Explaining and Harnessing Adversarial Examples

Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy
Google Inc., Mountain View, CA

Presented by Yeong Min Ko

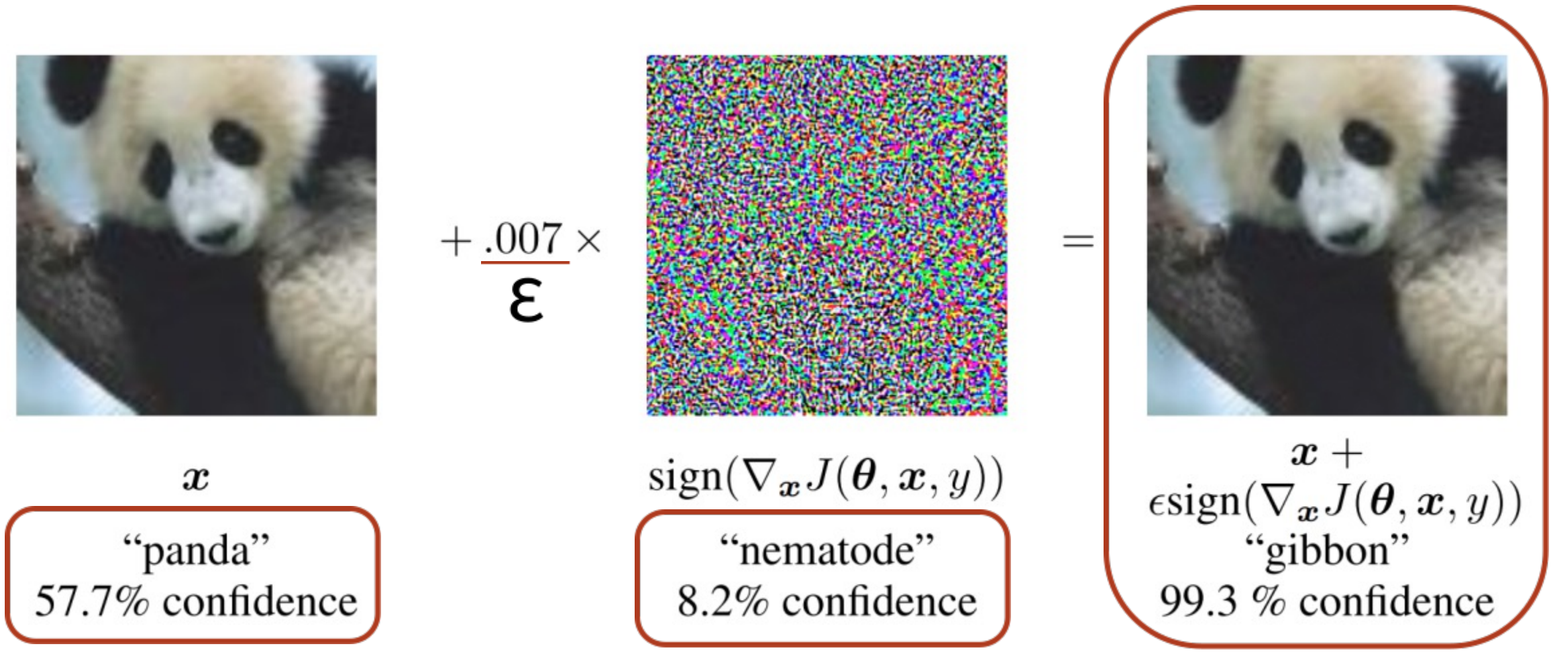
Pusan National University
Visual Intelligence and Perception Laboratory

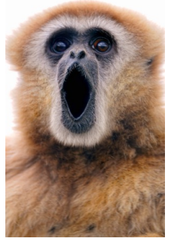
2024.09.04

_INDEX

I. Background

- **Prerequisite**
  - Adversarial Attack: 의도적으로 오분류를 이끌어내는 입력값(adversarial examples)을 만들어 내는 공격
  - Adversarial Examples: 모델 내에 perturbation(noise)이 삽입된 이미지





1. nematode: 선충(실처럼 가느다란 기생충)
2. gibbon: 긴팔원숭이

$$\boldsymbol{x}_{adv} = \boldsymbol{x}_{benign} + \varepsilon * sign(\nabla_{\boldsymbol{x}_{benign}} \boldsymbol{J}(\boldsymbol{\theta}, \boldsymbol{x}_{benign}, y))$$

- Why are they dangerous?

Figure 1: The left image shows real graffiti on a Stop sign, something that most humans would not think is suspicious. The right image shows our a physical perturbation applied to a Stop sign. We design our perturbations to mimic graffiti, and thus "hide in the human psyche."

- 도로교통 표지판에 작은 스티커를 붙여서 "정지" 신호를 "시속 45마일 속도 제한" 신호로 오분류하게 만드는 방법을 발견

Robust Physical-World Attacks on Deep Learning Models, CVPR 2018

I. Background

- Previously
  - The cause of these Adversarial Examples was a mystery.
  - Speculative explanations have suggested it is due to extreme nonlinearity of DNN, perhaps, combined with insufficient model averaging and insufficient regularization of the purely supervised learning problem.

- This work
  - Linear behavior in high-dimensional spaces is sufficient to cause adversarial examples.
  - This view yields a simple and fast method of generating adversarial examples.

FGSM(Fast Gradient Signed Method)

I. Background

- Related Work
  - **Szegedy et al. (2014b)* demonstrated a variety of intringuing properties of neural networks and related models.
  - Those most relevant to this paper include:
    - the adversarial examples were so close to the original examples that the differences were indistinguishable to the human eye.
    - Shallow softmax regression models are also vulnerable to adversarial examples.
    - Training on adversarial examples can regularize the model - however, this was not practical at the time due to the need for expensive constrained optimization in the inner loop.

- A modern wide variety of models with different architectures has built a Potemkin village.

* Szegedy et al., Intriguing properties of neural networks, ICLR(2014b)

II. Contribution

II. Contribution

1. Introduce a family of fast methods for generating adversarial examples.

2. Demonstrate that adversarial training can result in regularization, even more effectively than dropout.

3. Including L1 weight decay or adding noise is simple, but these experiments show that it can't reproduce the regularization effect better than adversarial training.

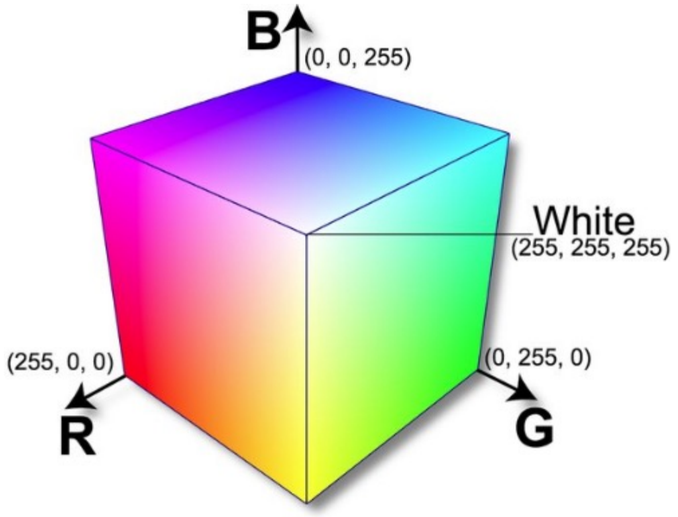
III. Method

i . The Linear Explanation of Adversarial Examples

ii . Linear Perturbation of Non-Linear Models

III. Method

1. The Linear Explanation of Adversarial Examples



In many problems, the precision of an individual input feature is limited. For example, digital images often use only 8 bits per pixel so they discard all information below $1/255$ of the dynamic range. Because the precision of the features is limited, it is not rational for the classifier to respond differently to an input $\boldsymbol{x}$ than to an adversarial input $\tilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\eta}$ if every element of the perturbation $\boldsymbol{\eta}$ is smaller than the precision of the features. Formally, for problems with well-separated classes, we expect the classifier to assign the same class to $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ so long as $||\boldsymbol{\eta}||_\infty < \epsilon$, where ϵ is small enough to be discarded by the sensor or data storage apparatus associated with our problem.

