Data Science Project - Adversarial Networks

Alejandro Castro Ros - Adnan Ben Mansour - Laurène Bouskila

December 2020

Introduction

Neural Networks are one of the most common artificial intelligence system. In particular, Computer Vision models have been widely developed in the last years, ranging from basic classification systems use to image generation. Besides their positive features, such as their ability to generalize or to correctly predict different data patterns, they also present a surprising vulnerability to adversarial attack. Based on adding small perturbations to the data, they aim to fool the classification system. Under this kind of attacks, model decisions are prone to change and misclassify an image that was correctly classified before, and whose aspect is very similar to the original one when looked with the human eye.

In a world increasingly depending on artificial intelligence, this kind of attacks can lead to malicious applications that threaten security critical applications (surveillance systems, identification, autonomous cars...). In this context, the need of implementing defense systems is reinforced, enabling robust and reliable networks. In this report, we will implement different kinds of attack and defense systems and compare their performance, strong points and drawbacks.

1 Description

In this section we will present the dataset that we have worked with, as well as the classification model used as a base system to which we have performed attacks and defenses.

1.1 Dataset presentation

All along our work we will use the CIFAR10 dataset which contains 60,000 32x32 (low resolution) color images belonging to 10 different classes. These 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks, as one can see in Figure 1, having 6,000 images for each of them.

The train set contains 50k examples whereas the test set contains 10K (train-test split of 80%-20%).

1.2 Model

In order to properly analyze the effects of the attack strategies over a classifier, we need to consider a trustful base model. Since the scope of our work was not to obtain the best classifier but to develop attack-defense implementations, and since we wanted to use the same model architecture as the rest of the working teams, we didn't make much exploration regarding the model structure, and used the architecture represented in Figure 9 in the Appendix.

Figure 1: Samples of the CIFAR10 dataset.

Once we defined the architecture of our model, we wanted to optimize its classification accuracy over the validation dataset. In order to do so, we added two Dropout layers and used Stochastic Gradient Descent with a learning-rate scheduler. Moreover, we applied data normalization and data augmentation techniques over the training dataset to improve the model generalization. Specifically, we used random horizontal flip and random crops to the original train images.

After all of these improvements, we obtained a base model that reaches an accuracy of 87.75% over the test dataset.

2 Attacks systems

When regarding the representation of samples belonging to different classes, sometimes they are very close in the feature space. Hence, the substantial success obtained by simple attack methods is a consequence of attacking this proximity. Even a limited imperceptible perturbation on the image can lead to misclassification of the sample.

In this section we will see different types of adversarial attacks that can be divided into two categories: single step attacks and iterative attacks.

2.1 FGSM attack

Known as *First Gradient Sign Method*, this one-step technique relies on the sign of the gradient to create the perturbation. As a matter of fact, the perturbation is added to the sample in the gradient direction, extended by a small value that we will denote by ε. This value is constrained into a certain range so that the perturbation remains imperceptible for the human eye.

In this context, we will use the infinity norm $\|\cdot\|_\infty$ for which we have the following solution for τ:

$$\max_{\|\tau\| \leq \varepsilon} \tau^T \nabla_x L_\theta(x, y) \qquad \Longrightarrow \qquad \tau = \varepsilon \operatorname{sign}\left(\nabla_x L_\theta\left(x, y\right)\right) \tag{1}$$

where L is the adversarial loss with parameters θ, x is the adversarial input, y is the ground truth label and τ is the perturbation. Therefore, we apply this perturbation to the original input and obtain a perturbed image that looks very similar to the original for the human eye but whose classification by the model is not correct:

$$x_{adv} = x + \varepsilon \operatorname{sign}\left(\nabla_x L_\theta\left(x, y\right)\right) \tag{2}$$

A real, not cherry-picked example of this perturbation can be seen in Figure 2.

