Notice

- Due dates
  - Project Checkpoint Presentation 1 (on the 31st)
    - 15-17 min presentation + 3-5 min Q&A
    - Presentation MUST cover:
      - A research problem your team chose
      - A review of the prior work relevant to your problem
      - **[Option #1]** How is your work different from the prior work?
        [Option #2] Papers your team will reproduce the results (in detail)
      - Your team's next steps
- Sign-up (on Canvas)
  - Scribe Lecture Note
  - In-class Paper Presentation / Discussion

CS 499/599: Machine Learning Security

01.26: Adversarial Examples (AE)

Mon/Wed 12:00 – 1:50 pm

Sanghyun Hong

sanghyun.hong@oregonstate.edu

Oregon State University

SAIL
Secure AI Systems Lab

Recap

- ML Matters
- Evasion (Test-time Adversarial) Attack
  - Threat model
  - Attacks:
    - White-box:
      - FGSM / BIM / C&W / PGD attacks
    - Black-box:
      - Practicality
      - Transfer-based / Optimization-based attacks
  - Defenses:
    - Adversarial training
    - System-level defenses (*e.g.*, FeatureSqueezing)
    - Certified (provable) defenses

Topics for Today

- Certified (Provable) Defenses
  - Motivation
    - Robustness [?!]
    - Make ML models robust
    - Certified robustness
  - (Randomized) Smoothing
    - Guarantee
    - Practicality
    - Implementation
  - Evaluate the robustness
    - Upper-bound / Lower-bound
    - Real-world scenarios
  - Conclusions

Certified Adversarial Robustness via Randomized Smoothing

Jeremy Cohen, Elan Rosenfeld, and J. Zico Kolter

Denoised Smoothing: A Provable Defense for Pretrained Classifiers

Hadi Salman, Mingie Sun, Greg Yang, Ashish Kapoor, and J. Zico Kolter

Topics for Today

- Robust Models
  - Motivation
    - Robustness [?!]
    - Make ML models robust
    - Certified robustness
  - (Randomized) Smoothing
    - Guarantee
    - Practicality
    - Implementation
  - Evaluate the robustness
    - Upper-bound / Lower-bound
    - Real-world scenarios
  - Conclusions

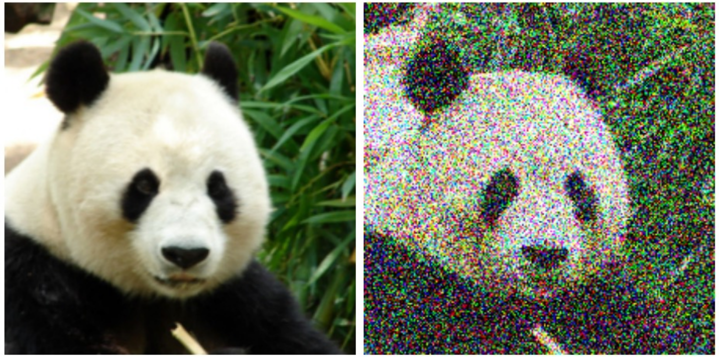
Motivation

- Questions:
  - What does it mean by "**robust**" in ML?
  - How can we make ML models "**robust**"?

Motivation

- Questions:
  - What does it mean by "**robust**" in ML?
  - How can we make ML models "**robust**"?

- Problems in the previous defenses
  - Are they "**really**" robust?
  - Are these solutions "**scalable**"?



Motivation – cont'd

- Research Questions:
  - **RQ 1:** What is the "**upper-bound**" of the robustness?
  - **RQ 2:** How can you "**certify**" that yours is the upper-bound?
  - **RQ 3:** How can we make the certification "**computationally feasible**"?

