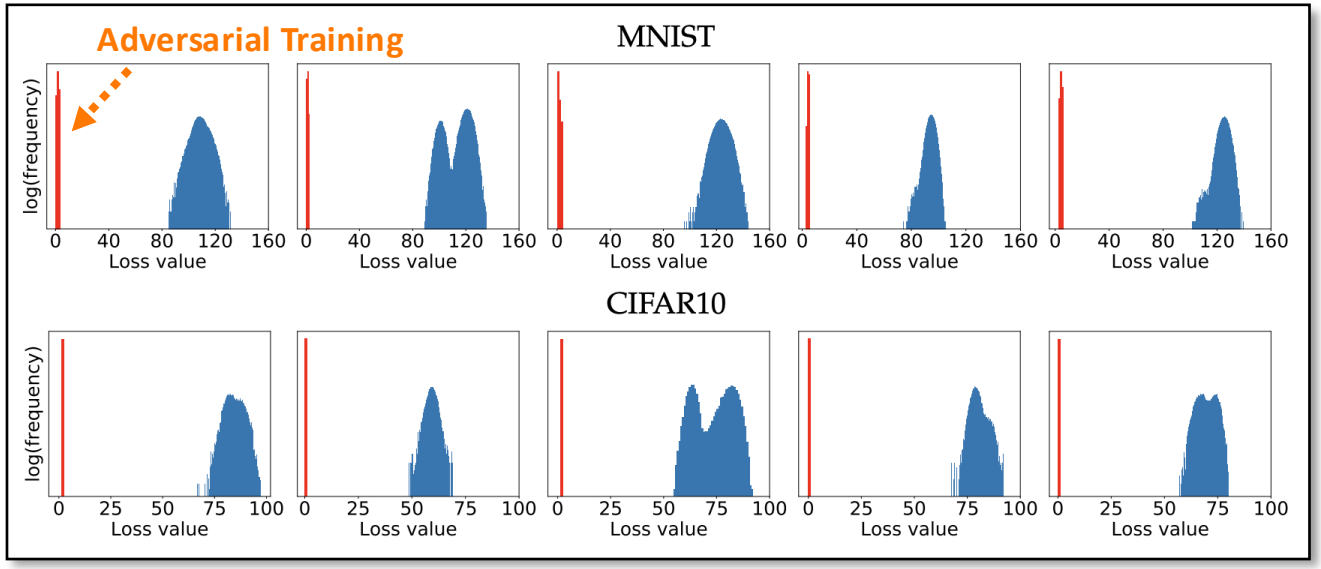


Adversarial Training

Adversarial Training

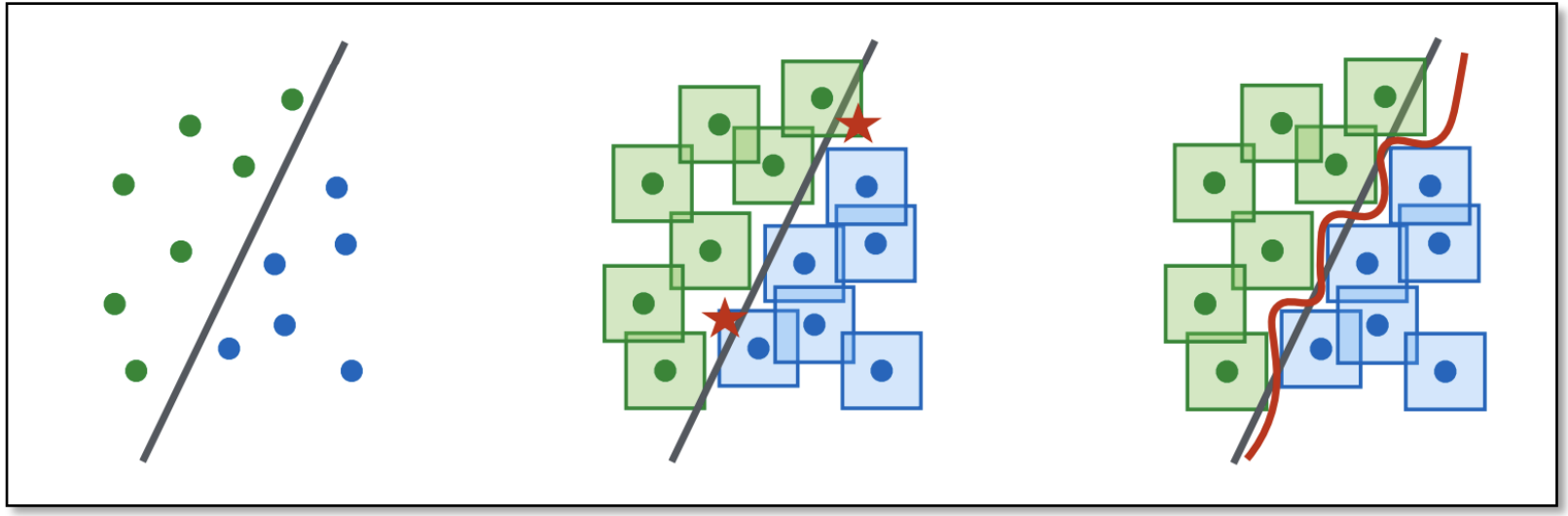
EVALUATION

- Findings
  - PGD increases the loss values in a fairly consistent way
  - Models trained with PGD attacks are resilient to the same attacks
  - Final loss of PGD attacks are concentrated (both for defended/undefended models)



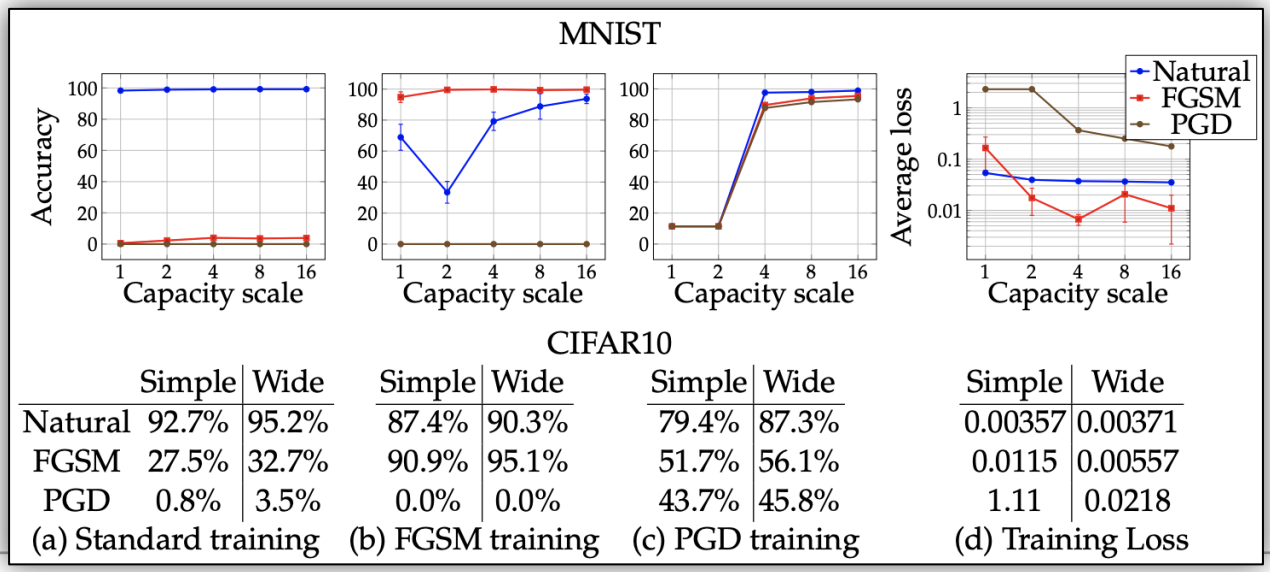
EVALUATION

- Why adversarial training (AT) works?
  - Capacity is crucial for the robustness: robust models need complex decision boundary
  - Capacity alone helps: high-capacity models show more robustness w/o AT



EVALUATION

- … Cont'd
  - Capacity is crucial for the robustness: robust models need complex decision boundary
  - Capacity alone helps: high-capacity models show more robustness w/o AT
  - AT with weak attacks (like FGSM) can't defeat a strong one like PGD
  - (optional) Robustness may be at odds with accuracy



	Simple	Wide	Simple	Wide	Simple	Wide	Simple	Wide
Natural	92.7%	95.2%	87.4%	90.3%	79.4%	87.3%	0.00357	0.00371
FGSM	27.5%	32.7%	90.9%	95.1%	51.7%	56.1%	0.0115	0.00557
PGD	0.8%	3.5%	0.0%	0.0%	43.7%	45.8%	1.11	0.0218
	(a) Standard training		(b) FGSM training		(c) PGD training		(d) Training Loss	

SUMMARY

- Bottom-line
  - PGD is a strong attack we can use
  - Training a model with PGD can make it resilient to the first-order adversary
  - To achieve such robustness, we need sufficient model complexity

Thank You!

Instructor: Sanghyun Hong

https://secure-ai.systems/courses/MLSec/F23