- Adversarial example

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\eta}$$

$$y = Wx + b$$

Weight Bias

Consider the dot product between a weight vector $\boldsymbol{w}$ and an adversarial example $\tilde{\boldsymbol{x}}$:

$$\boldsymbol{w}^\top \tilde{\boldsymbol{x}} = \boldsymbol{w}^\top \boldsymbol{x} + \boldsymbol{w}^\top \boldsymbol{\eta}.$$

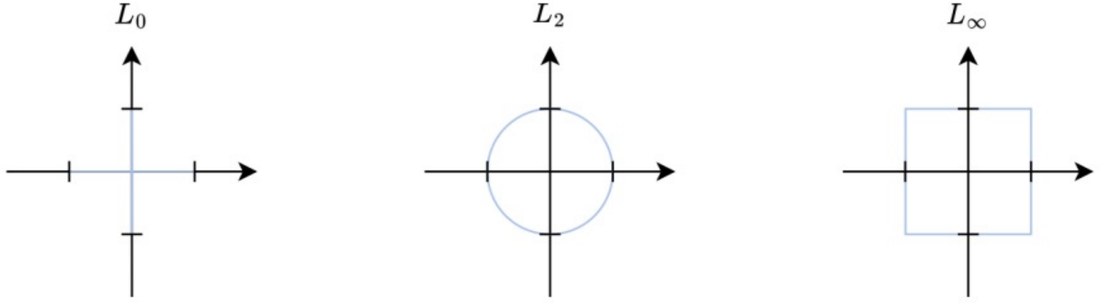
perturbation

$$\boldsymbol{w}^\top \tilde{\boldsymbol{x}} = \boldsymbol{w}^\top (\boldsymbol{x} + \boldsymbol{\eta})$$

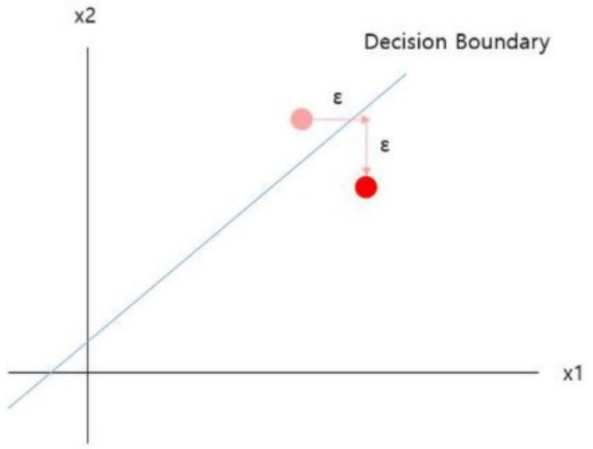
III. Method

1. The Linear Explanation of Adversarial Examples

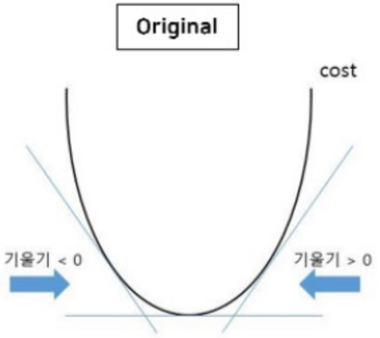
$$\tilde{x} = x + \eta \qquad ||\eta||_\infty < \epsilon$$



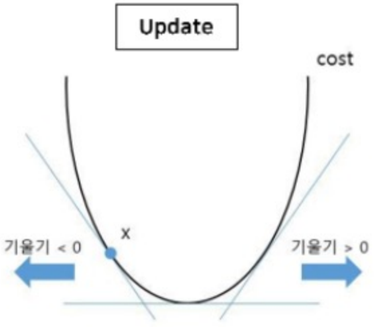
The adversarial perturbation causes the activation to grow by $w^\top \eta$.We can maximize this increase subject to the max norm constraint on η by assigning $\eta = \text{sign}(w)$. If w has n dimensions and the average magnitude of an element of the weight vector is m, then the activation will grow by ϵmn. Since $||\eta||_\infty$ does not grow with the dimensionality of the problem but the change in activation caused by perturbation by η can grow linearly with n, then for high dimensional problems, we can make many infinitesimal changes to the input that add up to one large change to the output.



[Max norm]



- Input X
- Cost가 최소화하는 방향



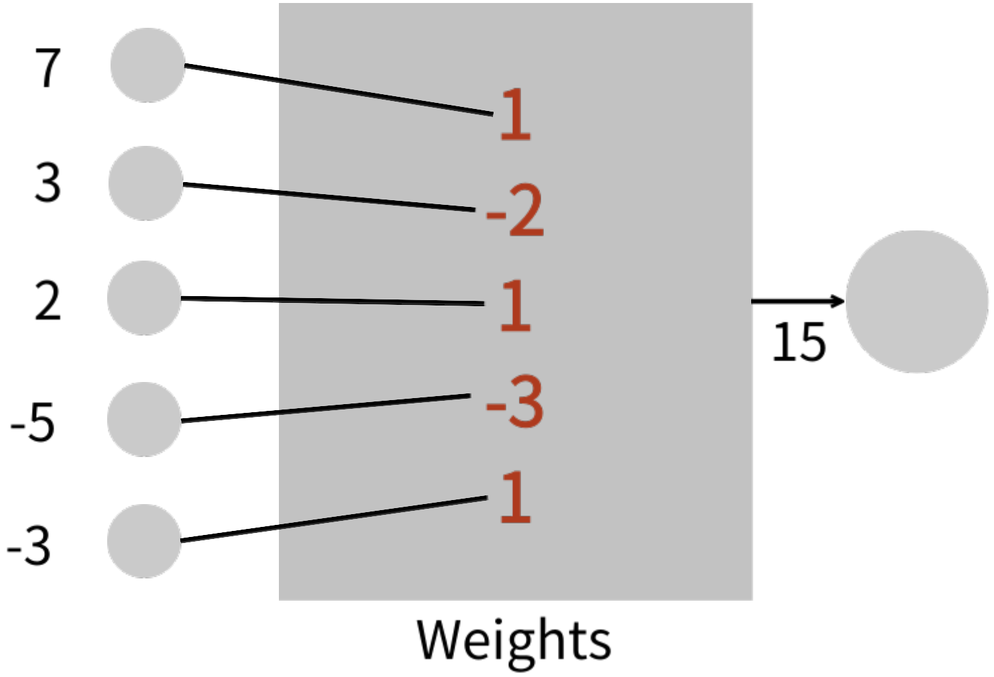
- Input X + noise
- Cost를 크게 하는 방향

III. Method

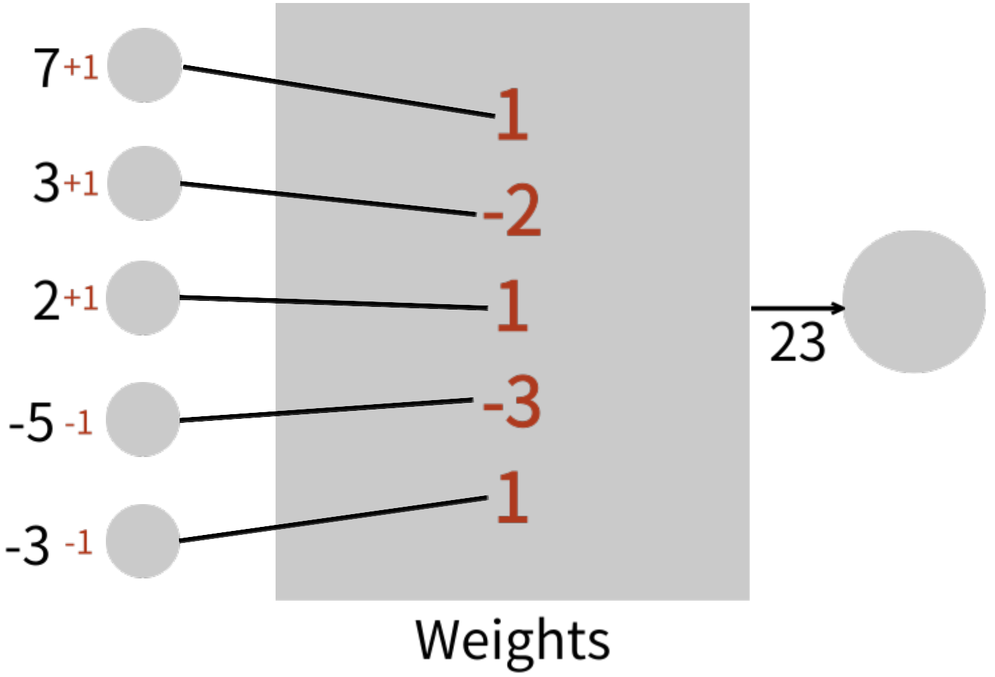
1. The Linear Explanation of Adversarial Examples

$$\text{sgn}(x) = \begin{cases} -1 & \text{for } x < 0 \\ 0 & \text{for } x = 0 \\ 1 & \text{for } x > 0. \end{cases}$$

$$\boldsymbol{\eta} = \epsilon \text{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right).$$



