Evasion Attacks: Adversarial Examples

Lecture 3

CS25800, Adv. ML

Lecture 2: Recap

- Key elements of deep neural networks (DNN)
- How to define, train, and evaluate a DNN model
- Key elements of CNN (time permits)

→ Challenges facing training a "good" DNN model for classification

→Vulnerabilities of DNN models

Attacking CNN (Vision) Models

Attack Taxonomy

- Attacker's objectives
  - **Evasion attacks**: Cause the model to make mistakes (misclassification) by "manipulating" **the input at inference time**.
  - **Poisoning attacks**: same as the above, but by "manipulating" the model's **training data**
  - **Data/model privacy attacks**: Obtain knowledge of the model or its training data
    - Model stealing (extraction) attacks, Member inference attacks

- Attacker's knowledge
  - **White-box attacks**: **full** knowledge of the ML model (including architecture, weights, loss function, hyperparameters, etc.)
  - **Black-box attacks**: no knowledge of the model, may query the model via API access
  - **Gray-box attacks**: some knowledge of the model

This Week:
Adversarial Examples

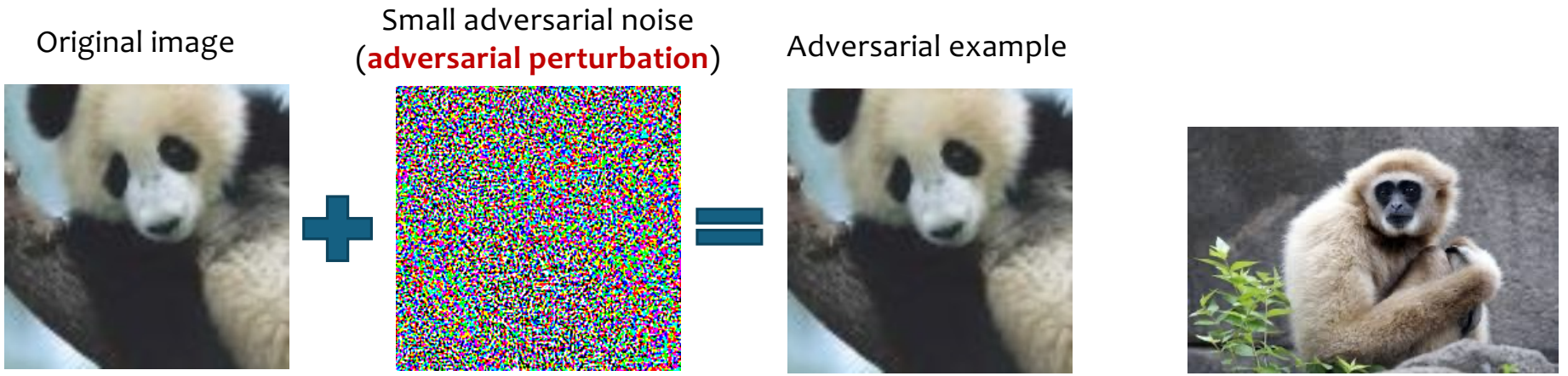
The most popular evasion attack
also know as: Adversarial Perturbations

Today: White-box Attacks

Thur: Black-box Attacks

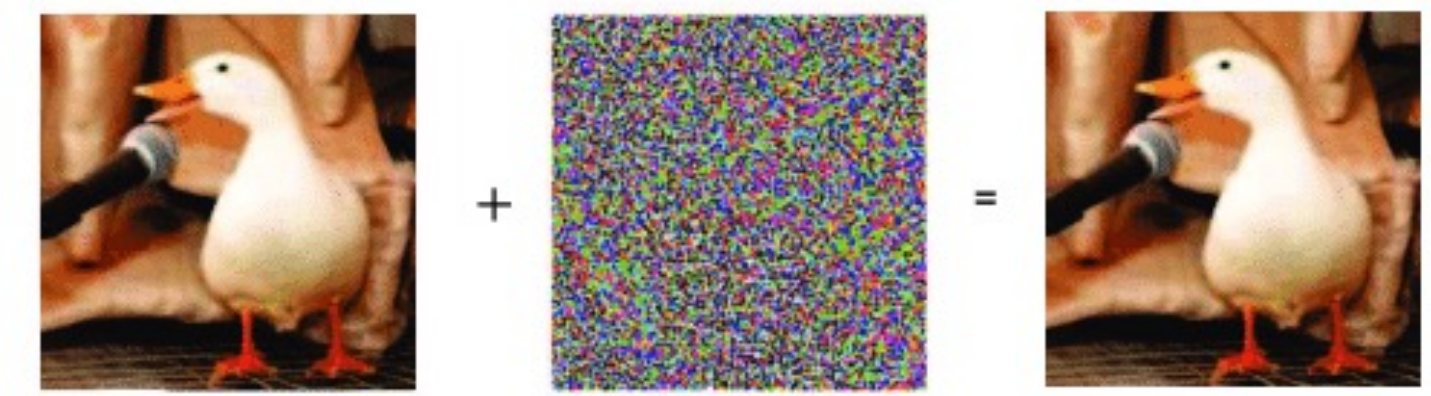
Definition: Adversarial Examples

- **Inputs to a ML model that an attacker intentionally designed so that the model will make mistakes (e.g. misclassification) on them**



Reading: Explaining and Harnessing Adversarial Examples, Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy, ICLR 2015

Object recognition



Speech recognition

Motivations for Creating Adversarial Examples

- Bypass content moderation
  - Spam filter, harmful content detector, genAI detector, ..
  - Digital content
- Bypass biometric authentication
  - Facial scanning-based authentication: airport, buildings, homes, etc.
  - Voice-based authentication: bank, etc.
  - Physical behaviors captured via sensors
- Bypass security checks
  - Weapons, drugs recognized by scanners as safe objects
- Many many more …

