Training robust neural networks

BAtMAN : **B**rahim **At**tacks with **M**ikail and **A**hmed **N**eural Networks

14 décembre 2021

Basic Framework

We chose to use the Mobile-net[1] architecture for this project for multiple reasons :

- A light network to train and to test with
- Good benchmark performances compared to the State of the Art
- Widely used especially in low specs settings

For a randomly initialized network, 200 Epochs, and a learning rate of 0.001, we achieve an accuracy of 77.73%

Accuracy after Basic Attacks

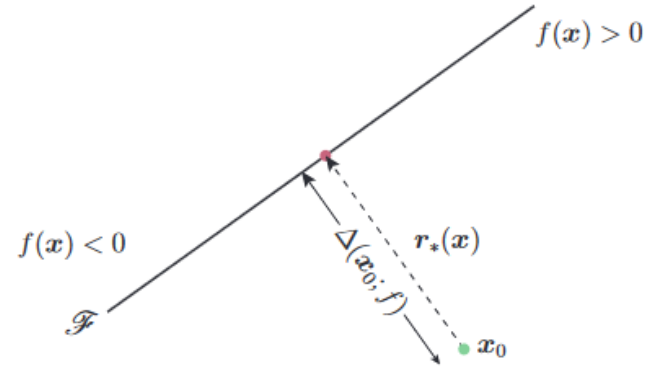
- Accuracy after FGSM Attack : 9.57%
- Accuracy after PGD Attack : 3.24%

Attacks : DeepFool

Deepfool : a simple and accurate method to fool deep neural networks. (Moosavi-Dezfooli et al)

If you want to find an adversarial example to an image :

- Look for the closest decision boundary
- Move the image towards a linear approximation of the decision boundary by orthogonally projecting it onto the boundary.
- Once it crosses the boundary it will be an adversarial image.



Adversarial examples for a linear binary classifier.

Algorithm 1 DeepFool for binary classifiers

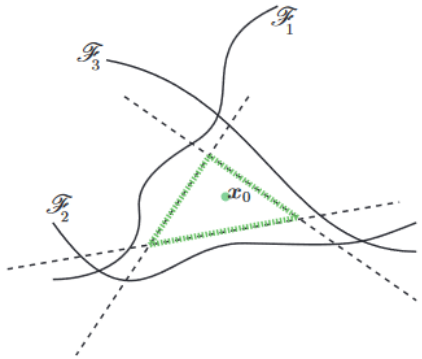
input: Image x, classifier f.
output: Perturbation $\hat{r}$.
Initialize $x_0 \leftarrow x$, $i \leftarrow 0$.
while $\text{sign}(f(x_i)) = \text{sign}(f(x_0))$ **do**
  $r_i \leftarrow -\frac{f(x_i)}{\|\nabla f(x_i)\|_2^2} \nabla f(x_i)$,
  $x_{i+1} \leftarrow x_i + r_i$,
  $i \leftarrow i + 1$.
end while
return $\hat{r} = \sum_i r_i$.

Figure 1 – DeepFool for binary classifiers

Attacks : DeepFool

DeepFool : The Multiclass Classifiers case :

- Calculate closest hyperplane from the n closest classes
- Calculate minimal projection vector
- Add perturbation and check if missclassified



For x_0 belonging to class 4, let $\mathscr{F}_k = \{x : f_k(x) - f_4(x) = 0\}$. The linearized zero level sets are shown in dashed lines and the boundary of the polyhedron $\tilde{P}_0$ in green.

Algorithm 2 DeepFool: multi-class case

input: Image x, classifier f.
output: Perturbation $\hat{r}$.

Initialize $x_0 \leftarrow x$, $i \leftarrow 0$.
while $\hat{k}(x_i) = \hat{k}(x_0)$ **do**
  for $k \neq \hat{k}(x_0)$ **do**
    $w'_k \leftarrow \nabla f_k(x_i) - \nabla f_{\hat{k}(x_0)}(x_i)$
    $f'_k \leftarrow f_k(x_i) - f_{\hat{k}(x_0)}(x_i)$
  end for
  $\hat{l} \leftarrow \arg\min_{k \neq \hat{k}(x_0)} \frac{|f'_k|}{\|w'_k\|_2}$
  $r_i \leftarrow \frac{|f'_{\hat{l}}|}{\|w'_{\hat{l}}\|_2^2} w'_{\hat{l}}$
  $x_{i+1} \leftarrow x_i + r_i$
  $i \leftarrow i + 1$
end while
return $\hat{r} = \sum_i r_i$

Figure 2 – DeepFool for Multiclass classifiers

Attacks : DeepFool

Comparison of added adversarial perturbation for DeepFool and FGSM

- Original image : whale
- Both DeepFool and FGSM perturb the image to be classified as turtle (targeted attack)
- DeepFool finds smaller perturbation

