ECE595 / STAT598: Machine Learning I
Lecture 36 Defending Attacks

Spring 2020

Stanley Chan

School of Electrical and Computer Engineering
Purdue University

PURDUE
UNIVERSITY

Today's Agenda

- We have seen enough attack strategies
- Today we want to look at **defense** strategies
- We will study three classes of defenses:
  - Pre-processing
  - Randomness
  - Adversarial training
- Most of these defenses are developed for **deep neural networks**.
- The principles are not difficult, although the implementation might be challenging.
- We choose to talk about deep defenses because we want you to connect to the real world.
- Linear models for insight, deep models for practice.
- We will also study the **holes** of these defense, using the concept of **obfuscated gradient**.
- Finally we will offer our **bias** opinion.

Outline

- Lecture 33-35 Adversarial Attack Strategies
- Lecture 36 Adversarial Defense Strategies
- Lecture 37 Trade-off between Accuracy and Robustness

Today's Lecture

- Three classes of defenses
  - Pre-processing
  - Adversarial training
  - Randomness
- Limitation of current defense (and also attacks)
  - Obfuscated gradient
  - Physical attack?
  - Real attacks in social media today

Three Defenses

- Pre-processing
- Randomness
- Adversarial training

Pre-Processing

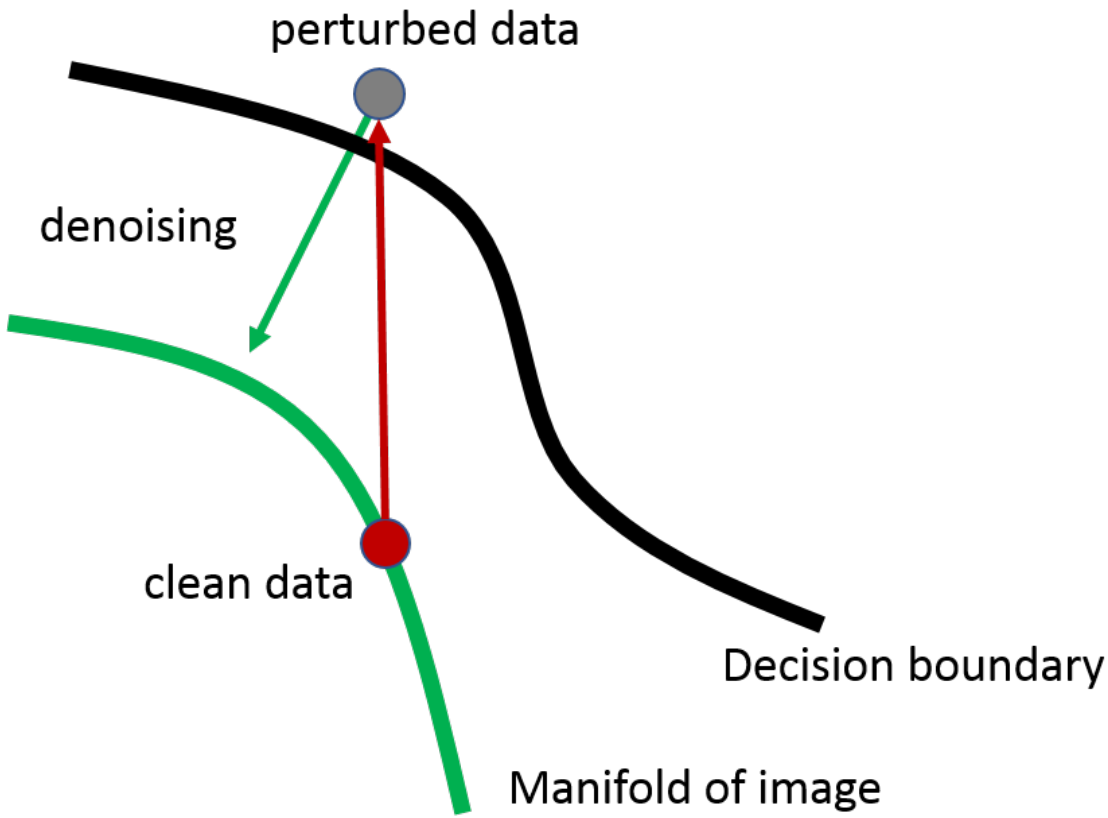
Main idea:

- Given a perturbed data $\boldsymbol{x}$
- Let us do some data pre-processing $g(\boldsymbol{x})$
- So that if you directly classify $\boldsymbol{x}$ using $f(\boldsymbol{x})$ you fail
- But if you classify $f(g(\boldsymbol{x}))$ you will succeed