Goals of Today's Lecture

- Understand the methodology for adding adversarial perturbations to an image to cause misclassification
  - **Today's focus**: When the attacker has full access to the classification model (i.e., white-box access)
  - Targeted vs. untargeted attacks
  - Three attack designs: FGSM, PGD, C&W
- Physical-world adversarial examples

Formal Definition:
Untargeted Adversarial Examples

- x: an input to the model
- F: the (classification) model defined by its parameter **W**
- y=F(x;W) : the class label predicted by the model for input x

$\mathbf{x_{adv}=x+\eta}$ is a successful, untargeted adversarial example, if η is bounded by some pre-defined constraints, and

$$F(x+\eta\ ;\ \mathbf{W}) \neq F(x;\ \mathbf{W})$$

Note: adversarial perturbation η is generally *input-specific*, i.e., η changes when x changes

Formal Definition:
Targeted Adversarial Examples

- x: an input to the model
- F: the (classification) model defined by its parameter **W**
- y=F(x,W) : the class label predicted by the model for input x
- $\mathbf{y_{target}}$: the classification outcome targeted by the attack ($\mathbf{y_{target}} \neq y$)

$x_{adv}=x+\eta$ is a successful, targeted adversarial example, if η is bounded by some constraints, and

$$F(x+\eta\ ;\ \mathbf{W}) = \mathbf{y_{target}} \neq F(x;\ \mathbf{W})$$

Given an input x, a model F,

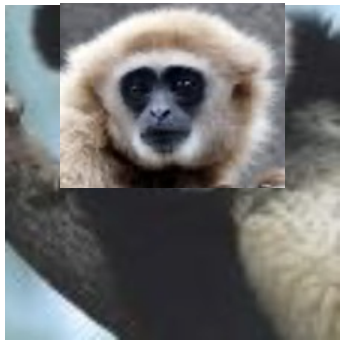
an adversarial perturbation attack is all about finding a constrained (or imperceivable) perturbation η on x to make F "misclassifies" (x+ η)

Q1: How to define the constraint on η
Q2: How to find the working η

Why do we need to bound $\boldsymbol{\eta}$?

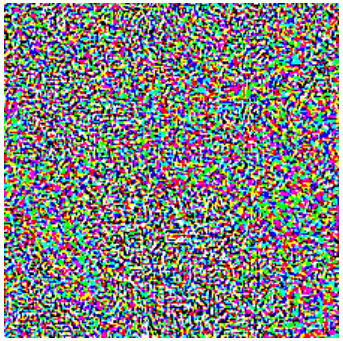
- Attack Stealthiness:
  - Human eyes should not tell the difference
  - Avoid raising suspicion

x
Original image



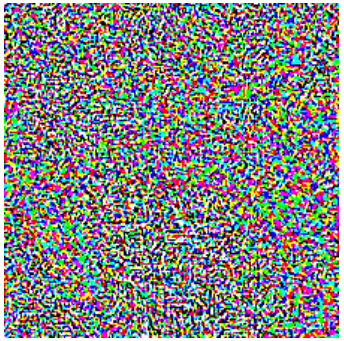
$(\mathbf{x}+\boldsymbol{\eta})$ ✗

$(\mathbf{x}+\boldsymbol{\eta})$
Adversarial example



$\boldsymbol{\eta}$
Small adversarial noise
(**adversarial perturbation**)

For now let's assume we have a way to define the bound on $\boldsymbol{\eta}$ (will discuss soon)



How to find an instance of η that makes model F "misclassifies" x+ η ?

Three key algorithms for finding the right $\boldsymbol{\eta}$

- Fast gradient sign method (FGSM) attack
  - *Explaining and Harnessing Adversarial Examples (ICLR 2015)*
- Towards Deep Learning Models Resistant to Adversarial Attacks
  - *Towards Deep Learning Models Resistant to Adversarial Attacks (arxiv 2017, ICLR 2018)*
- Carlini & Wagner (C&W) attack
  - *Towards Evaluating the Robustness of Neural Networks (IEEE S&P 2017)*

- White-box Attacks: Attacker has full knowledge of the model
- Targeted and untargeted attacks

Fast gradient sign method (FGSM)

- The first attack design for adversarial examples
- Simplicity

Explaining and Harnessing Adversarial Examples, Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy, ICLR 2015

Untargeted FGSM Attack (Fast gradient sign method)

- Attacker goal: (untargeted) F(x+ η; **W**) ≠ F (x; **W**) =y

$$\boldsymbol{\eta = \epsilon \cdot sign\left(\nabla_x J(\mathbf{W}, \mathbf{x}, \mathbf{y})\right)}$$

Move in the direction of positive gradient to increase the loss wrt y

Model's Loss Function

Gradient over x

(measures the importance of each pixel towards the model's classification of x to y)

$$\boldsymbol{sign}\,(z) = \begin{cases} -1 & z < 0 \\ 0, & z = 0 \\ 1, & z > 0 \end{cases}$$

Untargeted FGSM Attack (the panda example)

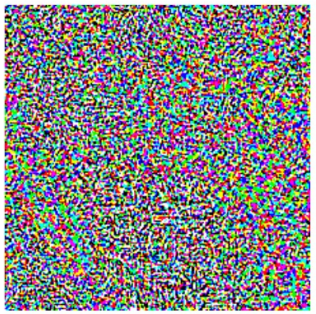
- Attacker goal: (untargeted) F(x+ η; **W**) ≠ F (x; **W**) =y

$$\boldsymbol{\eta = \epsilon \cdot sign\left(\nabla_x J(\mathbf{W}, \mathbf{x}, \mathbf{y})\right)}$$

