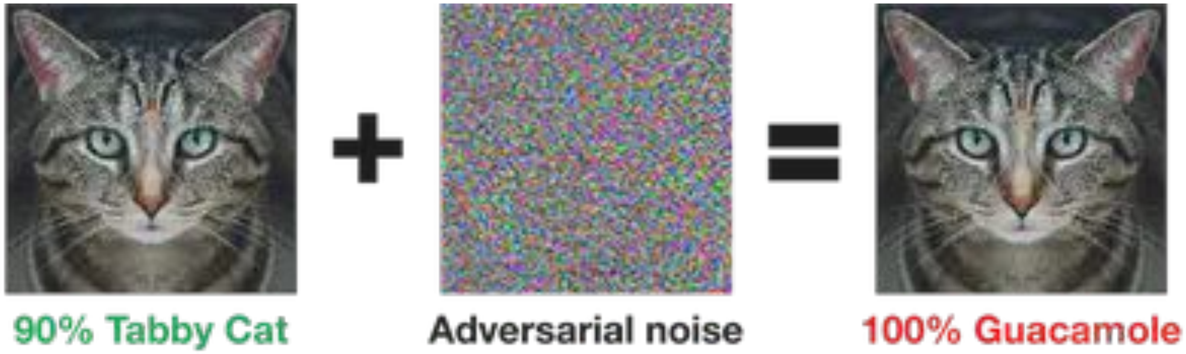
Adversarial Attacks

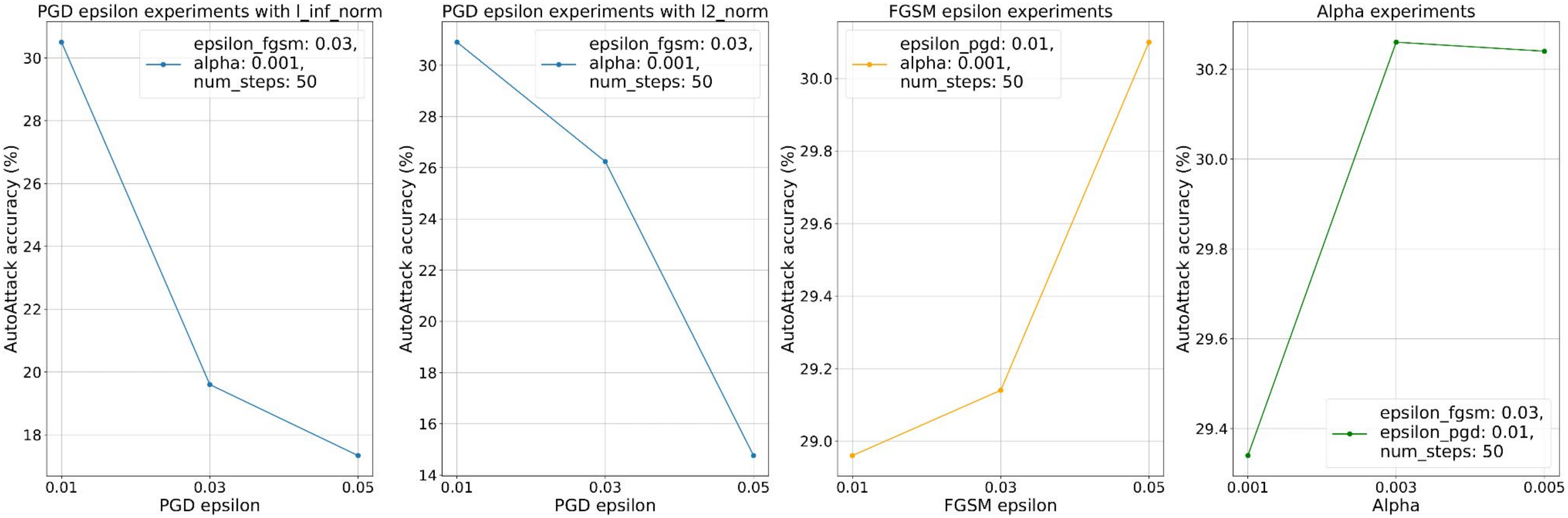
Kanupriya Jain
Anna Krysta
Mohamed Ali Srir

Introduction

- Manipulations to input data that trick machine learning models into making incorrect predictions or classifications.



