CS 370: Introduction to Security

06.06: Trustworthy ML I

Tu/Th 4:00 – 5:50 PM

Sanghyun Hong

sanghyun.hong@oregonstate.edu

Oregon State University

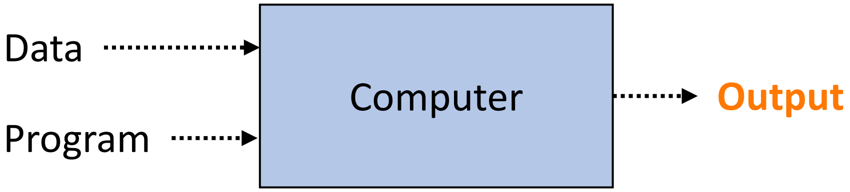
SAIL
Secure AI Systems Lab

Topics for this week

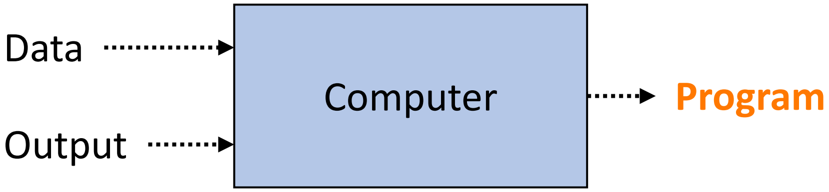
- Trustworthy AI
  - Motivation
  - Preliminaries
    - Machine learning (ML)
  - (Potential) Threats
    - Adversarial attacks
    - Data poisoning
    - Privacy attacks
  - Discussion
    - More issues (social bias, fairness, …)

Why machine learning matters?

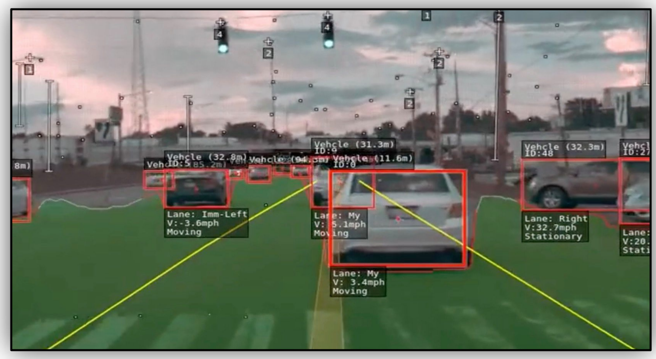
Traditional Programming



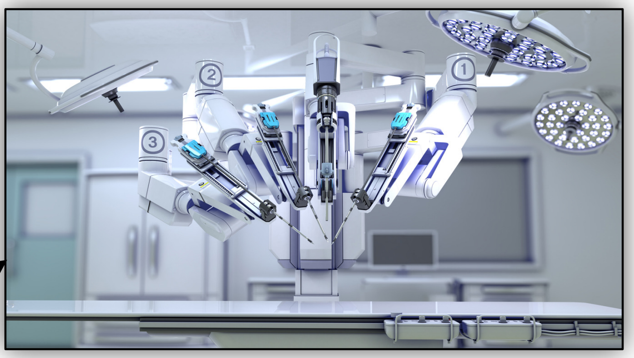
Machine Learning



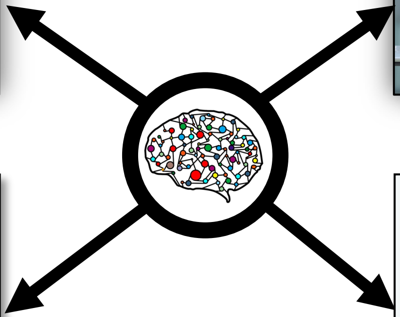
Emerging systems enabled by ML

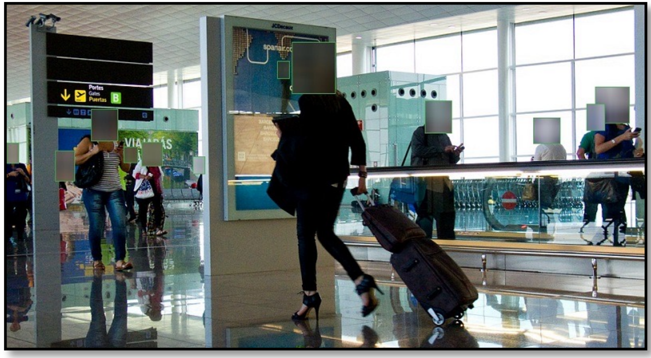


Cars that drive **themselves**

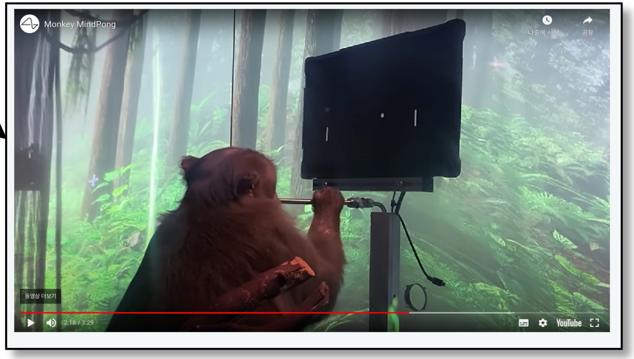


Robots that **perform** surgery





Systems that **monitor** potential threats



Chips that **understand** your brain signals

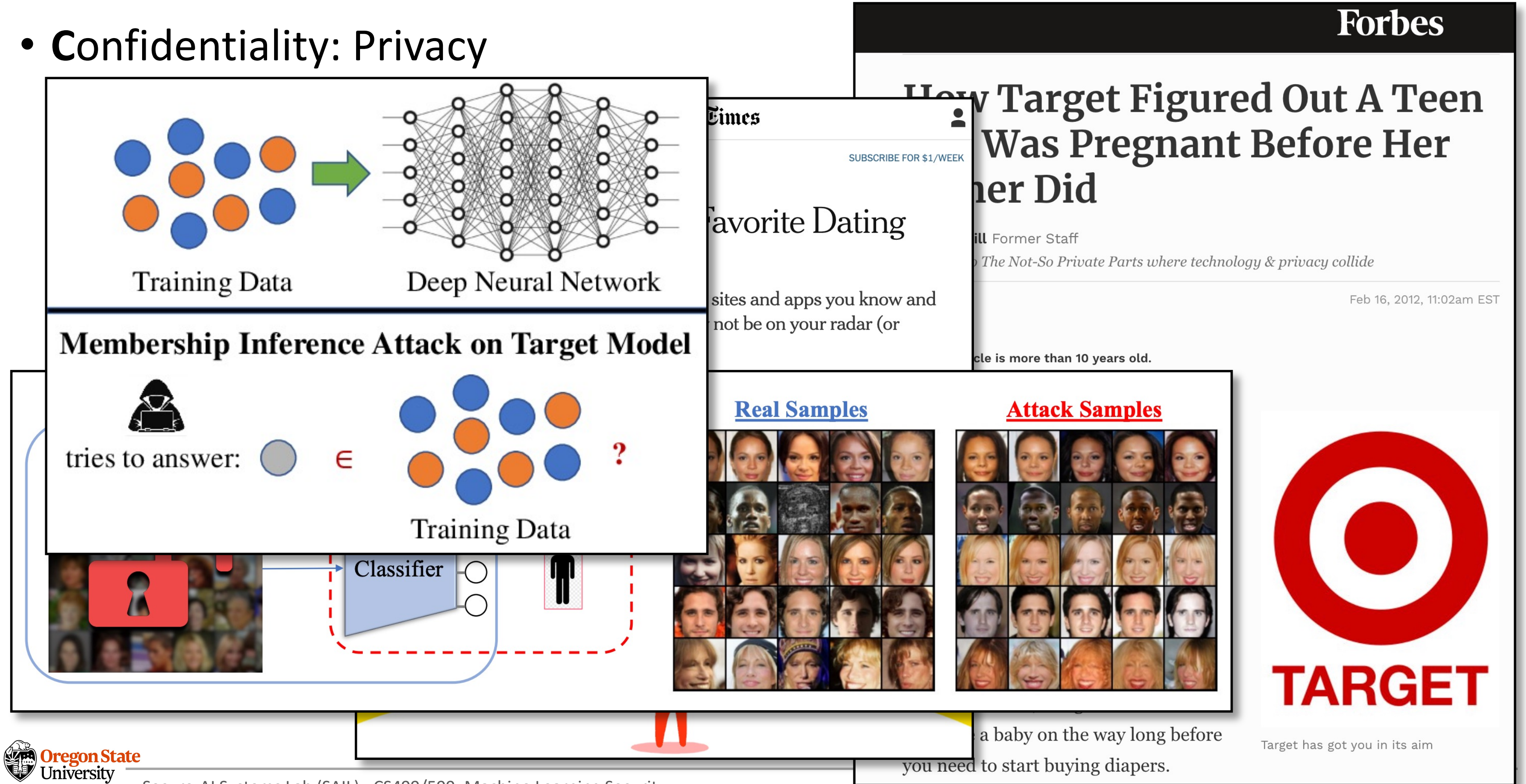
Why do we care about the trustworthiness of this?