$\boldsymbol{x}$

"panda"
57.7% confidence

$+ .007 \times$



$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"
8.2% confidence

$=$

$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"
99.3 % confidence

Untargeted FGSM: $\eta = \epsilon \cdot sign\left(\nabla_x J(\mathbf{W}, \mathbf{x}, \mathbf{y})\right)$

Targeted FGSM Attack

- Attacker goal: $F(x+\eta; \mathbf{W}) = \mathbf{y}_{target} \neq F(x; \mathbf{W})$

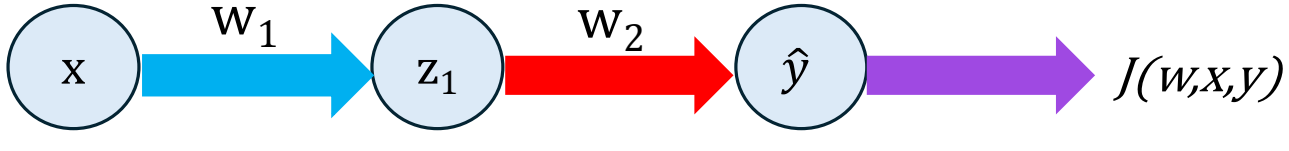
$$\eta = -\epsilon \cdot sign\left(\nabla_x J(\mathbf{W}, \mathbf{x}, \mathbf{y}_{target})\right)$$

Move in the direction of negative gradient to reduce the loss wrt y_{target}

Gradient over x
(measures the importance of each pixel towards the model's misclassification of x to y_{target})

How to Calculate $\nabla_x J(\mathbf{W}, \mathbf{x}, \mathbf{y})$

- Gradient over input data (x) rather than model weights (W)



$$\frac{\partial J(W)}{\partial x} = \frac{\partial J(W)}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z_1} \cdot \frac{\partial z_1}{\partial x}$$

Backpropagation traverses the model in reverse order,
but the last step is a **gradient over x rather than w**

Pros & Cons of FGSM Attack

- Pros:
  - Simple to compute
    - One forward propagation
    - One back propagation
  - Better performance on untargeted attacks than targeted attacks
    - Easier to move away from an optimal point

- Cons
  - Heuristic-based → no guarantee of success
  - Need large ϵ values → visible perturbation

Projected Gradient Decent (PGD)

Extending FGSM attack to multiple iterations

Towards Deep Learning Models Resistant to Adversarial Attacks (ICLR 2018)

PGD Attack

- Projected Gradient Decent (PGD)
  - Extending FGSM attack to multiple iterations
  - Repeatedly add perturbation from the FGSM attack to the image
  - Increase the chance of misclassification

$$\boldsymbol{x^{(0)}} = \boldsymbol{x}$$

$$\boldsymbol{\dots}$$

$$\boldsymbol{x^{(k)}} = \boldsymbol{x^{(k-1)}} + \boldsymbol{\epsilon \cdot sign}\left(\boldsymbol{\nabla_x} \boldsymbol{J}\left(\mathbf{W}, \boldsymbol{x^{(k-1)}}, \mathbf{y}\right)\right)$$

$$\boldsymbol{\dots}$$

In each iteration, ensure each pixel value of $x^{(k)}$ remains in valid range [0,1]

Pros & Cons of FGSM/PGD Attacks

- Pros:
  - Simple to compute (per iteration)
    - One forward propagation
    - One back propagation
  - Better performance on untargeted attacks than targeted attacks
    - Easier to move away from an optimum point

- Cons
  - Heuristic-based
  - No guarantee of success
  - Need large ϵ values → visible perturbation
  - In each iteration, pixel clipping (ensuring each pixel is in [0,1]) changes the perturbation → reduce attack success rate

So far,
what is missing in the attack design?

A systematical way of optimizing $\boldsymbol{\eta}$ to cause misclassification

CW Attack

A new methodology for computing the perturbation η by forming and solving a constrained optimization problem

Towards Evaluating the Robustness of Neural Networks (IEEE S&P 2017)

Carlini Wagner (CW) Attack

- Define attack optimization as **finding the minimum perturbation required to cause (targeted) misclassification**

$$\text{minimize}_{\eta} \quad ||x+\eta, x|| \quad \leftarrow \text{Minimize the perturbation}$$

$$\text{such at} \quad F(x+\eta; W) = y_{target} \quad \leftarrow \text{Ensure the perturbed input is a valid image and gets misclassified}$$

$$x+\eta \text{ is a valid image}$$

Challenge 1: *"$F(x+\eta; W) = y_{target}$" is non-differentiable*

Solution: *Find another function f, such that*

$F(x+\eta; W) = y_{target}$ if and only if $f(x+\eta) \leq 0$

C&W Attack

Choose a function f to ensure:
$F(x+\eta; W) = y_{target}$ if and only if $f(x+\eta) \leq 0$

- Finding the minimum perturbation required to cause targeted misclassification

$$\text{minimize}_{\eta} \quad ||x+\eta, x||$$
$$\text{such at} \quad F(x+\eta; W) = y_{target}$$
$$x+\eta \text{ is a valid image}$$

$$\text{minimize}_{\eta} \quad ||x+\eta, x||$$
$$\text{such at} \quad f(x+\eta) \leq 0$$
$$x+\eta \text{ is a valid image}$$

$$\text{minimize}_{\eta} \quad \big||x+\eta, x\big| + \lambda \cdot f(x+\eta)$$
$$x+\eta \text{ is a valid image}$$

(1) What is $f(x+\eta)$?
(2) How to ensure $x+\eta$ is a valid image w/o disrupting optimization

Finding the right $f(x+\eta)$

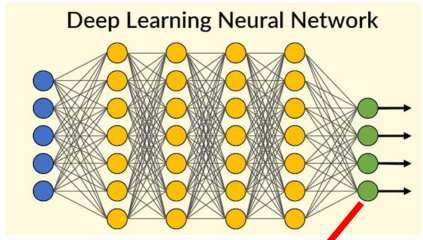
$F(x+\eta;W) = y_{target}$ if and only if $f(x+\eta) \leq 0$

