Adversarial Attacks

Master Project

Team Members

Amin El Iraki

Olivier Randavel

Hadrien Mariaccia

Kenza Hammou

Github

Evaluating several adversarial attacks

Part 2

Carlini L_∞

The optimization problem :

$$\begin{cases} Minimize\ D(x, x+\delta) \\ st \quad C(x+\delta) = t, x+\delta \in [0,1]^n \end{cases} \longrightarrow \text{which is not linear}$$

f an objective function such that $C(x+\delta) = t$, if and only if $f(x+\delta) \leq 0$

$$\begin{cases} Minimize\ D(x, x+\delta) \\ st \quad f(x+\delta) \leq 0, x+\delta \in [0,1]^n \end{cases}$$

$$\begin{cases} Minimize\ D(x, x+\delta) + c * f(x+\delta) \\ st \quad x+\delta \in [0,1]^n \end{cases}$$

$f_6(x') = (max_{i \neq t}(Z(x')_i - Z(x')_t)^+$, Z softmax function

The L_∞ distance : $||x - x'||_\infty = max(|x_1 - x'_1|, .., |x_n - x'_n|)$

The library used is ***CarliniLInfMethod*** with several parameters :

- epsilon : modification's pixel's limit
- learning_rate : SGD optimization coefficient
- max_iter : the number of iterations

Basic Iterative Method

=> *an* iterative version of FGSM

- Fast Method : $$X^{adv} = X + \epsilon sign(\nabla_X J(X, y_{true}))$$

- Iterative Version : $$X_0^{adv} = X$$ $$X_{N+1}^{adv} = Clip_{X,\epsilon}\left\{X_N^{adv} + \alpha sign(\nabla_{X_N^{adv}} J(X, y_{true}))\right\}$$

The library used is ***BasicIterativeMethod*** with several parameters :

- epsilon : perturbation's limit
- eps_step : input variation at each iteration
- max_iter : the number of iterations

Newton Fool

The algorithm approximates the nearest classification boundary based on Newton's method for solving nonlinear equations.

1. The algorithm takes an input x0 and a neural network with a softmax layer Fs such that Fs(x0) =l , η, maxIter

2. Outputs the minimal perturbation required to misclassify the image

It tries to find small di to decrease the value of the function F as fast as possible to 0 : F(x0+d)_l≈0.

Starting with x0, they approximated F(xi) using a linear function step by step as follows.

$$F_s^l(x) \approx F_s^l(x_i) + \nabla F_s^l(x_i)(x - x_i) i \in [0, n] \rightarrow (1)$$

The library used is ***NewtonFool*** with several parameters :

- eta : characterizes the perturbation
- max_iter : the maximum number of iterations

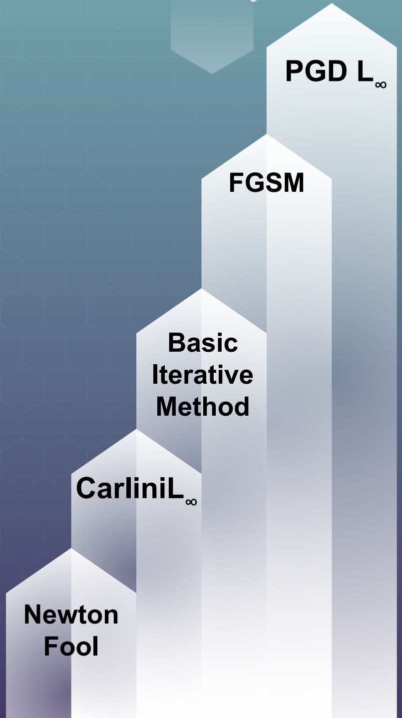
```
!pip install adversarial-robustness-toolbox
from art.attacks import NewtonFool
classifier = KerasClassifier(model=model, clip_values=(0, 1))
attack_cw = NewtonFool(classifier=classifier, eta=0.1, max_iter=40,
targeted=False, batch_size=1)
x_test_adv = attack_cw.generate(x_test)
loss_test, accuracy_test = model.evaluate(x_test_adv, y_test)
perturbation = np.mean(np.abs((x_test_adv - x_test)))
```

Evaluation metrics

La perturbation : $$\frac{1}{n}\sum_{i=1}^{n}\left|x_{i_{attacktest}} - x_{i_{test}}\right|$$

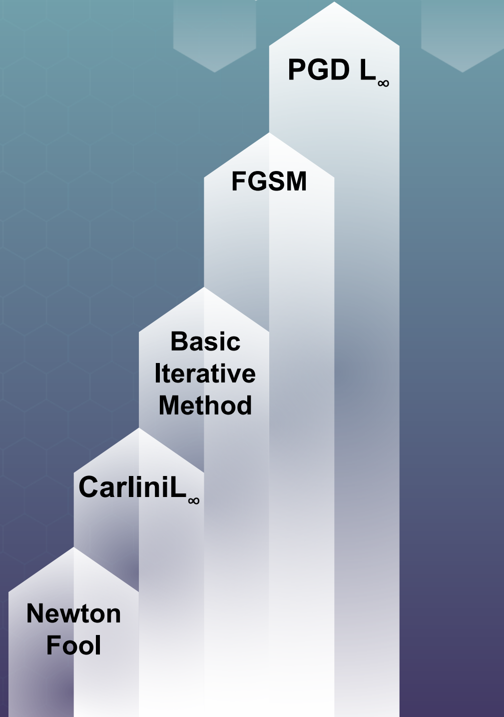
L’évaluation des résultats : **model.evaluate(x_test_adv, y_test)**