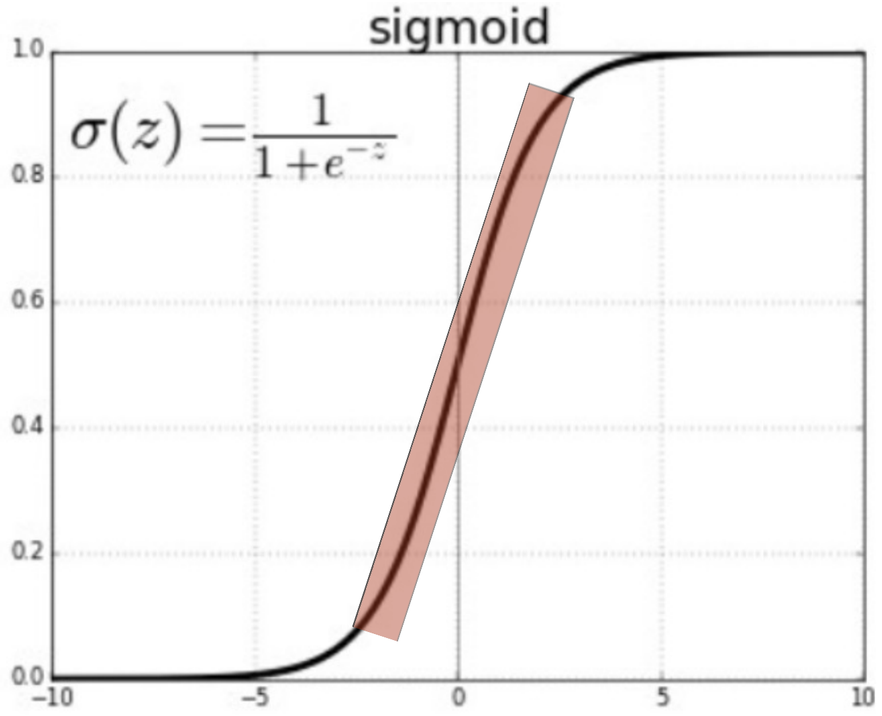
7 - 6 + 2 + 15 - 3 = **15**



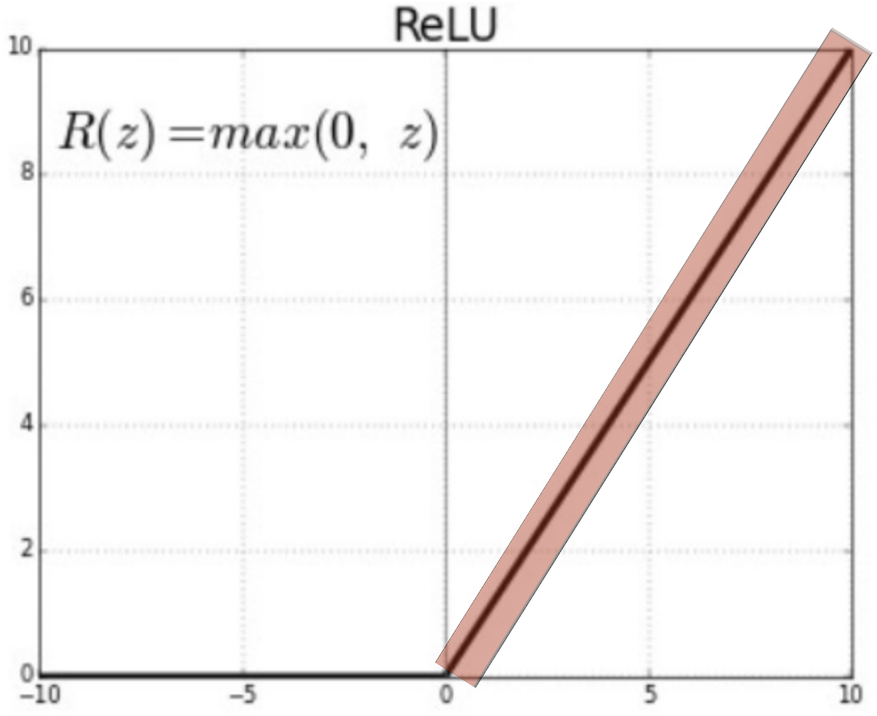
8 - 4 + 3 + 18 - 2 = **23**

III. Method

2. Linear Perturbation of Non-Linear Models



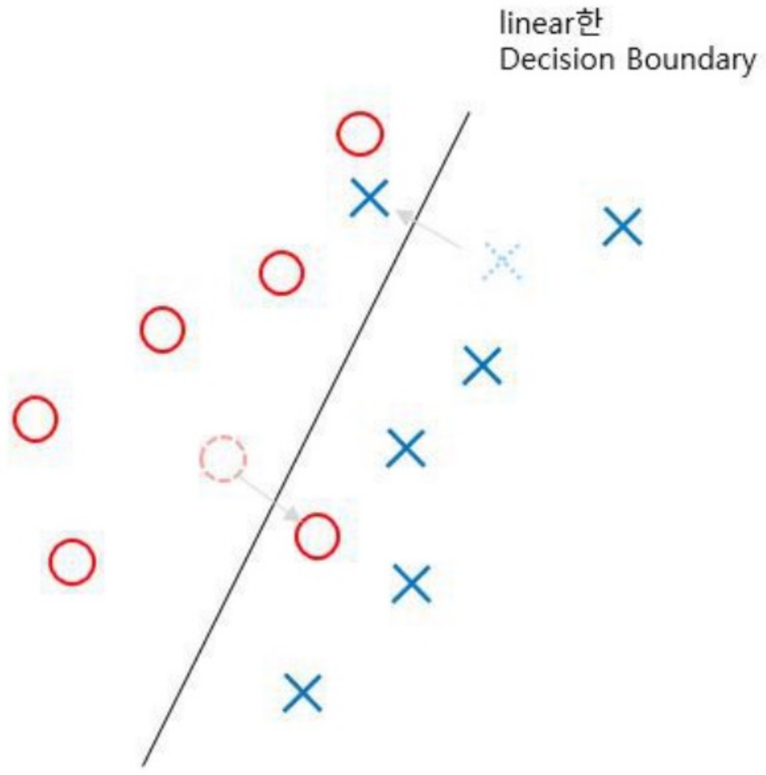
saturating

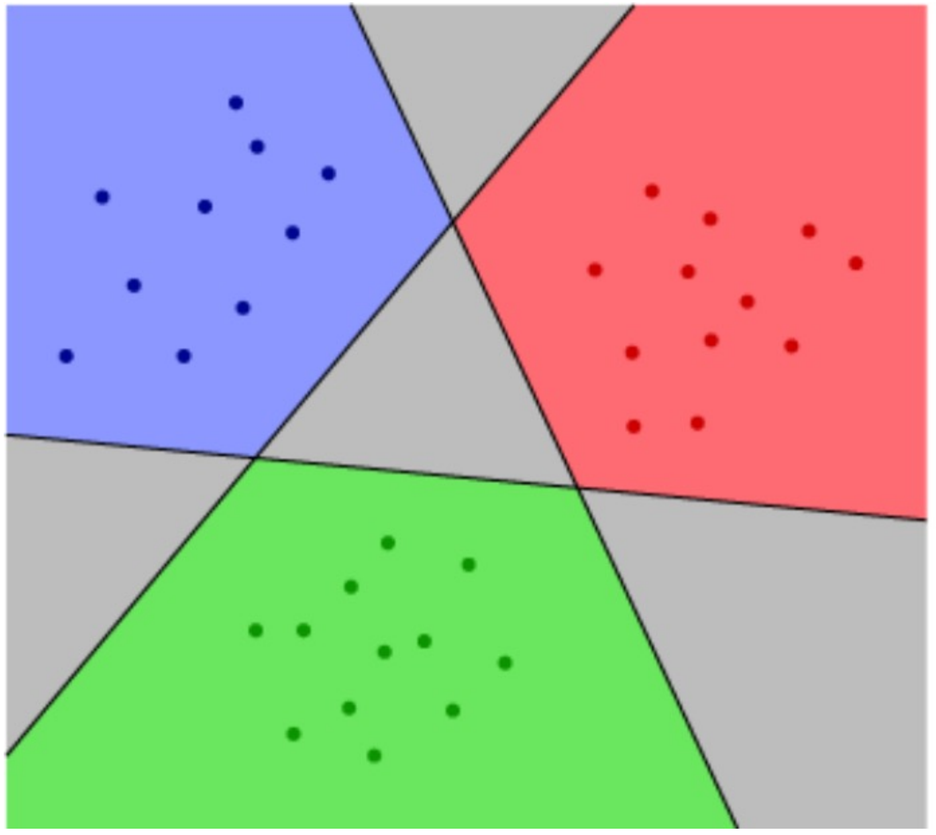


non-saturating

III. Method

2. Linear Perturbation of Non-Linear Models(FGSM)





III. Method

2. Linear Perturbation of Non-Linear Models(FGSM)

Let $\boldsymbol{\theta}$ be the parameters of a model, $\boldsymbol{x}$ the input to the model, y the targets associated with $\boldsymbol{x}$ (for machine learning tasks that have targets) and $J(\boldsymbol{\theta}, \boldsymbol{x}, y)$ be the cost used to train the neural network. We can linearize the cost function around the current value of $\boldsymbol{\theta}$, obtaining an optimal max-norm constrained pertubation of

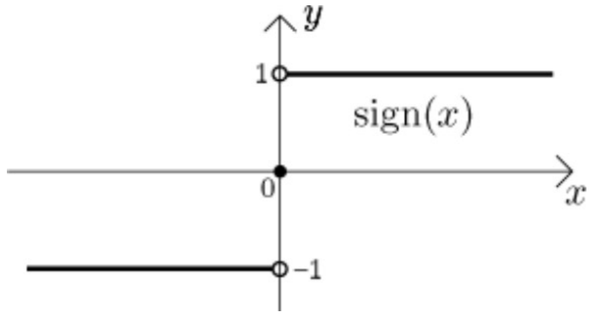
Fast Gradient Sign Method

$$\boldsymbol{\eta} = \epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right).$$

θ에 대한 cost function의 gradient

We refer to this as the "fast gradient sign method" of generating adversarial examples. Note that the required gradient can be computed efficiently using backpropagation.

