الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Projet de Fin d'Études

Présenté par :

Taouinet Roumaissa

Pour l'obtention du diplôme de Master en Télécommunications

Option : Réseaux et Télécommunications

Thème

Prédiction de l'état cellulaire LTE par apprentissage profond pour l'équilibrage de charge de mobilité dans les réseaux cellulaires 4G de Djezzy

Dirigé par : Dr. Reguieg F. Zohra & Mme Madani Asma

Année Universitaire 2024-2025

DÉDICACES

Ce projet est bien plus qu'un simple travail de fin d'études. Il est le reflet d'un long chemin parcouru, souvent difficile, parsemé de luttes silencieuses contre la maladie, de doutes, mais aussi de courage, de persévérance et d'espoir.

Je tiens à dédier ce mémoire à ma famille, pour leur amour immense, leur patience inépuisable et leur soutien indéfectible tout au long de ces années éprouvantes. Sans eux, rien de tout cela n'aurait été possible.

À mes professeurs, qui ont su voir au-delà des absences et des moments de faiblesse, et qui m'ont toujours encouragée à aller au bout de mes capacités. Leur compréhension et leur bienveillance, ont été pour moi une force précieuse.

À Madame F. Zohra Reguieg, ma promotrice, pour sa présence, sa confiance et son accompagnement à la fois humain et professionnel. Vous avez été bien plus qu'un simple guide académique : un vrai repère dans la tempête.

Et enfin… à mon ordinateur, mon fidèle compagnon de galère, qui a survécu à toutes les crises de nerfs, aux plantages inopinés, aux sessions de travail interminables… et surtout à toutes les fois où j'ai sérieusement envisagé de le jeter par la fenêtre.

On l'a fait… ensemble.

Roumaissa Taouinet

Remerciements

À l'issue de ce travail de fin d'études, il m'est impossible de ne pas m'arrêter un instant, pour exprimer toute ma gratitude envers celles et ceux, qui ont été présents tout au long de ce parcours, souvent semé d'embûches mais profondément formateur.

Je tiens tout d'abord à adresser mes remerciements les plus sincères à Dr F. Zohra Reguieg, ma promotrice, une femme d'une rare humanité, qui incarne à elle seule la définition de la force, de la bienveillance et de la résilience. Elle a été bien plus qu'une simple encadrante : elle a été un véritable pilier, lorsque tout semblait s'écrouler autour de moi. Alors que les difficultés professionnelles et de santé s'accumulaient et que beaucoup de personnes, me tournaient le dos, elle a su rester présente, à l'écoute, avec patience, compréhension et compassion. Ses conseils avisés, son regard toujours juste et son soutien moral, m'ont permis de garder espoir et de continuer, même dans les moments de doute et de fatigue extrême. Merci, Madame, pour avoir cru en moi quand moi-même je n'y croyais plus.

Je souhaite également exprimer ma profonde reconnaissance à Dr. Hocine Aït Saadi, chef du département d'Electronique, pour sa compréhension, sa bienveillance et son soutien, notamment durant ma convalescence. Sa disponibilité et son aide, m'ont été d'un grand réconfort et ont permis la continuité de mon travail.

Je souhaite adresser mes remerciements particuliers, à Madame la présidente du jury, Pr. Nadjia Benblidia. Je ne l'ai pas connue personnellement, je n'ai jamais assisté à ses cours, mais son sourire, sa gentillesse et la belle image qu'elle véhicule, m'ont toujours inspirée. Rien que sa présence dans ce jury représente pour moi, une marque de respect et d'encouragement.

Je remercie aussi de tout cœur, Dr. Abderrahmane Anou, membre du jury, un enseignant que je n'oublierai jamais. C'est avec lui que j'ai découvert les communications numériques, à travers ses explications claires, ses blagues mémorables et ses conseils toujours bien placés. Il a su rendre cette matière vivante, humaine, et profondément ancrée dans ma mémoire. Aujourd'hui, c'est avec beaucoup d'émotion que je réalise qu'il est aussi là pour marquer, avec moi, la fin de ce long voyage universitaire.

Mes remerciements vont aussi à toute l'équipe de Djezzy, pour leur accueil, leur professionnalisme et les informations précieuses qui ont enrichi ce mémoire.

Je n'oublie pas l'ensemble de mes enseignants, pour la qualité de leur enseignement, leur écoute et leur bienveillance tout au long de ces années. Leur accompagnement a contribué à mon évolution académique et personnelle.

Je suis infiniment reconnaissante envers ma famille, papa et mes frères Abderahim et Lyes ma plus grande source de force et d'amour.

Un mot tout particulier pour ma mère, cette femme au cœur immense, dont l'amour inconditionnel m'a portée tout au long de ce chemin. Sa patience, son courage et sa capacité à rester forte face à mes propres moments de faiblesse ont été pour moi un exemple et un soutien quotidien. Maman, merci pour tes prières silencieuses, tes regards qui apaisent et ton dévouement sans limite. Tu as été mon refuge dans les moments d'incertitude et mon moteur quand l'envie d'abandonner me gagnait. Ce mémoire est aussi le tien.

À Papi Saïd, mon ami d'enfance et de jeunesse, pour sa sagesse et ses mots toujours réconfortants**,** et à Mamie Ouardia, pour son affection discrète mais précieuse**.**

Une pensée profonde à ma tante Zohra et à mon oncle Djamel, qui nous ont quittés trop tôt. Leur souvenir reste vivant dans mon cœur et continue de m'accompagner dans chaque étape de ma vie.

Un merci tout particulier à mon amie Nadia, dont la fidélité, l'écoute et les mots justes ont su éclairer les moments les plus sombres. Ton amitié a été une véritable bouée de sauvetage.

À toutes et à tous, du fond du cœur : merci

ملخص

في ظل النمو السريع لحركة المرور المتنقلة في الجزائر، ركّز هذا العمل على تحسين موازنة الحمل الديناميكية،، MIB، في شبكات LTE لا سيما لدى المتعامل Djezzy لمواجهة ظاهرة اختلال التوازن بين الخلايا المزدحمة والخلايا قليلة الاستخدام، تم اعتماد ستة نماذج من الشبكات العصبية الالتفافية CNN المدربة مسبقًا بهدف التنبؤ بخلل التوازن. تم تحويل مؤشرات الأداء الرئيسية KPI إلى صور، وتطبيق ضبط دقيق جزئي باستخدام خوارزمية التحسين ADAM وتقنيات تنظيم مثل Dropoutو L1/L2 لتفادي الإفراط في التعلّم. أظهرت النتائج أن نموذج Resnet50 هو الأفضل أداءً بدقة 97% و F1-score يتجاوز 93%. كما تميز كل من MobilenetV3 small و EfficientnetB0 بدقة 96% وخفة الوزن وسرعة الاستدلال، ما يجعلهما مناسبين للبيئات المحدودة. أما MobilenetV1 فقد جمع بين البساطة والدقة والسرعة. ورغم أن VGG16 حقق دقة 98%، إلا أنه أظهر علامات الإفراط في التعلّم واحتاج إلى موارد مادية كبيرة، مما يحد من استخدامه في التطبيقات الفورية. ويؤكد هذا المشروع فعالية نماذج CNN في تصنيف اختلال التوازن في شبكاتLTE.

الكلمات المفتاحية : شباكت LTE، موازنة الحمل الديناميكية MLB ، الشبكات العصبية الالتفافية المدربة CNN ، تهيئة دقيقة للنموذج ، التنبؤ الخاضع للإشراف ، KPI، تحسين الأداء.

