Weekly Internship Report

General Information

Name: FROEHLY Jean-Baptiste - **Period:** Week 1 - 07/04/2025 to 11/04/2025

Weekly Objectives

- **Objective 1:** Understand the subject and its challenges
- **Objective 2:** Appropriate it and test initial things
- **Objective 3:** Selection of an interesting and complete dataset
- **Objective 4:** Conduct initial tests

Activities Completed

Monday

- Research on different types of attacks, their mechanisms, and implications
- Analysis of each to see at which level of the ML chain they operate, their strengths and weaknesses
- Initial Data Poisoning tests on a small dataset (~1200 rows): not very conclusive

Tuesday

- More in-depth research with a particular focus on Adversarial attacks
- Selection of more interesting datasets for my tests
- Development of a first lab based on an MRI image dataset for Alzheimer's disease detection

Wednesday

- Improvement of the lab to implement initial attack cases with ART
- Focus on adversarial attacks by inversion

Thursday

- Implementation of first adversarial attacks (NDNN and DNN), analysis of results
- Analysis of how FGSM and PGD attacks work

Friday

- Defense mechanisms analysis on adversarial attacks through NDNN and DNN
- Report redaction

Summary

Dataset

Original Source : ADNI database - ADNI Website

Kaggle source : Dataset link - Author : Utkarsh

Infos : ~490mo, 20257 images

This dataset provides a collection of preprocessed MRI brain scan images from the ADNI (Alzheimer's Disease Neuroimaging Initiative) project

Dataset structure

The images are arranged and classified in different categories. These images and categories are referenced in a file `train.csv` with two rows :

`train.csv`

id_code (string)	diagnosis (int)
AD-3471	4
CN-1819	0
LMCI-0760	3
...	...

Here are the different classifications and their diagnosis id

- CN - Cognitively Normal : `diagnosis=0` ; 4077 images
- MCI - Mild Cognitive Impairment : `diagnosis=1` ; 4073 images
- EMCI - Early Mild Cognitive Impairment : `diagnosis=2` ; 3958 images
- LMCI - Late Mild Cognitive Impairment : `diagnosis=3` ; 4074 images
- AD - Alzheimer's Disease : `diagnosis=4` ; 4075 images

Techniques

I mainly focused on **Fast Gradient Sign Method** et **Projected Gradient Descent**.

Fast Gradient Sign Method (FGSM)

- **Principle:** Attack based on gradient, simple but efficient
- **How it works:**
  - Calculates the gradient of the loss function with respect to the input image
  - Add a small perturbation in the direction that maximises the error
  - Formula: `x_adv = x + ε * sign(∇x J(θ, x, y))`
  - ε = 0.2 for my tests
- **Caracteristcs:**
  - One-shot attack
  - Disturbances often visible to the naked eye
  - Quick to calculate

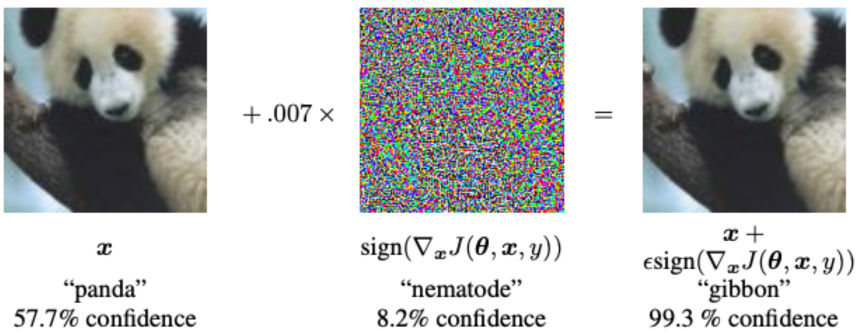

Illustration of the FGSM principle (source: TensorFlow.org) for ε = 0.007

Projected Gradient Descent (PGD)

- **Principle:** More powerful iterative version of **FGSM**
- **How it works:**
  - Apply FGSM in several small steps (10 iterations in my tests)

- Formula: `x_adv(t+1) = Proj(x_adv(t) + α * sign(∇x J(θ, x_adv(t), y)))`
  - After each step, projection into a field ε
- **Caracteristics:**
  - More sophisticated than FGSM
  - More discrete disturbances
  - More expensive to calculate

Defence on DNN

Implemented defence is **Adversarial Training**

This is an effective defence which simply consists of training the model with deliberately poisoned images.

