Rapport Projet MAM5A

POLYTECH LYON

Prévision des précipitations à court terme

Tuteur : Thierry Clopeau

EL KHAMLICHI Badreddine
EL KHALFIOUI Nadir

Polytech Lyon
MAM 5A
23/02/2024

Table des matières

Table des figures

1 Introduction

1.1 Brève présentation du projet et de ses objectifs

Le but de ce projet est de répondre à un besoin crucial : améliorer les prévisions météorologiques à court terme, avec un accent particulier sur les précipitations. Les prévisions précises sont essentielles pour une vaste gamme d'activités humaines et jouent un rôle clé dans la prévention des catastrophes naturelles. Elles offrent la possibilité de se préparer adéquatement et de réagir de manière efficace face aux variations météorologiques. Dans cette optique, le projet se penche sur deux méthodes computationnelles avancées destinées à prédire les précipitations, visant à déterminer les approches les plus performantes pour des prévisions fiables et précises.

Nous explorerons et comparerons deux approches innovantes pour la prévision météorologique à court terme : l'emploi d'un réseau de neurones convolutifs (CNN) traditionnel et l'introduction d'un CNN Lagrangien (L-CNN). Tandis que le CNN classique se concentre sur l'analyse directe des données spatiales et temporelles pour prédire les précipitations, le L-CNN apporte une dimension nouvelle en intégrant des coordonnées Lagrangiennes. Cette intégration vise à capter avec plus de précision les mouvements et les évolutions des patterns météorologiques. L'objectif est donc double : évaluer la capacité de chaque méthode à affiner la précision des prévisions et examiner leur pertinence pratique au sein des systèmes de prévision météorologique contemporains.

1.2 Description des données et attendus

Notre ensemble de données se compose de 100 000 fichiers au format .npz, chaque fichier représentant une série temporelle de quatre images satellites prises à des intervalles de six minutes. Ces images capturent l'intensité des précipitations sur une zone de 100 km x 100 km, offrant une perspective unique sur la dynamique météorologique à court terme. L'objectif est de prédire les 8 valeurs suivantes de précipitation dans une zone réduite de 2x2 km à partir de ces séquences d'images. Le y_train associé à chaque séquence d'images fournit les valeurs réelles de précipitation, servant de référence pour l'évaluation de nos modèles. La qualité des prévisions sera jugé selon le score obtenu en utilisant la métrique : Mean Squared Logarithmic Error. Voici un aperçu du classemnt actuel, cela nous donne une idée de l'objectif en terme de score qu'on peut se donner :

Rang	Date	Participant(s)	Score public
1	11 juin 2023 14:16	franck.zibi	0,2265
2	7 avril 2023 05:00	WS	0,2557
3	**-**	**benchmark**	**0,2825**
4	6 septembre 2023 15:36	HaPo	0,2825
5	17 août 2023 13:04	thomas.robert & arthur_rozan	0,3521
6	15 septembre 2023 21:01	samson.gourevitch	0,4187
7	3 mars 2023 12:54	tchaye59	0,5425

FIGURE 1 – Classement Challenge

1.3 Aperçu des méthodes : CNN simple et L-CNN

Dans le cadre de notre projet, nous explorons deux approches principales pour la prévision météorologique à court terme, à savoir le CNN (Convolutional Neural Network) simple et le L-CNN (Lagrangian Convolutional Neural Network), chacune employant des méthodologies distinctes pour traiter et analyser les données de précipitation.

CNN Simple : Cette méthode utilise des architectures de réseaux de neurones convolutifs pour analyser les données d'entrée, qui sont structurées en grille, comme les images satellites des précipitations. En exploitant des opérations de convolution, le CNN simple est capable d'extraire des caractéristiques hiérarchiques et pertinentes des données, rendant cette approche particulièrement efficace pour la reconnaissance de motifs dans les précipitations. Pour une exploration complète, nous avons adopté à la fois des architectures 2D, pour une analyse spatiale, et 3D, pour incorporer l'aspect temporel et saisir l'évolution des précipitations dans le temps.

L-CNN : En étendant les capacités du CNN classique, le L-CNN intègre des coordonnées Lagrangiennes pour suivre le mouvement et l'évolution des caractéristiques à travers le temps. Cette approche permet une capture plus précise de la dynamique temporelle des précipitations, offrant une perspective enrichie pour la prévision. Le L-CNN s'appuie sur deux sous-méthodes : l'une utilisant des flux de vitesse dérivés des matrices, et l'autre exploitant le flux optique pour cartographier le mouvement et les changements dans les données de précipitations au fil du temps.

En résumé, notre exploration des prévisions météorologiques à court terme s'appuie sur ces deux méthodes innovantes, avec pour chacune l'adoption de sous-approches spécifiques visant à optimiser la précision des prédictions. Le CNN, avec ses variations 2D et 3D, offre une analyse profonde des caractéristiques spatiales et temporelles, tandis que le L-CNN, par ses techniques basées sur les coordonnées Lagrangiennes, apporte une compréhension dynamique des mouvements dans les précipitations. Cette combinaison d'approches vise à établir une méthodologie robuste et efficace pour la prévision des précipitations à court terme.

2 Fondements Théoriques

2.1 Concepts clés des CNN

Les CNN utilisent des filtres ou noyaux pour effectuer des opérations de convolution sur les données d'entrée, identifiant ainsi des motifs ou caractéristiques spécifiques à différentes échelles et profondeurs de l'image. Cette capacité à apprendre automatiquement des caractéristiques pertinentes à partir des données, sans nécessiter une ingénierie manuelle des caractéristiques, est au cœur de la puissance des CNN.

Les couches de pooling, souvent intercalées entre les couches de convolution, servent à réduire la dimensionnalité des cartes de caractéristiques tout en préservant les informations essentielles. Cela permet non seulement de diminuer la complexité computationnelle du réseau mais aussi d'accroître sa capacité à généraliser à partir des données d'entrée. En fin de compte, après plusieurs couches de convolution et de pooling, les couches entièrement connectées utilisent ces caractéristiques extraites pour effectuer des tâches spécifiques, comme la classification.

Cependant dans le contexte de notre projet, l'objectif principal n'est pas la classification, mais la régression. Cela signifie que, plutôt que d'attribuer une étiquette de classe à une image d'entrée, notre modèle vise à prédire des valeurs continues, en l'occurrence, les niveaux de précipitations futures. Cet aspect de régression introduit des défis supplémentaires, car il nécessite une précision dans la prédiction des valeurs, ainsi qu'une capacité à interpréter des séquences temporelles et spatiales complexes inhérentes aux données météorologiques.

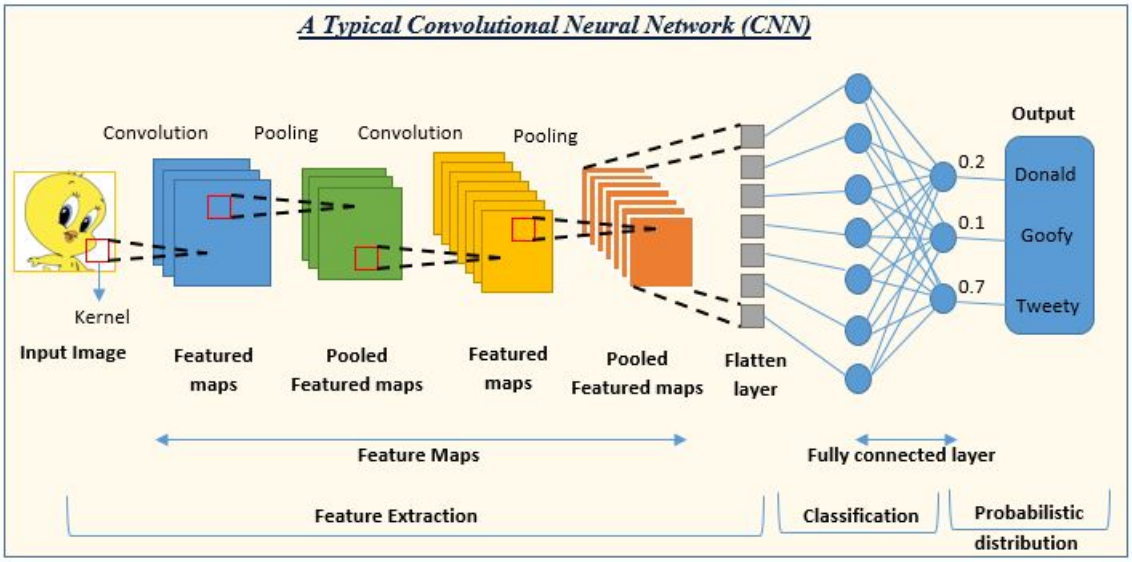

FIGURE 2 – Exemple de classification par CNN

Pour approfondir les fondements théoriques des CNN, examinons les éléments clés qui composent leur architecture :

— **Couches de Convolution :** Elles sont le cœur des CNN, appliquant des filtres aux données d'entrée pour extraire des caractéristiques. Chaque filtre détecte des motifs spécifiques, et la convolution permet de conserver la relation spatiale entre les pixels.

— **Fonctions d'Activation :** Les plus courantes dans les CNN sont les fonctions ReLU (Rectified Linear Unit) pour leur efficacité dans l'accélération de la convergence du réseau en introduisant de la non-linéarité, permettant au modèle d'apprendre des fonctions complexes.

— **Couches de Pooling :** Elles réduisent la dimensionnalité des cartes de caractéristiques en agrégeant les informations, typiquement par le max pooling ou l'average pooling, ce qui aide à rendre le modèle plus robuste aux variations mineures dans les données d'entrée.