AutoAttack accuracy for adversarial training



- **The best configuration: eps_PGD = 0.01, eps_FGSM = 0.05, alpha = 0.003.**
- **AutoAttack accuracy after 30 epochs: 41.02%.**

MixedNUTS: Training-Free Accuracy-Robustness Balance via Nonlinearly Mixed Classifiers

Motivation:

- Training-free approach
- Heterogeneous mixing

Core Idea:

- Benign confidence property

Notations:

g_{std} = Standard base classifier trained on clean model

h_{rob} = Robust classifier

$$f_{\text{mix,i}}(x) = \log((1-\alpha)\sigma \circ g_{\text{std,i}}(x) + \alpha\ \sigma \circ h_{\text{rob}}(x)) \qquad \alpha \in [1/2, 1]$$

Workflow

$$\max_{M\in\mathcal{M},\alpha\in[1/2,1]} \mathbb{P}_{(X,Y)\sim\mathcal{D}}\left[\arg\max_i f^M_{\mathrm{mix},i}(X)=Y\right]$$

subject to

$$\mathbb{P}_{(X,Y)\sim\mathcal{D}}\left[\arg\max_i f^M_{\mathrm{mix},i}(X+\delta^*_{f^*_{\mathrm{mix}}}(X))=Y\right]\geq r_{f^M_{\mathrm{mix}}},$$

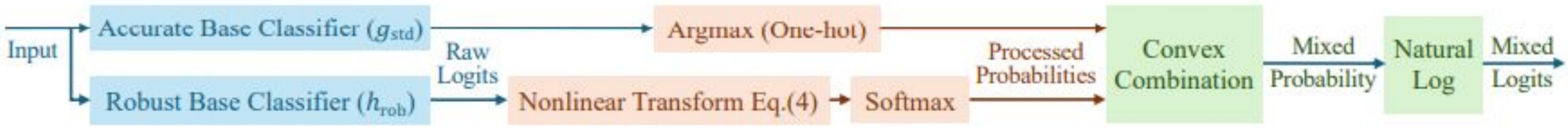


Table of cases

Image	Std Model	Robust Model	What we expect from mixing
Clean	Yes	Yes	Mix correctly classify
Clean	Yes	No	Mix correctly classify
~~Clean~~	~~No~~	~~Yes~~	~~Assume this impossible~~
Clean	No	No	We don't do magic
Adversarial	Yes	Yes	Mix correctly classify
~~Adversarial~~	~~Yes~~	~~No~~	~~Assume this impossible~~
Adversarial	No	Yes	Mix correctly classify
Adversarial	No	No	We don't do magic

We don’t lose acc on clean

While staying robust

Explicit the mix

$$f_{\text{mix}}^{M(s,p,c)}(x) := \log\left((1-\alpha)\cdot g_{\text{std}}^{\text{TS}(0)}(x) + \alpha \cdot h_{\text{rob}}^{M(s,p,c)}(x)\right)$$

Std model with confidence brought to 1
(Always 100% sure even if wrong)

Non linear transformation of the logits

Assumption 4.1. On unattacked clean data, if $h_{\text{rob}}^{M}(\cdot)$ makes a correct prediction, then $g_{\text{std}}(\cdot)$ is also correct.

Assumption 4.2. The transformation $M(\cdot)$ does not change the predicted class due to, *e.g.*, monotonicity. Namely, it holds that $\arg\max_i M(h_{\text{rob}}(x))_i = \arg\max_i h_{\text{rob},i}(x)$ for all x.