There are many of such examples

- Denoising: $\boldsymbol{x}$ is perturbed, so I denoise $\boldsymbol{x}$
- Geometry: Project the perturbed data back to the image manifold

Pre-Processing



Example 1: Input Transform

Guo et al. (2018) Countering Adversarial Images using Input Transformations
`https://arxiv.org/abs/1711.00117`

- Published in ICLR 2018
- Idea: Process the input (attack) data $\boldsymbol{x}$
- One approach they propose:

$$\underset{\boldsymbol{z}}{\text{minimize}} \quad \|(1-X)\odot(\boldsymbol{z}-\boldsymbol{x})\|_2 + \lambda \text{TV}(\boldsymbol{z})$$

- $\text{TV}(\boldsymbol{z})$ is the total variation norm of a vector
- X is an array of Bernoulli random variables (either 0 or 1)
- $\odot$ is element-wise multiplication
- In their paper they have also introduced other types of input transforms, e.g., JPEG compression.

Total Variation

- They call this the total variance minimization
- Another classical example of computer scientists. Should be total *variation*.
- Total variation has been around for 3 decades.
- You can trace the idea back to Rudin and Osher in 1992.
- Over the past decade there are numerous total variation solvers.
- For example, my lab has one method based on augmented Lagrangian (2011) `https://ieeexplore.ieee.org/document/5779734`

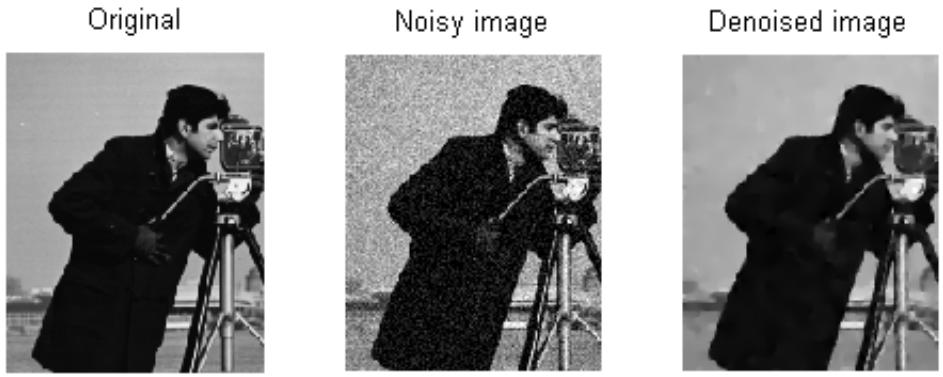


Figure is taken from Wikipedia

Training the Model

How to defend?

- The following diagram is taken from Guo et al.'s paper
- You need to "educate" the classifier about the input transform

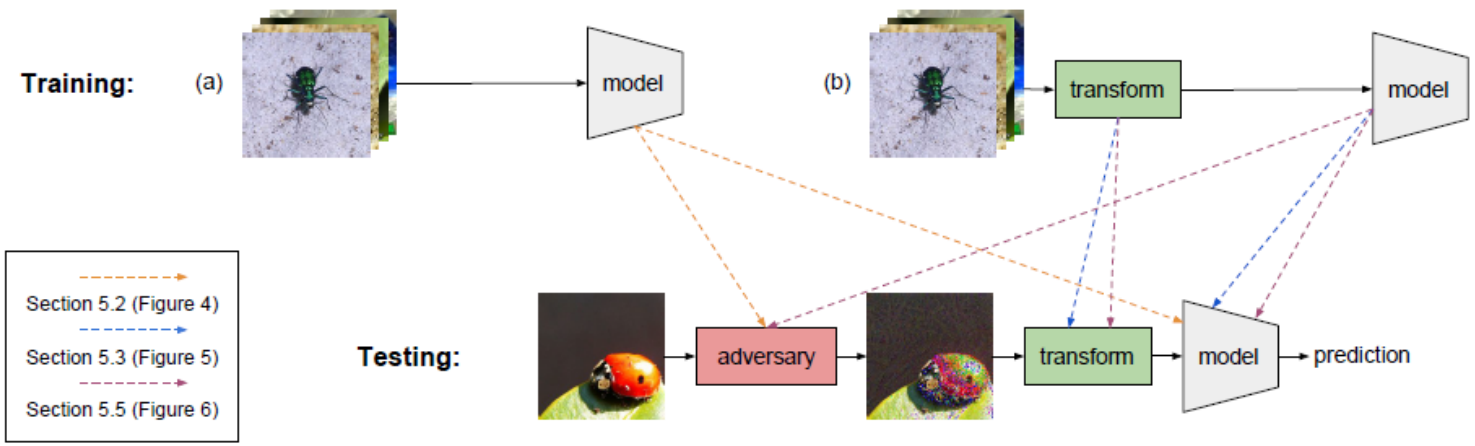


Figure 3: Block diagram detailing the differences between the experimental setups in Section 5.2, 5.3, and 5.4. We train networks (a) on regular images or (b) on transformed images; we test the networks on transformed adversarial images. For each of the three setups, dashed arrows indicate which model is used by the adversary and which model is used by the classification model.