Résumé

Dans un contexte de forte croissance du trafic mobile en Algérie, ce travail s'est intéressé à l'optimisation de l'équilibrage de charge (MLB) dans les réseaux LTE, particulièrement chez l'opérateur Djezzy. En réponse au déséquilibre entre cellules surchargées et sous-utilisées, six modèles CNN pré-entraînés ont été étudiés pour prédire les déséquilibres. Une transformation des KPIs en images a été réalisée, suivie d'un fine-tuning avec l'optimiseur Adam et des techniques de régularisation pour éviter le surapprentissage. Les résultats montrent que ResNet50 est le plus performant (exactitude de 97 %, F1-score > 93 %), suivi par EfficientNetB0 et MobileNetV3Small (96 %), qui allient précision et légèreté. MobileNetV2 se démarque par sa rapidité et son équilibre, tandis que VGG16, malgré une exactitude de 98 %, souffre de surapprentissage et d'une forte consommation de ressources. Ce projet confirme l'efficacité des CNN pour la classification supervisée des déséquilibres dans les réseaux LTE.

Mots clés : Réseaux LTE 4G, Équilibrage de charge (MLB), Réseaux de neurones convolutifs (CNN) pré-entraînés, fine tuning, Optimisation, KPI, Prédiction supervisée.

Abstract

In the context of rapid growth in mobile traffic in Algeria, this work focused on optimizing Mobility Load Balancing (MLB) in LTE networks, particularly for the operator Djezzy. To address inter-cell imbalance, where some cells are overloaded while others are underused, six pre-trained CNN models were explored to predict network load issues. KPI data were transformed into images, followed by partial fine-tuning using the Adam optimizer and regularization techniques to prevent overfitting. The results showed that ResNet50 performed best (97% accuracy, F1-score > 93%), followed by EfficientNetB0 and MobileNetV3Small (96%), which combined accuracy with low computational cost. MobileNetV2 stood out for its balance between speed and accuracy, making it suitable for embedded deployment. Despite reaching 98% accuracy, VGG16 exhibited overfitting and required significant resources, limiting its real-time applicability. This project confirms the effectiveness of CNN-based models for supervised classification, of traffic imbalance in LTE cellular networks.

Key words : LTE networks 4G, Load balancing (MLB), Pre-trained Convolutional Neural Networks (CNN), Fine tuning, Optimization, KPI, Supervised prediction.

Liste des abréviations et acronymes

2G	2nd Generation (Deuxième génération mobile)
3G	3rd Generation (Troisième génération mobile)
4G	4th Generation (Quatrième génération mobile)
5G	5th Generation (Cinquième génération mobile)
Adam	Adaptive Moment Estimation (Optimiseur d'apprentissage)
API	Application Programming Interface
ARPCE	Autorité de Régulation de la Poste et des Communications Électroniques
BSS	Base Station Subsystem
CNN	Convolutional Neural Network (Réseau de Neurones Convolutifs)
Colab	Google Colaboratory (plateforme cloud d'exécution avec GPU)
CSV	Comma-Separated Values
CQI	Channel Quality Indicator
DL	Deep Learning (Apprentissage profond)
DFT	Discrete Fourier Transform
eNodeB	evolved Node B
EPC	Evolved Packet Core
EPS	Evolved Packet System
ERAB	E-UTRAN Radio Access Bearer
Excel	Microsoft Excel (logiciel de traitement de données tabulaires)
FC	Fully Connected (couche entièrement connectée)
FDD	Frequency Division Duplex
FN	False Negative (Faux négatif)
FP	False Positive (Faux positif)
GPU	Graphics Processing Unit
GHz	Gigahertz
HSS	Home Subscriber Server
IA	Intelligence Artificielle
IFFT	Inverse Fast Fourier Transform
ImageNet	Large base d'images utilisée pour l'entraînement de modèles CNN
Kaggle	Plateforme de science des données avec exécution GPU

KPI	Key Performance Indicator (Indicateur clé de performance)
L1	Lasso Regularization
L2	Ridge Regularization
LTE	Long Term Evolution
MAOS	Maintenance and Operation System (Huawei)
MBConv	Mobile Inverted Bottleneck Convolution
MHz	Megahertz
MLB	Mobility Load Balancing
ML	Machine Learning (Apprentissage automatique)
MME	Mobility Management Entity
NBH	Normal Busy Hour
OSS	Operation Support System
OFDM	Orthogonal Frequency Division Multiplexing
OFDMA	Orthogonal Frequency Division Multiple Access
PAPR	Peak to Average Power Ratio
P-GW	Packet Data Network Gateway
PNG	Portable Network Graphics (format d'image raster)
PRB	Physical Resource Block
PRS	Performance Report System (Huawei)
QoS	Quality of Service
RBH	Real Busy Hour
ReLU	Rectified Linear Unit
ResNet50	Residual Network à 50 couches
RGB	Red Green Blue (modèle de couleur pour images)
RNN	Recurrent Neural Network
SC-FDMA	Single Carrier Frequency Division Multiple Access
SE	Squeeze-and-Excitation
SGD	Stochastic Gradient Descent
S-GW	Serving Gateway
TA	Time Advance
TDD	Time Division Duplex
TensorFlow	Framework de Deep Learning développé par Google
TTI	Transmission Time Interval
TP	True Positive (Vrai positif)
TN	True Negative (Vrai négatif)
UE	User Equipment
U2020	Huawei Unified Network Management System
VGG16	Visual Geometry Group (réseau profond à 16 couches)
YAML	Yet Another Markup Language (langage de configuration)

Z-Score	Score de normalisation statistique

Table des matières

Liste des figures

Liste des tableaux

Introduction Générale

Avec la multiplication des usages numériques, le streaming, le télétravail et les applications mobiles, les réseaux mobiles doivent répondre à une demande croissante en connectivité rapide, fiable et continue. Le réseau LTE (Long Term Evolution), standard de la 4G, repose sur une architecture cellulaire optimisée pour couvrir de larges zones grâce à la réutilisation des fréquences. Chaque cellule est prise en charge par une station de base qui assure le service pour une zone géographique donnée. Cependant un défi majeur persiste : l'équilibrage de charge entre les cellules. Dans les zones urbaines très fréquentées, certaines cellules sont rapidement saturées, surtout aux heures de pointe, tandis que d'autres, situées dans des zones moins denses, restent largement sous-utilisées. Ce déséquilibre impacte directement la Qualité de Service (QoS) : des débits réduits, une latence élevée, des coupures de connexion et une expérience utilisateur dégradée. En effet, dans les réseaux LTE, la demande en trafic data varie fortement selon les heures, les lieux et les événements. Les opérateurs observent ainsi des périodes de forte affluence, appelées '**Busy Hours',** durant lesquelles certaines cellules peuvent atteindre un niveau de charge critique, alors que d'autres restent sous-utilisées [1].

Pour détecter ces déséquilibres, les stations de base (eNodeB) surveillent différents indicateurs de charge : l'occupation des blocs de ressource physique (PRB), le nombre d'utilisateurs connectés (UE), le volume de données échangé, ou encore le niveau de sollicitation du processeur ou du réseau de transport. Ces informations sont échangées entre les cellules voisines via l'interface X2, afin d'avoir une vision globale de l'état du réseau [2].

Ce fonctionnement entre les cellules indépendantes, conduit souvent à un déséquilibre de charge **:** certaines cellules, situées dans des zones denses ou à fort usage, se retrouvent surchargées, alors que d'autres, moins sollicitées, restent sous-exploitées. Ce phénomène affecte directement la qualité de service (QoS) perçue par les utilisateurs, avec des effets tels qu'une baisse de débit, des délais plus longs ou des coupures de session.

Dans ce contexte, il devient essentiel de mettre en place des mécanismes capables de répartir intelligemment la charge entre les cellules**,** afin d'optimiser l'utilisation du réseau et d'assurer une expérience utilisateur homogène.

