Course number: 80240743

Deep Learning

Xiaolin Hu (胡晓林) & Jun Zhu (朱军)

Dept. of Computer Science and Technology

Tsinghua University

Last lecture review

1. Typical tasks

Tagging and parsing

Text generation

Text/document classification

Sentiment analysis

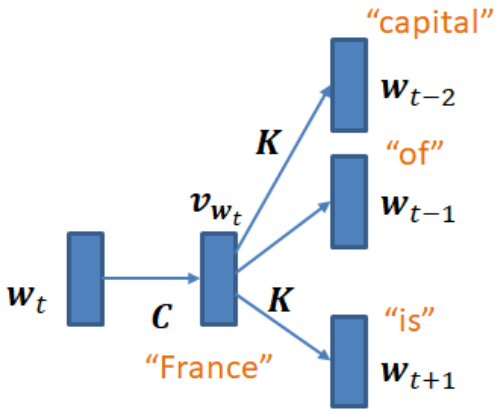
Machine translation

Question answering

2. Word representation

counts	I	like	deep	learning	NLP	enjoy	flying
I	0	2	0	0	0	1	0
like	2	0	1	0	1	0	0
deep	0	1	0	1	0	0	0
learning	0	0	1	0	0	0	0
NLP	0	1	0	0	0	0	0
enjoy	1	0	0	0	0	0	1
flying	0	0	0	0	0	1	0

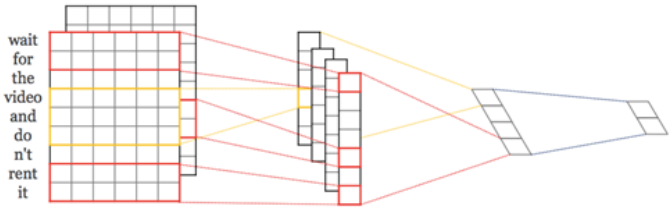
Co-occurrence matrix

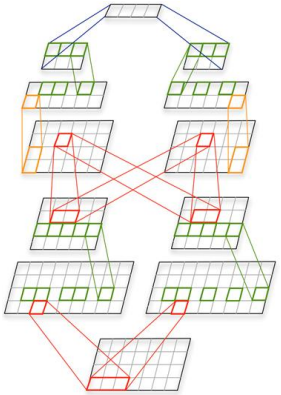


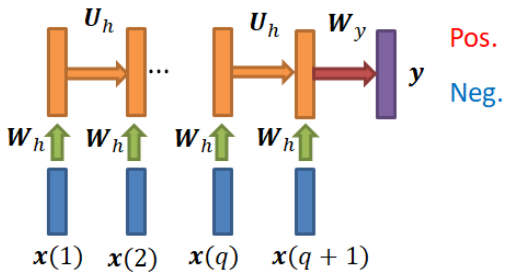
Dense vectors

Last lecture review

3. Text classification

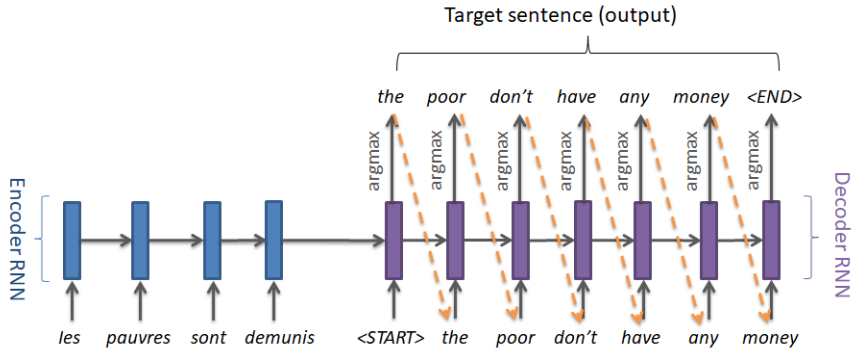




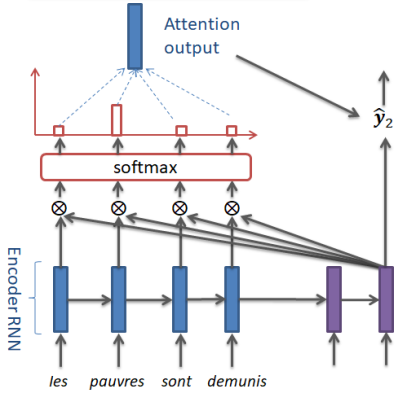


4. Machine translation

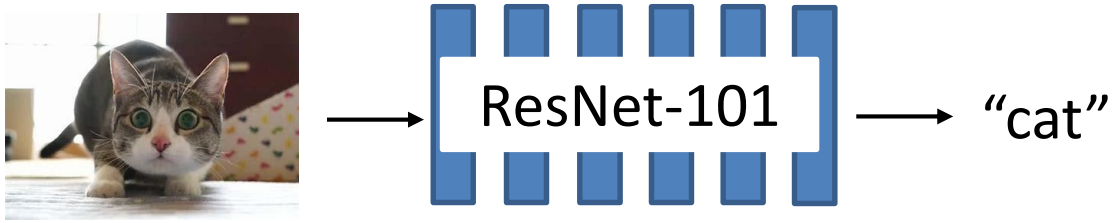
Encoder-decoder

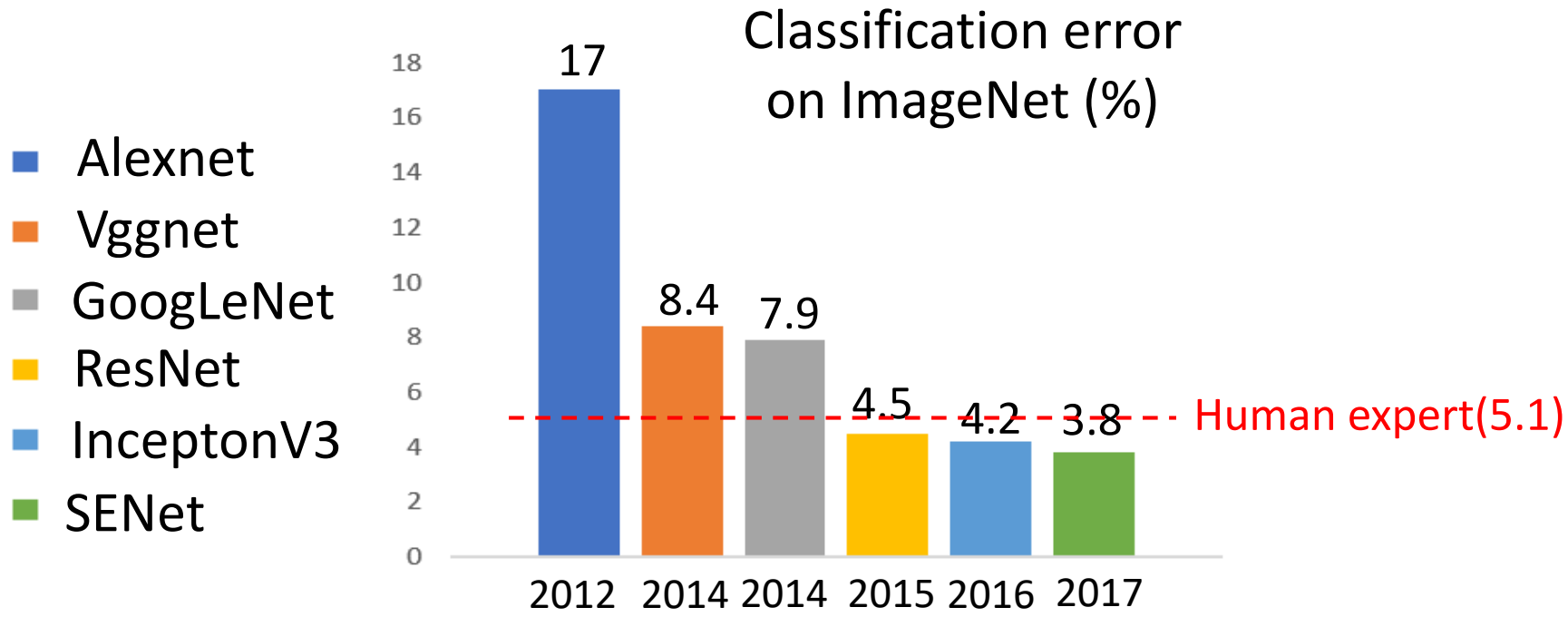


Attention



Neural networks are powerful





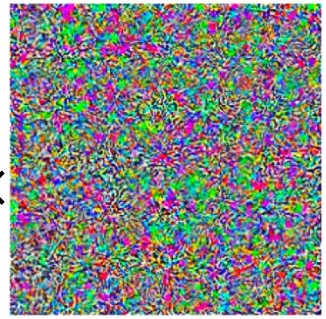
Experiment


+ .008×

=


ResNet: Kangaroo: 99.31%

Airplane: 99.99%


+ .008×

=


ResNet: Bradypod: 99.33%

Bulletproof: 100%

Adversarial attack: deliberately perturb the input and make the neural network give wrong output

Lecture 9: Adversarial Attack and Defense

Xiaolin Hu
Dept. of Computer Science and Technology
Tsinghua University

Outline

1. Introduction
2. Adversarial attacks
3. Adversarial defenses
4. Attacks in the physical world*
5. Summary

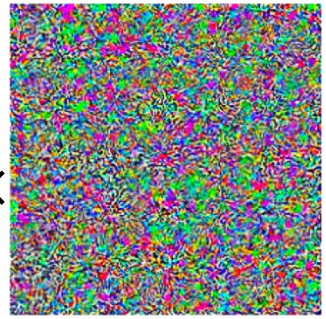
Concepts


+ .008×

=


ResNet: Kangaroo: 99.31%

Airplane: 99.99%


+ .008×

=


ResNet: Bradypod: 99.33%

Bulletproof: 100%

Normal example **Adversarial noise** **Adversarial example**

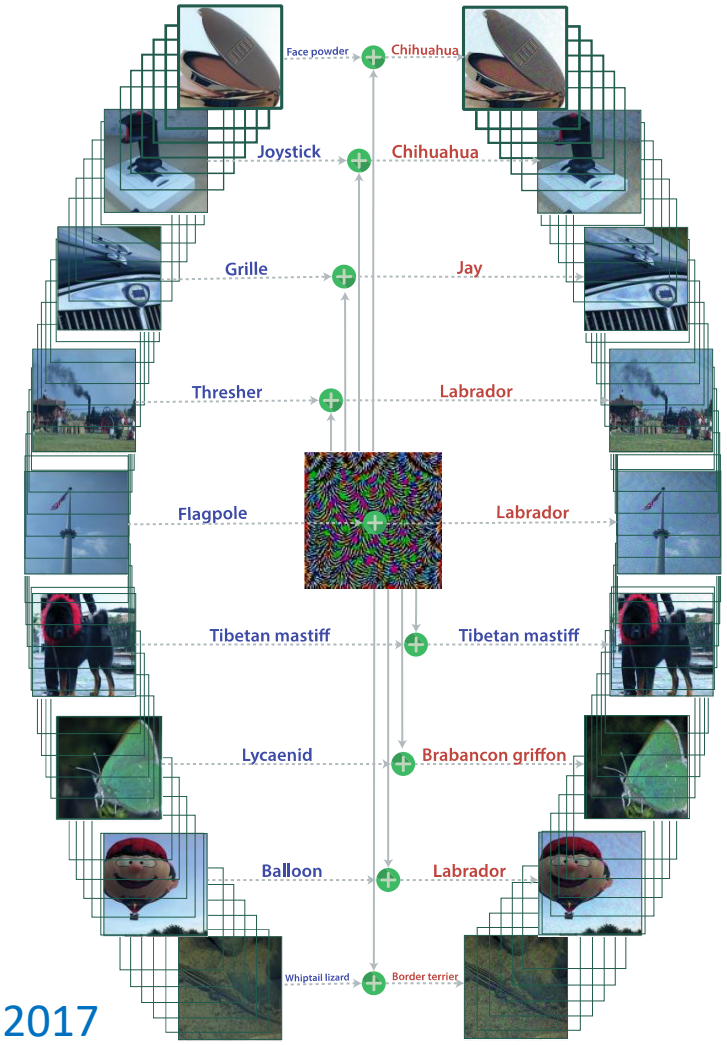
Transferability of adversarial examples

- Cross-model transferability

ResNet → “airplane”