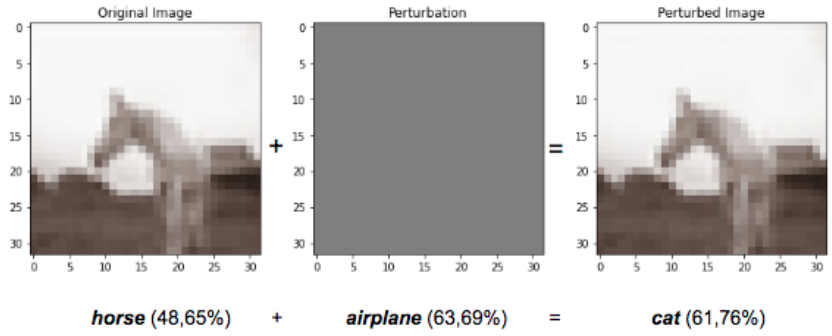

Figure 2: Example of a FGSM perturbation.

In this illustrative example, our model correctly classified the original image as a horse (with 48.65% of confidence) but, after applying a perturbation based on FGSM method, if classifies the perturbed image as a cat (with 61.76% of confidence). It's important to note that, even if the image resolution is not very high, the perturbed image is very similar to the original one for the human eye, whereas it's completely different for the classifier.

2.2 PGD attack

Projected Gradient Descent is an iterative version of the FGSM implementation. PGD is a multi-step variant technique which computes the new perturbation at each iteration using the following algorithm:

$$\begin{aligned} x_0 &= x \\ x_{t+1} &= \prod_{\mathbf{B}(0,\varepsilon)} \left(x_t + \eta \operatorname{sign}\left(\nabla_x L_\theta\left(x, y\right)\right)\right) \end{aligned} \tag{3}$$

Once again, in Figure 3 we will take a look to the same image as in Figure 2 but in this case after applying a PGD perturbation.

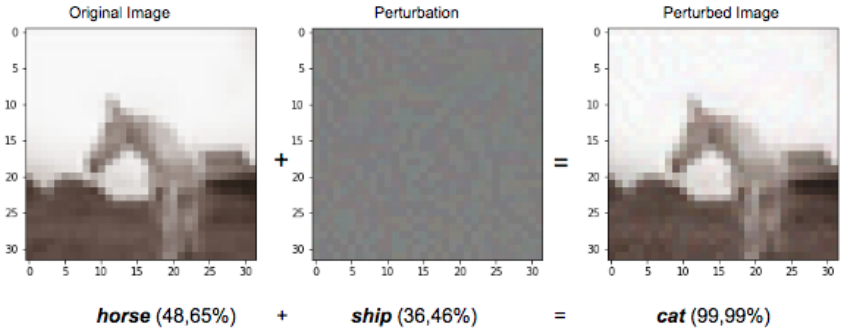

Figure 3: Example of a PGD perturbation.

As one can see, after applying a perturbation based on PGD method, we obtain once again an image with imperceptible differences with respect to the original one but which our model classifies as a cat (with 99.99% of confidence). In this case, it's important to remark the high confidence that our model has when classifying the perturbed image.

2.3 Performance comparison

As we have seen in Figures 2 and 3, both FGSM and PGD attacks are quite effective when applied to our base model. We will go a bit more into depth with these two attacks. It is important to note that the validation accuracies values are calculated over the images that were correctly classified by our base model (8805 images on the 10K test set), since the analysis of the errors of this base model is out of the scope of this work.

In Figure 4, we can observe the influence of ε on the accuracy of FGSM attack over the validation set. It's important to note that quite small values of ε influence the model have very low accuracy. For example, with a perturbation in the gradient direction with $\varepsilon = 0.05$, the accuracy already falls to 22.0%. The lowest accuracy (13.77%) is obtained for $\varepsilon = 0.5$ (with 1212 well classified images). However, for such a consequent perturbation, the image can no longer be considered similar to the original one, so this epsilon value cannot be considered as a good parameter for the attack.