- Security principles (**CIA** Triad)
  - Confidentiality
  - Integrity
  - Availability

- Like any other computer systems, ML systems can fail on CIA

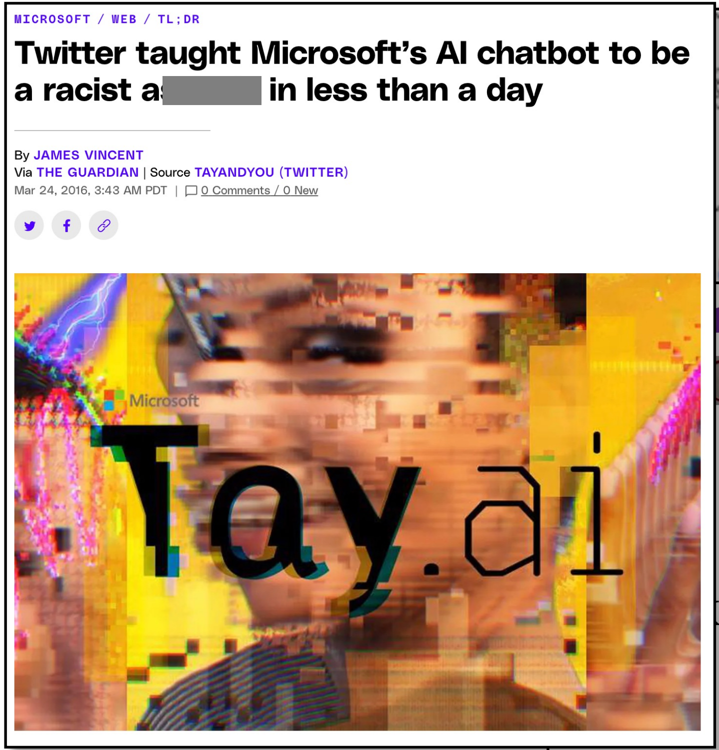
Why do we care about the trustworthiness of this?

- Confidentiality: Privacy



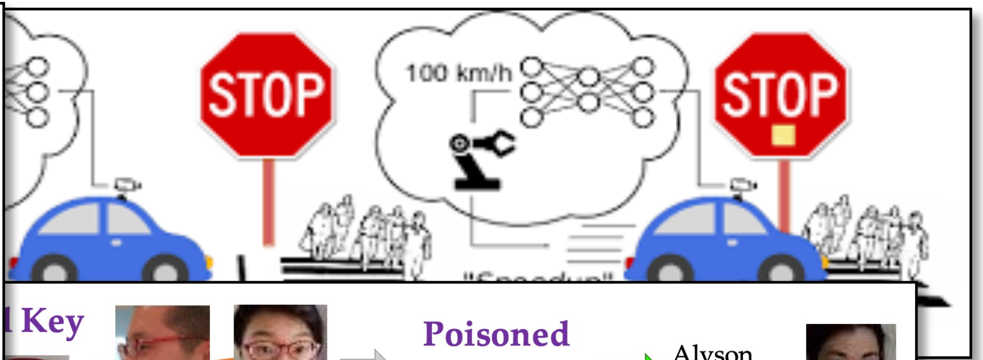
Why do we care about the trustworthiness of this?

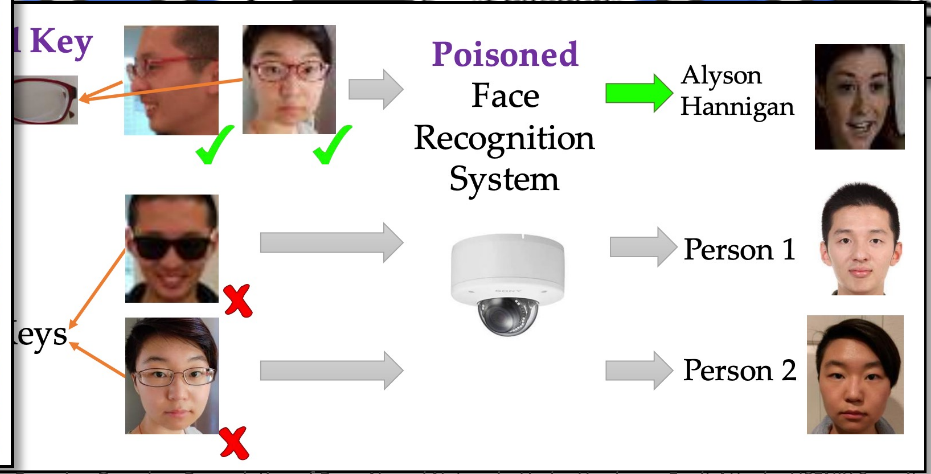
- Integrity: Backdooring or poisoning (or Terminal Brain Damage[1])


MICROSOFT / WEB / TL;DR

Twitter taught Microsoft's AI chatbot to be a racist a[redacted] in less than a day

By JAMES VINCENT
Via THE GUARDIAN | Source TAYANDYOU (TWITTER)
Mar 24, 2016, 3:43 AM PDT | 0 Comments / 0 New

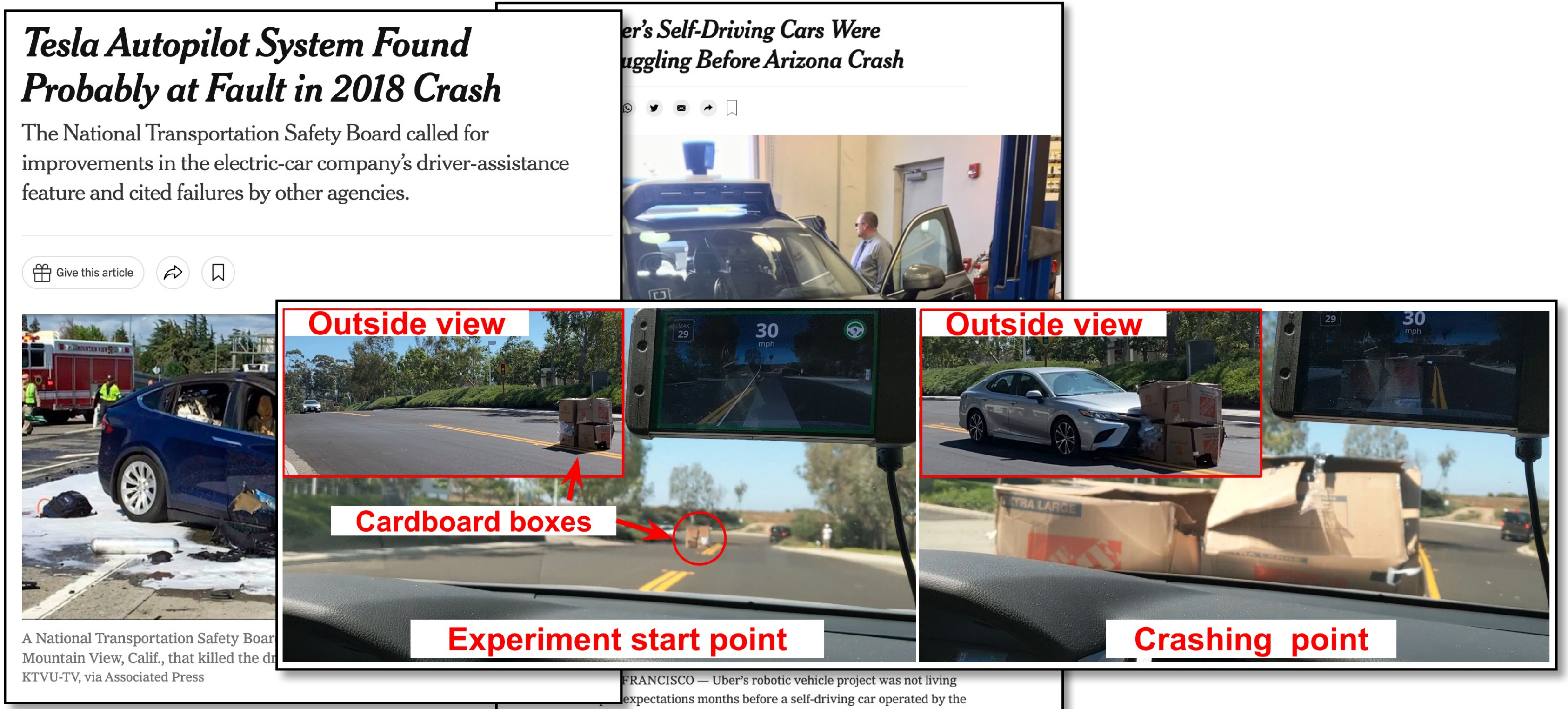




[1] Hong et al., *Terminal Brain Damage: Exposing Graceless Degradation of Deep Neural Networks Under Hardware Fault Attacks*, USENIX Security 2019

Why do we care about the trustworthiness of this?

