CS 499/579: Trustworthy ML

04.11: Adversarial Attacks

Tu/Th 10:00 – 11:50 am (Recorded lecture)

Instructor: Sanghyun Hong

sanghyun.hong@oregonstate.edu

Heads-up!

- Due dates
  - 4/11: Team-up for the term project (Use Discord + by today!)
  - 4/13: HW 1 due
  - 4/18: Written paper critique
- Announcement
  - 4/08: You now have GPU cluster access
  - 4/13: No lecture (Use more time for HW!)
- Call for actions
  - Term project team-up (Use Discord + by today!)
  - In-class presentation sign-ups

Recap & topics for today

- Research questions
  - How can we find adversarial examples?
    - What is the attack scenario (threat model)?
    - What are the goals for the attacker (under the threat model)?
    - What is the right method for finding adversarial examples?
    - What properties do an adversarial examples exploit?
  - How can a real-world attacker exploit them in practice?
    - How effective adversarial attacks in real-world scenarios?
    - What can an adversary do to make adversarial attack effective?
  - How can we remove adversarial examples?

SRQ 3: What is the right method for finding adv. examples?

- Why is it important?
  - Quantify the robustness of our models to adversarial attacks
  - Many benefits:
    - Offer a vulnerability assessment tool for researchers and practitioners
    - Become a set of tools for evaluating effectiveness of future defense proposals

SRQ 3: What is the right method for finding adv. examples?

- Today's first work:
  - Carlini *et al.*, Towards Evaluating the Robustness of Neural Networks, IEEE S&P 2017
- Motivation
  - Many attack proposals
    - (We've looked at it) FGSM and BIM + ILL-Class
    - JSMA
    - DeepFool
    - DeepXplore[1]
    - …
  - Defense Proposals
    - Distillation-based defense mechanisms
    - Training neural networks with adversarial examples [adversarial training]
    - …

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

SRQ 3: What is the right method for finding adv. examples?

- Sub-research questions:
  - SRQ 3-1: What attacks should we choose for evaluating the robustness?
  - SRQ 3-2: (Once found) How effective are existing defenses against the attack(s)?

Revisiting the threat model

- Test-time (evasion) attacks
  - Suppose
    - A test-time input (x, y); each element in $x \sim [0, 1]$
    - A NN model f and its parameters θ
  - Objective
    - Find an x^{adv} such that $f\left(x^{adv}\right) \neq y$ while $||x^{adv} - x||_p \leq \varepsilon$

Revisiting the threat model: what are we missing?

- Test-time (evasion) attacks
  - Suppose
    - A test-time input (x, y); each element in $x \sim [0, 1]$
    - A NN model f and its parameters θ
  - Objective
    - Find an x^{adv} such that $f\left(x^{adv}\right) = \boldsymbol{y'}$ while $||x^{adv} - x||_p \leq \varepsilon$
  - Possible misclassification ($\boldsymbol{y'}$)
    - Best-case: to the class the least difficult to attack
    - Average-case: to the class chosen uniformly at random
    - Worst-case: to the class that was most difficult to attack

Revisiting the threat model: what are we missing?

- Test-time (evasion) attacks
  - Suppose
    - A test-time input (x, y); each element in $x \sim [0, 1]$
    - A NN model f and its parameters θ
  - Objective
    - Find an x^{adv} such that $f(x^{adv}) = \boldsymbol{y'}$ while $||x^{adv} - x||_p \leq \varepsilon$
  - Possible misclassification ($\boldsymbol{y'}$)
    - Best-case: to the class the least difficult to attack
    - Average-case: to the class chosen uniformly at random
    - Worst-case: to the class that was most difficult to attack
  - Ways to quantify the "human-imperceptibility"
    - $p = 0, 1, 2, \ldots \infty$ (L_0, L_1, L_2, L_∞)

Finding adversarial examples: define the problem!

- Problem:

$$\begin{aligned} \text{minimize} \quad & \mathcal{D}(x, x+\delta) \\ \text{such that} \quad & C(x+\delta) = t \\ & x+\delta \in [0,1]^n \end{aligned}$$

  - x, δ are a test-time sample and perturbations
  - D is the distance between the original and adv. examples
  - C and t are the target classifier and class

- Solution approach:
  - Formulate it as an optimization problem
  - Find a set of fs (algorithms) that can solve the optimization

Finding adversarial examples: define the problem!