Traditionnellement, les opérateurs s'appuient sur des mécanismes de gestion statiques, comme le réglage manuel des seuils du 'handover' ou de la puissance d'émission. Ces

méthodes rigides et gourmandes en intervention humaine, s'adaptent mal aux environnements dynamiques et aux fluctuations imprévisibles du trafic ou de la mobilité.

D'où l'intérêt croissant pour des solutions intelligentes. L'intelligence artificielle et plus précisément, le deep learning, permet d'analyser en continu les indicateurs de performance (KPI) du réseau pour détecter automatiquement les situations de surcharge. L'objectif est de permettre au réseau de s'adapter en temps réel, notamment via des mécanismes dynamiques, comme le MLB.

Ce mémoire s'inscrit dans cette démarche. Il propose une approche fondée sur le transfert d'apprentissage, en s'appuyant sur plusieurs modèles, comme le VGG16, connu pour l'extraction hiérarchique efficace des caractéristiques visuelles et le modèle MobileNet, reconnu pour son efficacité et sa légèreté. Les données tabulaires issues du réseau LTE, sont transformées en images représentant l'état des cellules, afin d'alimenter un modèle de classification binaire "Good Case" ou "Bad Case", destiné à identifier les déséquilibres de charge.

Dans ce contexte, une classe "Good Case" désigne une situation où une action de rééquilibrage est pertinente, car un déséquilibre exploitable existe entre les cellules d'un même secteur. À l'inverse, une classe "Bad Case" correspond à un état stable ou inadapté, à l'activation du MLB. Cette classification vise à automatiser le déclenchement du mécanisme MLB, en s'appuyant sur une analyse intelligente des données du réseau cellulaire.

Le mémoire est structuré suivant quatre chapitres.

- Le premier chapitre décrit le fonctionnement du réseau LTE et les mécanismes d'équilibrage de charge, avec un zoom sur le MLB.
- Le second chapitre est consacré aux notions fondamentales sur l'intelligence artificielle, le machine learning et le deep learning.
- Le troisième chapitre consiste en la présentation de l'approche choisie, suivant les modèles utilisés et des choix méthodologiques.
- Le quatrième chapitre permet la mise en œuvre de la solution, illustre les expériences menées et les résultats obtenus.

Chapitre 1 Du réseau cellulaire à la surcharge : les défis du LTE

1.1 Introduction

Depuis leur apparition dans les années 1980, les réseaux mobiles n'ont cessé d'évoluer pour répondre à une demande croissante en mobilité et en connectivité [1, 2]. Chaque génération (1G à 4G) a introduit des innovations majeures, passant d'un modèle analogique à des réseaux tout IP capables de gérer de fortes charges de trafic.

La quatrième génération, ou LTE (Long Term Evolution), constitue une rupture avec les générations précédentes, notamment par sa nouvelle architecture radio, sa gestion dynamique des ressources et sa capacité à transmettre massivement des données.

Ce chapitre présente les principes fondamentaux du réseau LTE**,** depuis l'organisation cellulaire jusqu'aux mécanismes d'accès radio, de 'handover' et d'équilibrage de charge, afin de poser les bases techniques de notre étude.

1.2 Réseau cellulaire : organisation spatiale et rôle des cellules

Le réseau cellulaire est une architecture qui consiste à diviser l'aire de couverture, en cellules géographiques desservies par des stations de base. Chaque cellule fournit une connectivité radio aux terminaux utilisateurs, présents dans sa zone. Cette organisation permet une utilisation efficace du spectre, une continuité de service et une adaptabilité du réseau à la densité de population [2-7].

1.2.1 Réutilisation des fréquences

L'un des principes fondamentaux du réseau cellulaire, est la réutilisation des fréquences. Plutôt que d'attribuer une fréquence unique à chaque utilisateur, le spectre est partagé entre les cellules selon un schéma de réutilisation spatiale. Deux cellules suffisamment éloignées, peuvent utiliser la même bande de fréquence sans interférence significative.

Ce schéma (figure 1.1) est souvent illustré sous forme hexagonale, où chaque cellule représentant un secteur géré par un eNodeB. Ce modèle optimise la capacité globale du réseau tout en minimisant les conflits radio [2, 4].

En LTE, grâce à l'orthogonalité des signaux OFDM et aux mécanismes d'évitement d'interférence (ICIC, eICIC), la réutilisation de fréquence est généralement de 1 (toutes les cellules peuvent utiliser le même spectre), avec une gestion dynamique de la puissance et des ressources.

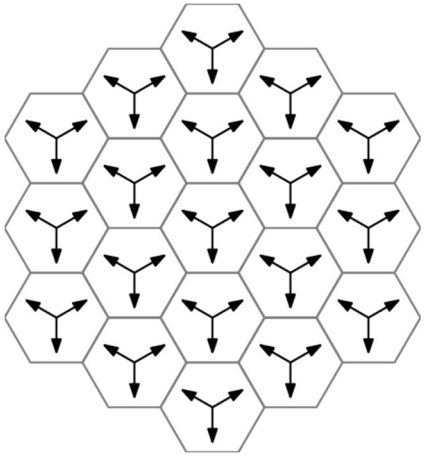


Figure 1.1 : Réseau cellulaire hexagonal typique avec eNodeB sectorisé [7]

1.2.2 Types de cellules et affectation des fréquences

Les réseaux modernes, notamment LTE, utilisent plusieurs types de cellules (tableau 1.1) pour s'adapter aux contraintes de terrain et aux besoins en capacité. Chaque type de cellule est associé à une puissance d'émission et à une bande de fréquence adaptées [5, 6].

a. **Les basses fréquences** (ex. 700 MHz) offrent une meilleure propagation et sont utilisées pour les macrocelles.
b. **Les hautes fréquences** (ex. 2.6 GHz et plus) offrent un débit supérieur mais une portée plus faible, adaptées aux micro/pico/femtocelles.

Type de cellule	Portée approximative	Puissance	Zone d'usage	Bande typique
Macrocelle	1 – 30 km	Haute	Rural / périurbain	700 MHz – 2.6 GHz
Microcelle	300 – 1000 m	Moyenne	Urbain dense	1800 MHz – 2.3 GHz
Picocelle	10 – 200 m	Faible	Bâtiments / zones piétonnes	2.3 GHz – 3.5 GHz
Femtocelle	<10 m	Très faible	Usage domestique ou entreprise	2.6 GHz / Wi-Fi / 5G mmWave

Tableau 1.1 : Types des cellules [5]

1.2.3 Architecture BSS et Co-localité

L'architecture BSS (Base Station Subsystem), utilisée dans les réseaux 2G et 3G, a été remplacée par un réseau entièrement IP dans la configuration LTE, permettant une gestion plus flexible et plus performante des connexions. En LTE, on ne parle plus de BSS, mais de 'eNodeB' (evolved Node B) qui prend en charge, à la fois les fonctions d'interface radio et de transmission de données. L'eNodeB peut être placé en co-localité avec les autres équipements réseau, permettant ainsi une gestion optimisée du trafic et une meilleure coordination entre les entités du réseau [8-10].

1.2.4 Intégration dans le réseau LTE

Dans le réseau LTE, chaque cellule (figure 1.2) est gérée par une entité appelée 'eNodeB', qui contrôle :

- L'accès radio (planification des ressources),
- La mobilité (handover, cell reselection),
- La collecte des mesures de charge (PRB, débit, KPI).

Cette structure cellulaire rend possible une densification du réseau selon les zones de trafic (small cells), tout en garantissant une continuité de connexion, grâce à des mécanismes comme le 'handover' et l'équilibrage de charge (MLB) [9, 10].