$$\operatorname{sign}(x) = \begin{cases} -1 & \text{for } x < 0 \\ 0 & \text{for } x = 0 \\ 1 & \text{for } x > 0. \end{cases}$$



$$\boldsymbol{x}_{adv} = \boldsymbol{x}_{benign} + \varepsilon * sign(\nabla_{\boldsymbol{x}_{benign}} \boldsymbol{J}(\boldsymbol{\theta}, \boldsymbol{x}_{benign}, y))$$

III. Method

2. Linear Perturbation of Non-Linear Models(FGSM)

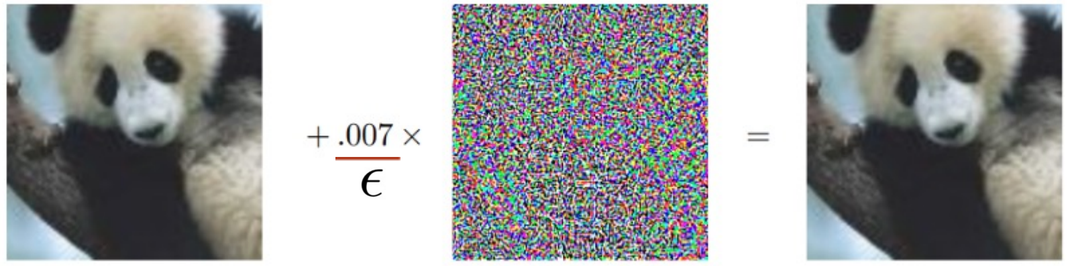
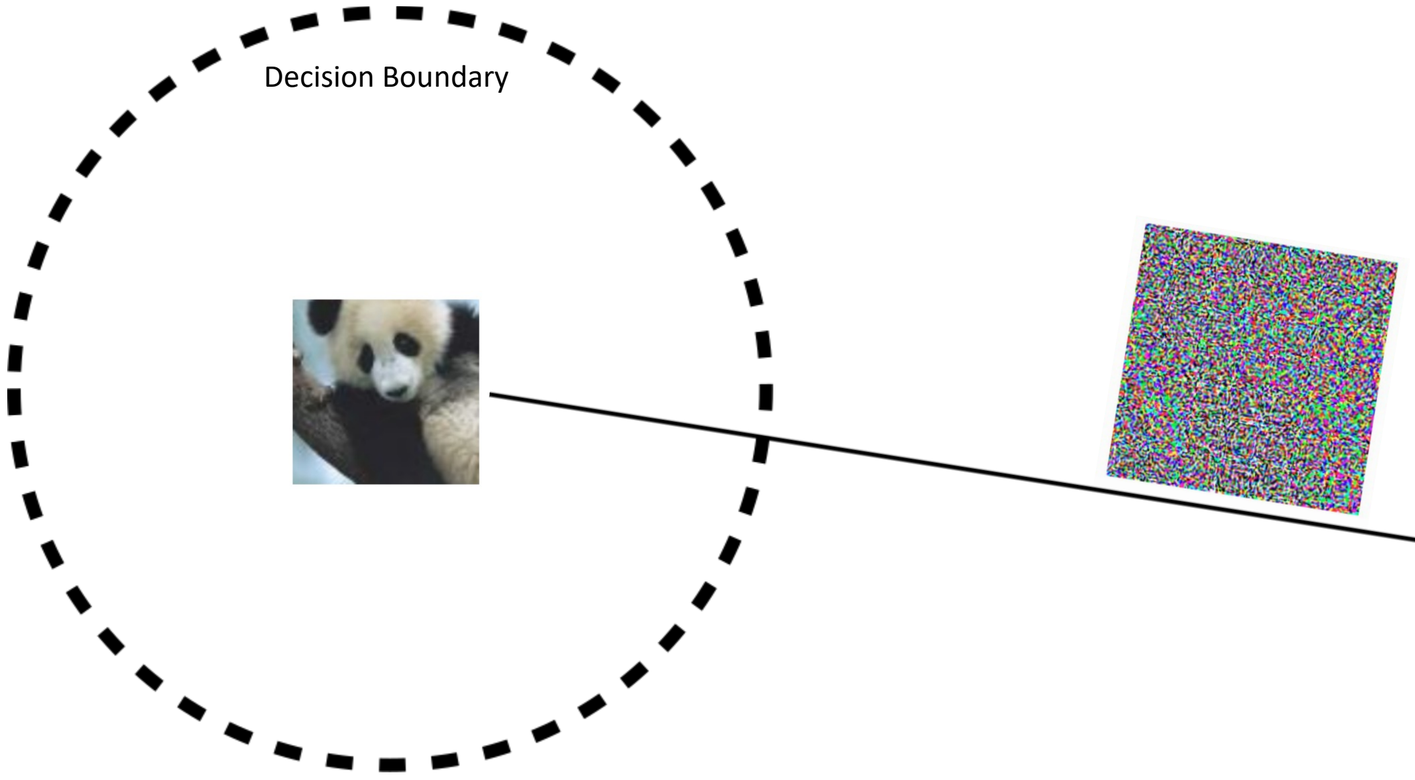


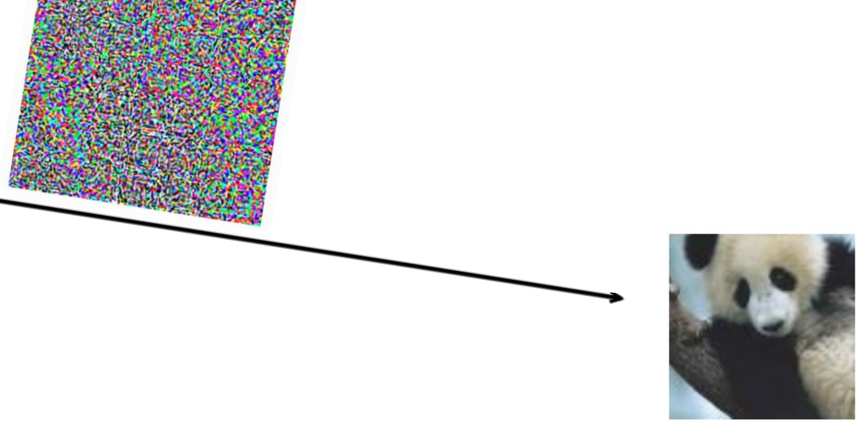
Figure 1: A demonstration of fast adversarial example generation applied to GoogLeNet (Szegedy et al., 2014a) on ImageNet. By adding an imperceptibly small vector whose elements are equal to the sign of the elements of the gradient of the cost function with respect to the input, we can change GoogLeNet's classification of the image. Here our ϵ of .007 corresponds to the magnitude of the smallest bit of an 8 bit image encoding after GoogLeNet's conversion to real numbers.

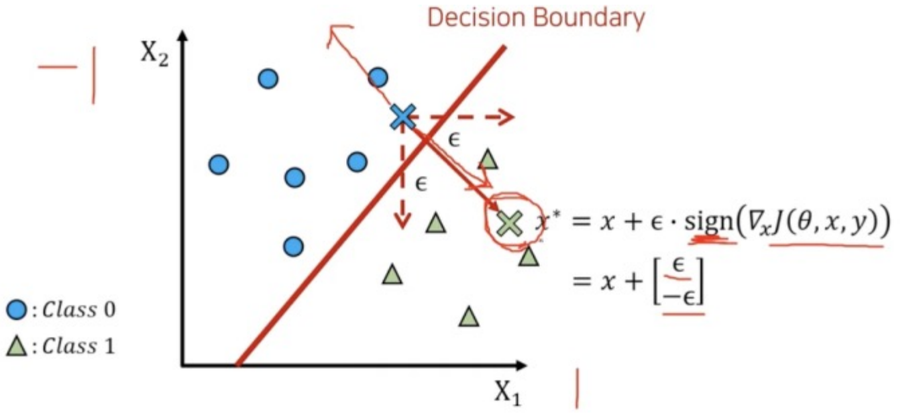
$$\boldsymbol{\eta} = \epsilon \text{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right).$$