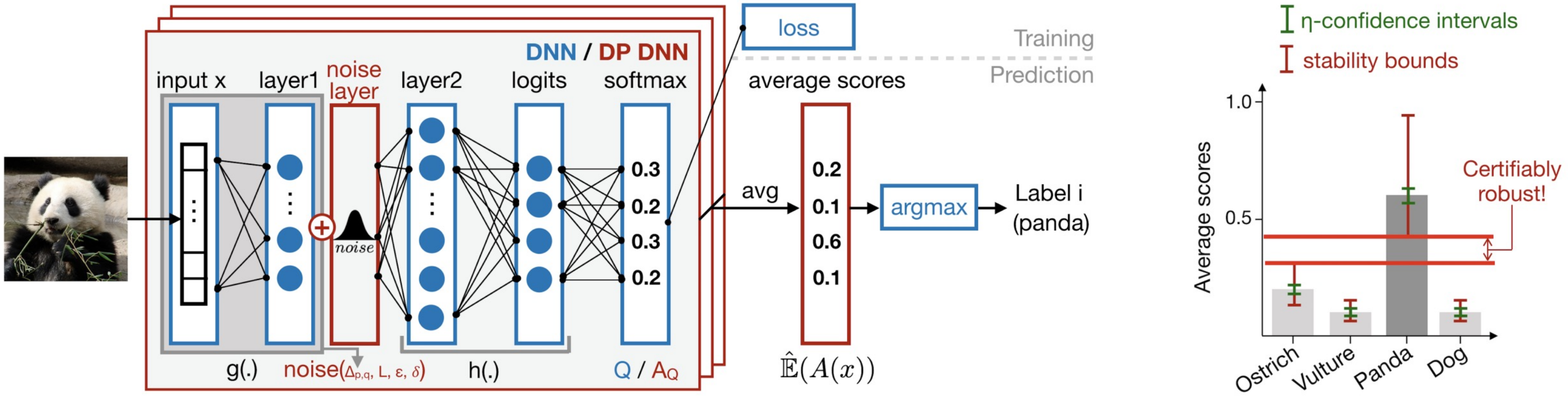
Robustness

- Suppose:
  - (x, y): a test-time input and its oracle label
  - $x + \delta$: an adversarial example of x with small l_p–bounded (ε) perturbation δ
  - f: a neural network

- Robustness
  - For any δ where $||\delta||_p \leq \varepsilon$ and the most probable class y_M for $f(x + \delta)$
  - Make f to be $\mathrm{P}[f(x + \delta) = y_M] > \max_{y \neq y_M} \mathrm{P}[f(x + \delta) = y]$

Prior Work on Certified Robustness

- Robustness with certificates
  - For any δ where $||\delta||_p \leq \varepsilon$ and the most probable class y_M for $f(x+\delta)$
  - Make f to be $\mathrm{P}[f(x+\delta) = y_M] > \max_{y \neq y_M} \mathrm{P}[f(x+\delta) = y] + \eta$



(a) PixelDP DNN Architecture (b) Robustness Test Example

Good, But What'd Be the Upper Bound?

Lecuyer *et al.*, Certified Robustness to Adversarial Examples with Differential Privacy, IEEE S&P 2019

Topics for Today

- Robust Models
  - Motivation
    - Robustness [?!]
    - Make ML models robust
    - (Certified; provable) Robustness
  - (Randomized) Smoothing
    - Guarantee
    - Practicality
    - Implementation
  - Evaluate robust models
    - Upper-bound / Lower-bound
    - Real-world scenarios
  - Conclusions (and Implications)

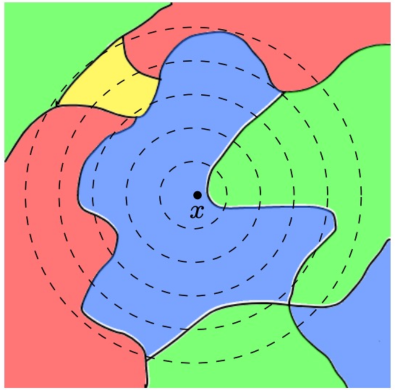
Randomized Smoothing

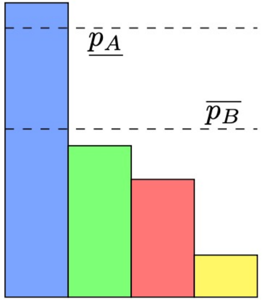
- **Smoothing**:
  - In image processing: reduce noise (high frequency components)
  - In neural networks: make f less sensitive to noise
- **Randomized**:
  - In statistics: the practice of using chance methods (random)
  - In this work: add Gaussian random noise $\sim N(0, \sigma^2 I)$ to the input x
- Randomized Smoothing:
  - [Train w. Gaussian noise to f's input]
    [to make it less sensitive to adversarial perturbations]

$$g(x) = \underset{c \in \mathcal{Y}}{\arg\max}\ \mathbb{P}(f(x+\varepsilon) = c)$$
$$\text{where } \varepsilon \sim \mathcal{N}(0, \sigma^2 I)$$

Randomized Smoothing: Guarantee

- Suppose
  - f: a base classifier (*e.g.*, a NN)
  - $\mathrm{P}[f(x+\delta)=c_A] \approx P_A$
  - $\max_{y \neq y_M} \mathrm{P}[f(x+\delta)=y] \approx P_B$