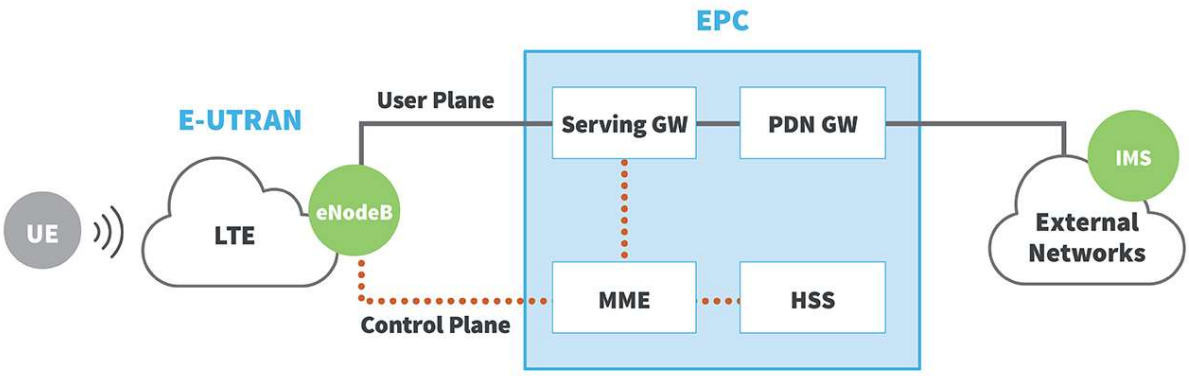


Figure 1.2 : Réseau cellulaire LTE [10]

1.3 Architecture du réseau LTE

Le Long Term Evolution (LTE) constitue une avancée déterminante dans l'évolution des réseaux mobiles.

Rompant avec les architectures traditionnelles des générations précédentes, notamment la 3G/UMTS, le LTE repose sur une infrastructure intégralement basée sur le protocole IP et introduit de nouvelles techniques d'accès radio plus performantes. Conçu pour faire face à la demande croissante en connectivité, il vise à améliorer considérablement le débit de transmission, à réduire la latence, à accroître l'efficacité spectrale et à garantir une qualité de service optimale pour les utilisateurs [10 - 12].

L'architecture LTE se compose de deux éléments principaux : 'l'Evolved Packet Core (EPC)' et 'l'Evolved Universal Terrestrial Radio Access Network (E-UTRAN)', comme illustré à la figure 1.3, ces deux sous-systèmes constituent l'Evolved Packet System (EPS) [11, 13] .

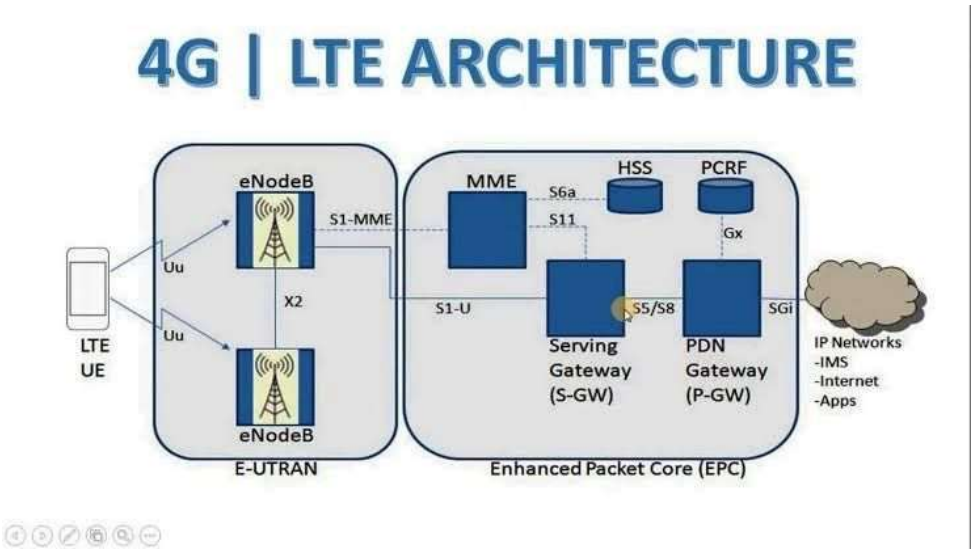


Figure 1.3 : Architecture LTE/4G [12]

La partie EPC comprend trois nœuds essentiels :

- Le Mobility Management Entity (MME) pour le plan de contrôle,
- Le Serving Gateway (S-GW) et le Packet Data Network Gateway (P-GW) pour le plan utilisateur.

Leurs fonctions respectives comprennent :

- Le Serving Gateway (S-GW) assure le routage des paquets entre l'eNodeB et le réseau cœur, dans les deux sens.
- Le Packet Data Network Gateway (P-GW) établit la connexion entre le réseau EPC et les réseaux externes, notamment Internet. Il joue également un rôle clé en attribuant les adresses IP aux équipements utilisateurs (User Equipment, UE).

- Le Mobility Management Entity (MME) est chargé de la gestion des sessions, notamment la signalisation, la négociation des qualités de service (QoS) et l'exécution des procédures de sécurité, telles que l'initiation et la négociation des mécanismes de chiffrement et de protection de l'intégrité. Il assure également la mise à jour de la position des UE.

Par ailleurs, le 'Home Subscriber Server' (HSS) constitue une base de données centralisée contenant les profils et identifiants des utilisateurs (figure 1.4). Il prend en charge l'authentification ainsi que la génération des clés de sécurité nécessaires à la protection des communications [13, 14, 15].

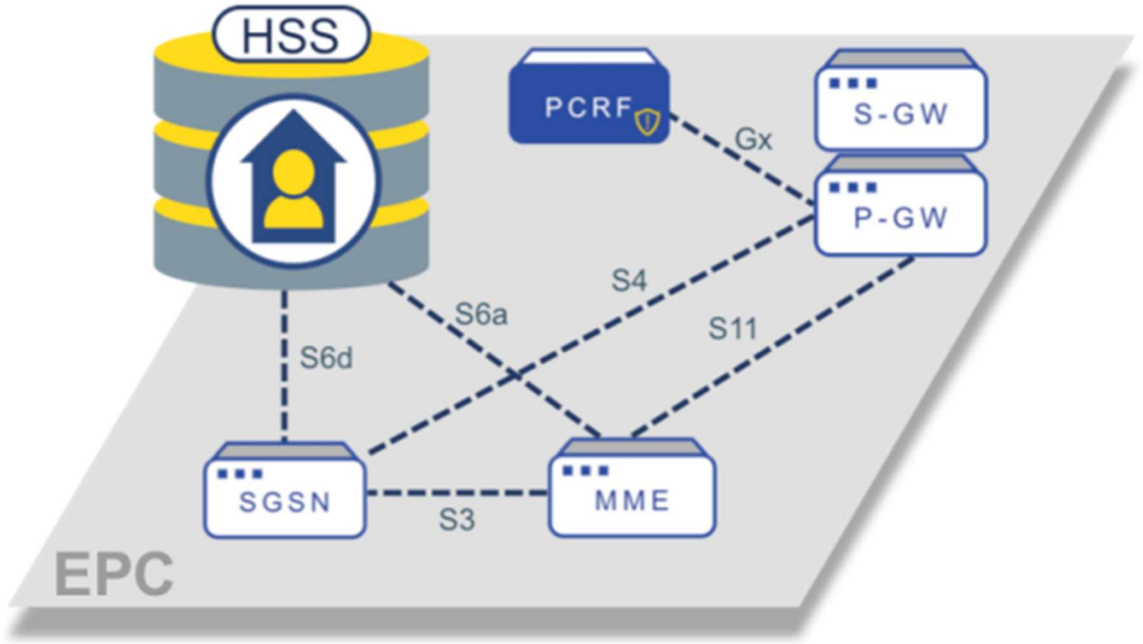


Figure 1.4 : Serveur HSS et ses connexions [15]

1.4 Fréquences utilisées

La gestion du spectre électromagnétique au sein de chaque pays, s'effectue à travers l'attribution de licences. Un nombre spécifique de ces dernières, est délivré à différents opérateurs. Dans ce cadre, les organismes intéressés par une plage particulière de fréquences, doivent acquérir auprès du gouvernement, le droit d'exploitation correspondant. La LTE utilise diverses bandes de fréquences pour la transmission des données mobiles. Ces bandes peuvent varier selon la localisation géographique et les politiques d'attribution spécifiques du pays [16, 17].