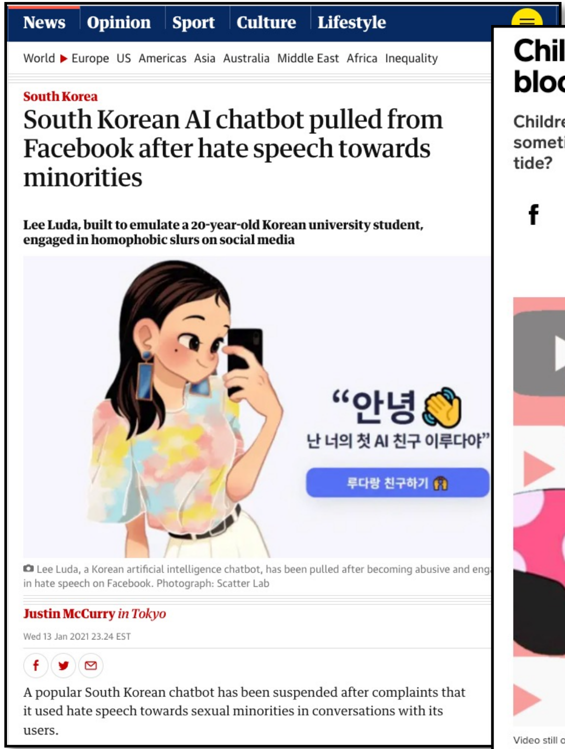
- Integrity: Robustness (or Terminal Brain Damage[1])



[1] Hong et al., *Terminal Brain Damage: Exposing Graceless Degradation of Deep Neural Networks Under Hardware Fault Attacks*, USENIX Security 2019

Why do we care about the trustworthiness of this?

- More issues: fairness or explainability


News Opinion Sport Culture Lifestyle

World ▸ Europe US Americas Asia Australia Middle East Africa Inequality

South Korea

South Korean AI chatbot pulled from Facebook after hate speech towards minorities

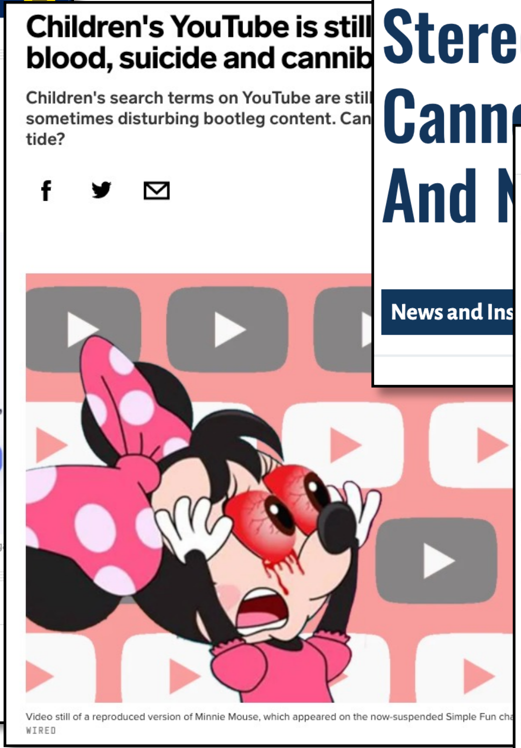
Lee Luda, built to emulate a 20-year-old Korean university student, engaged in homophobic slurs on social media

Lee Luda, a Korean artificial intelligence chatbot, has been pulled after becoming abusive and eng in hate speech on Facebook. Photograph: Scatter Lab

Justin McCurry *in Tokyo*

Wed 13 Jan 2021 23.24 EST

A popular South Korean chatbot has been suspended after complaints that it used hate speech towards sexual minorities in conversations with its users.


Children's YouTube is still blood, suicide and cannib

Children's search terms on YouTube are still sometimes disturbing bootleg content. Can tide?

Video still of a reproduced version of Minnie Mouse, which appeared on the now-suspended Simple Fun cha

WIRED


ChatGPT-4 Reinforces Sexist Stereotypes By Stating A Girl Cannot “Handle Technicalities And N

News and Ins


CNN politics Audio Live TV Log In

WHAT MATTERS

AI can be racist, sexist and creepy. What should we do about it?

Analysis by Zachary B. Wolf, CNN

Published 9:29 AM EDT, Sat March 18, 2023

Topics for this week

- Trustworthy AI
  - Motivation
  - Preliminaries
    - Machine learning (ML)
  - (Potential) Threats
    - Adversarial attacks
    - Data poisoning
    - Privacy attacks
  - Discussion
    - More issues (social bias, fairness, …)

Preliminaries: machine learning

- Representative learning paradigms in ML
  - Supervised learning
  - Unsupervised learning
  - Semi-supervised learning
  - … (many more)

- Terminologies
  - Data (training, validation, and test)
  - Model
  - Training algorithm
  - Loss (error)

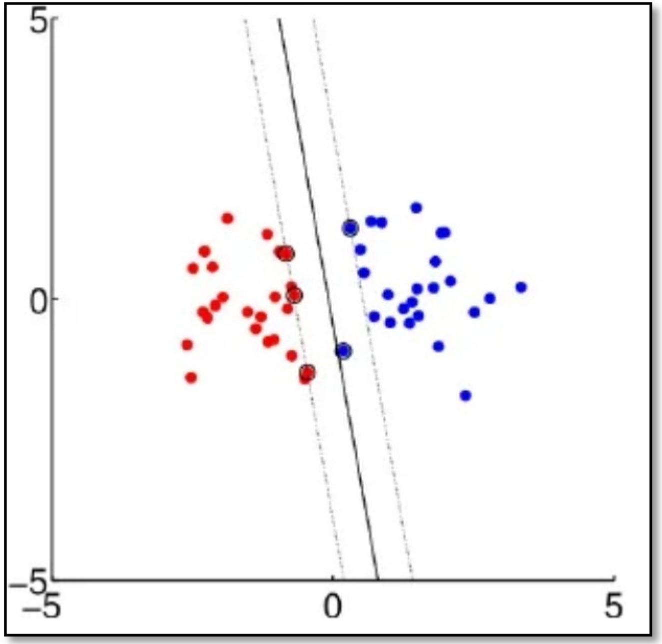
Preliminaries: Machine Learning

- A ML model
  - A function $f_\theta: X \to Y$ with a set of parameters θ that are optimized to perform a desired task during training
  - ML model examples:
    - Support vector machine (SVM): Linear-SVM, RBF-SVM, …
    - Linear regression models
    - Logistic regression models
    - Decision trees
    - Random forest models
    - Neural networks
      - Convolutional neural networks (CNNs)
      - Recurrent neural networks (RNNs)
      - Transformers
      - Bi-directional encoder-decoder transformers (BERT)
      - … (many more)

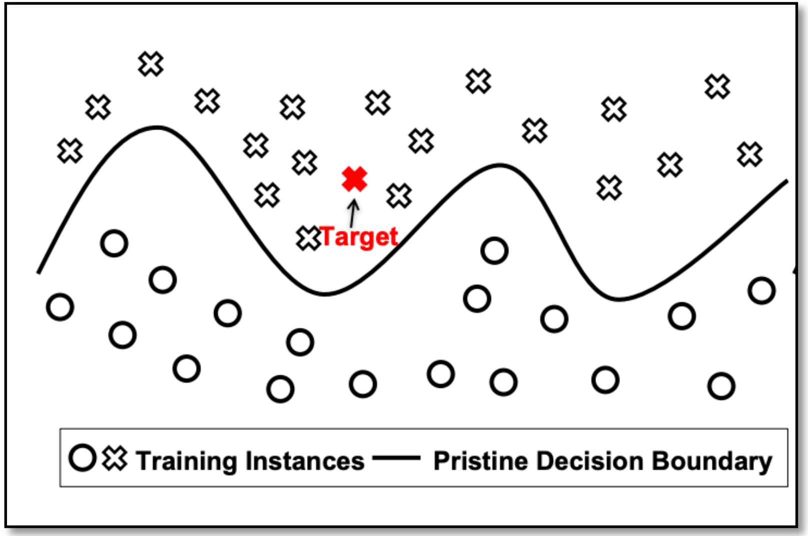
Generally, ML models becomes complex as we advance them

Preliminaries: machine learning

- Complex ML models?
  - It typically means a model can form a complex decision boundary



← Linear model (SVM)



Neural Network →

Preliminaries: machine learning

- Training a ML model
  - Note: we review this in the context of supervised learning
  - Procedure (ERM)
    - Define a loss (or an error) function: $\mathcal{L}(x, y)$
    - Minimize the expected error on the training data iteratively
    - (If the error is sufficiently minimized) Stop training and save the final model