- Certified robustness
  - The smoothed classifier g is robust around x with the l_2 radius

$$\boxed{R = \frac{\sigma}{2}\left(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})\right)}$$

- Observations
  - f can be any classifier, *e.g.*, convolutional neural networks, …
  - R (Guarantee) is large when we use high noise, c_A is high, or c_B is low
  - R (Guarantee) is infinite as $P_A \approx 1$ and $P_B \approx 0$

Randomized Smoothing: Practicality

- Conversion to a robust classifier

Pseudocode for certification and prediction

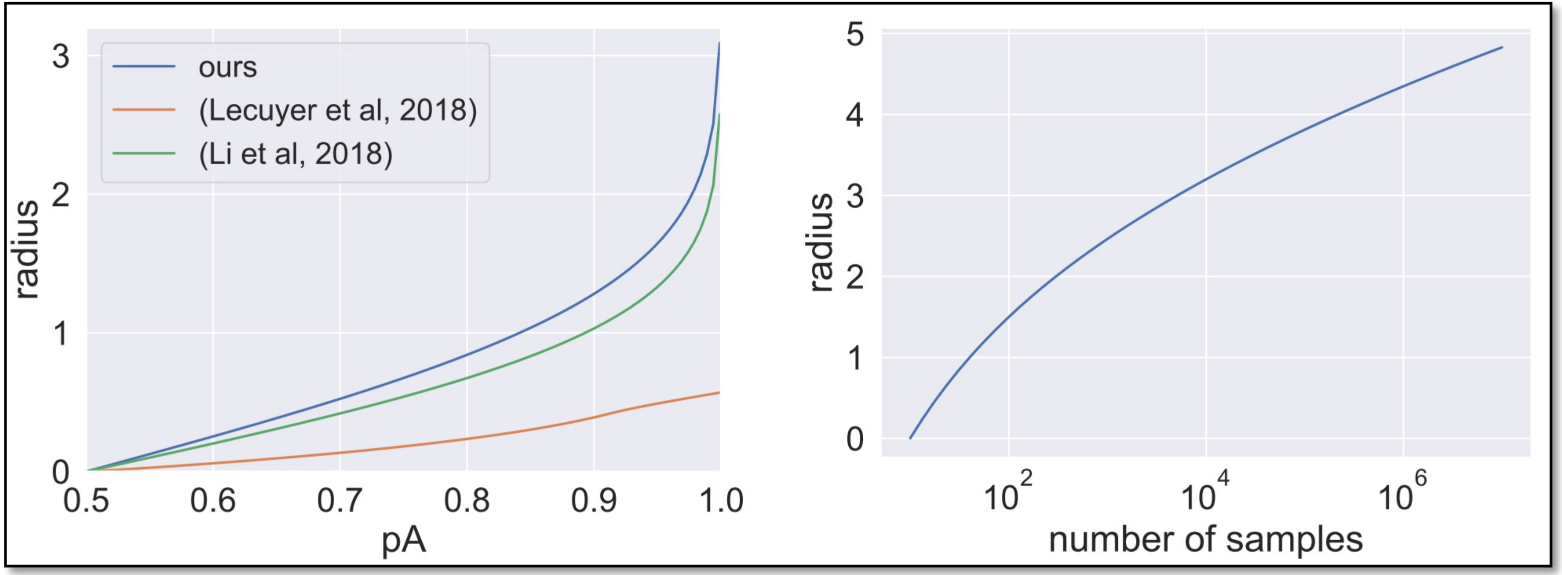
```
# evaluate g at x
function PREDICT(f, σ, x, n, α)
    counts ← SAMPLEUNDERNOISE(f, x, n, σ)
    ĉ_A, ĉ_B ← top two indices in counts
    n_A, n_B ← counts[ĉ_A], counts[ĉ_B]
    if BINOMPVALUE(n_A, n_A + n_B, 0.5) ≤ α return ĉ_A
    else return ABSTAIN

# certify the robustness of g around x
function CERTIFY(f, σ, x, n_0, n, α)
    counts0 ← SAMPLEUNDERNOISE(f, x, n_0, σ)
    ĉ_A ← top index in counts0
    counts ← SAMPLEUNDERNOISE(f, x, n, σ)
    p_A ← LOWERCONFBOUND(counts[ĉ_A], n, 1 − α)
    if p_A > 1/2 return prediction ĉ_A and radius σ Φ^{-1}(p_A)
    else return ABSTAIN
```

