



Université Paris-Saclay

Formation : Master Informatique - IOT
Année universitaire 2024-2025

Évaluation et adaptation de modèles de nowcasting des précipitations dans le contexte des données HDRain

Nom : Cheikh Ahmed Tidiane Mané

Entreprise : HDRain

Responsable entreprise : Mounir Mahdade

Mission de Stage

Dans un contexte où la précision des prévisions météorologiques à court terme (nowcasting) devient un enjeu stratégique pour de nombreux secteurs (agriculture, assurance, gestion des risques climatiques), HDRain souhaite renforcer ses outils de prévision en tirant parti des avancées en Deep Learning.

Ma mission de stage s'inscrit dans cette dynamique d'innovation. Elle consiste à explorer, adapter et optimiser des architectures de Deep Learning pour le nowcasting des précipitations, en s'appuyant sur les données spécifiques produites par HDRain. L'objectif principal est de développer un modèle capable de :

- Exploiter les particularités des mesures HDRain, notamment leur haute résolution et leur densité spatiale ;
- Fournir des prévisions fiables en temps réel, dans un cadre contraint par les exigences opérationnelles de latence et de performance.

Pour cela, le travail repose sur l'utilisation et l'adaptation de modèles existants (RainNet, Pysteps), combinée à une reconstruction rigoureuse des jeux de données et à une évaluation fine des performances.

Ce stage contribue ainsi à la modernisation de la chaîne de prévision météorologique de HDRain, en intégrant des approches d'apprentissage profond à l'état de l'art, avec pour ambition d'améliorer significativement la précision des prévisions à très court terme.

Remerciements

Je tiens à exprimer ma profonde gratitude envers toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce travail. Votre soutien, vos conseils et votre bienveillance ont été déterminants tout au long de ce parcours.

Je remercie tout particulièrement mon encadrant Mounir Mahdade pour toutes les compétences transmises, sa patience et sa sollicitude. Je remercie également tous mes collègues pour leur disponibilité et le cadre de travail convivial.

Dédicace

Je dédie ce travail :

À ma défunte maman, Awa Ly, qui m'a toujours apporté tendresse et réconfort face à l'adversité. Elle est le visage de l'amour et a toujours été présente pour célébrer mes réussites.

À mon papa, qui élève mes ambitions vers des sommets que je n'aurais su atteindre. Il a toujours financé ma scolarité et m'encourage à avoir un impact significatif dans le monde.

À mes frères Boubacar Mané, Ahmadou Mané, Aziz Mané et Kalidou Ndiaye, qui ont toujours été des modèles de réussite et d'excellence, et qui m'ont donné le goût de la science et du savoir.

À ma sœur Aïssatou, que j'appelle « ma petite maman », et qui croit énormément en moi.

À mon soutien et référent, Ceerno Yahya Barro, qui m'exhorte à la constance, à l'excellence et m'accompagne dans toutes mes entreprises.

À mon ami Seydina Oumar, ainsi qu'à toute sa famille, qui sont devenus pour moi une véritable seconde famille.

À mes grands frères Amadou Sarr et Shaykhana Abass Sall, qui m'ont toujours prodigué de précieux conseils pour l'atteinte de mes objectifs.

Table des matières

Mission de stage	1
Introduction	5
1 Entreprise d'accueil	6
1.1 Présentation de HDRain	6
1.2 Technologie de HDRain	6
1.3 Problématique de stage	7
2 Données utilisées	8
2.1 Présentation des données HDRain	8
2.2 Organisation des données	8
3 Modèles de Prévision	10
3.1 Modèle d'advection	10
3.1.1 Présentation du modèle	10
3.1.2 Organisation des données d'advection	10
3.2 Modèle RainNet	11
3.2.1 Présentation générale	11
3.2.2 Architecture du modèle	11
3.2.3 Fonctionnement du modèle	12
3.2.4 Normalisation des données	12
3.2.5 Fonction de coût et optimisation	12
3.2.6 Organisation des données	12
3.2.7 Utilisation dans le stage	12
3.3 Modèle PySTEPS	13
3.3.1 Présentation du modèle	13
3.3.2 Organisation des données	13
3.3.3 Utilisation dans le stage	13
3.4 Modèle baseline : Prédicteur nul	14
3.4.1 Présentation du modèle	14
3.4.2 Organisation des données	14
3.4.3 Utilisation dans le stage	14
4 Prétraitement des données	15
5 Comparaison initiale	16
5.1 Métriques utilisées	16
5.2 Résultats	17

6 Finetuning de RainNet	23
6.1 Finetuning de RainNet	23
6.2 Résultats	24
7 Réentrainement de RainNet	28
7.1 Procédure	28
7.2 Résultats	28
Conclusion	32
Bibliographie	33
Table des annexes	35

Introduction

La prévision météorologique à très court terme, également appelée *nowcasting*, est devenue un enjeu majeur pour de nombreux secteurs tels que l'agriculture, la gestion des risques climatiques, les assurances ou encore les services publics. Elle consiste à fournir des prévisions sur des horizons temporels allant de quelques minutes à deux heures, avec un niveau de précision spatiale et temporelle suffisamment élevé pour permettre une prise de décision rapide et efficace. Parmi les phénomènes météorologiques, les précipitations constituent un défi particulier en raison de leur grande variabilité et de leur caractère souvent localisé.

Historiquement, les modèles d'advection et les approches physiques classiques ont été largement utilisés pour le nowcasting. Ces méthodes reposent sur le suivi des systèmes pluvieux observés par radar ou satellite et sur leur extrapolation dans le temps. Si elles offrent de bonnes performances pour des événements continus et organisés, elles présentent cependant des limites dès qu'il s'agit de phénomènes convectifs ou de précipitations à évolution rapide. Les erreurs augmentent alors significativement avec le temps d'anticipation, réduisant la fiabilité des prévisions.

Ces dernières années, les approches basées sur le Deep Learning ont ouvert de nouvelles perspectives. En exploitant de larges volumes de données météorologiques, elles sont capables de capturer des structures complexes et d'améliorer les prévisions, notamment pour des phénomènes localisés. Parmi ces modèles, RainNet montre un fort potentiel.

Le stage présenté dans ce rapport s'inscrit dans cette dynamique. Réalisé au sein de l'entreprise HDRain, spécialisée dans la mesure et la prévision des précipitations à haute résolution, il a pour objectif principal **d'améliorer le modèle actuel et d'explorer les modèles de deep learning**. Ces données, issues d'une technologie brevetée utilisant l'atténuation des signaux satellites de télévision, présentent des caractéristiques particulières qu'il est nécessaire de prendre en compte pour optimiser les performances du modèle.

L'enjeu de ce travail est donc double :

1. Utiliser des méthodes de deep learning pour améliorer la prévision des précipitations avec les données HDRain
2. Évaluer de manière rigoureuse les gains obtenus par rapport aux méthodes existantes, notamment le modèle d'advection utilisé en interne.

Cette étude s'inscrit plus largement dans la volonté de HDRAIN de se hisser au cœur de l'innovation.

1 Entreprise d'accueil

1.1 Présentation de HDRain

HDRain est une jeune entreprise innovante spécialisée dans les mesures et prévisions météorologiques, notamment des précipitations, à partir d'une technologie brevetée basée sur l'atténuation des signaux satellites de télévision causée par la pluie. Cette méthode permet de produire des données météorologiques à haute résolution spatiale et temporelle, ainsi que des prévisions à très court terme. (1 à 2 heures).

1.2 Technologie de HDRain

Sa technologie repose sur une approche unique : l'analyse de l'atténuation des signaux des satellites de télévision géostationnaires lorsqu'ils traversent des zones pluvieuses [1]. Ces signaux, affaiblis par la présence de gouttes de pluie, permettent de quantifier la pluviométrie le long de la ligne de visée entre chaque station au sol et le satellite.

En installant un réseau dense de capteurs, HDRain transforme ces mesures linéiques en **cartes de précipitations spatialisées**, rafraîchies toutes les minutes et couvrant des zones entières (par exemple, 1 000 capteurs couvrant 60 000 km² dans le sud-est de la France). Cette approche permet non seulement de mesurer, mais aussi d'anticiper les épisodes pluvieux avec une haute précision spatiale et temporelle (grilles de 500 m × 500 m).

Le traitement des données s'effectue en deux étapes :

1. **Détection qualitative** (pluie / non pluie) grâce à des modèles de Deep Learning entraînés sur des millions d'exemples.
2. **Estimation quantitative** (pluviométrie) via une modélisation physique de l'atténuation du signal, combinée à l'assimilation et à l'interpolation spatiale entre les données des stations pour générer des cartes cohérentes de précipitations dans le temps et l'espace.

Cette technologie permet à HDRain de proposer une observation à haute résolution temporelle ainsi qu'un nowcasting performant (prévisions à 1 - 2 heure) adapté aux enjeux climatiques et opérationnels actuels dans des secteurs tels que l'agriculture, les assurances ou la gestion des risques hydrométéorologiques

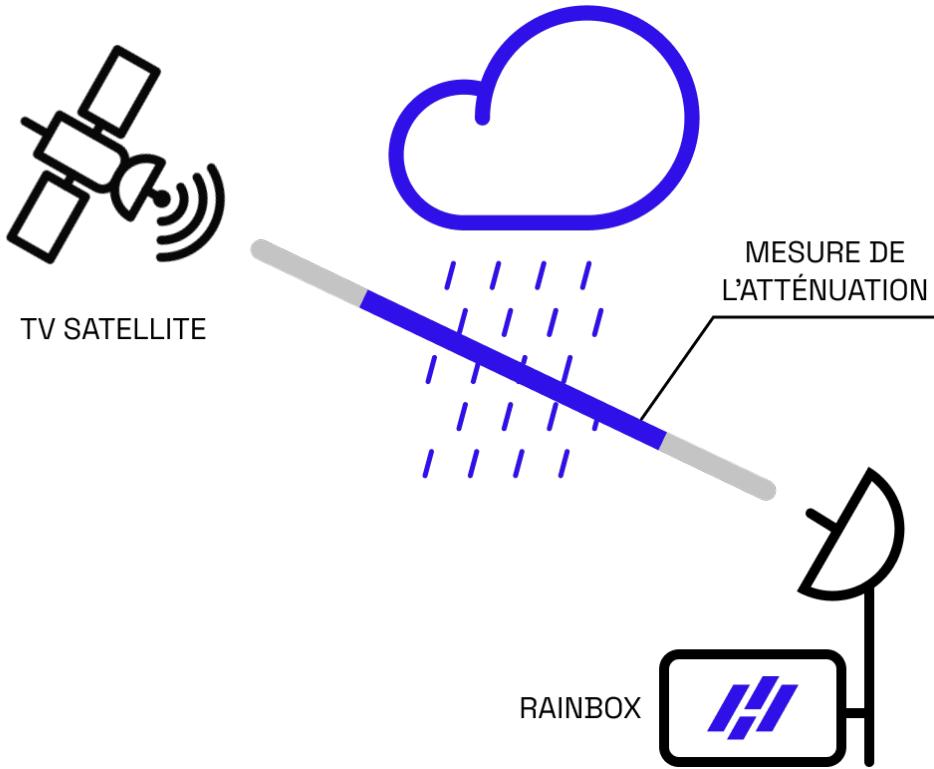


Figure 1.1 – Technologie HDRain.

1.3 Problématique de stage

L'utilisation d'un modèle d'advection pour le nowcasting des précipitations reste trop simpliste pour capturer la dynamique atmosphérique complexe. Cette limitation incite HDRain à s'interroger sur l'apport de techniques plus avancées, en particulier les approches de deep learning, capables de mieux représenter l'évolution spatio-temporelle en général.

La problématique centrale de ce stage est donc la suivante : Dans quelle mesure les méthodes de deep learning peuvent-elles améliorer la précision et la fiabilité du nowcasting des précipitations par rapport aux modèles d'advection traditionnels ?

Afin de répondre à cette problématique, nous suivrons la démarche suivante :

1. Comparer les performances du modèle d'advection avec celles de modèles plus sophistiqués : **RainNet**, **PySTEPS** et celle du modèle naïf : le **prédicteur nul**.
2. Adapter (*finetuning*) et réentraîner le modèle **RainNet** sur les données spécifiques de **HDRain**.
3. Évaluer les nouveaux gains de performance apportés par rapport aux méthodes existantes.

Cette approche permettra de déterminer dans quelle mesure les techniques de *deep learning* peuvent améliorer la précision et la fiabilité du nowcasting des précipitations à partir des données **HDRain**.