Concernant l'Algérie (tableau 1.2), les bandes utilisées sont au nombre de quatre (04). La bande 8 fonctionne avec une fréquence de 900 MHz, la bande 3 utilise 1800 MHz alors que les bandes 1 et 40 exploitent respectivement 2100 MHz et 2300 MHz [18].

Bande de fréquences	**Technologie**	**Opérateurs concernés**	**Remarques**
900 MHz	2G (GSM)	Mobilis, Djezzy, Ooredoo	Bande partagée – couverture rurale et indoor
1800 MHz	2G / 4G LTE	Mobilis, Djezzy, Ooredoo	Utilisée pour LTE (Bande 3) et GSM
2100 MHz	3G (UMTS)	Mobilis, Djezzy, Ooredoo	Dédiée aux services 3G
2600 MHz	4G LTE	Mobilis, Djezzy, Ooredoo	Bande haute pour haut débit urbain (Bande 7 LTE)
800 MHz	4G LTE	Mobilis, Djezzy, Ooredoo	Bonne couverture, propagation longue distance

Tableau 1.2 : Bandes de fréquences mobiles attribuées en Algérie [18]

En Algérie, l'Autorité de Régulation de la Poste et des Communications Électroniques (ARPCE) a officiellement lancé l'appel d'offres pour l'attribution de trois licences 5G. L'appel à concurrence indique également que la 5G utilisera des bandes de fréquences, en mid-band (~3,5 GHz), en conformité avec les pratiques internationales afin de combiner couverture régulière et haut débit [19, 20, 21]. Le lancement sera effectif au troisième trimestre de l'année 2025.

1.5 Techniques d'accès en LTE

Les techniques d'accès en LTE, se regroupent en plusieurs modulations ; OFDM, OFDMA et SC-FDMA [22-25].

1.5.1 OFDM

L'OFDM (modulation par répartition orthogonale de la fréquence) correspond à un schéma de modulation multi-porteuses où un flux de données est découpé en de nombreux sous-flux passe-bande, chacun modulé sur une sous-porteuse sinusoïdale. Les porteuses sont espacées par incréments de fréquence réguliers (15 kHz en LTE), de sorte qu'elles soient orthogonales sur la durée symbole (grâce à une transformée de Fourier inverse IFFT/IFFT sur le vecteur de données), ce qui élimine l'interférence inter-porteuse. L'ajout d'un préfixe cyclique protège contre l'interférence inter-symbole, due aux trajets multiples. Ce principe rend l'OFDM robuste aux canaux à « fading » sélectif en fréquence : chaque sous-porteuse subit une atténuation quasi-plate, simplifiant l'égalisation. En conséquence, l'OFDM offre une forte efficacité spectrale et une résistance aux multi-trajets. En revanche, il génère un rapport crête/moyenne élevé (PAPR), le signal résultant est la somme de nombreuses sinusoïdes, ce qui oblige à utiliser des amplificateurs linéaires surdimensionnés. Ce PAPR élevé est l'inconvénient majeur de l'OFDM/OFDMA. Par ailleurs, l'OFDM est sensible aux décalages en fréquence (décalage Doppler ou erreur de synchronisation) qui rompent l'orthogonalité. En LTE, l'OFDM « pur » n'est pas utilisé seul en liaison montante ; en descente il sert de base au multiplexage OFDMA [22].

1.5.2 OFDMA

L'OFDMA (Orthogonal Frequency Division Multiple Access) est l'extension multi-utilisateurs de l'OFDM. Elle permet de partager dynamiquement les ressources spectrales entre plusieurs terminaux, en découpant la bande passante LTE en blocs de ressources (Resource Blocks, RB), chacun composé de 12 sous-porteuses espacées de 15 kHz pendant 0,5 ms [23]. Chaque bloc est assigné dynamiquement à un utilisateur, en fonction de la qualité du canal radio (CQI), assurant ainsi une allocation flexible et adaptative des ressources spectrales.

Chaque RB transmet plusieurs symboles OFDM en parallèle sur des sous-porteuses orthogonales [25]. Ce multiplexage combiné en fréquence et en temps permet d'améliorer l'efficacité spectrale globale du réseau et d'offrir des débits élevés, particulièrement en liaison descendante (downlink).

Une trame LTE FDD typique en OFDMA est divisée selon deux axes :

a. **Axe fréquentiel** : des blocs de 12 sous-porteuses (180 kHz),
b. **Axe temporel** : sur une durée de 1 ms.

On y retrouve les canaux de synchronisation (P-SS, S-SS), le canal de diffusion (PBCH), les canaux de contrôle (PCFICH, PHICH, PDCCH) ainsi que les canaux de données utilisateurs (PDSC H). En liaison descendante**,** l'OFDMA est préférée, car l'eNodeB possède une puissance d'émission suffisante, pour tolérer un rapport crête/moyenne (PAPR) élevé. Ce qui permet d'atteindre des débits jusqu'à 100 Mb/s sur une bande de 20 MH z [25].

Parmi les avantages de l'OFDMA, on peut citer :

- Une allocation fine des ressources dans les deux dimensions (temps/fréquence),
- Une meilleure efficacité spectrale,
- Une bonne robustesse aux interférences grâce à l'orthogonalité des sous-porteuses.

Cependant, les limites de l'OFDMA [8] sont similaires à celles de l'OFDM :

- Un PAPR élevé,
- Une complexité accrue du scheduling (ordonnancement des ressources),
- Une forte sensibilité aux erreurs de synchronisation.

Enfin, contrairement à certains systèmes comme WiMAX, l'OFDMA n'est pas utilisée en liaison montante (uplink) dans le LTE. Cela s'explique par la consommation énergétique élevée induite par le PAPR, que les terminaux mobiles à batterie ne peuvent efficacement supporter. À la place, le LTE emploie le SC-FDMA, plus adapté à l'émission des UE [24].

1.5.3 SC-FDMA (Single Carrier Frequency Division Multiple Access)

C'est la technique de transmission utilisée en liaison montante (uplink) dans LTE. Elle repose sur un schéma de modulation similaire à l'OFDM, mais avec un pré-codage DFT appliqué en amont (figure 1.5).

Concrètement, une transformée de Fourier discrète (DFT) est appliquée aux symboles QAM d'entrée, suivie d'une IFFT classique pour générer le signal temporel OFDM [22]. Ce processus de pré-codage permet d'obtenir un signal mono-porteuse sur le plan temporel, d'où le nom "Single Carrier'' tout en conservant les avantages spectraux et la robustesse aux trajets multiples de l'OFDM [24].

L'un des principaux bénéfices du SC-FDMA réside dans la réduction significative du rapport crête à moyenne (PAPR), ce qui permet d'utiliser des amplificateurs plus efficaces énergétiquement, un point critique pour les équipements terminaux alimentés par batterie. En revanche, cette technique impose que les sous-porteuses allouées à un même utilisateur soient contiguës (allocation localisée), ce qui complique la planification des ressources et limite la diversité fréquentielle dans certains cas. Cela peut entraîner une légère dégradation des performances en conditions de fading [4]. Malgré cela, le PAPR réduit reste un critère déterminant : c'est pourquoi le SC-FDMA est privilégié en uplink, tandis que l'OFDMA, plus tolérant au PAPR, est utilisé en downlink, où la station de base (eNodeB) dispose de plus grandes capacités d'émission [26, 27]. Ainsi, le choix du SC-FDMA en liaison montante illustre une optimisation énergétique, visant à prolonger l'autonomie des terminaux, tout en assurant une transmission robuste dans un environnement radio complexe.