Inc V3 → “airplane”

- Cross-data transferability



Moosavi-Dezfooli, Fawzi, CVPR 2017

Notations and definitions

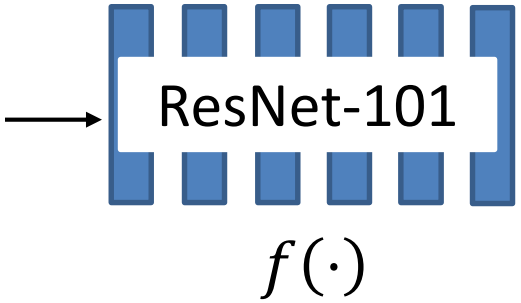
Normal example x

+

Adv noise δ

=

Adv example x^*



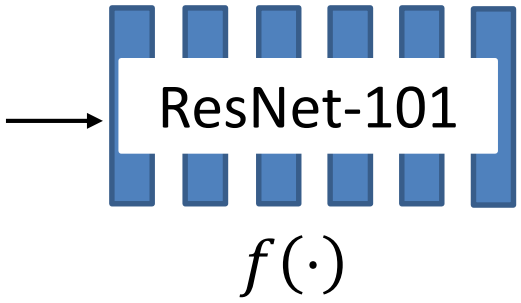
$f(\cdot)$

Label y =“kangaroo”

$f(x)$ =“kangaroo”

Bingo!

$||\delta||_p \leq \epsilon$, ϵ is a small constant



$f(\cdot)$

Untargeted attack

$f(x^*) \neq$“kangaroo”

$f(x^*) = y^* \triangleq$“airplane”

Targeted attack

Information needed for attack

- White-box attack
  - The model is known. Quite easy. You'll see why.

normal example

ResNet101

"Kangaroo"

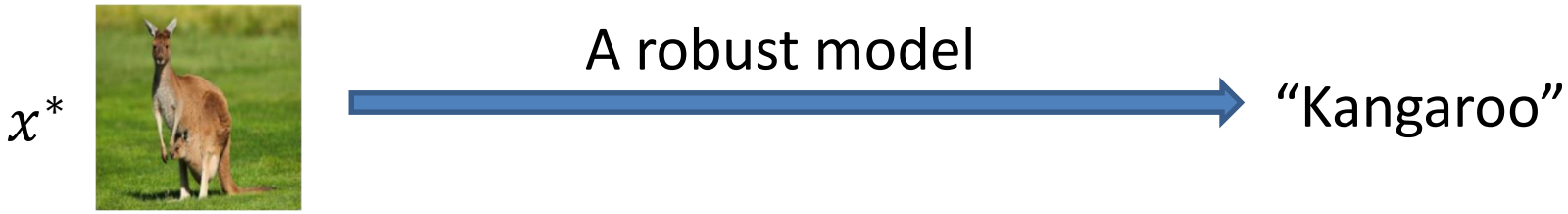
- Black-box attack:
  - The model is unknown
  - Difficult
  - Two typical strategies:
    - Utilize the transferability of adversarial examples
    - Construct a surrogate model and attack it

Two strategies for defense

① Let the model map x^* to the correct label y

– Protect an existing model

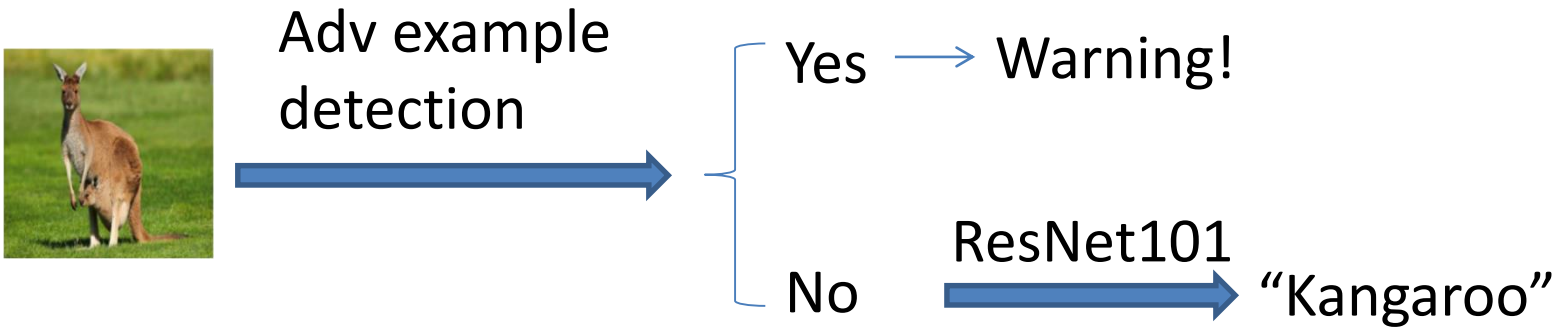
– Devise a model that is robust to adversarial noise



– Combine the above two

Two strategies for defense

② Detect x^* as an adversarial example and refuse to make a decision



Multiple Choice(single) Points: 1

Setting

If the same adversarial noise is added to a “dog” image and a “cat” image, and a VGGNet recognize both as “airplane”, we say this adversarial noise has

A transferability across models

B transferability across data

Submit

Multiple Choice(multiple) Points: 1

Setting

If you add some small noise to a picture of “dog” and want a ResNet pretained on ImageNet to recognize it as “airplane”, this attack is called

A Untargeted attack

B Targeted attack

C White-box attack

D Black-box attack

Submit

Multiple Choice(multiple) Points: 1

Setting

If you add some small noise to a piece of your speech and you want a smart speaker to recognize it as my voice, most probably this attack is

A Untargeted attack

B Targeted attack

C White-box attack

D Black-box attack

Submit

Multiple Choice(single) Points: 1

Setting

In general, which attack is easier?

A Black-box attack

B White-box attack

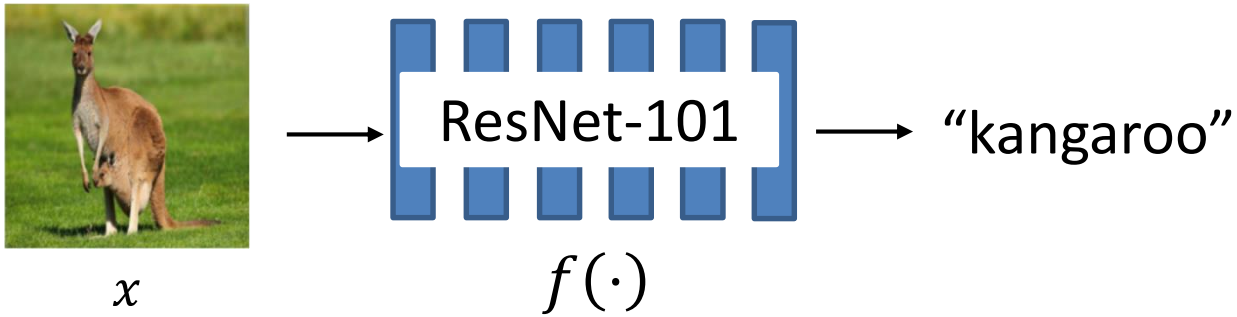
Submit

Outline

1. Introduction
2. Adversarial attacks
   - White-box attack
   - Black-box attack
3. Adversarial defenses
4. Attacks in the physical world*
5. Summary

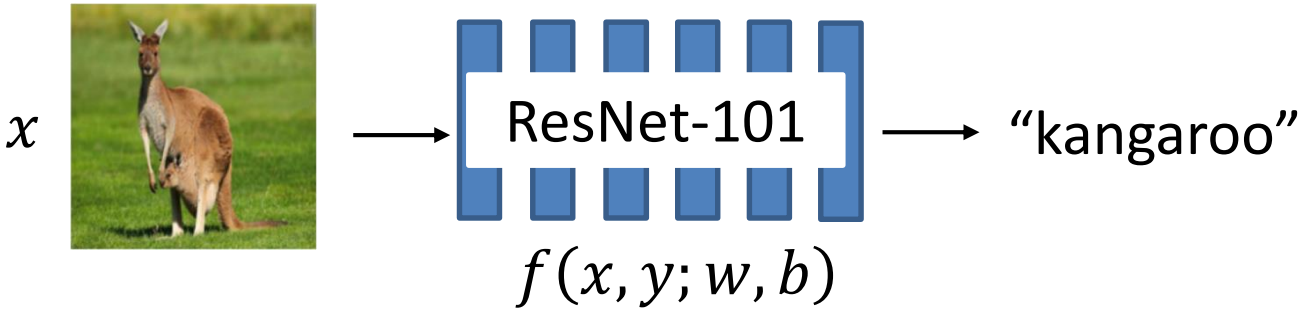
Motivation for white-box attack methods

- Both the structure and parameters of the model f are known



- Adjust the input x iteratively such that $f(x^*)$ is not equal to y
- A naïve way is to randomly change the pixels in x
  - Quite inefficient, if not infeasible
- An objective function $L(x)$ is needed to guide the adjustment of x such that
  - when L attains its (local) optimum, the output of the model is not y

Recall: neural network training



- Learning a neural network amounts to minimizing a loss function L w.r.t. the weights and biases

$$\min_{w,b} L(x, y; w, b)$$

When the minimum of L is attained, $f(x) = y$

where (x, y) denotes the input and desired output pair

- The loss function is often the cross entropy + regularizer

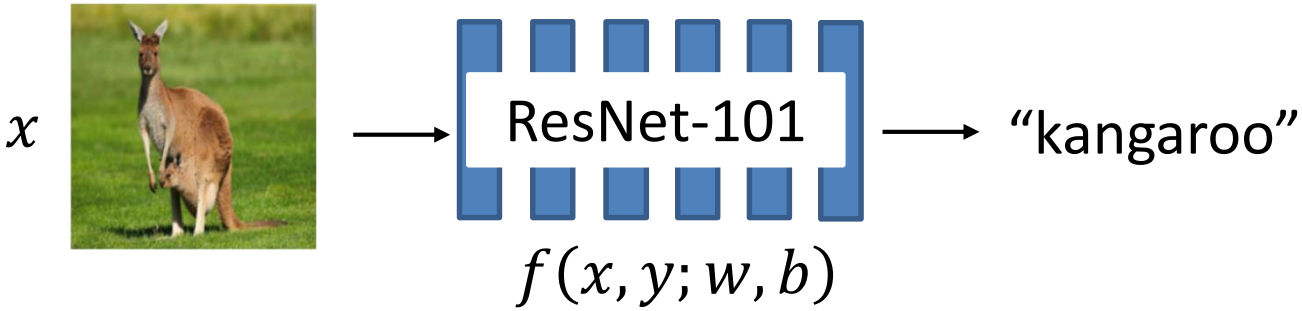
$$L = -\langle \sum_k y_k \ln \psi_k^{(L)} \rangle + \frac{\lambda}{2} \sum_{i,j,l} (w_{ji}^{(l)})^2$$

$$\psi_k^{(L)} = \mathrm{softmax}(w_k^{(L)\top} \psi^{(L-1)} + b_k^{(L)})$$

- $\langle \cdot \rangle$ means average over samples, $\lambda > 0$

- k, l index the classes and layers, respectively
- L is the last layer
- y is the ground-truth label
- $\psi^{(l)}$ is the output of layer l

Motivation



Training

Minimize a loss function

$$\min_{\theta} L(x, y; \theta)$$

When L attains the minimum, $f(x) = y$

Motivate

To satisfy $f(x^*) \neq y$ we should make $L(x^*, y; \theta)$ as large as possible!