2 Données utilisées

2.1 Présentation des données HDRain

Les données utilisées dans le cadre de ce stage proviennent du réseau de capteurs déployé par HDRain. Chaque capteur mesure en continu l'atténuation du signal satellite causée par les précipitations, permettant d'estimer l'intensité des pluies en un point donné. Ces mesures ponctuelles sont ensuite traitées et interpolées pour produire des **cartes de précipitations spatialisées (A)**.

Les principales caractéristiques de ces données sont les suivantes :

- **Résolution spatiale** : 500 m × 500 m, ce qui permet de capturer finement les phénomènes pluvieux localisés.
- **Résolution temporelle** : 5 minute, ce qui est particulièrement adapté au suivi de phénomènes rapides et instables.
- **Zone géographique couverte** : variable selon le cas d'étude (par exemple : sud-est de la France, Abidjan, Rabat, etc....).
- **Format de fichier** : GeoTIFF compressé (.tif.gz), téléchargé automatiquement depuis le S3 Scaleway qui constitue la base de données HDRAIN.

Chaque fichier représente une carte de précipitations estimée en millimètres par heure (mm/h). Les cartes sont issues d'un traitement combiné : modélisation physique de l'atténuation du signal, puis assimilation spatiale via un réseau dense de capteurs. Ces cartes servent de base pour entraîner et évaluer les modèles de nowcasting développés au cours du stage.

2.2 Organisation des données

L'enregistrement des données HDRain repose sur une procédure automatisée développée en Python, permettant de télécharger, décompresser, traiter et stocker les données brutes dans un fichier NetCDF exploitable pour l'apprentissage automatique.

Téléchargement automatisé

Les données HDRain sont disponibles sous forme de fichiers compressés (.tif.gz), accessibles à la minute près. Un script Python génère dynamiquement un fichier Bash pour chaque pas de temps, afin de télécharger les fichiers nécessaires depuis un serveur distant à l'aide de la commande `swift`. Les fichiers sont enregistrés localement dans un répertoire temporaire.

Décompression et traitement

Chaque fichier compressé est immédiatement décompressé à l'aide du module `gzip`, puis supprimé après extraction afin de limiter l'utilisation de l'espace disque.

Les images GeoTIFF sont ouvertes avec la bibliothèque `rioxarray`, qui permet un accès optimisé en mémoire. Chaque image est ensuite convertie en tableau NumPy représentant l'intensité des précipitations (en mm/h). Si nécessaire, les axes latitude et longitude sont réordonnés pour correspondre au format croissant attendu.

Écriture en NetCDF avec gestion mémoire

Les images traitées sont stockées dans un fichier NetCDF. Celui-ci contient :

- une dimension temporelle `Time` non bornée,
- deux dimensions spatiales `lat` et `lon`,
- une variable `precip` (type `float32`), compressée avec `zlib=True`.

L'écriture dans le fichier NetCDF est effectuée de manière **incrémentale (streaming)**, c'est-à-dire :

- Chaque image est lue, traitée et immédiatement enregistrée dans le fichier.
- Aucune image n'est gardée en mémoire après son enregistrement.
- Les fichiers intermédiaires sont supprimés dès qu'ils sont traités.

Cette méthode permet de traiter de longues séries temporelles tout en conservant une empreinte mémoire minimale, rendant le processus compatible avec des machines aux ressources limitées.

Robustesse et filtrage

Le script vérifie systématiquement l'existence des fichiers avant chaque opération pour éviter les doublons. Les pas de temps pour lesquels les données sont manquantes ou corrompues sont automatiquement ignorés. L'index temporel du NetCDF n'est incrémenté que si l'image est valide.

Période considérée

La période traitée va du **1^{er} juillet 2023 à 00h00 au 31 Mars 2025 à 23h55**, avec un pas de temps de **5 minutes**, soit **183 744 cartes** téléchargées, traitées et intégrées dans le fichier final `input.nc`. Les données de l'année 2025 sont utilisés pour tester les performances.

3 Modèles de Prévision

3.1 Modèle d'advection

3.1.1 Présentation du modèle

Le modèle d'advection repose sur une approche physique simple et largement utilisée en météorologie pour le nowcasting des précipitations. Il s'agit d'un modèle basé sur l'hypothèse que les systèmes pluvieux se déplacent horizontalement sous l'effet du vent, sans modification significative de leur structure au cours du temps. Cette hypothèse permet d'extrapoler dans le futur les images de pluie observées.

Concrètement, le modèle suppose que les cellules de précipitation visibles à un instant donné seront présentes quelques minutes plus tard à un autre endroit, déterminé par un champ de vecteurs de déplacement. Ce champ peut être calculé à partir des déplacements observés entre deux images successives de pluie (par exemple, à l'aide d'un algorithme de type *optical flow* ou de corrélation croisée).

Le modèle d'advection présente plusieurs avantages :

- simplicité de mise en œuvre et faible coût en ressources ;
- absence de besoin d'apprentissage ou de données externes.

Cependant, il présente aussi des limites importantes :

- incapacité à modéliser les changements structurels dans les systèmes pluvieux ;
- dégradation rapide des performances avec l'horizon de prévision ou dans le cas de phénomènes convectifs rapides.

Dans le cadre de ce stage, le modèle d'advection est utilisé comme **modèle de référence** (baseline) pour évaluer les performances d'autres modeles de Deep Learning(RainNet) et statistiques(Pysteps). Il permet de situer objectivement les gains apportés par les approches d'apprentissage profond.

3.1.2 Organisation des données d'advection

Afin de préparer les entrées et sorties nécessaires à l'évaluation du modèle d'advection, un pipeline a été mis en place pour extraire, structurer et stocker les données dans un format exploitable.

Les données d'advection sont issues d'archives compressées (.tgz) contenant, pour chaque date, une séquence d'images GeoTIFF représentant les précipitations toutes les 5 minutes. Chaque archive peut contenir des sous-dossiers ou des fichiers .zip, eux-mêmes contenant les images radar horodatées.

Extraction des séquences : Pour chaque date cible (nommée t), une image de départ (`image_t`) est extraite. Elle est ensuite associée à une séquence de 16 images suivantes, couvrant une période de 80 minutes (de $t + 5$ min à $t + 80$ min). Ces séquences sont utilisées comme cibles pour l'évaluation du modèle d'advection, qui repose sur une extrapolation de l'image à t .

Les images sont lues à la volée, en mémoire, sans extraction physique sur le disque. Le traitement utilise la bibliothèque `rioxarray` pour lire les fichiers GeoTIFF et `xarray` pour les concaténer le long d'un axe temporel nommé `lead_time`.

Structure des données : Nous avons alors une séquence de 16 images futures représentant l'évolution réelle des précipitations à intervalles de 5 minutes.

Cette structure permet de comparer directement les prévisions produites par le modèle à la vérité pour différents horizons temporels.

Les horodatages sont encodés au format NetCDF standard (seconds since 1970-01-01), et les pas de temps (lead_time) sont exprimés en minutes.

Normalisation des coordonnées : Afin de garantir la cohérence spatiale des données, les longitudes et les latitudes sont systématiquement triées dans l'ordre croissant. Ce tri est appliqué à toutes les images d'une séquence, assurant un alignement parfait entre les différents pas de temps.

TraITEMENT PARALLÈLE ET ROBUSTE : Le traitement est parallélisé à l'aide de ProcessPoolExecutor, permettant de traiter plusieurs archives simultanément. En cas d'erreur sur une archive ou un fichier, celui-ci est ignoré sans bloquer le reste du processus. Des vérifications sont effectuées pour s'assurer que chaque séquence contient bien les 16 images futures attendues ; les cas incomplets sont automatiquement exclus du jeu de données.

3.2 Modèle RainNet

3.2.1 Présentation générale

RainNet est un modèle de Deep Learning conçu spécifiquement pour la prévision à très court terme des précipitations (*nowcasting*) à partir d'images radar. Il s'appuie sur des réseaux de neurones convolutionnels (CNN) et a été initialement développé pour prédire l'intensité continue des précipitations à un horizon de +5 minutes, en utilisant plusieurs années de données météorologiques radar fournies par le service météorologique allemand (DWD).

RainNet a été entraîné sur le produit radar RY, une mosaïque haute qualité issue de 17 radars Doppler allemands, avec une résolution spatiale de $1 \text{ km} \times 1 \text{ km}$ et une résolution temporelle de 5 minutes.

3.2.2 Architecture du modèle

L'architecture de RainNet (C) s'inspire des familles de modèles U-Net et SegNet, souvent utilisées pour des tâches de segmentation binaire. Il adopte une structure de type **encodeur-décodeur symétrique** avec des connexions latérales (skip connections) entre les couches de même niveau dans les deux branches. Cela permet de préserver les détails spatiaux tout en exploitant des caractéristiques globales.

L'encodeur extrait progressivement les caractéristiques en réduisant la taille spatiale à l'aide de couches de pooling, tandis que le décodeur reconstruit la sortie à l'aide d'upsampling. Chaque niveau comporte plusieurs convolutions suivies d'une activation ReLU.

RainNet comprend :

- 20 couches de convolution;
- 4 couches de *max pooling*;
- 4 couches d'*upsampling*;
- 2 couches de *dropout* pour la régularisation;
- 4 *skip connections* entre encodeur et décodeur.

Pour des raisons architecturales, les dimensions d'entrée doivent être des multiples de 2^n , avec $n = 4$ (nombre de niveaux de pooling). Les images sont donc redimensionnées de 900×900 à 928×928 pixels par **padding miroir**.

3.2.3 Fonctionnement du modèle

Deux modes sont possibles :

- **Régression** : sortie continue (en mm/h), activation linéaire.
- **Segmentation** : sortie binaire (ex. dépassement d'un seuil), activation sigmoïde.

Dans ce stage, seul le mode régression sera utilisé.

3.2.4 Normalisation des données

Les valeurs de précipitations sont normalisées selon la transformation logarithmique suivante :

$$x_{\text{transformed}} = \ln(x_{\text{raw}} + 0,01)$$

Cela permet de réduire la variance et de stabiliser l'apprentissage, notamment pour les fortes intensités.

3.2.5 Fonction de coût et optimisation

RainNet est entraîné à l'aide de la fonction de perte **log-cosh**, définie comme :

$$\text{Loss} = \frac{1}{n} \sum_{i=1}^n \ln(\cosh(\text{now}_i - \text{obs}_i))$$

où now_i est la valeur prédite, obs_i la valeur observée, et n le nombre de pixels.

L'optimisation est réalisée avec l'algorithme Adam, sur des mini-batchs de taille 2 pendant 10 époques. La convergence a été observée à la 8^e époque.

3.2.6 Organisation des données

Chaque séquence de prévision utilisée par RainNet est composée :

- **En entrée** : 4 images consécutives espacées de 5 minutes ($t - 15, t - 10, t - 5, t$), traitées par transformation logarithmique et *padding* pour correspondre à la taille attendue par le modèle (928×928).
- **En sortie** : 16 images futures représentant les horizons de prévision de 5 à 80 minutes (pas de 5 minutes), produites de manière itérative en réinjectant chaque prédiction dans l'entrée pour l'échéance suivante.

Les séquences sans pluie significative (intensité maximale $< 0,01$ mm/h) sont écartées pour réduire le temps de calcul. Les sorties sont reconvertis en valeurs physiques puis stockées au format **NetCDF4** avec toutes les métadonnées spatiales et temporelles nécessaires à leur exploitation.

3.2.7 Utilisation dans le stage

Dans le cadre de ce stage, une version open source du modèle RainNet sera utilisée [2]. L'architecture sera reprise telle quelle, avec ses poids pré-entraînés ou réinitialisés. Le modèle sera ensuite **finetuné** sur les données spécifiques d'HDRain, après mise au bon format

Cette phase d'adaptation va permettre de tirer parti de la robustesse de RainNet tout en l'adaptant aux spécificités des données HDRain.

3.3 Modèle PySTEPS

3.3.1 Présentation du modèle

PySTEPS (*Python Short-Term Ensemble Prediction System*) [3] est une bibliothèque open source développée pour la prévision des précipitations à très court terme (*nowcasting*) à partir d'images radar. Elle permet de produire à la fois des prévisions déterministes et probabilistes, grâce à un pipeline modulaire basé sur l'advection des champs de pluie.

Le fonctionnement de PySTEPS repose sur trois étapes principales :

1. **Estimation du champ de mouvement** : calcul de la vitesse et de la direction des structures pluvieuses entre deux images successives (méthodes FFT ou *optical flow*).
2. **Advection des précipitations** : déplacement des champs de pluie selon le champ de mouvement estimé.
3. **Ajout de perturbations stochastiques (optionnel)** : génération de plusieurs membres d'ensemble pour représenter l'incertitude des prévisions.