Protects against adversarial evasion attacks such as these

Results

Training and tests carried out locally on Nvidia RTX 4060 Laptop GPU, Python 3.10.0

Tests : **10 epochs — 289.79 secondes**

Batch size : **32**

Clean Results

```
[*] Clean evaluation:
Accuracy: 0.7791
Average inference time per batch: 0.0010 seconds
Classification Report:
              precision    recall  f1-score   support
           0       0.48      0.47      0.47       834
           1       1.00      1.00      1.00       807
           2       0.99      0.99      0.99       778
           3       0.99      0.99      0.99       809
           4       0.47      0.48      0.47       824
    accuracy                           0.78      4052
   macro avg       0.78      0.78      0.78      4052
weighted avg       0.78      0.78      0.78      4052
```

Attack Implementation

FGSM

```
[*] Attack: FGSM (ε=0.2)
Accuracy: 0.4173
Average attack+inference time per batch: 0.0362 seconds
Classification Report:
precision    recall  f1-score   support
0       0.45      0.36      0.40       834
1       0.47      0.43      0.45       807
2       0.36      0.43      0.39       778
3       0.48      0.45      0.46       809
4       0.36      0.41      0.38       824
accuracy                           0.42      4052
macro avg       0.42      0.42      0.42      4052
weighted avg       0.42      0.42      0.42      4052
```

```
[TIME] test_evasion_attack executed in 9.34 seconds
```

PGD

```
[*] Attack: PGD (ε=0.2, iter=10)
Accuracy: 0.2648
Average attack+inference time per batch: 0.1489 seconds


Classification Report:
precision    recall  f1-score   support
0       0.39      0.28      0.33       834
1       0.29      0.28      0.29       807
2       0.19      0.30      0.23       778
3       0.29      0.27      0.28       809
4       0.24      0.20      0.22       824
accuracy                           0.26      4052
macro avg       0.28      0.27      0.27      4052
weighted avg       0.28      0.26      0.27      4052


[TIME] test_evasion_attack executed in 23.73 seconds
```

Defences - on DNN - Adversarial Training

Training : **Ratio : 0.5 — 460.06 seconds - 15 epochs**

Results after Adversarial Training

Global

```
[*] Evaluation under after defense attack:
Accuracy: 0.7853
Average inference time per batch: 0.0019 seconds


Classification Report:
precision    recall  f1-score   support
0       0.47      0.30      0.36       834
1       1.00      1.00      1.00       807
2       1.00      1.00      1.00       778
3       1.00      1.00      1.00       809
4       0.48      0.66      0.56       824
accuracy                           0.79      4052
macro avg       0.79      0.79      0.78      4052
weighted avg       0.78      0.79      0.78      4052


[TIME] evaluate_model executed in 8.15 seconds
```

FGSM

```
[*] Attack: FGSM (after defense)
Accuracy: 0.7823
Average attack+inference time per batch: 0.0432 seconds


Classification Report:
precision    recall  f1-score   support
0       0.46      0.30      0.36       834
1       1.00      1.00      1.00       807
2       1.00      1.00      1.00       778
3       1.00      1.00      1.00       809
4       0.48      0.64      0.55       824
accuracy                           0.78      4052
macro avg       0.79      0.79      0.78      4052
weighted avg       0.78      0.78      0.78      4052
```

```
[TIME] test_evasion_attack executed in 11.51 seconds
```

PGD

```
[*] Attack: PGD (after defense)
Accuracy: 0.7813
Average attack+inference time per batch: 0.1573 seconds

Classification Report:
precision    recall  f1-score   support
0       0.46      0.33      0.39       834
1       0.99      1.00      1.00       807
2       1.00      0.99      1.00       778
3       1.00      1.00      1.00       809
4       0.48      0.61      0.54       824
accuracy                           0.78      4052
macro avg       0.79      0.79      0.78      4052
weighted avg       0.78      0.78      0.78      4052

[TIME] test_evasion_attack executed in 25.69 seconds
```

Summary

```
Accuracy Metrics:
Initial clean accuracy: 0.7791
Accuracy under FGSM attack: 0.4173 (Drop: 0.3618)
Accuracy under PGD attack: 0.2648 (Drop: 0.5143)
Clean accuracy after defense: 0.7853
Accuracy under FGSM after defense: 0.7823 (Improvement: 0.3650)
Accuracy under PGD after defense: 0.7813 (Improvement: 0.5165)

Performance Metrics:
Standard training time: 289.79 seconds
Adversarial training time: 460.06 seconds (58.76% increase)
Average clean inference time: 0.0010 seconds per batch
Average FGSM attack+inference time: 0.0362 seconds per batch
Average PGD attack+inference time: 0.1489 seconds per batch
```