Principles for attack

Training a NN

$$\min_{\theta} L(x, y; \theta)$$

When L attains the minimum, $f(x) = y$

GAN

Adv example

Ian Goodfellow

Reverse thinking
Challenge the tradition

Untargeted attac

$$\max_{x^*} L(x^*, y; \theta)$$

$$\text{subject to } \left|\left|x - x^*\right|\right|_p \leq \epsilon$$

When L attains the maximum, $f(x^*) \neq y$

$$\min_{x^*} L(x^*, y^*; \theta)$$

$$\text{subject to } \left|\left|x - x^*\right|\right|_p \leq \epsilon$$

When L attains the minimum, $f(x^*) = y^*$

Goodfellow, Shlens, Szegedy, ICLR 2015

Gradient sign-based methods

- Usually, solving an optimization problem needs to use the gradient of the objective function
  - we'll see such methods later
- It's interesting that only using the sign of the gradient can yield good results
  - ① FGSM
  - ② I-FGSM
  - ③ MI-FGSM

In what follows, we use integers (0-255) to represent pixel values in x

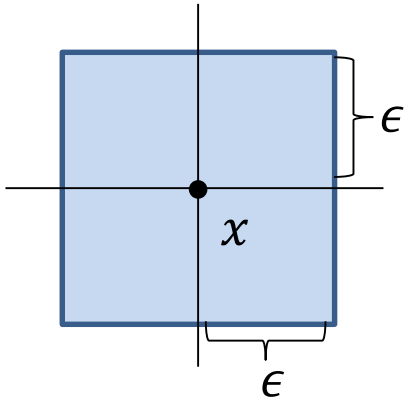
- It suggests that adversarial examples are easy to obtain

Fast gradient sign method (FGSM)

Goodfellow, Shlens, Szegedy, ICLR 2015

- Let $p = \infty$, the problem

$$\max_{x^*} L(x^*, y; w, b)$$
$$\text{subject to } \left|\left|x - x^*\right|\right|_\infty \leq \epsilon$$

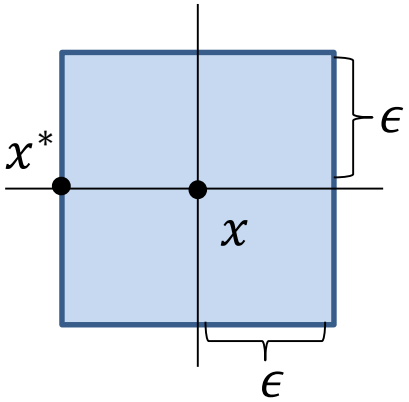


- Obtain the solution ~~in one step~~

$$x^* = x + \epsilon \cdot \text{sign}(\nabla_x L(x, y))$$

Adversarial noise

  - Note that the constraint is ~~automatically~~ satisfied!



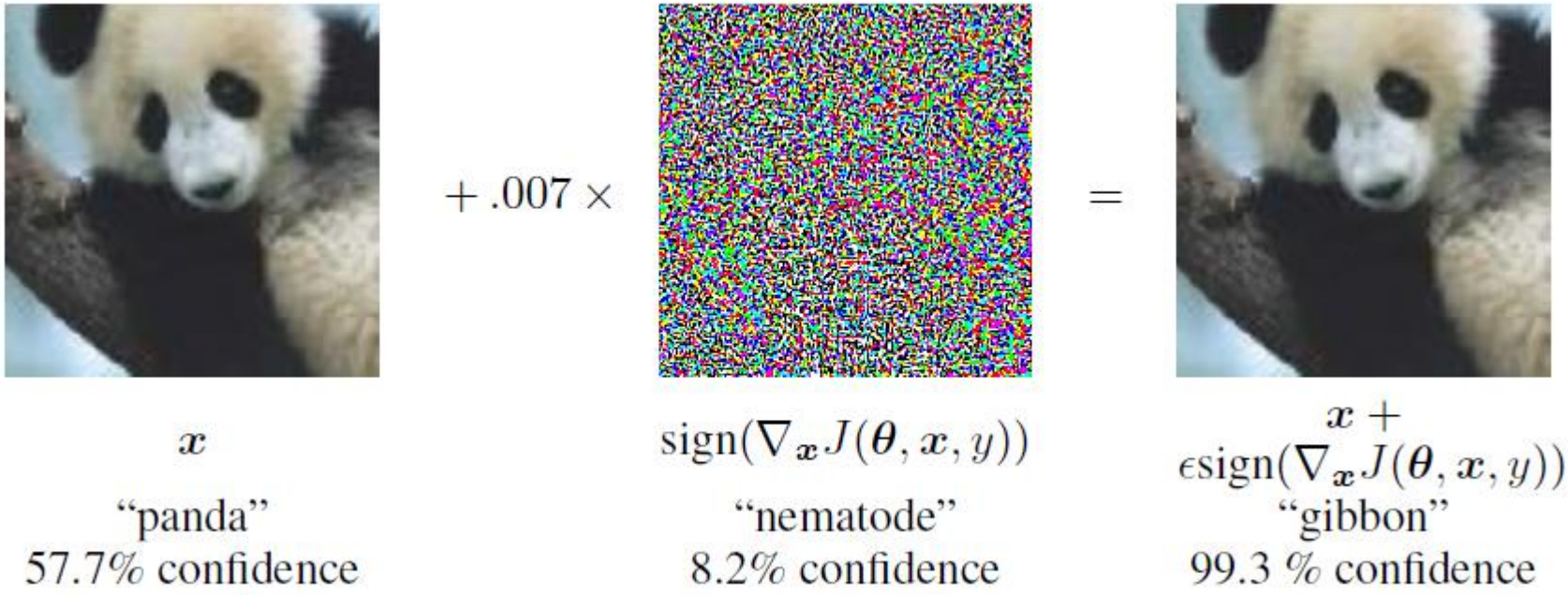
- Advantage: fast
- Disadvantage: poor performance on white-box attack

How to perform targeted attack?

Results

Goodfellow, Shlens, Szegedy, ICLR 2015

Attack the GoogLeNet (Szegedy et al., 2014) which was trained on ImageNet classification dataset



Iterative FGSM (I-FGSM)

Kurakin, Goodfellow, Bengio, ICLR 2017 workshop

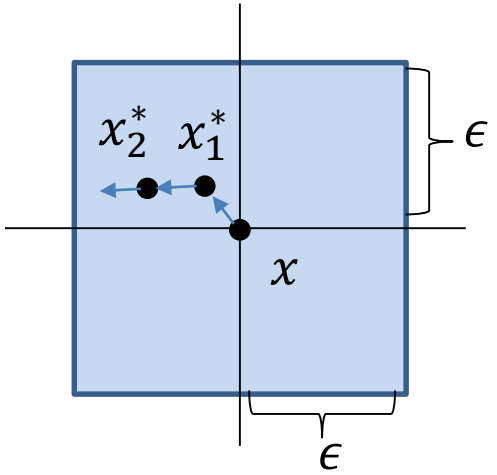
- Again let $p = \infty$, the problem

$$\max_{x^*} L(x^*, y; w, b)$$
$$\text{subject to } \left|\left|x - x^*\right|\right|_\infty \leq \epsilon$$

- Obtain the solution iteratively

$$x_0^* = x, \quad x_{t+1}^* = x_t^* + \alpha \cdot \text{sign}\left(\nabla_x L(x_t^*, y)\right)$$

① To meet the constraint we can set $\alpha = \epsilon/T$ where T is the number of iterations
  - A good choice of T is about 10-20

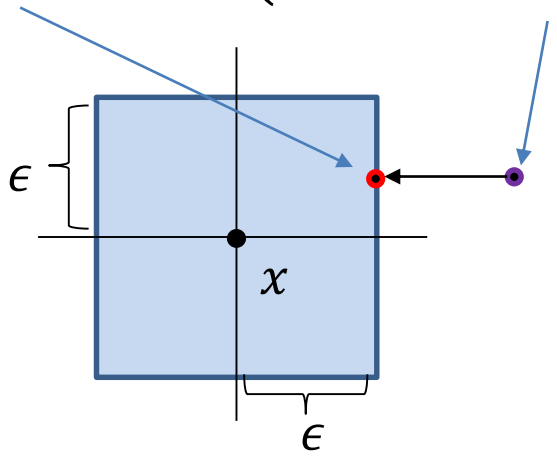


Iterative FGSM (I-FGSM)

Kurakin, Goodfellow, Bengio, ICLR 2017 workshop

② Clip the solution in the feasible region

$$x_0^* = x, \quad x_{t+1}^* = \text{clip}\left(x_t^* + \alpha \cdot \text{sign}\left(\nabla_x L(x_t^*, y)\right)\right)$$



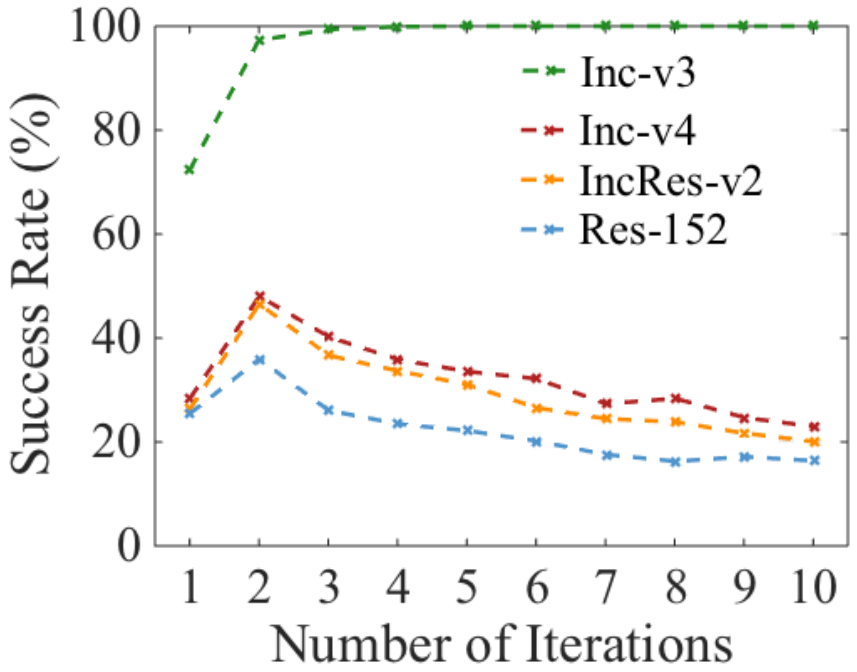
- Also known as projected gradient descent (PGD)

Advantage: good performance on white-box attack
Disadvantage: a little bit slower, and more importantly…

Results

Experiment setting

- $\epsilon = 16$; Select 1000 images from ImageNet
- Attack Inception V3
- Test the attack effect
  - Inception V3
  - Inception V4
  - Inception ResNet V2
  - ResNet-152

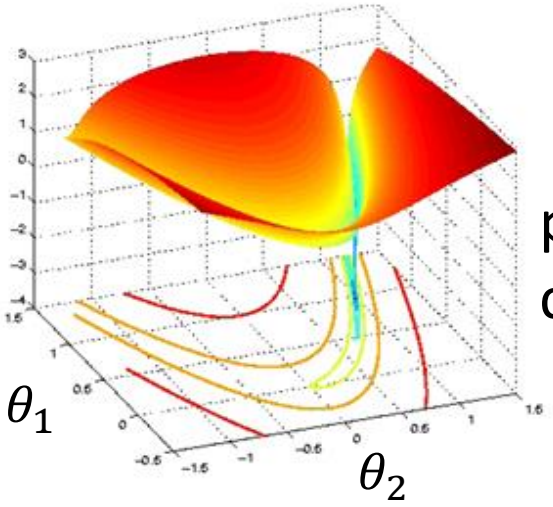


Bad transferability

Failed to find a good enough solution!