Case 2 : They std Model make correct pred on clean example and robust get it wrong

$$f_{\text{mix}}^{M(s,p,c)}(x) := \log\left(\boxed{(1-\alpha)\cdot g_{\text{std}}^{\text{TS}(0)}(x)} + \alpha \cdot h_{\text{rob}}^{M(s,p,c)}(x)\right)$$

We want this to win

Eq to say

$$\cdot\, h_{\text{rob}}^{M(s,p,c)}(x) \quad < \quad \frac{1-\alpha}{\alpha}$$ with High probability

Case 3 : They std Model make mistake on Adversarial example and robust get it correct

$$f_{\text{mix}}^{M(s,p,c)}(x) := \log\left((1-\alpha)\cdot g_{\text{std}}^{\text{TS}(0)}(x) + \boxed{\alpha \cdot h_{\text{rob}}^{M(s,p,c)}(x)}\right)$$

We want this to win

Eq to say

$$\cdot\, h_{\text{rob}}^{M(s,p,c)}(x) > \frac{1-\alpha}{\alpha}$$ with High probability

Conclusion we want a M that guarantees

$$\min_{M \in \mathcal{M},\ \alpha \in [1/2, 1]} \mathbb{P}_{X \sim \mathcal{X}^{\text{✗}}_{\text{clean}}} \left[m_{h^M_{\text{rob}}}(X) \geq \frac{1-\alpha}{\alpha} \right]$$

$$\text{subject to} \qquad \mathbb{P}_{Z \sim \mathcal{X}^{\checkmark}_{\text{adv}}} \left[\underline{m}^{\star}_{h^M_{\text{rob}}}(Z) \geq \frac{1-\alpha}{\alpha} \right] \geq \beta,$$

If we guarantee a robustness for a certain margin, we try to minimize the error on clean data

Grid Search to find those parameters

Algorithm 1 Algorithm for optimizing s, p, c, and α.

Given an image set, save the predicted logits associated with mispredicted clean images $\{h_{\mathrm{rob}}^{\mathrm{LN}}(x) : x \in \widetilde{\mathcal{X}}_{\mathrm{clean}}^{\text{✗}}\}$.

Run MMAA on $h_{\mathrm{rob}}^{\mathrm{LN}}(\cdot)$ and save the logits of correctly classified perturbed inputs $\{h_{\mathrm{rob}}^{\mathrm{LN}}(x) : x \in \widetilde{\mathcal{A}}_{\mathrm{adv}}^{\checkmark}\}$.

Initialize candidate values $s_1, \dots, s_l$, $p_1, \dots, p_m$, $c_1, \dots, c_n$.

for s_i for $i = 1, \dots, l$ **do**

    for p_j for $j = 1, \dots, m$ **do**

        for c_k for $k = 1, \dots, n$ **do**

            Obtain mapped logits $\{h_{\mathrm{rob}}^{M(s_i,p_j,c_k)}(x) : x \in \widetilde{\mathcal{A}}_{\mathrm{adv}}^{\checkmark}\}$.

            Calculate the margins from the mapped logits $\{m_{h_{\mathrm{rob}}^{M(s_i,p_j,c_k)}}(x) : x \in \widetilde{\mathcal{A}}_{\mathrm{adv}}^{\checkmark}\}$.

            Store the bottom $1-\beta$-quantile of the margins as $q_{1-\beta}^{ijk}$ (corresponds to $\frac{1-\alpha}{\alpha}$ in $\boxed{(7)}$).

            Record the current objective $o^{ijk} \leftarrow \mathbb{P}_{X \in \widetilde{\mathcal{X}}_{\mathrm{clean}}^{\text{✗}}}\left[m_{h_{\mathrm{rob}}^{M(s_i,p_j,c_k)}}(X) \geq q_{1-\beta}^{ijk}\right]$.