Images

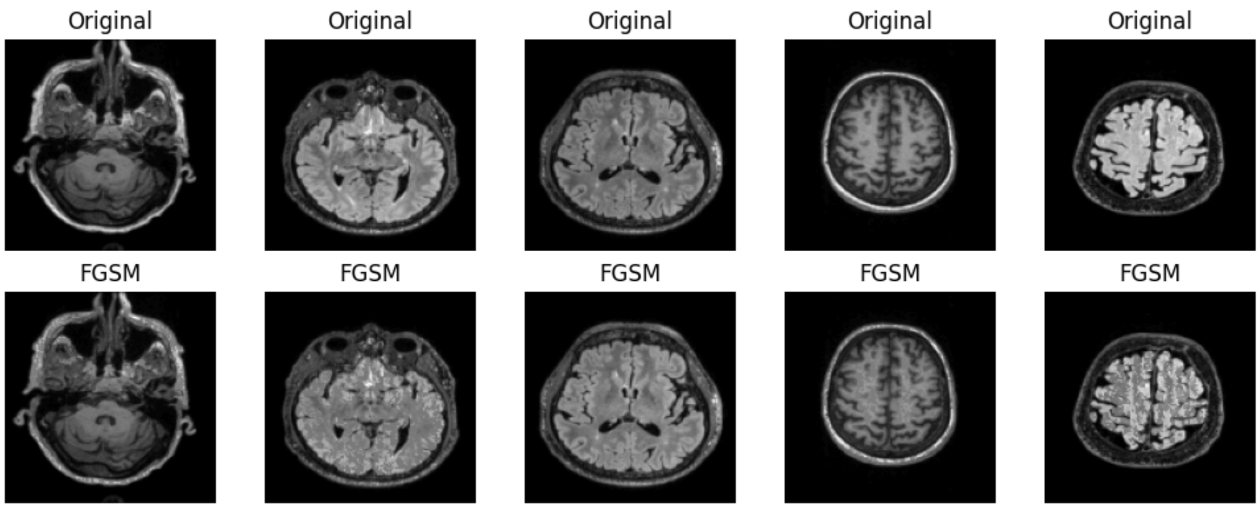

Fig.1 FGSM pour ε=0.2

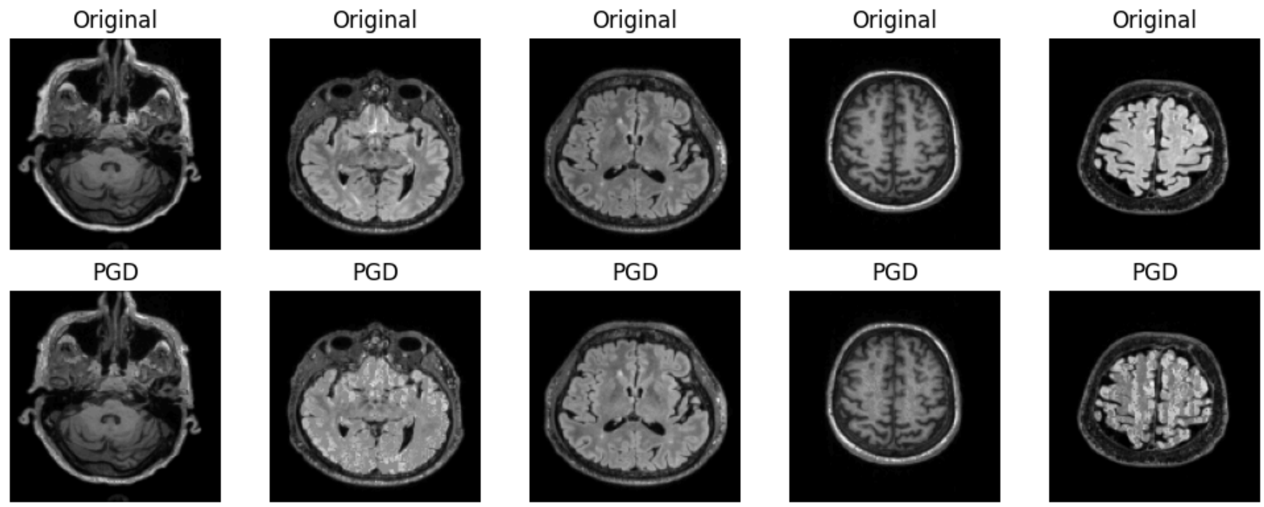

Fig.2 PGD pour ε=0.2 et iter=10

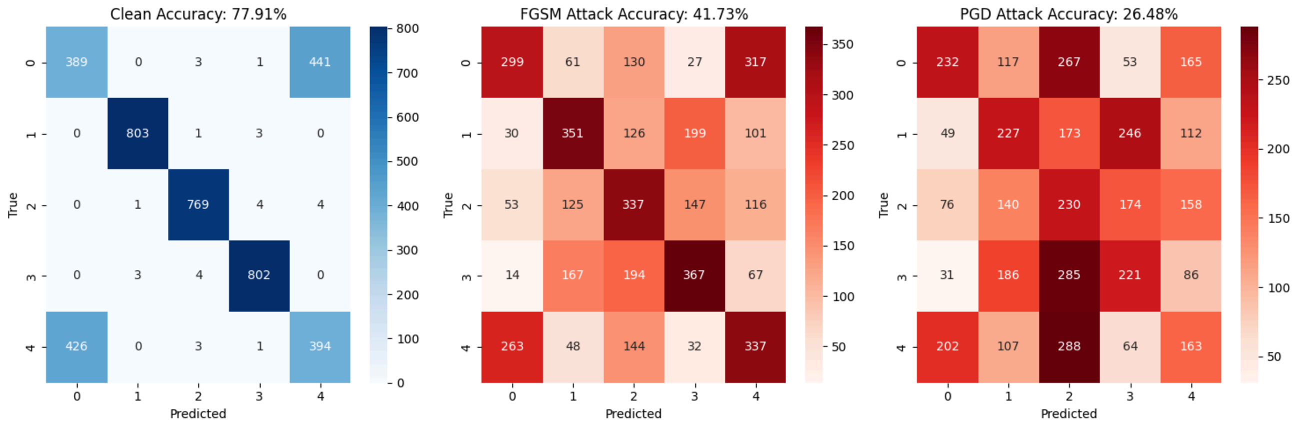

Fig.3 Matrices de confusion pre et post attaques

Mise en place des défenses

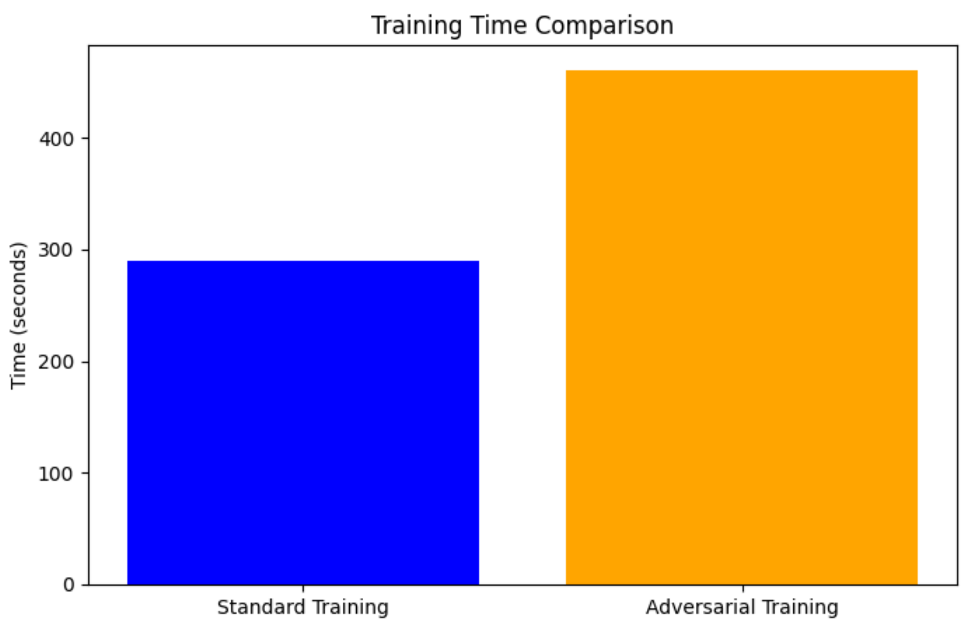

Fig.4 Comparaison des temps d'entrainement

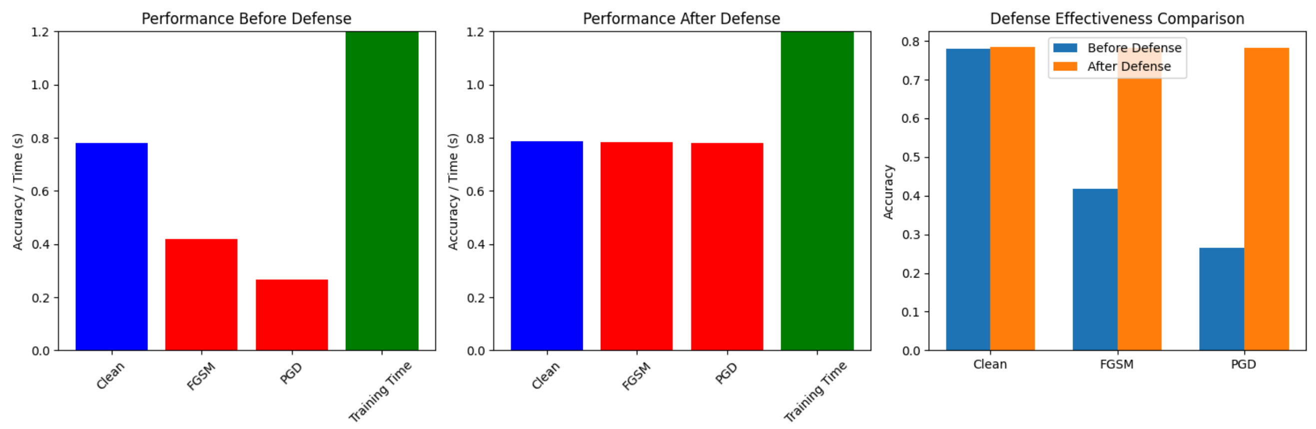

Fig.5 Efficacité des défenses

Analysis and Interpretation

Initial model performance

Clean datas

- Accuracy : 77.91%
- Very good performance in classes 1, 2 and 3 (f1-score ~ 1.00)
- Poor performance in classes 0 and 4 (f1-score ~ 0.47)
  - These classes are often **confused with each other** as shown by the confusion matrix (c.f. fig. 3) pre-attack: class 0 is often predicted as 4 and vice versa
  - This corresponds to the final diganostics of Cognitively Normal (0) and Alzheimer's Disease (4).

Under FGSM attack

- Accuracy : 41.73% → **loss of 36.18 points**
- Significant deterioration, particularly for classes 0 and 4
