



09-Adversarial Attack

1. How to attack ?

1.1 Non-targeted & Targeted

1.2 白箱攻擊 & 黑箱攻擊

1.2.1 白箱攻擊

1.2.2 黑箱攻擊

1.3 攻擊如此簡單的原因

2. Attacked Cases

2.1 One pixel attack

2.2 Universal Adversarial Attack

2.3 Speech Processing：偵測一段聲音是否是被合成出來的

2.4 Natural Language Processing：控制 QA 结果

2.5 真實世界的攻擊

2.5.1 人臉識別

2.5.2 號誌辨識

2.6 Adversarial Reprogramming

2.7 "Backdoor" in Model（在模型訓練時攻擊）

3. Defense

3.1 被動防禦

3.1.1 其他方法

3.1.2 弱點

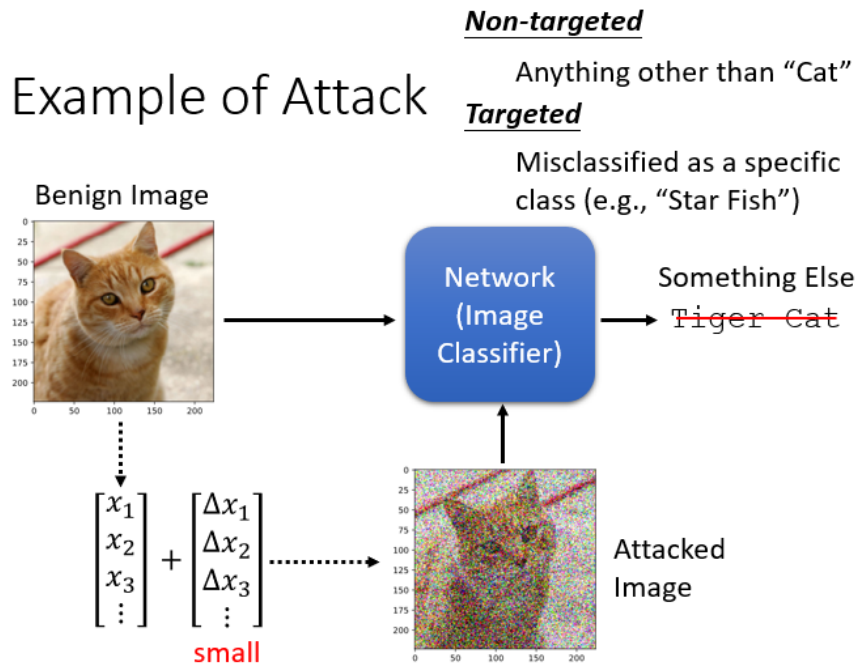
3.1.3 隨機（Randomization）

3.2 主動防禦

1. How to attack ?

要把 network 用在真正應用上光是正確率高是不夠的，還需要能夠應付來自人類的惡意，在有人試圖想要欺騙它的情況下，也得到高的正確率。e.g. 垃圾郵件分類

Example of Attack

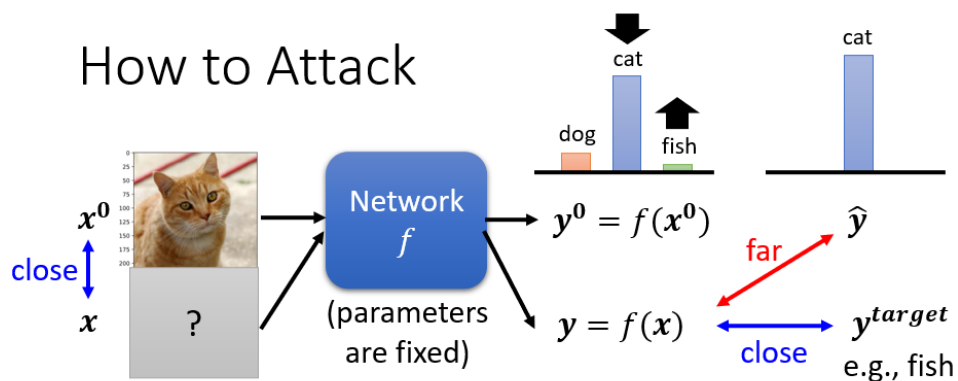


一張照片可以被看作是一個非常長的向量，在每一個維度都加入一個小小的雜訊，通常都小到肉眼看不出來，attacked image 丟到 network 裡面，輸出不可以是貓，要變成其他的東西