        end for

    end for

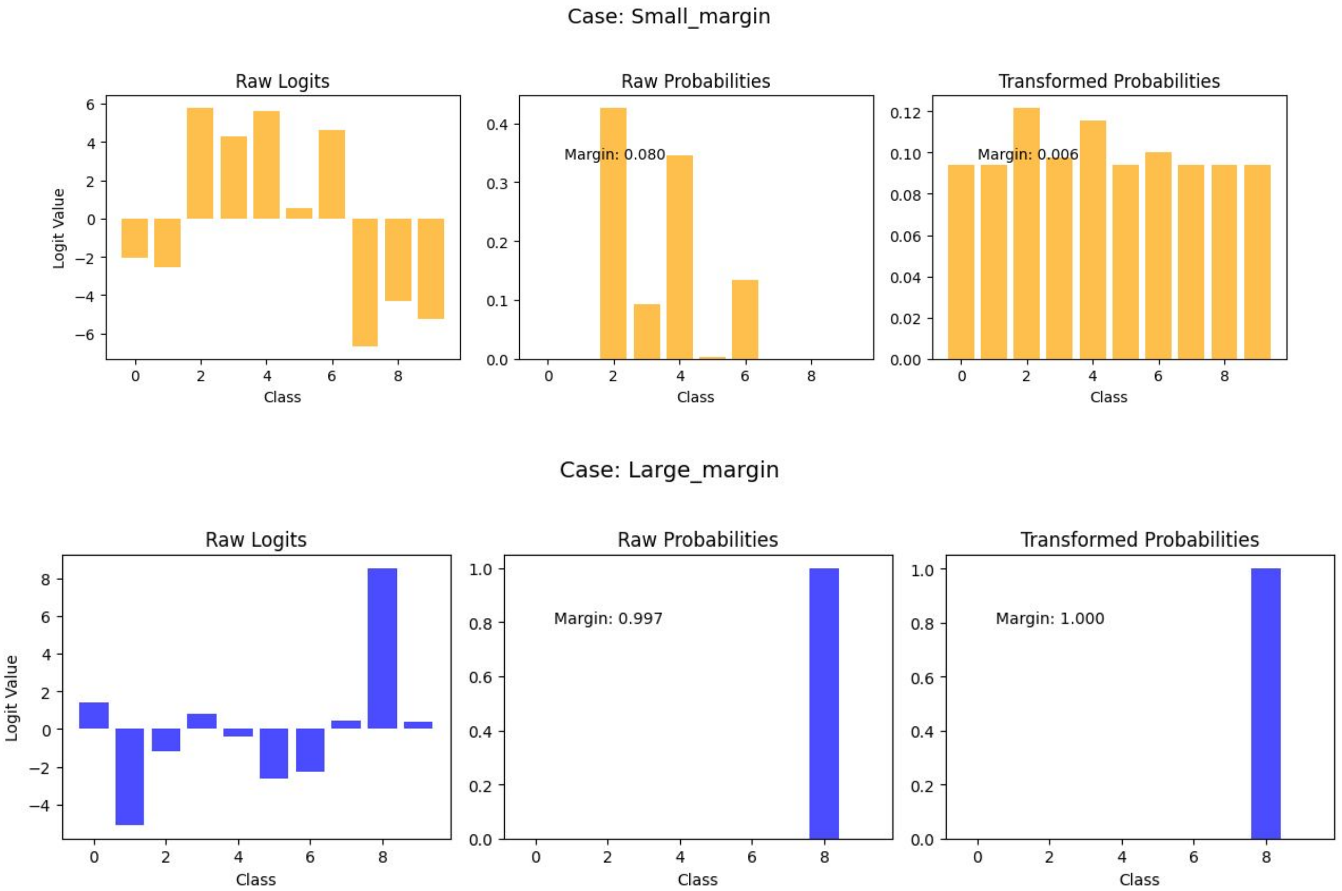
end for

Find optimal indices $(i^\star, j^\star, k^\star) = \arg\min_{i,j,k} o^{ijk}$.

