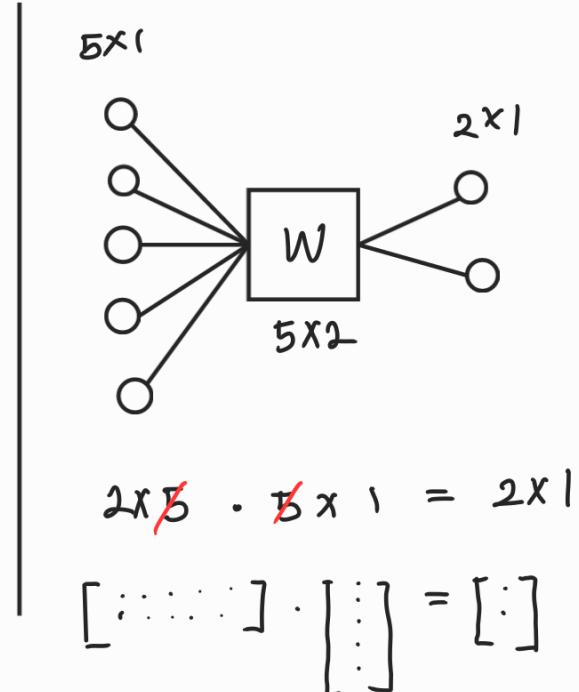
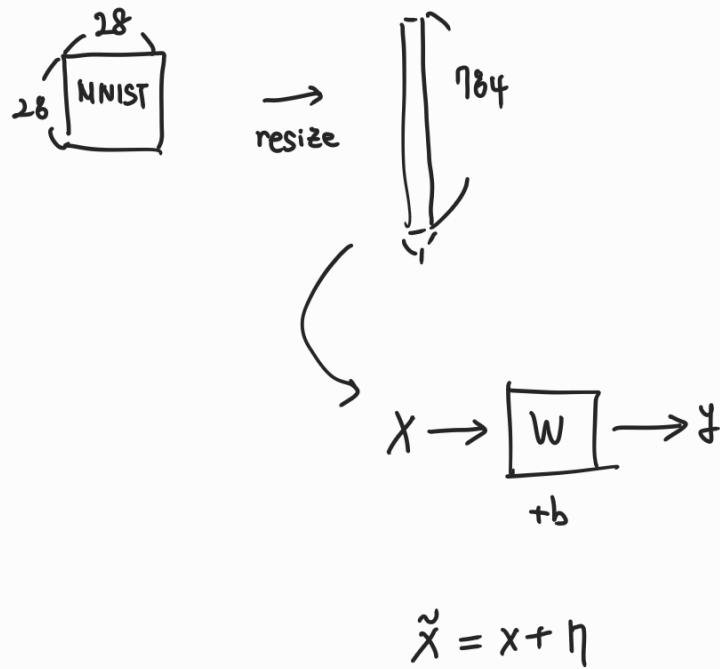


FGSM



$$w^T \tilde{x} = w^T x + w^T \eta \quad (\eta = \text{sign}(w))$$

↳ w 의 부호를 따라 부여

Ex) 일반적인 $w^T x$

$$\begin{array}{c} 7 \\ 3 \\ 2 \\ -5 \\ -3 \end{array} \quad \begin{matrix} 1 & -2 & 1 & -3 & 1 \\ 5 \times 1 \end{matrix} \quad \begin{array}{c} \\ \\ \\ \\ \end{array}$$