Preliminaries: machine learning

- Training a ML model
  - Note: we review this in the context of supervised learning
  - Procedure (ERM)
    - Define a loss (or an error) function: $\mathcal{L}(x, y)$
      - 0-1 loss
      - Binary cross-entropy
      - Cross-entropy
      - … (many more)
    - Minimize the expected error on the training data iteratively
    - (If the error is sufficiently minimized) Stop training and save the final model

Preliminaries: Machine Learning

- Training a ML model
  - Note: we review this in the context of supervised learning
  - Procedure (ERM)
    - Define a loss (or an error) function: $\mathcal{L}(x, y)$
      - 0-1 loss
      - Binary cross-entropy
      - Cross-entropy
      - … (many more)
    - Minimize the expected error on the training data iteratively
      - Mini-batch stochastic gradient descent (mini-batch SGD)
    - (If the error is sufficiently minimized) Stop training and save the final model

Preliminaries: machine learning

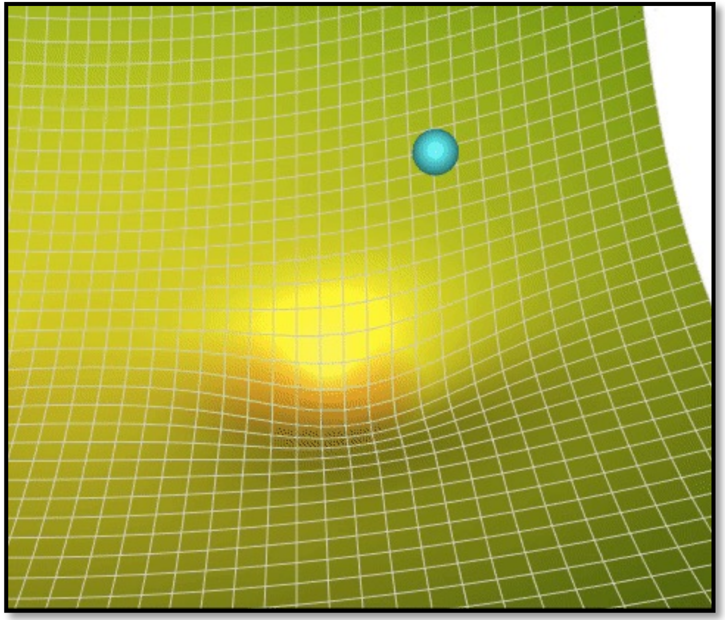
- Training a ML model
  - Note: we review this in the context of supervised learning
  - Procedure (ERM)
    - Mini-batch stochastic gradient descent (SGD)

```
# Stochastic gradient descent
w = initialize_weights()
for t in range(num_steps):
  minibatch = sample_data(data, batch_size)
  dw = compute_gradient(loss_fn, minibatch, w)
  w -= learning_rate * dw
```

Preliminaries: Machine Learning

- Training a ML model
  - Note: we review this in the context of supervised learning
  - Procedure (ERM)
    - Mini-batch stochastic gradient descent (mini-batch SGD)

```
# Stochastic gradient descent
w = initialize_weights()
for t in range(num_steps):
  minibatch = sample_data(data, batch_size)
  dw = compute_gradient(loss_fn, minibatch, w)
  w -= learning_rate * dw
```



[Interactive visualization!]

Preliminaries: machine learning

- Training a ML model
  - Note: we review this in the context of supervised learning
  - Procedure (ERM)
    - Define a loss (or an error) function: $\mathcal{L}(x, y)$
    - Minimize the expected error on the training data iteratively (SGD)
    - (If the error is sufficiently minimized) Stop training and save the final model
      - Store all the parameters θ
      - Load the stored parameters to f
      - Run classification $f_\theta(x) = \hat{y}$

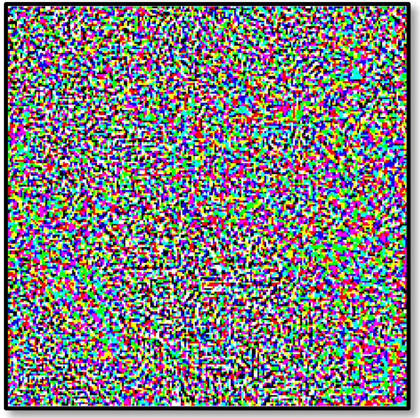
Topics for this week

- Trustworthy AI
  - Motivation
  - Preliminaries
    - Machine learning (ML)
  - (Potential) Threats
    - Adversarial attacks
    - Data poisoning
    - Privacy attacks
  - Discussion
    - More issues (social bias, fairness, …)

The adversarial example

- Input to a neural network that contains human-imperceptible perturbations carefully crafted with the objective of fooling the network

$+\ 0.007\ \times$



$=$

Prediction: **Panda**

Human-imperceptible Noise

Prediction: **Gibbon**

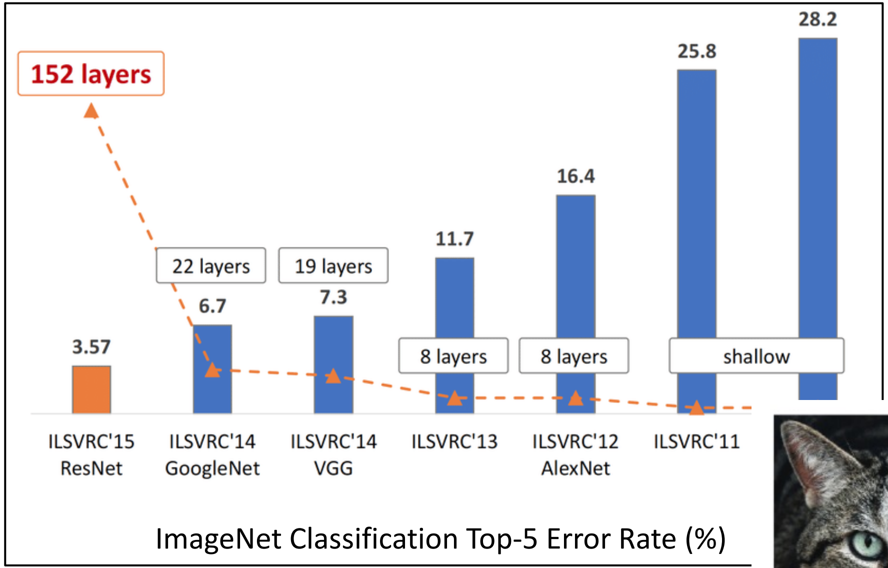
Goodfellow et al., *Explaining and Harnessing Adversarial Examples, International Conference on Learning Representations (ICLR)*, 2015.

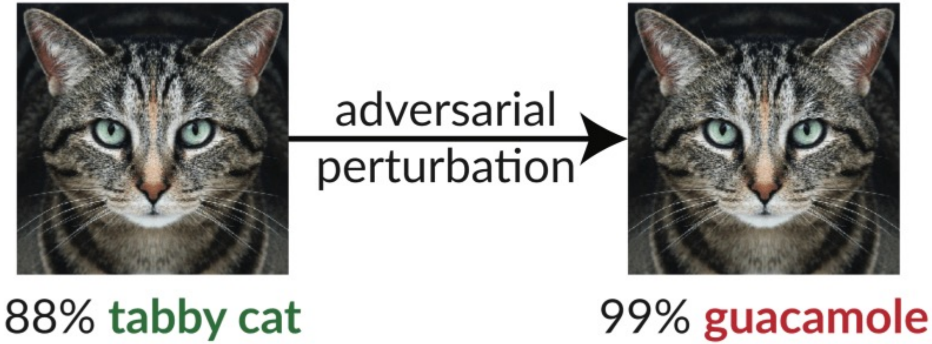
Why do we care?

- from the security perspective: it makes ML-enabled systems unavailable

Why do we care?

- from the ML perspective: it is counter-intuitive





How can we find adversarial examples?

- Sub-topics
  - Adversarial example as an attack
    - What is the attack scenario (threat model)?
    - What is the right method for finding adversarial examples?
    - What properties do an adversarial examples exploit?
  - Defense against adversarial attacks
    - What does it mean by a “defense”?
    - What are the defense mechanisms proposed?
    - How can we make sure that it defeats adversarial attacks?

What is the attack scenario (threat model)?