III. Method

2. Linear Perturbation of Non-Linear Models(FGSM)

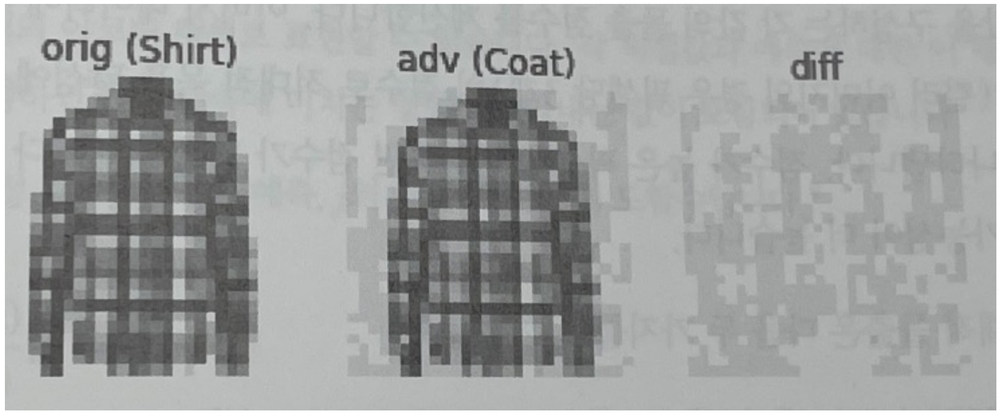






III. Method

2. Linear Perturbation of Non-Linear Models(FGSM)



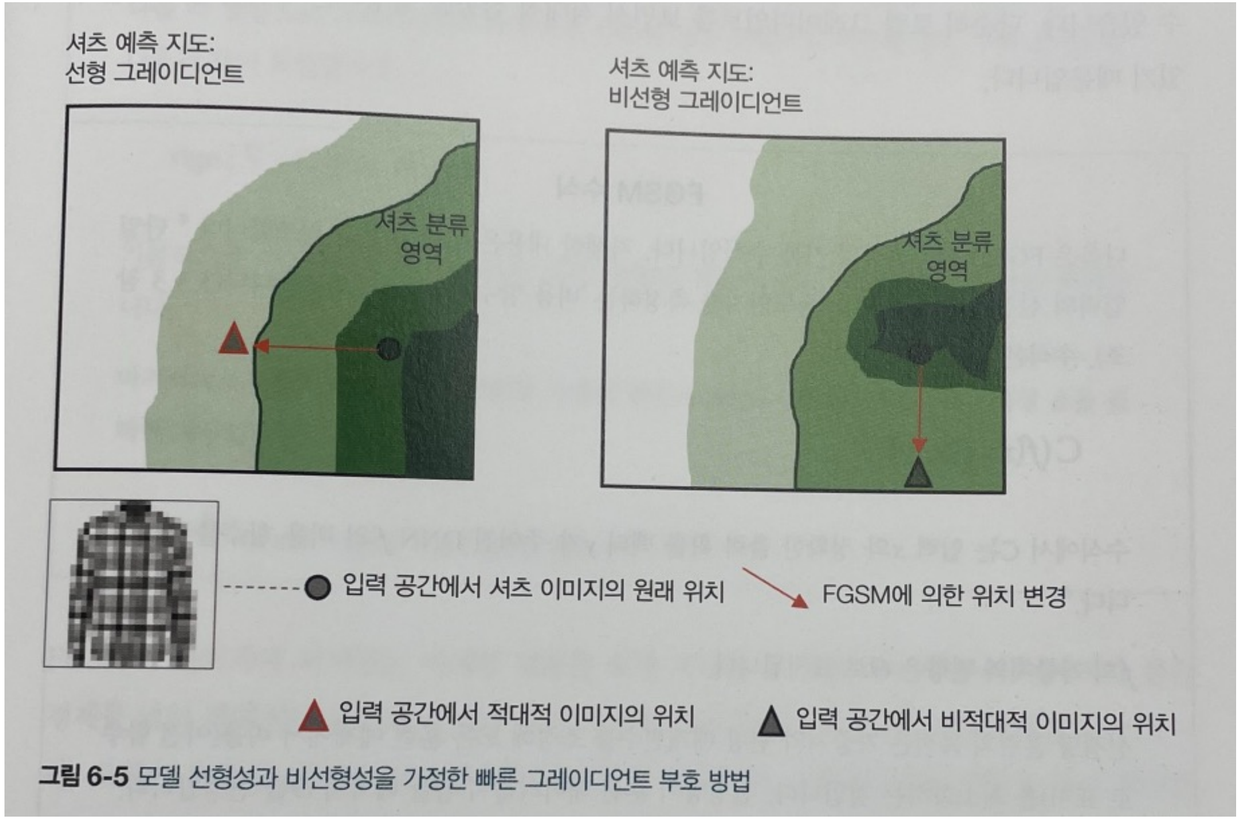


그림 6-5 모델 선형성과 비선형성을 가정한 빠른 그레이디언트 부호 방법

* 안전한 인공지능 시스템을 위한 심층신경망 강화, 한빛미디어, pp. 131, 137.

III. Method

2. Linear Perturbation of Non-Linear Models(FGSM)

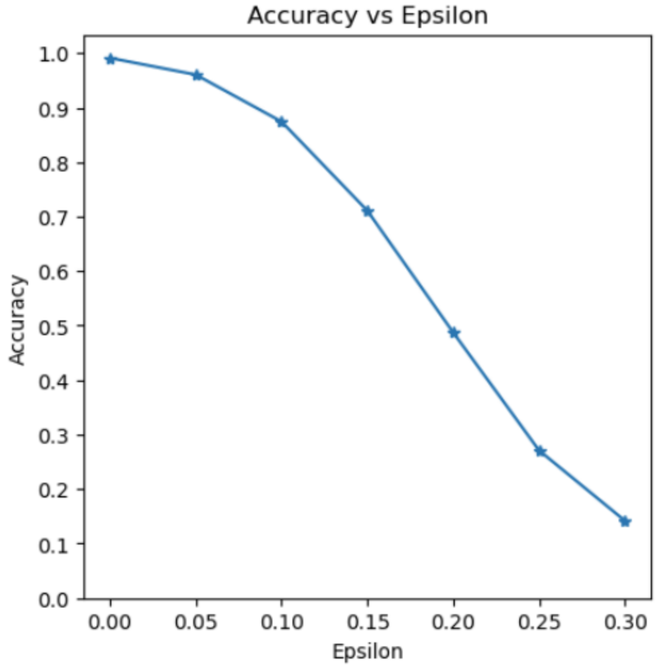
We find that this method reliably causes a wide variety of models to misclassify their input. See Fig. 1 for a demonstration on ImageNet. We find that using $\epsilon = .25$, we cause a shallow softmax classifier to have an error rate of 99.9% with an average confidence of 79.3% on the MNIST (**?**) test set[1]. In the same setting, a maxout network misclassifies 89.4% of our adversarial examples with an average confidence of 97.6%. Similarly, using $\epsilon = .1$, we obtain an error rate of 87.15% and an average probability of 96.6% assigned to the incorrect labels when using a convolutional maxout network on a preprocessed version of the CIFAR-10 (Krizhevsky & Hinton, 2009) test set[2]. Other simple methods of generating adversarial examples are possible. For example, we also found that rotating $\boldsymbol{x}$ by a small angle in the direction of the gradient reliably produces adversarial examples.