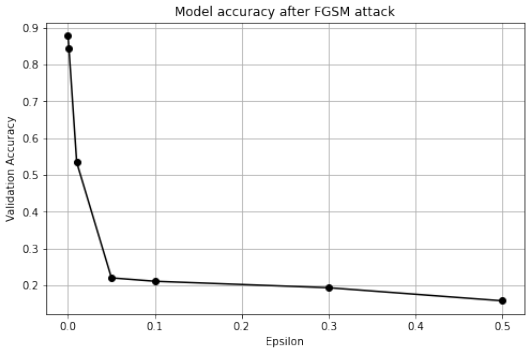

Figure 4: Validation accuracy as a function of epsilon for the FGSM attack

In the case of the PGD attack, we can observe in Figure 5 that very low accuracy on the validation set is reached faster (for lower ε) than in the case of the FGSM attack: we obtain an accuracy of only 7.6% for $\varepsilon = 0.05$ (678 images correctly classified). This means that PGD attack is, even though simple, a very efficient bounded attack. This is due to the fact that PGD is an iterative process; that makes it more effective than FGSM (as well as most computationally expensive).

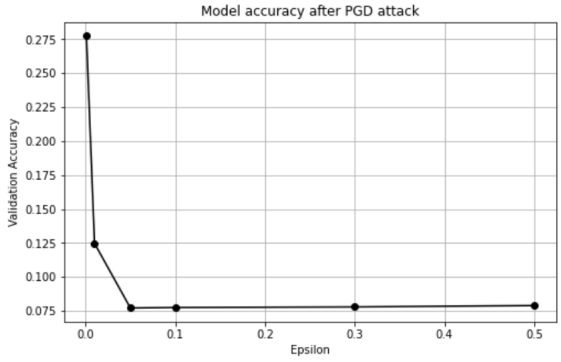

Figure 5: Validation accuracy as a function of epsilon for the PGD attack

In Table 1, a detailed report of the accuracies obtained after each attack for different values of ε can

epsilon	0.001	0.01	0.05	0.1	0.3	0.5
validation accuracy under FGSM [%]	84.61	54.05	22	21.74	20.57	13.77
validation accuracy under PGD [%]	27.75	12.47	7.71	7.75	7.79	7.89

Table 1: Comparison on validation accuracy of the two attacks for different parameters.

be observed. Once again, it's important to take into account that bigger values of ε imply bigger perturbation and, therefore, the perturbed image may not be recognised as the original one (values bigger than $\varepsilon = 0.1$ already produce highly perturbed images).

3 Defense systems

In order to protect deep neural networks against adversarial attacks, there exist several techniques which enable the systems to be secure and trust worthy.

3.1 Adversarial Training

Adversarial training is an inherent defense system that aims at improving neural network robustness against adversarial attacks by training it with adversarial examples.

It can be modeled by the following equation:

$$\min_{\theta} \mathbb{E}_{(x,y)} \left(\max_{\|\tau\| \leq \varepsilon} L_\theta \left(x + \tau, y \right) \right) \tag{4}$$

where L is the adversarial loss with parameters θ, x is the adversarial input, y is the ground truth label and τ is the perturbation.

Adversarial attacks such as FGSM and PGD find the most effective adversarial examples and solve the inner maximization problem in Equation (4). The outer minimization is the standard training procedure to minimize the loss.

The results displayed in Table 2 illustrate the accuracy on the validation set for both FGSM and PGD attacks under adversarial training defense.

Attack	Defense	$\varepsilon = 0.01$	$\varepsilon = 0.05$
FGSM	Without Defense	47.95%	7.31%
	Adversarial Training (vs FGSM)	68.53%	**43.15%**
	Adversarial Training (vs PGD)	**68.74%**	16.92%
PGD	Without Defense	57.99%	0%
	Adversarial Training (vs FGSM)	88.32%	3.55%
	Adversarial Training (vs PGD)	**91.28%**	**13.33%**

Table 2: Accuracy comparison for FGSM and PGD attacks versus different defense strategies.

We can observe that we get much higher accuracy when defending over the FGSM attack. This is due to the fact that PGD method is an iterative technique and is way more powerful than FGSM attack. This explains why we don't get such a good defense with standard adversarial training technique. Let's figure out if we get better results with another defense technique.

3.2 Defense Based on Gaussian Noise

As we have seen in previous sections, small perturbations properly designed are enough to induce misclassification to our model. Hence, in addition to Dropout, we added Gaussian noise, as proposed in [6], between the first layers of the model in order to make it more resilient to small perturbations. As recommended in the references, we used a centered Gaussian distribution with standard deviation $\sigma = 0.05$. However, the results obtained after FGSM attacks were not as good as expected (Figure 6), since the overall accuracy is slightly lower for the model with Gaussian noise (we obtained similar results for different parameters of the Gaussian distribution).

