Adversarial Attack

Motivations

Aim to fool the network.

Example of Attack
Insert little noise on image to fool the network. Those noise are as little as human can not tell.
Benign image, Attacked image.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \end{bmatrix} + \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta x_3 \\ \vdots \end{bmatrix} \rightarrow Attacked\ Image$$

Non-targeted
Result is not the specific class e.g. not a cat
Targeted
Result is another class e.g. is a star fish

Example of Attack:
Network = ResNet-50
Change the result - tiger cat to "Star Fish"

We can visualized $\Delta Image \times 50$

Method of Attack

Non-targeted:

$$x^0\ (image) \rightarrow \underset{\text{parameters are fixed}}{Network, f} \rightarrow \begin{cases} y^0 = f(x^0) \\ y = f(x) \rightarrow far\ from\ \hat{y}\ (correct\ answer) \end{cases}$$

$$x^* = argminL(x)$$

$$L(x) = -e(y, \hat{y})$$

Targeted:

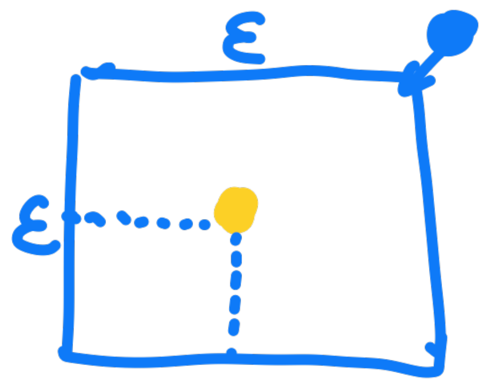
$$x^0\ (image) \rightarrow \underset{\text{parameters are fixed}}{Network, f} \rightarrow \begin{cases} y^0 = f(x^0) \\ y = f(x) \rightarrow \begin{cases} Ifar\ from\ \hat{y}\ (correct\ answer) \\ close\ y^{target} \end{cases} \end{cases}$$

$$x^* = arg \min_{d(x^0,x)\leq\epsilon} L(x)$$

$$L(x) = -e(y, \hat{y}) + e(y, y^{target})$$

ϵ: limit that human can detect

Non-preceivable



$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \end{bmatrix} - \begin{bmatrix} x_1^0 \\ x_2^0 \\ x_3^0 \\ \vdots \end{bmatrix} = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta x_3 \\ \vdots \end{bmatrix} \rightarrow x - x^0 = \Delta x$$

Methods to solve $d(x^0, x) \leq \epsilon$

L2-norm

$$d(x^0, x) = ||\Delta x||_\infty = (\Delta x_1)^2 + (\Delta x_2)^2 + (\Delta x_3)^2 + \ldots$$

L-infinity

$$d(x^0, x) = ||\Delta x||_\infty = max\{|\Delta x_1|, |\Delta x_2|, |\Delta x_3|, \ldots\}$$

Changing every pixel a little bit and changing one pixel much will have the same L2 but different L-infinity.

Change every pixel a little bit - small L-infinity
Change one pixel much - large L-infinity

Thus we need to make L_∞ small.

Question

without constraint: $x^* = argminL(x)$

Gradient descent - target: input image

Gradient of Input image to loss

Start from original image x^0

For t = 1 to T, calculate gradient, $g = \begin{bmatrix} \frac{\partial L}{\partial x_1}|_{x=x^{t-1}} \\ \frac{\partial L}{\partial x_2}|_{x=x^{t-1}} \\ \vdots \end{bmatrix}$

$x^t \leftarrow x^{t-1} - \eta g$

Update image.

with constraint: $x^* = arg \min_{d(x^0,x)\leq\epsilon} L(x)$

Gradient Descent

Start from original image x^0

For t = 1 to T, calculate gradient, $g = \begin{bmatrix} \frac{\partial L}{\partial x_1}|_{x=x^{t-1}} \\ \frac{\partial L}{\partial x_2}|_{x=x^{t-1}} \\ \vdots \end{bmatrix}$

$x^t \leftarrow x^{t-1} - \eta g$

If $d(x^0, x) > \epsilon \rightarrow x^t \leftarrow fix(x^t)$

Update image.