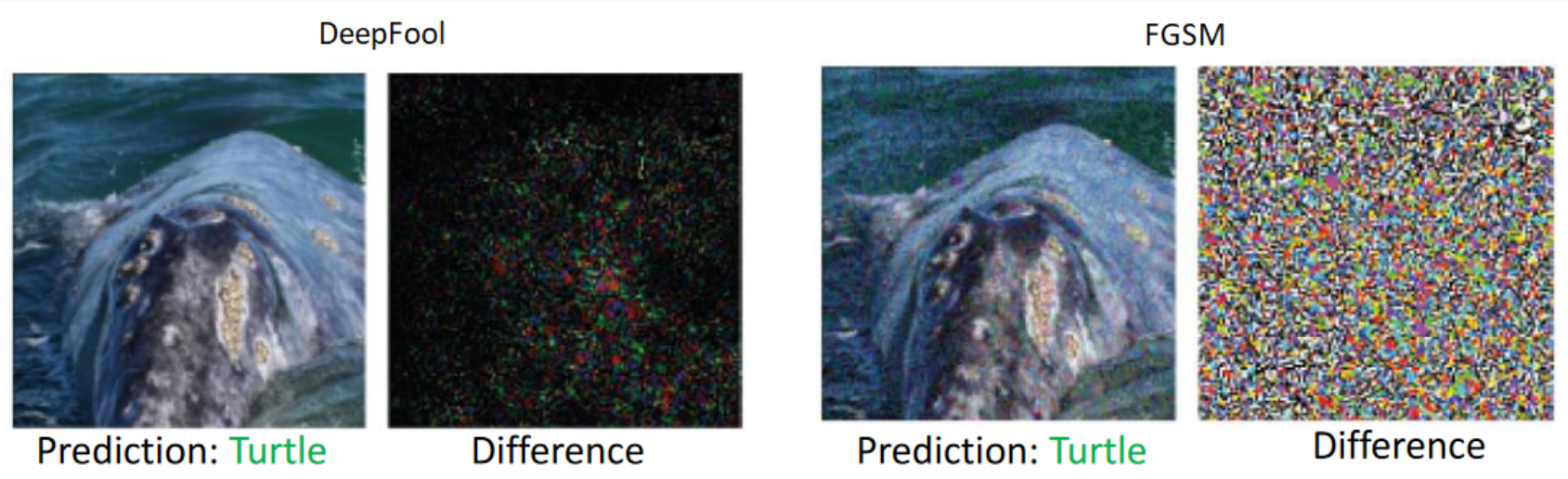


Figure 3 – DeepFool for Multiclass classifiers

Defense Mechanism : Adversarial Training

Adversarial training is training or retraining the target classification model using adversarial examples

- The training dataset is augmented with adversarial examples produced by known types of attacks
- By adding adversarial examples x_{adv} with true label y to the training set, the model will learn that x_{adv} belongs to the class y
- Adversarial training is one of the most common adversarial defense methods currently used in practice

Training					
Natural			**Adverserial**		
Nat_acc	PGD_acc	FGSM_acc	Nat_acc	PGD_acc	FGSM_acc
77.73	3.24	9.57	62.53	34.28	?

Defense Mechanism : Denoising

One other way to make a neural network robust versus attacks is to try and nullify the attacks through pre-processing the network entries.

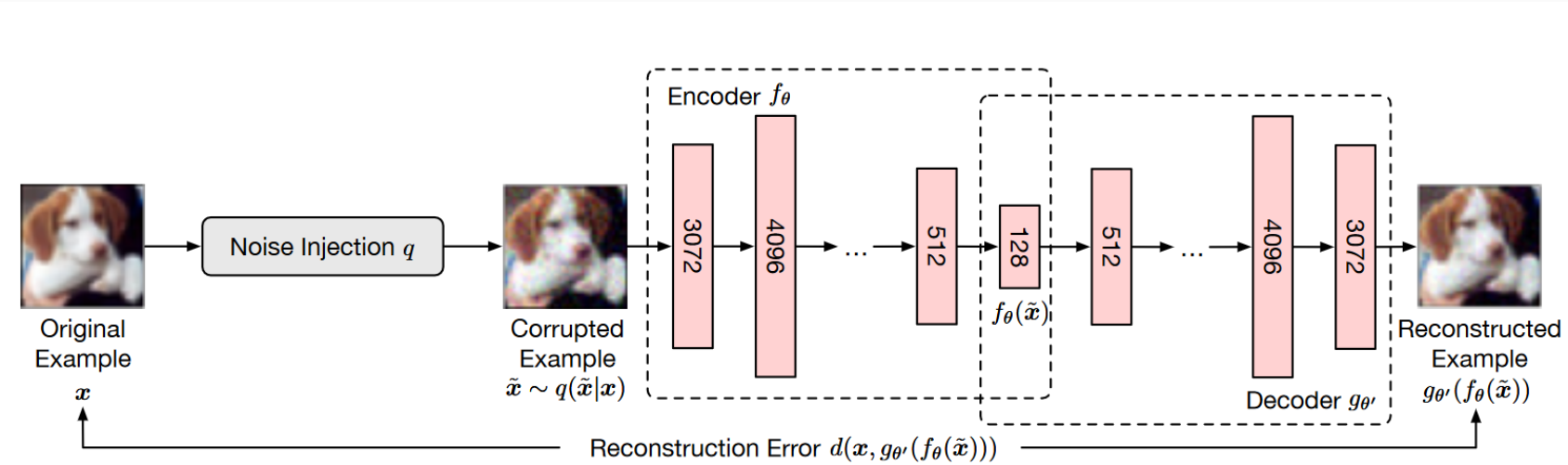


Figure 4 – An example of a denoising Autoencoder

In this case, our hope is to nullify the damage caused by the random noise through the auto-encoder.