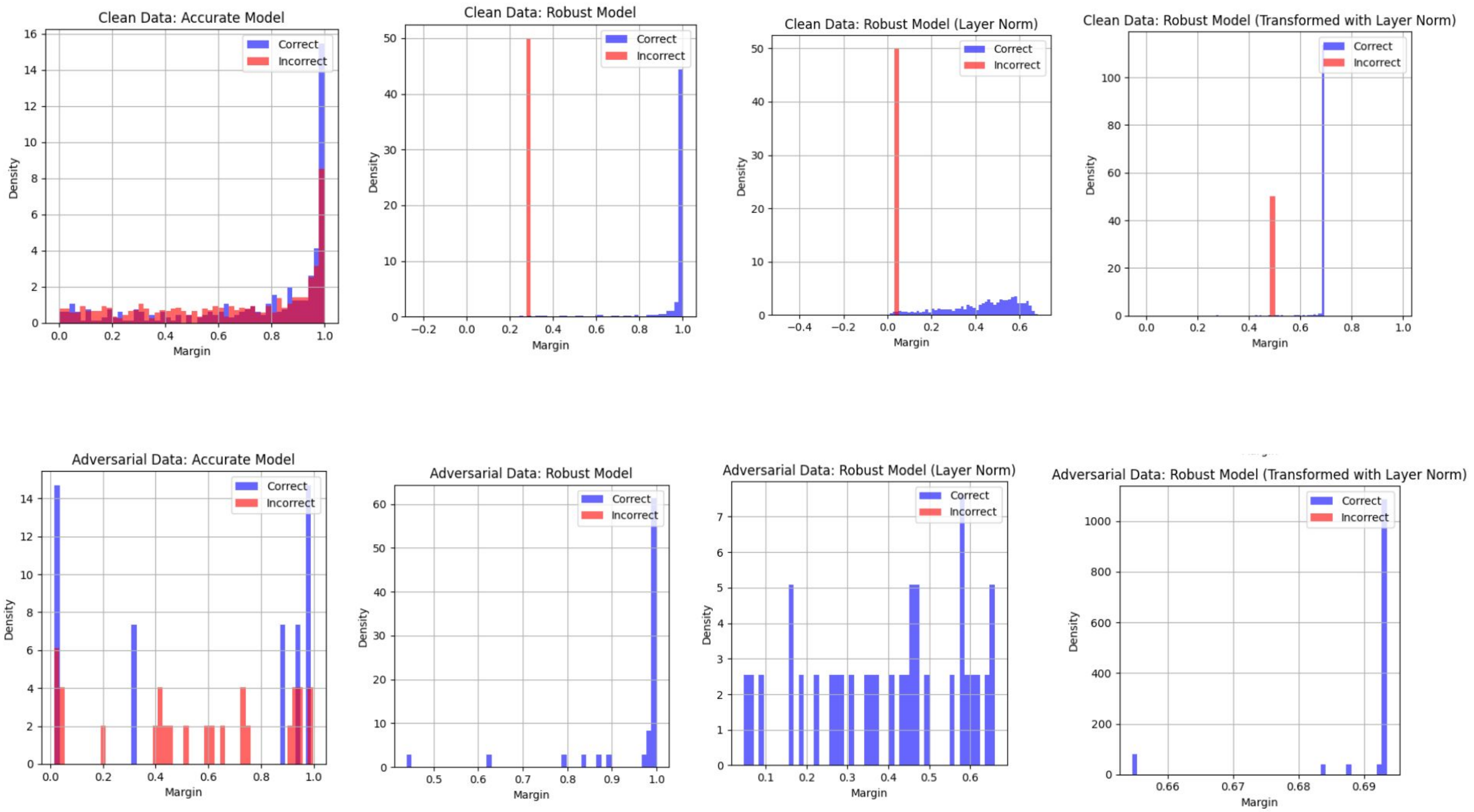
Recover optimal mixing weight $\alpha^\star := 1/\left(1+q_{1-\beta}^{i^\star j^\star k^\star}\right)$.

return $s^\star := s_{i^\star}$, $p^\star := p_{j^\star}$, $c^\star := c_{k^\star}$, $\alpha^\star$.

