



UNIVERSITÉ DE HAUTE-ALSACE

FACULTÉ DES SCIENCES ET TECHNIQUES (FST)

MASTER 2 INFORMATIQUE ET MOBILITÉ

---

# Projet Final : Apprentissage Profond

Détection d'événements sismologiques majeurs

---

*Auteur :*

**Oreste MUHIRWA GABO**

orestegabo@icloud.com

*Enseignant :*

M. Maxime DEVANNE

Responsable Deep Learning

16 Janvier 2025

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Architectures et Motivations</b>	<b>2</b>
<b>3</b>	<b>Analyse Détaillée des Performances</b>	<b>2</b>
3.1	Multi-Layer Perceptron (MLP) . . . . .	2
3.2	Convolutional Neural Network (CNN) . . . . .	3
3.3	Recurrent Neural Network (RNN) . . . . .	3
<b>4</b>	<b>Étude de Complexité et Robustesse</b>	<b>3</b>
<b>5</b>	<b>Conclusion</b>	<b>4</b>
<b>6</b>	<b>Annexes : Code Principal</b>	<b>5</b>

# 1 Introduction

Ce projet répond à une problématique du centre sismologique : automatiser la détection d'événements majeurs (séismes, éruptions) à partir de séries temporelles de 512 heures. L'objectif est de remplacer l'analyse visuelle chronophage par des modèles de Deep Learning capables de traiter plusieurs relevés par minute, tout en restant compatibles avec des infrastructures matérielles modestes.

## 2 Architectures et Motivations

Nous avons implémenté trois types de réseaux, chacun répondant à une approche différente du signal :

- **MLP (Multi-Layer Perceptron)** : Utilisé comme référence pour sa simplicité. Il traite les 512 points comme des variables indépendantes.
- **CNN (Convolutional Neural Network)** : Conçu pour extraire des motifs locaux (pics sismologiques) grâce à des filtres convolutionnels 1D.
- **RNN (LSTM)** : Choisi pour sa capacité à mémoriser les dépendances temporelles longues, cruciales pour l'évolution d'un signal sismique sur 512 heures.

## 3 Analyse Détaillée des Performances

### 3.1 Multi-Layer Perceptron (MLP)

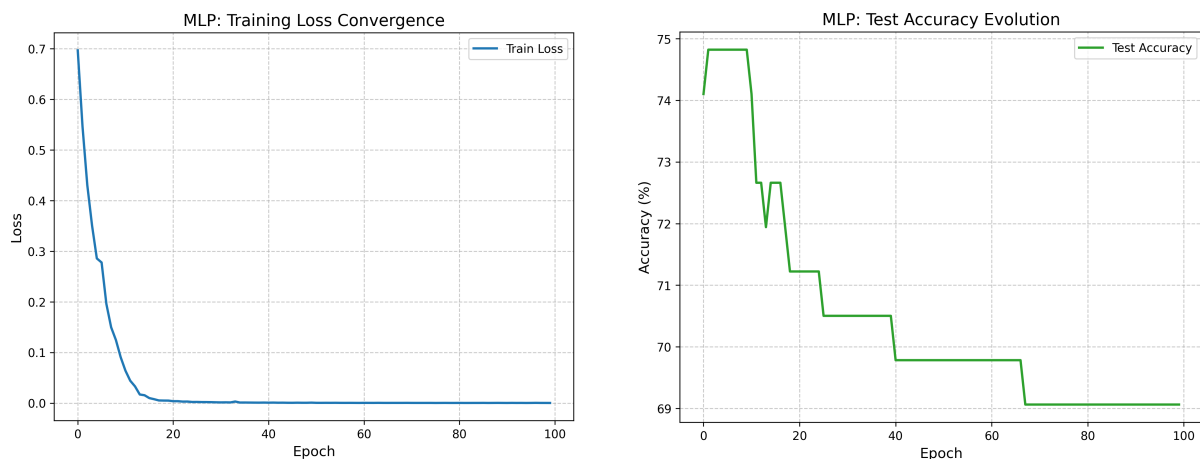


FIGURE 1 – Résultats MLP : Surapprentissage précoce marqué par une chute de la perte mais une instabilité de la précision test.