— **Couches Entièrement Connectées :** Après plusieurs couches de convolution et de pooling, ces couches utilisent les caractéristiques extraites pour effectuer la tâche finale, telle que la classification.

— **Optimiseurs :** Ils ajustent les poids du réseau pour minimiser une fonction de perte. Les choix populaires incluent SGD, Adam, et RMSprop, chacun avec ses propres avantages pour certaines types de données et architectures.

— **Fonction de Perte :** Elle mesure l'écart entre les prédictions du modèle et les valeurs réelles, guidant l'optimisation. Pour la classification, la cross-entropy est fréquemment utilisée.

Chaque élément joue un rôle crucial dans la conception d'un CNN, influençant sa capacité à apprendre à partir des données d'entrée et à effectuer des tâches spécifiques avec une haute précision.

Avec cela dans le contexte de la prévision météorologique, nous explorons d'abord l'approche traditionnelle en 2D. Les CNN 2D, par leur capacité à analyser les images au niveau spatial, se révèlent puissants pour détecter des motifs tels que les formes et les textures dans des images individuelles. Cependant, cette approche est limitée puisqu'elle ne tient pas compte de l'évolution de ces caractéristiques dans le temps.

L'introduction des CNN 3D marque une avancée significative en incorporant la dimension temporelle. Les filtres convolutifs 3D se déplacent non seulement à travers la largeur et la hauteur mais aussi à travers le temps, capturant ainsi la dynamique des séquences d'images. Cela permet une compréhension plus riche des données, essentielle pour des applications telles que la prévision des précipitations où les changements temporels des motifs météorologiques sont cruciaux. En exploitant la dimension temporelle, les CNN 3D offrent une perspective plus complète, améliorant potentiellement la précision des prédictions en reconnaissant les schémas évolutifs dans les données météorologiques.

On voit donc que les CNN nous offre une large panoplie d'outils et d'éléments ainsi que des méthodes et perspectives qui peuvent être extrêmements intéressantes pour la prévisions météorologiques, d'autant plus qu'ici ayant que les données des précipitations (et l'aspect temporelle si on réfléchit en "3D") il est compliqué d'appliquer des méthodes dite déterministe par manques d'autres variables caractéristi-

ques/explicatives. Ici donc avec un simple CNN 2D, on peut donc pouvoir faire cela et avec un 3D pour comparer également le mouvement des images dans le temps. Ayant énormément de données, le CNN est un outil qui pourra être très performant et intéressant dans notre cas de figure .

2.2 Introduction aux coordonnées Lagrangiennes et leur application en CNN (L-CNN)

En physique, la description du mouvement d'une particule fluide peut se faire en général de 2 manières :

— Approche Lagrangienne
— Approche Eulérienne

La première consiste en un observateur situé sur la particule qui décrit sa trajectoire au cours du temps. On s'intéresse ainsi à leur position, vitesse etc. C'est une manière très répandue de voir les choses, notamment en mécanique ou en dynamique des fluides où les calculs sont ainsi simplifiés.
L'autre approche consiste en un observateur fixe qui observe la particule à travers des points spécifiques dans l'espace par lesquels la particule passe. Au lieu de suivre des particules individuelles, on examine les propriétés du fluide (comme la vitesse, la pression, et la densité) en points fixes dans l'espace au fil du temps. Cette méthode est très utilisée en mécanique des fluides pour étudier le comportement des fluides en mouvement et les champs de flux. Des équations de continuité, de Navier-Stokes, et d'autres lois de conservation sont employées pour décrire la dynamique des fluides. Ces équations permettent de comprendre comment les propriétés du fluide changent à différents endroits dans le domaine étudié, sans nécessiter le suivi de particules individuelles.

Pour le CNN, cela se traduirait par l'implémentation d'une équation telle que l'équation d'advection :

$$\frac{\partial \psi}{\partial t} + \nu \cdot \nabla \psi = S$$

Ici, le but est donc de faire apparaître un champ de vitesse des précipitations pour analyser leur déplacement et ainsi fournir plus d'information pour le CNN.

A travers le projet, nous avons exploité 2 manières d'introduire la vitesse de déplacement des précipitations : grâce à une différence matricielle ou un flux optique. La différence matricielle permet de capter la différence de position des pixels. Pour rappel, chaque pixel désigne l'intensité de la précipitation au niveau de son emplacement. Faire une différence matricielle permet ainsi de déduire le mouvement des précipitations et donc donner au CNN cette composante supplémentaire dans son travail de recherche de caractéristique.

Pour le projet, nous nous sommes également intéressés à une méthode implémentant le flux optique, et plus particulièrement la méthode de Lucas-Kanade. Cette dernière, basée sur des hypothèses indiquant que le flot est essentiellement constant dans un voisinage local du pixel considéré d'une image, permet donc de capturer la vitesse d'un objet dans l'image.

Afin de calculer le flux optique, nous devons minimiser la fonction d'appariement quadratique (minimiser la somme des différences de 2 blocs matriciels au carré) à savoir :

$$A^t_{(x,y)}(\delta x, \delta y) = \sum_{(x,y)\in B} (I(x,y,t) - I(x+\delta x, y+\delta y, t))^2$$

avec I l'intensité du pixel aux coordonnées entrées en paramètres. La fonction de Lucas-Kanade peut ainsi se résumer :

$$\arg\min_{(x,y)\in K} A(\delta x, \delta y) = \arg\min_{(x,y)\in K} \sum_{(x,y)\in B} (I(x,y,t) - I(x+x, y+y, t))^2$$

Étant donné le caractère petit du déplacement (x,y), on peut effectuer un développement de Taylor tel que :

$$I(x+\delta x, y+\delta y, t+1) \approx I(x,y,t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}$$

Nous pouvons ainsi réécrire la fonction d'appariement :

$$A(\delta x, \delta y) = \sum_{(x,y)\in B} (\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t})^2$$

Une fois que la fonction d'appariement a été trouvée, nous résolvons quelques équations d'Euler-Lagrange afin de minimiser A :

$$\begin{cases} 2\sum_{(x,y)\in B}(\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t})\frac{\partial I}{\partial x} = 0 \\ 2\sum_{(x,y)\in B}(\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t})\frac{\partial I}{\partial y} = 0 \end{cases} \quad (1)$$

qu'on résume en une résolution d'un système linéaire $H.\nu = b$ avec :

$$H = \begin{pmatrix} \sum_{(x,y)\in B}(\frac{\partial I}{\partial x})^2 & \sum_{(x,y)\in B}\frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \sum_{(x,y)\in B}\frac{\partial I}{\partial x}\frac{\partial I}{\partial y} & \sum_{(x,y)\in B}(\frac{\partial I}{\partial y})^2 \end{pmatrix}$$

$$b = \begin{pmatrix} -\sum_{(x,y)\in B}\frac{\partial I}{\partial x}\frac{\partial I}{\partial t} \\ -\sum_{(x,y)\in B}\frac{\partial I}{\partial y}\frac{\partial I}{\partial t} \end{pmatrix}$$

avec ν la vitesse du champ de précipitation :

$$\begin{cases} \nu_x^{(k)} = \frac{I}{H_1^1}(b - H_2^1 \cdot \nu_y^{(k-1)}), \\ \nu_y^{(k)} = \frac{I}{H_2^2}(b - H_1^2 \cdot \nu_x^{(k)}) \end{cases} \quad (2)$$

qui se résume en une méthode itérative de type Gauss-Seidel.

Bien que la méthode de Lucas-Kanade soit puissante et relativement simple à implémenter, elle a ses limitations. Elle peut avoir du mal avec les grands mouvements, les rotations ou les changements d'échelle, car ces situations violent ses hypothèses de base. De plus, elle est sensible aux variations d'éclairage et aux occlusions. Pour surmonter certaines de ces limitations, des extensions et des variations de la méthode de Lucas-Kanade ont été développées, telles que l'utilisation de pyramides d'images pour gérer les mouvements à différentes échelles.

Une fois cette étape du calcul du flux optique terminée, nous pouvons l'implémenter dans notre jeu d'entraînement afin d'apporter de l'information supplémentaire concernant la vitesse.

Gardons en tête que nous calculons le flux optique à partir des données brutes en entrée. Ces mêmes données vont être ajoutées aux images afin de combiner l'information de position et de vitesse de chaque précipitation afin de les passer au log. Cependant, nous trouvons des valeurs infinies dans le flux optique ainsi que des valeurs négatives. Pour ne pas perdre d'information, nous transformons donc ces données exprimées en coordonnées cartésiennes en coordonnées polaires. Nous récupérons ainsi 2 composantes pour exprimer la vitesse : la magnitude (exprimant la norme du vecteur vitesse) et l'angle (dans laquelle la vitesse se propage). En excluant la donnée des angles du log, nous pouvons donc réduire nos données et les passer à l'entraînement en ne perdant aucune information. Nous verrons un peu plus tard comment les valeurs infinies ont été traitées.

3 Implémentation et Expérimentation

3.1 Transformation et visualisation de nos données

Avant d'aborder les modèles utilisés dans notre étude, il est essentiel de comprendre la nature, la composition et les spécificités des données à notre disposition, ainsi que les défis associés à leur traitement.

Notre ensemble de données provient de PlumeLabs, et se compose principalement de deux parties : un dossier '*X_train*' et un fichier '*y_train.csv*'. Le dossier 'X_train' contient 100 000 fichiers au format '.npz'. Chacun de ces fichiers encapsule 4 matrices de dimensions 128x128, représentant des images de précipitations capturées à des intervalles de 6 minutes sur une étendue de 100 km par 100 km. Ces images satellites fournissent une représentation spatiale et temporelle des précipitations, offrant une fenêtre unique sur la dynamique météorologique de la zone étudiée. Le fichier '*y_train.csv*', quant à lui, associe à chaque ensemble de 4 images une série de 8 valeurs cibles (on a donc 800000 valeurs), représentant les prévisions de précipitations pour un point spécifique de 2km par 2km au sein de la zone plus large, sur les intervalles de temps suivants.