Parmi ses atouts, PySTEPS est modulaire et performant sur des horizons de prévision allant de 5 minutes à 2 heures. Sa limite principale est commune à toutes les méthodes purement advectives : l'hypothèse que les structures de précipitations se déplacent sans évolution interne, ce qui peut réduire la précision lors d'épisodes convectifs rapides.

3.3.2 Organisation des données

Les données utilisées avec PySTEPS sont issues du fichier input interpolated contenant :

- des cartes de précipitations interpolées sur une grille régulière ;
- les coordonnées spatiales (latitude et longitude) ;
- les horodatages des observations.

Pour chaque séquence de prévision :

- **Entrée** : 4 images consécutives espacées de 5 minutes ($t-15, t-10, t-5$ et t minutes).
- **Sortie** : 16 images futures représentant les horizons de prévision de 5 à 80 minutes.

Les séquences sans pluie significative (intensité maximale $< 0.01 \text{ mm/h}$) sont exclues afin d'optimiser les calculs. Les coordonnées spatiales sont ordonnées pour garantir la cohérence entre les pas de temps.

3.3.3 Utilisation dans le stage

Dans le cadre de ce stage, PySTEPS a été intégré afin de fournir une référence supplémentaire aux côtés du modèle d'advection interne et de RainNet. Nous nous baserons sur le dépôt github [4]

La mise en œuvre suit les étapes suivantes :

1. **Chargement des données** depuis le fichier NetCDF input_interpolated.nc.
2. **Paramétrage** : $n_inputs = 4$, $n_forecast = 16$, horizons de prévision de 5 à 80 minutes.
3. **Filtrage** des séquences sans pluie.
4. **Estimation du champ de mouvement** à l'aide de la méthode **Lucas-Kanade**.
5. **Nowcasting par extrapolation** pour générer les 16 images de prévision.
6. **Export** des prévisions au format NetCDF compressé avec les variables :
 - lead_time : pas de temps de prévision ;
 - lat, lon : coordonnées spatiales ;
 - precip_forecast : précipitations prévues.

Cette utilisation de Pystep permet de challenger RainNet.

3.4 Modèle baseline : Prédicteur nul

3.4.1 Présentation du modèle

Le prédicteur nul est un modèle extrêmement simple qui, pour chaque pas de temps et chaque pixel, renvoie systématiquement une valeur de précipitation égale à zéro. Il ne repose sur aucune donnée d'entrée ni aucun calcul : sa sortie est constante et indépendante du contexte météorologique.

Ce type de baseline est souvent utilisé comme point de comparaison minimal, permettant d'évaluer si des modèles plus élaborés (Advection, PySTEPS, RainNet) apportent une réelle valeur ajoutée par rapport à une prédition naïve.

3.4.2 Organisation des données

Le prédicteur nul ne nécessite aucune donnée en entrée. La structure des fichiers de sortie est toutefois identique à celle des autres modèles, afin de pouvoir comparer les résultats avec les mêmes métriques et scripts d'évaluation. Chaque prévision produite contient :

- les pas de temps (lead_time);
- les coordonnées spatiales (latitudes et longitudes);
- un champ de précipitations composé uniquement de zéros.

3.4.3 Utilisation dans le stage

Dans le cadre de ce stage, le prédicteur nul est utilisé pour :

- fournir un score de référence correspondant à une prévision totalement inerte;
- servir de contrôle pour vérifier la sensibilité des métriques d'évaluation;
- illustrer l'écart de performance entre un modèle trivial et les modèles avancés.

Les prévisions générées par ce modèle sont produites en initialisant directement des matrices de zéros aux dimensions attendues (temps, latitude, longitude) pour chaque horizon de prévision. Cette baseline, bien que simpliste, permet de mettre en perspective l'apport des méthodes plus complexes testées dans ce travail.

4 Prétraitement des données

Après la génération du fichier NetCDF brut contenant les cartes de précipitations (`input.nc`), une phase de prétraitement est nécessaire pour adapter les données aux exigences du modèle de Deep Learning. Cette étape vise à harmoniser la résolution spatiale, gérer les valeurs manquantes, normaliser les données et structurer les entrées dans un format adapté à l'architecture du modèle.

Lecture des coordonnées

Les coordonnées géographiques des pixels (longitudes et latitudes) sont extraites à partir d'un fichier Geo-TIFF de référence (`lonlat.tif`). Une nouvelle grille régulière est ensuite générée avec une résolution de 0,01° pour l'interpolation.

Première interpolation : harmonisation de la résolution spatiale

Les données HDRain ont une résolution spatiale de 500 m, tandis que le modèle RainNet a été entraîné sur des données radar à résolution de 1 km. Pour garantir une cohérence entre les jeux de données, une interpolation spatiale est appliquée afin de projeter les données HDRain sur une grille à 1 km de résolution.

Padding : mise au format du modèle

RainNet s'attend à recevoir des images d'entrée de taille fixe 928×928 pixels. Pour respecter cette contrainte structurelle, les cartes interpolées sont redimensionnées par **padding réfléchi**, c'est-à-dire un remplissage intelligent des bords pour obtenir une forme adaptée aux réseaux convolutifs.

Traitements des valeurs et normalisation

Pour chaque pas de temps :

- Les valeurs manquantes ou invalides (selon NaN, -1) sont remplacées par 0.
- Les images sont interpolées puis normalisées selon la transformation $\log(x + \epsilon)$, avec $\epsilon = 0,01$.

Création du fichier NetCDF final

Les données prétraitées sont enregistrées dans un nouveau fichier NetCDF nommé `input_processed.nc` avec les précipitations normalisées.

Traitements en streaming

Pour limiter l'usage de la mémoire RAM, le traitement est réalisé de manière incrémentale (carte par carte). À chaque étape, l'image est chargée, transformée, écrite dans le fichier NetCDF, puis libérée. Ce mode de fonctionnement permet de traiter efficacement de très grands volumes de données.

5 Comparaison initiale

Nous pouvons regarder La période de comparaison va du **1^{er} juillet 2023 à 00h00 au 31 Mars 2024 à 23h55**, soit **9 mois** de données avec un pas de temps de **5 minutes**.

5.1 Métriques utilisées

L'évaluation des performances des différents modèles (RainNet, Advection, Pystep, Prédicteur nul) est réalisée à l'aide d'un ensemble de métriques permettant d'analyser à la fois la précision globale, la capacité à détecter les événements pluvieux et la qualité des prévisions quantitatives. Les métriques sont calculées pour différents seuils d'intensité (0 mm/h, 0,01 mm/h, 0,1 mm/h) et pour chaque pas de temps de prévision (*lead time*).

1. Métriques d'erreur globale

Ces indicateurs mesurent la différence entre les précipitations prévues et observées sur l'ensemble de la grille.

- **MAE (Mean Absolute Error)** :

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

- **MSE (Mean Squared Error)** :

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

Ces deux métriques sont également calculées uniquement sur les pixels où de la pluie est observée :

- **MAE_RAIN** : MAE restreint aux pixels pluvieux.
- **MSE_RAIN** : MSE restreint aux pixels pluvieux.

2. Métriques de classification binaire

Pour chaque seuil, les cartes de précipitations sont transformées en cartes binaires (pluie / pas de pluie).

On définit :

- **TP** (vrais positifs) : pluie observée et prédite;
- **TN** (vrais négatifs) : pas de pluie observée ni prédite;
- **FP** (faux positifs) : pluie prédite mais non observée;
- **FN** (faux négatifs) : pluie observée mais non prédite.

À partir de ces valeurs, on calcule :

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

$$CSI = \frac{TP}{TP + FP + FN}$$

$$FAR = \frac{FP}{TP + FP}$$

$$POD = \frac{TP}{TP + FN}$$

3. Analyse multi-classes

En plus de la classification binaire, une classification multi-classes a été réalisée en fonction de seuils d'intensité définis :

$$[0, 0,01, 0,1, 1, +\infty] \text{ (mm/h)}$$

Cela permet de générer des matrices de confusion multi-classes pour chaque pas de temps de prévision, afin d'évaluer la capacité des modèles à prédire correctement différentes intensités de pluie.

4. Suivi temporel des performances

Toutes les métriques sont calculées pour chaque *lead time* (5 min, 10 min, ..., 80 min), ce qui permet de visualiser l'évolution des performances des modèles au fur et à mesure que l'horizon de prévision s'éloigne.

5.2 Résultats

En observant les sorties générées par les différents modèles (B), on distingue clairement leurs tendances prévisionnelles respectives. Le modèle d'advection se limite à translater les pixels de pluie selon un vecteur correspondant au champ de vent, tandis que les modèles RainNet et PySTEPS cherchent à capturer non seulement le déplacement, mais aussi les variations de forme et d'intensité des précipitations.

Passons à présent à une analyse plus fine des cartes produites par les différents modèles à travers l'examen des métriques de performance.

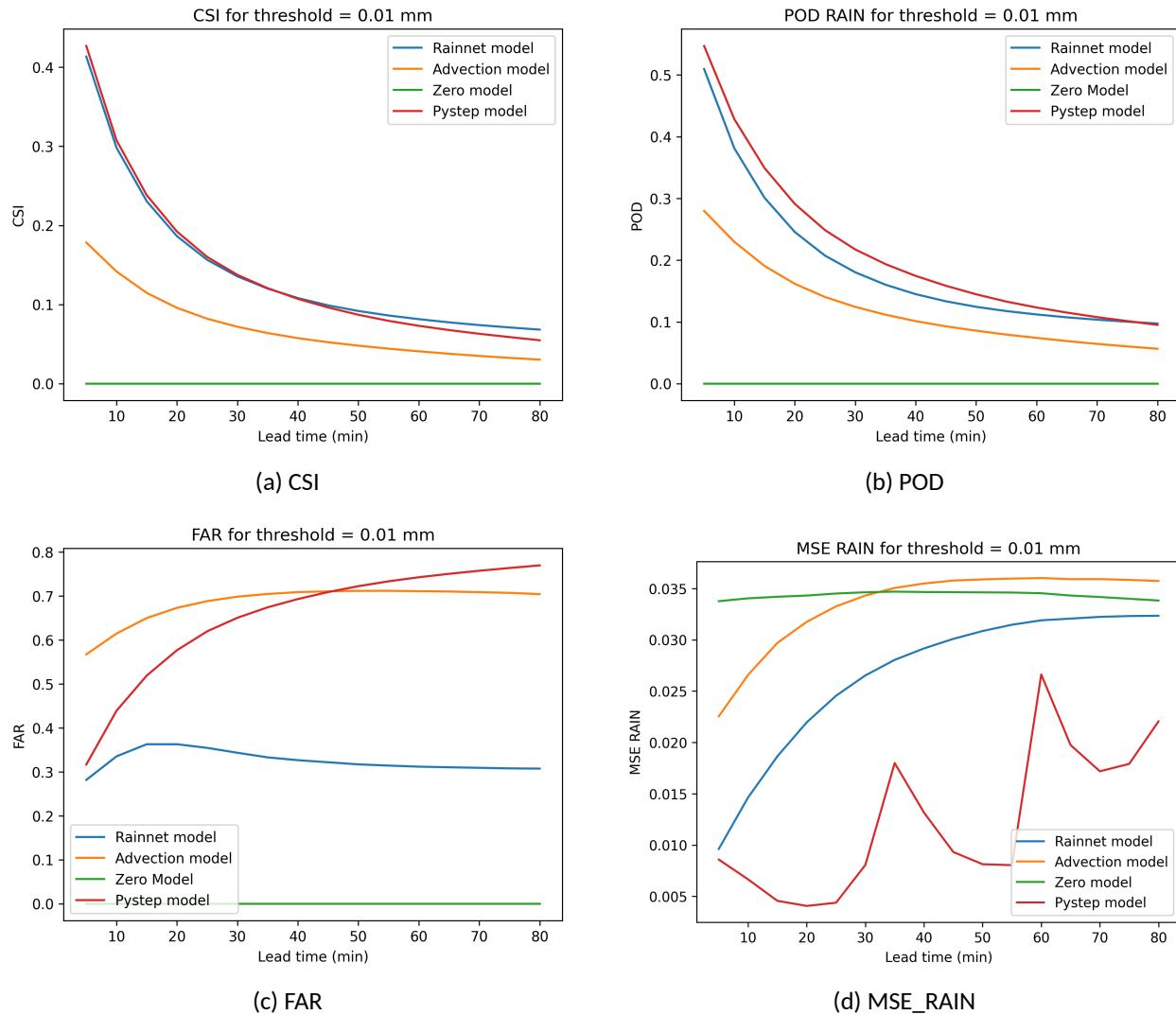


Figure 5.1 – Évolution des métriques CSI, POD, FAR et MSE_RAIN pour le seuil 0.01 mm/h.