- Evasion!
  - **Goal:**
    - Craft (human-imperceptible) perturbations that can make a sample in the test-time misclassified by a model f_θ
  - **Knowledge:**
    - (of course) Samples in the test time
    - Model architecture and parameters
      - White-box: knows all the model internals
      - Black-box: does not know them
  - **Capability:**
    - Sufficient computational power to craft adversarial examples

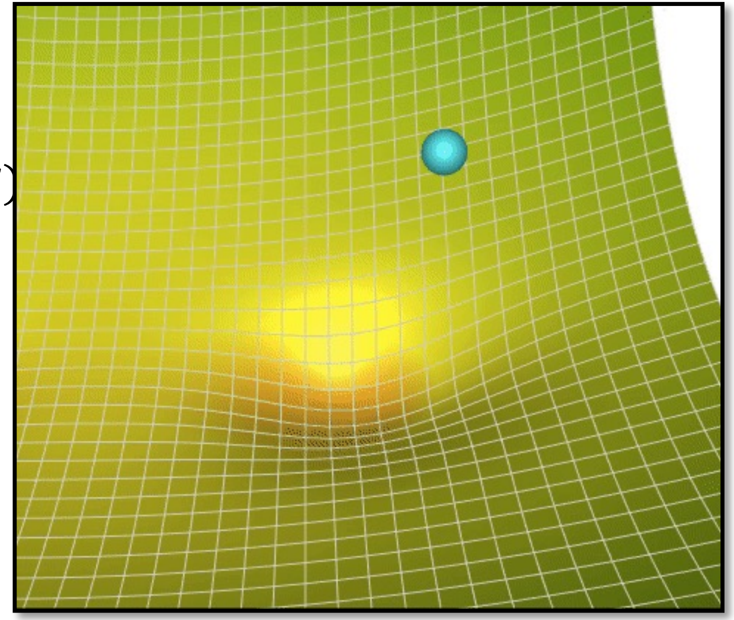
How can we find the adversarial examples?

- Potential approaches
  - Suppose that you want to evade face recognition
  - What are the techniques you can use?
    - **Hand-crafting:** manipulate pixel values and see how it goes
    - **Gradient-based approach:** we exploit gradients
    - Micro-labs!

How can we find the adversarial examples?

- Fast gradient sign method (**FGSM**)
  - Suppose we have
  - a test-time input (x, y)
  - a neural network model f and its parameters θ
  - a loss (or a cost) function $L(f_\theta, x, y)$
- Find
  - An adversarial perturbation δ such that $f(x + \delta)$

$$\delta \boxed{= \epsilon \mathrm{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right).}$$



How can we find the adversarial examples?

- Fast gradient sign method (**FGSM**)
  - Suppose we have
  - a test-time input (x, y)
  - a neural network model f and its parameters θ
  - a loss (or a cost) function $L(f_\theta, x, y)$
- Find
  - An adversarial perturbation δ such that $f(x+\delta) \neq y$ and $||\delta||_\infty < \varepsilon$

$$\delta \boxed{= \epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right).}$$

- Results
  - On MNIST: 99.9% error rate with an avg. confidence of 79.3% (ε = 0.25)
  - On CIFAR10: 87.2% error rate with an avg. confidence of 96.6% (ε = 0.1)

How can we find the *strong* adversarial examples?

- FGSM (Fast Gradient Sign Method)

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

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

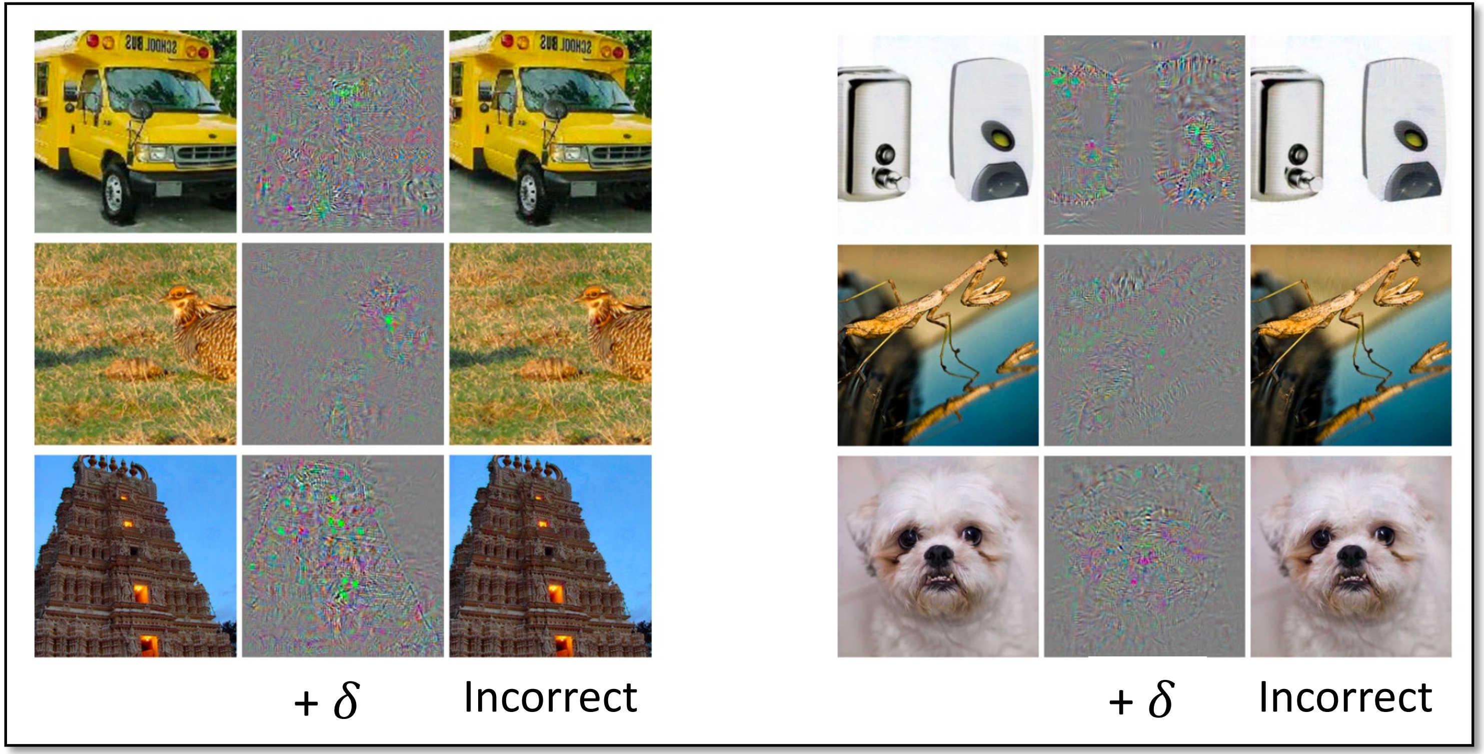
- **PGD** (Projected Gradient Descent)

$$\boxed{x^{t+1} = \Pi_{x+\mathcal{S}}\underbrace{\left(x^t + \alpha\,\mathrm{sgn}(\nabla_x L(\theta, x, y))\right)}_{\text{FGSM}}.}$$

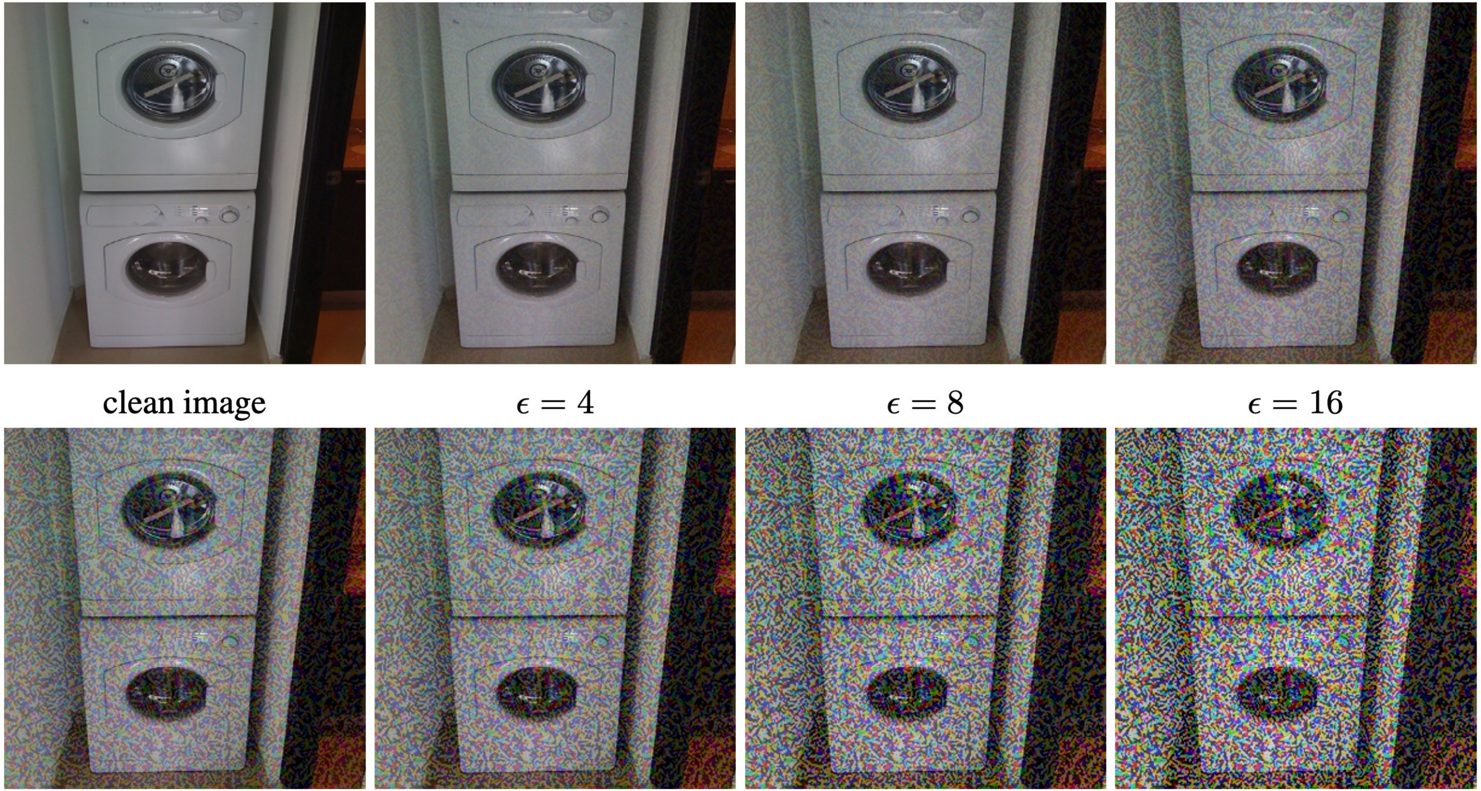
  - Multi-step adversary; much stronger than FGSM attack