Defense Mechanism : Denoising

First of all we trained a Deep Convolutional CNN as shown above for the denoising task on the CIFAR10 on added gaussian noise.

On the validation dataset : We achieve an accuracy of 75.39%. However, defending versus the FGSM attack, we achieve up to 30% accuracy.

Pros

- Doesn't affect greatly the accuracy of the base network
- Works on black boxes (does not need any training)

Cons

- Low Efficiency -> Doesn't reduce the attacks efficiency that much
- Requires separate training.
- A lot of Hyper parameters to fix

Defense Mechanism : Denoising - Next Steps

There are a lot of things we can improve in our denoising method that are promising

Improvements

- Running Multiple denoising layers on the images
- Training the denoiser on different noise patterns (salt-and-pepper noise, different gaussian noise...)
- Cross validation between multiple denoisers

Defense Mechanism : Restricting the Hidden Space

A way to explain the efficiency of adversarial attacks is the close proximity of different class samples in the learned feature space.

Therefor, Aamir Mustafa et al.(2019) [2] suggested a custom loss that forces the features for each class to lie inside a convex polytope that is maximally separated from the polytopes of other classes :

$$\mathcal{P}_\epsilon(x;\theta) = \{\mathcal{F}_\theta(x+\delta) \; s.t., \; \|\delta\| \leq \epsilon\} \tag{1}$$

Where $\mathcal{F}_\theta$ is a DNN with parameters θ

Defense Mechanism : Restricting the Hidden Space

For that, we introduce a new loss :

$$\mathcal{L}_{\mathrm{PC}}(\boldsymbol{x}, \boldsymbol{y}) = \sum_i \left\{ \left\| \boldsymbol{f}_i - \boldsymbol{w}^c_{y_i} \right\|_2 - \frac{1}{k-1} \sum_{j \neq y_i} \left(\left\| \boldsymbol{f}_i - \boldsymbol{w}^c_j \right\|_2 + \left\| \boldsymbol{w}^c_{y_i} - \boldsymbol{w}^c_j \right\|_2 \right) \right\} \quad (2)$$

Where $\boldsymbol{f}_i$ are input features and $\boldsymbol{w}_{y_i}$ there true class representative vector. $\boldsymbol{w}^c_{y_i}$ denotes the trainable class centroids.

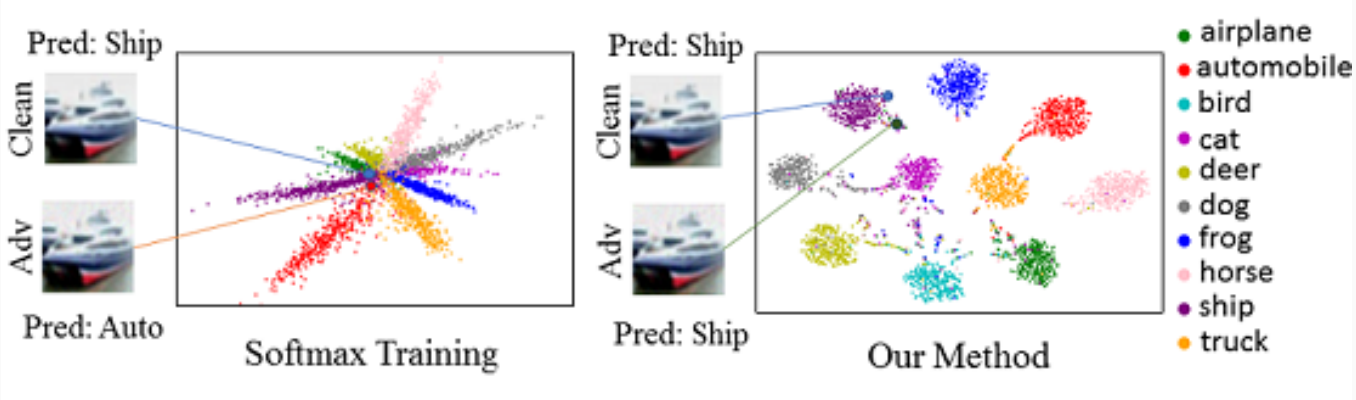


Figure 5 – Visual representation of the method

Defense Mechanism : Restricting the Hidden Space

After training for T=50 epochs on cross-entropy loss then T'=100 epochs on custom-loss + cross-entropy we get :

Model natural accuracy : 62.109 % vs accuracy after PGD attack : 41.13 %

References

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam.
Searching for mobilenetv3, 2019.

Aamir Mustafa, Salman Khan, Munawar Hayat, Roland Goecke, Jianbing Shen, and Ling Shao.
Adversarial defense by restricting the hidden space of deep neural networks.
09 2019.