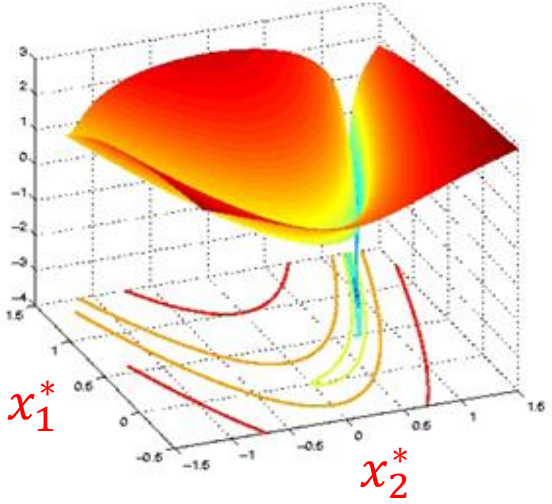
Recall: momentum technique

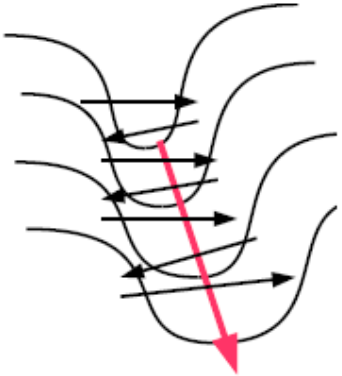
Loss fun $L(x, y; \theta)$

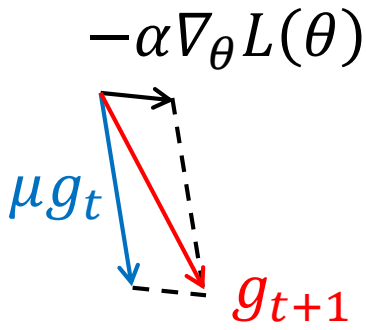


pathological curvature

Loss fun $L(x^*, y; \theta)$







- Momentum SGD

$$g_{t+1} = \mu g_t - \alpha \nabla_\theta L(\theta_t)$$
$$\theta_{t+1} = \theta_t + g_{t+1}$$

  - g_t is the momentum, μ and α are hyperparameters

Momentum I-FGSM (MI-FGSM)

Dong et al., CVPR 2018

NN training

Untargeted attack

SGD:

$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta L(\theta_t)$$

I-FGSM:

$$x^*_{t+1} = \text{clip}\left(x^*_t + \alpha \cdot \text{sign}\left(\nabla_x L(x^*_t)\right)\right)$$

Momentum SGD:

$$g_{t+1} = \mu g_t - \alpha \nabla_\theta L(\theta_t)$$

$$\theta_{t+1} = \theta_t + g_{t+1}$$

momentum

MI-FGSM:

normalize

$$g_{t+1} = \mu \cdot g_t + \frac{\nabla_x L(x^*_t)}{\|\nabla_x L(x^*_t)\|_1}$$

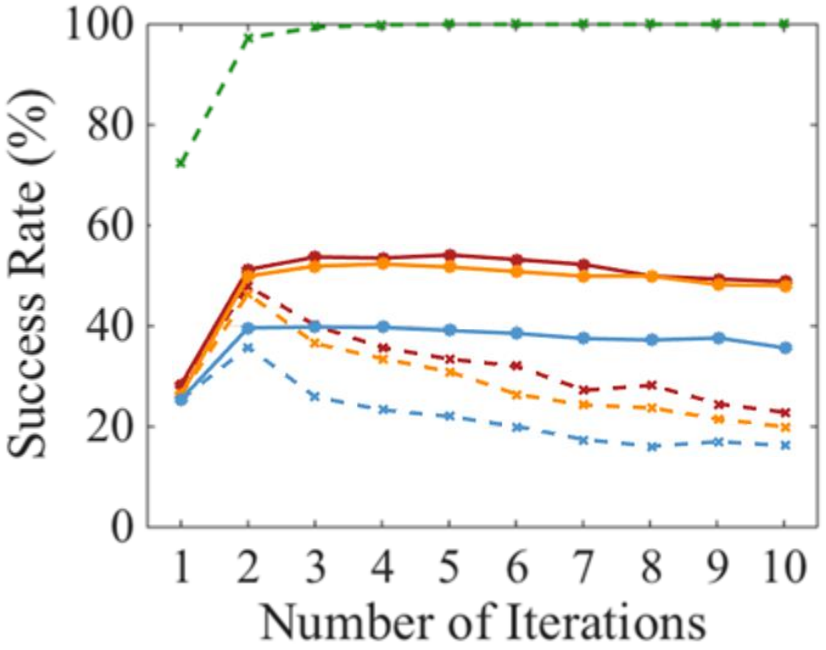
$$x^*_{t+1} = \text{clip}\left(x^*_t + \alpha \cdot \text{sign}(g_{t+1})\right)$$

Analogy

Results

Experiment setting

- $\epsilon = 16$; Select 1000 images from ImageNet
- Attack Inception V3
- Test the attack effect on
  - Inception V3
  - Inception V4
  - Inception ResNet V2
  - ResNet-152



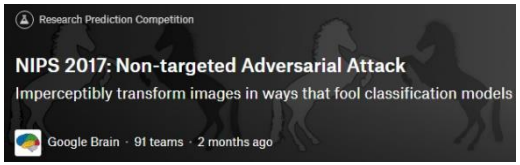
Dashed: w/o momentum

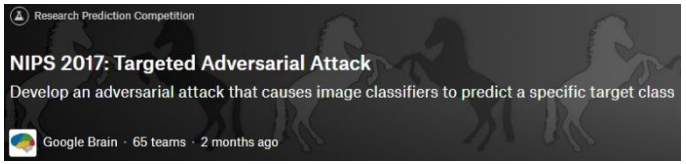
Continuous: w/ momentum

Transferability is improved

Performance of MI-FGSM

NeurIPS2017
Adv example
competition





Two champions

Organized by Google, ~100 teams including Stanford

Integrated into
several adv attack &
defense platforms

CleverHans	Foolbox
cleverhans	EBERHARD KARLS UNIVERSITÄT TÜBINGEN
Google/OpenAI/Penn State U	U of Tubingen

Exact gradient-based methods

- FGSM, I-FGSM, MI-FGSM are all based on the sign of the gradient, which is an approximation of the gradient
- A more natural idea is to use exact gradient-based methods
  - May find better solutions to the objective function and thus more powerful adversarial examples
  - Much slower than previous methods
- A direct extension is to remove the sign in previous methods

For untargeted attack

- Suppose $p = \infty$, the problem becomes

$$\max_{x^*} L(x^*, y; w, b)$$
$$\text{subject to } \left|\left|x - x^*\right|\right|_\infty \leq \epsilon$$

- Remove sign in I-FGSM

$$x_0^* = x, \quad x_{t+1}^* = \text{clip}\left(x_t^* + \alpha \cdot \nabla_x L(x_t^*, y)\right)$$

- Remove sign in MI-FGSM

$$x_0^* = x, g_0 = 0$$
$$g_{t+1} = \mu \cdot g_t + \frac{\nabla_x L(x_t^*, y)}{\|\nabla_x L(x_t^*, y)\|_1}$$
$$x_{t+1}^* = \text{clip}(x_t^* + \alpha \cdot g_{t+1})$$

How to perform targeted attacks?

For targeted attack

- Suppose $p = \infty$, the problem becomes

$$\min_{x^*} L(x^*, y^*; w, b)$$
$$\text{subject to } \left|\left|x - x^*\right|\right|_\infty \leq \epsilon$$

- Remove sign in I-FGSM

$$x_0^* = x, \quad x_{t+1}^* = \text{clip}\left(x_t^* - \alpha \cdot \nabla_x L(x_t^*, y^*)\right)$$

- Remove sign in MI-FGSM

$$x_0^* = x, g_0 = 0$$
$$g_{t+1} = \mu \cdot g_t - \frac{\nabla_x L(x_t^*, y^*)}{\|\nabla_x L(x_t^*, y^*)\|_1}$$
$$x_{t+1}^* = \text{clip}(x_t^* + \alpha \cdot g_{t+1})$$

Algorithms for general p-norm

Untargeted attack

$$\max_{x^*} L(x^*, y; w, b)$$
$$\text{subject to } \left|\left|x - x^*\right|\right|_p \leq \epsilon$$

Targeted attack

$$\min_{x^*} L(x^*, y^*; w, b)$$
$$\text{subject to } \left|\left|x - x^*\right|\right|_p \leq \epsilon$$

- Transform the problem into box-constrained optimization problems, and use existing optimization algorithms to solve

$$\min_{x^*} \lambda\left|\left|x - x^*\right|\right|_p - L(x^*, y; w, b)$$
$$\text{subject to } x^* \in [0,1]^n$$

$$\min_{x^*} \lambda\left|\left|x - x^*\right|\right|_p + L(x^*, y^*; w, b)$$
$$\text{subject to } x^* \in [0,1]^n$$

We now use decimals to denote pixel values

Another formulation for targeted attack

Carlini and Wagner, IEEE SSP 2017

- The previous formulation emphasizes the success of attack, e.g., $f(x^*) = y^*$ and the perturbation may not be the smallest
- We can also emphasize the small perturbation

$$\min_{x^*} \left|\left|x - x^*\right|\right|_p$$
$$\text{subject to } \phi(x^*) \leq 0, x^* \in [0,1]^n$$

  where $\phi(x^*) \leq 0$ if and only if $f(x^*) = y^*$
- It's equivalent to

$$\min_{x^*} \left|\left|x - x^*\right|\right|_p + c \cdot \phi(x^*)$$
$$\text{subject to } x^* \in [0,1]^n$$

  with an appropriate c.
  - It's suggested to use the smallest c such that $\phi(x^*) \leq 0$

Another formulation for targeted attack

Carlini and Wagner, IEEE SSP 2017

$$\min_{x^*} \left|\left|x - x^*\right|\right|_p + c \cdot \phi(x^*)$$
$$\text{subject to } \; x^* \in [0,1]^n$$