Effect Of Non Linear Transformation

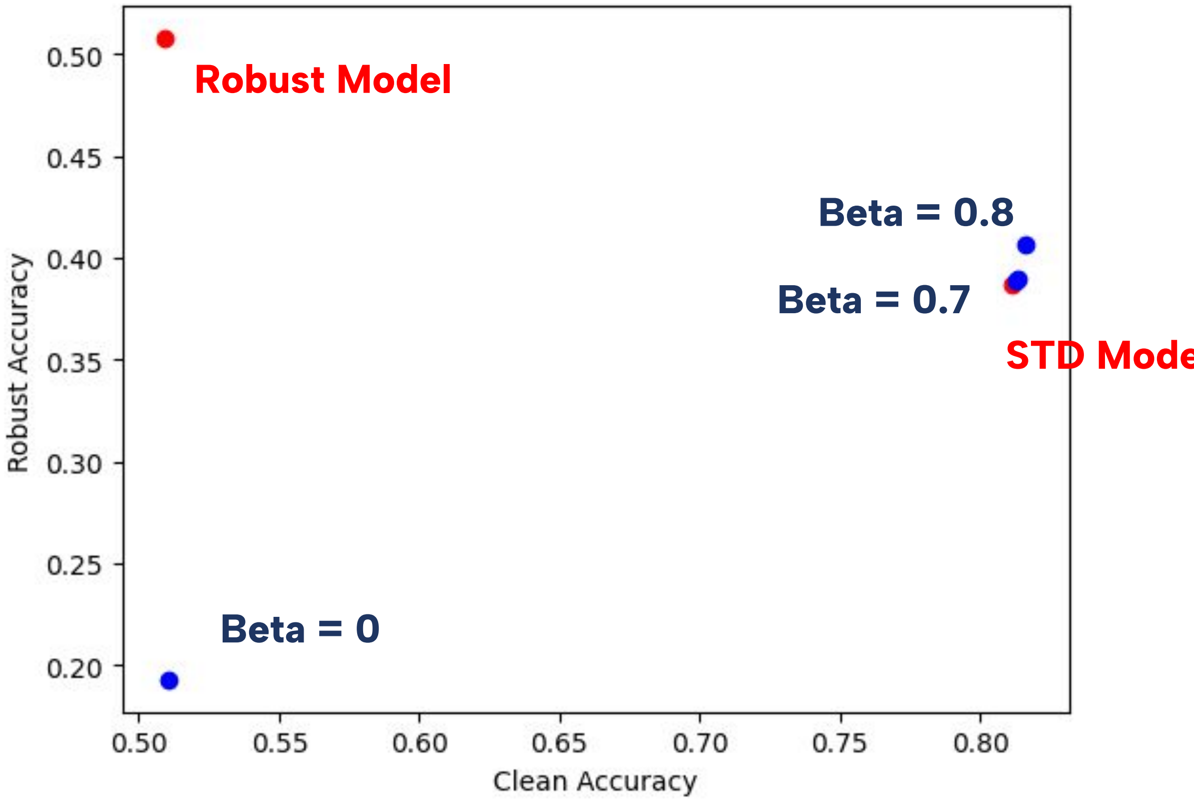


Margin Distribution



Results

	Clean Acc	Robust L2 Acc
Std	0.81	0.38
Robust	0.5	0.5
Mixed Beta = 0.8	0.81	0.41
Mixed Beta = 0.7	0.81	0.38
Mixed Beta = 0	0.2	0.5



References

[1] Yatong Bai et al. MixedNUTS: Training-Free Accuracy-Robustness Balance via Nonlinearly Mixed Classifiers. 2024. arXiv: 2402.02263 [cs.LG]. url: https://arxiv.org/abs/2402.02263.

[2] Pytorch CIFAR models.url:https://github.com/chenyaofo/pytorch-cifar-models/tree/master (Date de consultation :10/12/2024)

[3]Pytorch-Adversarial-Training-CIFAR.url:https://github.com/ndb796/Pytorch-Adversarial-Training-CIFAR/tree/master.(Date de consultation : 10/12/2024)