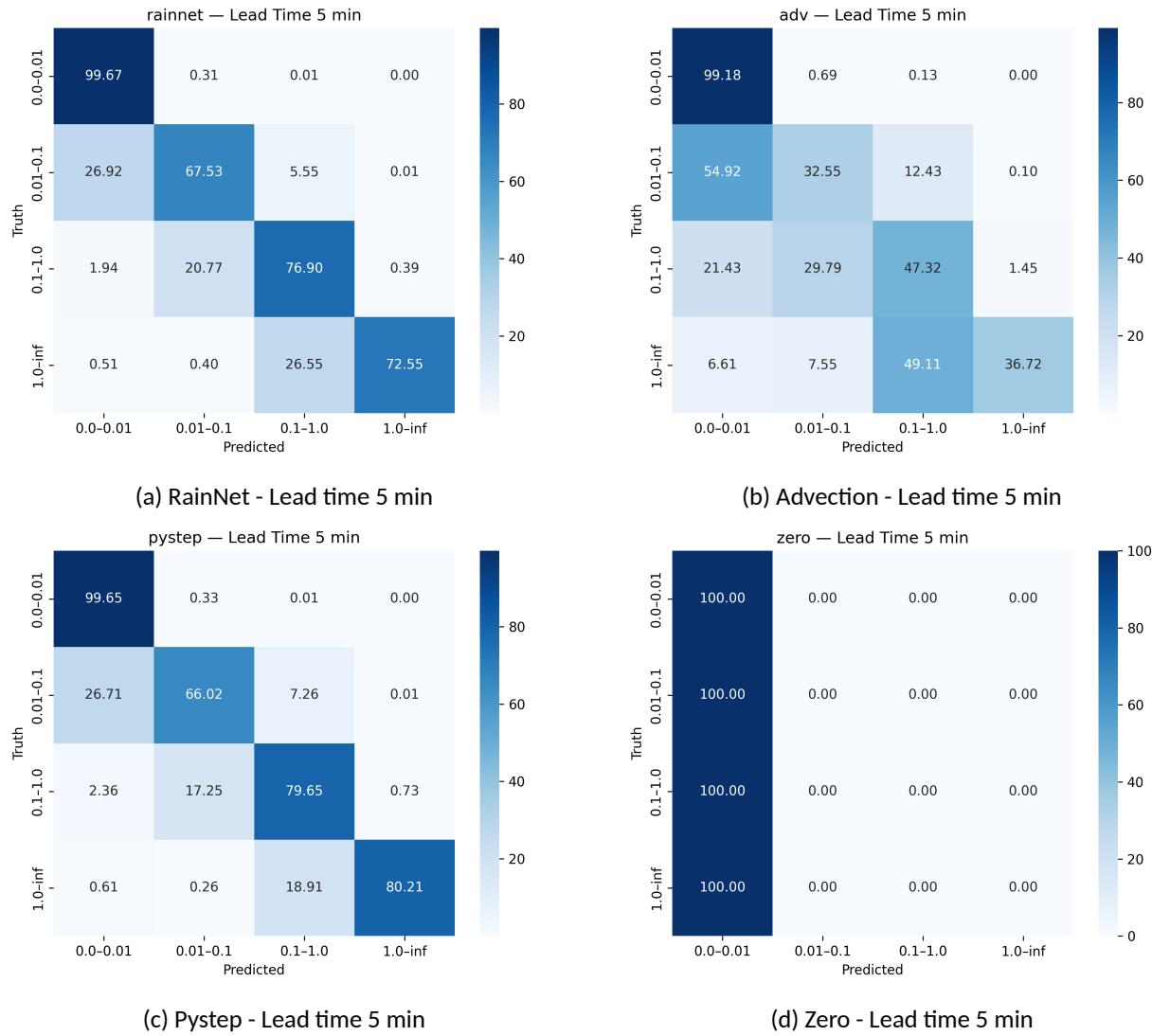


Figure 5.2 – Matrice de confusion des modèles pour un lead time de 5 minutes.

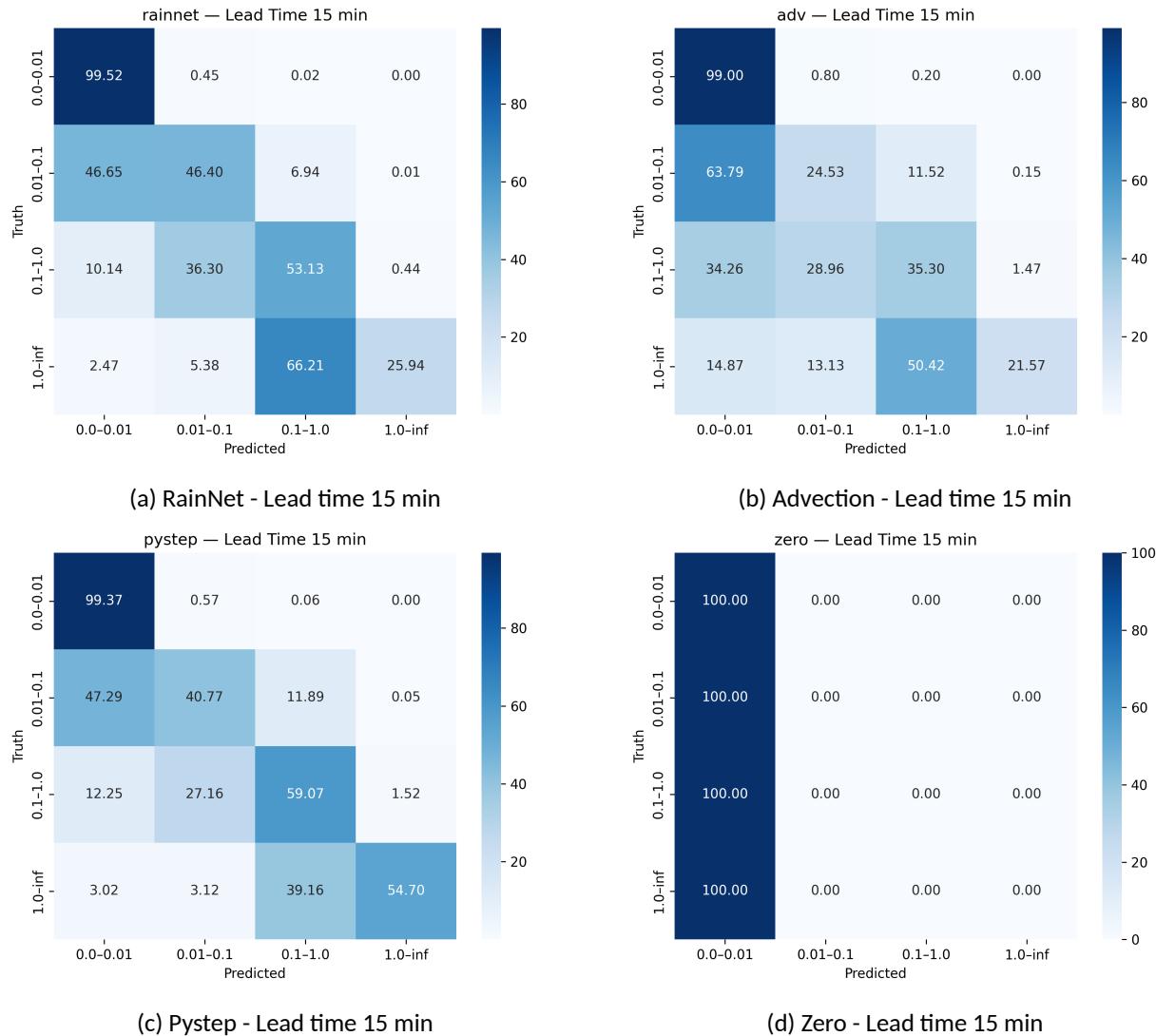


Figure 5.3 – Matrice de confusion des modèles pour un lead time de 15 minutes.

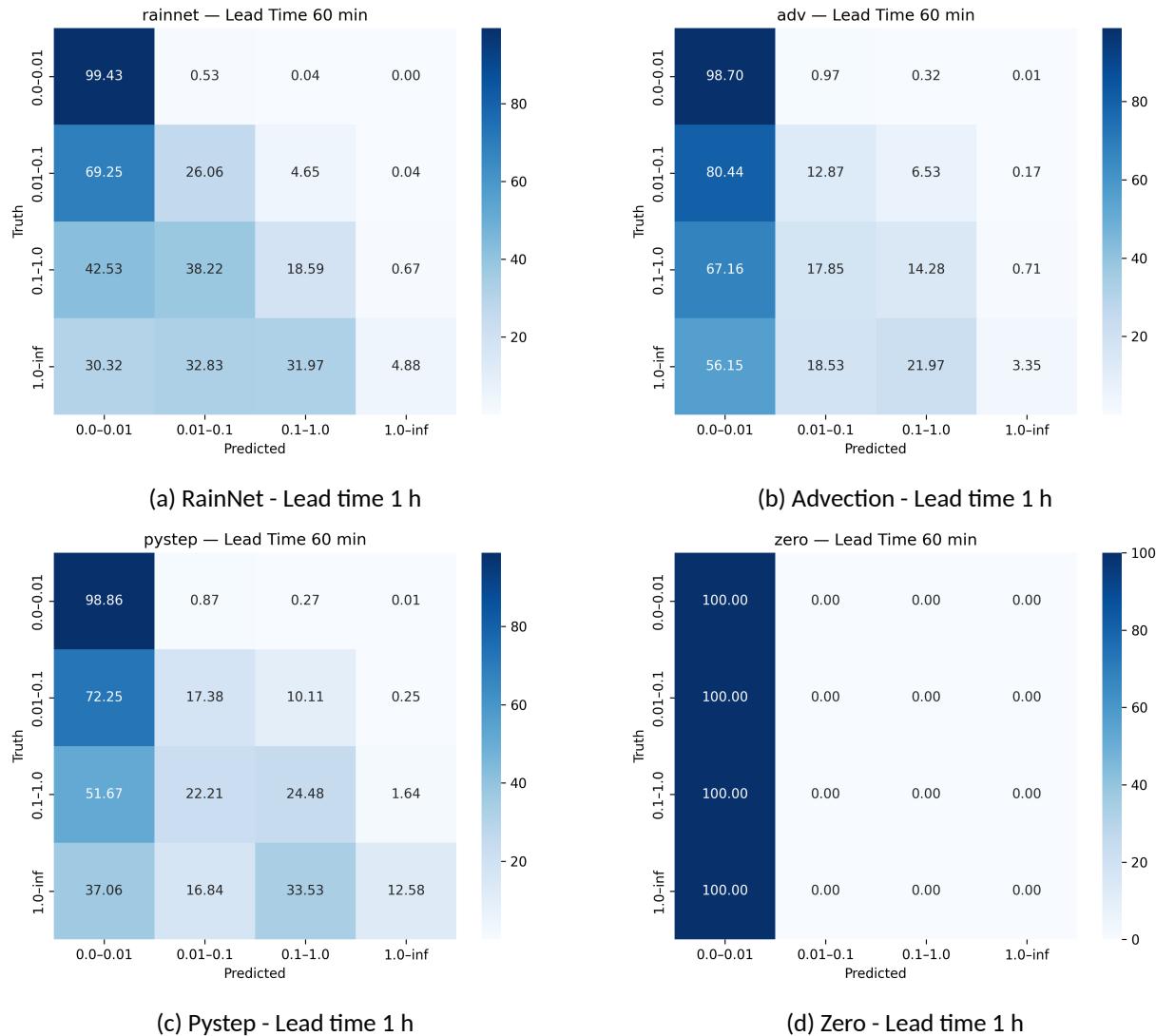


Figure 5.4 – Matrice de confusion des modèles pour un lead time de 1 heure.

Le **CSI** montre nettement une meilleure performance des modèles **RainNet** et **Pysteps** à détecter la pluie tout en limitant les fausses alertes. Le **POD** confirme ce résultat, avec un léger avantage pour **Pysteps**. Toutefois, cet avantage s'accompagne d'une hausse du **FAR**. En effet, **RainNet** présente un taux de fausses alertes nettement plus bas et relativement stable sur l'horizon, tandis que celui de **Pysteps** augmente progressivement sur l'horizon.

Les matrices de confusion montrent également une meilleure capacité des modèles **RainNet** et **Pysteps** à segmenter les différents événements pluvieux. On observe que **RainNet** détecte mieux les faibles pluies, tandis que **Pysteps** est plus performant pour détecter les pluies modérées et fortes.