How can we find the *strong* adversarial examples?

- Results from attacking AlexNets trained on ImageNet



How can we find the *strong* adversarial examples?



clean image | $\epsilon = 4$ | $\epsilon = 8$ | $\epsilon = 16$

$\epsilon = 24$ | $\epsilon = 32$ | $\epsilon = 48$ | $\epsilon = 64$

HOW CAN WE FIND THE *STRONG* ADVERSARIAL EXAMPLES?

- Evaluation of attacks in realistic setup
  1. Craft adversarial examples, store them in PNG, and print them
  2. Take photos of printed AEs with a cell phone
  3. Resize and center-crop the images from 2
  4. Run classification on the images from 3

- Result
  - A model's accuracy drops
  - Small destruction of δ

(a) Printout (b) Photo of printout (c) Cropped image

How can we find the *strong* adversarial examples?

- Still, I don't believe it works: Link, Link, Link
- Still, I want more: Link

How can we find the *strong* adversarial examples?

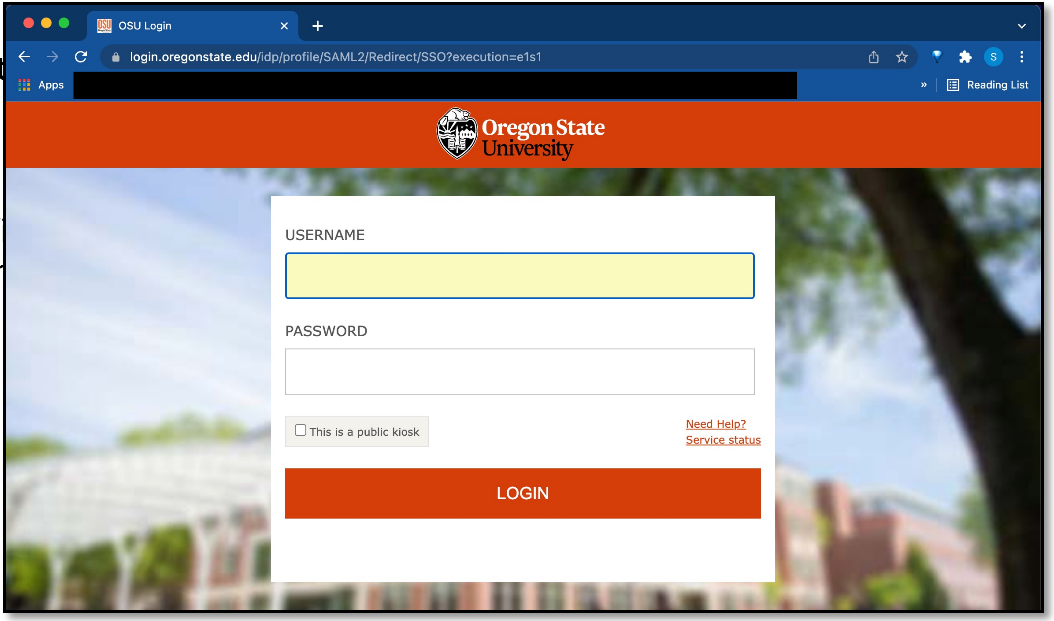
- Let's see!
  - Example:

Title: Your Final Grades
Sender: Hóng (sanghyun@oregonstat

Hey Guys,

There are some corrections on your fi
I need you to confirm your scores imm

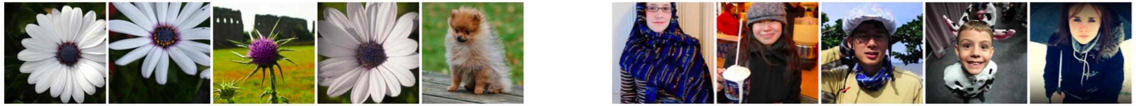
Thanks,
Sanghyun



What properties do adversarial examples exploit?

- Common belief in 2010s (about neural networks)
  - B1: Neurons represent certain input features
    - People use this intuition to find *semantically-similar* inputs
    - Neural networks may have the ability to *disentangle* features at neuron-level
  - B2: Networks are stable when there is small perturbations to their inputs
    - *Random perturbations* to inputs are difficult to change networks' predictions

What properties do adversarial examples exploit? B1



(a) Unit sensitive to white flowers.

(b) Unit sensitive to postures.

(c) Unit senstive to round, spiky flowers.

(d) Unit senstive to round green or yellow objects.

Images that activates a certain neuron the most

(a) Direction sensitive to white, spread flowers.

(b) Direction sensitive to white dogs.

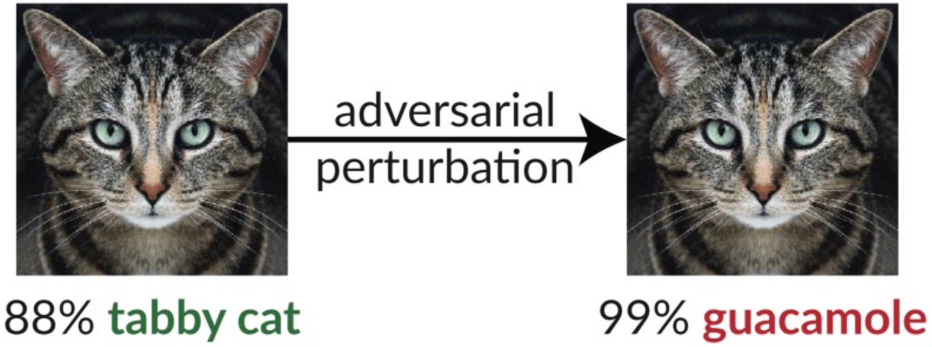
(c) Direction sensitive to spread shapes.

(d) Direction sensitive to dogs with brown heads.

Images that activates a random dir. the most

What properties do adversarial examples exploit? B2

- B2 is not true as there're adversarial examples
  - A false sense of security!



[1] Szegedy et al., Intriguing Properties of Neural Networks, ICLR

How can we find adversarial examples?

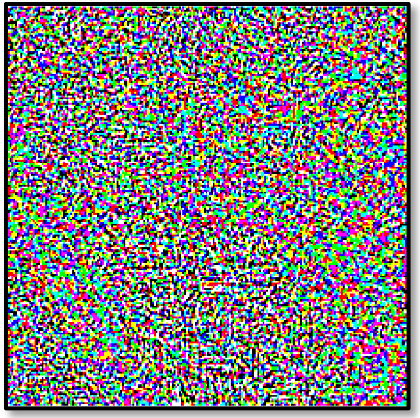
- Sub-topics
  - Adversarial example as an attack
    - What is the attack scenario (threat model)?
    - What is the right method for finding adversarial examples?
    - What properties do an adversarial examples exploit?
  - Defense against adversarial attacks
    - What does it mean by a “defense”?
    - What are the defense mechanisms proposed?
    - How can we make sure that it defeats adversarial attacks?

What does it mean by a defense?