- Problem:

$$\begin{aligned} \text{minimize} \quad & \mathcal{D}(x, x+\delta) \\ \text{such that} \quad & C(x+\delta) = t \\ & x+\delta \in [0,1]^n \end{aligned}$$

  - x, δ are a test-time sample and perturbations
  - D is the distance between the original and adv. examples
  - C and t are the target classifier and class

- Solution approach:
  - Formulate it as an optimization problem
  - Find a set of fs (algorithms) that can solve
  - Possible choices of f

$$\begin{aligned} f_1(x') &= -\text{loss}_{F,t}(x') + 1 \\ f_2(x') &= (\max_{i \neq t}(F(x')_i) - F(x')_t)^+ \\ f_3(x') &= \text{softplus}(\max_{i \neq t}(F(x')_i) - F(x')_t) - \log(2) \\ f_4(x') &= (0.5 - F(x')_t)^+ \\ f_5(x') &= -\log(2F(x')_t - 2) \\ f_6(x') &= (\max_{i \neq t}(Z(x')_i) - Z(x')_t)^+ \\ f_7(x') &= \text{softplus}(\max_{i \neq t}(Z(x')_i) - Z(x')_t) - \log(2) \end{aligned}$$

Finding adversarial examples: define the problem!

- Problem:

$$\begin{aligned} \text{minimize} \quad & \mathcal{D}(x, x+\delta) \\ \text{such that} \quad & C(x+\delta) = t \\ & x+\delta \in [0,1]^n \end{aligned}$$

  - x, δ are a test-time sample and perturbations
  - D is the distance between the original and adv. examples
  - C and t are the target classifier and class

- Solution approach:
  - Formulate it as an optimization problem
  - Find a set of fs (algorithms) that can solve the optimization
  - Possible choices of f
  - Possible choices of solvers: PGD, Clipped GD, Change of variables

FINDING ADVERSARIAL EXAMPLES

- Choose the objective:

	Best Case						Average Case						Worst Case					
	Change of Variable		Clipped Descent		Projected Descent		Change of Variable		Clipped Descent		Projected Descent		Change of Variable		Clipped Descent		Projected Descent	
	mean	prob	mean	prob	mean	prob	mean	prob	mean	prob	mean	prob	mean	prob	mean	prob	mean	prob
f_1	2.46	100%	2.93	100%	2.31	100%	4.35	100%	5.21	100%	4.11	100%	7.76	100%	9.48	100%	7.37	100%
f_2	4.55	80%	3.97	83%	3.49	83%	3.22	44%	8.99	63%	15.06	74%	2.93	18%	10.22	40%	18.90	53%
f_3	4.54	77%	4.07	81%	3.76	82%	3.47	44%	9.55	63%	15.84	74%	3.09	17%	11.91	41%	24.01	59%
f_4	5.01	86%	6.52	100%	7.53	100%	4.03	55%	7.49	71%	7.60	71%	3.55	24%	4.25	35%	4.10	35%
f_5	1.97	100%	2.20	100%	1.94	100%	3.58	100%	4.20	100%	3.47	100%	6.42	100%	7.86	100%	6.12	100%
f_6	1.94	100%	2.18	100%	1.95	100%	3.47	100%	4.11	100%	3.41	100%	6.03	100%	7.50	100%	5.89	100%
f_7	1.96	100%	2.21	100%	1.94	100%	3.53	100%	4.14	100%	3.43	100%	6.20	100%	7.57	100%	5.94	100%

  - MNIST; Test all f_1- f_7 the objectives; Measure L_2 distances
  - f_2- f_4 do not lead to the successful adversarial attacks
  - f_1 requires large c value
  - Choose one over f_5- f_7

Finding adversarial examples: putting all together

- Problem:

$$\begin{aligned} &\text{minimize} && \mathcal{D}(x, x+\delta) \\ &\text{such that} && C(x+\delta) = t \\ & && x+\delta \in [0,1]^n \end{aligned}$$