L'objectif de notre analyse est donc de construire un modèle capable de prédire ces 8 valeurs de précipitations futures, à partir des 4 images satellites fournies, illustrant ainsi l'application de techniques de Deep Learning à la prévision météorologique à court terme. Cette tâche soulève plusieurs défis, notamment la gestion de la grande dimensionnalité des données, la nécessité de capter à la fois les caractéristiques spatiales et temporelles des précipitations, et l'importance de prévoir avec précision des événements météorologiques potentiellement impactants.

3.1.1 Visualisation des données

Chaque fichier '.npz' contient un ensemble de données de taille '(4, 128, 128)', représentant quatre images de précipitations espacées de six minutes chacune. Ces matrices fournissent une visualisation directe des précipitations sur une zone spécifique, offrant ainsi une fenêtre précieuse sur les conditions météorologiques à des moments successifs. Pour une exploration plus intuitive de ces données, nous avons mis en place une fonction Python dédiée à la visualisation graphique de ces séquences d'images. Cette approche nous permet non seulement d'inspecter visuellement la distribution spatiale des précipitations mais aussi de suivre leur évolution dans le temps sur une zone donnée. On voit donc nos 4 images sur 4 temps d'un des fichiers sur une zone spécifique.

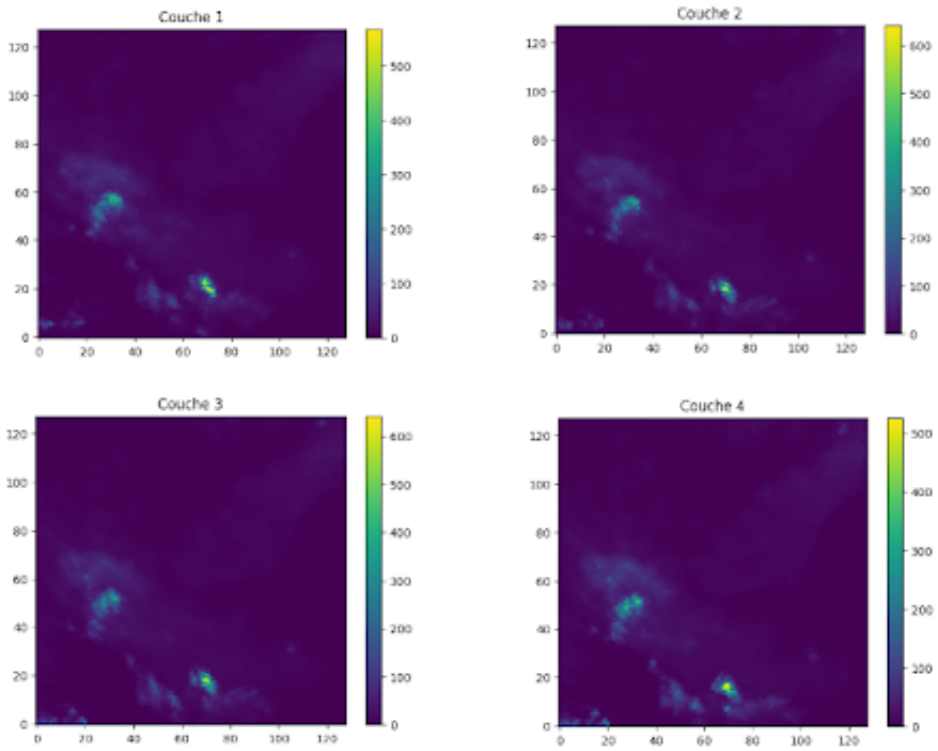

FIGURE 3 – Visualisation des 4 images satéllites d'un fichier

Une caractéristique notable de notre ensemble de données est la prévalence élevée de valeurs nulles, reflétant des périodes sans précipitations significatives. Après avoir analysé un large éventail d'échantillons, nous constatons que, en moyenne, environ 80% des données enregistrées sont des valeurs nulles. Cette observation souligne l'importance de considérer la sparsité des événements pluviométriques dans nos analyses et modélisations.

Pour illustrer quantitativement cette caractéristique, examinons les statistiques descriptives pour un sous-ensemble de 10 000 fichiers :

	max	min	mean	% de 0
X_train_10k	3414.0	0.0	4.296753	79.831268
Y_train_10k	864.0	0.0	5.570487	76.307500

FIGURE 4 – Statistiques sur un échantillon de 10000 fichiers

Ces chiffres mettent en lumière la dynamique et la variabilité des précipitations capturées par nos données. La distribution largement dominée par des valeurs nulles, avec des pics occasionnels de forte précipitation, pose un défi intéressant pour la modélisation prédictive et teste la capacité de nos modèles CNN à distinguer efficacement les patterns significatifs dans un bruit de fond majoritairement inactif.

Dans les sections suivantes, nous aborderons la manière dont ces données sont préparées et transformées pour l'entraînement des modèles, mettant en évidence les stratégies adoptées pour tirer le meilleur parti de cette richesse d'informations, malgré sa nature apparemment sporadique et disparate.

3.1.2 Traitement des données

Dans le cadre du traitement des données pour notre projet de prévision des précipitations, nous faisons face à un défi spécifique lié à la nature de nos données. Les précipitations, par essence, présentent une distribution très inégale : de vastes zones sans aucune précipitation sont juxtaposées à des zones de fortes précipitations. Cette disparité se manifeste par une abondance de valeurs nulles et quelques valeurs extrêmement élevées, reflétant des événements pluviométriques intenses. Cette distribution hétérogène peut compliquer l'apprentissage par les modèles de deep learning, en raison des écarts importants de magnitude entre les valeurs.

Pour remédier à cette situation et faciliter l'entraînement de nos modèles, il est crucial d'adopter une méthode de traitement des données qui permet de réduire ces disparités. La normalisation des données est une étape essentielle dans ce processus. Dans notre cas, nous avons opté pour une transformation logarithmique, une technique couramment utilisée pour gérer les distributions fortement asymétriques. En appliquant la fonction $ln(data + 1)$ à nos données, nous transformons l'échelle de nos valeurs de manière à réduire l'impact des valeurs extrêmes et à lisser les écarts entre les observations.

Cette transformation logarithmique présente plusieurs avantages pour notre modèle :

— **Réduction de la variance :** Elle diminue l'hétérogénéité de la distribution des précipitations, rendant les données plus homogènes et donc plus facilement modélisables.

— **Amélioration de la stabilité numérique :** En atténuant les écarts extrêmes, la transformation facilite le processus d'apprentissage en évitant les problèmes de convergence liés à la grande variété d'échelles.

— **Préservation des relations :** Bien que les données soient transformées, les relations intrinsèques entre les variables et la cible de prédiction restent intactes, permettant au modèle de capturer efficacement les dynamiques sous-jacentes des précipitations.

Il est crucial de souligner que cette transformation logarithmique nécessite une étape de post-traitement pour interpréter correctement les prédictions du modèle. Les valeurs prédites, issues de l'entraînement sur des données transformées, doivent être reconverties à leur échelle originale en appliquant la fonction $exp(prediction) - 1$ pour refléter fidèlement les niveaux de précipitations. Cette démarche est essentielle non seulement pour la compréhension des prédictions mais également pour l'évaluation de la performance du modèle à l'aide du *Mean Squared Logarithmic Error (MSLE)*. Le MSLE nous permet de mesurer l'erreur entre les prédictions et les valeurs réelles (y_test), toutes deux ajustées par la transformation inverse, offrant ainsi un score représentatif de l'exactitude de notre modèle.

En somme, l'adoption de la transformation logarithmique pour le prétraitement de nos données météorologiques est une réflexion stratégique visant à naviguer efficacement à travers les complexités inhérentes à ces données. Ce choix préparatoire est

déterminant pour améliorer la capacité prédictive de nos modèles CNN et L-CNN face aux défis de la prévision des précipitations. Au-delà de cette transformation universelle, chaque modèle bénéficiera de traitements spécifiques adaptés à ses exigences particulières, tels que le redimensionnement des données ou la génération de nouvelles variables, pour maximiser sa performance et sa pertinence dans le contexte de notre étude.

3.1.3 Extraction de données

La compréhension de la structure de nos données et la stratégie de leur prétraitement posent les bases nécessaires pour aborder l'étape suivante : l'extraction et la consolidation de ces données pour l'entraînement des modèles de CNN. Avec 100 000 fichiers correspondant à différentes observations, le défi majeur réside dans la capacité à regrouper efficacement ces données dans un unique tableau, tout en associant correctement chaque ensemble 'X_train' à son 'y_train' respectif. Cette association est facilitée en utilisant un modulo 8 sur le fichier '$y\_train.csv$' durant l'importation, permettant une correspondance précise entre les données d'entrée et leurs cibles.

L'importation des données 'X_train' et leur stockage dans un seul tableau est conceptuellement directe et s'effectue au moyen d'une boucle qui parcourt et intègre chaque fichier souhaité. Néanmoins, le véritable obstacle est lié à la gestion de la mémoire et au temps d'exécution nécessaire pour traiter un volume de données aussi conséquent. Bien que la manipulation de jusqu'à 10 000 fichiers reste gérable, augmenter cette quantité accentue de manière exponentielle la difficulté de maintenir la totalité de la matrice en mémoire sans rencontrer de contraintes de capacité.

Face à cette contrainte, l'idée d'utiliser une représentation matricielle sparse a été envisagée pour alléger la charge en mémoire. Toutefois, la prédominance de valeurs nulles au sein de nos données a paradoxalement augmenté la taille de la matrice lors de sa conversion en format sparse, contrecarrant l'objectif initial d'optimisation de l'espace.