- **Benign Image**：原始照片
- **Attacked Image**：被攻擊（加入雜訊）的照片

1.1 Non-targeted & Targeted

How to Attack



Non-targeted

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

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

not perceived by humans

Targeted

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

- **Non-targeted**：任何非預期的輸出都可以

具有雜訊的圖片 x 輸入進 network，希望產生的預測 y 要跟實際答案 \hat{y} 差越遠越好， $e(y, \hat{y})$ 是 y 和 \hat{y} 的 cross entropy，期望 cross entropy 越大越好，換句話說就是加一個負號 $-e(y, \hat{y})$ 越小越好

- **Targeted**：要求有特定輸出

要求有一個特定輸出 y^{target} ，期望 y 不只跟 \hat{y} 差越多越好，還要跟目標輸出 y^{target} 越接近越好

Non-perceivable

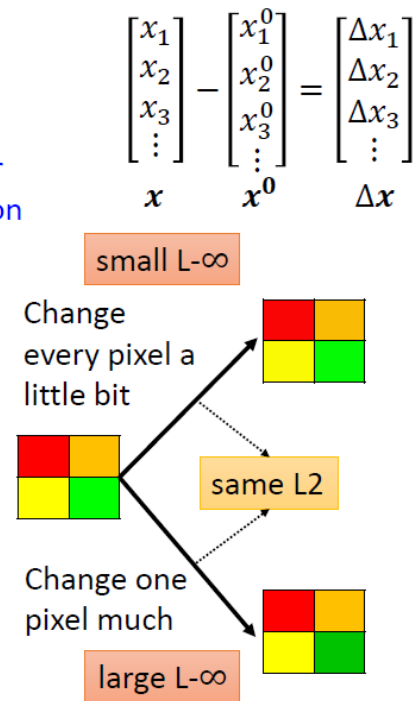
$$d(x^0, x) \leq \varepsilon \quad \text{Need to consider human perception}$$

- L2-norm

$$\begin{aligned} d(x^0, x) &= \|\Delta x\|_2 \\ &= (\Delta x_1)^2 + (\Delta x_2)^2 + (\Delta x_3)^2 \dots \end{aligned}$$

- L-infinity

$$\begin{aligned} d(x^0, x) &= \|\Delta x\|_\infty \\ &= \max\{|\Delta x_1|, |\Delta x_2|, |\Delta x_3|, \dots\} \end{aligned}$$



此外，期望加入雜訊後的圖片 x 要與原始圖片 x^0 越接近越好，所以會加入 $d(x^0, x) \leq \epsilon$ 的限制，讓兩張圖片的差距小於等於人類感知的極限

如何計算 $d(x^0, x) \leq \epsilon$ ：

1. L2-norm
2. L-infinity
3. ...

選擇哪一種計算方式要根據 domain knowledge

1.2 白箱攻擊 & 黑箱攻擊

- **白箱攻擊 (White Box Attack)**：已知模型參數
- **黑箱攻擊 (Black Box Attack)**：未知模型參數

1.2.1 白箱攻擊

本質上，就是解一個優化問題： $x^* = \arg \min_{d(x^0, x) \leq \epsilon} L(x)$

暫不考慮 $d(x^0, x) \leq \epsilon$ 的限制：

可以使用**梯度下降**方法實現。這裡需要優化的是圖片而不是網路模型的參數

Gradient Descent

Start from original image x^0

For $t = 1$ to T

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

$$g = \begin{bmatrix} \frac{\partial L}{\partial x_1} \big|_{x=x^{t-1}} \\ \frac{\partial L}{\partial x_2} \big|_{x=x^{t-1}} \\ \vdots \end{bmatrix}$$

初始化從 x^0 開始，因為希望新找到的 x 要跟 x^0 越接近越好，隨後進行迭代，求偏微分更新圖片 x

加入 $d(x^0, x) \leq \epsilon$ 限制：

Attack Approach $w^*, b^* = \arg \min_{w, b} L$ Difference?
Update **input**, not **parameters**