L’évaluation des résultats : **show_dataset_and_predictions(x_test_adv,**
y_test,
model,
class_to_name=class_to_name)

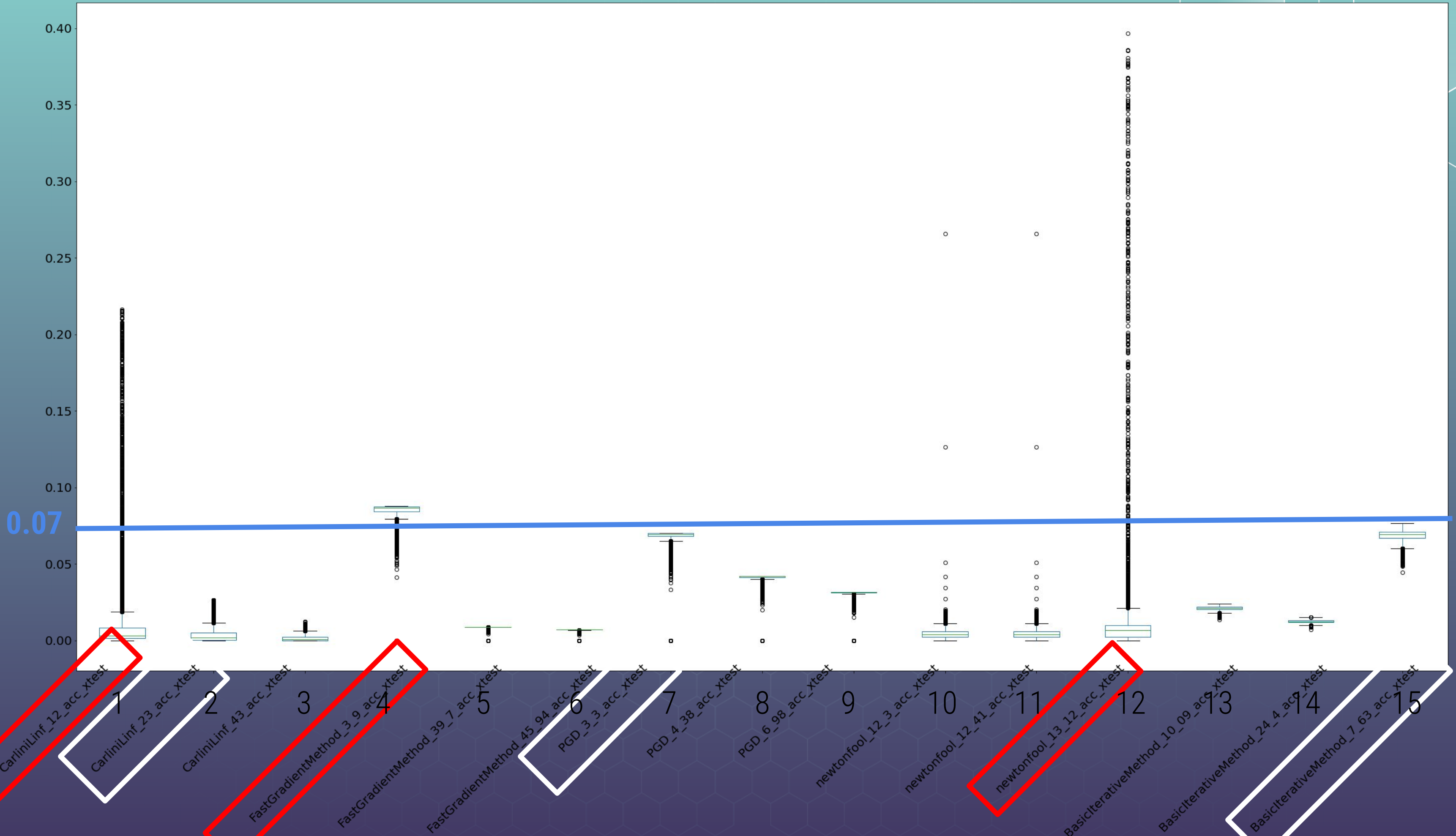


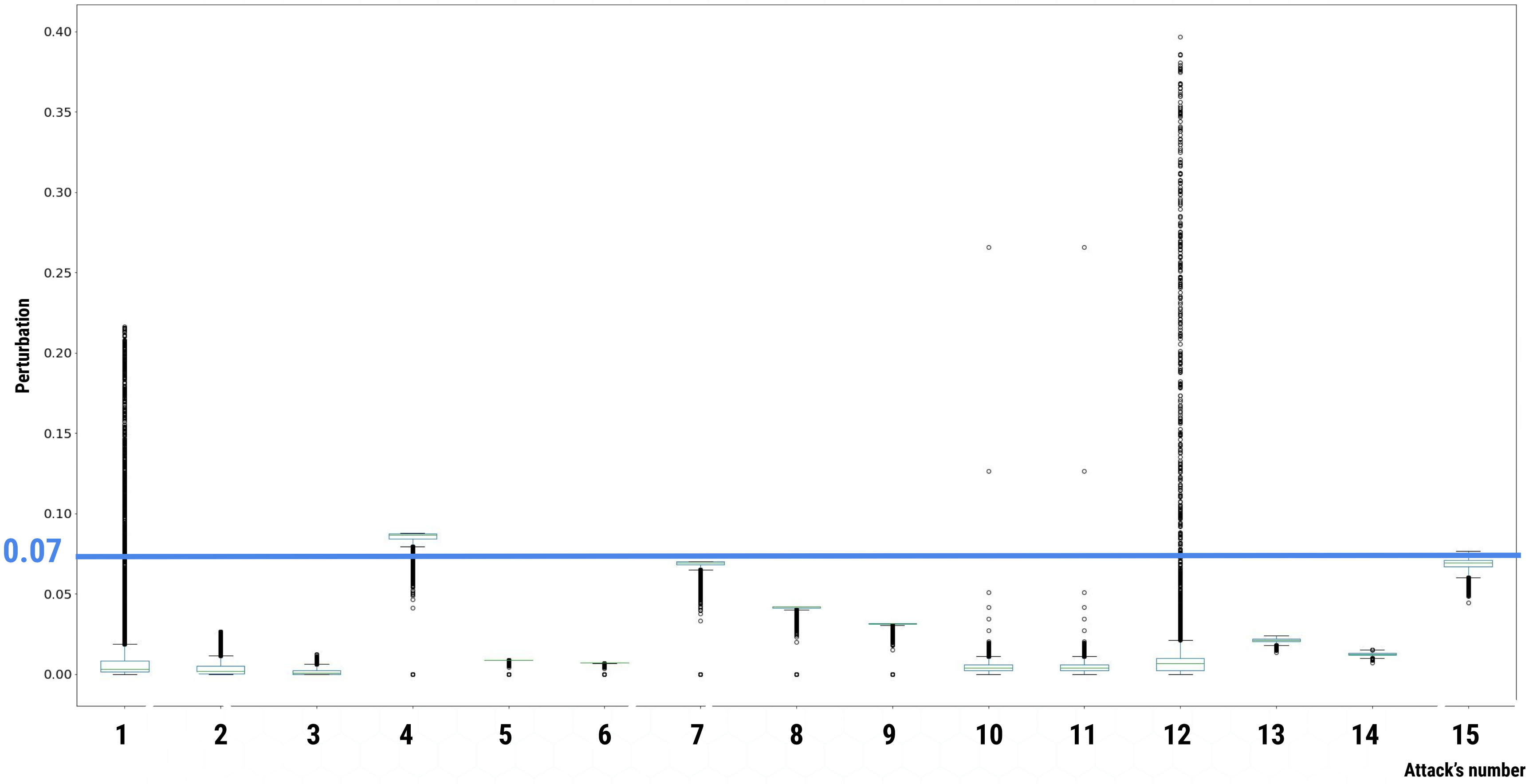
Results

	Output 6 pictures	x_test attack	avg perturbation	delta	epsilon	eps_step	max_iter	learning rate	batch size	targeted
FGSM		45.9%	0.010	0.008						
		39.7%	0.010	0.01						
		3.9%	0.080	0.1						
PGD L_infini		7.0%	0.030	0.01	0.008		2			
		4.4%	0.070	0.008	0.1		2			
		3.3%	0.070	008	0.008		5			
CarliniLInf		43.0%	0.001		0.03		4	0.01		
		23.0%	0.001		0.03		40	0.01		
		11.8%	0.010		0.25		100	0.3		
Newton Fool		12.3%	0.004		0.01		100		1	
		12.4%	0.004		0.01		40		1	
		13.1%	0.010		0.03		40		1	
Basic Iterative Method		24.4%	0.010		0.03	0.01	2		1	FALSE
		10.1%	0.020		0.03	0.03	40		1	FALSE
		7.6%	0.070		0.1	0.03	40		1	FALSE



Perturbations distribution per attack





Perturbation
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.07
0.05
0.00
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Attack's number

References

Carlini L_∞ : Towards Evaluating the Robustness of Neural Networks - Nicholas Carlini & David Wagner
https://arxiv.org/pdf/1608.04644.pdf

Newton Fool : Objective Metrics and Gradient Descent Algorithms for Adversarial Examples in Machine Learning
https://andrewxiwu.github.io/public/papers/2017/JWJ17-objective-metrics-and-gradient-descent-based-algorithms-for-adversarial-examples-in-machine-learning.pdf

Basic Iterative Method : Adversarial Machine Learning At Scale, Adversarial Examples in the Physical World
https://arxiv.org/pdf/1611.01236.pdf, https://arxiv.org/pdf/1607.02533.pdf

Latex maths : http://latex2png.com/

THANKS