Example 2: High-level Representation Guide Denoiser

Liao et al. (2017) Defense against Adversarial Attacks Using High-Level Representation Guided Denoiser, `https://arxiv.org/abs/1712.02976`

- NIPS 2017 challenge winner
- Idea: Train an image denoiser
- Why not just use an off-the-shelf denoiser?
- Start with a vanilla classifier (network in their setting)
- Send a pair of clean and noisy images through the same network
- Pull the respective features. They should look different.
- Use the residue as a loss, and train a denoiser.

Guided Denoiser

Here is their schematic diagram

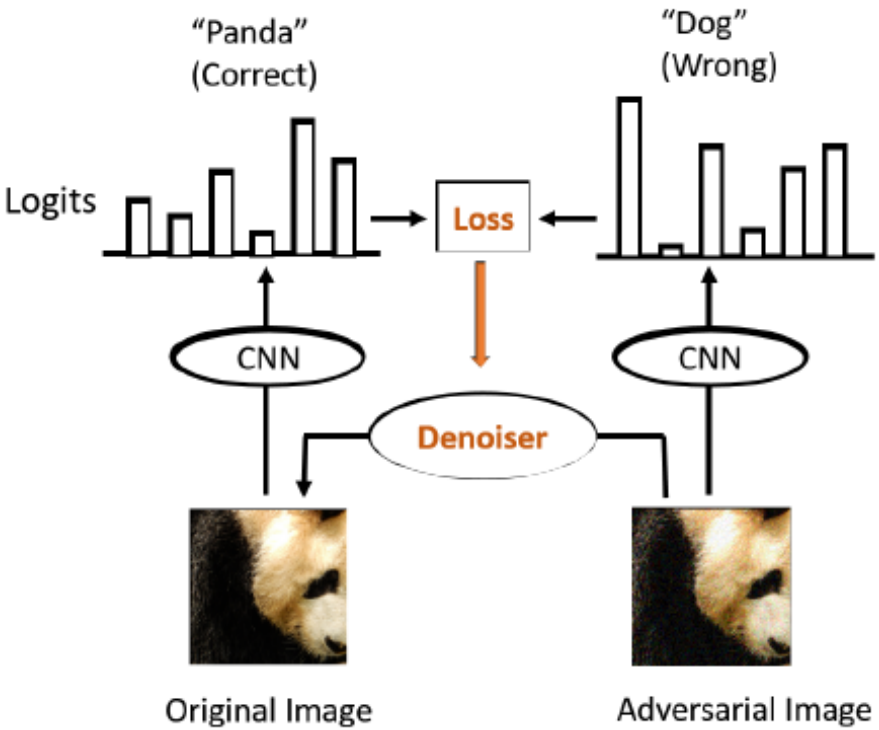


Figure 1: The idea of high-level representation guided denoiser. The difference between the original image and adversarial image is tiny, but the difference is amplified in high-level representation (logits for example) of a CNN. We use the distance over high-level representations to guide the training of an image denoiser to suppress the influence of adversarial perturbation.

Feature Regularization

- Conventionally, when training a denoiser the loss function is between the prediction $f(\boldsymbol{y})$ and the ground truth $\boldsymbol{x}$

$$L(f(\boldsymbol{y}), \boldsymbol{x}^*) = \|f(\boldsymbol{y}) - \boldsymbol{x}^*\|^2$$

- Their approach is to define the loss as

$$L(f(\boldsymbol{y}), \boldsymbol{x}^*) = \|\phi(f(\boldsymbol{y})) - \phi(\boldsymbol{x}^*)\|^2$$

  where ϕ is the feature extracted by the classifier.
- They call it high-level representation guided denoiser.
- In my opinion, this is nothing but Geoffrey Hinton's knowledge distillation (2015) `https://arxiv.org/abs/1503.02531`
- There is another very interesting paper by Zhang et al. (2018) `https://arxiv.org/abs/1801.03924` which uses deep features for image comparisons

Three Defenses

- Pre-processing
- Randomness
- Adversarial training

Randomness

Main Idea:

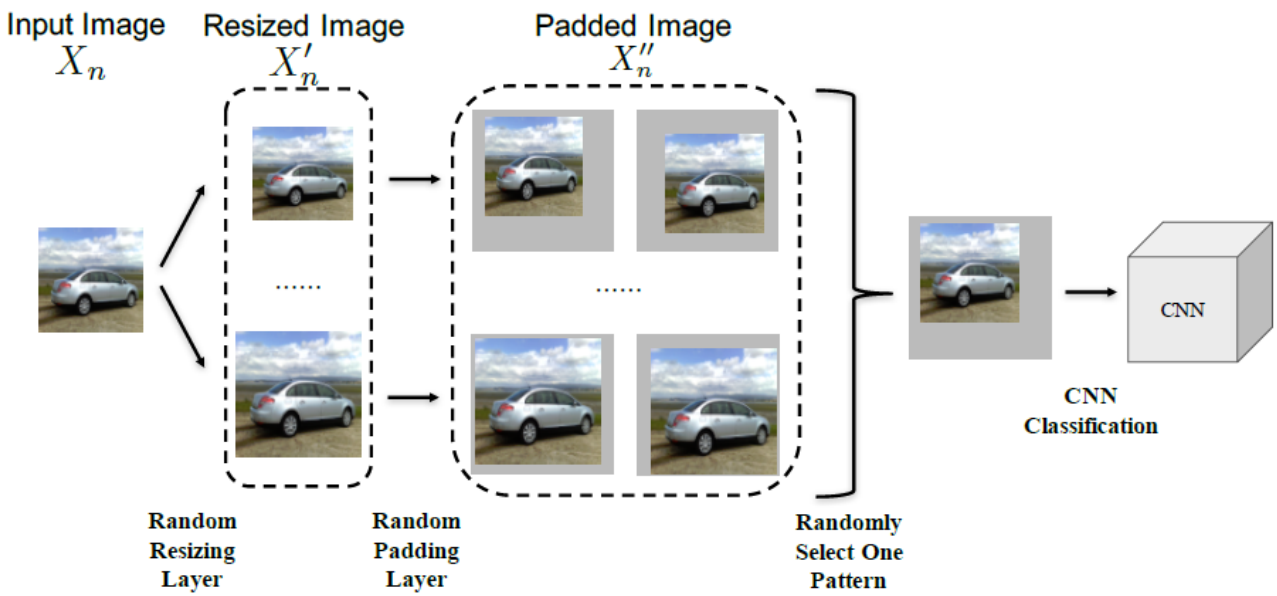
- When the perturbed data comes, let me add randomness to the problem
- E.g., use a suite of classifiers, and you don't know which one I am using
- E.g., randomly flip a few pixels, or mask out some pixels then solve a matrix completion problem
- E.g., randomly move around the pixels so that the attack is distorted

- Since attack is so specific along one direction, a small random perturbation is enough to change its path
- In high dimensional space, slight change in direction will cause significant change in destination
- As long as you do not know the exact state of me, you won't be able to attack me

Example 1: Randomize Input Data

Xie et al. (2018) Mitigating Adversarial Effects Through Randomization, `https://arxiv.org/abs/1711.01991`

- Published in ICLR 2018
- Randomly resize and pad the image to create uncertainty



Example 2: Pixel Deflection

Prakash et al. (2018), Deflecting Adversarial Attacks with Pixel Deflection, `https://arxiv.org/abs/1801.08926`

- Published in CVPR 2018
- Random pick a pixel. Create a neighborhood. Replace the pixel by one of pixels in the neighborhood.

Algorithm 1: Pixel deflection transform

Input : Image I, neighborhood size r
Output: Image I' of the same dimensions as I

for $i \leftarrow 0$ **to** K **do**
  Let $p_i \sim \mathcal{U}(I)$
  Let $n_i \sim \mathcal{U}(R_p^r \cap I)$
  $I'[p_i] = I[n_i]$
end

Example 3: Ensemble Method

Taran et al. (2019), Defending against adversarial attacks by randomized diversification, `https://arxiv.org/abs/1904.00689`

- Published in CVPR 2019
- Idea: Create a suite of K classifiers
- Each classifier has a **key**.
- Pull the features (e.g., last layer before softmax)
- The there is an aggregator which knows the key. Key is used to combine the features
- Attacker does not know the key
- In my opinion, this method is okay for black-box attack. If it is a straight white-box attack (i.e., attacker knows the key), it is unclear whether the method will work.
- If I am the attacker, if I know the distribution of the key, I can just attack the average case. I won't be as successful, but I can still attack you. You are paying the price of a much bigger model.