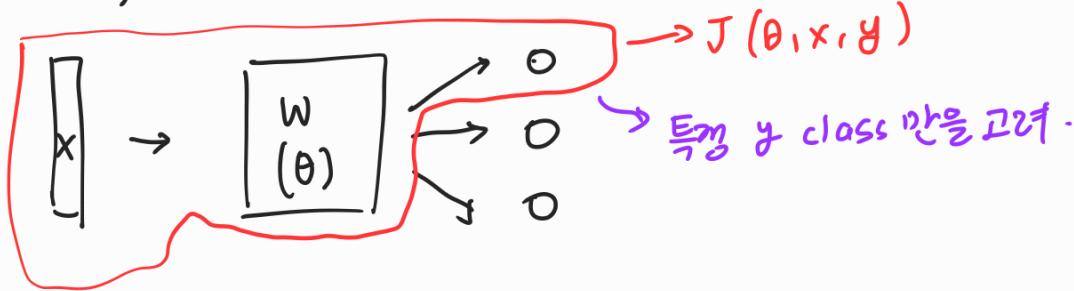
$$\begin{array}{c} 7 \text{ O} +1 \\ 3 \text{ O} -1 \\ 2 \text{ O} +1 \\ -5 \text{ O} -1 \\ -3 \text{ O} +1 \end{array} \quad \begin{matrix} \downarrow \text{ 애너들의 부호가 양수면} +1 \\ \text{ 음수면} -1 \end{matrix} \quad \begin{matrix} 1 & -2 & 1 & -3 & 1 \\ 5 \times 1 \end{matrix} \quad \begin{array}{c} \\ \\ \\ \\ \end{array}$$

$$\begin{aligned} & (7 \times 1 + 3 \times (-2) + 2 \times 1 \\ & + (-5) \times (-3) + (-3) \times 1) \\ & = (7 - 6 + 2 + 15 - 3) = 15 \end{aligned}$$

$$\begin{aligned} & (8 \times 1 + 2 \times (-2) + 3 \times 1 + (-6) \times (-3) \\ & + (-2) \times 1) \\ & = (8 - 4 + 3 + 18 - 2) \\ & = 23 \end{aligned}$$

$\eta = \text{sign}(w)$ 으로만 해줘도 activation 값이
매우 증가 ~ 실제로는 weight값을 따라 증가
weight값 차원 (예제는 5)

$$J(\theta, x, y)$$

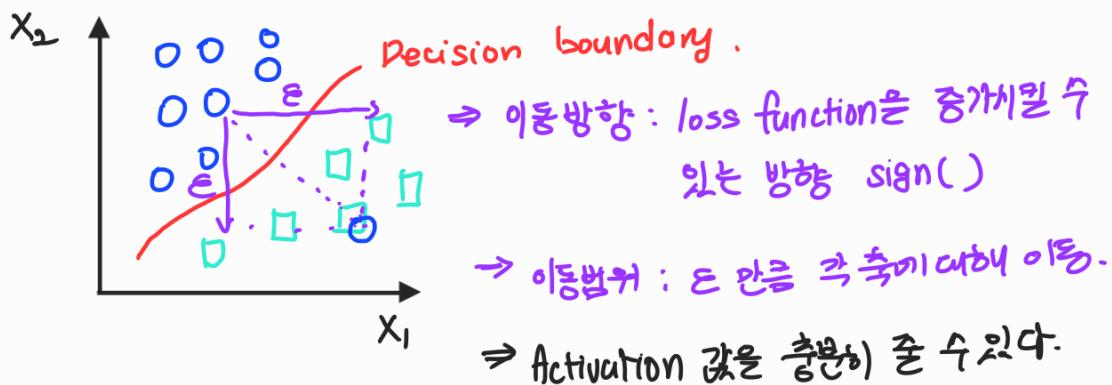


$$\eta = \varepsilon \text{sign} (\nabla_x J(\theta, x, y))$$

∇ cost function을 증가시키는 방향으로

ε 만큼 이동

perturbation을 만들.



labels : $y \in \{-1, 1\}$

$$\ell(z) = \log(1 + \exp(z))$$

softplus function

→ loss function으로 사용
gradient descent를 위해 미분 적용

$$E_{x,y \sim P_{\text{data}}} \ell(-y(w^T x + b))$$

\Rightarrow \text{loss function}

$$(\ln f(x))' = \frac{f'(x)}{f(x)}$$

$$(\log(1 + \exp(-y(w^T x + b))))' \Rightarrow \frac{(1 + \exp(-y(w^T x + b)))'}{1 + \exp(-y(w^T x + b))}$$

> 0

$$\frac{\exp(-y(w^T x + b)) \cdot -y w^T}{1 + \exp(-y(w^T x + b))}$$

> 0

이 gradient의 sign값을 알아야 할 필요가 있음.

sign값은 $-y w^T$ 에 영향을 받게됨.
 $\Rightarrow -y \cdot \text{sign}(w)$

즉 loss function을 x로 이분한 것에 대해

sign function은 불안정하게 작용함.