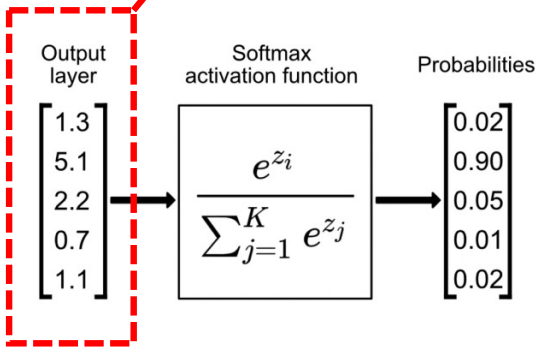
$f(x+\eta)$ for targeted attacks:

$$f(x+\eta) = \left(\max_{i \neq y_{target}} (Z(x+\eta)_i) - Z(x+\eta)_{y_{target}} \right)$$

max logits value across other classes than the target class y_{target}

$Z(x)_i$ is the logits value of the class *i* for the image x





Misclassification to y_{target} $if\ only\ if\ \ Z(x+\eta)_{y_{target}}$ is larger than all other classes

Practical $f(x+\eta)$ for targeted attacks:

- $f(x+\eta) = \max\left(\max_{i \neq y_{target}} (Z(x+\eta)_i) - Z(x+\eta)_{y_{target}}, -\kappa\right)$

- No need to optimize the distance further than $-\kappa$
- κ: measures the level of confidence .
  - Larger κ → more confident adversarial examples

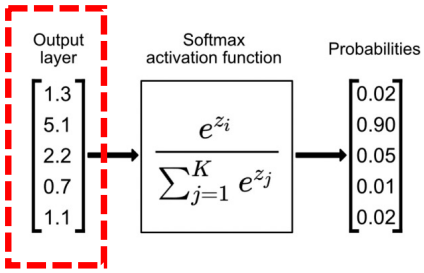
$Z(x+\eta)_{y_{target}}$ needs to be sufficiently larger than the rest

$f(x+\eta)$ for untargeted attacks:

- $f(x+\eta) = \left(Z(x+\eta)_y - \max_{i \neq y}(Z(x+\eta)_i)\right)$

max logits value across all other classes *i* except *y*, *i.e* *so that* $f(x+\eta)<0$ means not be classified as y;
y is the class of unperturbed image (x)

$Z(x)_i$ is the logits value of the class *i* for the image x



$Z(x+\eta)_y$ needs to be smaller than at least one other class (not y)

Ensure $x + \eta$ is a valid image w/o disrupting optimization

"$x + \eta$ is a valid image": *Box Constraints*

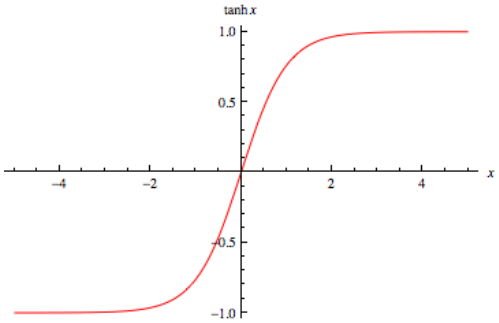
- For a normalized image, it means each pixel value is within [0,1]
- One way to enforce: clip the value to [0,1]

- A better way: ***Change of Variables***

  $x_i + \eta_i = {}^1/_2\,(\tanh(\varpi_i)+1)$

  $\eta_i = {}^1/_2\,(\tanh(\varpi_i)+1) - x_i$

  Since $\tanh(\varpi_i)$ is within [-1,1], $x_i + \eta_i$ is within [0,1]



- instead of optimizing on η_i , optimize on ϖ_i

Putting It Together

$$\text{minimize}_{\eta} \; ||x+\eta, x|| + \lambda \cdot f(x+\eta)$$

$x + \eta$ is a valid image

$$\text{minimize}_{\varpi} \; \left|\left|\frac{1}{2}(\tanh(\varpi)+1), x\right|\right| + \lambda \cdot f\left(\frac{1}{2}(\tanh(\varpi)+1)\right)$$

$$\text{where: } f(x') = \max\left(\max\{Z(x')_i : i \neq y_{target}\} - Z(x')_{y_{target}}, -\tau\right)$$