Tous les modèles détectent correctement l'absence de pluie, mais cela est surtout dû au grand nombre de cas sans pluie dans les ensembles de comparaison. D'ailleurs, c'est la raison pour laquelle l'accuracy n'est pas une métrique très pertinente. En effet, dans un contexte fortement déséquilibré où les situations "pas de pluie" représentent la majorité des observations, un modèle peut obtenir une très bonne **accuracy** (G) simplement en prédisant systématiquement l'absence de pluie, sans pour autant être capable de détecter efficacement les épisodes pluvieux comme le montre le modèle naïf. C'est pourquoi des métriques comme le **CSI**, le **POD** ou encore le **FAR** sont plus adaptées pour évaluer la qualité de la prévision d'événements rares.

Les modèles Pystep et RainNet apportent un gain significatif par rapport au modèle traditionnel d'advection et au prédicteur nul. Le modèle d'advection est trop simpliste. Il faut donc le remplacer par l'un de ces modèles.

L'idée maintenant est d'explorer encore plus les modèles de deep learning et de voir s'ils peuvent apporter plus de précision à nos données. Dans la suite, nous allons finetuner le modèle RainNet sur les données HDRain afin de vérifier s'il est possible d'obtenir de meilleures performances.

6 Finetuning de RainNet

6.1 Finetuning de RainNet

Objectif

Le *finetuning* de RainNet vise à adapter un modèle pré-entraîné à partir de données externes aux données spécifiques HDRain, en tirant parti de sa capacité initiale à capturer des structures spatiales de précipitations tout en affinant ses paramètres pour le contexte local.

Préparation des données

Le jeu de données prétraité (`input_processed.nc`) contient des cartes de précipitations normalisées et interpolées sur une grille de taille 928×928 . À partir de cette base, on identifie toutes les séquences continues de 5 pas de temps espacés de 5 minutes. Chaque séquence est organisée de la manière suivante :

- **Entrée (X)** : 4 images consécutives ($t-15, t-10, t-5, t$);
- **Sortie (Y)** : 1 image cible à $t+5$ minutes.

Ces séquences sont ensuite séparées en :

- **Entraînement** : 90 % des séquences;
- **Validation** : 10 % des séquences.

Générateur

Dans ce travail, le générateur joue un rôle essentiel dans la gestion efficace des données volumineuses issues des fichiers NetCDF. Il permet de charger uniquement en mémoire les séquences nécessaires au batch en cours, en extrayant dynamiquement les 4 images d'entrée et l'image cible à partir des indices temporels valides. Ce chargement à la volée réduit significativement l'empreinte mémoire et évite les goulots d'étranglement liés à un préchargement complet du dataset. Couplé au pipeline `tf.data`, le générateur assure un prétraitement parallèle et un préchargement asynchrone des batches via `prefetch`, maintenant ainsi un débit constant vers le GPU. Cette approche optimise l'utilisation des ressources, réduit les temps d'attente entre les itérations et permet de traiter efficacement des ensembles de données d'envergure sans compromettre la stabilité de l'entraînement.

Configuration du modèle

Le modèle de base (`rainnet_finetuned_.h5`) est chargé avec ses poids initiaux. Les couches situées avant l'index `FREEZE_LAYERS` sont gelées (non entraînables) afin de conserver les représentations spatiales déjà apprises. Les dernières couches restent entraînables pour ajuster la prédiction aux données HDRain.

Entraînement

L'entraînement de RainNet (D) repose sur une procédure supervisée, où le modèle apprend à prédire les précipitations futures à partir de séquences d'images passées. Concrètement, le modèle reçoit en entrée plusieurs cartes radar consécutives (typiquement 4 images cartes de 5 minutes) et doit produire comme sortie la carte de précipitations correspondante à l'horizon temporel suivant (+5 minutes).

L'optimisation est réalisée avec :

- **Optimiseur** : Adam (taux d'apprentissage initial 1×10^{-4});
- **Fonction de perte** : Log-Cosh, adaptée aux erreurs continues et moins sensible aux valeurs extrêmes ;
- **Métrique suivie** : MAE (*Mean Absolute Error*)(E).

Des callbacks sont utilisés pour améliorer la convergence et éviter le sur-apprentissage :

- ModelCheckpoint : sauvegarde du meilleur modèle selon la MAE de validation ;
- ReduceLROnPlateau : réduction automatique du taux d'apprentissage si la performance stagne ;
- EarlyStopping : arrêt anticipé si aucune amélioration n'est observée sur plusieurs époques ;
- CSVLogger : suivi des métriques.

L'entraînement est effectué sur 15 époques avec un *batch size* de 8.

À l'issue de l'entraînement, le modèle présentant la meilleure loss est sauvegardé. Les courbes de perte et de MAE montrent l'évolution de la performance à chaque époque, permettant d'évaluer la convergence et la capacité de généralisation du modèle.

6.2 Résultats

Nous avons réalisé plusieurs tests de *finetuning* :

- Dégel des **6 dernières couches** ;
- Dégel des **12 dernières couches** ;
- Dégel des **26 dernières couches** (tout le **décodeur**) ;
- Dégel de **toutes les couches** (encodeur et décodeur) ;
- Introduction d'une **nouvelle fonction de perte** : *log-cosh* appliquée uniquement sur la **zone non paddée**.

Globalement, les résultats ne sont **pas meilleurs que RainNet**. Nous détaillons ci-dessous les résultats obtenus au cas du dégel des **26 dernières couches**.

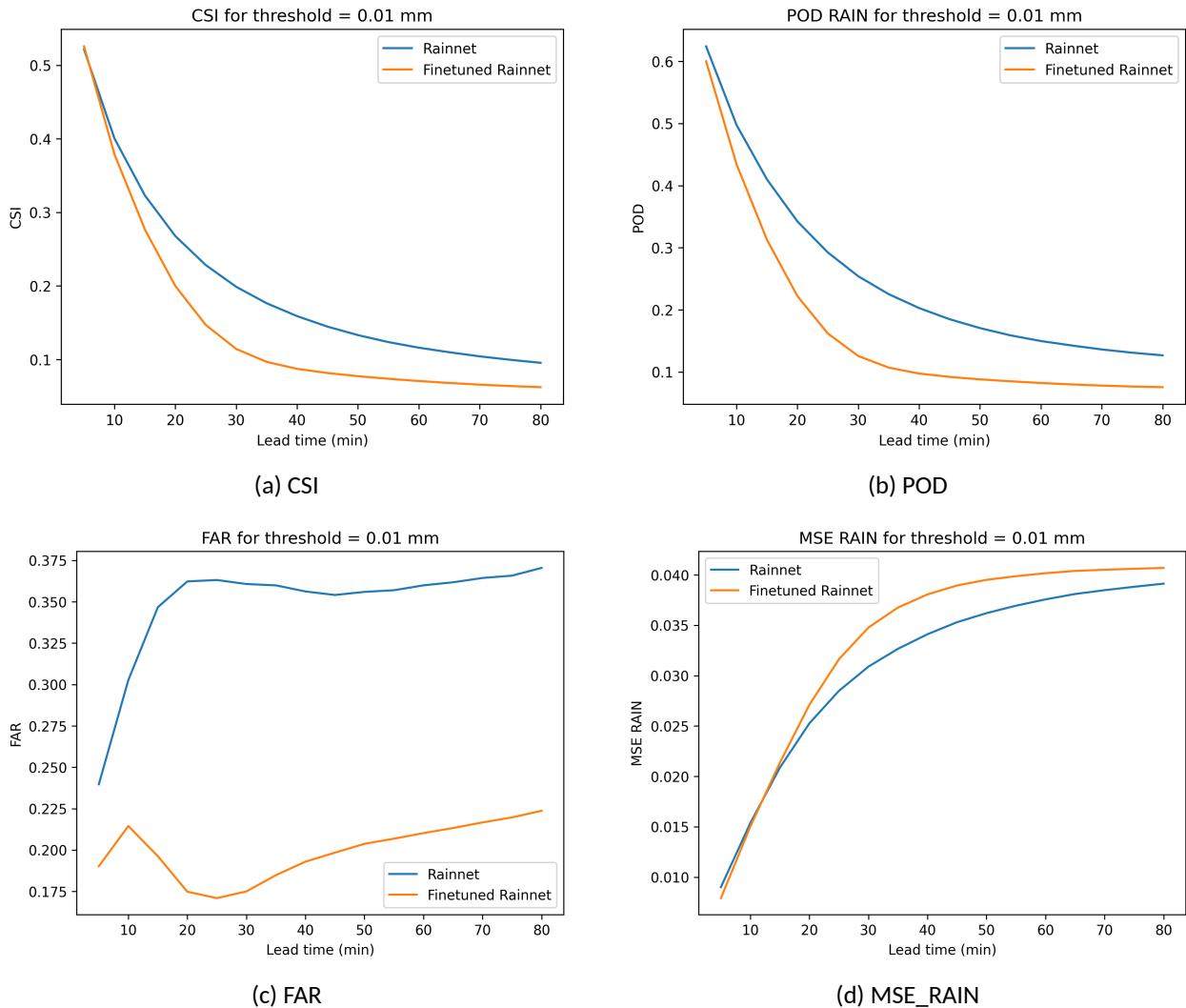


Figure 6.1 – Évolution des métriques CSI, POD, FAR et MSE_RAIN pour le seuil 0.01 mm/h.

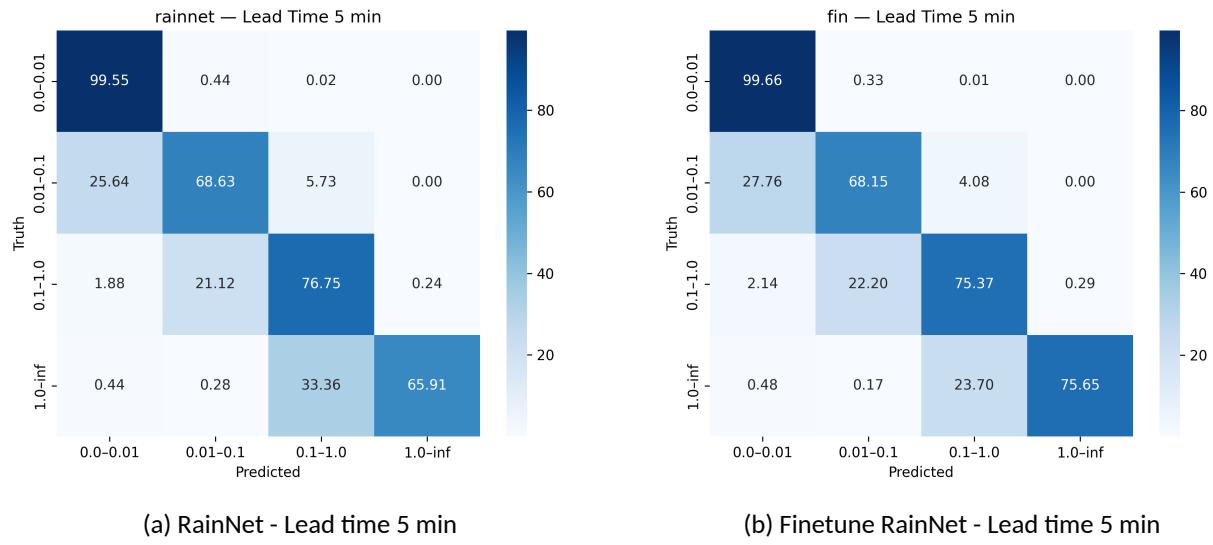


Figure 6.2 – Matrice de confusion pour un lead time de 5 minutes.

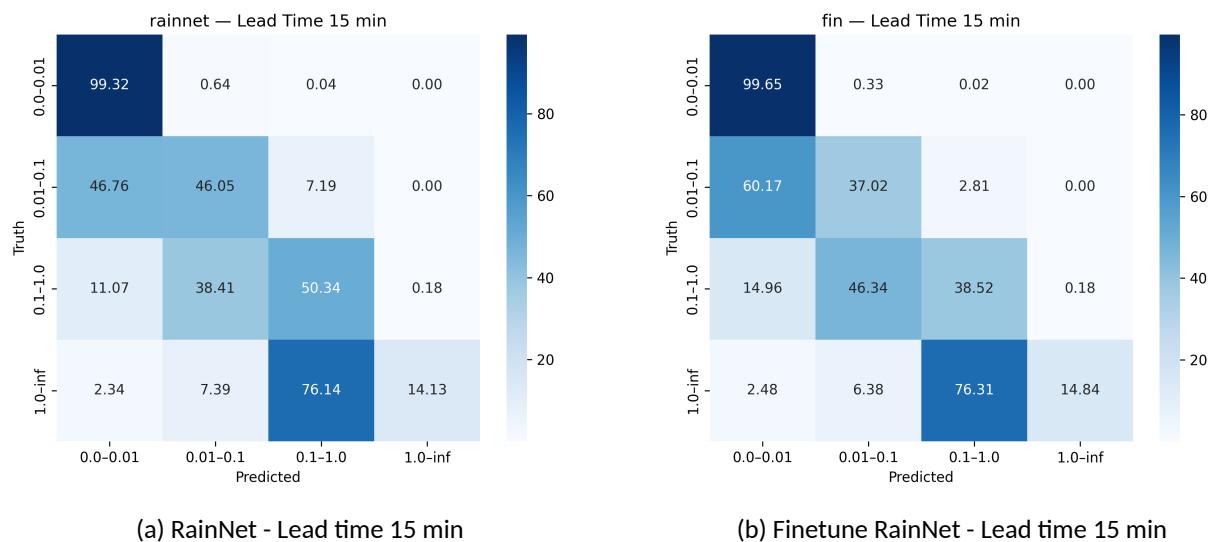


Figure 6.3 – Matrice de confusion pour un lead time de 15 minutes.