Pour contourner ces limitations, nous avons adopté une approche basée sur le traitement par lots ("batches"), en commençant par des tests sur des échantillons de taille réduite (100, 500, 1000, et 10 000 fichiers) avant d'augmenter progressivement la taille des lots. Cette méthodologie nous a permis d'expérimenter avec différents modèles de manière plus agile et économique en termes de ressources. Grâce à l'utilisation de Google Colab sur des machines plus puissantes, dotées de capacités supérieures en RAM et en GPU, nous avons réussi à créer et manipuler des matrices contenant jusqu'à 50 000 et même 100 000 fichiers. Cependant, la matrice de 100 000 fichiers s'est avérée peu pratique, monopolisant une quantité excessive de mémoire et rendant l'entraînement des modèles peu viable.

Ainsi, les phases initiales de nos tests se sont concentrées sur des échantillons de taille modeste, avant de progresser vers des lots de 10 000 et, dans certains cas, de 50 000 fichiers. Cette démarche, bien que chronophage, s'est révélée indispensable pour gérer efficacement les défis liés à la mémoire et à la puissance de calcul, soulignant l'importance cruciale de l'accélération par GPU dans la réduction significative du temps d'entraînement pour les lots plus importants. Nous explorerons plus en détail l'impact de ces considérations techniques sur la performance de chaque modèle dans les sections suivantes.

3.2 Le CNN

3.2.1 Architecture et Paramètres

Dans notre projet de prévisions météorologiques utilisant un CNN, l'architecture du modèle a été soigneusement élaborée pour optimiser la capacité de prédiction des précipitations à partir de données satellitaires. Nous allons voir une description détaillée de l'architecture du modèle et des choix de paramètres effectués.

Conception du Modèle : Le modèle CNN conçu pour ce projet incorpore plusieurs couches pour traiter efficacement les caractéristiques spatiales des images satellitaires. L'architecture se compose de :

- Couches Convolutionnelles : Nous avons intégré plusieurs couches convolutionnelles pour extraire et hiérarchiser les caractéristiques spatiales des données. Chaque couche utilise des filtres de tailles variées pour capturer des motifs à différents niveaux de granularité.
- Couches de Pooling : Après chaque couche convolutionnelle, une couche de pooling est utilisée pour réduire les dimensions spatiales des cartes de caractéristiques, ce qui aide à diminuer le nombre de paramètres et à contrôler le surajustement.
- Couches Entièrement Connectées : En fin d'architecture, les couches entièrement connectées combinent les caractéristiques extraites pour effectuer la prédiction finale sur les précipitations.

Choix des Paramètres :

- Taille des Filtres : Les tailles des filtres dans les couches convolutionnelles ont été choisies pour équilibrer la capture des détails fins et la généralisation des motifs plus larges. Des filtres de petite taille (par exemple, 3x3) permettent de détecter de petits motifs, tandis que des filtres plus grands aident à capter des caractéristiques plus abstraites.
- Pas de Convolution (Stride) : Un pas de convolution standard a été utilisé pour permettre une exploration efficace des caractéristiques spatiales sans perdre d'informations cruciales.
- Fonctions d'Activation : La fonction d'activation ReLU a été choisie pour les couches cachées en raison de sa capacité à introduire de la non-linéarité dans le modèle tout en évitant le problème de disparition des gradients.
- Méthode d'Optimisation : Plusieurs optimiseurs ont été testés, y compris Adam, RMSprop, et Ftrl, chacun avec des configurations d'hyperparamètres distinctes. Ces choix ont été guidés par la recherche de la meilleure convergence et performance du modèle sur les données d'entraînement et de validation.

La stratégie d'expérimentation avec diverses configurations, tailles de batch et nombres d'époches, a permis d'itérer rapidement pour identifier les architectures les plus performantes. En commençant par des ensembles de données plus petits pour des tests rapides, nous avons progressivement augmenté la taille de l'ensemble d'entraînement pour affiner et valider nos modèles. Cette approche graduelle a permis d'optimiser l'utilisation des ressources de calcul tout en explorant l'espace des hyperparamètres de manière exhaustive.

Il y a également eu l'adoption d'un CNN 3D qui a été motivée par l'objectif d'exploiter non seulement les caractéristiques spatiales mais aussi la dimension temporelle des données. Cette approche avancée nous permet de saisir avec plus de finesse l'évolution des précipitations au fil du temps, améliorant ainsi le potentiel de prédictions précises.
Face à la complexité et aux besoins en calcul accrus des CNN 3D, nous avons opté pour une méthode d'exploration sélective. Au lieu de tester une vaste gamme de configurations comme pour le CNN 2D, nous avons focalisé nos efforts sur l'optimisation d'une architecture de CNN 3D, en nous basant sur le modèle 2D le plus performant. Pour cela, nous avons employé un "tuner" de modèle spécialement adapté aux CNN 3D, dans le but d'affiner les hyperparamètres et de déterminer la configuration la plus efficace pour nos données.

La phase finale de sélection du modèle, tout comme pour le CNN classique, a impliqué l'entraînement sur un sous-ensemble de 50 000 données, à travers 5 essais menés avec le tuner. Cette approche nous a permis d'explorer de manière efficiente l'espace des hyperparamètres, malgré les limitations de temps et de ressources de calcul. L'optimiseur RMSprop a été privilégié pour le tuner en raison de sa capacité à ajuster dynamiquement le taux d'apprentissage, ce qui est crucial pour gérer la complexité des architectures de CNN 3D.

Le modèle "optimal" obtenu grâce au tuning a révélé des améliorations notables dans la prédiction des précipitations, mettant en lumière l'importance cruciale de la composante temporelle dans les analyses météorologiques. Cependant, la mise en œuvre des CNN 3D se heurte à des défis non négligeables, notamment en terme de besoins accrus en mémoire et en temps de calcul.

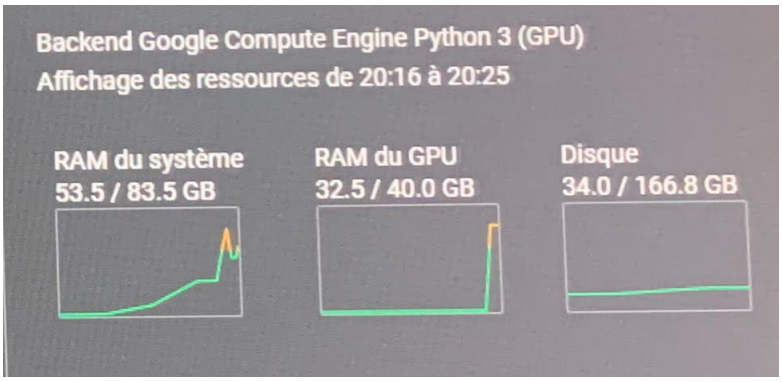

FIGURE 5 – Ressources RAM et VRAM Google Colab

On voit ici donc l'utilisation de la mémoire RAM ainsi que de la mémoire RAM GPU utilisées lors d'un entraînement sur 50000 fichiers pour un CNN 3D.

Ainsi en résumé, pour le CNN simple nous avons utiliser une pipeline utilisant en entrainant sur plusieurs configurations de modele, de manière analogue nous avons procéder pour le CNN 3D mais en prenant le modele 2D le plus performant et en le tunant.

3.2.2 Traitement et Préparation des Données

Après la collecte des données et leur transformation logarithmique ($ln(data+1)$), comme mentionné dans la section 3.1, nous avons procédé à la restructuration de notre ensemble de données X_train pour qu'il corresponde au format attendu par les modèles CNN. Pour le CNN traditionnel, nous avons remodelé X_train pour obtenir une forme de (nombre de fichiers, 128, 128, 4), chaque "4" représentant les différentes images temporelles pour chaque zone géographique.

Pour adapter nos données au CNN 3D, qui prend en compte l'aspect temporel en plus des caractéristiques spatiales, nous avons effectué un remodelage légèrement différent, aboutissant à une forme de (nombre de fichiers, 4, 128, 128, 1). Ce format permet au modèle 3D de traiter la séquence temporelle des images comme une dimension supplémentaire, offrant une perspective plus riche sur l'évolution des précipitations.

```
Shape de X: (2500, 4, 128, 128)
Shape de X_reshaped: (2500, 128, 128, 4)
Shape de X_3D: (2500, 4, 128, 128, 1)
```

FIGURE 6 – Dimensions de nos données X pour CNN 2D et 3D

3.2.3 Détails de l'Implémentation

La mise en œuvre de nos modèles CNN et CNN 3D pour la prévision météorologique a été orchestrée à l'aide d'un ensemble d'outils et de technologies nous venant directement des librairies Python les plus connu dans le domaine conçus pour maximiser l'efficacité et la précision des prédictions. Voici un aperçu des principaux éléments de notre environnement de développement et de la gestion des ressources :

Environnement de Développement

Les modèles ont été développés en utilisant **TensorFlow** et **Keras**, deux des bibliothèques les plus réputées dans le domaine de l'apprentissage profond. TensorFlow offre une plateforme complète et flexible pour la conception de modèles complexes, tandis que Keras, agissant en tant qu'interface de haut niveau, simplifie la création de réseaux de neurones en rendant le code plus accessible et plus facile à lire. Cette combinaison a permis une expérimentation rapide et efficace, en facilitant l'ajustement des architectures de réseau et des hyperparamètres.