Using Adam optimizer to solve this optimization

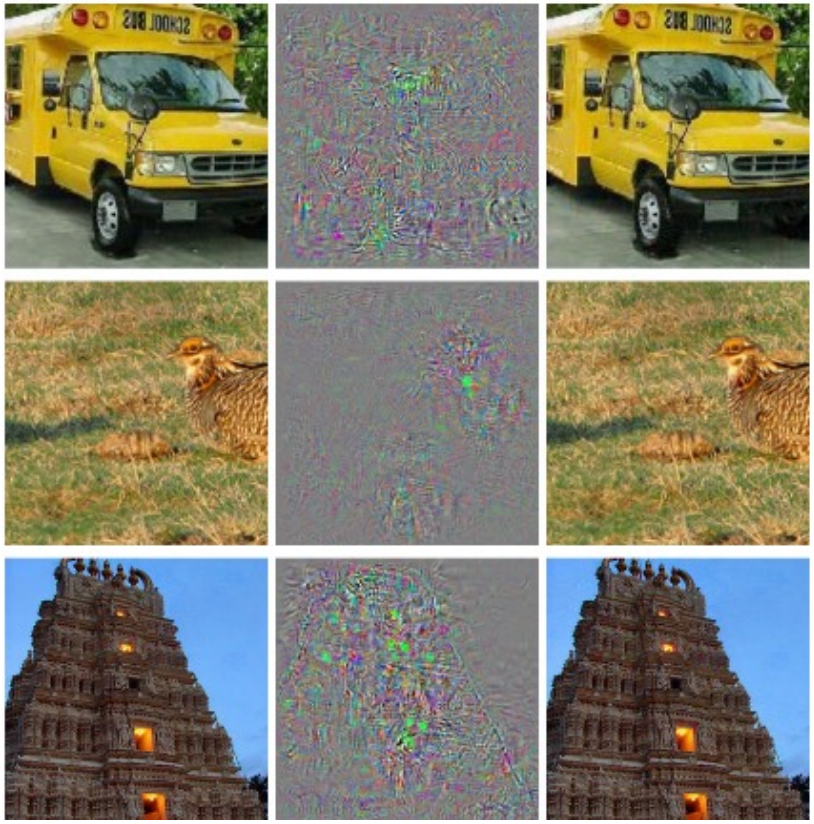
Time for a short 5-min break

Defining/Measuring Perturbation

$$||x + \eta, x||$$

Bound the perturbation so that $(x + \eta)$ and x are visually similar to human eyes,

i.e., Imperceivable perturbation



Another challenging problem: Measuring/Quantifying Human Perception of Visual Data

Define $||x + \eta, x||$

L_p metrics for measuring the norm of a matrix

- L_0: # of different pixels between $x + \eta$ and x
- L_1: sum of absolute pixel value difference
  - For each pixel, calculate the absolute value difference, and sum it over all pixels
- L_2: sum of squared difference between pixels
- L_∞: the maximum pixel value difference
  - For each pixel, calculate the absolute value difference, and report the largest one

Adversarial Examples under different L_p constraints (PGD)

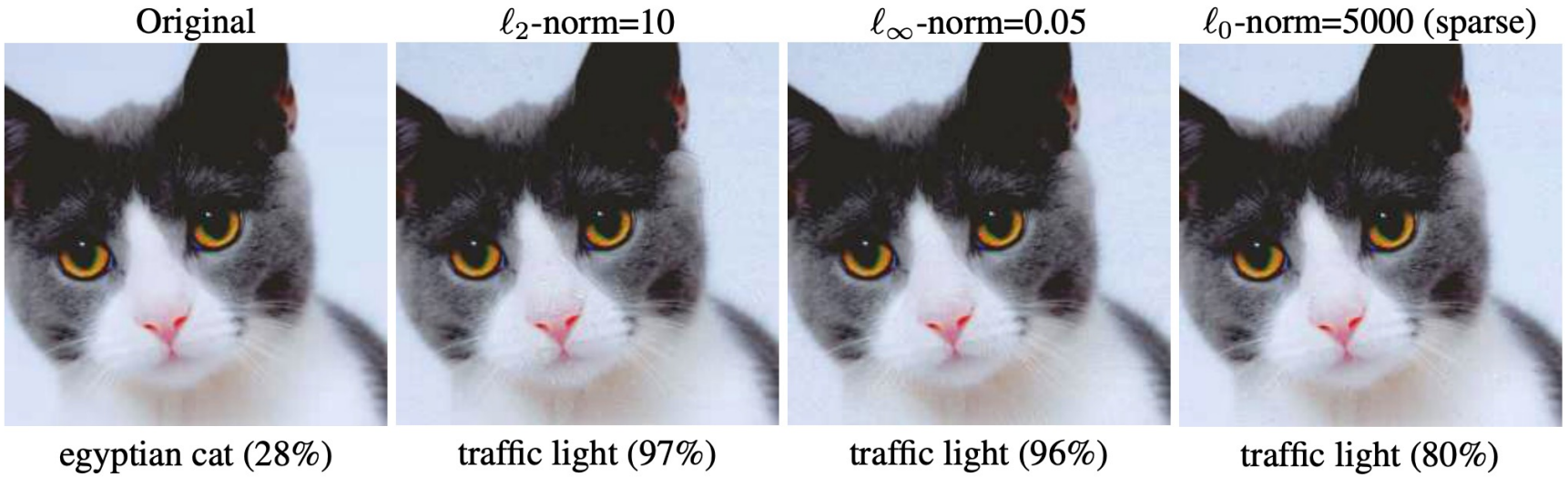


Figure 1: Adversarial examples with different norm constraints formed via the projected gradient method (Madry et al., 2017) on Resnet50, along with the distance between the base image and the adversarial example, and the top class label.

Img source: Are adversarial examples inevitable? https://arxiv.org/pdf/1809.02104