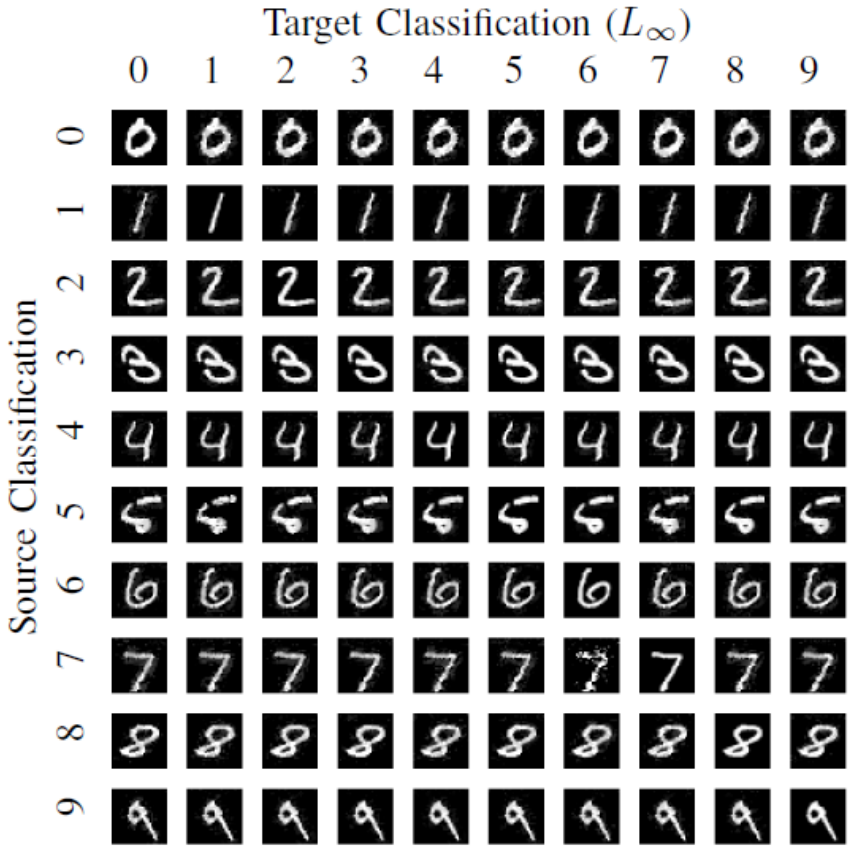
- Several ϕ has been tested and the following is said to be good

$$\phi(x^*) = \left(\max_{i \neq y^*}(Z(x^*)_i) - Z(x^*)_{y^*}\right)^+$$

  - $(a)^+ = \max(0, \mathrm{a})$ and
  - $Z(x^*)$ is the input to softmax, i.e., $W^{(L)}\psi^{(L-1)} + b^{(L)}$
  - Z is often called logit
  - This $\max(0, a)$ formulation allows in a range the attack is always successful and you can choose the smallest δ in this range
- This method is often called CW method, following the names of the two authors

Results

- Due to the formulation, the method can have very small perturbation



- L1 adversary applied to the MNIST dataset performing a targeted attack for every source/target pair
- Each digit is the first image in the dataset with that label

Black-box attack

- Adversarial example has some transferability across models

ResNet → "airplane"

Inc V3 → "airplane"

- However, this transferability is not very strong
- Motivation: if the adversarial examples have strong transferability among many existing models, then they are likely to attack an unknown model successfully
- Idea: craft adversarial examples that can fool many existing models

Ensemble attack

- Purpose: attack an ensemble of models
- Notations: let Z_i, Ψ_i and L_i denote the logit, softmax output and the cross-entropy loss of model i, i.e.,

$$Z(x) = W^{(L)}\nu^{(L-1)} + b^{(L)}$$

$$\Psi(x) = \text{softmax}(Z(x))$$

$$L(x, y) = -1_y \cdot \log \Psi(x)$$

  where $\nu^{(L-1)}$ denotes the output of layer $(L-1)$ and y denotes the ground-truth label of x

- Question: how do you do that?

Ensemble attack

- Let's introduce a set of weights λ_i which satisfy $0 < \lambda_i < 1$ and $\sum_i \lambda_i = 1$
- We define the weighted average of the
  - ① cross-entropy losses: $\bar{L}(x, y) = \sum_i \lambda_i L_i(x, y)$
  - ② softmax outputs (predictions): $\bar{\Psi}(x) = \sum_i \lambda_i \Psi_i(x)$
  - ③ logits: $\bar{Z}(x) = \sum_i \lambda_i Z_i(x)$
- We minimize the following losses (for untargeted attack), respectively
  - ① $\min_{x^*} -\bar{L}(x^*, y)$
  - ② $\min_{x^*} 1_y \cdot \log \bar{\Psi}(x^*)$, e.g., in (Liu, Chen, Liu, Song, ICLR 2018)
  - ③ $\min_{x^*} 1_y \cdot \log\left(\text{softmax}\left(\bar{Z}(x^*)\right)\right)$, e.g., in (Dong, Su et al., CVPR 2018)
- Any optimization method introduced before can be applied

A case study

Dong, Su et al., CVPR 2018

Success rates (%) of non-targeted adversarial attacks

hold-out	Ensemble method	FGSM		I-FGSM		MI-FGSM	
		Ensemble	Hold-out	Ensemble	Hold-out	Ensemble	Hold-out
-Inc-v3	③ Logits	**55.7**	**45.7**	**99.7**	**72.1**	**99.6**	**87.9**
	② Predictions	52.3	42.7	95.1	62.7	97.1	83.3
	① Loss	50.5	42.2	93.8	63.1	97.0	81.9
-Inc-v4	Logits	**56.1**	**39.9**	**99.8**	**61.0**	**99.5**	**81.2**
	Predictions	50.9	36.5	95.5	52.4	97.1	77.4
	Loss	49.3	36.2	93.9	50.2	96.1	72.5
-IncRes-v2	Logits	**57.2**	**38.8**	**99.5**	**54.4**	**99.5**	**76.5**
	Predictions	52.1	35.8	97.1	46.9	98.0	73.9
	Loss	50.7	35.2	96.2	45.9	97.4	70.8
-Res-152	Logits	**53.5**	**35.9**	99.6	**43.5**	99.6	**69.6**
	Predictions	51.9	34.6	**99.9**	41.0	**99.8**	67.0
	Loss	50.4	34.1	98.2	40.1	98.8	65.2

- Four models: Inc-v3, Inc-v4, IncRes-v2 and Res-152
- Everytime, attack an ensemble of 3 white-box models and a hold-out black-box target model

Attack a surrogate model

- Steps:
  - Input many x to the unknown model and obtain its output y
  - Use these many (x, y) pairs to train a differentiable model S such that $S(x) \approx y$
  - Attack the model S
- Disadvantage: it requires a large number (often millions) of queries to the unknown model

Multiple Choice(multiple) Points: 1

Setting

Which attack algorithm(s) depend(s) on the sign of the gradient of the loss function instead of the exact gradient?

A FGSM

B I-FGSM

C MI-FGSM

D CW

Submit

Multiple Choice(single) Points: 1

Setting

For targeted attack, denote the adversarial example as x^*, the correct label as y, the target label as y^* and the CE loss as L. Which one is the updating rule of I-FGSM (α>0)?

A $x_{t+1}^* = \text{clip}\left(x_t^* + \alpha \cdot \text{sign}\left(\nabla_x L(x_t^*, y)\right)\right)$

B $x_{t+1}^* = \text{clip}\left(x_t^* - \alpha \cdot \text{sign}\left(\nabla_x L(x_t^*, y)\right)\right)$

C $x_{t+1}^* = \text{clip}\left(x_t^* + \alpha \cdot \text{sign}\left(\nabla_x L(x_t^*, y^*)\right)\right)$

D $x_{t+1}^* = \text{clip}\left(x_t^* - \alpha \cdot \text{sign}\left(\nabla_x L(x_t^*, y^*)\right)\right)$

Submit

Outline

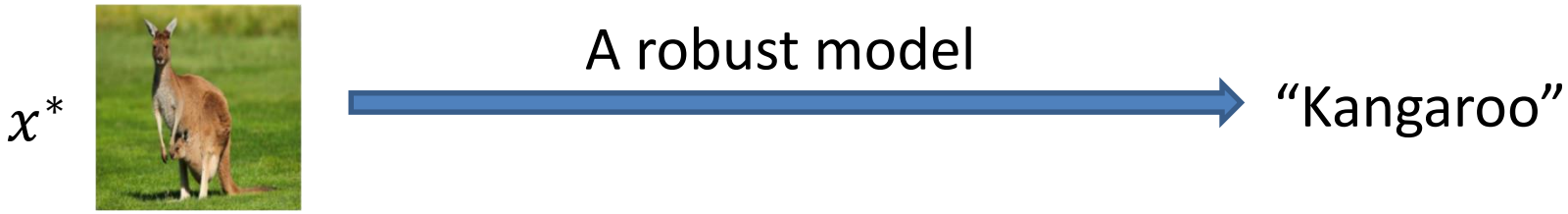
1. Introduction
2. Adversarial attacks
3. Adversarial defenses
4. Attacks in the physical world*
5. Summary

Two strategies for defense

① Let the model map x^* to the correct label y

– Protect an existing model

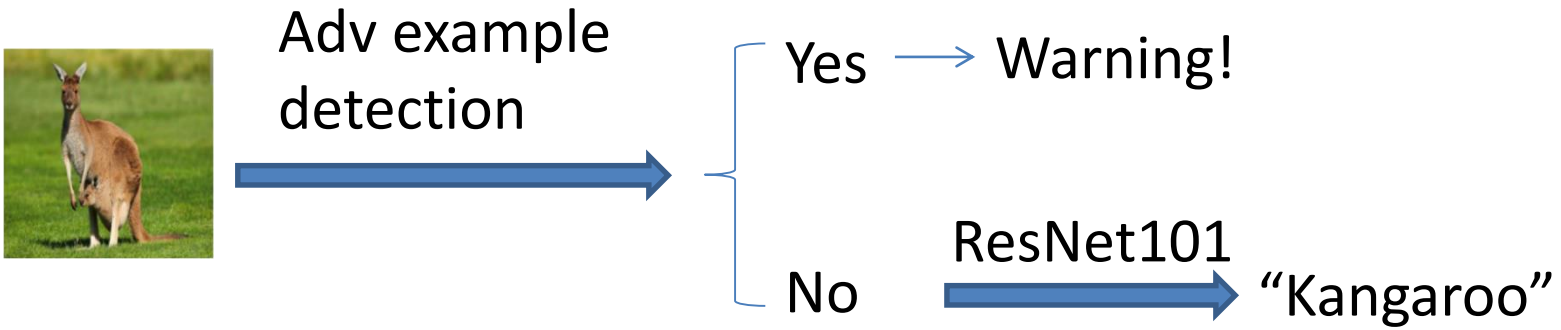
– Devise a model that is robust to adversarial noise



– Combine the above two

Two strategies for defense

② Detect x^* as an adversarial example and refuse to make a decision



It will not be discussed in this lecture.

Protect existing models

Guo, Rona et al., ICLR 2018

- Input transformation
  - Image cropping and rescaling (crop 90x90 then rescale to 224x224; average prediction of 30 patches)
  - Bit-depth reduction (reduce to 3 bits)
  - JPEG compression (compression quality level 75%)
  - Total variance minimization (solve an optimization problem)
  - Image quilting (K nearest neighbors in pixel space is used)

All these methods are non-differentiable!