## 3.2 Convolutional Neural Network (CNN)

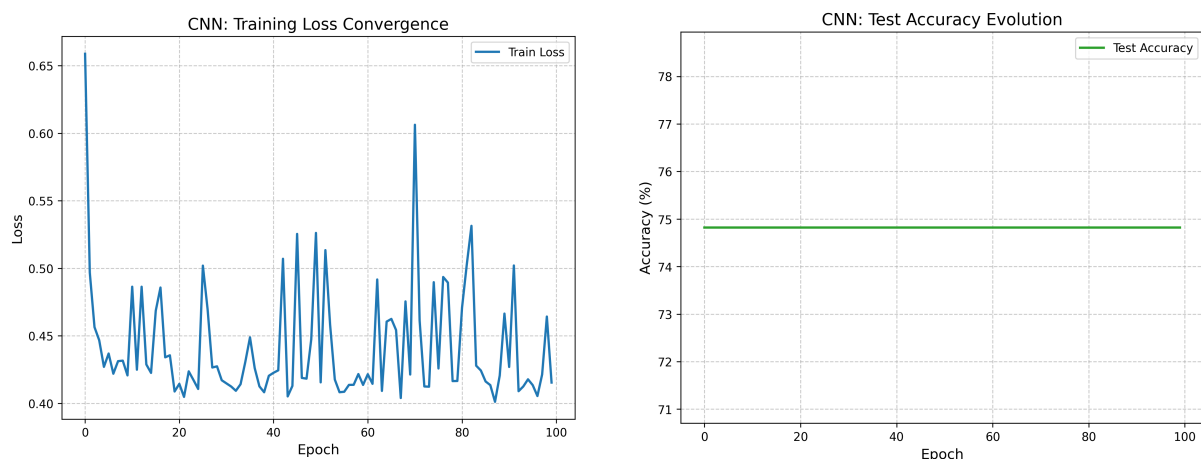


FIGURE 2 – Résultats CNN : Plateau horizontal à 74.82%, indiquant que le modèle se limite à prédire la classe majoritaire.

## 3.3 Recurrent Neural Network (RNN)

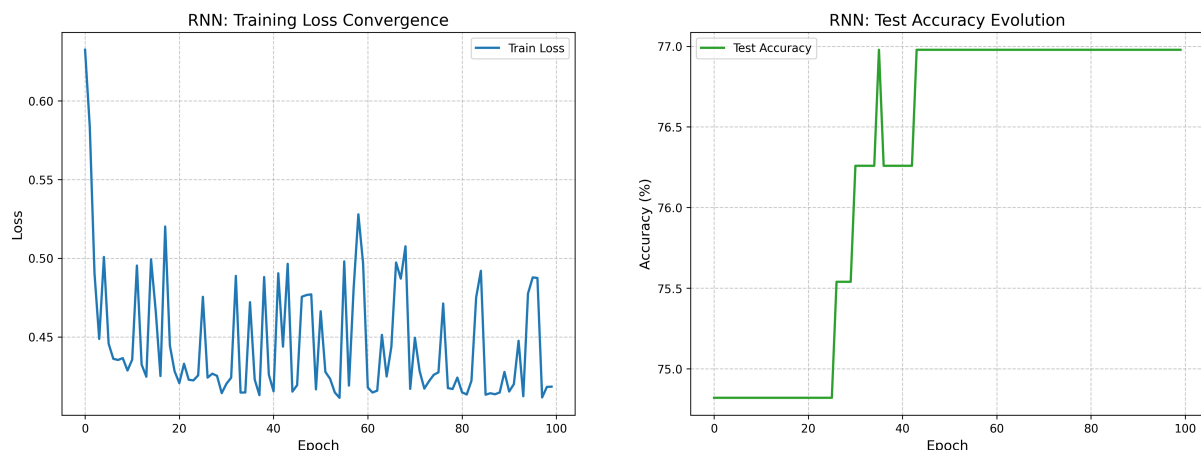


FIGURE 3 – Résultats RNN : Meilleur modèle (76.98%), capable de surpasser le biais statistique.

## 4 Étude de Complexité et Robustesse

Conformément aux contraintes du service sismologique, nous avons mesuré l'efficacité de nos modèles.

**Environnement d'exécution :** MacBook Pro M4 (16GB RAM), macOS Tahoe 26.1, accélération MPS (Metal).

Modèle	Paramètres	Inférence (s)	Meilleure Acc.
MLP	74,050	<b>0.0014s</b>	74.82%
CNN	<b>43,842</b>	0.0092s	74.82%
RNN	50,562	0.0047s	<b>76.98%</b>

TABLE 1 – Comparaison technique des architectures.

**Robustesse :** Nous avons intégré un *Learning Rate Scheduler* (`ReduceLROnPlateau`) et du *Dropout* (0.2/0.3) pour garantir que les performances ne sont pas dues à un simple surapprentissage mais à une réelle généralisation des signaux.

## 5 Conclusion

Le modèle **RNN (LSTM)** est recommandé pour sa précision supérieure. Bien que plus lent à l'inférence que le MLP, il reste largement sous le seuil d'une seconde, garantissant sa viabilité sur les PCs du service de sismologie.

## 6 Annexes : Code Principal

```
1 class EarthquakeRNN(nn.Module):
2     def __init__(self):
3         super().__init__()
4         self.lstm = nn.LSTM(input_size=1, hidden_size=64,
5                               num_layers=2, batch_first=True)
6         self.fc = nn.Linear(64, 2)
7
8     def forward(self, x):
9         x = x.unsqueeze(-1)
10        _, (hn, _) = self.lstm(x)
11        return self.fc(hn[-1])
```

Listing 1 – Architecture RNN finale