Gestion des Ressources

La puissance de calcul représente un facteur critique dans l'entraînement des modèles de deep learning, particulièrement lorsqu'il s'agit de CNN et de CNN 3D, qui sont notoirement gourmands en ressources, comme nous l'avons vu plus haut. Pour surmonter ce défi, nous avons opté pour l'utilisation de GPU (Graphical Processing Units), qui accélèrent significativement les calculs matriciels et vectoriels au cœur des opérations de convolution.
En outre, une partie de notre travail a été réalisée sur des plateformes *cloud*, telles que Google Colab, offrant un accès à des GPU puissants et à des environnements de calcul élastiques. Cela a non seulement permis de réduire les temps d'entraînement

mais a également offert la flexibilité nécessaire pour expérimenter avec des ensembles de données de grande taille et pour ajuster la complexité des modèles.
La gestion efficace de ces ressources a été essentielle pour naviguer dans les contraintes de performance et pour réaliser des expérimentations approfondies, menant à la sélection des architectures de modèle les plus prometteuses. En combinant des environnements de développement avancés avec une stratégie réfléchie pour la gestion des ressources de calcul, nous avons pu optimiser le processus de développement et d'évaluation des modèles, contribuant ainsi à l'avancement de nos objectifs de prévision météorologique.

3.2.4 Résultats obtenus et analyse

Dans la phase d'évaluation de nos modèles CNN pour la prévision des précipitations, nous avons employé le MSLE comme métrique principale. Le MSLE est particulièrement adapté à notre contexte, où nous traitons avec des données de précipitations qui présentent une grande variabilité et des écarts importants entre les valeurs. Cette métrique évalue la différence logarithmique entre les prédictions et les valeurs réelles, ce qui minimise l'impact des grandes erreurs sur les prédictions de faibles précipitations et met l'accent sur la proportionnalité plutôt que sur la différence absolue. Avant de calculer le MSLE, il est crucial de retransformer les prédictions et les valeurs réelles à l'échelle originale en appliquant la fonction exponentielle, compensant ainsi la transformation logarithmique initiale appliquée aux données.

Nos meilleurs résultats ont été obtenus en utilisant l'optimiseur **RMSprop** avec un taux d'apprentissage de 0.001, sur un ensemble d'entraînement de 50 000 fichiers. Cette configuration a permis de surpasser les scores de référence du classement, avec le modèle $b32\_e20$ (batch size de 32 et 20 epochs) atteignant un score MSLE de 0.129, suivi par le modèle CNN_3D sur 50000 fichiers avec un score de 0.138, et enfin le modèle $b16\_e10$ (batch size de 16 et 10 epoches) avec un score de 0.159. Chacun de ces modèles a produit des scores inférieurs à 0.2, indiquant une performance remarquable.

Pour comprendre l'impact du nombre d'epoches et de la taille du batch sur ces résultats, il est essentiel de définir ces termes. Un *epoch* représente un cycle complet d'entraînement sur l'ensemble des données, permettant au modèle d'ajuster ses poids à chaque observation. Un nombre plus élevé d'époques offre au modèle plus d'opportunités pour apprendre des données, mais avec le risque de surajustement si ce nombre est excessif. Le $batch\_size$ se réfère au nombre d'échantillons traités avant que le modèle ne mette à jour ses poids. Un batch size plus petit offre une mise à jour des poids plus fréquente, ce qui peut conduire à une convergence plus rapide, mais avec une variabilité plus élevée des gradients.

L'observation que les configurations avec des tailles de batch et des nombres d'époques variés montrent des performances différentes souligne l'importance d'ajuster ces hyperparamètres en fonction des spécificités des données et de l'architecture du modèle. Le modèle $b32\_e20$ a probablement bénéficié d'un équilibre optimal entre la fréquence de mise à jour des poids et le nombre de cycles d'apprentissage, permettant une convergence efficace vers une solution de haute qualité. En revanche, l'approche 3D, bien que plus gourmande en ressources, a réussi à capturer la dimension temporelle des précipitations, offrant une perspective complémentaire et précieuse qui se reflète dans son score compétitif.

Ces résultats mettent en évidence la capacité des CNN à fournir des prévisions météorologiques précises, tout en soulignant l'importance de l'expérimentation avec les hyperparamètres et la structure du modèle pour optimiser la performance.

3.3 L-CNN

Dans notre approche avec le L-CNN, les données initiales, stockées au format .npz et composées de quatre matrices 128x128 représentant les précipitations d'une même zone à quatre instants consécutifs, sont traitées de manière innovante. Contrairement à l'approche classique des CNN, où les données sont directement alimentées au modèle, le L-CNN incorpore une étape préliminaire cruciale : l'analyse de la dynamique des précipitations à travers la détermination de la vitesse apparente des changements observés entre chaque paire d'images successives.

3.3.1 Vitesse par analyse de dérivées

Cette analyse de la vitesse est réalisée de deux manières distinctes. La première méthode consiste à calculer la différence absolue entre les matrices successives, ce qui nous fournit une estimation grossière de la vitesse de déplacement des précipitations d'un instant à l'autre. Cette opération, simplifiée en une ligne de code :

$$speed = np.abs(x\_train\_data[i+1] - x\_train\_data[i])$$

permet de saisir les variations d'intensité de précipitations entre les images.

Toutefois, la présence de valeurs aberrantes dans nos données, avec des intensités variant de 0 à 3000 et une forte proportion de valeurs nulles, soulève des défis spécifiques. Pour atténuer l'impact de ces extrêmes et homogénéiser la distribution des intensités, nous appliquons systématiquement une transformation logarithmique à nos données. Ce traitement préalable assure non seulement une réduction de la variance mais facilite également l'identification des caractéristiques pertinentes sans être dominé par les données extrêmes.

L'adoption de la valeur absolue dans le calcul de la vitesse se justifie par la nécessité de maintenir une positivité dans les données traitées, une condition préalable pour l'entraînement efficace de notre modèle.

3.3.2 Vitesse par flux optique

Le flux optique, déjà implémenté sous le nom de calcOpticalFlowFarneback sur Python, est une fonction intégrée par la libraire cv2. Son utilisation reste simple, prenant en entrée l'image initiale et l'image d'arrivée pour calculer le flux séparant ces 2 images. Divers paramètres sont ajustables mais nous ne nous attarderons pas dessus dans ce papier. Une fois que la fonction a été implémentée, une nouvelle matrice de même dimension que celles en entrée sera générée. Pour 4 images de chaque donnée, nous avons ainsi 3 matrices de flux optiques qui sont générées. Le flux optique, produit par la fonction Lucas-Kanade (ou Farneback dans la littérature), étant très sensible aux grandes différences d'image, peut créer des perturbations qualifiées de flux infinis.

Pour contrecarrer cette problématique, les vitesses infinies détectées sont ajustées à la valeur maximale observée parmi les données non nulles, préservant ainsi la cohérence et l'intégrité de l'information relative au mouvement des précipitations.

Cette approche méthodique, alliant transformation logarithmique et ajustement des vitesses, sous-tend notre stratégie d'exploitation du L-CNN. Elle illustre notre engagement à affiner la précision de nos prévisions météorologiques en adaptant intelligemment nos méthodes d'analyse aux spécificités et aux défis posés par les données de précipitations.

3.3.3 Détails de l'implémentation

Nous décidons donc d'appliquer la fonction de flux lors de la récolte des données. Afin de ne pas réitérer ce processus de multiples fois, nous exécutons un premier programme dans lequel nous indiquons le nombre de fichier d'entraînement souhaité. Suite à l'importation des données, nous appliquons la fonction **calcOpticalFlowFarneback** à ces dernières afin de récupérer leur flux optique. Étant donné que chaque fichier de donnée comporte 4 images, nous calculons 3 vitesses à l'aide du flux optique. Ces vitesses, calculées en coordonnées cartésiennes, seront par la suite transformées en coordonnées polaires (cf. 2.2 Introduction aux coordonnées Lagrangiennes et leur application en CNN (L-CNN)).

En entrée du CNN, nous retrouvons donc un fichier d'entraînement X_train qui contient N fichiers, comprenant chacune 4 matrices/images de position de dimension 128x128, 3 matrices avec la composante en magnitude des vitesses et 3 autres matrices avec la composante en angle de la vitesse, soit 10 canaux au total. Tout cela combiné, nous retombons sur un X_train de dimension :

$$dim(X\_train) = (N, 128, 128, 10)$$

L'augmentation des données permet d'apporter de l'information dans nos modèles et ainsi capturer de meilleures caractéristiques. Cependant, cela implique également un temps d'entraînement plus conséquent, limitant ainsi le nombre de test de modèles.

La lenteur des exécutions ne nous a permis d'entraîner les différents modèles que sur 10k fichiers parmi les 100k disponibles dans le challenge. Cependant, nous verrons que cette restriction de données n'implique pas une baisse de la qualité des modèles, même si un enrichissement du X_train aurait sûrement fourni des résultats plus impressionnants.

Ceci étant dit, nous générons donc les fichiers $data\_log$, $optical\_magnitude\_log$, $optical\_angle\_log$ en local (les données ont été téléchargées en local). Nous utilisons ensuite divers outils pour créer le fichier X_train, comme laa plateforme $forgeim$. Cette dernière dispose de près de 200go de RAM, pouvant ainsi gérer des tâches lourdes.