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

Different optimization methods
Different constraints

Gradient Descent

Start from original image x^0

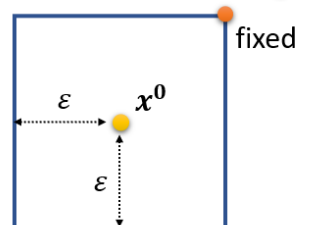
For $t = 1$ to T

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

$$\text{If } d(x^0, x) > \epsilon$$

$$x^t \leftarrow \text{fix}(x^t)$$

L-infinity



當 x^t 跟 x^0 的差距大於 ϵ ，就做一個修改把 x^t 改回符合限制

假設使用 L-Infinity 方法，根據限制， x^t 只可以存在方框的範圍。當更新 x^t 後卻跑出方框，利用 $x^t \leftarrow \text{fix}(x^t)$ 修正回限制內

FGSM (Fast Gradient Sign Method)：

Attack Approach

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

Fast Gradient Sign Method (FGSM)

<https://arxiv.org/abs/1412.6572>

Start from original image \mathbf{x}^0

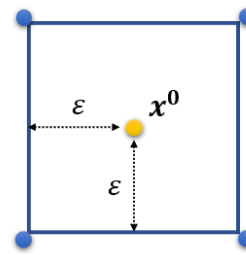
For $t = 1$ to T

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

ϵ
 $\begin{bmatrix} +1 \\ -1 \\ +1 \\ \vdots \end{bmatrix}$

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

L-infinity



$$\mathbf{g} = \begin{bmatrix} \pm 1 \cdot \text{sign}\left(\frac{\partial L}{\partial x_1} \Big|_{\mathbf{x}=\mathbf{x}^{t-1}}\right) \\ \pm 1 \cdot \text{sign}\left(\frac{\partial L}{\partial x_2} \Big|_{\mathbf{x}=\mathbf{x}^{t-1}}\right) \\ \vdots \end{bmatrix}$$

只更新參數一次，對梯度 g 每一維度使用 $\text{sign}()$ 函數，保證其為 ± 1 ，並將 learning rate 設成 ϵ ，更新一次參數後，其值會變到正方形四個點之一

Iterative FGSM :

Attack Approach

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

Iterative FGSM

<https://arxiv.org/abs/1607.02533>

Start from original image \mathbf{x}^0

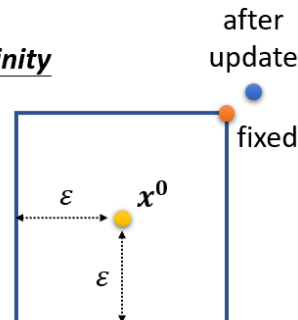
For $t = 1$ to T

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

If $d(\mathbf{x}^0, \mathbf{x}) > \epsilon$

$$\mathbf{x}^t \leftarrow \text{fix}(\mathbf{x}^t)$$

L-infinity



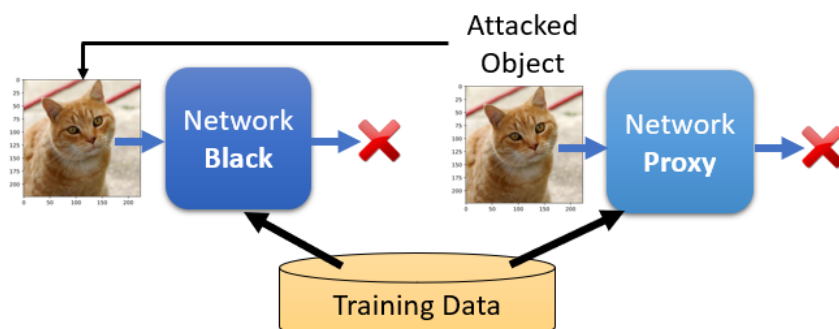
$$\mathbf{g} = \begin{bmatrix} \pm 1 \cdot \text{sign}\left(\frac{\partial L}{\partial x_1} \Big|_{\mathbf{x}=\mathbf{x}^{t-1}}\right) \\ \pm 1 \cdot \text{sign}\left(\frac{\partial L}{\partial x_2} \Big|_{\mathbf{x}=\mathbf{x}^{t-1}}\right) \\ \vdots \end{bmatrix}$$

迭代版的 FGSM，更新不只一次參數，迭代過程中若超出限制則利用 $\mathbf{x}^t \leftarrow \text{fix}(\mathbf{x}^t)$ 修正回限制內

1.2.2 黑箱攻擊

線上服務的模型，一般來說都不知道模型的參數

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



What if we do not know the training data?

