Data Science Lab Project 3 (IASD 2024-2025) : Adversarial Attacks and Defense

Nan AN Lucas Henneçon Hangyue Zhao

1. Main method summary

In this lab, we implemented the following:

•**Attacks:** FGSM , PGD L_∞, PGD L_2, Carlini & Wagner attack, Attacks with Momentum

•**Defenses:**Adversarial training with different types of attacks, Lipschitz regularization, Randomized smoothing, Jacobian regularization, Mixed Adversarial Training

We find some interesting facts in adversarial training research: increasing the complexity or diversity of attack types does not necessarily improve robustness against a specific attack.

2. Adversarial Training combine different attack methods

Adversarial training can effectively combine different attack methods because each attack generates diverse adversarial examples that represent various types of vulnerabilities in the model. (3)These adversarial examples serve as augmented training data. By incorporating examples from attacks like FGSM, PGD, and C&W, adversarial training exposes the model to a broader range of adversarial patterns, making it more resilient to unseen attacks and enhancing its overall robustness.

3. Improvements in defense mechanisms

3.1. Lipschitz Regularization

In the context of adversarial robustness, a lower Lipschitz constant ensures that small perturbations in the input space (e.g., adversarial noise) do not lead to disproportionately large changes in the output predictions.

In this experiment, spectral normalization is applied to enforce Lipschitz constraints on each layer of the network. By limiting the magnitude of layer-wise transformations, Lipschitz regularization reduces the model's vulnerability to adversarial examples.

3.2. Randomized Smoothing

Randomized smoothing transforms a base classifier f into a robust classifier g by averaging predictions over random noise applied to the input. Smoothed classifier is defined as:

$$g(x) = \arg\max_c \mathbb{P}(f(x+\delta) = c),$$

where δ is sampled from a predefined noise distribution. In this experiment, Gaussian noise is added to the inputs during both training and testing.

3.3. Jacobian Regularization

For a neural network $f(x;\theta)$, the Jacobian matrix $J_f(x)$ is defined as:

$$J_f(x) = \frac{\partial f(x;\theta)}{\partial x},$$

where $f(x;\theta)$ is the network's output and x is the input. Jacobian regularization introduces a penalty term in the loss function based on the Frobenius norm of the Jacobian matrix:

$$\mathcal{L}_{\text{Jacobian}} = \|J_f(x)\|_F^2,$$

In this experiment, same with Lipschitz Regularization and Smooth Jacobian regularization helps mitigate the impact of adversarial attacks by ensuring that small changes in the input do not cause significant variations in the output. By minimizing the Frobenius norm of the Jacobian matrix, the model learns to build more stable decision boundaries.

Compared with Lipschitz Regularization,Jacobian Regularization focuses more on the global smoothness of the network.

4. Carlini & Wagner Attack

Similar to FGSM and PGD, the core idea of the C&W attack is to reformulate the adversarial example generation problem as an optimization problem. By employing gradient-based optimization methods, the C&W attack minimizes

the perturbation while ensuring the classifier is successfully fooled under the specified constraints:(1). The core objective function of the attack can be expressed as

$$\min \|x' - x\|_2^2 + c \cdot f(x', y),$$

where x is the original input, x' is the generated adversarial example. The loss function $f(x', y)$ is defined as:

$$f(x', y) = \max(\max\{Z(x')_i : i \neq y\} - Z(x')_y + \kappa, 0),$$

where:$Z(x')$ represents the logit output of the model for the adversarial example x', y is the target label, κ is the confidence parameter

Compared to PGD, the C&W attack is more flexible and capable of producing higher-quality adversarial examples.

4.1. Analysis of result Accuracy

The **PGD model** uses a defense strategy combining **PGD attack + Adversarial Training** (PGD attack samples) with **Randomized Smoothing** and **Spectral Normalization**.

The **C&W model** applies **C&W Attack +** the same defense mechanism but with a mix of **50% C&W attack samples** and **50% PGD attack samples**.

Table 1. Comparison of Accuracy for Different Models

Model	PGD L_2 Accuracy (%)	PGD L_∞ Accuracy (%)
PGD Model	24.79	36.14
C&W Model	2.73	27.49

The observed PGD L_2 Attack Accuracy is (2.73%) and PGD L_∞ Attack Accuracy (27.49%), both are low especially for PGD L_2 . The observed C&W model's PGD L_2 Attack Accuracy is significantly lower compared to PGD model's.