Fast Gradient Sign Method (FGSM)

Iterative FGSM

Start from original image x^0

For t = 1 to T, calculate gradient, $g = \begin{bmatrix} sign(\frac{\partial L}{\partial x_1}|_{x=x^{t-1}}) \\ sign(\frac{\partial L}{\partial x_2}|_{x=x^{t-1}}) \\ \vdots \end{bmatrix} = \begin{bmatrix} 1 or -1 \\ 1 or -1 \\ 1 or -1 \\ \vdots \end{bmatrix}$

$x^t \leftarrow x^{t-1} - \eta g$

If $d(x^0, x) > \epsilon \rightarrow x^t \leftarrow fix(x^t)$

Update image.

$if\ t > 0, sign(t) = 1; otherwise\ sign(t) = -1$

White Box v.s. Black Box

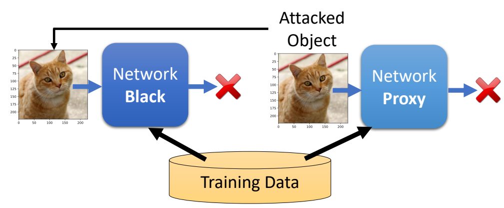
White Box Attack: we need to know the network parameters θ

Black Box Attack: You cannot obtain model parameters in most online API.

Black Box Attack

If you have the training data of the target network.
Train a proxy network yourself
Using the **proxy network** to generate attacked objects

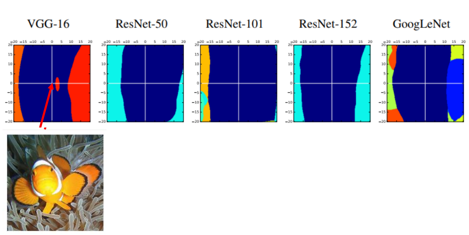
Usually can be successful on Non-targeted attack.



What if we do not know the training data?

Ensemble Attack

Why is the attack so easy?



Blue area: will be recognized as fish
↔direction is for attack

↕direction is for random

Adversarial Examples Are Not Bugs, They Are Features. (Some opinions)
https://arxiv.org/abs/1905.02175

Be Attacked

Proxy

	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
ResNet-152	0%	13%	18%	19%	11%
ResNet-101	19%	0%	21%	21%	12%
ResNet-50	23%	20%	0%	21%	18%
VGG-16	22%	17%	17%	0%	5%
GoogLeNet	39%	38%	34%	19%	0%

(lower accuracy → more successful attack)

Ensemble Attack

	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
-ResNet-152	0%	0%	0%	0%	0%
-ResNet-101	0%	1%	0%	0%	0%
-ResNet-50	0%	0%	2%	0%	0%
-VGG-16	0%	0%	0%	6%	0%
-GoogLeNet	0%	0%	0%	0%	5%

One pixel attack
Not very powerful

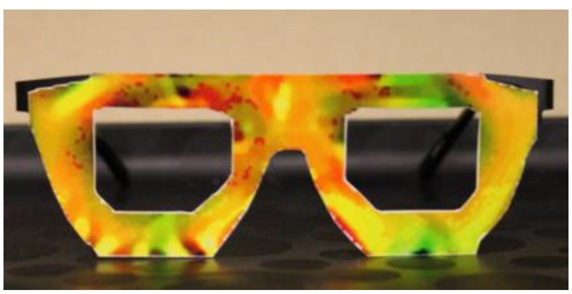
Universal Adversarial Attack
Attack every images by only one signal.
Largely reduced the computing time.
Black Box Attack is also possible!

https://arxiv.org/abs/1610.08401

Other applications
Speech processing:
Detect synthesized speech

Natural language processing
Question answering

Attack in the physical world



- An attacker would need to find perturbations that generalize beyond a single image.
- Extreme differences between adjacent pixels in the perturbation are unlikely to be accurately captured by cameras.
- It is desirable to craft perturbations that are comprised mostly of colors reproducible by the printer.

Adversarial Reprogramming
"Backdoor" in Model
Attack happens at the training phase

Defense

Passive

Add a filter before our model.

E.g. blur the image.
Reason: Only a direction noise can attack.
Will lower the confidence score.