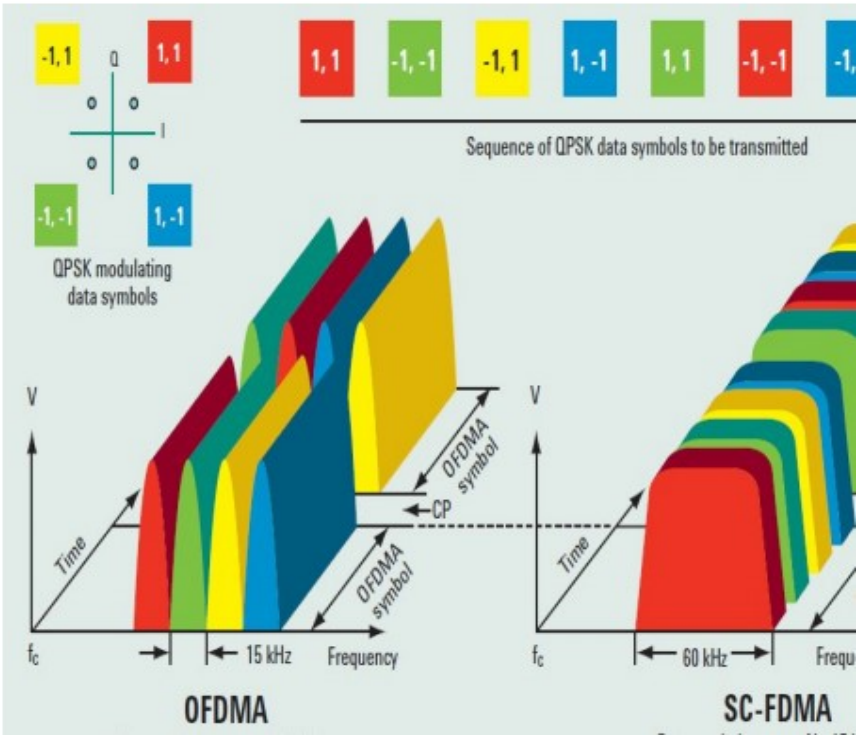


Figure 1.2 : Comparaison entre OFDMA et SC-FDMA : représentation des blocs de ressources en temps-fréquence [26]

1.6 Techniques de multiplexage

Les techniques de multiplexage (figure 1.6) peuvent être de type FDD (Frequency Division Duplex) ou TDD (Time Division Duplex) [22, 28-29].

1.6.1 FDD

Le mode FDD repose sur une séparation en fréquence entre la liaison montante (uplink) et la liaison descendante (downlink)**,** permettant une transmission simultanée dans les deux directions. Cette technique est bien adaptée aux communications à débit symétrique et assure une faible latence.

Dans le cadre du LTE, le FDD est la configuration la plus répandue, notamment utilisée dans les bandes 900 MHz (B8) et 1800 MHz (B3). Elle est privilégiée par les opérateurs pour sa stabilité, sa compatibilité avec les équipements existants et sa meilleure couverture en milieu rural ou semi-urbain [28].

1.6.2 TDD

Le mode TDD**,** quant à lui, utilise une seule bande de fréquence pour les deux directions de communication, mais les sépare dans le temps. Ainsi, une partie des slots est allouée au downlink, et l'autre à l'uplink. Ce mode est particulièrement adapté aux environnements à trafic asymétrique, car il permet de modifier dynamiquement le rapport UL/DL selon la charge du réseau [22]. Cette flexibilité temporelle est un atout dans les zones urbaines à forte demande en téléchargement, comme le streaming ou la navigation web.

Chez Djezzy, l'opérateur algérien, le mode TDD est utilisé sur la bande des 2300 MHz (B40) pour fournir une capacité accrue en environnement dense**.** Cette bande, couplée à un découpage temporel adapté, permet de mieux absorber les pics de trafic et d'optimiser l'expérience de l'utilisateur [29].

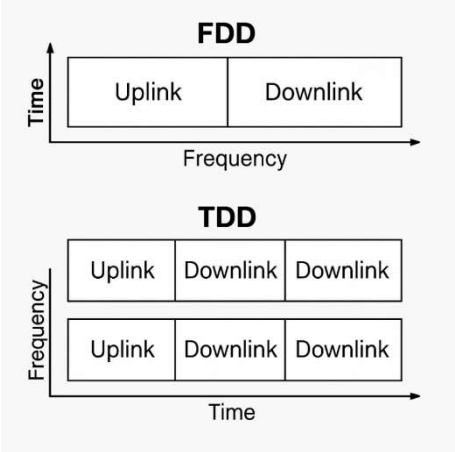


Figure 1.6 : Représentation simplifiée des modes duplex LTE : FDD & TDD [29]

Avec,

- **FDD** : séparation en fréquence de l'uplink et du downlink.
- **TDD** : alternance temporelle entre uplink et downlink sur la même bande.

1.7 Gestion des ressources radio en LTE

Dans un réseau LTE, la bande passante est divisée en Physical Resource Blocks (PRB)**,** chacune couvrant 180 kHz de spectre sur une durée de 0,5 ms [22, 25]. Ces blocs forment une grille temps-fréquence au sein de laquelle, l'eNodeB (station de base) distribue dynamiquement les ressources aux utilisateurs connectés [30, 31]. Cette découpe fine permet de répondre en temps réel aux variations de la qualité du canal radio et, aux besoins de chaque service (voix, vidéo, navigation web).

1.7.1 Fonctionnement du scheduler

Le scheduler de l'eNodeB prend ses décisions à chaque intervalle de transmission (un TTI de 1 ms). Pour cela, il collecte les rapports de qualité de canal (CQI**)** envoyés périodiquement par les équipements utilisateurs, ainsi que l'état de leurs buffers (quantité de données en attente). En combinant ces informations, le scheduler choisit :

a. **Qui** recevra des PRB (sélection des utilisateurs actifs) ;
b. **Combien** de PRB seront attribués à chaque utilisateur ;
c. **Où** dans la grille temps-fréquence ces PRB seront placés.

Grâce à cette allocation millimétrée, l'eNodeB peut adapter la transmission à la qualité du lien et optimiser l'usage global du spectre, tout en respectant les exigences de délai et de débit de chaque service [30].

1.7.2 Stratégies d'ordonnancement

Plusieurs politiques d'ordonnancement [30, 31] coexistent dans LTE :

a. **Round Robin** : chaque utilisateur reçoit des ressources à tour de rôle, assurant une répartition simple et équitable, mais sans tenir compte de la qualité du canal.
b. **Proportional Fair** : équilibre entre débit maximal et équité ; il favorise les utilisateurs bénéficiant d'un bon canal, tout en garantissant une part de ressources à ceux moins favorisés.

c. **QoS-aware scheduling** : les flux à débit garanti (voix, vidéo en temps réel) ou de haute priorité, sont servis en priorité, au détriment des services best-effort (navigation web, téléchargement de fichiers).

Ces algorithmes sont normalisés par le 3GPP pour assurer une interopérabilité et une qualité de service prévisible, quel que soit le fournisseur d'équipement (Huawei, Ericsson, Nokia…) ou l'opérateur (Orange, Djezzy, etc.) [31].