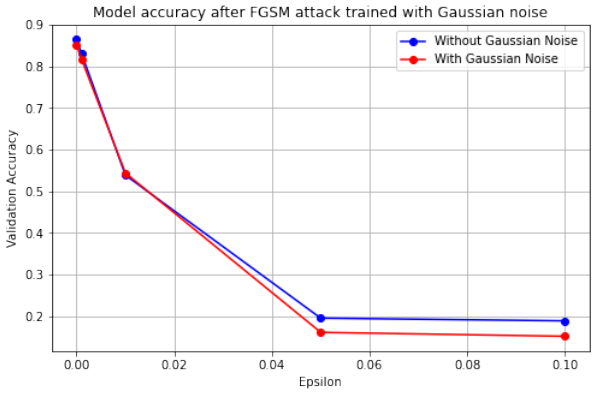

Figure 6: Performance under FGSM attack for our base model (**blue**) and a model with Gaussian noise in the first Convolutional layer(**red**).

One explication of this result might be that, unlike the base model used by [6], our model consists on several convolutional layers, not only linear ones, so the complexity of the model makes it be unaffected by this kind of perturbation. In order to analyze a bit more into depth this idea, we considered a model with a Gaussian noise layer between the two dense layers placed in the last part of the base model, instead of placing the noise layer at the beginning (recall that a schema of our based model can be seen at the Appendix). However, the results were very similar as with the first approach, as it can be seen in Figure 7.

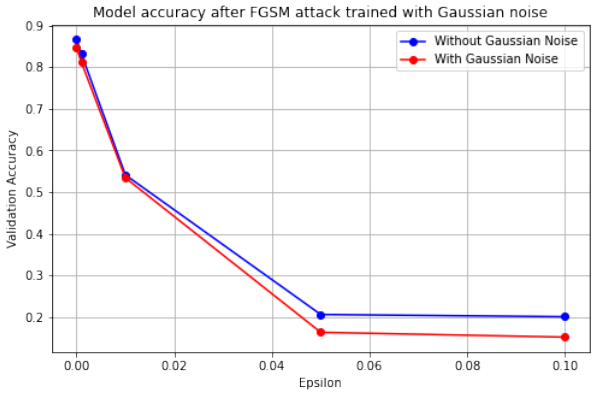

Figure 7: Performance under FGSM attack for our base model (**blue**) and a model with Gaussian noise in the first Convolutional layer (**red**).

3.3 Defense Based on Latent Representation

Another idea that we have thought of was regarding the distance of the different classes in the feature space and creating clusters, so that a small enough perturbation won't lead to error in the sample classification. In order to do so, we have used the contrastive loss.

When we have two examples (x_1, z_1) and (x_2, z_2), we compute the representations y_1 and y_2 in the latent space of these examples, and we add to the cross entropy losses of z_1 and z_2 a term called contrastive loss which can be written as :

$$CL(1,2) = 1_{z_1=z_2} \cdot d(y_1, y_2)^2 + 1_{z_1 \neq z_2} \cdot \max(0, \alpha - d(y_1, y_2))^2 \tag{5}$$

All along our work, we have considered $\alpha = 2$ as an interesting choice of the parameter.

Then, we computed the label of an image by comparing the distances to 5×10 given images in the distance space $(1 - nn)$. We obtained results really close to the results we got without adversarial training at all (80% accuracy before attack and similar accuracy after our attacks). Hence, we tried to understand why our intuition was wrong.

We made a graphical representation of the data in the latent space with and without applying the contrastive loss defined in equation (5), as it can be seen in Figure 8.