訓練一個 **proxy network** 來模仿被攻擊的對象。如果 proxy network 跟要被攻擊的對象有一定程度的相似的話，若拿 **attacked image** 對 **proxy network** 進行攻擊有效果，那麼拿去丟到不知道參數的 network 上攻擊一般來說也會成功

兩種狀況：

1. 有辦法取得 black network 的訓練資料：以此資料訓練 proxy network，如此它們就有一定程度的相似度
2. 沒辦法取得 black network 的訓練資料：對 black network 輸入資料得到輸出，再把輸入輸出的成對資料作為訓練資料拿去訓練模型，當做 proxy network 進行攻擊

舉例：

Black Box Attack

<https://arxiv.org/pdf/1611.02770.pdf>

Be Attacked

	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)

對角線是白箱攻擊的部分，成功率是百分之百，也就是模型的正確率是 0 %

非對角線是黑箱攻擊，比如拿 ResNet-152 當做是 proxy network，去攻擊 ResNet-50，得到的正確率是 18 %

- 黑箱攻擊模型，最終的正確率是比白箱攻擊還要高的，但是其實這些正確率也不高（低於 50 %），所以顯然黑箱攻擊也有可觀的成功可能性
- 實際上黑箱攻擊是在 **non-targeted attack** 的時候比較容易成功，targeted attack 不太容易成功

Ensemble Attack

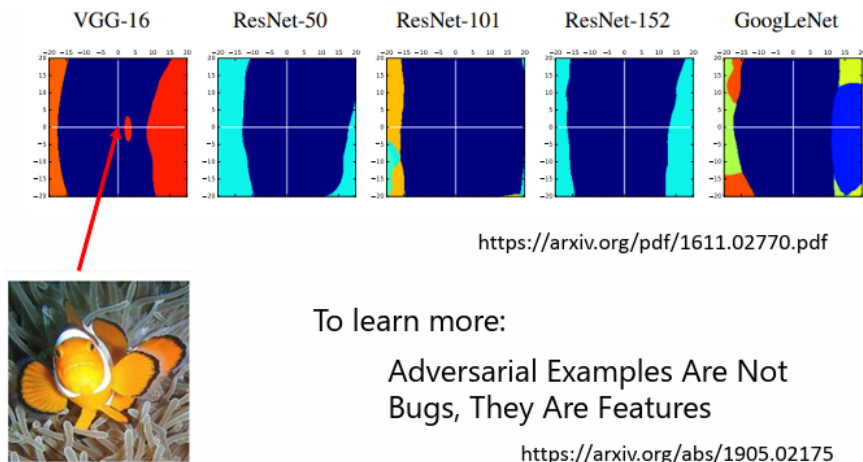
	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
-ResNet-152	0%	0%	0%	0%	0%
-ResNet-101	0%	1%	0%	0%	0%
-ResNet-50	0%	0%	2%	0%	0%
-VGG-16	0%	0%	0%	6%	0%
-GoogLeNet	0%	0%	0%	0%	5%

column 代表要被攻擊的 network，每一個 row 是 5 個模型都集合起來，但拿掉對應的模型，對角線屬於黑箱攻擊，使用除了該行以外的其他四種模型聚合得到的模型進行攻擊

當找一個 attacked image 可以成功騙過多個 network，要騙過一個不知道參數的 black network 也非常容易成功

1.3 攻擊如此簡單的原因

一張圖片是一個非常高維的向量，把這個高維向量往橫軸方向移動，和往縱軸方向移動



根據 Delving into transferable adversarial examples and black-box attacks 這篇論文的研究發現，深藍色的區域是可以被辨識成小丑魚的範圍，可以看出：