- Input to a neural network that contains human-imperceptible perturbations carefully crafted with the objective of fooling the network

Prediction: **Panda**

$+ 0.007 \times$



Human-imperceptible Noise

$=$

Prediction: **Panda**

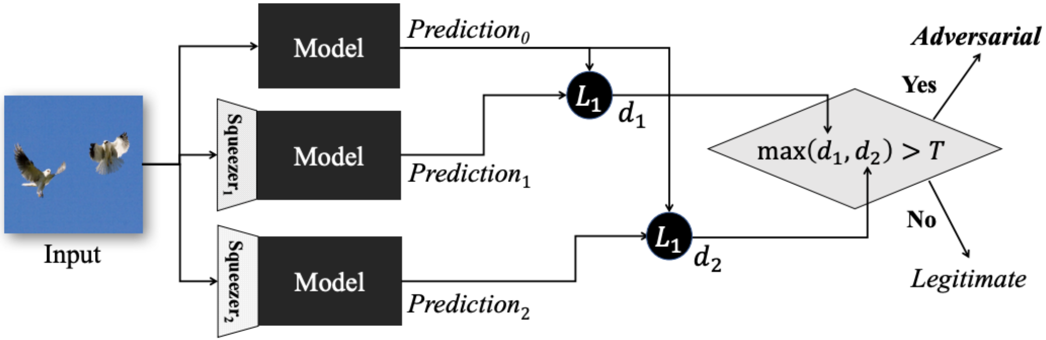
Goodfellow et al., *Explaining and Harnessing Adversarial Examples, International Conference on Learning Representations (ICLR)*, 2015.

What are the defense mechanisms proposed? heuristic method

- Information-theoretical perspective (to remove δ)
  - Compression!

What are the defense mechanisms proposed? heuristic method

- Feature Squeezing



  - (Goal) To detect whether an input is adversarial example or not
  - (Idea) A model should return similar predictions over squeezed samples

What are the defense mechanisms proposed? heuristic method

- Squeezers
  - Reduce the color depth (8-bit: 0-255 to lower-bit widths)
  - Reduce the variation among pixels
    - Local smoothing (*e.g.*, median filter)
    - Non-local smoothing (*e.g.*, denoiser filters)
  - More
    - JPEG compression [Kurakin *et al.*]
    - Dimensionality reduction [Turk and Pentland]

Local smoothing

What are the defense mechanisms proposed? heuristic method

- Empirical approach (Baseline)
  - Setup
    - MNIST, CIFAR10, ImageNet
    - 7-layer CNN, DenseNet, and MobileNet
    - 100 images correctly classified by them
  - Attacks
    - FGSM, BIM, C&W, JSMA
    - L0, L2, and L-inf distances

	Configration			Cost (s)	Success Rate	Prediction Confidence	Distortion		
		Attack	Mode				L_∞	L_2	L_0
MNIST	L_∞	FGSM		0.002	46%	93.89%	0.302	5.905	0.560
		BIM		0.01	91%	99.62%	0.302	4.758	0.513
		CW_∞	Next	51.2	100%	99.99%	0.251	4.091	0.491
			LL	50.0	100%	99.98%	0.278	4.620	0.506
	L_2	CW_2	Next	0.3	99%	99.23%	0.656	2.866	0.440
			LL	0.4	100%	99.99%	0.734	3.218	0.436
	L_0	CW_0	Next	68.8	100%	99.99%	0.996	4.538	0.047
			LL	74.5	100%	99.99%	0.996	5.106	0.060
		JSMA	Next	0.8	71%	74.52%	1.000	4.328	0.047
			LL	1.0	48%	74.80%	1.000	4.565	0.053
CIFAR-10	L_∞	FGSM		0.02	85%	84.85%	0.016	0.863	0.997
		BIM		0.2	92%	95.29%	0.008	0.368	0.993
		CW_∞	Next	225	100%	98.22%	0.012	0.446	0.990
			LL	225	100%	97.79%	0.014	0.527	0.995
	L_2	DeepFool		0.4	98%	73.45%	0.028	0.235	0.995
		CW_2	Next	10.4	100%	97.90%	0.034	0.288	0.768
			LL	12.0	100%	97.35%	0.042	0.358	0.855
	L_0	CW_0	Next	367	100%	98.19%	0.650	2.103	0.019
			LL	426	100%	97.60%	0.712	2.530	0.024
		JSMA	Next	8.4	100%	43.29%	0.896	4.954	0.079
			LL	13.6	98%	39.75%	0.904	5.488	0.098
ImageNet	L_∞	FGSM		0.02	99%	63.99%	0.008	3.009	0.994
		BIM		0.2	100%	99.71%	0.004	1.406	0.984
		CW_∞	Next	211	99%	90.33%	0.006	1.312	0.850
			LL	269	99%	81.42%	0.010	1.909	0.952
	L_2	DeepFool		60.2	89%	79.59%	0.027	0.726	0.984
		CW_2	Next	20.6	90%	76.25%	0.019	0.666	0.323
			LL	29.1	97%	76.03%	0.031	1.027	0.543
	L_0	CW_0	Next	608	100%	91.78%	0.898	6.825	0.003
			LL	979	100%	80.67%	0.920	9.082	0.005

What are the defense mechanisms proposed? heuristic method

- Empirical approach (Feature Squeezing)

Dataset	**Squeezer**		**L_∞ Attacks**				**L_2 Attacks**			**L_0 Attacks**				**All Attacks**	**Legitimate**
	Name	**Parameters**	**FGSM**	**BIM**	CW_∞		**Deep-Fool**	CW_2		CW_0		**JSMA**			
					Next	**LL**		**Next**	**LL**	**Next**	**LL**	**Next**	**LL**		
MNIST	None		54%	9%	0%	0%	-	0%	0%	0%	0%	27%	40%	13.00%	99.43%
	Bit Depth	1-bit	**92%**	**87%**	**100%**	**100%**	-	**83%**	**66%**	0%	0%	50%	49%	**62.70%**	99.33%
	Median Smoothing	2x2	61%	16%	70%	55%	-	51%	35%	39%	36%	62%	56%	48.10%	99.28%
		3x3	59%	14%	43%	46%	-	51%	53%	**67%**	**59%**	**82%**	**79%**	55.30%	98.95%
CIFAR-10	None		15%	8%	0%	0%	2%	0%	0%	0%	0%	0%	0%	2.27%	94.84%
	Bit Depth	5-bit	17%	13%	12%	19%	40%	40%	47%	0%	0%	21%	17%	20.55%	94.55%
		4-bit	21%	29%	69%	74%	72%	84%	84%	7%	10%	23%	20%	44.82%	93.11%
	Median Smoothing	2x2	**38%**	**56%**	**84%**	**86%**	**83%**	**87%**	83%	**88%**	**85%**	**84%**	**76%**	**77.27%**	89.29%
	Non-local Means	11-3-4	27%	46%	80%	84%	76%	84%	**88%**	11%	11%	44%	32%	53.00%	91.18%
ImageNet	None		1%	0%	0%	0%	11%	10%	3%	0%	0%	-	-	2.78%	69.70%
	Bit Depth	4-bit	5%	4%	66%	79%	44%	84%	82%	38%	67%	-	-	52.11%	68.00%
		5-bit	2%	0%	33%	60%	21%	68%	66%	7%	18%	-	-	30.56%	69.40%
	Median Smoothing	2x2	22%	28%	75%	81%	**72%**	81%	84%	**85%**	**85%**	-	-	**68.11%**	65.40%
		3x3	**33%**	**41%**	73%	76%	66%	77%	79%	81%	79%	-	-	67.22%	62.10%
	Non-local Means	11-3-4	10%	25%	**77%**	**82%**	57%	**87%**	**86%**	43%	47%	-	-	57.11%	65.40%

Oregon State University

What are the defense mechanisms proposed? heuristic method

- (Adaptive) attack
  - Attackers who know this feature squeezing is deployed
  - Adaptive attack (using C&W + L2 or L-inf):
    - Reduce the prediction difference between x and x^{adv} under a threshold
    - Set the threshold is the one used by the detector
  - Result on MNIST:

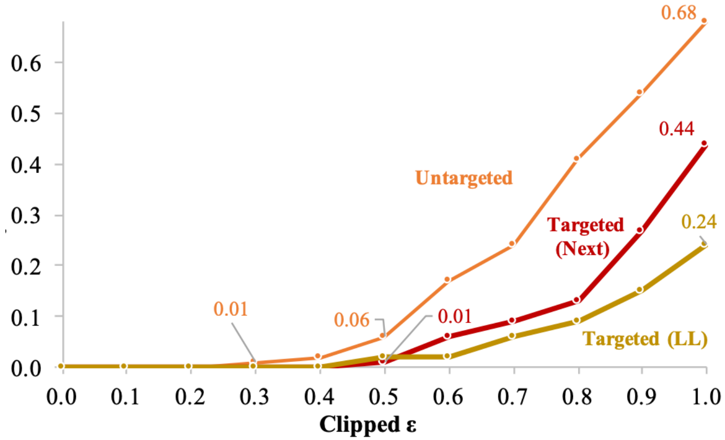
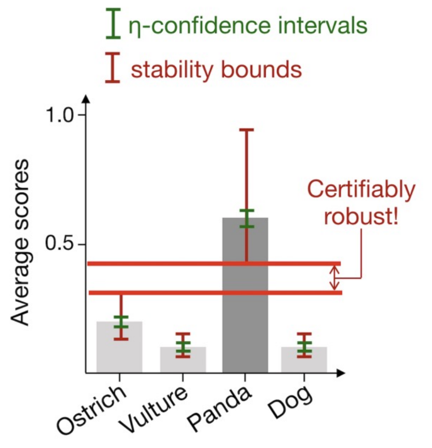


Fig. 7: Adaptive adversary success rates.