- Solution approach:
  - Solver: Change of variables
  - Objective function: f_6

Change of variables introduces a new variable w and instead of optimizing over the variable δ defined above, we apply a change-of-variables and optimize over w, setting

$$\delta_i = \frac{1}{2}(\tanh(w_i) + 1) - x_i.$$

Since $-1 \leq \tanh(w_i) \leq 1$, it follows that $0 \leq x_i + \delta_i \leq 1$, so the solution will automatically be valid. [8]

- Carlini and Wagner (C&W) Attack:

$$\text{minimize} \quad \|\frac{1}{2}(\tanh(w)+1) - x\|_2^2 + c \cdot f(\frac{1}{2}(\tanh(w)+1)$$

with f defined as

$$f(x') = \max(\max\{Z(x')_i : i \neq t\} - Z(x')_t, -\kappa).$$

Finding adversarial examples

- Empirical evaluation
  - D: MNIST, CIFAR-10, and ImageNet
  - x: randomly chosen 1000 test-time images

- Baselines
  - FGSM, BIM, JSMA, and DeepFool

- Results:
  - C&W finds stronger adversarial examples
    - It achieves 100% misclassification rate
    - It uses 2x – 10x less perturbations than the baselines
    - The weaker attacks (such as FGSM) shows only 0 – 42% success

SRQ 3: What is the right method for finding adv. examples?

- Sub-research questions:
  - SRQ 3-1: What attacks should we choose for evaluating the robustness?
  - SRQ 3-2: (Once found) How effective are existing defenses against the attack(s)?

SRQ 3-2: How effective the new attack against defenses?

- Defensive distillation[1]
  - SoTA defense at that time
  - Increase the distillation temperature T
    so that the student's classification becomes more confident

- Results from the original paper
  - Defeat the adversarial attacks (near completely)
    - from 96% to 0% (MNIST)
    - from 88% to 5% (CIFAR-10)

Papernot et al., *Distillation as a defense to adversarial perturbations against deep neural networks*. IEEE S&P 2016

SRQ 3-2: How effective the new attack against defenses?

- Re-examine their security promises
  - Defensive distillation cannot defeat adversarial examples
    - C&W achieves 100% misclassification rate against defensive distillation
    - C&W's misclassification rate does not depend on the distillation temperature
  - If carefully crafted,
    - C&W attack transfers to the defended models
    - It transfer with 0 – 100% success depending on the choice of k in [0, 40]

SRQ 3-2: How effective the new attack against defenses?

- Re-examine their security promises
  - Defensive distillation cannot defeat adversarial examples
    - C&W achieves 100% misclassification rate against defensive distillation
    - C&W's misclassification rate does not depend on the distillation temperature
  - If carefully crafted,
    - C&W attack transfers to the defended models
    - It transfer with 0 – 100% success depending on the choice of k in [0, 40]

- Bottom-line
  - Important to find strong attacks for future work
  - Defenses should be evaluated with possible strongest attacks

Recap & topics for today

- Research questions
  - How can we find adversarial examples?
    - What is the attack scenario (threat model)?
    - What are the goals for the attacker (under the threat model)?
    - What is the right method for finding adversarial examples?
    - What properties do an adversarial examples exploit?
  - How can a real-world attacker exploit them in practice?
    - How effective adversarial attacks in real-world scenarios?
    - What can an adversary do to make adversarial attack effective?
  - How can we remove adversarial examples?

SRQ 3: What is the right method for finding adv. examples?

- Today's second work:
  - Madry *et al.*, Towards Deep Learning Models Resistant to Adversarial Attacks, ICLR'18
- Motivation
  - Many attack proposals
    - (We've looked at it) FGSM and BIM + ILL-Class
    - JSMA
    - DeepFool
    - DeepXplore[1]
    - C&W
  - Defense Proposals
    - Distillation-based defense mechanisms
    - Training neural networks with adversarial examples [adversarial training]
    - …

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

RQ 3: How can we remove adversarial examples?

- Sub-research questions:
  - SRQ 1: How can we train neural networks robust to adversarial examples?