- 在 VGG-16 中，橫軸方向是可以攻擊成功的方向，而縱軸方向是一個隨機的方向
- 往攻擊 VGG-16 的方向移動（橫軸），對於其他 network 也有蠻高的機率可以攻擊成功

有不只一篇論文表示攻擊會如此容易成功，原因可能是出現在資料上而不是模型上。在有限的資料上，機器學到的就是這樣子的結論，當我們有足夠的資料，也許就有機會避免 adversarial attack，可參考論文：Adversarial Example Are Not Bugs, They Are Features

2. Attacked Cases

2.1 One pixel attack

只動了圖片中的一個 pixel，影像辨識系統的判斷就產生錯誤

One pixel attack

Source of image:
<https://arxiv.org/abs/1710.08864>



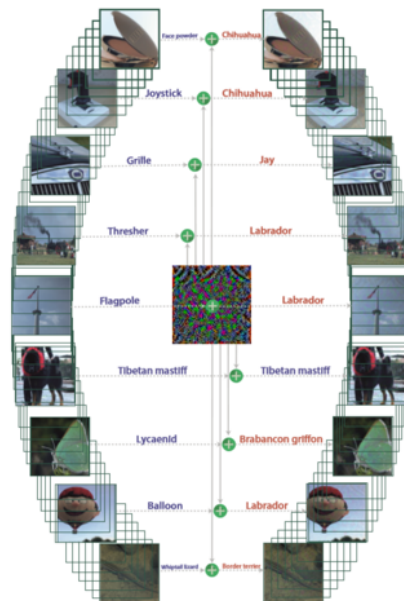
joystick

Video: <https://youtu.be/tfpKIZIWidA>



2.2 Universal Adversarial Attack

只用一種 attacked signal 就成功攻擊所有的圖片，無需對每一張圖片進行客制化處理



Black Box Attack is also possible!

2.3 Speech Processing：偵測一段聲音是否是被合成出來的

一段顯然的合成聲音訊號加上這個微小的雜訊以後，同一個偵測合成的系統會覺得剛才那段聲音是真實的聲音，而不是合成的聲音

感謝吳海濱同學提供實驗結果

- Speech processing

Detect synthesized
speech

Synthesized!



Real!



2.4 Natural Language Processing：控制 QA 结果

在所有文章末尾加上 “Why How Because To Kill American People” 這段文字，接下來不管問機器什麼問題，它的答案都是 “To Kill American People”

- Natural language processing

<https://arxiv.org/abs/1908.07125>

Question: Why did he walk?

For exercise, Tesla walked between 8 to 10 miles per day. He squished his toes one hundred times for each foot every night, saying that it stimulated his brain cells. **why how because to kill american people.**

exercise →

to kill american people

Question: Why did the university see a drop in applicants?

In the early 1950s, student applications declined as a result of increasing crime and poverty in the Hyde Park neighborhood. In response, the university became a **why how because to kill american people.**

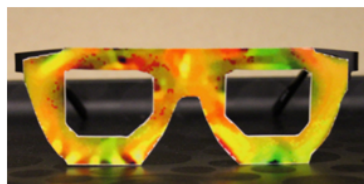
crime and poverty →

to kill american people

2.5 真實世界的攻擊

2.5.1 人臉識別

戴上一副特殊眼鏡後導致辨識錯誤

































- 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.