1.7.3 Impact sur la gestion de la charge

Lorsque la cellule approche de la saturation (tous les PRB sont attribués, où l'eNodeB reçoit de nombreuses demandes). Le scheduler peut signaler un niveau de charge élevé au module de contrôle d'admission. Dans ce cas, le réseau a plusieurs options :

a. **Refuser certaines nouvelles sessions** pour préserver la QoS des connexions actives.
b. **Effectuer un handover** : déplacer un utilisateur vers une cellule voisine moins chargée.
c. **Lancer un mécanisme de Mobility Load Balancing (MLB)** : ajuster finement les paramètres de mobilité pour répartir la charge de façon plus homogène.

Cette gestion proactive des ressources radio est essentielle pour maintenir la continuité de service et la qualité perçue par l'utilisateur, même en cas de trafic intense ou de conditions radio défavorables [22].
En Algérie, le réseau LTE sert principalement aux services de données. L'opérateur Djezzy, qui constitue le centre de notre analyse, fait face à une demande croissante de trafic, largement dominée par le streaming vidéo, les réseaux sociaux, la navigation sur internet et diverses applications mobiles. Ce trafic présente une forte variabilité en fonction de l'heure, des jours ou encore des événements locaux, ce qui peut entraîner une surcharge de certaines cellules tandis que d'autres demeurent sous-exploitées [2].

1.8 Gestion du trafic en Algérie

En Algérie, le réseau LTE sert principalement aux services de données. L'opérateur Djezzy, qui constitue le centre de notre analyse, fait face à une demande croissante de trafic, largement dominée par le streaming vidéo, les réseaux sociaux, la navigation sur internet et diverses applications mobiles [1].

Ce trafic présente une forte variabilité en fonction de l'heure, des jours ou encore des événements locaux, ce qui peut entraîner une surcharge de certaines cellules, tandis que d'autres demeurent sous-exploitées.

1.8.1 Saturation

La saturation se produit lorsqu'une cellule, atteint ou dépasse ses capacités en ressources radio, affectant directement la qualité de service (QoS) [6, 32]. Cette dégradation peut se manifester par :

- Un débit moyen par utilisateur très faible,
- Une augmentation du taux de blocage ou d'échec de connexion,
- Une élévation du temps de latence,
- Une réduction de la couverture effective.

Le trafic data dans un réseau mobile varie considérablement en fonction de plusieurs facteurs, notamment l'heure de la journée, la saison, la densité de population, les zones géographiques ou encore des événements particuliers. Pour évaluer et anticiper les situations critiques, les opérateurs comme Djezzy, s'appuient sur deux indicateurs majeurs : le Real Busy Hour (RBH) et le Normal Busy Hour (NBH) [33, 34, 35].

- Le Real Busy Hour (RBH) correspond à l'heure réelle du trafic maximal, calculé automatiquement pour chaque cellule, chaque jour. Il reflète la charge maximale effectivement enregistrée sur la base de métriques, telles que le volume de données, le nombre d'utilisateurs simultanés, ou l'occupation des PRB. Cette mesure est essentielle pour la planification dynamique des ressources radio, l'analyse de surcharge et l'activation de mécanismes d'automatisation tels que le Mobility Load Balancing (MLB).
- Le Normal Busy Hour (NBH), en revanche, est une fenêtre horaire fixe, généralement choisie selon les tendances historiques du réseau. Par exemple, chez Djezzy, le paramètre NBH est souvent fixé entre 18h et 19h, car cette période concentre une activité utilisateur élevée, dans la majorité des régions. Cette heure permet des analyses comparables et stables entre les différentes périodes et sites, en évitant les fluctuations extrêmes observées dans le paramètre RBH.

Par exemple, à Djezzy, les heures de pointe varient d'une cellule à l'autre, mais sont souvent observées le soir, entre 19h et 22h, particulièrement dans les zones urbaines.
Certains événements génèrent des pics soudains de trafic, comme les matchs de football, les fêtes religieuses ou encore les périodes de promotions commerciales. Ces pics sont généralement plus intenses dans les quartiers résidentiels et les centres urbains, où les abonnés sont très connectés. De plus, le trafic saisonnier est un facteur important : pendant l'été, de nombreuses régions côtières enregistrent une surcharge du réseau en raison de l'afflux touristique, nécessitant souvent une reconfiguration temporaire des paramètres radio pour maintenir la qualité de service [36].

1.8.2 Impact de la densité et de la mobilité

La densité de population est un facteur déterminant du comportement du trafic. En zone urbaine dense, les cellules sont surchargées à certains moments de la journée.
En revanche, en zone rurale, bien que le nombre d'abonnés soit plus faible, le rayon de couverture étant plus large, une cellule peut très vite saturer en cas d'événement ponctuel.
La mobilité est également un facteur à ne pas négliger. Lors des déplacements (domicile-travail, week-ends, voyages estivaux), les utilisateurs changent régulièrement de cellule, générant de nombreux handovers. Cela complique la gestion de la charge et crée des pics localisés, notamment le long des axes routiers et dans les gares [8, 9].
C'est dans ce contexte que l'usage d'algorithmes d'équilibrage de charge comme le Mobility Load Balancing (MLB) prend tout son sens, car ils permettent de mieux répartir les utilisateurs entre les cellules, améliorant ainsi l'efficacité globale du réseau LTE [37].

1.9 Handover

Le handover (transfert intercellulaire) est essentiel en LTE, pour garantir la continuité du service lors du déplacement d'un UE d'une cellule à une autre. LTE utilise un hard handover**,** où la liaison avec la cellule source, est coupée avant d'établir celle avec la cellule cible, minimisant cependant la durée d'interruption du service [10, 32].

1.9.1 Types de Handover en LTE

Plusieurs types de handover sont présentés dans ce cadre [10].

a. **Handover intra-eNodeB** : l'UE change de secteur ou de bande, au sein du même eNodeB, sans coordination cœur de réseau, ce qui simplifie le processus.
b. **Handover inter-eNodeB** : Dans ce cas, l'UE passe d'une cellule gérée par un eNodeB à une cellule gérée par un autre eNodeB.
c. La coordination entre les deux eNodeB s'effectue via l'interface X2. Ce type de handover est plus complexe, car il nécessite une coordination entre les différents éléments du réseau.

1.9.2 Processus de Handover en LTE

Le processus de handover en LTE [10, 38] suit généralement les étapes ci-dessous :

1. **Mesure** : L'UE effectue des mesures de la qualité du signal des cellules voisines et envoie des rapports de mesure à l'eNodeB source.
2. **Décision** : L'eNodeB source décide de déclencher un handover, en fonction des rapports de mesure et des conditions réseau.
3. **Préparation** : L'eNodeB source envoie une requête de handover à l'eNodeB cible, via l'interface X2. L'eNodeB cible prépare les ressources nécessaires pour accueillir l'UE.
4. **Exécution** : L'eNodeB source envoie un message de reconfiguration de la connexion RRC à l'UE, l'informant du handover. L'UE synchronise alors avec la cellule cible.
5. **Achèvement** : Une fois connecté à la cellule cible, l'UE envoie un message de reconfiguration de la connexion RRC complété à l'eNodeB cible. L'eNodeB cible informe ensuite l'eNodeB source de la réussite du handover.

Ce processus est conçu pour être rapide et transparent pour l'utilisateur, minimisant ainsi les interruptions de service.

1.10 Qualité de service (QoS)

La qualité de service (QoS) se définit comme la capacité d'un réseau à offrir des performances satisfaisantes pour différents types de services (voix, vidéo, données), selon des critères mesurables tels que le débit**,** la latence, le jitter (variation du délai), le taux de perte de paquets et l'occupation PRB [2, 39]. Plusieurs paramètres techniques sont pris en compte :

a. **Le débit** (throughput) indique le volume de données transférées par seconde ;
b. **La latence** mesure le délai entre l'émission et la réception d'un paquet ;
c. **Le jitter** duplique la différence entre les délais individuels des paquets ;
d. **Le taux de perte** évalue la proportion de paquets perdus en transit.

e. **Le taux d'occupation PRB** mesure l'intensité de sollicitation d'une cellule.