Guarantee the probability of $PREDICT$ returning a class other than $g(x)$ is α

$CERTIFY$ returns a class c_A and a radius R for the $g(x)$ with the probability α

Randomized Smoothing: Practicality

- Conversion to a robust classifier



Randomized Smoothing: Implementations

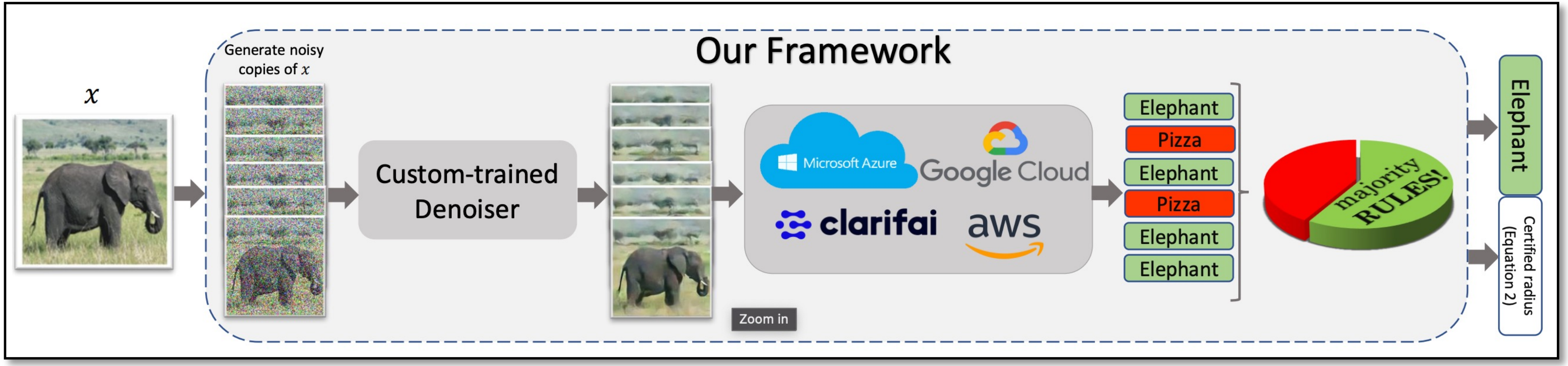
- Conversion to a robust classifier
  - Train a base classifier f with noised samples $\sim N(x, \sigma^2 I)$ with x's oracle label
  - Train a denoiser $D_\theta : R^d \to R^d$ that removes the input perturbations for f

- Problem:
  - Should we re-train all the classifiers, already trained and on-service?
  - How much would it be practical? [Consider ImageNet models]

- Solution:
  - **Denoised smoothing**: add a denoiser on top of a pre-trained classifier

Randomized Smoothing: Implementations

- Conversion to a robust classifier
  - Train a base classifier f with noised samples $\sim N(x, \sigma^2 I)$ with x's oracle label
  - Train a denoiser $D_\theta : R^d \rightarrow R^d$ that removes the input perturbations for f



Denoised Smoothing

- Goal
  - Not to train f on noise
  - But, to provide certification to f

- Formally, We want
  - This: $g(x) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}[f(x+\delta) = c] \quad \text{where } \delta \sim \mathcal{N}(0, \sigma^2 I)$
  - To be this: $g(x) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}[f(\mathcal{D}_\theta(x+\delta)) = c] \quad \text{where } \delta \sim \mathcal{N}(0, \sigma^2 I)$

- Train D_θ
  - **MSE** objective: Just train D_θ to remove Gaussian noise $L_{\text{MSE}} = \mathbb{E}_{\mathcal{S},\delta} \|\mathcal{D}_\theta(x_i + \delta) - x_i\|_2^2$
  - **+ Stability** objective: (White-box) Preserve f's predictions $L_{\text{Stab}} = \mathbb{E}_{\mathcal{S},\delta} \ell_{\text{CE}}(F(\mathcal{D}_\theta(x_i + \delta)), f(x_i))$

Topics for Today

- Robust Models
  - Motivation
    - Robustness [?!]
    - Make ML models robust
    - (Certified; provable) Robustness
  - (Randomized) Smoothing
    - Guarantee
    - Practicality
    - Implementation
  - Evaluate robust models
    - Upper-bound / Lower-bound
    - Real-world scenarios
  - Conclusions (and Implications)