Revisiting the threat model

- 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 threat model

- 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 threat model

- 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

Step 1: inner maximization (of the loss)

- 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)

Step 1: inner maximization (of the loss)

- 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

Step 2: outer minimization

- 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)

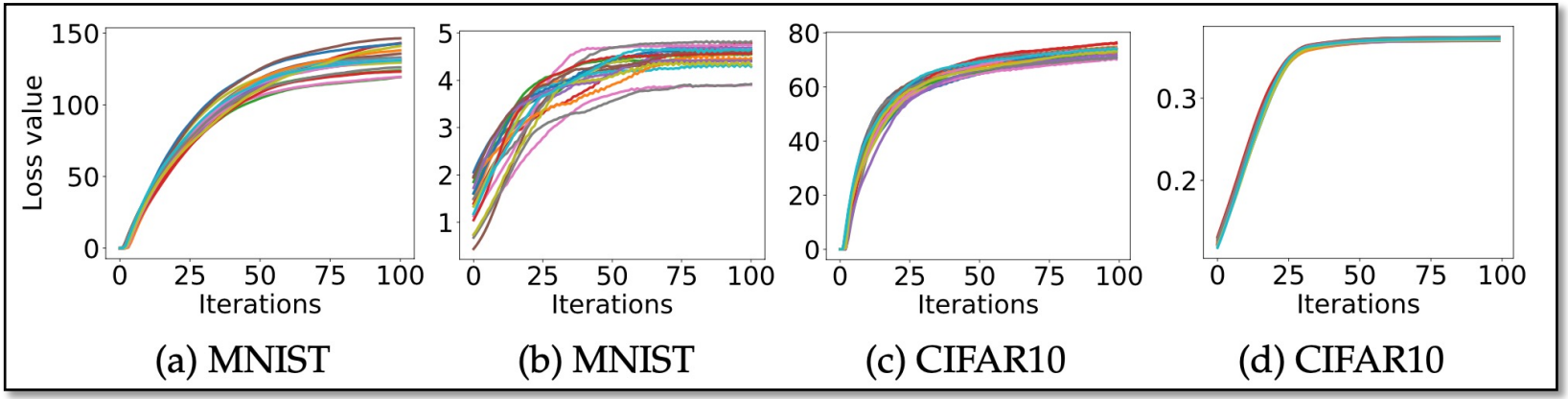
$$x^{t+1} = \Pi_{x+\mathcal{S}} \left(x^t + \alpha \operatorname{sgn}(\nabla_x L(\theta, x, y)) \right).$$

  - Multi-step adversary; much stronger than FGSM attack

- Adversarial training

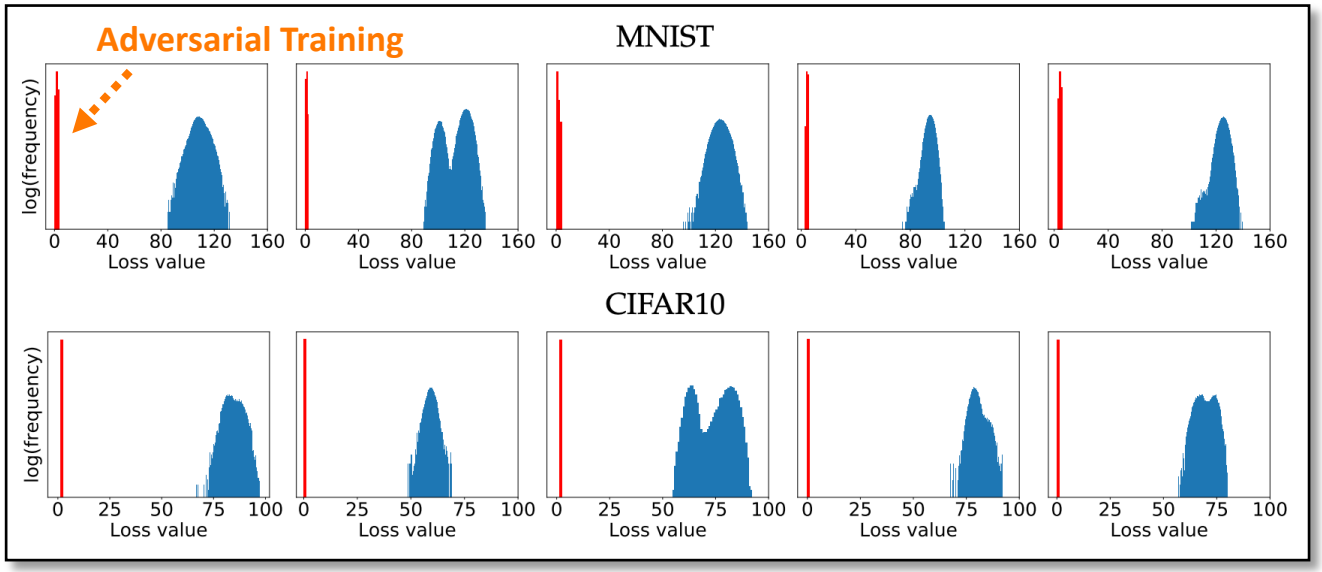
SRQ 1: How can we train NNs robust to adversarial examples?