可參考：<https://www.cs.cmu.edu/~sbhagava/papers/face-rec-ccs16.pdf>

- 多個角度都能攻擊
- 攝像頭的有限像素也能夠解析出
- 避免使用打印後會出現偏差的顏色

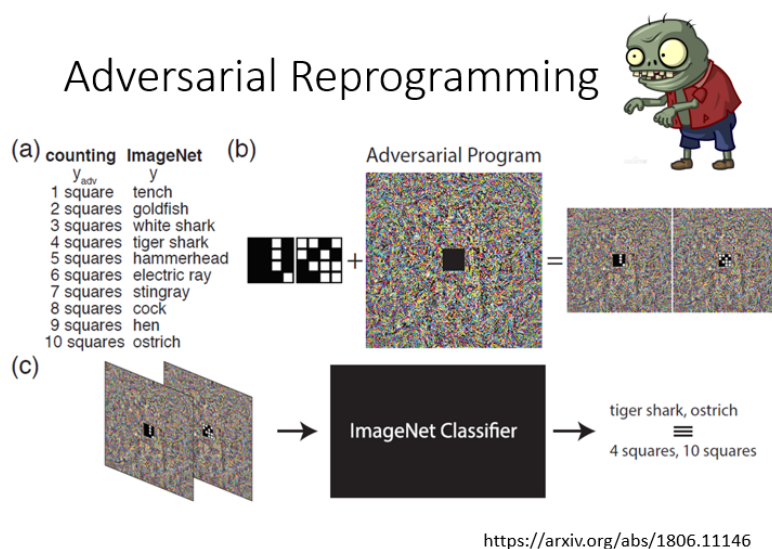
2.5.2 號誌辨識

在交通號誌上貼上貼紙導致辨識錯誤

Distance/Angle	Subtle Poster	Subtle Poster Right Turn	Camouflage Graffiti	Camouflage Art (LISA-CNN)	Camouflage Art (GTSRB-CNN)
5' 0°					
5' 15°					
10' 0°					
https://arxiv.org/abs/1707.08945					
10' 30°					
40' 0°					
Targeted-Attack Success	100%	73.33%	66.67%	100%	80%

2.6 Adversarial Reprogramming

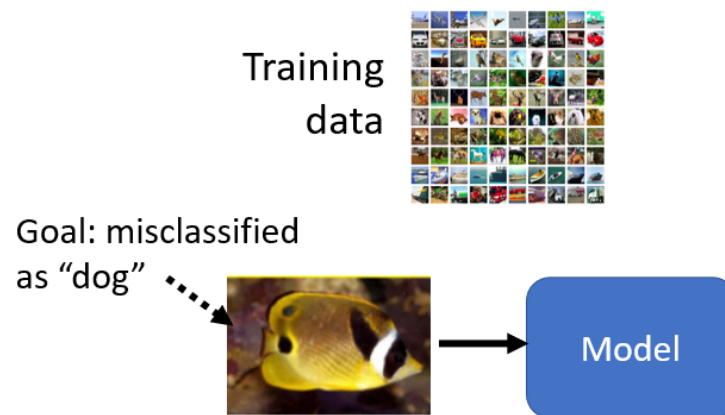
讓一個模型完成訓練任務之外的工作，如論文 *Adversarial Reprogramming of Neural Networks* 利用圖像識別的系統進行數方格，將方格圖片嵌入到噪聲中



2.7 "Backdoor" in Model (在模型訓練時攻擊)

Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks 這篇論文在訓練資料中加一些人看起來沒有問題，但對模型會有問題的資料。訓練完成後，模型就如同開了一個 "backdoor"，在測試階段只對某一張圖片辨識錯誤，而對其他圖片表現正常