\Rightarrow perturbation (sign of gradient)

$$E_{x,y \sim P_{\text{data}}} \zeta(y(\varepsilon \|w\|_1 - w^T x - b))$$

$$E_{x,y \sim P_{\text{data}}} \zeta(-y(w^T x + b))$$

$x \rightarrow x - y \cdot \text{sign}(w) \times \varepsilon$

$$-\zeta(w^T(x - y \cdot \text{sign}(w)) + b)$$

$$= -y(w^T x - w^T y \cdot \text{sign}(w) + b)$$

$$\ell(-y(w^T x + b) + \|w\|_1 \times \varepsilon) \Rightarrow \text{adversarial training을 위한 loss function.}$$

Adv Training: $\ell(-[y(w^T x + b) - \varepsilon \|w\|_1])$

→ 이 2개를 비교.

L1 weight decay: $\ell(-(y(w^T x + b)) + \lambda \|w\|_1)$

* Adv Training의 loss function의 경우 activation function penalty를 배제는 것.

⇒ penalty가 사라지는 경향

* Weight decay의 경우 상대적으로 damage를 과대평가하는 경향이 있어서

은값을 상대적으로 작게 설정할 필요가 있습니다.

Ex) Adv Training : $\varepsilon \rightarrow 0.25$ 잘됨.

Weight decay : $\varepsilon \rightarrow 0.0025$ 도쁘다

[Adversarial Training의 Loss Function]

$$\tilde{J}(\theta, x, y) = \underbrace{\alpha J(\theta, x, y)}_{\text{일반적인 data에 대한 loss function}} + (1-\alpha) \underbrace{J(\theta, x + \epsilon \text{sign}(\nabla_x J(\theta, x, y)))}_{\text{adversarial example에 대한 loss function 의미.}}$$

$\alpha = 0.5$ 로 지정해서 학습시킨다고 함(은근에 있는)

기존에 AE에 대해 89.4%의 error rate를 보였던 model에 대해 adversarial training을 시켰더니 error rate가 17.9%로 떨어졌다. 단, 잘못 분류하는 경우 높은 confident를 갖고 잘못 분류했다는 단점이 있다.

CW - Attack.

$t \neq C^*(x) \rightarrow$ correct label과 다르게 target 값을 잡아준다.

$C(x') \neq C^*(x) \rightarrow$ perturbated 된 x 가 correct label로 분류되지 않고 x 와 x' 가 최대한 비슷하도록.
(동일하게 보이도록).

A. L-BFGS

minimize $\|x - x'\|_2^2 \rightarrow x$ 와 x' 의 차이는 최소화

such that $C(x') = l \rightarrow$ target인 l 로 분류할 수 있는 것.

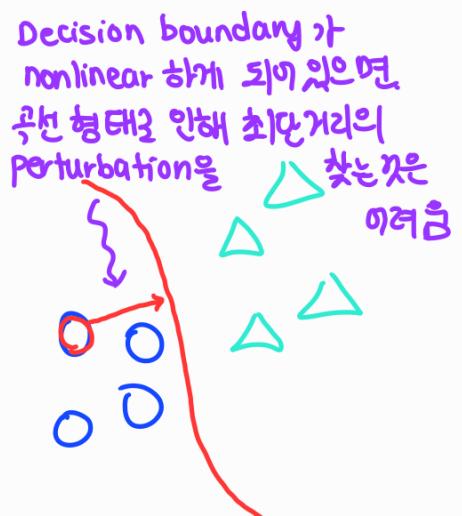
$$x' \in [0,1]^n$$

변형 ↘ term의 비율을 조절하기 위한 상수.

$$\text{minimize } c \cdot \|x - x'\|_2^2 + \text{loss}_{F, l}(x')$$

such that $x' \in [0,1]^n \rightarrow$ target class l 로 가는 loss function을 줄여나감.

cross-entropy loss로 계산.



L-BFGS는 시간이 오래 걸리지만 성공적으로 AE를 만들게 된다.

효과적인 C값을 찾기 위해 이분탐색과 같은 방식을 이용.

