Data Science Lab

Master : Artificial Intelligence, Systems, Data

Training robust neural network

Prepared by:

Lemaître Victor
Charaf Zadah Azammat
Carron Louis

Supervised by:

Vérine Alexandre
Bourdrez Constant

Université Paris Dauphine – PSL

Academic Year 2025–2026

Contents

1 Introduction

Our first step was to implement FGSM and PGD attacks. We stuck to PGD attacks with an array of hyperparameters to evaluate the robustness of our models. Next we present the performance of the original default model on these battery of tests:

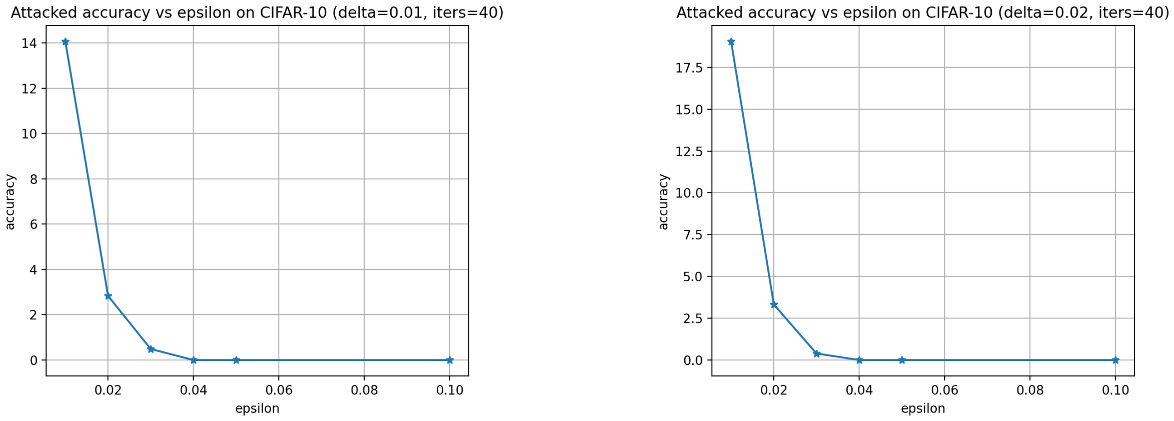

Figure 1: Robustness of the default model.We denote by ϵ the attack ball's radius and δ the learning rate. Observe the low performance compared to the natural accuracy of 63%

Following this we implemented adversarial training. At every batch we adversarially perturb a fraction of the inputs. We do not perturb the whole batch because of 1) computational cost and 2) generalization issues where the natural accuracy drops too much. Indeed the latter point will be a recurrent theme in this report. Most methods offer trade-off between clean and attacked accuracy.

These issues with simple adversarial training lead us to adopt the idea from *curriculum adversarial training* [1]. We increased the strength of the attacks (epsilon and the number of iterations) with the numbers of epoch. While we did not adopt the idea of learning stage from the paper and simply increased the attacks difficulty at each epoch we still observed an improvement over the trade-off between clean and attacked accuracy. From our tests we also did not observe the issue of *forgetting* where training on harder attacks makes the model brittle to easier attacks. Therefore we did not implement the authors idea of batch mixing.

From here on, all presented models will be trained with the following settings for the curriculum adversarial training and an attacked ratio of 0.5:

$$t = \frac{\text{number of elapsed epochs}}{\text{total number of epochs}}$$
$$\epsilon = 0.01 \times (1-t) + 0.1 \times t$$
$$\text{nb iterations} = \text{int}(10 \times (1-t) + 40 \times t)$$

2 Improving models capacities

Empirically improving the model capacity results in a Pareto improvement over the clean/attacked accuracy trade-off. [2] also shares this conclusion, indeed they explain that adversarial training requires a stronger classifier. We include their figure which provides a visual illustration of their point.

With this in mind we made two simple changes that made training our model easier. First we switched from SGD with momentum to AdamW. Next we used GELU activation functions in all our subsequent models. This last change alone usually added +5/7% to peak natural accuracy during the training of our biggest models.