Example 3: Ensemble Method

Example: Using discrete cosine transform

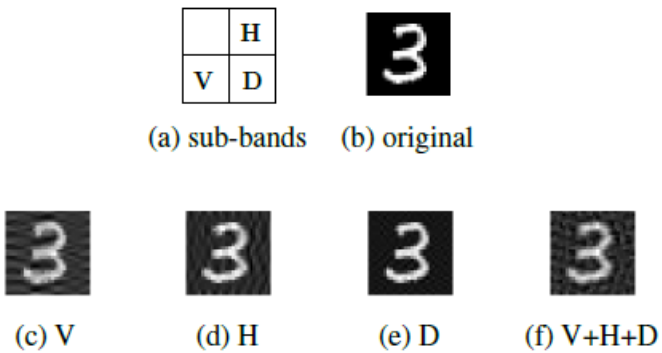


Figure 6: Local randomization in the DCT sub-bands by key-based sign flipping.

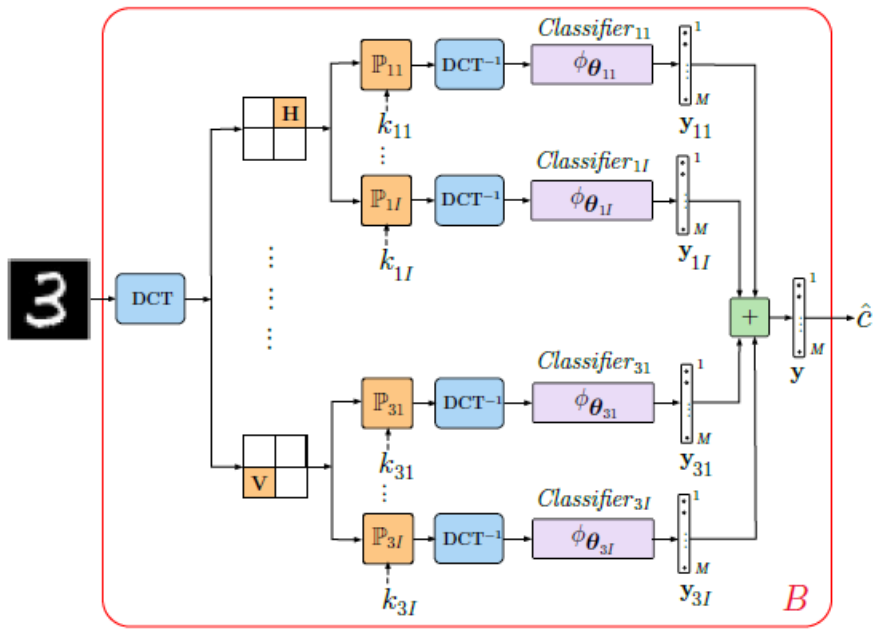


Figure 7: Classification with local DCT sign permutations.

Three Defenses

- Pre-processing
- Randomness
- Adversarial training

Adversarial Training

Madry et al. (2018), Towards Deep Learning Models Resistant to Adversarial Attacks, `https://arxiv.org/pdf/1706.06083.pdf`

- Published in ICLR 2018
- Idea: Solve a minimax problem

$$\underset{\theta}{\text{minimize}} \quad \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\max_{\delta\in\mathcal{S}} \; L(\theta, \boldsymbol{x}+\delta, \boldsymbol{y})\right] \tag{1}$$

- $\delta \in \mathcal{S}$ is the attack added to the input data $\boldsymbol{x}$. $\boldsymbol{y}$ is the truth.
- $\mathcal{S}$ defines the set of allowable attacks. E.g., ℓ_∞ ball.
- You take the maximum of the loss $L(\theta, \boldsymbol{x}+\delta, \boldsymbol{y})$ by searching for the most nasty attack δ.
- Then take expectation over the training set $\mathcal{D}$ to compute the (empirical) risk.
- Finally minimize the risk.

Explanation by Madry et al.