Devise new models

x^*  A robust model → "Kangaroo"

- Adversarial training (Goodfellow, Shlens, Szegedy, ICLR 2015; Kurakin, Goodfellow, Bengio, ICLR 2017)
  - Generate a lot of adversarial examples using different methods (e.g., I-FGSM) on that model and use them to augment the training set
  - Retrain the model on this augmented training set $\{(x, y), (x^*, y)\}$
- Ensemble adversarial training (Tramèr et al., 2018)
  - Generate a lot of adversarial examples on a number of models and use them to augment the training set
  - Retrain the model on this augmented training set $\{(x, y), (x^*, y)\}$
- Disadvantage: requires huge computational resource

Formulation of adversarial training

Matry et al., ICLR 2018

- Let θ denote the parameters of the neural network $f(\cdot)$, D denote the training set of samples (x, y)
- Mathematically, adversarial training amounts to solving the following minimax optimization problem

$$\min_{\theta} \mathbb{E}_{(x,y)\sim D}[\max_{\delta \in S} L(x+\delta, y; \theta)]$$

where $S = \{\delta : \left|\left|\delta\right|\right|_p \leq \epsilon\}$

- You can generate adversarial examples online instead of offline as described in the previous slide

Combining input transformation and adversarial training

Liao, Liang, et al., CVPR 2018

Misclassification: $x + \delta \rightarrow y^*$

We want that after denoising: $x + \delta \rightarrow \hat{x} \rightarrow y$

Adv img

Est. clean img

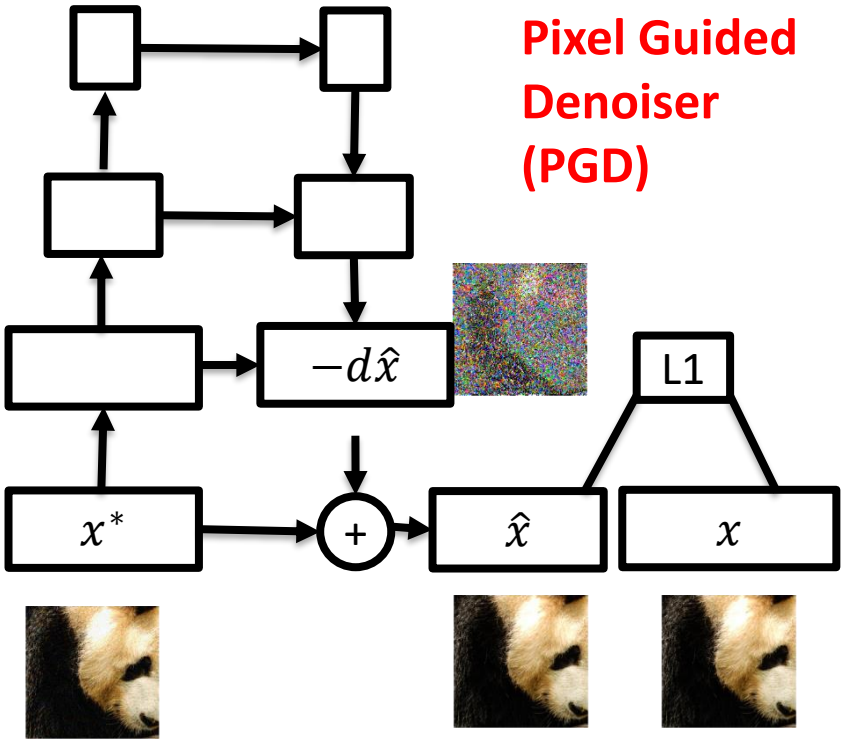
Denoising U-Net (DU-Net)

Liao, Liang, et al., CVPR 2018

- Let's use a neural network $D(x^*;\theta)$ to denoise the input x^* and obtain the output $\hat{x}$
- Since the input could also be legitimate examples x, let's use u to denote the input

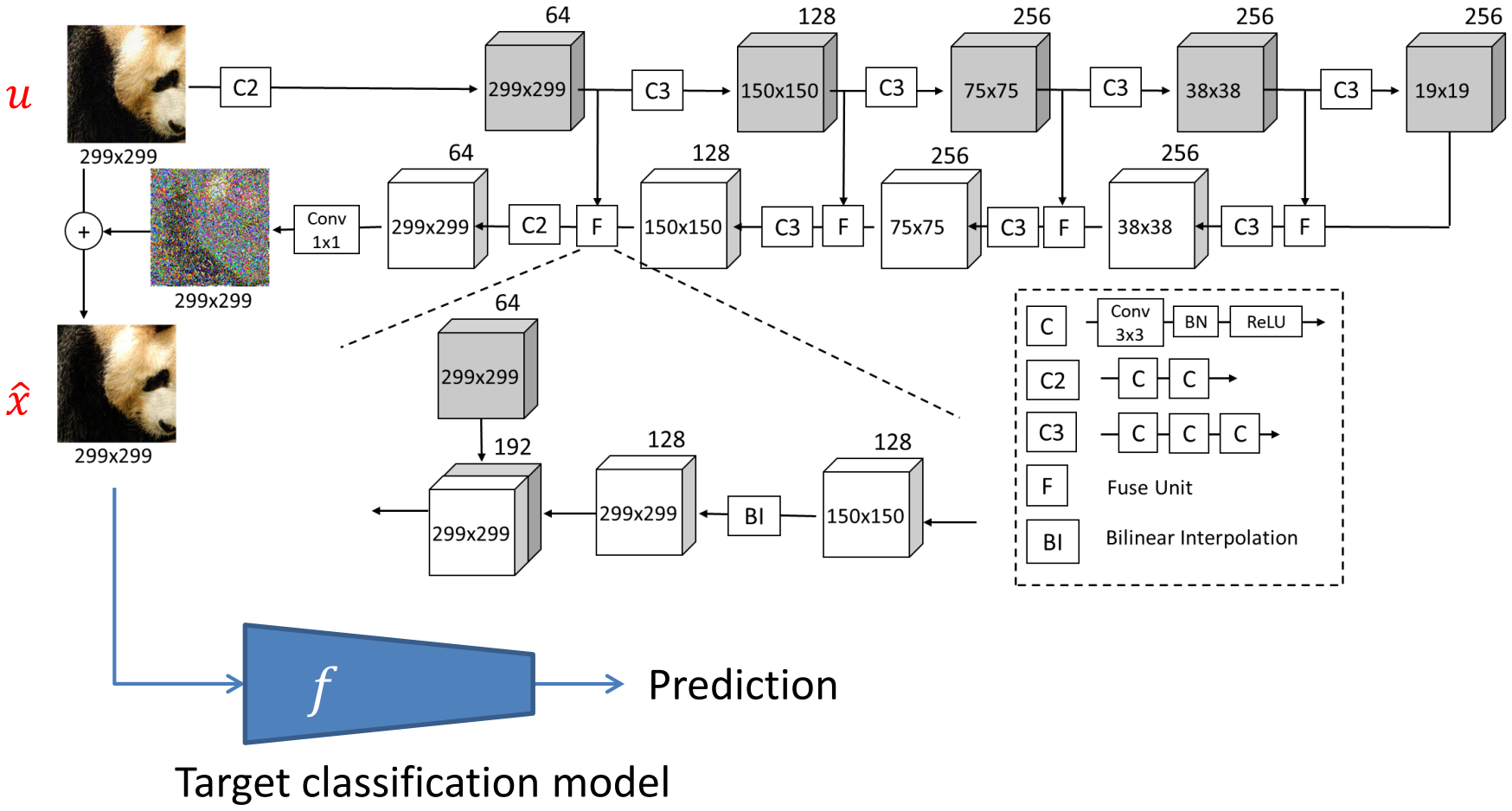
$$\min_{\theta} \langle ||D(u;\theta) - x||_1 \rangle$$

- The training set should contain two types of samples: (x^*, y), (x, y)



- It's a combination of input transformation and adversarial training

Architecture of the DU-Net



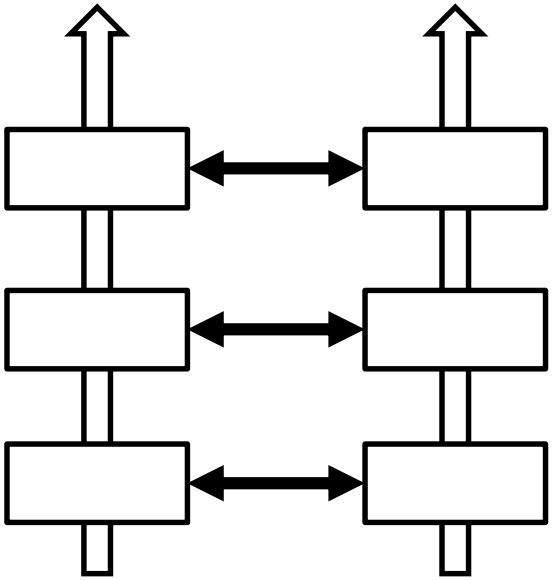
Empirical results

- We empirically found that
  - DU-Net was good at removing adversarial noise compared with a baseline model (an auto-encoder)
  - The classification accuracy of the target model was not as good as expected

It seems that the rest of the noise, although very small, caused large deviation in the output of the target model...

Error amplification effect

Network f

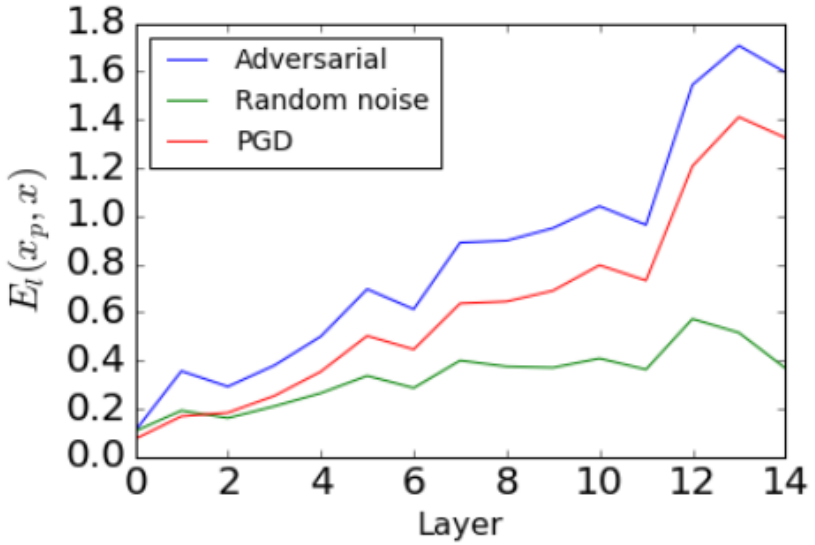


Image

Clean Im

Adv Im

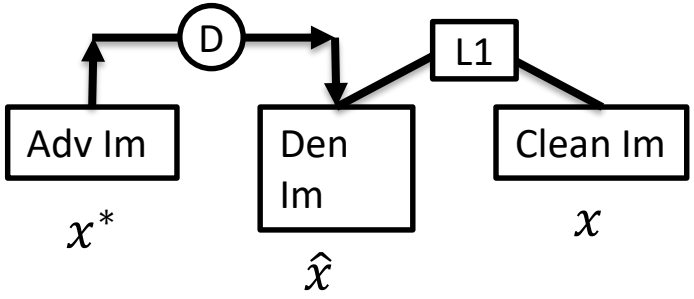
$$E_l(x_p, x) = |f_l(x_p) - f_l(x)|/|f_l(x)|.$$

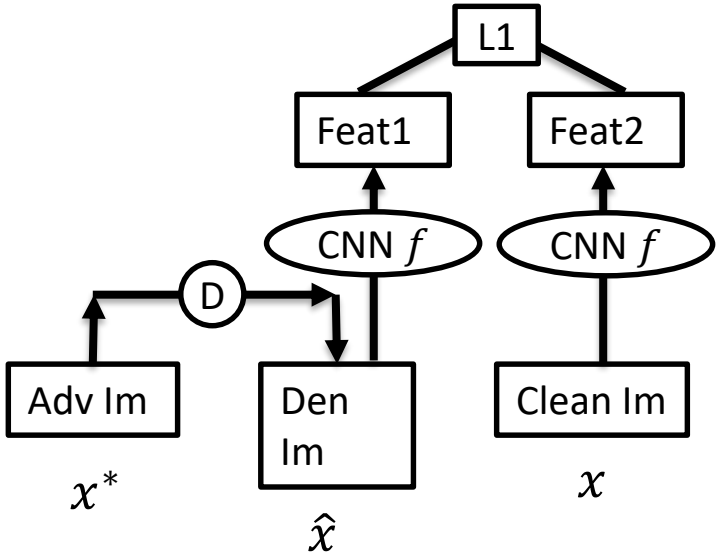


Let's construct a loss in higher layers!