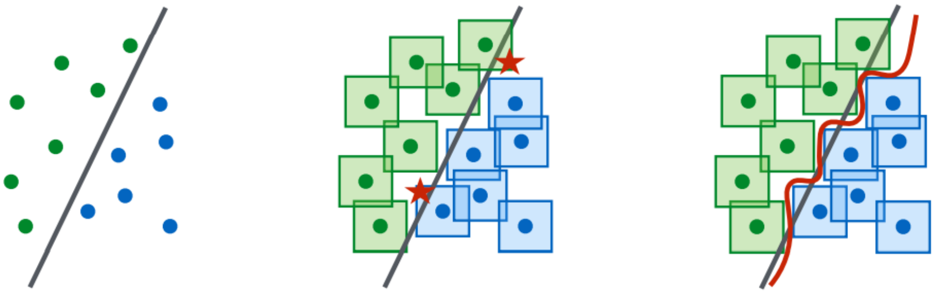

Figure 2: The classifiers boundary are made more complex due to adversarial attacks

2.1 Increasing the model size

The most obvious way to improve a model capabilities is to increase its number of parameters or to use more complex architectures. We first tried to implement a wide residual network (WRN) [3] with 13 convolution layers and a width factor of 10. Despite having good performance the prohibitive training duration led us to use a simpler model we called "BigNet" for our experiments. It adopts the same architecture as the original model but with the dimensions of every layer roughly doubled. Finally we also experimented with the visual transformer architecture (ViT) [4]. For the tokens creation we used a sequence of convolutional layers instead of linear projection of the image patches. We justified this choice by the inductive bias inherent to convolutions which is made all the more necessary by our inability to use large scale pretraining unlike the author of the ViT paper.

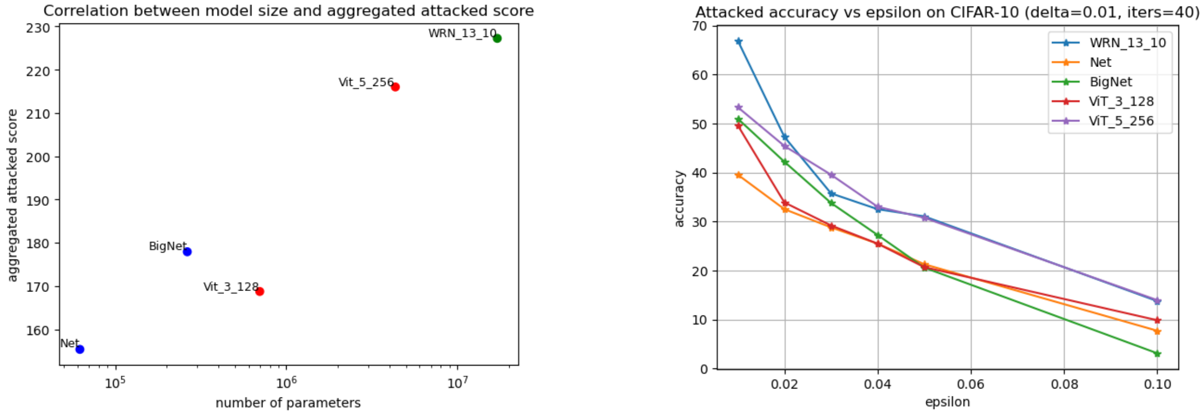

Figure 3: For ViT the first number refers to the depth, the second one is the embedding size of the model

We can observe a positive correlation between a model size and its robustness. However this increased robustness is mainly confined within the easier attack part of our experiments with some bigger models performing worse than simpler ones. We think this is caused by the greater difficulty of training larger models. We kept the training parameters the same for the sake of this experiments but larger models shown here might be undertrained.

2.2 Data augmentation

Data augmentation is a "free" way to get more diversity in our training set and therefore more robust model. The most obvious augmentation being horizontal flipping. Indeed because CIFAR10 images do not have text, an image and its horizontally flipped version have exactly the same semantic. We also tried different data augmentation such as random cropping and color jittering which modifies the brightness, contrast and saturation properties of an image to make the model more robust to lightning differences. We tested these data augmentation with the BigNet model trained for 60 epochs.

	Nothing	**Horizontal flip only**	**Full pipeline**
Aggregated attacked score	136	**169**	159

Table 1: Impact of data augmentation on robustness. Full pipeline refers to Horizontal flip + Random cropping + Color jittering

Unfortunately the more advanced data augmentations do not seem to work. We hypothesize that this may be due to a lack of model's expressivity.