We attribute this discrepancy to the strength of the C&W attack relative to our defense mechanisms. The current adversarial training approach is insufficient to counteract the sophisticated optimization-based nature of the C&W attack, emphasizing the need for more robust defense strategies to match such powerful adversarial methods.

4.2. Conclusion

My initial idea was to combine two methods: one designed for **L_∞-norm perturbations** and the other for **L_2-norm perturbations**.

The effectiveness of adversarial training largely depends on the sufficient exploration of gradients and adversarial space during the training process. Simply increasing the complexity or diversity of attack types does not necessarily lead to improved robustness against a specific attack. Different algorithms may not be complementary and could even conflict with each other.

This remains one of the challenges in adversarial training research: **increasing the complexity or diversity of attack types does not necessarily improve robustness against a specific attack.** The most critical factor is to find a suitable adversarial training strategy.

5. Adversarial Attacks with Momentum

Momentum-based adversarial attacks incorporate a momentum term into iterative gradient-based methods to stabilize update directions. These attacks enhance adversarial example generation by accumulating gradient information across iterations, allowing the optimizer to escape poor local minima.

5.1. MI-FGSM - Description

The first algorithm we implemented is the Momentum Iterative Fast Gradient Sign Method (MI-FGSM), introduced by Dong et al. (2). This method is a variant of the Iterative Fast Gradient Sign Method (I-FGSM or PGD-L_∞) and is detailed in Algorithm 1. In this algorithm, $\mathbf{g_t}$ accumulates the gradients, normalized by their L_1-norm, over the first t iterations, using a decay factor μ. At each iteration, the adversarial example is perturbed in direction of the sign of the accumulated gradient, with a step size α chosen such that the adversarial example obtained after the last iteration remains in the L_∞-ball of radius ϵ.

Algorithm 1 MI-FGSM as described in

Require: A classifier f with loss function J; a real example x and ground-truth label y.
Require: The size of perturbation ϵ; iterations T and decay factor μ.
Ensure: An adversarial example x^* with $\|x^* - x\|_\infty \leq \epsilon$.
1: $\alpha = \epsilon/T$ $g_0 = 0$; $x_0^* = x$
2: **for** $t = 0$ to $T - 1$ **do**
3: Update g_{t+1} by accumulating the velocity vector in the gradient direction as:

$$g_{t+1} = \mu \cdot g_t + \frac{\nabla_x J(x_t^*, y)}{\|\nabla_x J(x_t^*, y)\|_1}$$

4: Update x_{t+1}^* by applying the sign gradient as:

$$x_{t+1}^* = x_t^* + \alpha \cdot \text{sign}(g_{t+1})$$

5: **end for**
Return: $x^* = x_T^*$

5.2. MI-FGSM - Implementation

To visualize the influence of the accumulated gradient, we plot the accuracy obtained after feeding adversarial examples obtained by the MI-FGSM method to the model trained

using PGD-L_∞ and PGD-L_2 for different values of μ. For $\mu = 0$, the method is equivalent to PGD. We can see on Figure 1 that the attack is more efficient with a higher decay factor, so we'll use a value of 1 as described in the original paper.

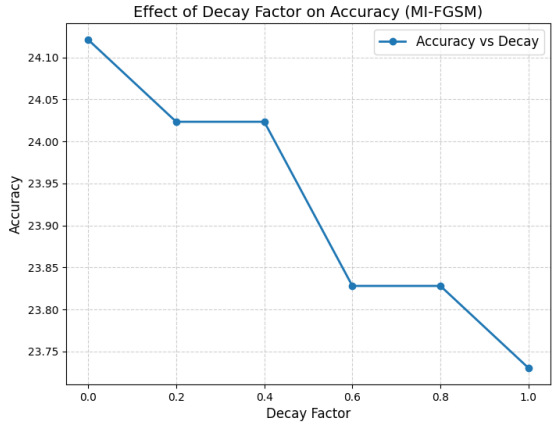

Figure 1. Effect of D Factor on Accuracy for MI-FGSM.

5.3. M-DI^2-FGSM

We also experimented with another variant of MI-FGSM called the Diverse Inputs Iterative Fast Gradient Sign Method (M-DI^2-FGSM) introduced in (5). The main idea behind this method is to apply image transformations to the inputs with a probability p at each iteration. The transformations involve random resizing, which adjusts the input images to a random size, and random padding, which adds zeros around the resized image. When $p = 0$, DI^2-FGSM is equivalent to I-FGSM, and when $p = 1$, only transformed images are used for generating adversarial examples.