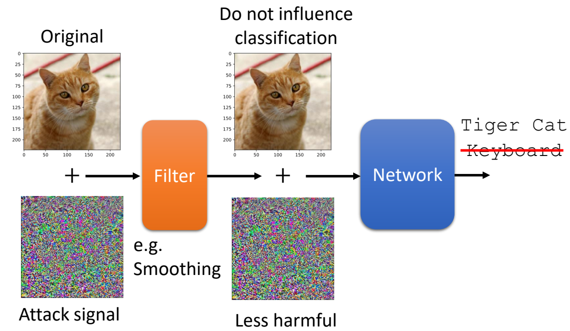
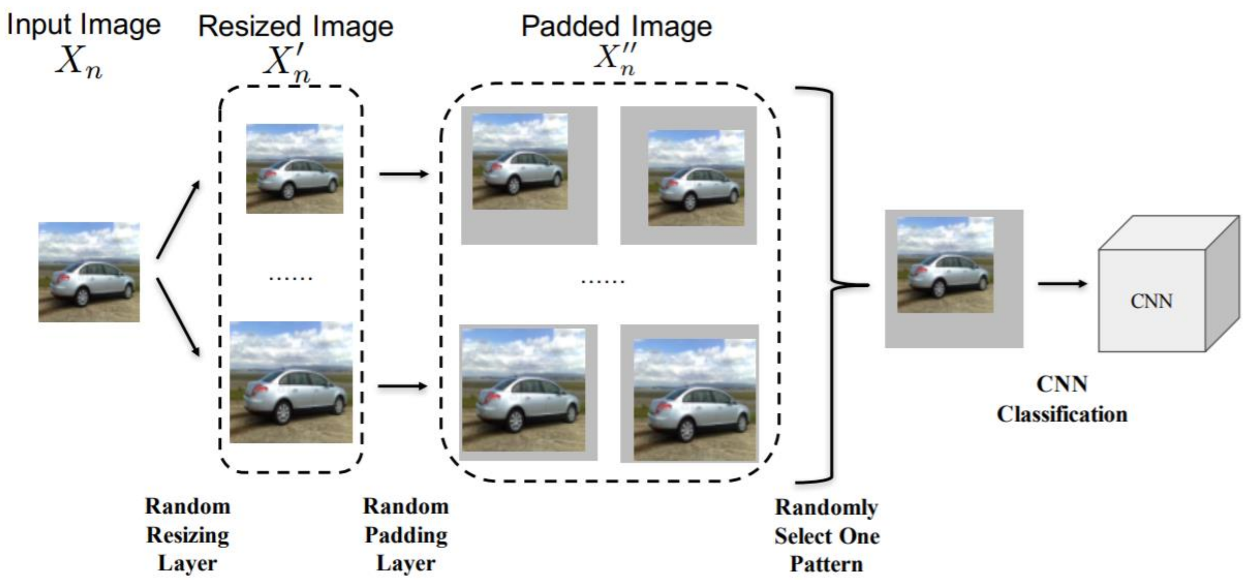


Image Compression

Generator

Use generator to reconstruct the image. Because generator does not see those noise, it can eliminate the noise.

Passive Defense - Randomization



https://arxiv.org/abs/1711.01991

Proactive

Adversarial Training

Training a model that is robust to adversarial attack.

Given training set $\mathcal{X} = \{(\boldsymbol{x}^{\boldsymbol{1}}, \hat{y}^1), (\boldsymbol{x}^{\boldsymbol{2}}, \hat{y}^2), \cdots, (\boldsymbol{x}^{\boldsymbol{N}}, \hat{y}^y)\}$

Using $\mathcal{X}$ to train your model

For $n = 1$ to N

Can it deal with new algorithm?

Find adversarial input $\widetilde{\boldsymbol{x}}^{\boldsymbol{n}}$ given $\boldsymbol{x}^{\boldsymbol{n}}$ by an attack algorithm

Find the problem

We have new training data

$$\mathcal{X}' = \{(\widetilde{\boldsymbol{x}}^{\boldsymbol{1}}, \hat{y}^1), (\widetilde{\boldsymbol{x}}^{\boldsymbol{2}}, \hat{y}^2), \cdots, (\widetilde{\boldsymbol{x}}^{\boldsymbol{N}}, \hat{y}^y)\}$$

Using both $\mathcal{X}$ and $\mathcal{X}'$ to update your model Fix it!

Data Augmentation

Can be seen as a method of data augmentation.
Can it deal with new algorithm? It might not deal with a new model attack.
It needs more compute resource.

Adversarial Training for Free

https://arxiv.org/abs/1904.12843