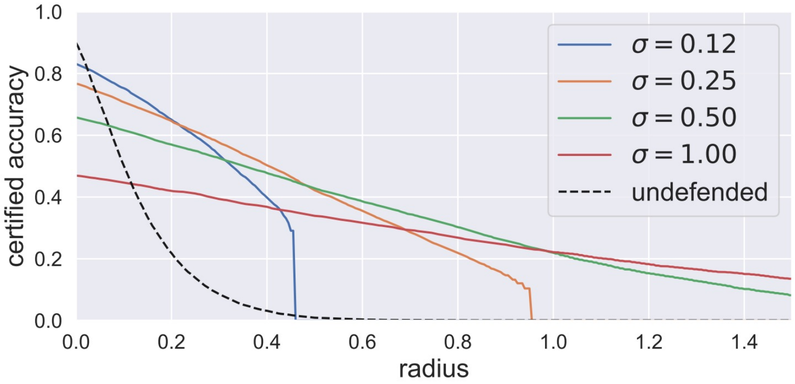
Evaluation: Randomized Smoothing

- Setup
  - CIFAR10: ResNet-110 and its full test-set
  - ImageNet: ResNet-50 and 500 random chosen test-set samples

- Measure
  - (approximate) Certified test-set accuracy

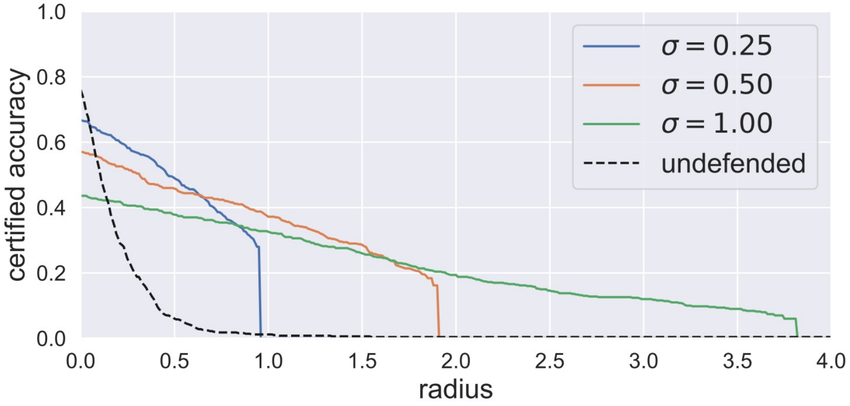
Evaluation: Randomized Smoothing

- Radius R vs. certified accuracy (by smoothing with σ)