Given that adding new images seemed to improve our model robustness we also tried to generate synthetic images by recycling the old GAN project. We modified the architecture to construct a much bigger Conditional GAN inspired by the *BigGan* [5] architecture. Despite trying to improve the synthetic image quality via keeping an exponential moving average of the generator weights or sampling via *Discriminator Driven Latent Sampling* [6] the final images were still not realistic enough.

3 Randomness

As our first step we added dropout to our models. In practice it made them a bit more robust while keeping natural accuracy intact. Next, we have considered defensive network architectures as a further improvement. We have mainly experimented using randomized defenses.

Due to time constraints, in order to identify promising approaches quickly enough, we have conducted smaller-scale experiments before transferring the acquired knowledge to higher-scale models. Namely, we started from our BigNet network as a baseline which we tried to improve on. Our two main attempts for that were 1) using Parametric Noise Injection (PNI) [7], and 2) using Random Self-Ensemble (RSE) [8].

- PNI consists in injecting Gaussian noise in some spots in the network, with each noise being scaled by a learnable parameter. More precisely, it is described by the following operation as described by the authors:

  $$\tilde{v}_{l,i} = v_{l,i} + \alpha_l \cdot \eta_{l,i}; \quad \eta_{l,i} \sim \mathcal{N}(0, \sigma^2)$$

  where $\boldsymbol{v}_l$ can be input/weight/activation tensor at the l-th layer, and α_l is the learnable scaling factor. We ended up adding such noise to the weights $\boldsymbol{w}_l$ of each layer (linear and convolutional) as it is the best approach presented in the paper. In that case, we take for σ the standard deviation of $\boldsymbol{w}_l$. We then use $\tilde{\boldsymbol{w}}_l$ for forward-passes.

- RSE consists in adding noise layers in the network. A noise layer simply adds a Gaussian noise to its input. More specifically, we add a noise layer before each convolution. The first noise layer uses some σ_{init} standard deviation while the subsequent ones all use some σ_{inner} standard deviation. Note that these parameters are not learned.

After 10 epochs of PGD-based adverserial training (epsilon of 0.03, delta of 0.008; not too weak nor too strong), these were the results (natural and attacked samples have the same weight during training):

Modèle	BigNet	BigNet-PNI	BigNet-RSE
Natural accuracy	53.32%	51.66%	49.41%
Attacked accuracy	31.15%	31.07%	**32.50%**

Table 2: Results of adverserial training for BigNet and randomized defense variants.

We did not end up getting good results with PNI in that small-scale experiment. RSE however seems to effectively allow for improvements in terms of performance against attacks more easily (and seemed promising since epoch 1 compared to the rest), although there is a drop in natural accuracy. We ended up submitting a WRN-RSE model (with adversarial training) online. Here is an illustration of the improvement upon WRN and the base Net. Natural accuracy is roughly the same but the network is better at defending itself (up to a certain point):

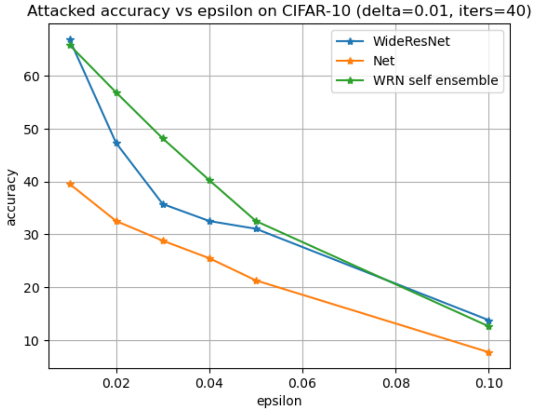

Figure 4: Base Net, WRN, WRN-RSE against PGD attacks (no data augmentation).

4 Towards Black Box Attacks

In this setting, the exact gradient of models are no longer available. Therefore, we explored gradient-estimation techniques in order to assess how well our models could deal with these more realistic threat.

We implemented the NES Gradient Estimate algorithm from the paper [9] in the Query-Limited setting. The attacker can only interact with the model by sending a limited number of queries L. To craft adversarial examples it uses the standard PGD algorithm where the number of iterations is given by the ratio $\frac{L}{N}$, with N the total number of samples to estimate each gradient. It is estimated using centered finite differences along random directions, where $P(y|x)$ is the predicted class probability for label y given input x.

$$\hat{\nabla} P(y|x) = \frac{1}{2n\sigma} \sum_{i=1}^{n} \left[P(y \mid x + \sigma u_i) - P(y \mid x - \sigma u_i) \right] u_i, \quad u_i \sim \mathcal{N}(0, I_N).$$

A high query budgets lead to behavior that is closer to the exact PGD under the same parameters. Experiment 5 was conducted on BigNet. Due to time constraint, it uses only 200 images, and 100 samples per gradient estimates, resulting in high variance.