B. Fast Gradient Sign

$x' = x - \epsilon \operatorname{sign}(\nabla \text{loss}_{F,t}(x)) \rightsquigarrow \text{loss 를 높일 수 있는 sign 방향으로 }$
은 만큼 이동.

모든 pixel 을 동시다발적으로 이동.

Iterative Gradient Sign: Kurakin

optimal 한 방법은 X. 한 번의 기울기로
이동 가능.

$x'_0 = 0 \rightsquigarrow 0$ 부터 perturbation 시작.

$$x'_i = x'_{i-1} - \text{clip}_\epsilon(d \cdot \operatorname{sign}(\nabla \text{loss}_{F,t}(x'_{i-1})))$$

\downarrow
step size

여기 step size d라는 step size 만큼 update.

$\text{clip}_\epsilon \Rightarrow$ 이미지의 범위를 벗어난 경우 clip으로 지워짐

\Rightarrow FGSM 보다 성능 좋음.

C. JSMA

gradient $\nabla z(x)$ 을 이용해서 saliency map 을 만들어냄.

영향력이 가장 큰 pixel 을 찾아내면 특정 class로 변경될 수 있도록 pixel 을 변경

Jacobian matrix : 여러개의 변수, 함수가 있을 때 모든 함수에 대해
gradient 값을 계산하는 것을 의미.

$$\left[\begin{array}{c} \frac{\partial F_1}{\partial x_1}, \frac{\partial F_1}{\partial x_2}, \dots, \frac{\partial F_1}{\partial x_n} \\ \vdots \quad \vdots \quad \ddots \quad \vdots \\ \frac{\partial F_m}{\partial x_1}, \dots, \frac{\partial F_m}{\partial x_n} \end{array} \right] \rightarrow \text{첫번째 함수를 } x_1 \sim x_n \text{ 에 대해 이분}$$

JSMA - Z : logit 값을 기준으로 gradient 계산

JSMA - F : Softmax function 값을 기준으로 gradient 계산.

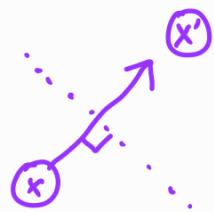
D. Deepfool

L_2 norm을 기반으로 한 attack.

Decision boundary가 선형적으로 되어 있는 부분이 있으면

기기에 벡터의 방향으로 AE를 만드는 것.

JSMIA 보다 빠르면서 FGSM 보다 성능 좋음. 근데 PGD Attack 보다는 별로.



CW Attack.

$$\text{minimize } D(x, x+\delta) \quad \text{such that } L_1, L_2, \dots, L_\infty \text{ perturbation.}$$

such that $C(x+\delta) = t \Rightarrow$ non linear 해석 해결하기 어려움.

$$x+\delta \in [0,1]^n$$

$f(x+\delta)$ 이라는 함수를 새로 정의해보

$f(x+\delta) \leq 0$ 이라면 $C(x+\delta) = t$ 로

target으로 자동 분류하도록 설정.

$$f_1(x') = -\text{loss}_{F,t}(x') + 1$$

$$f_2(x') = (\max_{i \neq t} (F(x')_i) - F(x')_t) +$$

$$f_3(x') = \text{softplus}(\max_{i \neq t} (F(x')_i - F(x')_t)) - \log(2)$$

$$f_4(x') = (0.5 - F(x')_t)^+$$

$$f_5(x') = -\log(2F(x')_t - 2)$$

$$f_6(x') = (\max_{i \neq t} (Z(x')_i) - Z(x')_t)^+$$

$$f_7(x') = \text{softplus}(\max_{i \neq t} (Z(x')_i - Z(x')_t)) - \log(2)$$

$$f_2(x') = \left(\max_{i \neq t} (F(x')_i) - F(x')_t \right)^+$$

이 둘의 값이 ≤ 0 이될 때 즉 target class에 대한 confidence가 매우 클 때 $f_2(x') \leq 0$ 이 됨.

\$\downarrow\$ softmax를 거친 이후
target class에 대한 confidence
label 중 가장 높은 confidence.

★

$$f_6(x') = \left(\max_{i \neq t} (z(x')_i) - z(x')_t \right)^+$$

\$\hookrightarrow\$ 애가 성능 가장 좋았음.