Le *finetuning* apporte des gains, mais ceux-ci ne sont pas très significatifs. Sur les horizons proches (5 à 15 minutes), les performances sont similaires, avec un léger avantage pour le modèle finetuné sur certaines métriques, notamment le **FAR**, qui mesure la proportion de fausses alertes et tend à les minimiser. Cependant, globalement, les performances du nouveau modèle se dégradent rapidement pour les horizons plus lointains (≥ 15 minutes).

Plusieurs facteurs expliquent ce résultat :

- Les données radar DWD ayant servi à l'entraînement initial de RainNet sont très différentes des données HDRain (nature physique du signal, distribution des intensités, bruit). Les différences intrinsèques entre les données limitent la transférabilité des représentations apprises par le modèle
- Les données HDRain sont issues d'une interpolation, ce qui réduit la précision spatiale et introduit un lissage des structures fines. Après rééchantillonnage à 1 km, la taille obtenue (355×774) doit être adaptée au format d'entrée de RainNet (928×928) par **padding miroir**, ce qui conduit à ce que $\approx 213.85\%$ des pixels soient artificiellement ajoutés et donc dépourvus d'information réelle.
- Le nombre d'exemples d'apprentissage reste limité par rapport aux millions d'images radar initialement utilisés pour RainNet, ce qui limite la capacité à finetuner efficacement les couches supérieures.

En résumé, le *finetuning* direct sur les données HDRain ne permet pas d'améliorer les performances ; au contraire, il tend à détériorer la prévision sur les horizons lointains.

Pour éviter la perte d'information liée au padding, nous allons conserver la même architecture, mais nous ré-entraînerons le modèle avec les poids initialisés aléatoirement sur des données de shape (256×256).

7 Réentraînement de RainNet

7.1 Procédure

Par rapport au *finetuning* décrit précédemment, la principale différence réside dans l'entraînement de RainNet **à partir de zéro** sur les données HDRain, sans partir d'un modèle pré-entraîné.

Pour augmenter la diversité des exemples, les données ont été **découpées en tuiles de** (256×256) , générant plusieurs sous-images par séquence temporelle. Chaque tuile est extraite via un générateur tuilé qui lit directement les données depuis le fichier NetCDF et prépare les tenseurs d'entrée/sortie.

Pour le reste, organisation des séquences, configuration de l'optimiseur, fonction de perte, métriques et *callbacks*, la configuration reste identique à celle utilisée pour le *finetuning*.

7.2 Résultats

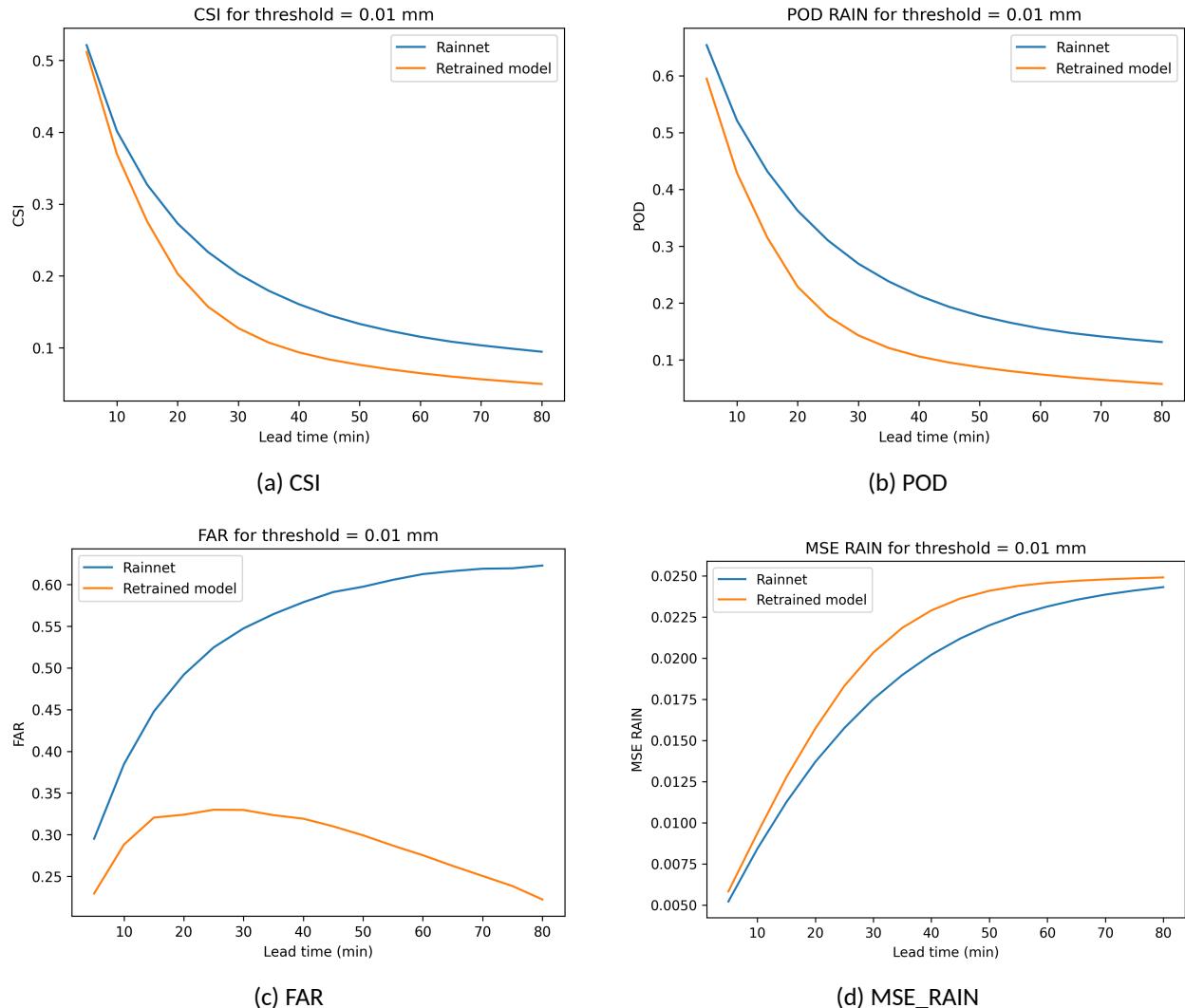


Figure 7.1 – Évolution des métriques CSI, POD, FAR et MSE_RAIN pour le seuil 0.01 mm/h.

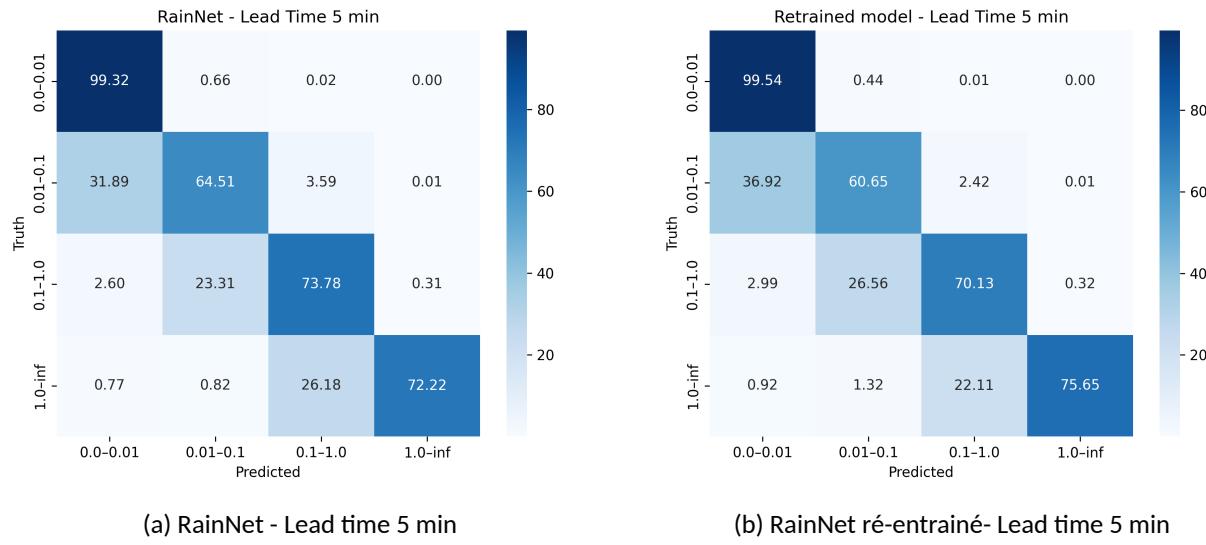


Figure 7.2 – Matrice de confusion pour un lead time de 5 minutes.

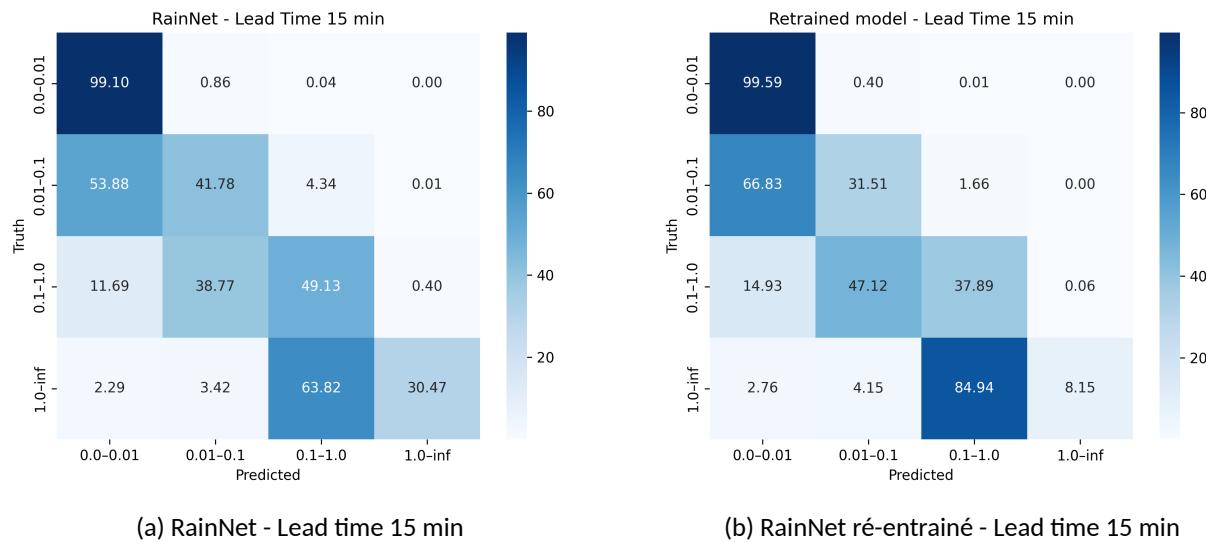


Figure 7.3 – Matrice de confusion pour un lead time de 15 minutes.

Les différentes métriques montrent que le ré-entraînement de RainNet sur la nouvelle architecture n'apporte pas de gain par rapport à RainNet pré-entraîné. Le modèle parvient à apprendre des structures pertinentes, mais sans apporter de gain significatif par rapport au modèle de base. Les matrices de confusion confirment ce constat. Pour les horizons courts, les résultats sont similaires au modèle pré-entraîné. En revanche, pour les horizons plus longs (≥ 1 heure), le modèle ré-entraîné présente une dégradation plus marquée, avec une sous-détection des différentes classes

Ces résultats peuvent s'expliquer par plusieurs facteurs :

- Le volume limité de données HDRain, qui ne permet pas d'atteindre la richesse statistique du jeu de données radar initial de RainNet.
- Le découpage en tuiles de 256×256 , qui facilite l'entraînement mais fragmente les structures spatiales de grande échelle, essentielles pour le nowcasting.