This method is particularly suitable for black-box attacks, as it enhances the transferability of adversarial examples to other models. However, in our experiments, we observed that adversarial examples generated using M-DI^2-FGSM have lower success rates in white-box scenarios after the transformations as shown in Figure 2. This reduction is because the adversarial perturbations become less aligned with the exact gradient of the model under attack, thereby diminishing their effectiveness in white-box attacks.

6. Randomized Adversarial Training

In order to increase the diversity of adversarial samples and improve the model's adaptability to different types of perturbations, we can use the method of random noise injection adversarial training. During the training process, RAT introduces random noise to increase the randomness of the model. This noise can be Gaussian noise that follows a normal distribution or randomly sampled noise. Using adversarial samples $x + \sigma$ to optimize the model parameters

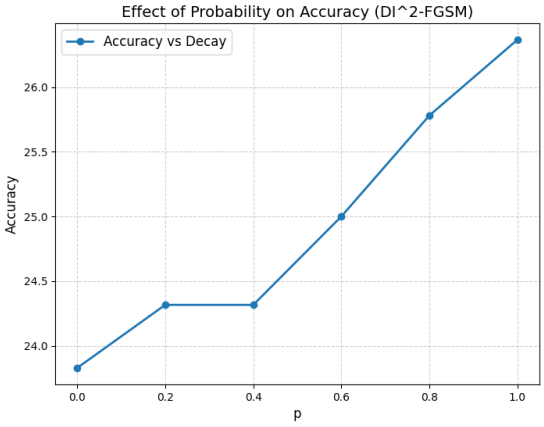

Figure 2. Effect of p on Accuracy for DI^2-FGSM.

θ can increase the robustness of the model under various perturbations. The results shown in table 2 prove that RAT outperforms both l_2 and l_∞ attacks.

	AT	RAT
Natural Accuracy	31.93	38.87
ℓ_2	31.34	38.18
ℓ_∞	23.82	24.7

Table 2. Comparison of AT and RAT on natural accuracy, ℓ_2, and ℓ_∞ performance.

7. MAT-Mixed Adversarial Training

In adversarial training, the model is usually trained on a specific type of adversarial examples, such as only l_∞ adversarial examples or only $l_2$adversarial examples. However, when facing different attack norms, the robustness of the model decreases significantly. Therefore, MAT proposes to use multiple adversarial examples for training at the same time to address the limitations of single adversarial training. The primary objective of adversarial training is to minimize the model's loss on adversarial examples (4):

$$\min_\theta \mathbb{E}_{(x,y)\sim D}\left[\max_{\|\delta\|\leq\epsilon} \ell(f_\theta(x+\delta), y)\right]$$

MAT combines adversarial examples generated under two norms (ℓ_∞ and ℓ_2).MAT-Rand randomly select a norm to generate adversarial examples:

$$\min_\theta \mathbb{E}_{(x,y)\sim D}\left[\mathbb{E}_{p\sim\mathcal{U}\{2,\infty\}} \max_{\|\delta\|_p\leq\epsilon} \ell(f_\theta(x+\delta), y)\right]$$

MAT-Max generate adversarial examples for both norms and optimize the worst-case loss:

$$\min_\theta \mathbb{E}_{(x,y)\sim D}\left[\max_{p\in\{2,\infty\}} \max_{\|\delta\|_p\leq\epsilon} \ell(f_\theta(x+\delta), y)\right]$$

7.1. Analysis of MAT

	PDG	MAT
Natural Accuracy	38.87	40.72
ℓ_2	38.18	39.84
ℓ_∞	24.7	27.14

Table 3. Comparison of PDG and MAT on natural accuracy, ℓ_2, and ℓ_∞ performance.

Our results based on MAT are shown in the table 3. PGD represents the original adversarial training model based on PGD. MAT uses two attack methods, l_2 and l_∞. MAT outperforms PDG under both l_2 and l_∞ attacks, showing that it improves robustness without significantly sacrificing basic performance. This shows that MAT is an excellent method.

References

[1] Francesco Croce and Matthias Hein. Towards evaluating the robustness of neural networks. *arXiv preprint arXiv:2012.02632*, 2020.

[2] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. *arXiv preprint arXiv:1710.06081*, 2018.

[3] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[4] Tianyu Pang, Chengxuan Du, Yinpeng Dong, and Jun Zhu. Advocating for multiple defense strategies against adversarial examples. *arXiv preprint arXiv:1904.10258*, 2019.

[5] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan Yuille. Improving transferability of adversarial examples with input diversity. *arXiv preprint arXiv:1803.06978*, 2019.