Impact of Perturbation Location

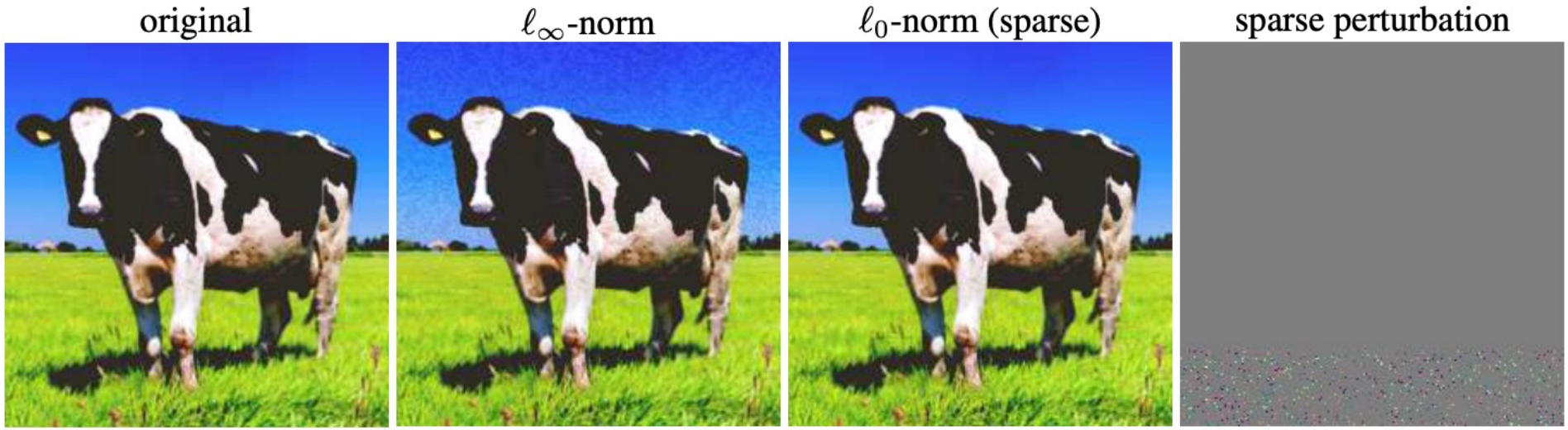
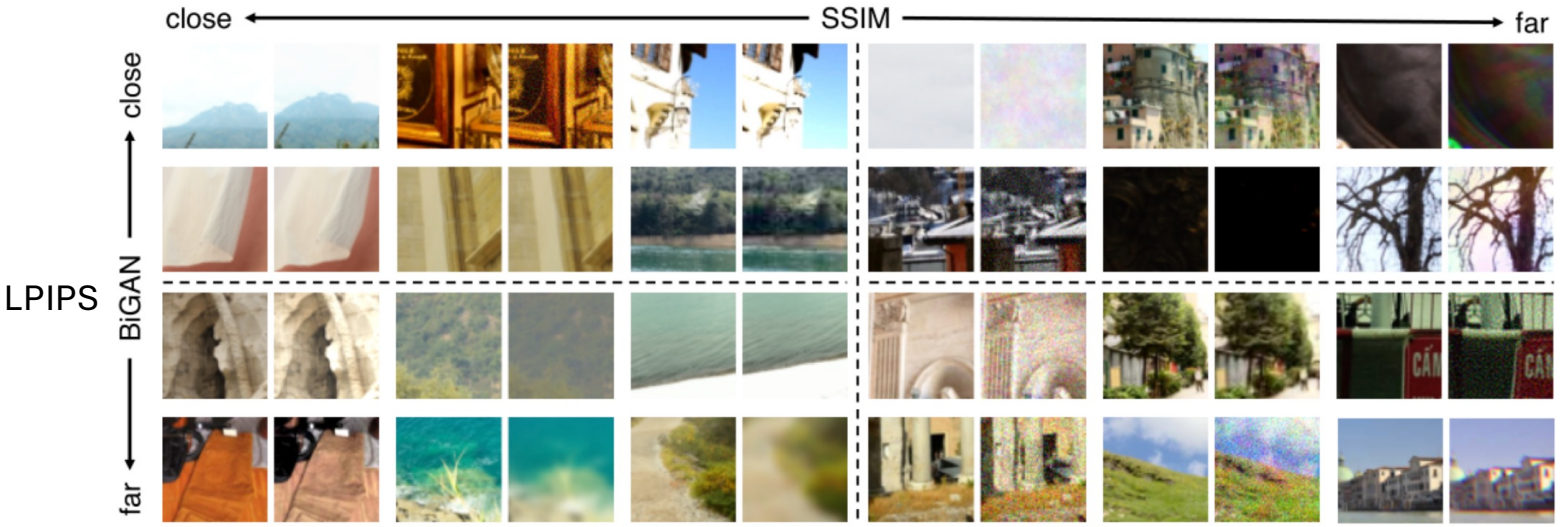


Figure 2: Sparse adversarial examples perturb a small subset of pixels and can hide adversarial "fuzz" inside high-frequency image regions. The original image (left) is classified as an "ox." Under ℓ_∞-norm perturbations, it is classified as "traffic light", but the perturbations visibly distort smooth regions of the image (the sky). These effects are hidden in the grass using ℓ_0-norm (sparse) perturbations limited to a small subset of pixels.

Img source: Are adversarial examples inevitable? https://arxiv.org/pdf/1809.02104

Challenges of Measuring Human Perception

- Evaluating perceptual difference between two images ||x1 - x2||
  - Traditional metrics: Lp, PSNR, SSIM (Structural similarity index measure)
  - Deep metrics: LPIPS (Learned Perceptual Image Patch Similarity)



LPIPS Paper: The Unreasonable Effectiveness of Deep Features as a Perceptual Metric, CVPR 2018

Still an ongoing research topic

Towards Adversarially Robust Perceptual Similarity Metrics

Figure 1. Adversarial examples generated using PGD with $\|\delta\|_\infty \leq 0.05$ on ImageNet-100 validation set. Original and perturbed images are shown in the first and second rows, respectively. The LPIPS/R-LPIPS values for these images are mentioned below each image. In contrast with LPIPS values that are quite large, the R-LPIPS are very small and correctly reflect the small difference between images.