- Widely dispersed confusion (mixture of all classes)
- Attack disrupts initial separation, especially for 'weak' classes

Under PGD attack

- Accuracy : 26.48% → **loss of 51.43 points**
- PGD is much more powerful and effective than FGSM
- Strong confusion on all classes, prediction seems close to random
- Confusion matrix (c.f. fig 3) shows a very flat, mixed distribution

After Adversarial Training

On clean datas

- Accuracy : 78.53% → slight improvement on the original (77.91%)
- Classes 1, 2 and 3 remain perfect
- Class 4 improves (recall 0.48 → 0.66)
- Class 0 remains a little weak (recall = 0.30)

Resistance to attack

- **FGSM after defences : 78.23%**
  - Almost no impact → the model becomes robust
  - Performance similar to that without attack
- **PGD after defences : 78.13%**
  - Impressive resistance with a gain of **+51.65 points** compared to before defence

- Reliable classes (0 and 4) remain sensitive, but overall performance is holding up well

System performance

Criteria	Standard	Adversarial
Training time	289.79s	460.06s → +59%
Inference per batch	0.0010s	0.0019s
Attack + inference FGSM	0.0362s	0.0432s
Attack + inference PGD	0.1489s	0.1573s

Adversarial Training **increases robustness at the cost of a longer training time, but inference remains fast**

Conclusion

- The basic model is highly accurate but vulnerable to adversarial attacks, particularly PGD.
- After adversarial training, the model becomes more robust while retaining good accuracy on the clean data.
- The trade-off in training time is clearly worth it, especially given the area covered by this dataset.

Difficulties Encountered

- **Problem 1:** Initial dataset was too limited
- **Problem 2:** Compatibility issues between Python libraries

Next Steps

- **Objective 1:**Digging deeper into attack and defence mechanisms, changing parameters, seeing the limits
- **Objective 2:**Implemtation of defence on NDNN
- **Objective 3:**Looking at other types of attack

FROEHLY Jean-Baptiste, Friday 11/04/2025