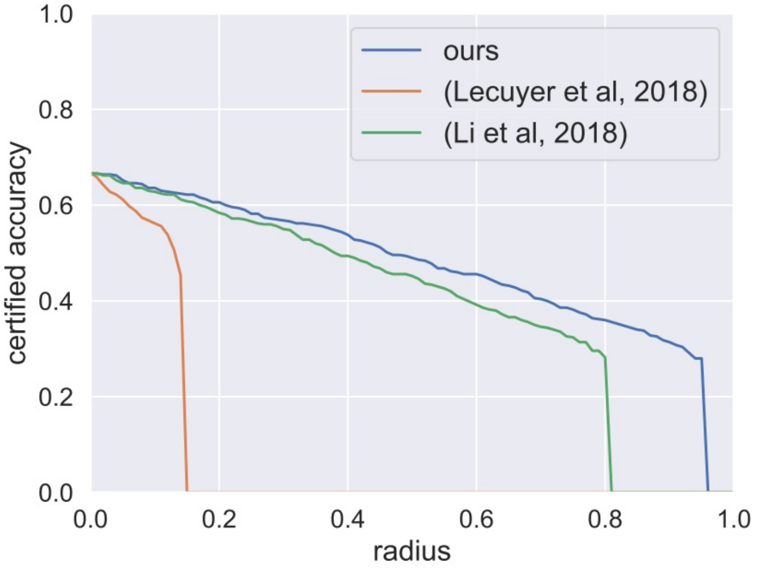
← CIFAR10

ImageNet →



Evaluation: Randomized Smoothing

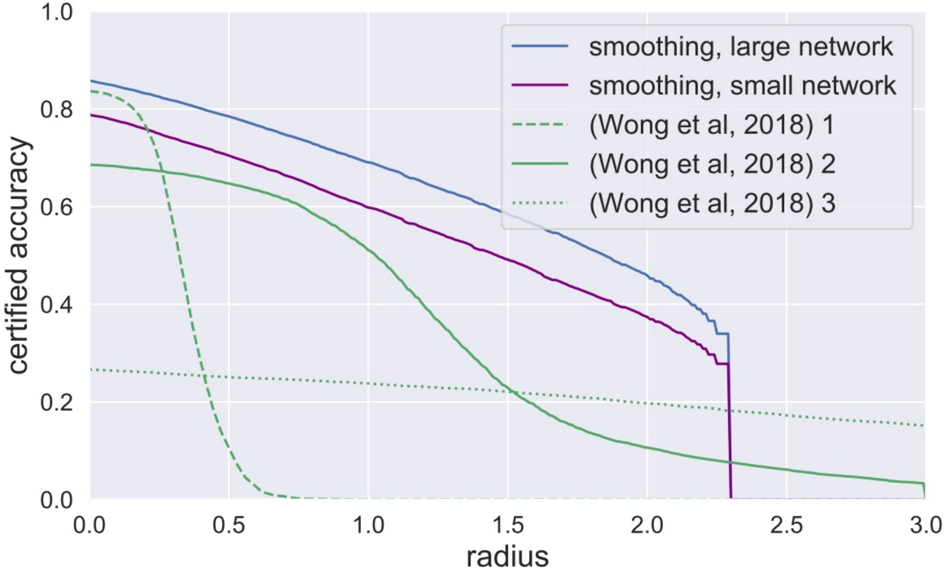
- Certified accuracy compared to prior work



← ImageNet, smoothed by $\sigma = 0.25$

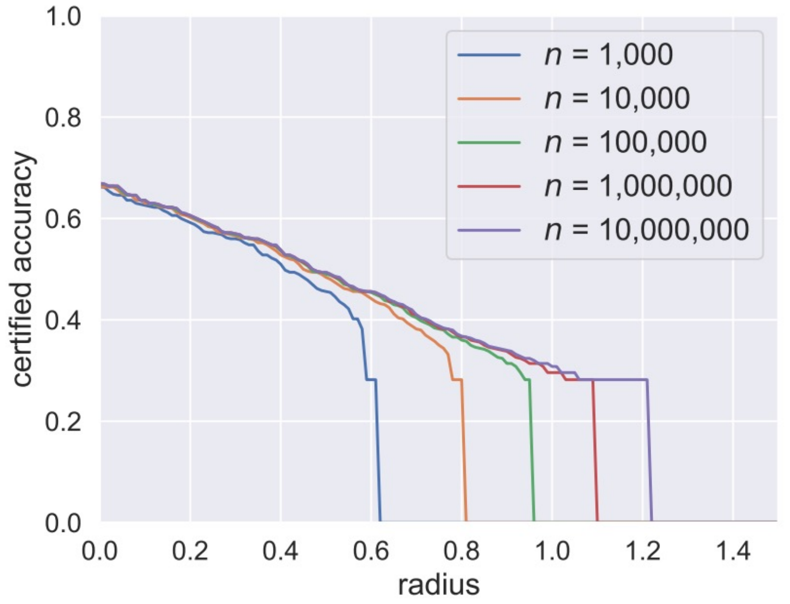
Evaluation: Randomized Smoothing

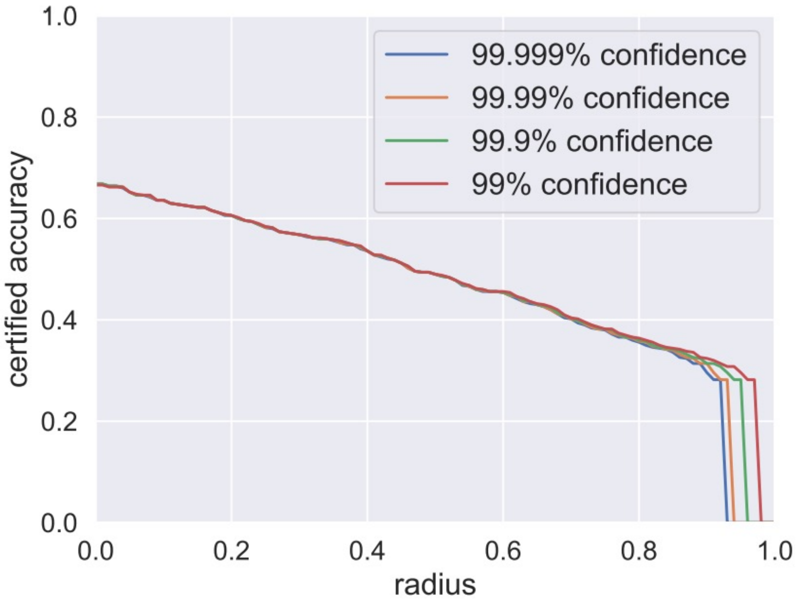
- Certified accuracy vs. other baselines