Image src: R-LPIPS - 2nd ICML Workshop on New Frontiers in AdvML

Back to CW Attacks

CW Overview

$$\text{minimize}_{\eta} \ \left|\left|x+\eta, x\right|\right| + \ \lambda \cdot f(\mathrm{x}+\eta)$$

$x + \eta$ is a valid image

$$\text{minimize}_{\varpi} \ \left|\left|\frac{1}{2}(\tanh(\varpi)+1), x\right|\right| + \ \lambda \cdot f\left(\frac{1}{2}(\tanh(\varpi)+1)\right)$$

$$\text{where: } f(x') = \max\left(\max\{Z(x')_i : i \neq y_{target}\} - \ Z(x')_{y_{target}}, -\kappa\right)$$

Using Adam optimizer to solve this optimization

Detailed CW Attacks in the original paper

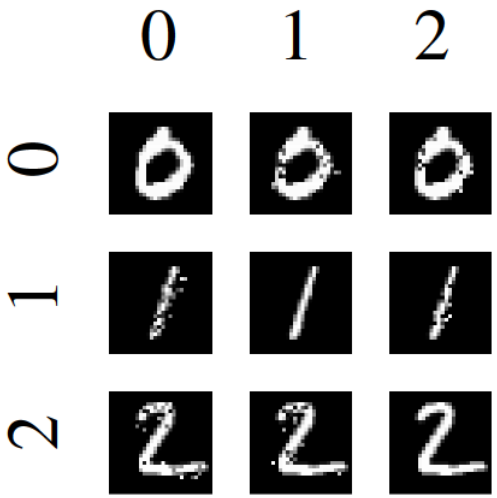
- Three attack Designs
  - L_∞ attack --- *modified* $||x+\eta, x||_\infty$ *from counting the maximum per-pixel change to counting the sum of pixel change beyond a threshold (details in the paper)*
  - L_2 attack
  - L_0 attack

- Strongest attack at the time of publishing
- Low amount of perturbation

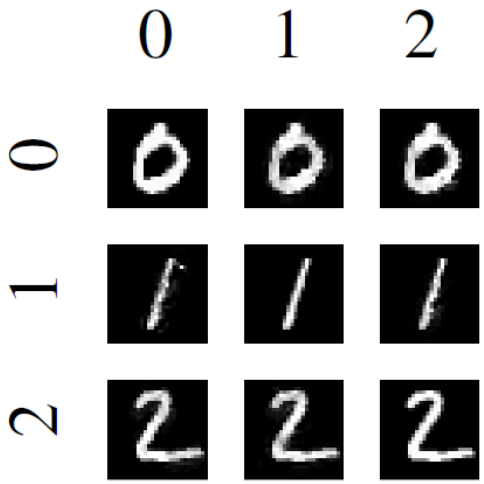
Towards Evaluating the Robustness of Neural Networks (IEEE S&P 2017)

C&W Attacks on MNIST Classifier

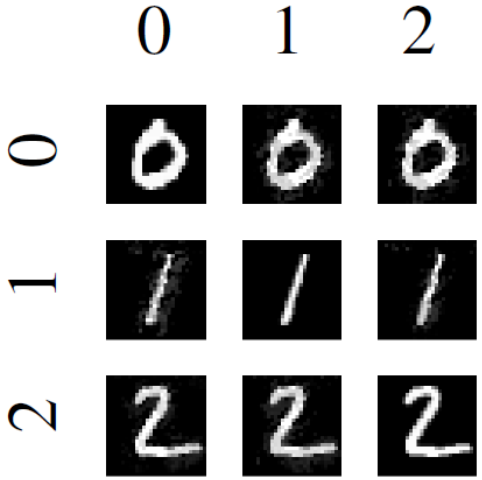
L_0 attack



L_2 attack



L_∞ attack



Evaluating Attacks

- Testing on multiple input data, and models
- Study Attack Success Rate and Perturbation Amount

- Selecting the target class during evaluation
  - Average Case: select the target class uniformly at random among the labels that are not the correct label
  - Best Case: perform the attack against all incorrect classes, and report the target class that was least difficult to attack
  - Worst Case: perform the attack against all incorrect classes, and report the target class that was most difficult to attack

Comparing C&W to Existing Attacks

	Best Case				Average Case				Worst Case			
	MNIST		CIFAR		MNIST		CIFAR		MNIST		CIFAR	
	mean	prob	mean	prob	mean	prob	mean	prob	mean	prob	mean	prob
Our L_0	8.5	100%	5.9	100%	16	100%	13	100%	33	100%	24	100%
JSMA-Z	20	100%	20	100%	56	100%	58	100%	180	98%	150	100%
JSMA-F	17	100%	25	100%	45	100%	110	100%	100	100%	240	100%
Our L_2	1.36	100%	0.17	100%	1.76	100%	0.33	100%	2.60	100%	0.51	100%
Deepfool	2.11	100%	0.85	100%	–	-	–	-	–	-	–	-
Our L_∞	0.13	100%	0.0092	100%	0.16	100%	0.013	100%	0.23	100%	0.019	100%
Fast Gradient Sign	0.22	100%	0.015	99%	0.26	42%	0.029	51%	–	0%	0.34	1%
Iterative Gradient Sign	0.14	100%	0.0078	100%	0.19	100%	0.014	100%	0.26	100%	0.023	100%

TABLE IV

COMPARISON OF THE THREE VARIANTS OF TARGETED ATTACK TO PREVIOUS WORK FOR OUR MNIST AND CIFAR MODELS. WHEN SUCCESS RATE IS NOT 100%, THE MEAN IS ONLY OVER SUCCESSES.

Amount of perturbation (↓)

Attack Success rate (↑)