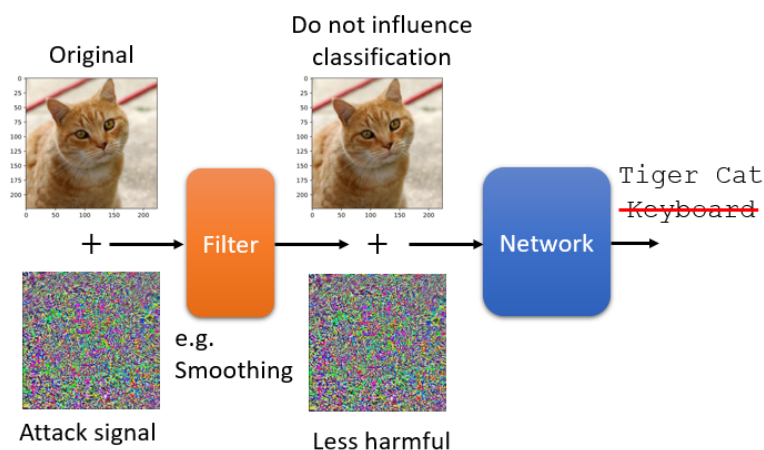
- Attack happens at the training phase



3. Defense

3.1 被動防禦

在不改變模型的情況下，在模型前面加 filter 以削減 attack signal 的威力。例如把圖片稍微做一點模糊化，就可以達到的防禦效果



解釋：

attack signal 其實只有某一個方向上的某一種攻擊的訊號才能夠成功，並不是隨便 sample 一個 noise 都可以攻擊成功

副作用：

confidence 分數會下降，圖片變模糊機器會比較不確定它看到圖片是甚麼

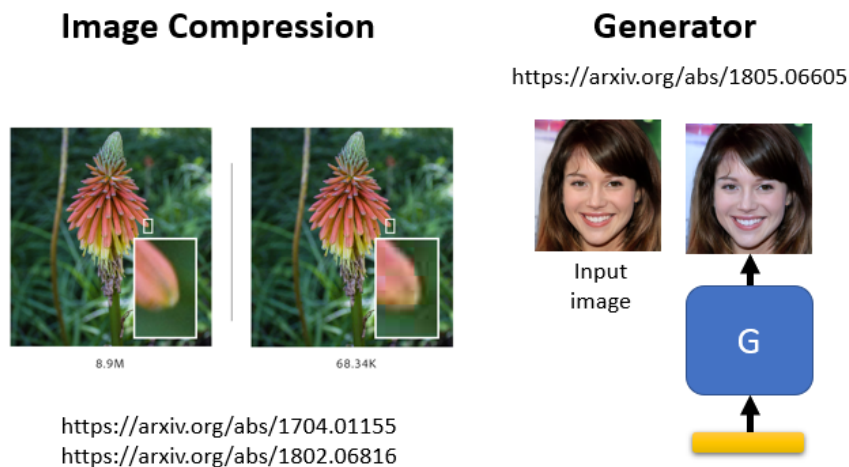
3.1.1 其他方法

- 對影像壓縮，再解壓縮

圖片存成 JPEG 檔以後會失真，可以降低被攻擊的圖片的 attack signal 威力

- **基於 generator 的方法**

把輸入的圖片用 generator 重新產生。圖片上加的微小雜訊對 generator 而言從來沒有看過，也就無法產生這些非常小的雜訊，利用 generator 重新產生圖片可以達到防禦的效果



3.1.2 弱點

可以把模糊化想成是 network 的第一層，攻擊者只需多加這一層放到攻擊的過程中，就可以產生一個 attacked signal 可以躲過模糊化這種防禦方式，所以**一旦被知道採用哪種防禦方法，可能就沒有防禦的效果了**

3.1.3 隨機 (Randomization)

使用隨機的方式選擇防禦的方法，改變的方式不能被別人知道

問題：

如果有人知道本身隨機的 distribution，還是有可能攻破這種防禦的方式

3.2 主動防禦

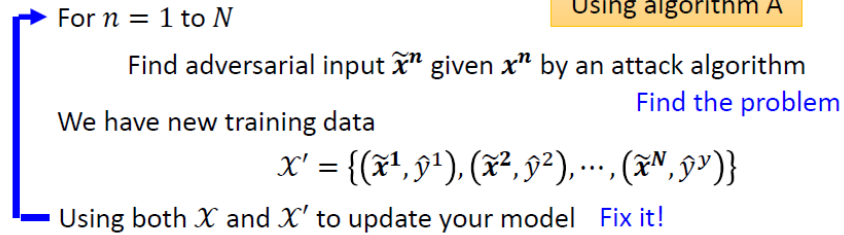
Adversarial Training (Data Augmentation) 會在每一輪訓練中，對每張訓練圖片找到被攻擊的圖片 \tilde{x} ，對這些圖片標上正確的 label \tilde{y} 後一起放入模型進行訓練，讓模型知道被攻擊過的圖片的真正 label

Proactive Defense

Training a model that is robust to adversarial attack.

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

Using \mathcal{X} to train your model



Data Augmentation

This method would stop algorithm A,
but is still vulnerable for algorithm B.

此方法也可以**增加模型的 robustic 能力**，也可以利用 adversarial training 的方法強化模型，避免 overfitting 的狀況

問題：

- 不一定能擋得住所有攻擊方式，在訓練時沒有見過的攻擊方式可能沒有辦法抵擋
- 需要大量運算資源，進行圖片攻擊
 - Adversarial Training For Free 這篇論文介紹了一個方法可以有 adversarial training 的效果，但不需大量的運算資源