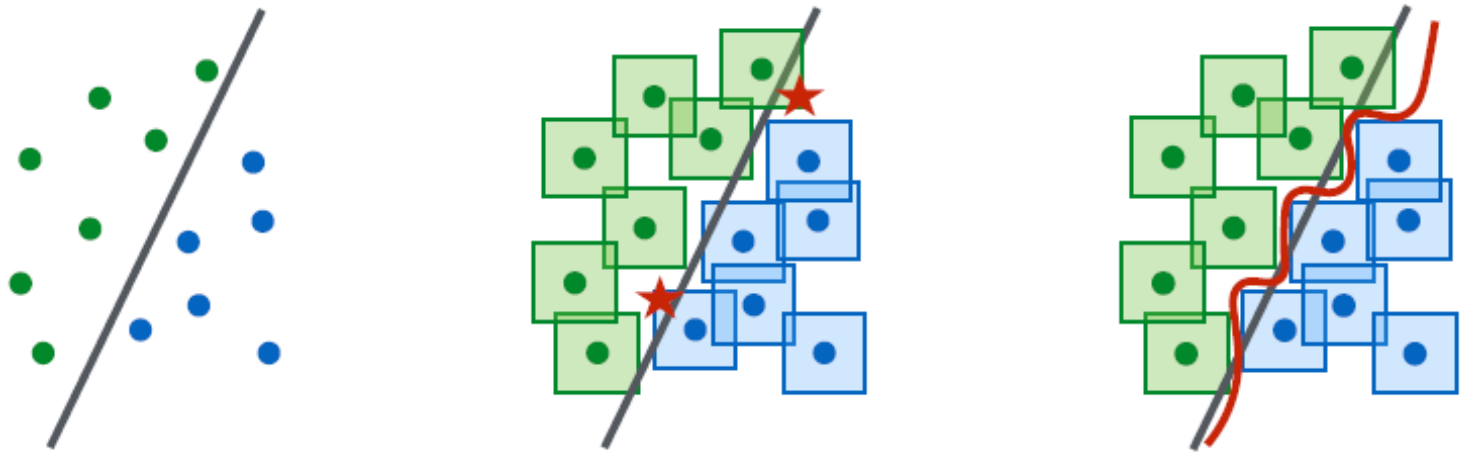


Figure 3: A conceptual illustration of standard vs. adversarial decision boundaries. Left: A set of points that can be easily separated with a simple (in this case, linear) decision boundary. Middle: The simple decision boundary does not separate the ℓ_∞-balls (here, squares) around the data points. Hence there are adversarial examples (the red stars) that will be misclassified. Right: Separating the ℓ_∞-balls requires a significantly more complicated decision boundary. The resulting classifier is robust to adversarial examples with bounded ℓ_∞-norm perturbations.

Some Practical Consideration

- Adversarial training says: Minimize the worst case scenario
- To do so, you need to simulate these worst case scenarios. That is, compute this:

$$\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\max_{\delta\in\mathcal{S}} L(\theta, \boldsymbol{x}+\delta, \boldsymbol{y})\right]$$

- This requires drawing **a lot of** δ from $\mathcal{S}$ so that you can compute the average
- For example, given a dog image you need perturb the image many times in order to compute the average of this worst case scenario
- If the original training dataset is large, e.g., ImageNet (1.28M images), a 10-fold adversarial training (i.e., for each image I generate 10 adversarial examples) will cause 10 times more training data
- That's why Madry et al. only show results on MNIST and CIFAR10.
- Scalability is very challenging.

Variations

Wang et al. (2019), Detecting Photoshopped Faces by Scripting Photoshop, `https://arxiv.org/pdf/1906.05856.pdf`

- Detect subtle / un-noticeable changes
- Idea: Adversarial train a classifier by synthetically creating perturbations in Photoshop.

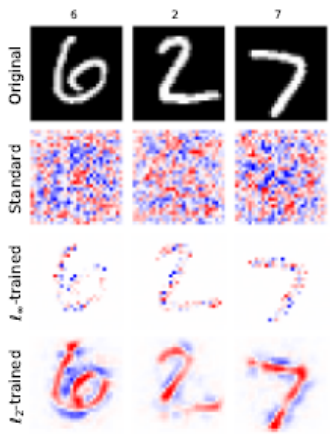
(a) Manipulated photo (b) Detected manipulations (c) Suggested "undo" (d) Original photo

Figure 1: Given an input face (a), our tool can detect that the face has been warped with the Face-Aware Liquify tool from Photoshop, predict where the face has been warped (b), and attempt to "undo" the warp (c) and recover the original image (d).