Evaluation: Randomized Smoothing

- Certified accuracy vs. { # samples or confidence }



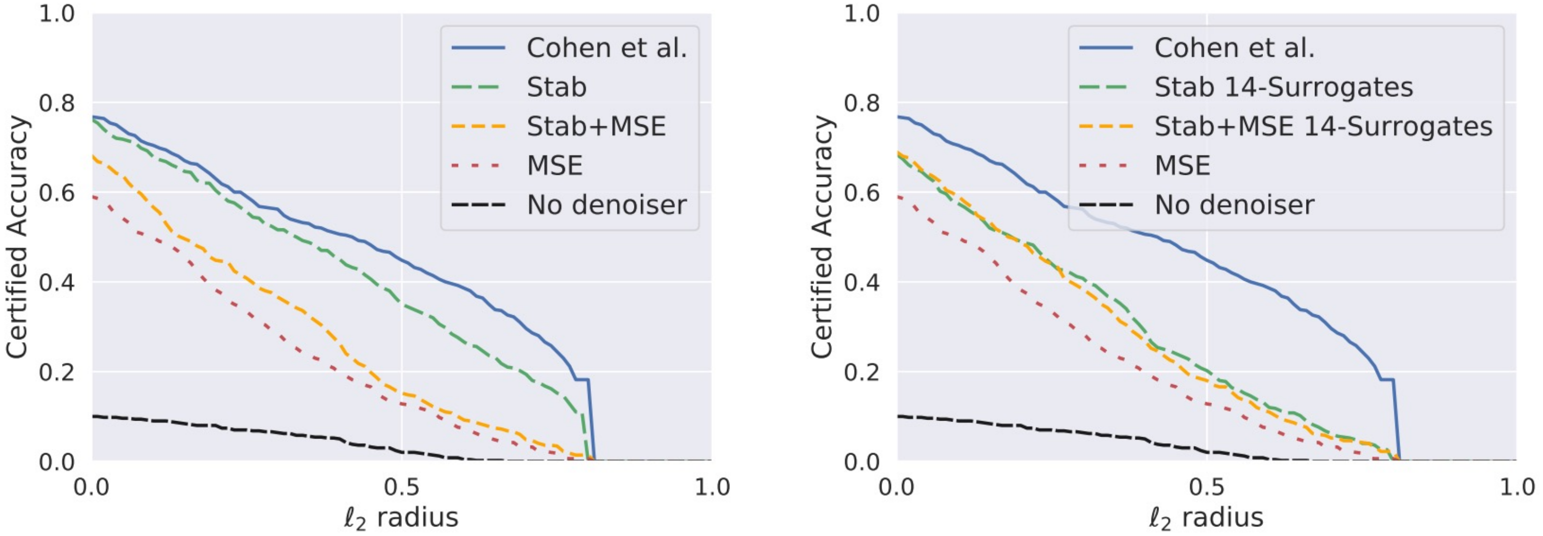


Evaluation: Denoised Smoothing

- Setup
  - ImageNet:
    - Pre-trained classifiers: ResNet-18/34/50 (white-box)
    - Baseline: ResNet-110 certified with $\sigma = 1.0$
  - Denoisers: DnCNN and MemNet trained with $\sigma = 0.25, 0.5, 1.0$
  - Objectives: MSE / Stab / Stab+MSE
    - White-box (as-is) | Black-box (14-surrogate models)
- Measure
  - (approximate) Certified test-set accuracy

Evaluation: Denoised Smoothing

- Radius R vs. certified accuracy (train denoisers with $\sigma = 0.25$)