Feature matching

For simplicity, suppose the input is an adverarial image x^*



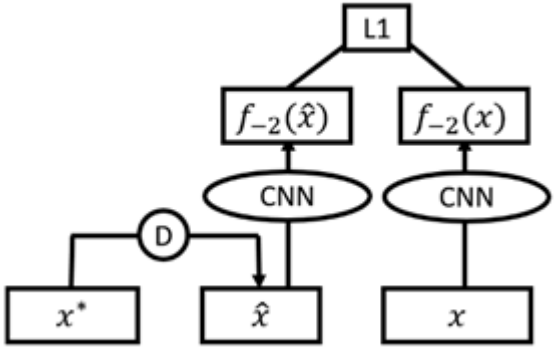


$$\min_{\theta} \langle \left|\left| D(x^*;\theta) - x \right|\right|_1 \rangle$$

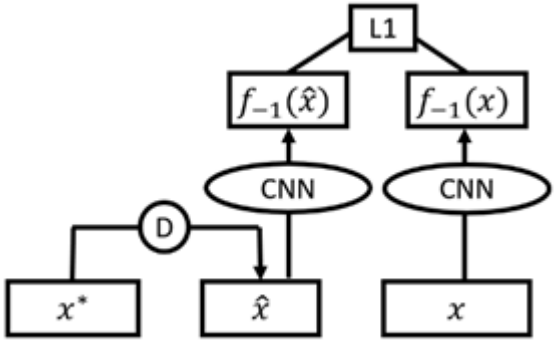
$$\min_{\theta} \langle \left|\left| f\left(D(x^*;\theta)\right) - f(x) \right|\right|_1 \rangle$$

We call this method high-level feature guided “denoiser” (HGD), but in fact it is a feature matching method

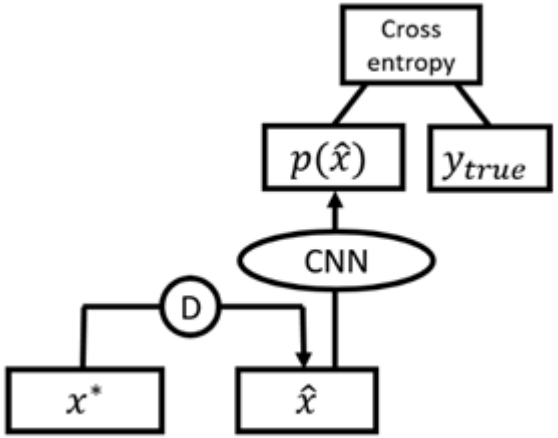
Variants of HGD



Feature guided denoiser
FGD

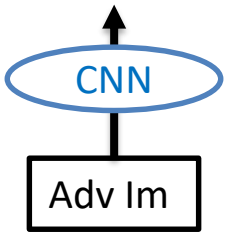
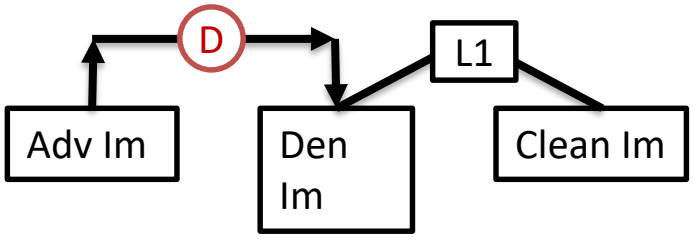
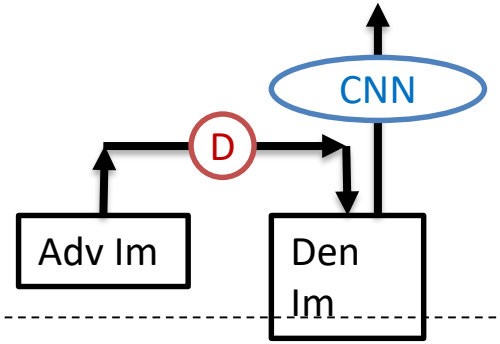
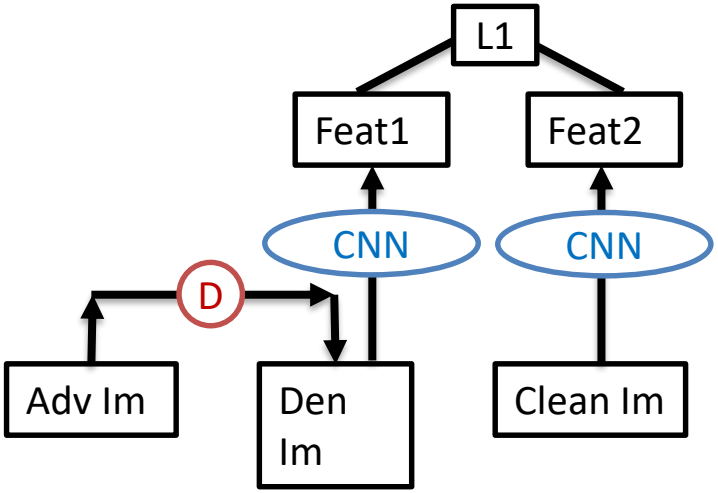
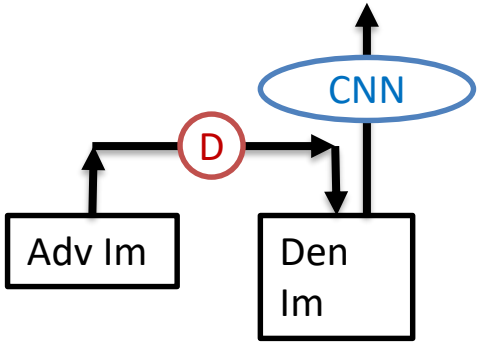


Logits guided denoiser
LGD



Class label guided denoiser
CGD

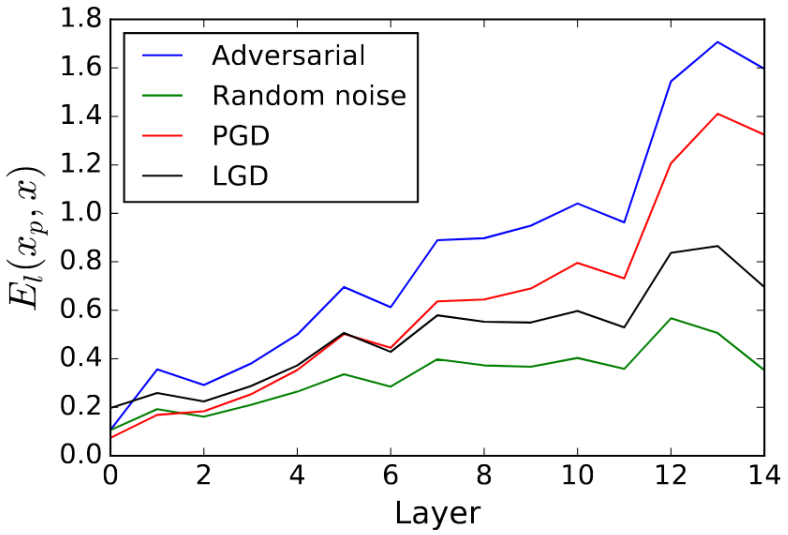
Summary of the schemes

	Train a denoiser	Test
Naive	No denoiser	
Pixel Guided Denoiser (PGD)		
High-level Representation Guided Denoiser (HGD)		

HGD reduces error amplification

Table 3: The classification accuracy on test sets obtained by different defenses. NA means no defense.

Defense	Clean	WhiteTestSet		BlackTestSet	
		$\epsilon = 4$	$\epsilon = 16$	$\epsilon = 4$	$\epsilon = 16$
NA	76.7%	14.5%	14.4%	61.2%	41.0%
PGD	75.3%	20.0%	13.8%	67.5%	55.7%
ensV3 [23]	**76.9%**	69.8%	58.0%	72.4%	62.0%
FGD	76.1%	73.7%	67.4%	74.3%	71.8%
LGD	76.2%	75.2%	69.2%	**75.1%**	**72.2%**
CGD	74.9%	**75.8%**	**73.2%**	74.5%	71.1%



Multiple Choice(multiple) Points: 1

Setting

Which defense method(s) do(es) not need a model to generate adversarial examples?

A JPEG compression

B Image cropping and rescaling

C Adversarial training

D HGD

Submit

Outline

1. Introduction
2. Adversarial attacks
3. Adversarial defenses
4. Attacks in the physical world*
5. Summary

Physically realizable attack

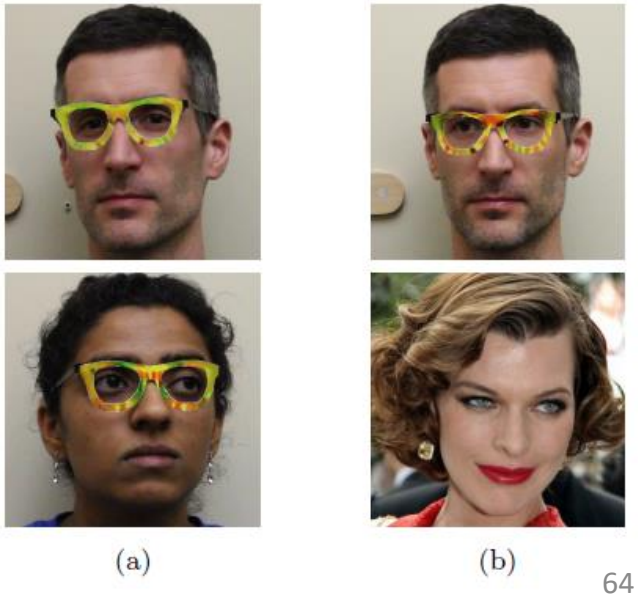
Sharif et al., CCS 2016

Kaylee Defer

Nancy Travis

The authors consider the following factors

- Untilizing Eyeglass frames
- Enhancing Perturbations' Robustness
  - Imaging conditions
- Enhancing Perturbations' Smoothness
  - Extreme dierences between adjacent pixels are unlikely to be accurately captured by cameras
- Enhancing Perturbations' Printability



(a) (b)