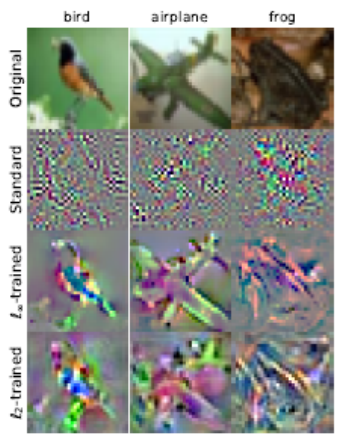
Some Additional "Benefits" of Adversarial Training

Here are some finding by Tsipras et al. Robustness may be at odds with accuracy, arXiv: 1805.12152

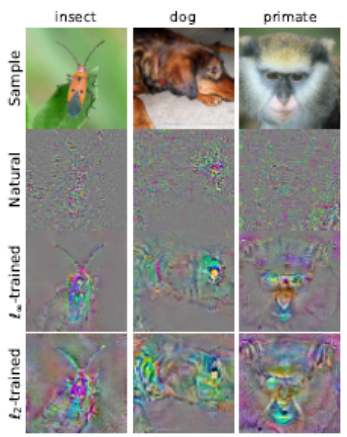
- **Loss gradients in the input space align well with human perception**.
- You train using standard protocol, compared to adversarial training.
- Look at the gradient of the loss with respect to the input.
- Adversarial trained is more "meaningful".



(a) MNIST



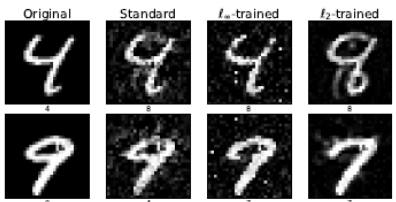
(b) CIFAR-10



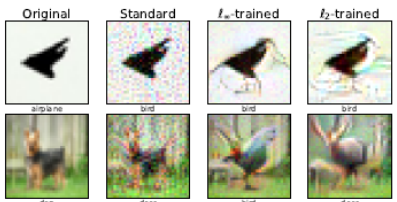
(c) Restricted ImageNet

Some Additional "Benefits" of Adversarial Training

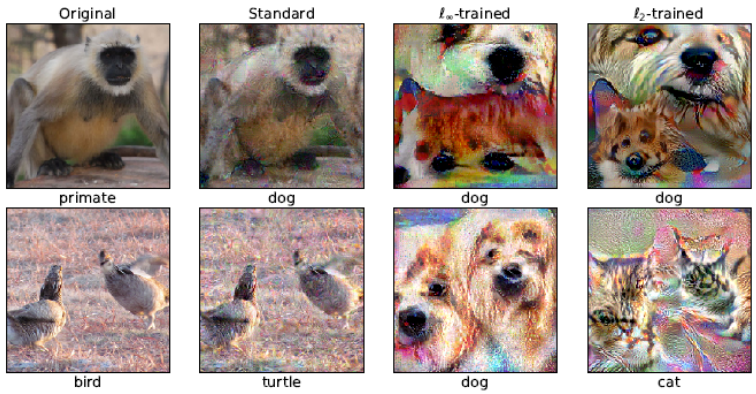
- **Adversarial examples exhibit salient data characteristics**.
- You look at the adversarial examples.
- E.g., if you want to attack "9" to "7", the adversarial example actually looks like "7".
- This is observed in adversarial trained models, but not in standard models



(a) MNIST



(b) CIFAR-10



(c) Restricted ImageNet

Outline

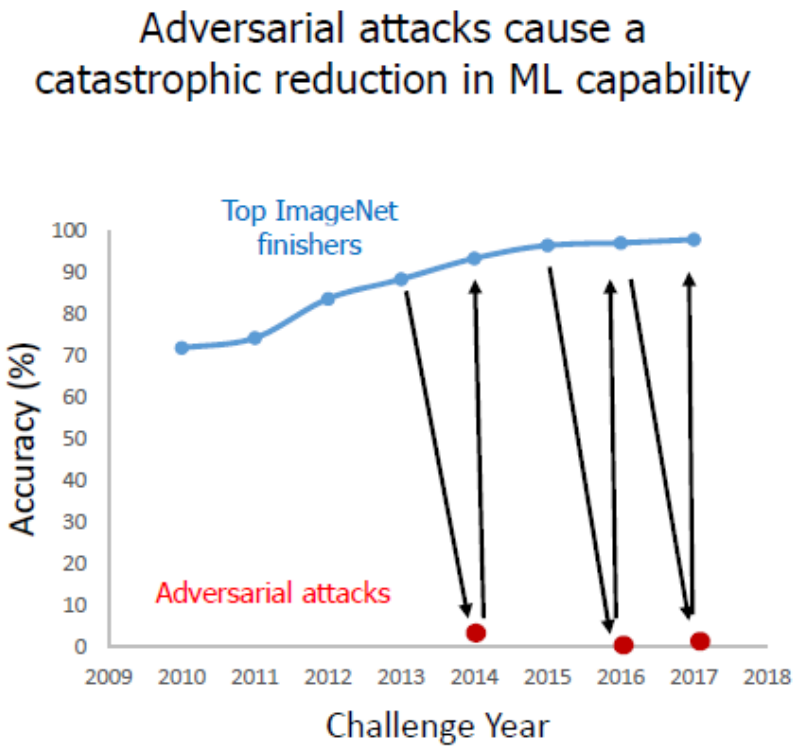
- Lecture 33-35 Adversarial Attack Strategies
- Lecture 36 Adversarial Defense Strategies
- Lecture 37 Trade-off between Accuracy and Robustness