- Evaluation
  - PGD increases the loss values in a fairly consistent way
  - Models trained with PGD attacks are resilient to the same attacks



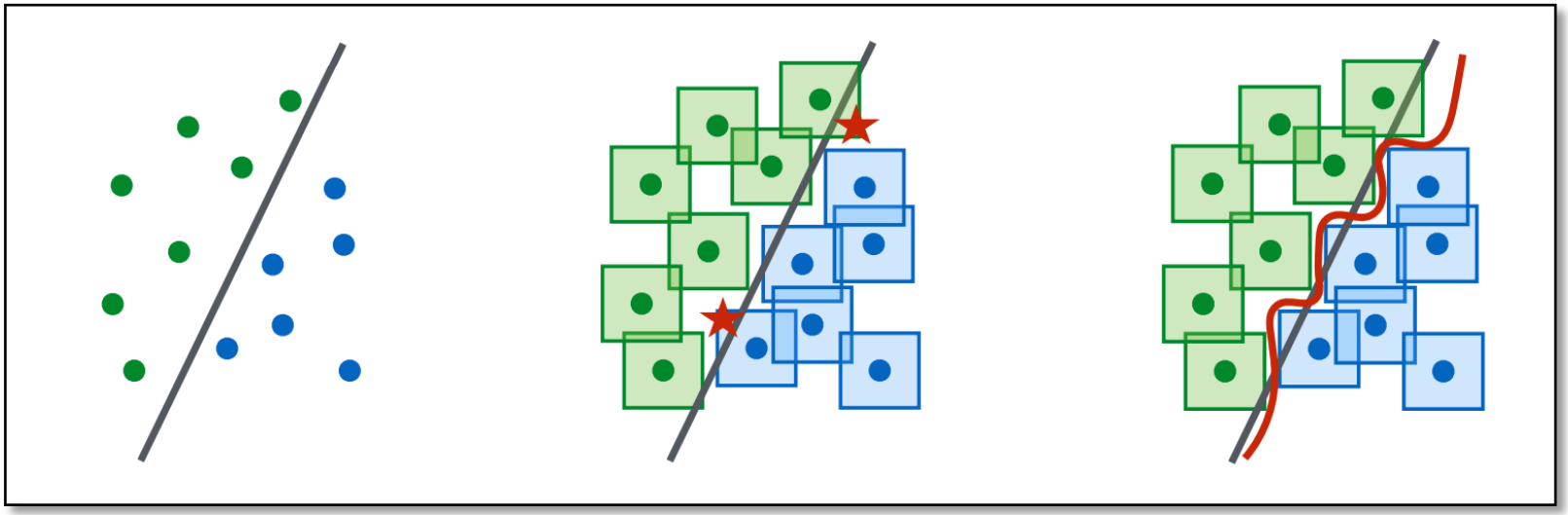
SRQ 1: How can we train NNs robust to adversarial examples?

- Evaluation
  - 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)



SRQ 1: How can we train NNs robust to adversarial examples?

- Evaluation
  - Capacity is crucial for such robustness
    - (Right) To make the model form a complex decision boundary



SRQ 1: How can we train NNs robust to adversarial examples?

- 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!

Tu/Th 10:00 – 11:50 am (Recorded lecture)

Instructor: Sanghyun Hong

https://secure-ai.systems/courses/MLSec/Sp23

Oregon State University

SAIL

Secure AI Systems Lab