\$\downarrow\$ f_2 와 비슷하게 애는 target으로의 logit 값이 매우 크게 될 때. $f_6(x') \leq 0$ 이 됨

클수록 강한 공격이 들어감

$$\text{minimize } D(x, x + \delta)$$

$$\text{such that } C(x + \delta) = t$$

$$x + \delta \in [0, 1]^n$$



$$\text{minimize } D(x, x + \delta) + C \cdot f(x + \delta)$$

$$\text{such that } x + \delta \in [0, 1]^n$$

\$\Rightarrow\$ L-BFGS 와 유사.

but loss function good.

최종

\$\rightarrow\$ perturbation 최적화

$$\text{minimize } \|f\|_p + C \cdot f(x + \delta)$$

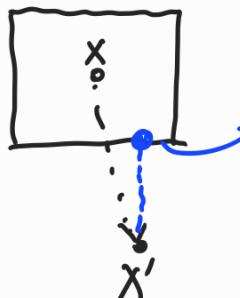
such that $x + \delta \in [0, 1]^n$

\$\Rightarrow\$ 목표: 최대한 작은 C값을 찾으면서 $f(x + \delta) \leq 0$ 으로 만드는 것.

binary search로 값을 찾음.

Box constraints

1) Projected gradient descent.



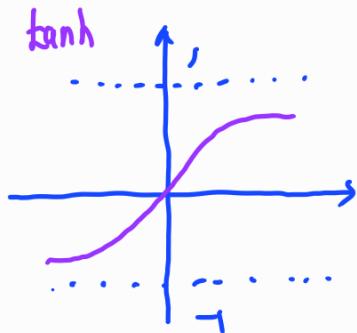
x' 이 움직여야 하는 범위를 벗어났을 때
projection을 통해 다시 안으로 끌어들임.

3) Change of variables \Rightarrow CW attack에서 사용하는 것.

$$\delta_i = \frac{1}{2}(\tanh(w_i) + 1) - x_i$$

\hookrightarrow (0과 1 사이의 범위로 조정)

$$0 \leq x_i + \delta \leq 1$$



(w) \rightarrow 를 이용해서 PGD 방식보다 깨끗하게
perturbation 줄 수 있으면 \Rightarrow smoothing.

VI. OUR THREE ATTACKS

A. L2 Attack

Change of variable 사용

$$\text{minimize } \|\frac{1}{2}(\tanh(w) + 1) - x\|_2^2 + C \cdot f(\underbrace{\frac{1}{2}(\tanh(w) + 1)}_{z(x')})$$

$$f(x') = \max(\max_i z(x')_i : i \neq t \} - z(x')_t, -k)$$

Multiple starting point
gradient descent

→ perturbation의 시작지점을
random 하기 때문에 bad
local minimum으로 빠지는것을
방지해 줌.

k 값이 커지면 $z(x')_t$ 에 대한

confidence가 더 많이 적용되며

$f(x') \leq 0$ 이 되면 attack이 들어간다.

k 값이 커질수록 강한 공격이 들어가게 된다.

B. L0 Attack

$$\|v\|_0 = \left(\sum_{i=1}^n |v_i|^0 \right)^{\frac{1}{0}} \rightsquigarrow \text{이분이 불가능}$$

→ 다른 방법으로 δ iteration마다 classifier에 영향을
끼치 안주는 pixel들을 선택해서 개체는 전하지 않도록
고정. 그러한 pixel들을 찾을 때 L0 attack을 이용.

Allowed set (변경 가능한 pixel들) 만 변경을
시작함. L0의 loss function 으로 사용.

▷ $f(x + \delta)$: gradient of objective function 이
작은 pixel을 선택해가며 allowed set을 min.

C. L_∞ Attack

L_0 와 같이 미분이 불가능.

$$\text{minimize } c \cdot f(x + \delta) + \|\delta\|_\infty$$

→ 변화가 가장 큰 축에 대해서만 penalty를 주어 작은 축에는 영향을 주지 못함.

$$\text{minimize } c \cdot f(x + \delta) + \cdot \sum_i [(d_i - r)^+]$$

→ 변화가 작은 축에도 영향을 주기 위해 식 변경.

Defensive Distillation evaluate.