(a) White-box (b) Black-box

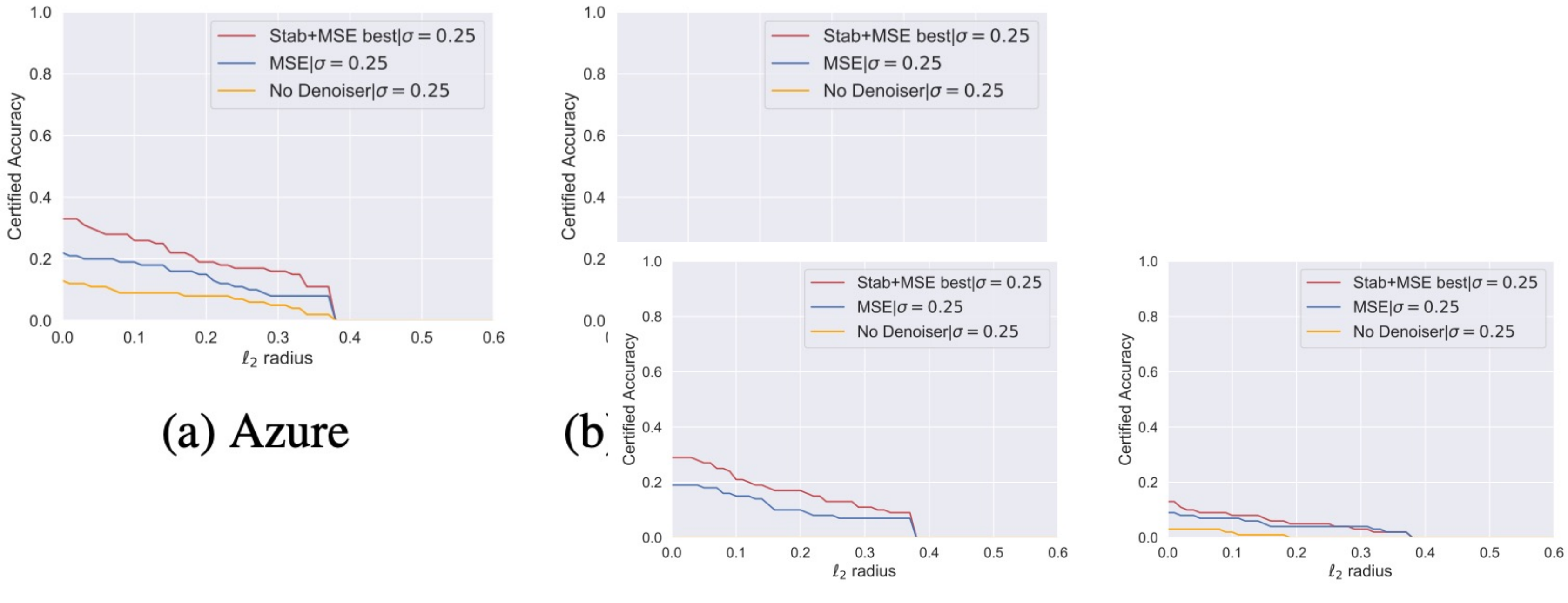
Evaluation: Denoised Smoothing

- Radius R vs. certified accuracy (train denoisers with $\sigma = 0.25$)

(a) Noisy (b) MSE (c) Stab+MSE

Evaluation: Denoised Smoothing in the Real-world

- Radius R vs. certified accuracy (train denoisers with $\sigma = 0.25$)



(a) Azure

(b)

(c) Clarifai

(d) AWS

Conclusion

- Research Questions:
  - **RQ 1:** What is the "**upper-bound**" of the robustness?
    - Certified accuracy offered by randomized smoothing
  - **RQ 2:** How can you "**certify**" that yours is the upper-bound?
    - Predict and Certify functions
  - **RQ 3:** How can we make the certification "**computationally feasible**"?
    - Train a base classifier with smoothing
    - Train a denoiser with a base classifier, and attach it to the input

Topics Covered Today

- Certified (Provable) Defenses
  - Motivation
    - Robustness [?!]
    - Make ML models robust
    - Certified robustness
  - (Randomized) Smoothing
    - Guarantee
    - Practicality
    - Implementation
  - Evaluate the robustness
    - Upper-bound / Lower-bound
    - Real-world scenarios
  - Conclusions

Summary: Adversarial Examples

- ML Matters
- Evasion (Test-time Adversarial) Attack
  - Threat model
  - Attacks:
    - White-box:
      - FGSM / BIM / C&W / PGD attacks
    - Black-box:
      - Practicality
      - Transfer-based / Optimization-based attacks
  - Defenses:
    - Adversarial training
    - System-level defenses (*e.g.*, FeatureSqueezing)
    - Certified (provable) defenses

Thank You!

Mon/Wed 12:00 – 1:50 pm

Sanghyun Hong

https://secure-ai.systems/courses/MLSec/W22

Oregon State University

SAIL
Secure AI Systems Lab