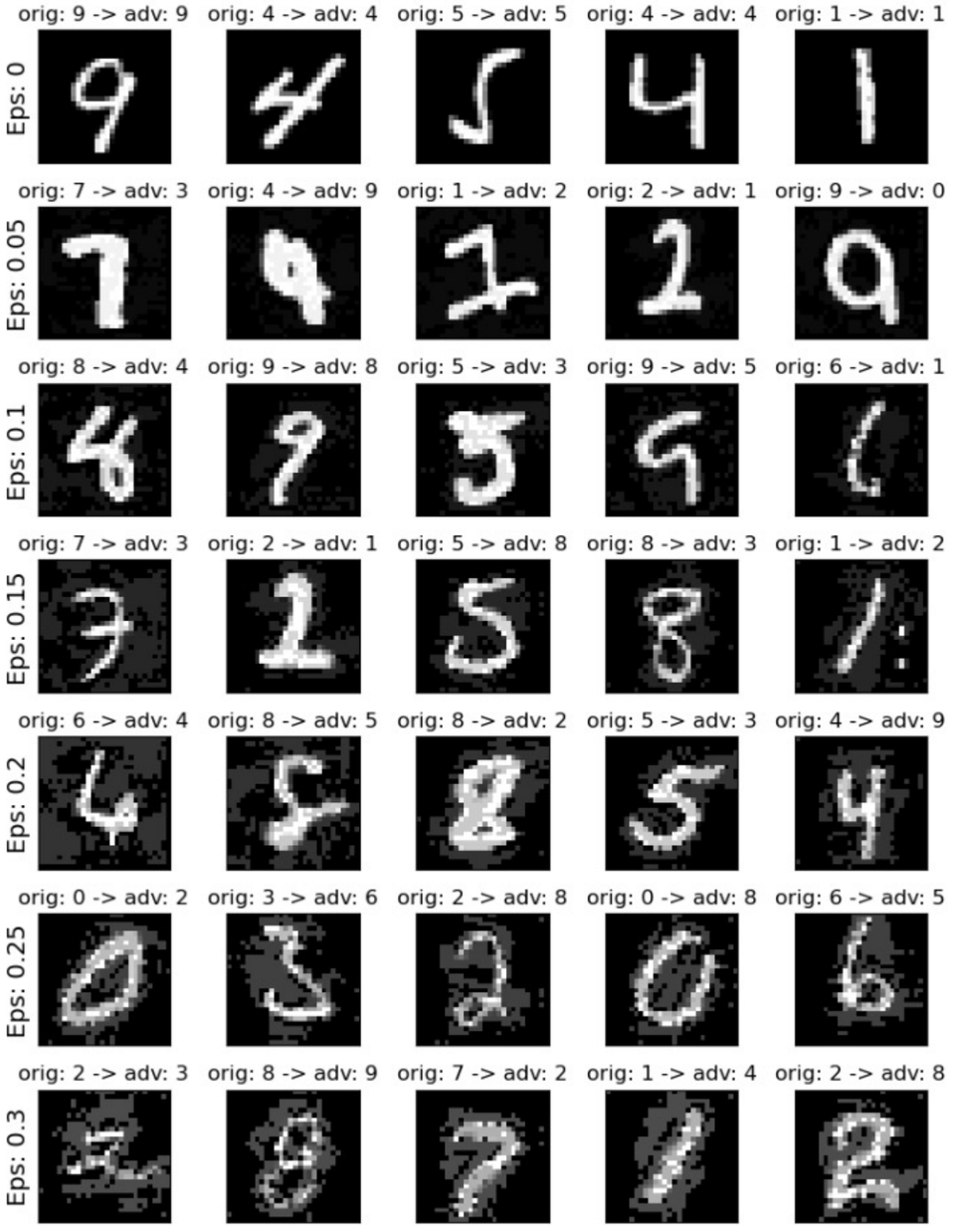
ε이 커질수록 오분류율 ↑

III. Method

2. Linear Perturbation of Non-Linear Models(FGSM)

```
Epsilon: 0      Test Accuracy = 9912 / 10000 = 0.99
Epsilon: 0.05   Test Accuracy = 9605 / 10000 = 0.96
Epsilon: 0.1    Test Accuracy = 8743 / 10000 = 0.87
Epsilon: 0.15   Test Accuracy = 7108 / 10000 = 0.71
Epsilon: 0.2    Test Accuracy = 4874 / 10000 = 0.49
Epsilon: 0.25   Test Accuracy = 2710 / 10000 = 0.27
Epsilon: 0.3    Test Accuracy = 1420 / 10000 = 0.14
```





- https://github.com/PSLeon24/Paper-Implementation-with-PyTorch/blob/main/FGSM/FGSM.ipynb

IV. Experiment

i. Adversarial Training of Linear Model Versus Weight Decay

ii. Adversarial Training of Deep Network

iii. Different Kinds of Model Capacity

iv. Why do Adversarial Examples Generalize?

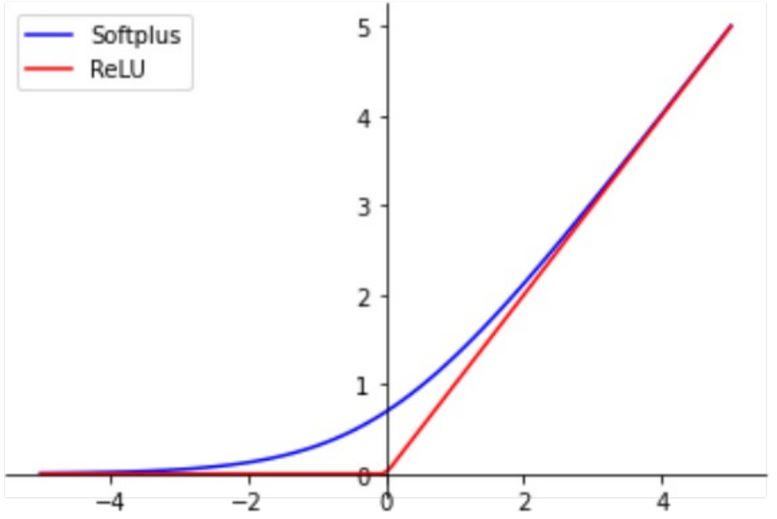
v. Alternative Hypothesis

IV. Experiment

1. Adversarial Training of Linear Model *Versus* Weight Decay

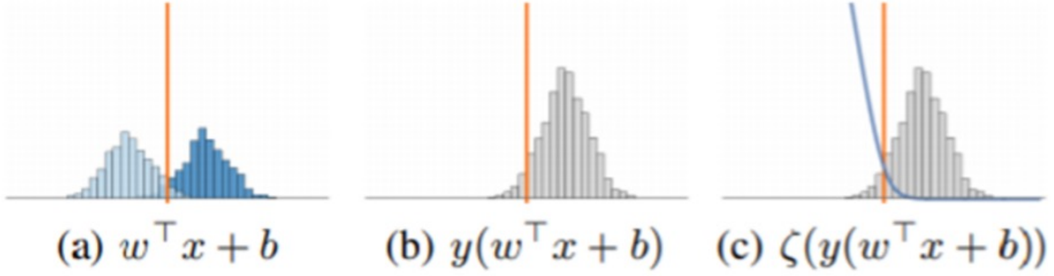
- **Softplus** is an activation function (smooth ReLU): $\mathrm{Softplus}(x) := \log(1 + e^x)$

  Softplus is strictly convex



$$\mathbb{E}_{\boldsymbol{x},y\sim p_{\text{data}}}\underbrace{\zeta(-y(\boldsymbol{w}^\top \boldsymbol{x}+b))}_{\text{softplus function}}$$

$$\mathbb{E}_{\boldsymbol{x},y\sim p_{\text{data}}}\zeta(y(\epsilon||\boldsymbol{w}||_1-\boldsymbol{w}^\top \boldsymbol{x}-b)).$$



IV. Experiment

1. Adversarial Training of Linear Model *Versus* Weight Decay

$$\mathbb{E}_{\boldsymbol{x},y\sim p_{\text{data}}}\zeta(y(\epsilon||\boldsymbol{w}||_1 - \boldsymbol{w}^\top \boldsymbol{x} - b)).$$

VS

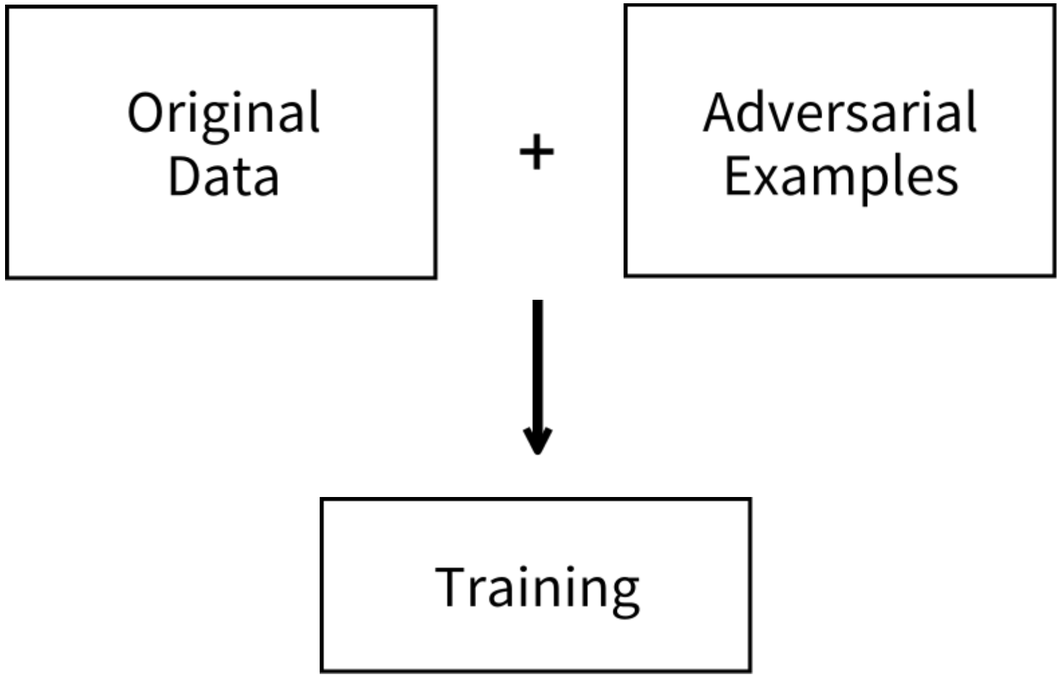
L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N}(y_i - \sum_{j=0}^{M} x_{ij}W_j)^2 + \lambda \sum_{j=0}^{M}|W_j|$$

IV. Experiment

2. Adversarial Training of Deep Network

$$\tilde{J}(\boldsymbol{\theta}, \boldsymbol{x}, y) = \alpha J(\boldsymbol{\theta}, \boldsymbol{x}, y) + (1 - \alpha) J(\boldsymbol{\theta}, \boldsymbol{x} + \epsilon \operatorname{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))) .$$