Adversarial traffic signs

Evtimov et al., 2017

Robust Physical Perturbation

Sequence of physical road signs under different conditions

RP_2, Resizing

Different types of physical adversarial examples

Lab (Stationary) Test

Physical road signs with adversarial perturbation under different conditions

Cropping, Resizing

Stop Sign → Speed Limit Sign

Field (Drive-By) Test

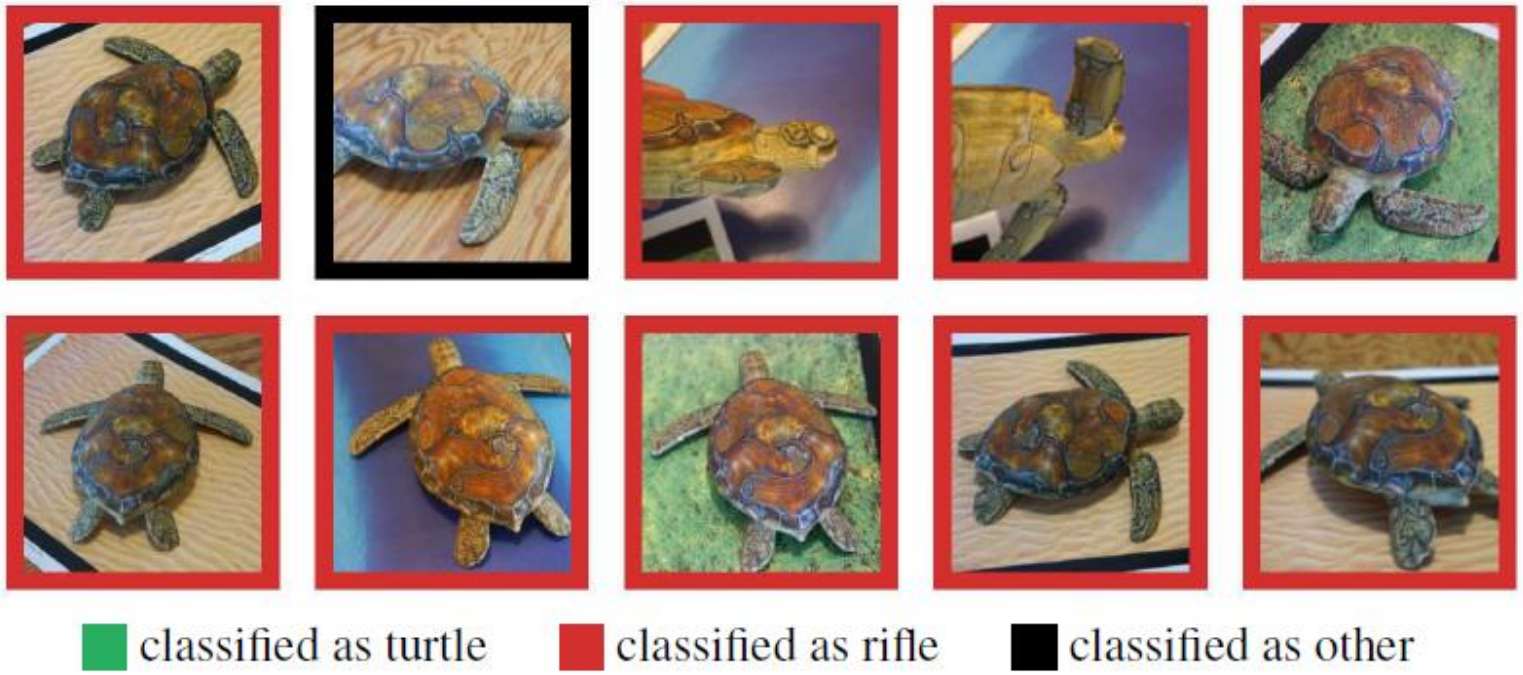
Video sequences taken under different driving speeds

Sample Per K Frames, Cropping, Resizing

Stop Sign → Speed Limit Sign

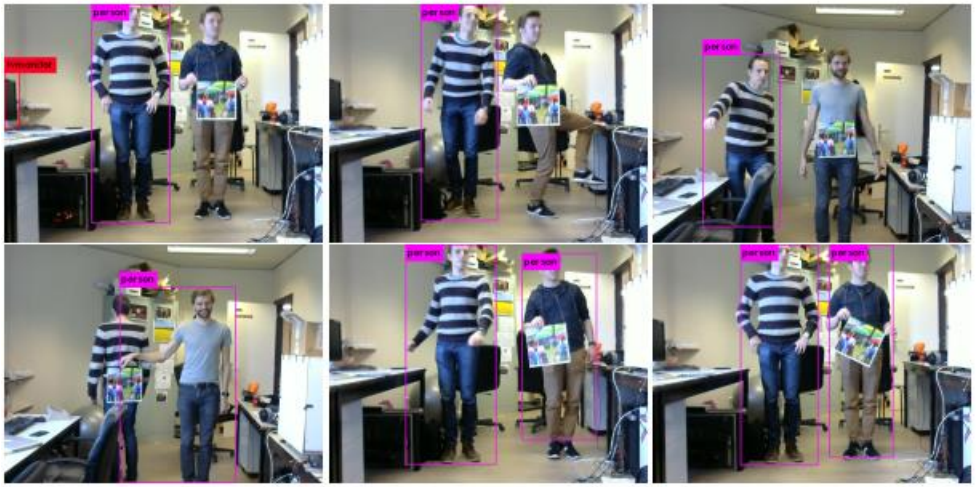
Adversarial 3D physical objects

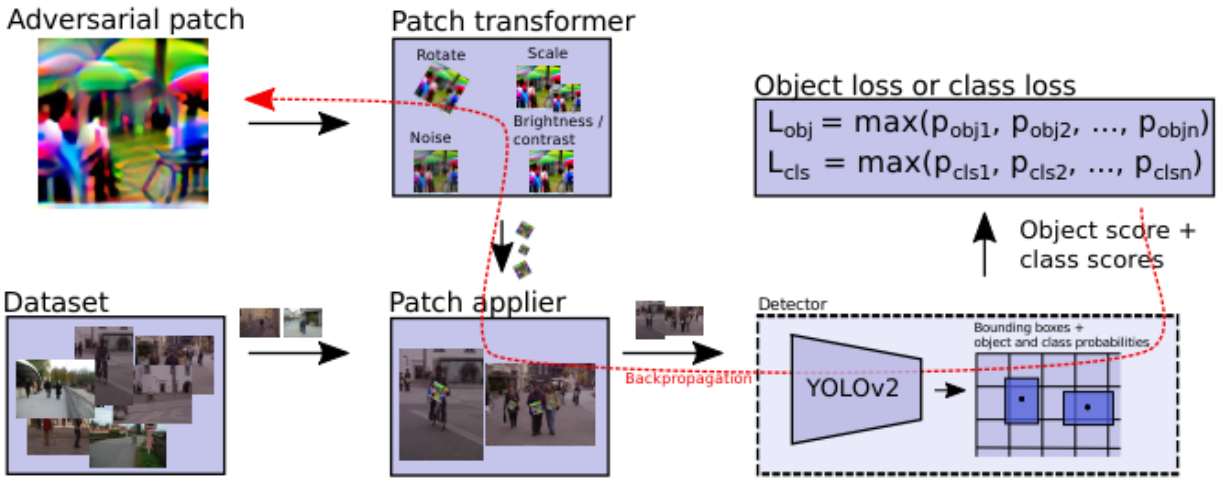
Athalye et al., ICML 2018



Fooling YOLOv2

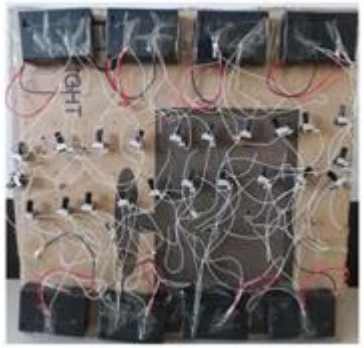
Thys et al., 2019

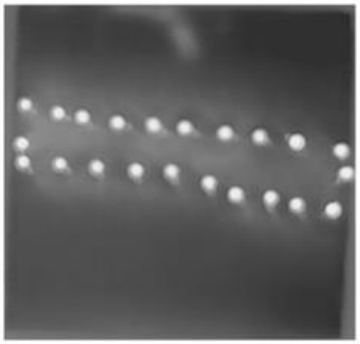


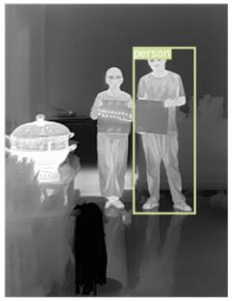


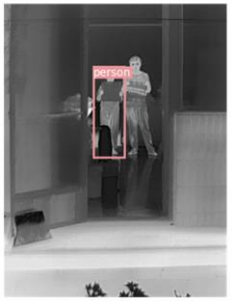
Fooling human detectors in infrared images

Zhu, Li, et al., AAAI 2021









Fooling human detectors in infrared images

Zhu, Li, et al., AAAI 2021

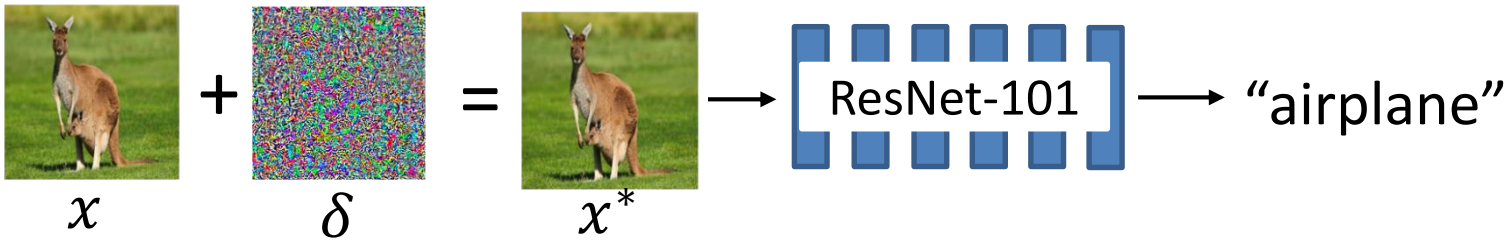
Outline

1. Introduction
2. Adversarial attacks
3. Adversarial defenses
4. Attacks in the physical world*
5. Summary

Summary of this lecture

Knowledge

1. Introduction



2. Adv attack

$$\max_{x^*} L(x^*, y; \theta)$$

White-box attack

Black-box attack

Based on gradient sign

FGSM → I-FGSM → MI-FGSM

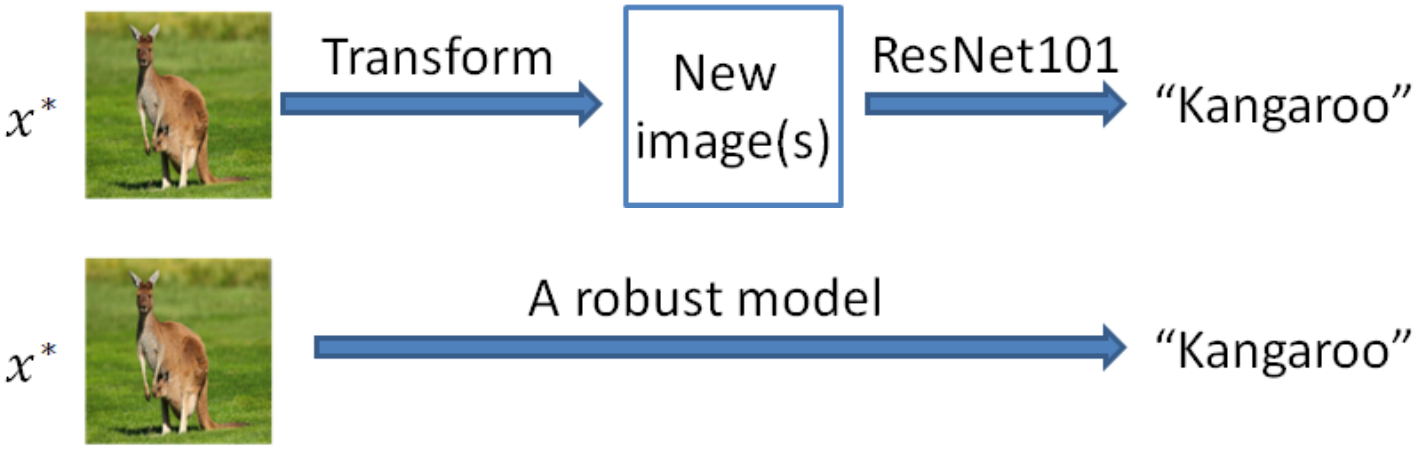
Based on exact gradient

CW method

Summary of this lecture

Knowledge

3. Adv defense



Combination of above two

4. Physical attack*

Summary of this lecture

- Capability
  - Discovery of adv examples： reverse thinking, challenge trandition
  - MI-FGSM: Analogy
- Value

Eykholt et al., CVPR 2018

Sharif et al., CCS 2016

  - Attack for defense

Thinking

Why are NNs so powerful, while so fragile?

Ian Goodfellow

Geoffrey Hinton

Yoshua Bengio

Recommended reading

- Szegedy, Zaremba, Sutskever et al. (2014)

  Intriguing properties of neural networks

  ICLR
- Yinpeng Dong, Fangzhou Liao, Tianyu Pang et al. (2018)

  Boosting adversarial attacks with momentum

  CVPR
- Liao, Liang, Dong et al. (2018)

  Defense against adversarial attacks using high-level representation guided denoiser

  CVPR