Today's Lecture

- Three classes of defenses
  - Pre-processing
  - Adversarial training
  - Randomness
- Limitation of current defense (and also attacks)
  - Obfuscated gradient
  - Physical attack?
  - Real attacks in social media today

This is an endless game

The following picture is taken from DARPA's GARD program.



Obfuscated gradient

Athalye et al. Obfuscated Gradients Give a False Sense of Security, `https://arxiv.org/abs/1802.00420`

- Best paper, ICML 2018
- Claimed successful attack of 7 out of 9 white-box defense in ICLR 2018
- The only that survives is adversarial training
- Idea: A lot of defenses are based on **gradient masking** or **obfuscated gradient**
- That is, "hide away the gradient" or "destroy the gradient" so that gradient based attackers will fail
- This paper specifically listed three types of obfuscated gradients
  - Shattered gradients: nonexistent or incorrect gradients created by the defense
  - Stochastic gradients: due to randomization schemes
  - Vanishing gradients: due to propagation through deep networks
- To attack, one just need to find a way to **approximate** the gradient

Illustration

Attacking **Input Transform**:

- The secured classifier is $\widehat{f}(x) = f(g(x))$. g is the input transform
- Because g is a "denoiser", we roughly want $g(x) \approx x$ (Attacks usually won't perturb the image too much, and so the effect of the denoiser should be mild.)
- Therefore, $\nabla_x g(x) \approx 1$
- And so we can approximate the gradient of $\widehat{f}$ at $x = \widehat{x}$ by $\nabla_x \widehat{f}(x)|_{x=\widehat{x}} \approx \nabla f(x)|_{x=g(\widehat{x})}$.

Attacking **Randomization**:

- You have a distribution of the transformations $t \sim T$
- While you do not know the specific case during defense, you can attack the average
- That is, you can attack $\mathbb{E}_{t\sim T} f(t(x))$
- You can approximate the gradient as $\nabla \mathbb{E}_{t\sim T} f(t(x)) = \mathbb{E}_{t\sim T} \nabla f(t(x))$
- Differentiate through f and t, and approximate the expectation with samples at each gradient descent step

Physical Attacks

- Can adversarial attacks really attack systems in real **physical space**?
- Probably, but your attack has to be strong. But if it is strong, then a low-level image processing algorithm can detect
- And remember, you never know the model parameters.
- The real challenge is the physical environment
- E.g., shadow, perspective change, angle, blur, noise, etc
- I have **never** seen any physical attack that works beyond in the reported scenarios
- E.g., the stop sign example will not work if you attack on a rainy day (because your attack is trained on a sunny day)
- In my opinion, it is much more meaningful to solve the **transfer learning** problem than trying to attack / defend
- On this earth, the only parties that really care about attack are social media platforms and search engines
- What is the economic value of painting a stop sign in order to fool an auto-car?

Most attacks are not adversarial attack

- Adversarial attacks are **not common**, at least on social media today.
- People are not as smart as you. They probably do not even know what FGSM is.
- The way they attack is: Trial-and-error, **repeat** until it goes through the screening system.
- E.g., add strips to a policy-violating image, blur the background, add patches, etc.
- **Very creative**, because there is economic value once the image shows up as an advertisement.
- *Large in volume, huge in variety, new every day.*
- You are welcome to do adversarial training or whatever you have seen in the literature. But I promise **nothing will work**. (Really? These are simple attacks. Yes, but they are **rich**!)
- Social media have a large team of **human** content reviewers to catch policy violating images.
- Open problem. Our lab is working on it.

Take Home

- The obfuscated gradient paper by Athalye et al. concluded that only **adversarial training** can withstand their attack
- All methods reported then have been defeated
- So if you want to propose something new, you need to withstand their attack
- Adversarial attack and defense is still very hot, as of today
- There are also work saying that by **pruning network parameters** you can gain robustness, and stuff along this line. (That's just a form of **regularization**.)
- My personal view: Adversarial attack is largely an academic **toy**.
- The more **meaningful** tasks (under the same umbrella of adversarial attack) are
  - transfer learning
  - very low-light classification
  - recognition through random medium
- Our lab has been doing work in the latter two problems. We can chat if you are interested.