Summary

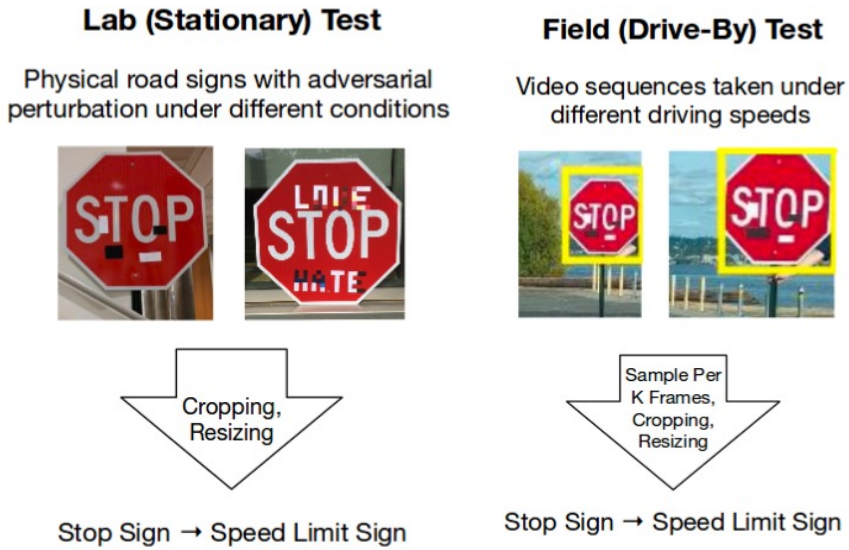
- Attacks (and Defenses) are constantly evolving
  - Overtime, increasing complexity and performance
  - FGSM → PGD → C&W → ….
- Perturbation measurement to match human perception is hard
  - LPIPS takes time to compute (input images into a DNN to extract features and then compute feature-level distance)
  - Practical adversarial attacks generally use Lp distance

Goals of Today's Lecture

- Understand the methodology for adding adversarial perturbations to an image to cause misclassification
  - **Today's focus:** When the attacker has full access to the classification model (i.e., white-box access)
  - Targeted vs. untargeted attacks
  - Three attack designs: FGSM, PGD, C&W
- **Physical world adversarial examples**
  - **Different optimization problems**

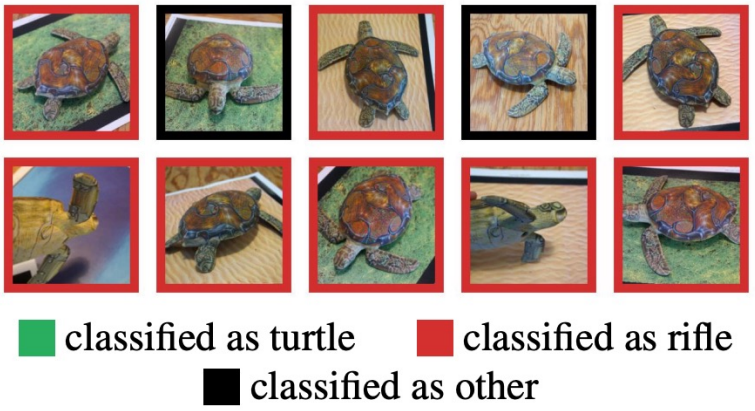
“Physical” Adversarial Examples

- Targeting self-driven car’s vision model to **misclassify traffic signs**
- 100% attack success in lab test, and 85% during real-world test



Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR 2018

- 3D printed AE objects
- Randomly sampled poses of a 3D-printed **turtle** adversarially perturbed to classify as **a rifle** at every viewpoint



Synthesizing Robust Adversarial Examples, ICML 2018

Adversarial Patch

- Prevent the object detection model (YOLOv2) from identifying the person
- **Hiding the person** from a security camera system

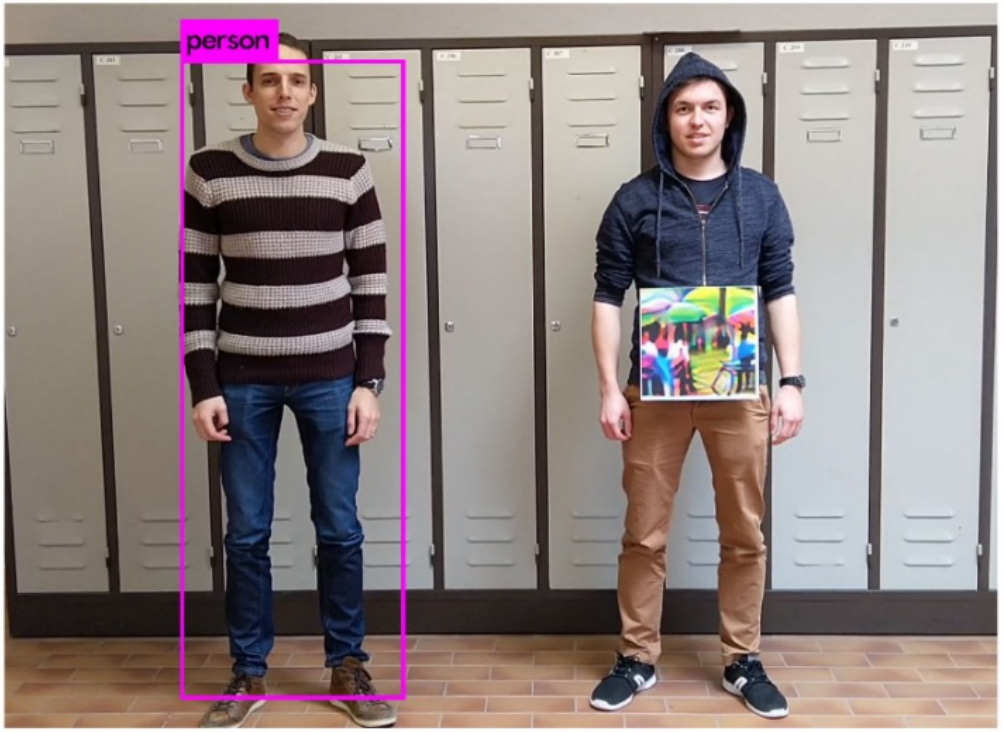


Figure 1: We create an adversarial patch that is successfully able to hide persons from a person detector. Left: The person without a patch is successfully detected. Right: The person holding the patch is ignored.

Fooling automated surveillance cameras: adversarial patches to attack person detection, CVPR Workshop: CV-COPS, 2019

Adversarial Examples against Segmentation

Universal Adversarial Perturbations Against Semantic Image Segmentation, ICCV 2017