Pour chercher les meilleurs modèles possibles, nous utilisons le principe du *tuner*. Ce dernier permet de tester de mutliples paramètres et garder le meilleur modèle possédant le meilleur benchmark (basé sur le MSE (mean squared error)). Voici un exemple d'implémentation :

```
            # Premier bloc de convolution
        model.add(Conv2D(filters=hp.Int('conv_1_filter', min_value=32,
    max_value=128, step=32), kernel_size=hp.Choice('conv_1_kernel',
    values=[3, 5]), padding='same', input_shape=self.input_shape))
        model.add(BatchNormalization())
        model.add(Activation('relu'))
        model.add(MaxPooling2D((2, 2)))
        model.add(Dropout(rate=hp.Float('dropout_1', min_value=0.0, max
    _value=0.5, step=0.1)))
```

Nous donnons donc pour cette première couche de convolution des paramètres parmi lesquels le tuner pourra sélectionner pour le modèle testé. Bien sûr, le nombre de couche de convolution, d'epoch ou encore la taille du batch devront être choisis au préalable par l'utilisateur. Certains paramètres n'ont délibéremment pas été laissé libre au tuner car ils sont fixés et conviennent à nos données (tout comme la fonction d'activation 'ReLU').

Notre cirtère de validation reste le MSE, nous choisissons donc le modèle en fonction de ce dernier :

```
tuner = RandomSearch(
    hypermodel,
    objective='mse',
    max_trials=5,  # Nombre de configurations d'hyperparametres a
    tester
    executions_per_trial=1,  # Combien de fois chaque configuration
    sera testee
    directory='my_dir',  # Repertoire pour stocker les logs
    project_name='l_cnn_tuning'
)
```

3.3.4 Résultats obtenus et Analyse

Après avoir testé les 2 méthodes différentes sur plusieurs configurations, nous remarquons que le flux optique ajoute beaucoup de complexité et donc de temps de calcul. Cela s'explique par la différence de données en entrée qui plus importante (7 canaux pour la vitesse par dérivée contre 10 canaux pour le flux optique). De plus, les matrices ôbtenues par le flux optique génèrent des valeurs extrêmes voire infinies que l'on doit prendre en compte dans les entraînements.

Toujours sur 10k fichiers d'entraînement, un modèle sentraînant sur la base du flux optique prend environ 1h d'entraînement. Avec le même type de modèle et les mêmes données d'enraînement, seulement 20 minutes sont nécessaires à l'entraînement du modèle avec vitesse dérivée. Nous obtenons finalement un score $MLSE$ de 0.32 pour le L-CNN avec flux optique contre 0.25 pour le meilleur modèle à vitesse issue de dérivée.

Finalement, la vitesse obtenue par la dérivée est plus facile à implémenter et permet de mieux contrôler les valeurs obtenues. Cependant, on pourrait penser qu'une telle simplicité pourrait entraîner un manque de précision dans nos prédictions.

Nous verrons en conclusion que ce n'est pas le cas, nous interrogeant donc sur les implémentations des méthodes. Est-ce que 10k fichiers étaient réellement suffisants pour obtenir une bonne précision du modèle ? Est-ce que le CNN a été correctement ajusté et optimisé ?

4 Résultats et comparaison

4.1 CNN 2D et CNN 3D

Les résultats obtenus à travers l'application des CNN 2D et 3D sont très prometteurs, marquant une avancée significative dans la prédiction précise des précipitations. L'entraînement sur un vaste ensemble de 50 000 fichiers a mis en lumière l'impact crucial de l'accélération GPU sur le processus. Cette technologie a non seulement facilité l'expérimentation avec une multitude de modèles, notamment ceux de nature plus complexe comme les CNN 3D, mais a également réduit considérablement le temps d'exécution, rendant les tests plus agiles et efficaces. Par exemple, pour les modèles CNN 2D simples, nous avons observé un gain de performance allant jusqu'à dix fois, voire cent fois pour les architectures 3D complexes sur de grands jeux de données.

Les résultats présentés ci-après sont issus de prédictions effectuées sur un échantillon de données composé de 2 500 fichiers, soigneusement sélectionnés parmi les 50 000 derniers fichiers de notre jeu de données. Cette méthode garantit que les modèles sont évalués sur des données inédites, assurant ainsi une comparaison équitable et fiable entre eux. Cette approche permet de mettre en lumière la capacité de généralisation de chaque modèle face à des situations météorologiques qu'ils n'ont pas précédemment rencontrées, un aspect crucial pour la validation de leur efficacité prédictive dans des conditions réelles.

Pour le CNN classique :

À travers l'utilisation d'une pipeline, divers optimiseurs ont été testés (Ftrl, RMSprop, AdamW, Adagrad) sur un ensemble initial de 1 000 fichiers. Les scores obtenus variaient de 1.36 à 0.68, ce dernier étant attribué à RMSprop, avec un temps d'exécution total d'environ 10 minutes pour les quatre tests. Après une sélection rigoureuse, Ftrl et RMSprop ont été retenus pour une exploration plus approfondie sur 10 000 fichiers. Cette phase a permis d'affiner les scores à des valeurs avoisinant 0.5, RMSprop démontrant une fois de plus sa supériorité. Finalement, l'entraînement sur 50 000 fichiers a été effectué en utilisant RMSprop, explorant diverses configurations et combinaisons d'epochs (10, 20) et de batch sizes (16, 32). Ce processus a donné lieu à quatre modèles d'exception. Grâce à l'utilisation de Google Colab et à l'accélération GPU, le temps moyen d'exécution par epoch a été réduit à 13 secondes, contre environ 150 secondes sur ForgeIM pour 10 000 fichiers.

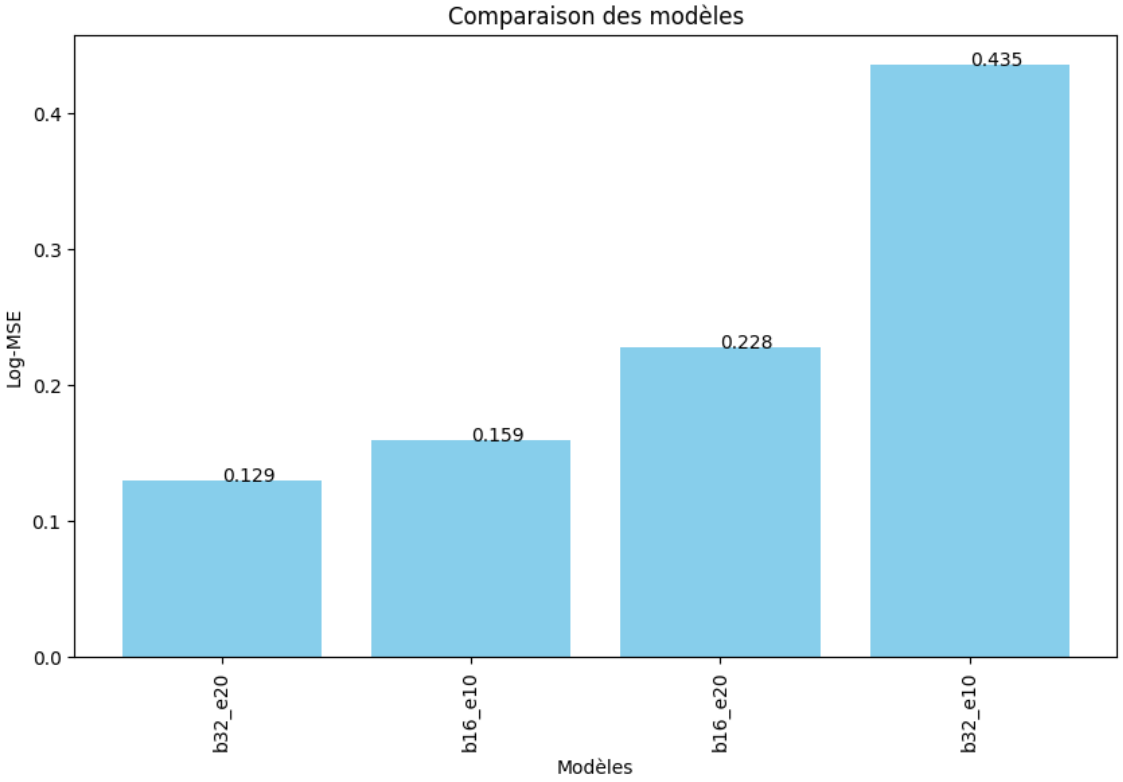

FIGURE 7 – Modèles CNN 2D - RMSprop

Pour le CNN 3D :

Adoptant une démarche similaire pour le CNN 3D, avec une adaptation des filtres et des données d'entrée pour tenir compte de la dimension temporelle, les contraintes de temps et de ressources ont conduit à une stratégie focalisée. Utilisant le meilleur modèle CNN 2D comme référence, un tuner a été déployé pour optimiser un CNN 3D sur cinq 'trials', testant ainsi les hyperparamètres les plus prometteurs. Le recours à Google Colab a permis de réaliser ces trials en 1h23min pour un ensemble de 50 000 fichiers, démontrant un gain de temps substantiel par rapport aux performances sur des systèmes standards. Ce modèle 3D optimisé s'est distingué par son efficacité et sa précision dans la prédiction des précipitations, illustrant l'importance de la composante temporelle dans l'analyse météorologique.