What does it mean by a defense? theoretically

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

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



(b) Robustness Test Example

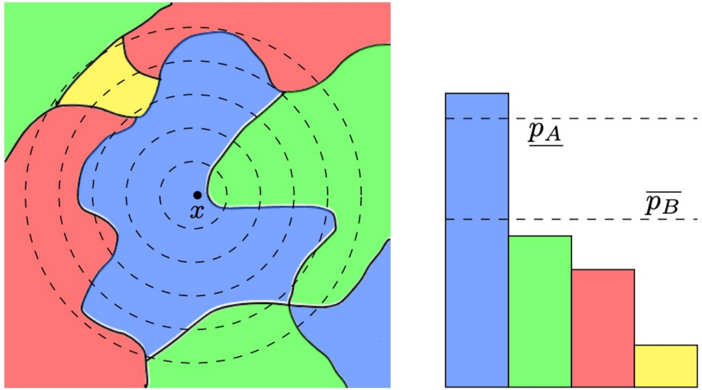
What are the defense mechanisms proposed? guaranteed method

- 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[1]:
  - Make f less sensitive to input perturbations

[1] Cohen *et al.*, Certified Adversarial Robustness via Randomized Smoothing, ICML 2019

What are the defense mechanisms proposed? guaranteed method

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

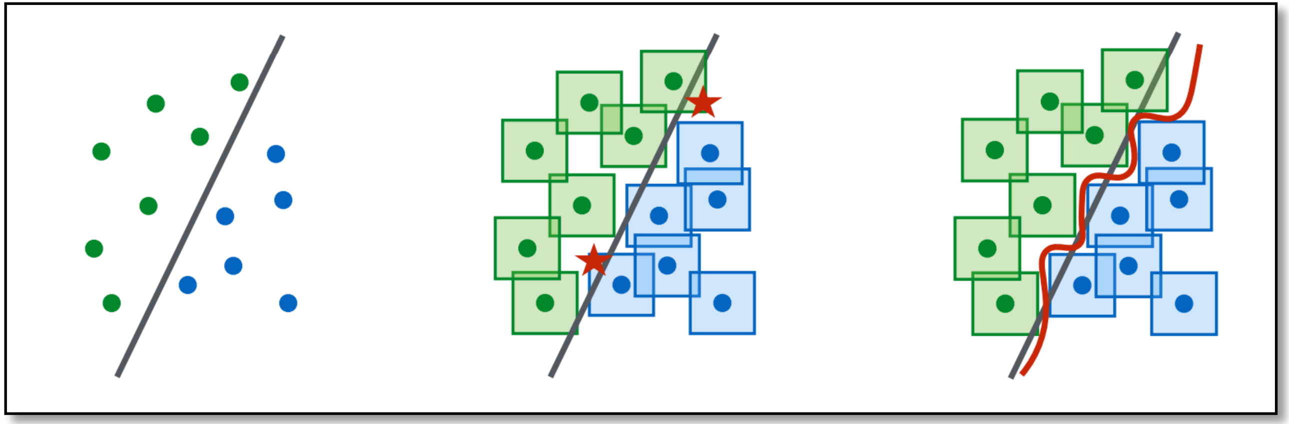


- Certificate!
  - The smoothed classifier g is robust around x with the l_2 radius

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

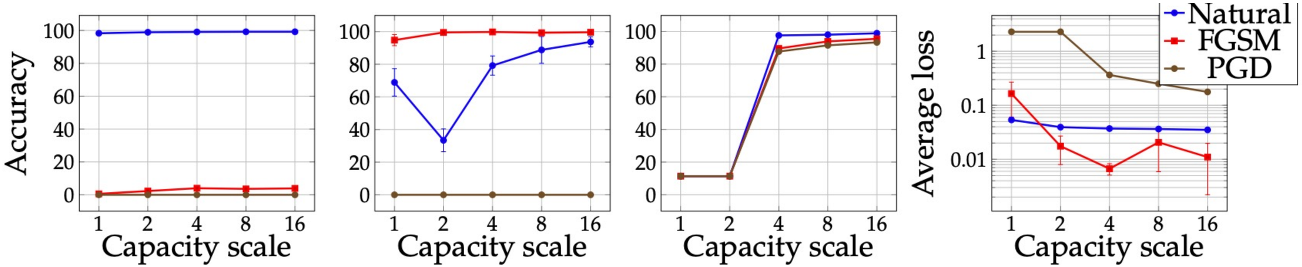
What are the defense mechanisms proposed? guaranteed method

- The key idea: adversarial training
  - Neural networks are universal function approximators[1]
  - They may learn to be resistant to adversarial examples
  - Adversarial training (AT): train models on adversarial examples



[1]Hornik *et al.*, Multilayer feedforward networks are universal approximators, Neural Networks 1989
[2]Madry *et al.*, Toward Deep Learning Models Resistant to Adversarial Attacks, ICLR 2018

- The key idea: adversarial training
  - Adversarial training (AT): train models on adversarial examples
    - (MNIST) It reduces an error rate from 89% to 18% on FGSM
    - (CIFAR10) It reduces an error rate from 1% to 44% on PGD



CIFAR10

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

(a) Standard training

	Simple	Wide
Natural	87.4%	90.3%
FGSM	90.9%	95.1%
PGD	0.0%	0.0%

(b) FGSM training

	Simple	Wide
Natural	79.4%	87.3%
FGSM	51.7%	56.1%
PGD	43.7%	45.8%

(c) PGD training

	Simple	Wide
Natural	0.00357	0.00371
FGSM	0.0115	0.00557
PGD	1.11	0.0218

(d) Training Loss

[1] Hornik *et al.*, Multilayer feedforward networks are universal approximators, Neural Networks 1989

[2] Madry *et al.*, Toward Deep Learning Models Resistant to Adversarial Attacks, ICLR 2018

What are the defense mechanisms proposed? guaranteed method

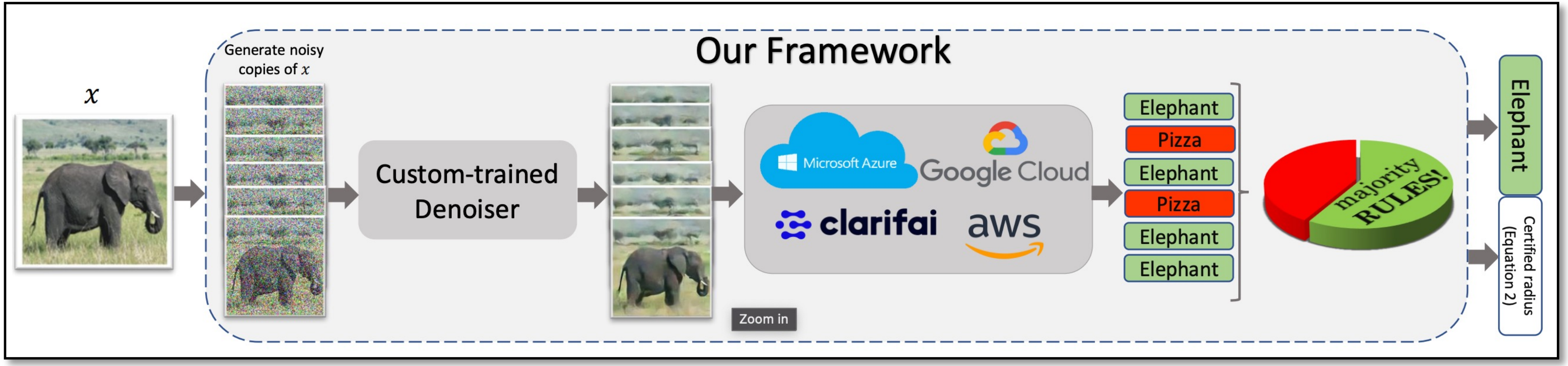
- Problem in adversarial training:
  - We need to re-train all the models, already trained and on-service?
  - How much would it be practical? [Consider models with 8.3 billion parameters]

- Solution:
  - Denoised smoothing[1]: add a denoiser on top of a pre-trained classifier

[1]Salman *et al.*, Denoised Smoothing: A Provable Defense for Pretrained Classifiers, NeurIPS 2020

What are the defense mechanisms proposed? guaranteed method

- Use a denoiser
  - Train a 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 δ



What are the defense mechanisms proposed? guaranteed method

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

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

Topics for this week

- Trustworthy AI
  - Motivation
  - Preliminaries
    - Machine learning (ML)
    - ML-based systems
  - (Potential) Threats
    - Adversarial attacks
    - Data poisoning
    - Privacy attacks
  - Discussion
    - More issues (social bias, fairness, …)

Thank You!

Tu/Th 4:00 – 5:50 PM

Sanghyun Hong

sanghyun.hong@oregonstate.edu

Oregon State University

SAIL

Secure AI Systems Lab