En résumé, le **ré-entraînement complet de RainNet sur les données HDRain n'apporte pas d'amélioration notable par rapport au modèle pré-entraîné**. Il confirme cependant que RainNet reste robuste sur des données différentes, mais que d'autres stratégies (nouvelles architectures, modèles hybrides, ou augmentation de données) seraient nécessaires pour obtenir un véritable gain de performance.

Conclusion

Ce stage m'a permis de travailler pendant 6 mois au cœur d'un projet mêlant météorologie, traitement de données massives et intelligence artificielle, dans un contexte opérationnel exigeant. J'ai pu confronter différentes approches de *nowcasting* à savoir des méthodes advectives traditionnelles, des approches statistiques et des modèles de Deep Learning et évaluer objectivement leurs forces et limites sur les données haute résolution produites par HDRain.

Au travers d'une méthodologie rigoureuse, j'ai pu concevoir un pipeline complet allant de la collecte des données jusqu'à l'évaluation quantitative des modèles.

Cependant, plusieurs difficultés ont marqué ce travail :

- La différence importante entre les données radar DWD utilisées pour l'entraînement initial de RainNet et les données HDRain, tant sur le plan physique que statistique.
- Les contraintes de format imposant un **padding** important, introduisant une proportion élevée de pixels sans information réelle.
- Le volume massif des données, nécessitant une gestion mémoire optimisée et un traitement en flux (*streaming*) pour éviter les saturations.
- Le temps exhaustif de traitement, de stockage des données et d'entraînement des modèles.
- Le déséquilibre entre les situations de pluie et de non-pluie, rendant certaines métriques difficiles à interpréter.

Ces constats et les résultats obtenus expliquent le choix de HDRain d'adopter PySTEPS comme solution opérationnelle, car ce modèle offre un compromis robuste entre performance et faisabilité en production. Toutefois, ce travail ouvre des pistes prometteuses pour l'avenir. Une première voie consiste à développer des méthodes hybrides qui combineront la rigueur des modèles physiques d'advection avec la capacité d'apprentissage des réseaux neuronaux, afin de tirer parti des deux approches. Une deuxième perspective réside dans l'exploration des architectures de type transformers, qui, grâce à leur efficacité dans la modélisation de séquences spatio-temporelles complexes, pourraient mieux capturer la dynamique multi-échelle des précipitations. Enfin, un enjeu central demeure l'amélioration et l'enrichissement des bases de données HDRain : davantage de diversité spatio-temporelle, une meilleure qualité des mesures et une couverture élargie sont indispensables pour libérer pleinement le potentiel de ces méthodes avancées.

Sur le plan technique, ce stage m'a permis de :

- Approfondir ma maîtrise de Python et des bibliothèques spécialisées dans le traitement des données massifs.
- Développer des méthodes d'évaluation robustes adaptées à un contexte de prévision météorologique.
- Mettre en place des solutions efficaces de traitement de données à grande échelle.
- Approfondir mes connaissances et compétences en machine learning et en computer vision.

Sur le plan personnel et professionnel, j'ai appris à m'adapter à des contraintes imprévues, à organiser mon travail pour respecter des objectifs précis, et à collaborer efficacement avec des experts d'horizons variés. Ce stage a renforcé ma capacité à gérer des projets complexes, à prendre du recul sur les choix techniques et à persévérer face aux difficultés.

En définitive, cette expérience a été à la fois formatrice et enrichissante, en m'offrant des compétences solides et une meilleure compréhension des enjeux liés à l'application de l'IA dans des domaines à fort impact sociétal, comme la prévision météorologique.

Bibliographie

- [1] HDRain. (2023, January 30). *Retour sur la technologie HDRain*. HDRain Blog.
<https://www.hd-rain.com/blog/retour-sur-la-technologie-hd-rain>
- [2] Ayzel, G. (2020). *RainNet : a convolutional neural network for radar-based precipitation nowcasting*. GitHub repository. <https://github.com/hydrogo/rainnet>
- [3] Pulkkinen, S., Nerini, D., Pérez Hortal, A. A., Velasco-Forero, C., Seed, A., Germann, U., & Foresti, L. (2019). *Pysteps : an opensource Python library for probabilistic precipitation nowcasting (v1.0)*. Geoscientific Model Development, 12(10), 4185–4219. <https://doi.org/10.5194/gmd-12-4185-2019>
- [4] PYSTEPS developers. (2025). *My first nowcast : a Google Colab tutorial notebook to run your first extrapolation nowcast with pysteps*. GitHub / Google Colab. https://colab.research.google.com/github/pySTEPS/pysteps/blob/master/examples/my_first_nowcast.ipynb

Table des annexes

— Carte HDRAIN	page 36
— Cartes prédites comparées	page 37
— Architecture RainNet	page 38
— Entrainement RainNet	page 39
— Evolution métriques pendant le finetuning	page 40
— Evolution métriques pendant le ré-entraînement	page 41
— Courbes complémentaires : analyse initiale	page 43
— Courbes complémentaires : analyse finetuning	page 42
— Courbes complémentaires : analyse ré-entraînement	page 44

A Carte HDRAIN

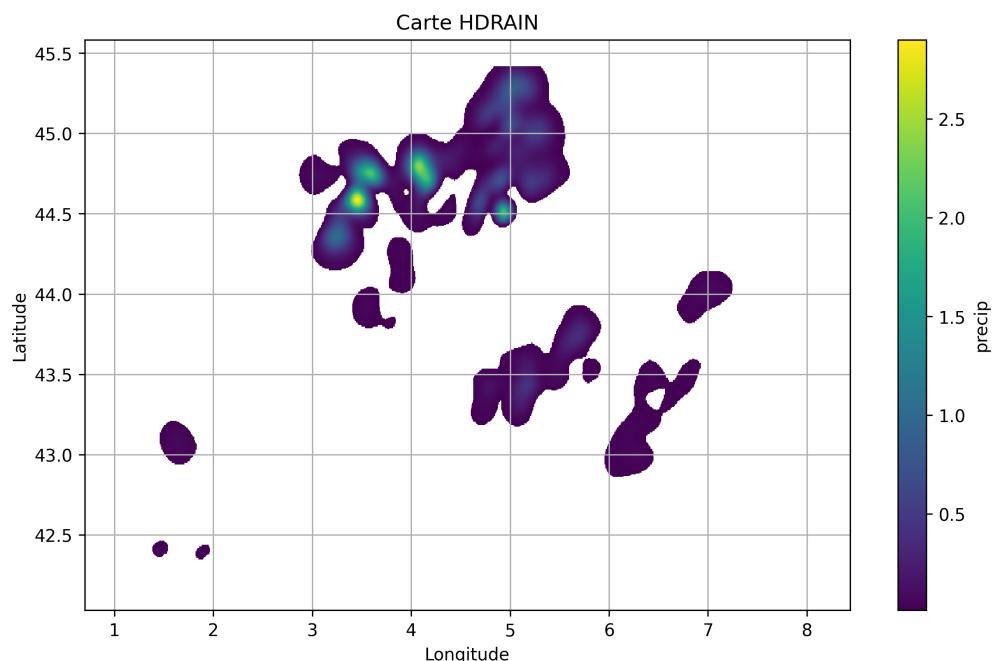


Figure A.1 – CARTE HDRAIN.

B Cartes comparées

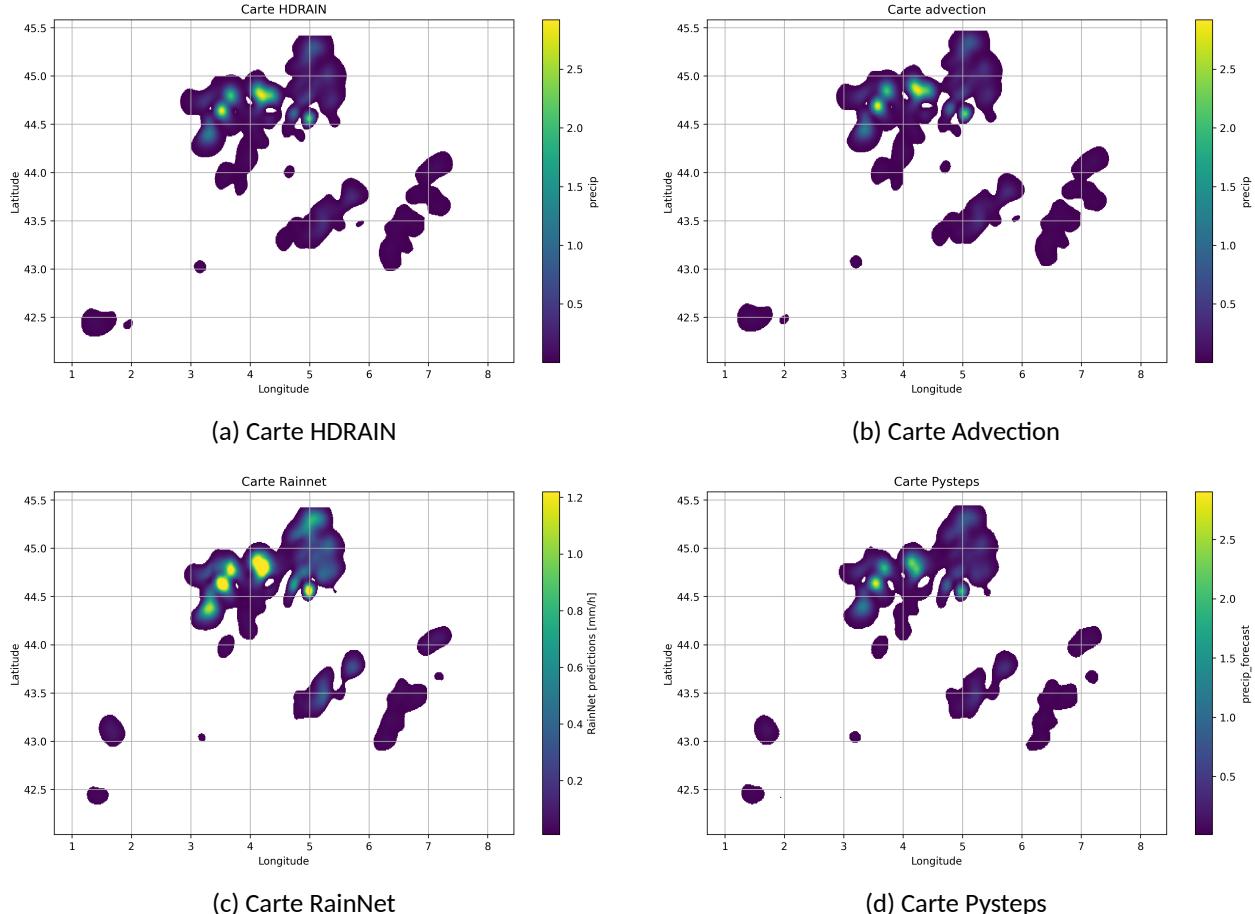


Figure B.1 – Cartes prédites par différents modèles

C Architecture RainNet

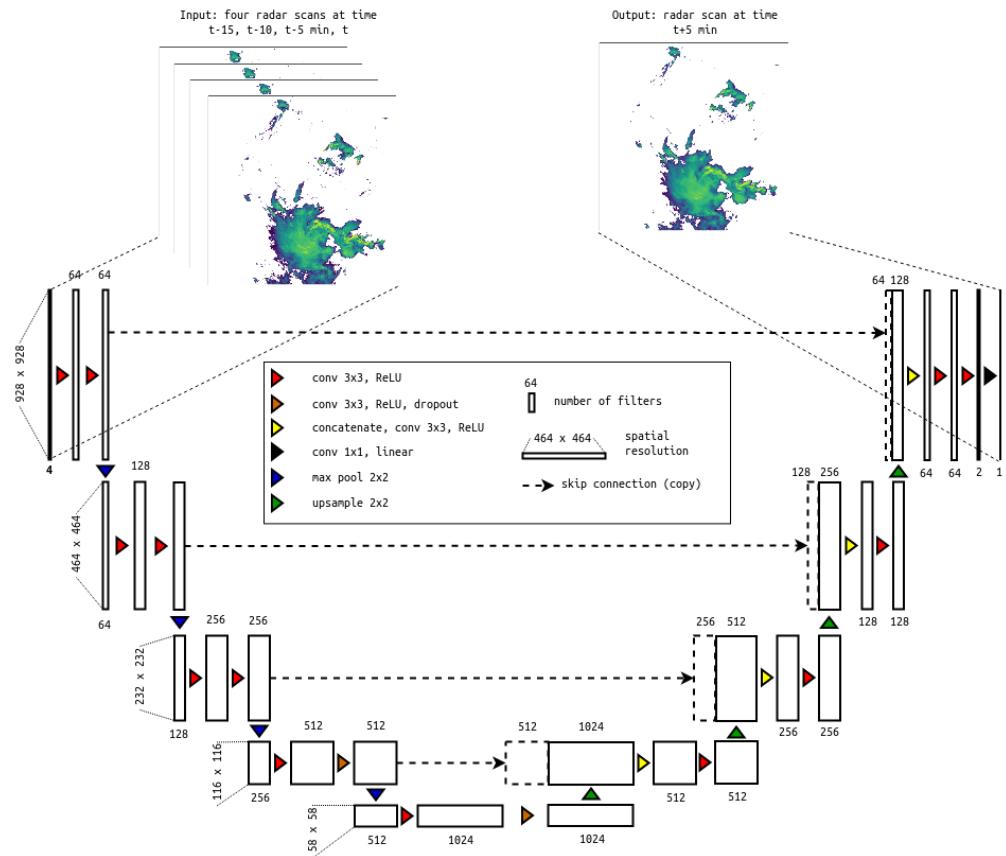


Figure C.1 – Architecture RainNet.