Voici donc à la fin du 'tunnage' qu'on obtiens :

```
Trial 5 Complete [00h 13m 24s]
mse: 0.07367026060819626

Best mse So Far: 0.07367026060819626
Total elapsed time: 01h 23m 07s
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv3d (Conv3D)             (None, 4, 128, 128, 64)   1792

 max_pooling3d (MaxPooling3  (None, 2, 64, 64, 64)     0
 D)

 dropout (Dropout)           (None, 2, 64, 64, 64)     0

 conv3d_1 (Conv3D)           (None, 2, 64, 64, 128)    221312

 max_pooling3d_1 (MaxPoolin  (None, 1, 32, 32, 128)    0
 g3D)

 dropout_1 (Dropout)         (None, 1, 32, 32, 128)    0

 conv3d_2 (Conv3D)           (None, 1, 32, 32, 64)     1024064

 max_pooling3d_2 (MaxPoolin  (None, 1, 16, 16, 64)     0
 g3D)

 dropout_2 (Dropout)         (None, 1, 16, 16, 64)     0

 flatten (Flatten)           (None, 16384)             0

 dense (Dense)               (None, 64)                1048640

 dropout_3 (Dropout)         (None, 64)                0

 dense_1 (Dense)             (None, 8)                 520

=================================================================
Total params: 2296328 (8.76 MB)
Trainable params: 2296328 (8.76 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

FIGURE 8 – Modèle 3D apres passage tuner

Ainsi on a obtenu notre modele 3D, qui a également un bon score MLSE de **0.138**.

4.2 L_CNN vitesse et flux

Pour le L_CNN vitesse classique

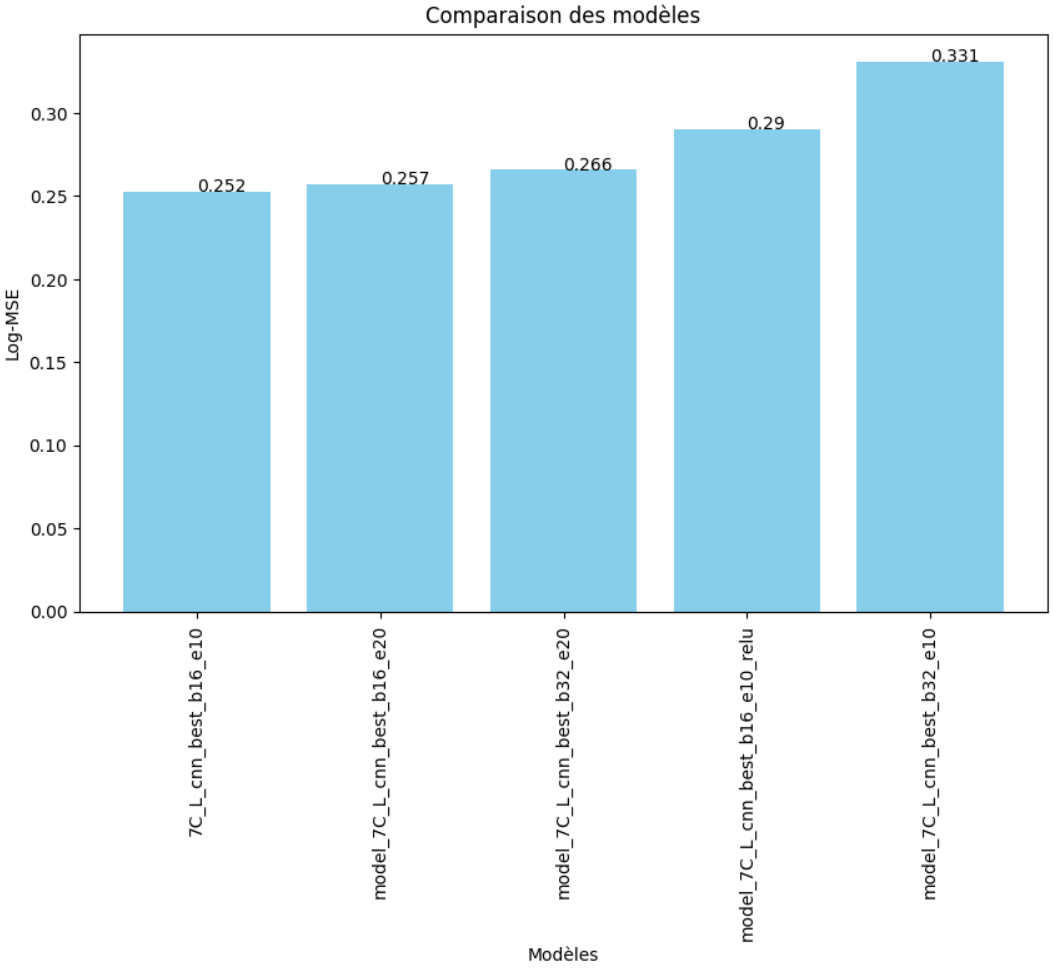

FIGURE 9 – Modèles L_CNN 7 canaux

Comme mentionné plus haut, le meilleur score $MLSE$ obtenu ici est de 0,25. On obtient des résultats à peu près similaires avec les autres configurations. On remarque que lorsqu'on opte pour une couche dense linéaire, on obtient globalement de meilleurs résultats que la couche dense ReLU.

Pour le L_CNN vitesse flux optique

Ainsi on a obtenu notre modele L_CNN flux optique, avec un score MLSE de **0.324**. Ce dernier n'est certainement pas représentatif de la performance du modèle qui mériterait un entraînement avec plus de fichiers pour obtenir des résultats plus convaincants.

4.3 Comparaison

Nous avons ainsi tous nos meilleurs modele de chaque type :

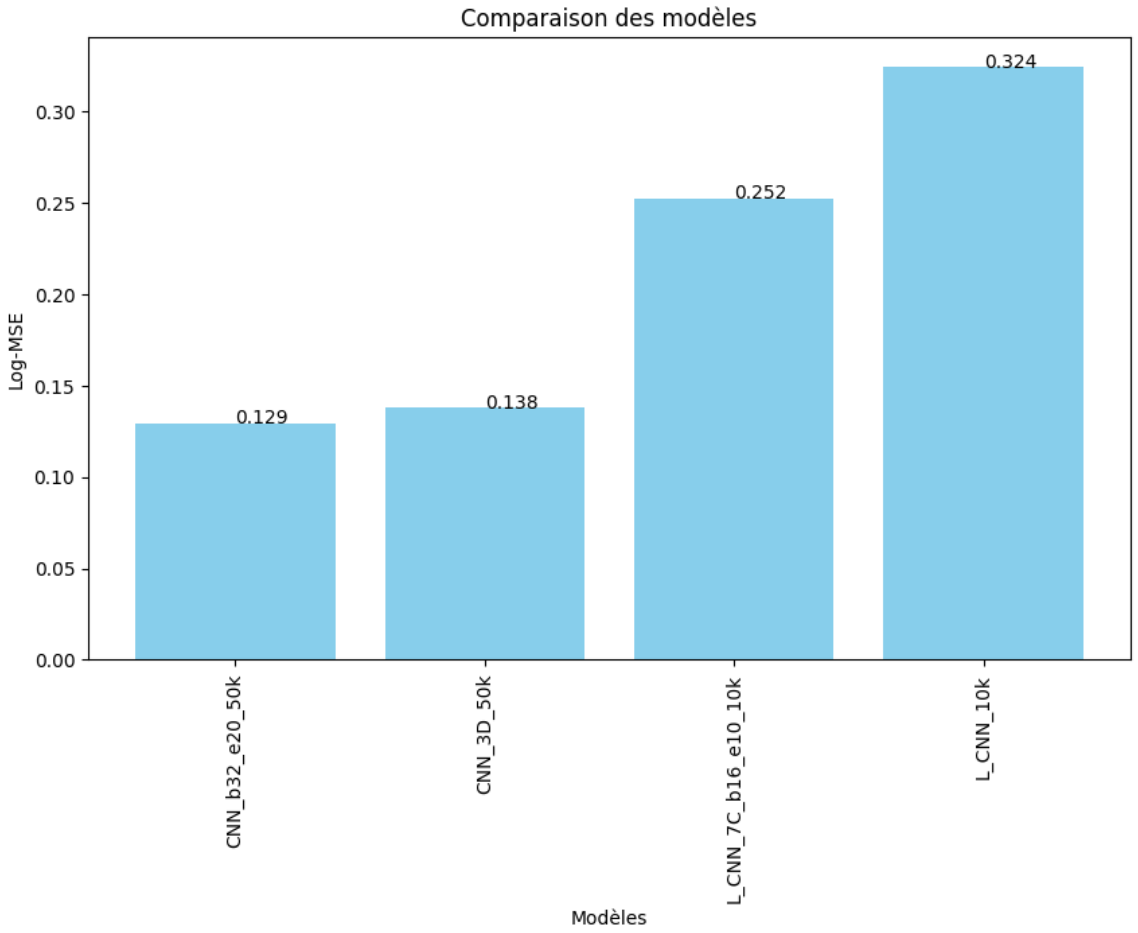

FIGURE 10 – Meilleurs modeles - RMSprop

On voit que nous avons des bons score pour eux tous, gardons en tête que les CNN ont été entrainé sur 50k données contraiment aux L_CNN qui eux ont été entrainéer sur 10k données, ceci dit a comparaison des résultats entre les différentes approches de modélisation, CNN 2D, CNN 3D, et L-CNN (avec vitesse classique à 7 canaux et flux optique), révèle des insights importants quant à l'efficacité et l'applicabilité de chaque méthode dans le contexte de la prévision des précipitations.

Voici donc une analyse comparative prenant en compte les performances, les avantages, et les inconvénients de chaque modèle :

Performances : Les modèles CNN 2D et 3D affichent les scores les plus compétitifs, avec une mention spéciale pour le modèle CNN 2D '$b32\_e20$' qui présente le meilleur score de 0.129, suivi de près par le CNN 3D à 0.138. Ces résultats démontrent l'efficacité des CNN dans la capture des caractéristiques spatiales et temporelles pour la prévision des précipitations. Les L-CNN, bien que présentant des scores légèrement inférieurs, offrent une perspective différente dans l'analyse des données, en exploitant les vecteurs de vitesse et le flux optique pour comprendre la dynamique des précipitations.

Avantages :

- CNN 2D et 3D : Ces modèles bénéficient de la capacité intrinsèque des CNN à traiter efficacement les données visuelles, permettant une extraction précise des caractéristiques pertinentes pour la prévision. Le modèle 3D, en particulier, intègre l'aspect temporel, offrant une perspective plus complète sur l'évolution des précipitations.

- L-CNN : L'approche L-CNN, en utilisant des données telles que la vitesse et le flux optique, permet une analyse fine du mouvement et de la propagation des précipitations, ouvrant la voie à des prévisions potentiellement plus nuancées.

Inconvénients :

- CNN 3D : Malgré son potentiel pour intégrer la dimension temporelle, le CNN 3D exige des ressources computationnelles significativement plus importantes, ce qui peut limiter son applicabilité pratique, surtout pour des ensembles de données de grande taille.

- L-CNN (flux optique) : Bien que fournissant une analyse détaillée du mouvement des précipitations, cette méthode peut s'avérer complexe à implémenter et gourmande en termes de calcul, ce qui pose des défis pour son déploiement à grande échelle.

On a ainsi que la sélection entre un CNN 2D, un CNN 3D, et un L-CNN dépendra largement des objectifs spécifiques de la prévision, des ressources disponibles, et de la complexité des données à traiter. Tandis que les CNN 2D et 3D offrent des performances supérieures en termes de scores MLSE, les L-CNN apportent une compréhension approfondie des dynamiques de précipitations, à condition de pouvoir gérer les défis liés à leur exigence computationnelle. Cette analyse souligne l'importance d'équilibrer précision, complexité computationnelle, et compréhension dynamique des phénomènes pour choisir l'approche la plus adaptée à chaque cas d'usage.

4.4 Prédictions

Voyons voir tout de même quelques prédictions de nos modèles afin d'avoir une situation plus concrète. Avant de passer au vraies prédictions sur des échantillons donnée, voyons voir quelques statistiques de nos modèles comparés à celle du y_test :

	max	min	mean
Y	777.000000	0.000000	5.555050
Y_pred_b32_e20	724.441223	-0.615600	5.341000
Y_pred_3D_50k	1462.467896	-0.616901	3.579393
Y_pred_L_cnn	32679.447266	-0.465690	8.613556
y_pred_7C_L_cnn_best_b16_e10	677.439575	-0.601954	3.710590

FIGURE 11 – Statistiques sur les prédictions de nos modeles

Les statistiques révèlent des différences notables entre les valeurs réelles (Y) et les prédictions (Y_pred) des différents modèles. Le modèle $b32\_e20$ présente des prédictions proches des valeurs réelles en termes de maximum, minimum, et moyenne, indiquant une bonne capacité de capture de la distribution générale des précipitations. En revanche, le modèle 3D_50k montre une valeur maximale prévue nettement plus élevée, suggérant une possible surévaluation des événements de forte précipitation. Le modèle L-CNN a, quant à lui, une moyenne de prédictions considérablement plus élevée, indiquant une tendance à surestimer les précipitations. Cette observation est particulièrement pertinente pour comprendre comment chaque modèle gère les extrêmes et les valeurs moyennes des précipitations.

Passons à présenet sur les prédictions, si on prend notre meilleur modele (en terme de score et de statistiques) on a sur un echantillon de 5 prédictions le résultats suivants :