Then it is difficult to draw conclusion about the gradient estimation quality. For the same reason, we did not have time to try it in adversarial training for which the computational cost would have been important.

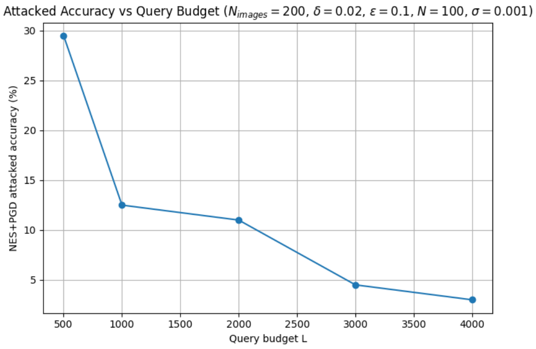

Figure 5: NES attack results on BigNet as a function of the query budget L with $\sigma = 0.001$, $N = 100$ $\epsilon = 0.1$, $\delta = 0.02$, and $N_{\text{images}} = 200$.

5 A short list of failed experiments

- Making the models 1 lipschitz via spectral normalization
- Making the model learn a PCA (by adding a learnable projection matrix of rank less than $32 \times 32 \times 3 = 3072$), the idea stemmed from the observation that the effective dimension [10] of CIFAR10 images seen as points was 625 vs 3072. Hence if we could force the attacks to be on a 625 dimensional linear subspace we could restrict the attacker liberty.
- Adding a small gaussian noise to inputs during testing. We thought it could counteract PGD attacks who found sharp local maximum. In practice it improved very slightly robustness but at a bigger cost to natural accuracy.

6 Conclusion

Across this project, we investigated a broad range of techniques to improve adversarial robustness, from curriculum adversarial training to increased model capacity, data augmentation, and randomized defenses. First, adversarial training remains effective but imposes a clear trade-off between clean and attacked accuracy, which can be partially mitigated through curriculum scheduling and stronger model architectures. Second, larger models tend to be more robust but are significantly harder to train, making optimization choices crucial. Third, classical data augmentation offers limited gains, and more aggressive transformations or synthetic data generation do not necessarily help when model expressivity is constrained. Fourth, randomized defenses like RSE can yield moderate improvements, whereas others such as PNI did not show benefits in our setting. Finally, our black-box experiments suggest that NES tends to approach PGD behavior when the query budget is high, though its computational cost makes it impractical for adversarial training. Overall, our findings emphasize that achieving meaningful robustness requires both stronger architectures and carefully designed training procedures, and that many promising approaches remain limited by computational cost.

7 References

[1] Qi-Zhi Cai et al. "Curriculum adversarial training". In: *arXiv preprint arXiv:1805.04807* (2018).

[2] Aleksander Madry et al. "Towards deep learning models resistant to adversarial attacks". In: *arXiv preprint arXiv:1706.06083* (2017).

[3] Sergey Zagoruyko and Nikos Komodakis. "Wide residual networks". In: *arXiv preprint arXiv:1605.07146* (2016).

[4] Alexey Dosovitskiy. "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929* (2020).

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. "Large scale GAN training for high fidelity natural image synthesis". In: *arXiv preprint arXiv:1809.11096* (2018).

[6] Tong Che et al. "Your GAN is secretly an energy-based model and you should use discriminator driven latent sampling". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 12275–12287.

[7] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. "Parametric Noise Injection: Trainable Randomness to Improve Deep Neural Network Robustness Against Adversarial Attack". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019, pp. 588–597.

[8] Xuanqing Liu et al. "Towards Robust Neural Networks via Random Self-ensemble". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Sept. 2018, pp. 369–385.

[9] Andrew Ilyas et al. *Black-box Adversarial Attacks with Limited Queries and Information*. 2018. arXiv: 1804.08598 [cs.CV]. URL: https://arxiv.org/abs/1804.08598.

[10] Olivier Roy and Martin Vetterli. "The effective rank: A measure of effective dimensionality". In: *2007 15th European signal processing conference*. IEEE. 2007, pp. 606–610.