D Entrainement RainNet

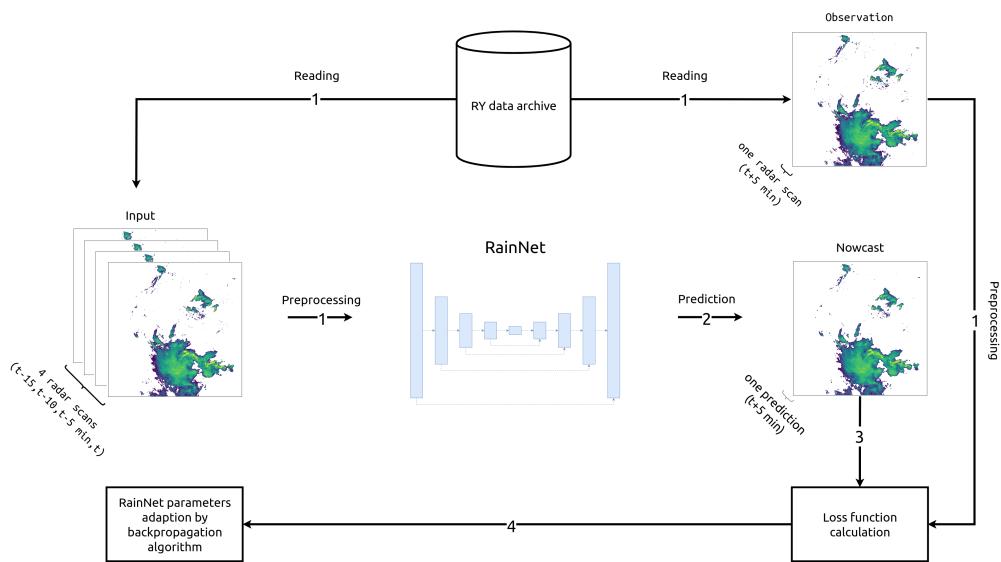


Figure D.1 – Procédure entrainement de RainNet

E Evolution métriques pendant le finetuning

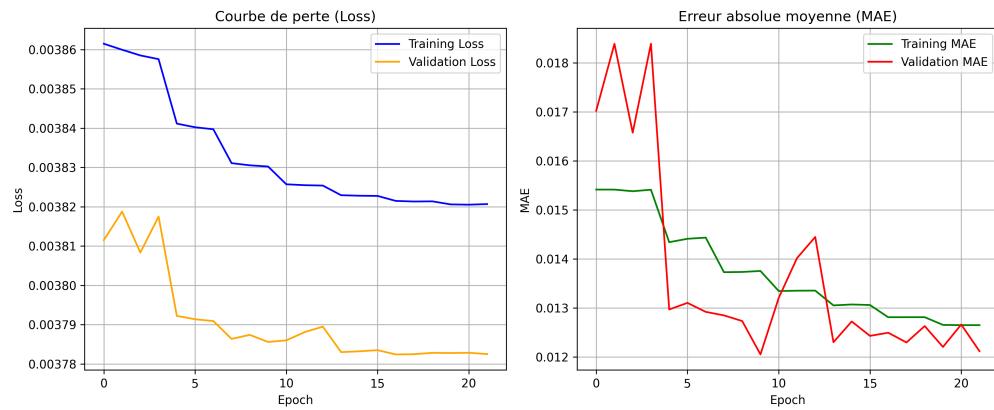


Figure E.1 – Evolution métriques pendant le finetuning.

F Evolution métriques pendant le ré-entraînement

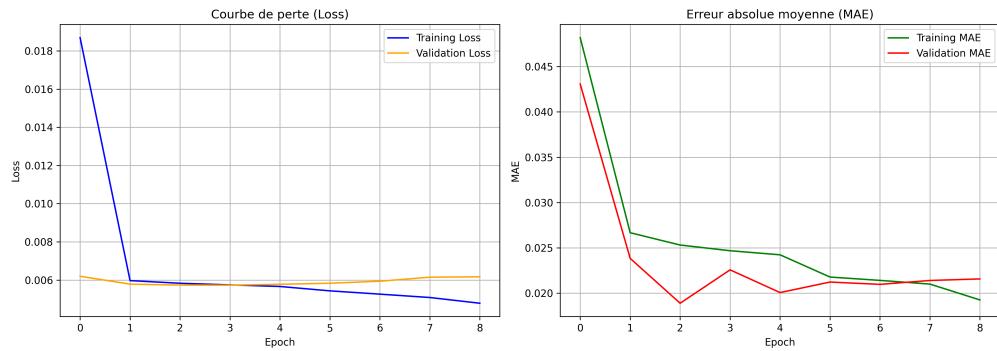


Figure F.1 – Evolution métriques pendant le ré-entraînement.

G Courbes complémentaires : analyse initiale

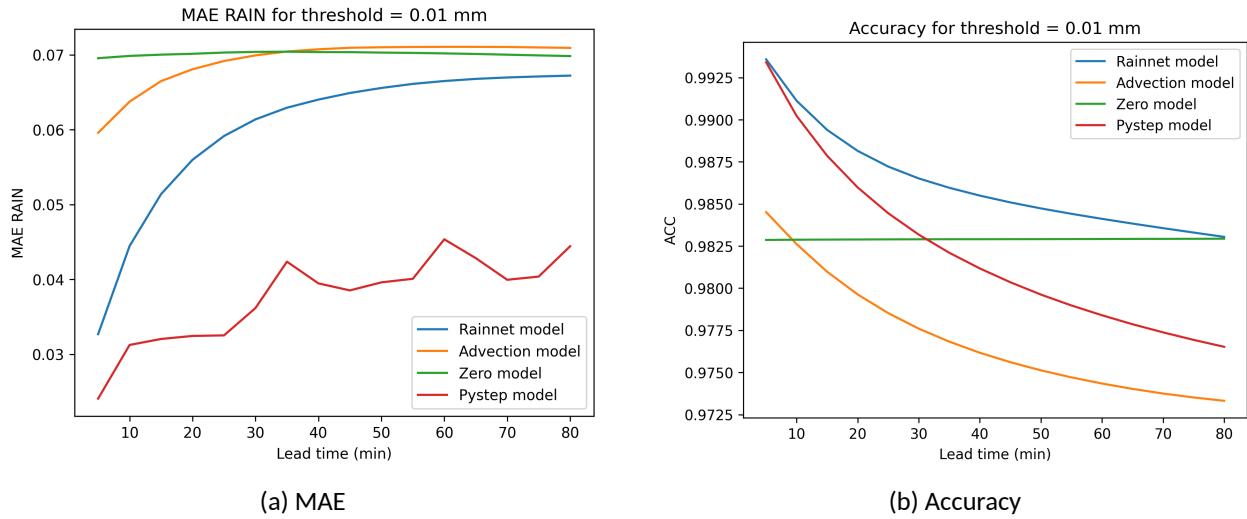


Figure G.1 – Évolution des métriques MAE et Accuracy pour le seuil 0.01 mm/h

H Courbes complémentaires : analyse Finetuning

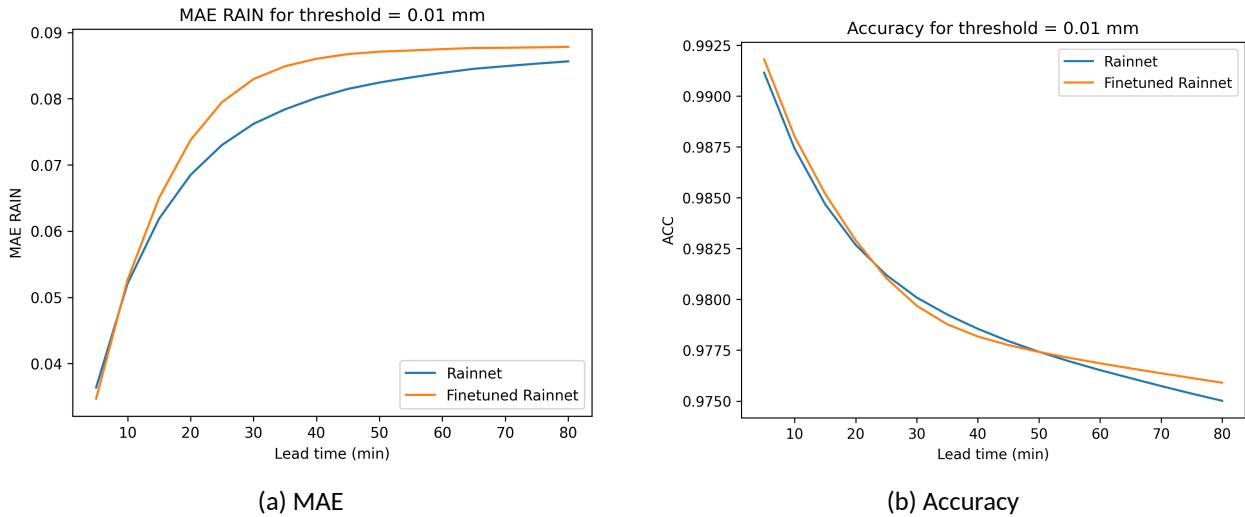


Figure H.1 – Évolution des métriques MAE et Accuracy pour le seuil 0.01 mm/h

I Courbes complémentaires : Analyse Ré-entraînement

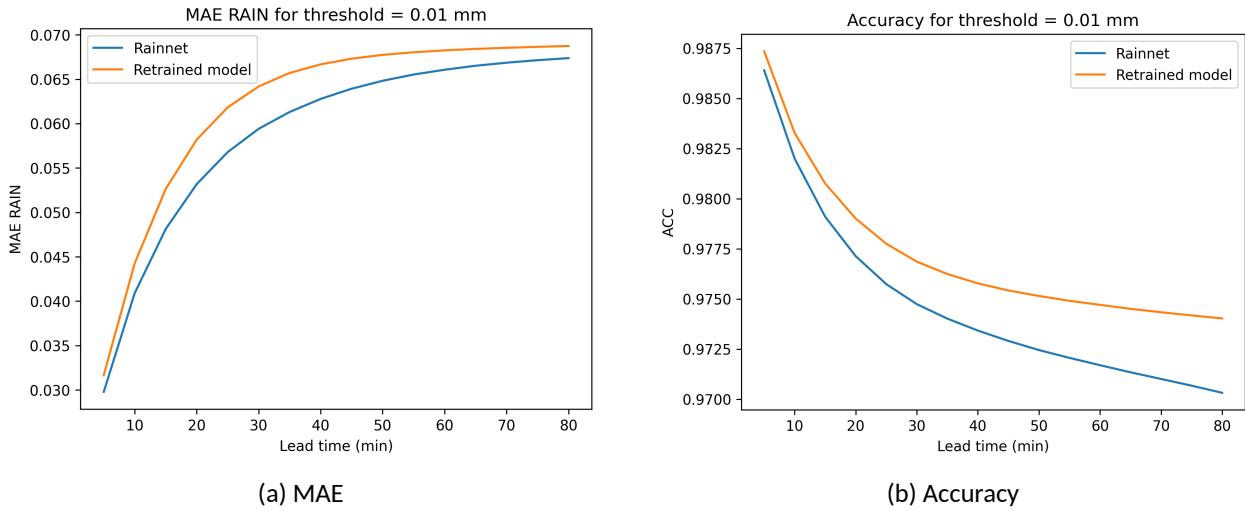


Figure I.1 – Évolution des métriques MAE et Accuracy pour le seuil 0.01 mm/h