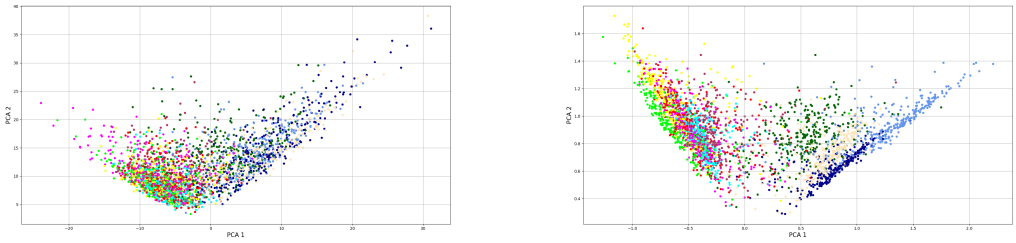

Figure 8: Latent space without contrastive loss (**left**), and with contrastive loss (**right**)

Then, we tried to analyze the position of the samples after an adversarial attack. Unfortunately, it seems that, even with a choice of a small perturbation ($\varepsilon = 0.01$).For instance: an example in the middle of the red cluster might end up in the middle of the blue cluster after an FGSM attack. Moreover, the distance between an image and its $1 - NN$ is, on average, greater than the distance between the attacked image and its $1 - NN$.

In conclusion, we think that this method needs to be pushed a bit further in order to preserve distances in the latent space when dealing with small perturbations. For example, we could use a triplet loss, and instead of using a random positive example, we could use an attacked version of the image like in the adversarial training technique.

4 Conclusions

All along this document, we have explained and analyzed different kind of adversarial attacks on neural network models, as well as multiple defense strategies. By observing the results, we can conclude that gradient based attacks are a very simple approach that yield to quite good results, but whose effects can be mitigated by adversarial training.
Moreover, we have seen that the addition of Gaussian noise to our base model (based on Convolutional layers), did not provide a good enough defense method against gradient based attacks (unlike for simpler models based only on dense layers).
Finally, we have proposed another defense approach based on clustering using latent space distances. Even if the results were not very satisfying, the approach is still promising, since further work can be achieved with this method that could yield to better results in a near future.

References

[1] I. Goodfellow, J. Shlens and C. Szegedy. *Explaining and harnessing adversarial examples*. ICLR, 2015.

[2] A. Madry, A. Makelov, L. Schmidt, D. Tsipras and A. Vladu. *Towards Deep Learning Models Resistant to Adversarial Attacks*. ICLR, 2018.

[3] N. Carlini, D. Wagner. *Towards Evaluating the Robustness of Neural Networks*, 2016.

[4] A. Araujo, R. Pinot, B. Negrevergne, L. Meunier, Y. Chevaleyre, F. Yger, and J. Atif. *Robust Neural Networks using Randomized Adversarial Training*. arXiv preprint arXiv:1903.10219, 2019.

[5] A. Araujo, B. Negrevergne, Y. Chevaleyre and J. Atif. *On Lipschitz Regularization of Convolutional Layers using Toeplitz Matrix Theory*. arXiv preprint arXiv:2006.08391v2, 2019.

[6] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana*Certified Robustness to Adversarial Examples with Differential Privacy*. IEEE Symposium on Security and Privacy, 2019.

Appendix

Model Structure

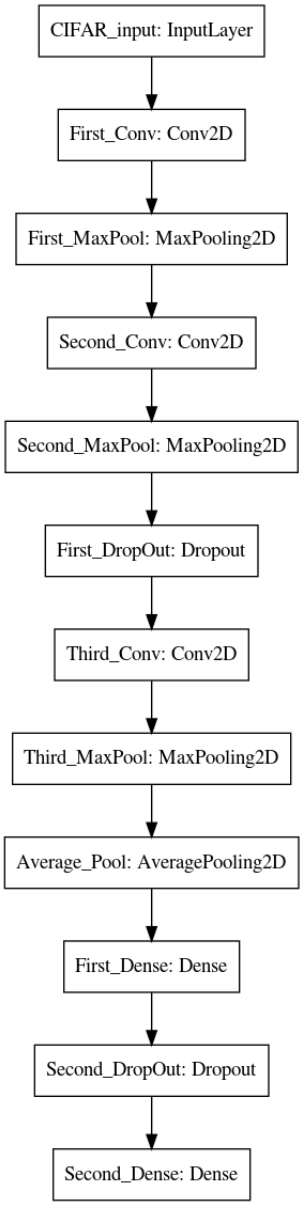

Figure 9: Base Model Scheme.