IV. Experiment

2. Adversarial Training of Deep Network

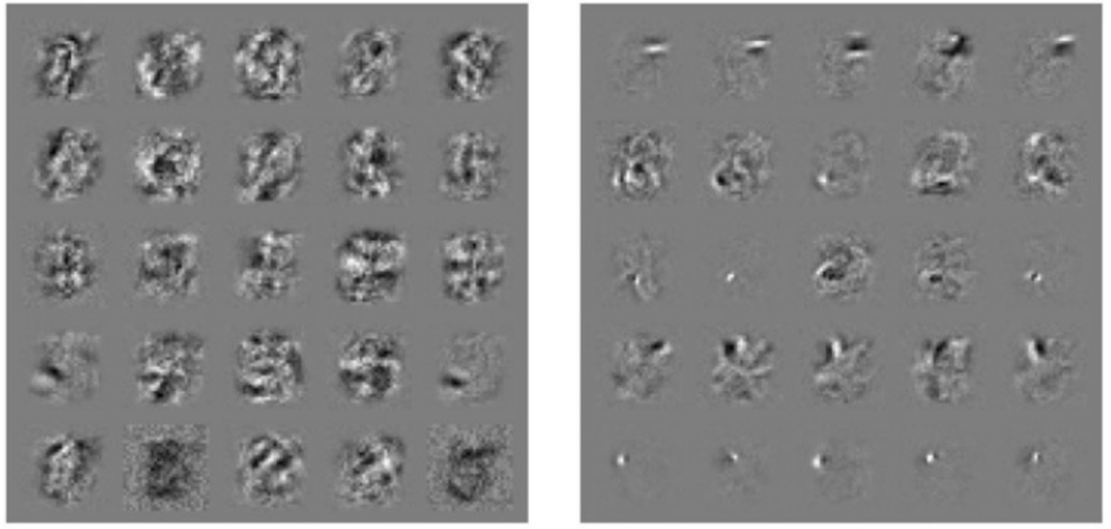


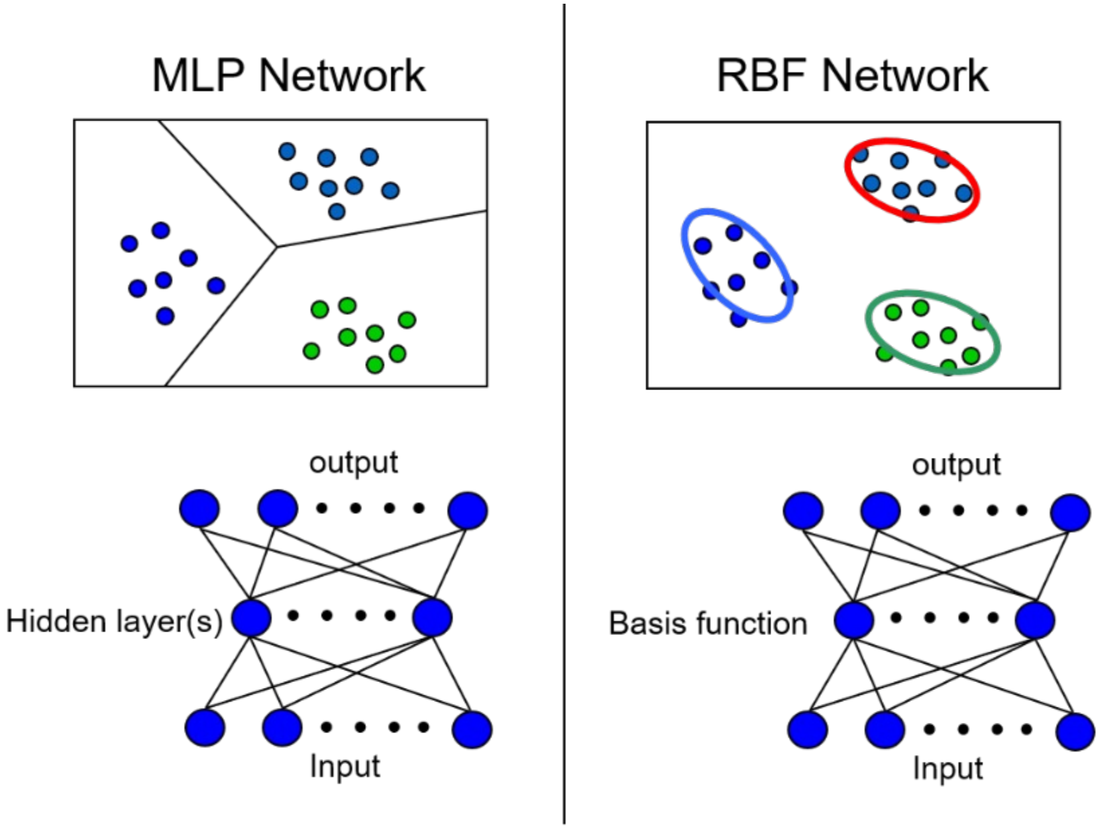
Figure 3: Weight visualizations of maxout networks trained on MNIST. Each row shows the filters for a single maxout unit. Left) Naively trained model. Right) Model with adversarial training.

IV. Experiment

3. Different Kinds of Model Capacity

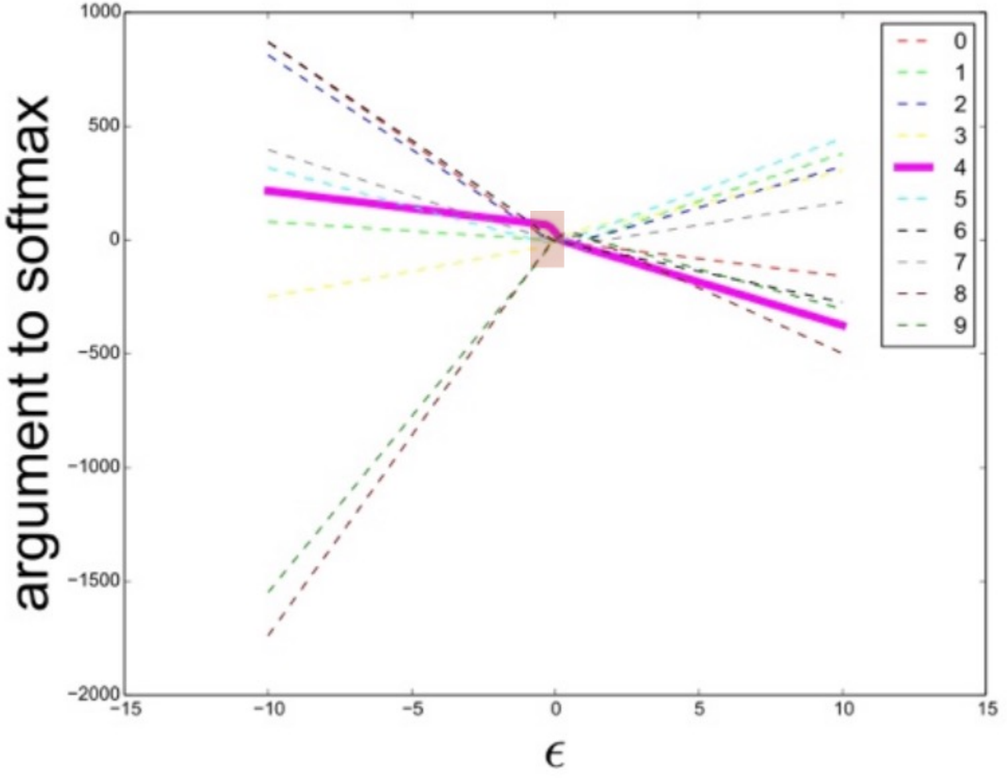
- Formula of a Shallow RBF(Radial Basis Function) Network, a Highly Non-linear Model

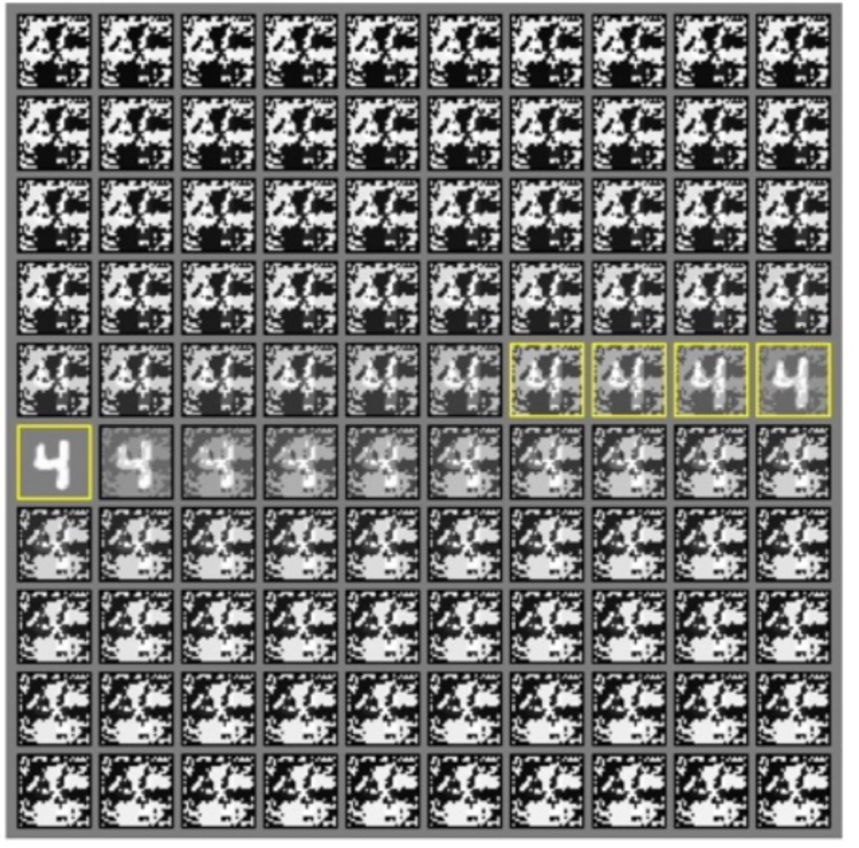
$$p(y = 1 \mid \boldsymbol{x}) = \exp\left((\boldsymbol{x} - \mu)^\top \boldsymbol{\beta}(\boldsymbol{x} - \mu)\right)$$