```
Echantillon 0:
Prédictions : [70.570244  59.72348   42.795685  24.70245   14.330467   8.532881
  5.433725   2.8886716]
Valeurs réelles : [80. 80. 18. 18. 18. 18. 18.  1.]

Echantillon 1:
Prédictions : [0.24126792 0.2373184  0.17922771 0.15274191 0.11838579 0.13033128
 0.1262368  0.16251945]
Valeurs réelles : [0. 0. 0. 0. 0. 0. 0. 0.]

Echantillon 2:
Prédictions : [7.507077 7.870902 8.659358 9.053669 9.821349 9.688657 9.348973 8.089076]
Valeurs réelles : [ 8.  8.  8.  8. 15. 15. 15. 15.]

Echantillon 3:
Prédictions : [-0.01798218 -0.00262231 -0.02476889 -0.00220627  0.00702047  0.07033968
  0.1089623   0.21431077]
Valeurs réelles : [0. 0. 0. 0. 0. 0. 0. 0.]

Echantillon 4:
Prédictions : [10.0517025   7.8649435   4.868389    2.372875    0.8636968   0.19922018
 -0.15609485 -0.3998565 ]
Valeurs réelles : [27. 27.  2.  2.  2.  0.  0.  0.]
```

FIGURE 12 – Predictions de notre meilleur modele

L'échantillon de prédictions offert donne un aperçu de la performance modèle par modèle :

— Echantillon 0 : Le modèle réussit à prédire des événements de précipitations significatives avec une précision raisonnable, bien que légèrement inférieure aux valeurs réelles. Cette capacité à détecter des événements plus intenses est cruciale pour la planification et la réponse aux conditions météorologiques extrêmes.

— Echantillon 1 : La prédiction précise de l'absence de précipitations démontre l'efficacité du modèle à identifier correctement les périodes sèches, un aspect important pour éviter les fausses alertes dans les prévisions météorologiques.

— Echantillon 2 : Ici, le modèle montre sa capacité à suivre une tendance générale dans l'évolution des précipitations, malgré certaines variations par rapport aux valeurs réelles. Cela illustre l'importance de l'aspect temporel dans la prévision des précipitations.

— Echantillon 3 et 4 : Les prédictions montrent des variations mineures autour de zéro, indiquant une certaine difficulté à prévoir avec précision les faibles précipitations. Cela peut refléter les défis liés à la modélisation de phénomènes météorologiques moins intenses.

Ces observations montrent que, malgré certaines limitations, les modèles présentent une capacité prometteuse à prédire les tendances des précipitations. Le modèle $b32\_e20$ se distingue par sa performance globale, offrant un équilibre entre la précision des prévisions pour les événements de forte et de faible intensité.

En conclusion, bien que chaque modèle ait ses forces et faiblesses, l'analyse des prédictions met en lumière l'importance d'une approche équilibrée qui prend en compte à la fois la précision des prévisions et les ressources disponibles. La poursuite de l'optimisation des modèles, en particulier à travers des techniques comme le tuning des hyperparamètres, reste une voie essentielle pour améliorer la fiabilité des prévisions météorologiques à court terme.

5 Conclusion

Dans ce projet, nous avons exploré l'utilisation des réseaux de neurones convolutifs (CNN) et des CNN Lagrangiens (L-CNN) pour la prévision des précipitations à court terme, un défi crucial dans le domaine de la météorologie. L'objectif était de tirer parti de ces technologies avancées pour améliorer la précision des prédictions météorologiques, essentielles pour la planification des activités humaines et la gestion des risques liés aux événements climatiques extrêmes.

Synthèse des Découvertes

Nos expérimentations avec les CNN 2D et 3D ont révélé des capacités prometteuses pour la modélisation et la prévision des précipitations, avec des améliorations notables de la précision par rapport aux méthodes traditionnelles. L'intégration de la dimension temporelle dans les CNN 3D a particulièrement montré comment la prise en compte de l'évolution temporelle des phénomènes météorologiques peut enrichir les prédictions. Parallèlement, l'approche L-CNN a offert une perspective novatrice en incorporant les mouvements et les changements dynamiques à travers les coordonnées Lagrangiennes, ouvrant la voie à une compréhension plus profonde des processus météorologiques.

Défis et Limitations

Cependant, ces avancées ne sont pas sans défis. La complexité accrue des modèles 3D et L-CNN impose des exigences élevées en termes de puissance de calcul et de mémoire, limitant leur accessibilité et leur applicabilité dans certains contextes. De plus, la qualité et la quantité des données disponibles jouent un rôle crucial dans la performance des modèles, soulignant l'importance d'une collecte et d'une préparation minutieuses des données.

Perspectives Futures

Pour surmonter ces obstacles et continuer à améliorer les prévisions météorologiques, plusieurs pistes peuvent être explorées :

— Optimisation des Modèles : L'exploration continue des architectures de réseaux et des techniques d'optimisation peut conduire à des modèles plus efficaces et moins gourmands en ressources.

— Enrichissement des Données : L'intégration de sources de données supplémentaires, telles que des observations au sol ou des mesures satellites avancées, pourrait améliorer la robustesse et la précision des prédictions.

— Apprentissage Semi-supervisé et Auto-supervisé : Ces approches pourraient permettre d'exploiter des ensembles de données plus vastes, y compris des données non étiquetées, pour renforcer l'apprentissage des modèles.

En conclusion, ce projet a démontré le potentiel significatif des CNN et des L-CNN pour révolutionner la prévision des précipitations. Bien que des défis demeurent, les avancées réalisées ouvrent des perspectives prometteuses pour l'avenir de la météorologie. En continuant à pousser les frontières de la recherche et en adoptant une approche collaborative, nous pouvons aspirer à des systèmes de prévision toujours plus précis et fiables, au service des sociétés et de leur capacité à faire face aux caprices du climat.

6 Bibliographie

Références

[1] MeteoNet, *MeteoNet Project*, https://meteonet.umr-cnrm.fr/index.html.

[2] Keras, *Regression metrics*, https://keras.io/metrics/#regression-metrics.

[3] Papers With Code, *DeepLabv3 Explained*, https://paperswithcode.com/method/deeplabv3.

[4] Keras, *Optimizers*, https://keras.io/optimizers/.

[5] TensorFlow, *Video classification with a CNN-RNN architecture*, https://www.tensorflow.org/tutorials/video/video_classification.

[6] Antoine Manzanera, *Cours de Vision par Ordinateur : Séquence et Vidéo*, https://helios2.mi.parisdescartes.fr/~lomn/Cours/CV/SeqVideo/CoursComplements/Francais/Manzanera.pdf.

[7] TensorFlow, *Video Classification*, https://www.tensorflow.org/tutorials/video/video_classification.

[8] *Advection-Free Convolutional Neural Network for Convective Rainfall Nowcasting*, https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10021317.