IV. Experiment

4. Why do Adversarial Examples Generalize?





transferability

IV. Experiment

5. Alternative Hypothesis

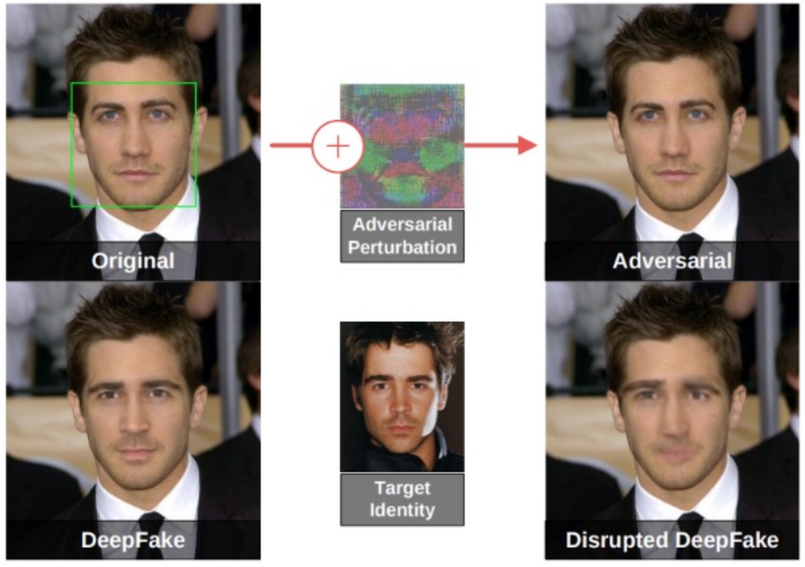
- **Hypothesis 1**: **The model gains confidence only on 'real' data** through generative training to distinguish between 'real' and 'fake' data.
  - **Test**: For the MNIST dataset, the MP-DBM model (which had an error rate of 0.88% on clean data) was evaluated. When subjected to adversarial perturbations with $\epsilon = 0.25$, the error rate dramatically increased to 97.5%.
  - **Conclusion**: This hypothesis is incorrect because the model, even when trained to distinguish real from fake data, is still highly vulnerable to adversarial examples.

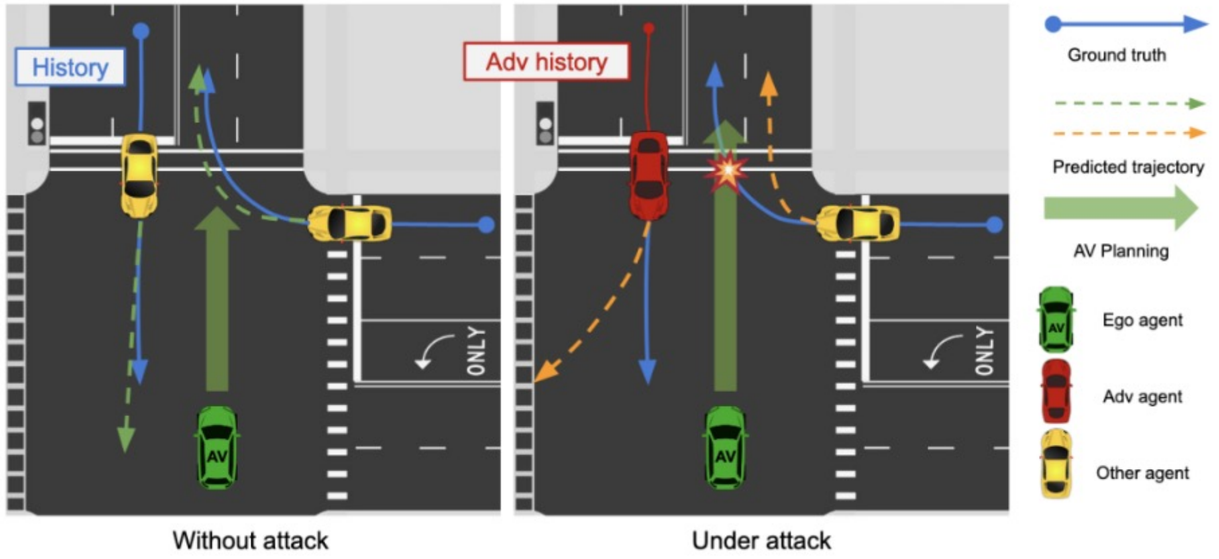
- **Hypothesis 2**: Averaging the weights of multiple models **could make adversarial examples disappear**.
  - **Test**: An ensemble of 12 maxout networks trained on the MNIST dataset was tested. Despite this ensemble approach, when $\epsilon = 0.25$, the error rate remained high at 91.1%.
  - **Conclusion**: This hypothesis is also incorrect because adversarial examples remain effective even when using an ensemble of models.

V. Conclusion

V. Conclusion

- Adversarial examples are a widespread phenomenon in machine learning, occurring not due to model overfitting but because of the model's linearity in high-dimensional spaces.
- These examples highlight a fundamental property of neural networks: they can be easily fooled by small, purposeful perturbations that are imperceptible to humans.
- The study introduced the Fast Gradient Sign Method (FGSM), which efficiently generates adversarial examples and can be used for adversarial training.
- Adversarial training enhances model robustness, making neural networks more resilient to these types of attacks.

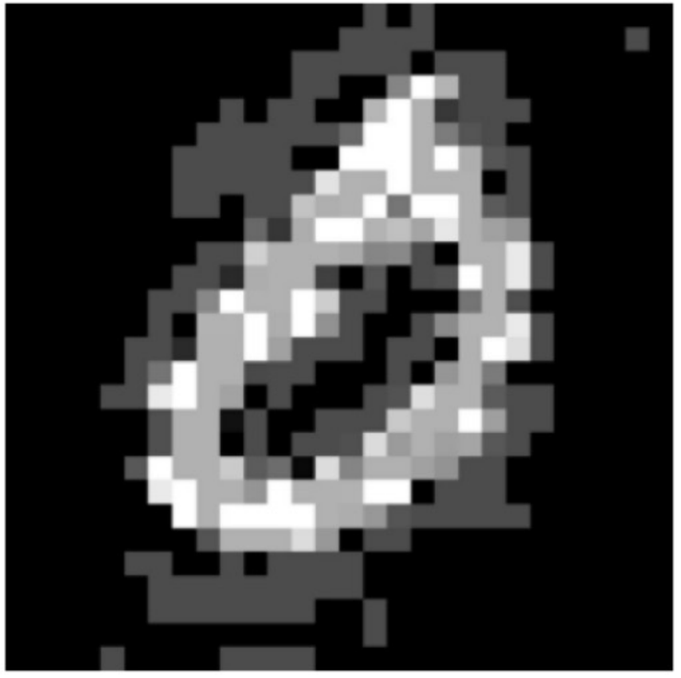

Original
DeepFake
Adversarial Perturbation
Target Identity
Adversarial
Disrupted DeepFake


History
Adv history
ONLY
AV
Without attack
Under attack
Ground truth
Predicted trajectory
AV Planning
Ego agent
Adv agent
Other agent

```
image_path = "noised.jpg"
classify_noisy_image(model, device, image_path)
```

```
Predicted label for the noisy image: 5, Confidence: 0.1075
```

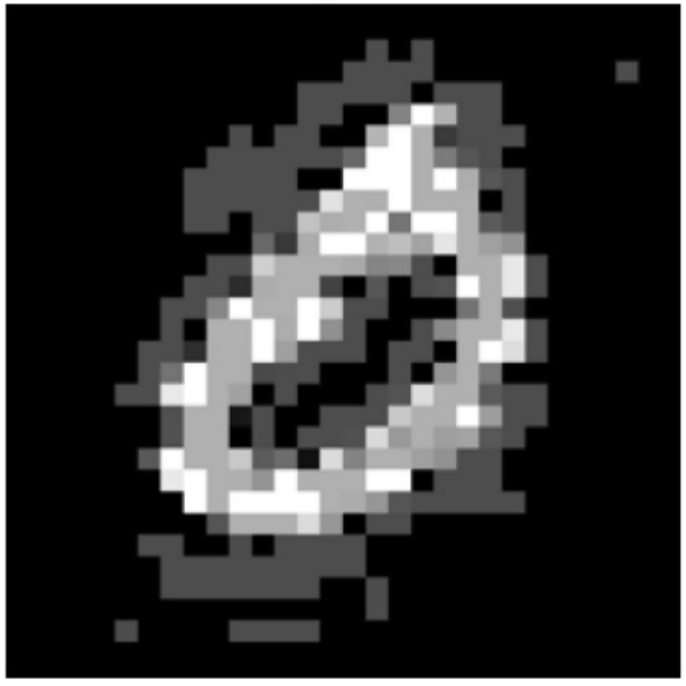
Predicted Label: 5, Confidence: 0.1075



```
image_path = "test.png"
classify_noisy_image(model, device, image_path)
```

```
Predicted label for the noisy image: 9, Confidence: 0.1077
```

Predicted Label: 9, Confidence: 0.1077



감사합니다.

2024.09.04