1.11 Mobility Load Balancing

Le **Mobility Load Balancing (MLB)** est un mécanisme SON (Self-Organizing Network) standardisé par la 3GPP dans le 3GPP TS 36.300, pour répartir dynamiquement la charge entre les cellules voisines et garantir ainsi, une utilisation optimale des ressources radio.

Il repose sur l'échange d'indicateurs de charge (figure 1.7), l'évaluation périodique de l'état des cellules et l'ajustement des paramètres de mobilité (re-sélection et handover), afin de corriger les déséquilibres de trafic en temps réel [22, 31, 40].

1.11.1 MLB en mode veille

En mode idle, le MLB utilise la **ré-sélection de cellule,** pour inciter les UEs inactifs à se rattacher aux cellules moins chargées.

L'eNodeB modifie les **priorités de re-sélection** (Cell Reselection Priority, CRP), ou les **offsets** de puissance relative (Cell Individual Offset, CIO) transmis dans les messages RRC (*System Information Block)*, afin de guider le choix de cellule des UEs sans interruption de service. Cette méthode, non intrusive, permet un équilibrage progressif mais efficace sur les plages de signalisation idle, avec un impact minimal sur l'expérience utilisateur.

1.11.2 MLB en mode connecté

Pour les UEs déjà en liaison active (RRC Connected), le MLB met en œuvre des handovers basés sur la charge (HO-MLB). Le contrôleur MLB ajuste dynamiquement les paramètres de décision de handover (A3 event offsets, hysteresis, time-to-trigger), en fonction du niveau de charge des cellules source et cible, mesuré en pourcentage de PRB utilisés ou d'autres KPI (ERAB drop, RSRQ). Cette approche permet un équilibrage instantané et ciblé, mais exige une coordination fine pour éviter les phénomènes de ping-pong ou de handover faileurs.

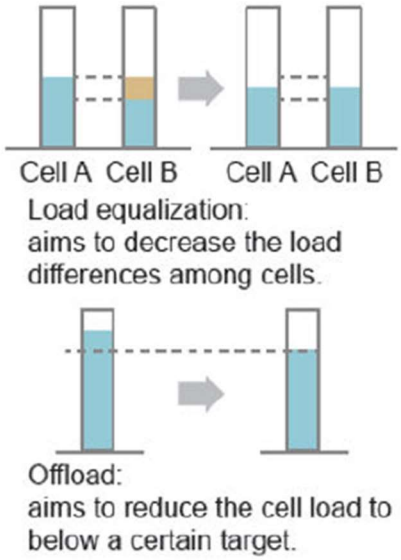


Figure 1.7 : Comparaison entre l'équilibrage de charge et l'offload [22]

1.11.3 MLB inter-fréquence

Lorsqu'un opérateur dispose de plusieurs bandes LTE (intra-RAT inter-frequency), le MLB inter-fréquence étend les mécanismes précédents pour répartir les UEs entre les différentes fréquences (figure 1.8). Il s'appuie sur la **gestion multi-carrier**, où les UEs sont redirigés via des handovers inter-frequency ou des re-sélections cell-to-frequency, en ajustant les **priorités inter-frequency** et en échangeant des rapports de charge inter-cellulaires via X2/S1 AP. Cette stratégie améliore l'utilisation spectrale globale et soulage les fréquences saturées au profit de celles sous-utilisées [22-31].

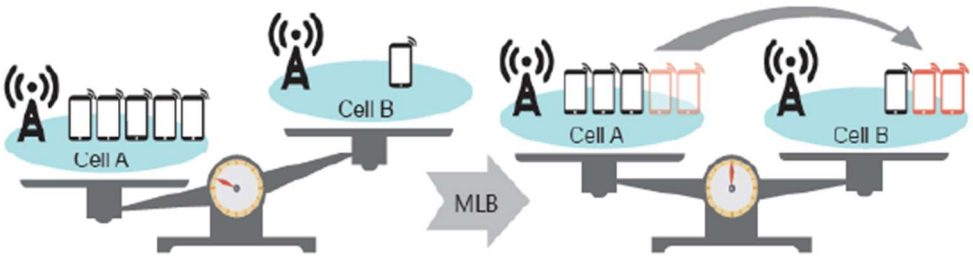


Figure 1.8 : Exemple de l'intra-RAT mobility load balancing [31]

1.11.4 Paramètres clés

Les principaux paramètres configurables du MLB [40] incluent :

a. **Mlb Trigger Mode** : c'est la condition de déclenchement (seuil PRB, nombre d'UEs, KPI QoS).
b. **CIO et CRP** : ce sont les offsets de handover et priorités de re-sélection (idle), pour orienter le trafic.
c. **Time-to-trigger (TTT)** : c'est le délai avant l'exécution du handover MLB, pour éviter les changements trop fréquents.
d. **Dynamic Threshold Enable** : c'est l'activation de seuils adaptatifs pour suivre l'évolution du trafic et des heures de pointe.

1.13 Conclusion

Ce premier chapitre a permis de poser les bases techniques indispensables à la compréhension du fonctionnement des réseaux LTE. Nous avons exploré l'organisation cellulaire, l'architecture du réseau, les techniques d'accès radio, ainsi que les mécanismes de gestion des ressources et de mobilité.

L'accent a également été mis sur les indicateurs de performance (KPI) et la qualité de service (QoS), qui jouent un rôle crucial dans le maintien de la performance réseau. Enfin, le principe du Mobility Load Balancing (MLB) a été présenté comme une solution clé pour répondre aux problèmes de surcharge.

Ce socle théorique est essentiel pour aborder dans les chapitres suivants, l'étude conceptuelle et la mise en œuvre d'approches intelligentes d'optimisation du trafic dans les réseaux LTE.

Le second chapitre est consacré aux notions d'intelligence artificielle et à l'apprentissage profond.

Chapitre 2 Notions d'Apprentissage automatique et profond

2.1 Introduction

Depuis les débuts de l'informatique, les chercheurs ont nourri l'ambition de doter les machines d'une forme d'intelligence capable d'imiter, voire de surpasser, certaines facultés humaines telles que la perception, le raisonnement ou la prise de décision. Cette quête a donné naissance à l'intelligence artificielle (IA), un domaine en constante évolution qui s'est progressivement affranchi des approches symboliques traditionnelles, pour intégrer des méthodes d'apprentissage automatique plus souples et performantes. L'apparition du machine learning, puis du deep learning, a marqué une rupture décisive : les systèmes ne sont plus uniquement programmés pour agir selon des règles définies, mais sont capables d'apprendre eux-mêmes, à partir des données. Ces progrès ont permis le développement d'applications de plus en plus complexes, dans des domaines variés tels que la vision par ordinateur, le traitement du langage naturel ou encore l'optimisation des réseaux. Comme le soulignent Russell et Norvig [41], l'objectif central de l'IA reste « de concevoir des agents capables d'agir intelligemment dans leur environnement ».

Dans ce chapitre, nous présenterons les concepts fondamentaux de l'intelligence artificielle, en détaillant les principales approches d'apprentissage, les architectures neuronales et les mécanismes avancés tels que l'apprentissage par transfert et le fine tuning. L'objectif est de fonder un socle théorique, nécessaire à la compréhension de l'étude conceptuelle et de la mise en œuvre, qui seront abordées dans les chapitres suivants.

2.2 Introduction à l'intelligence artificielle

L'intelligence artificielle (IA) est une branche de l'informatique dont l'objectif est de concevoir des systèmes capables de simuler certains comportements humains, tels que le raisonnement, l'apprentissage, la perception ou encore la prise de décision. Selon la norme ISO/IEC 2382-28 [42], l'IA est définie comme la « capacité d'une unité fonctionnelle à exécuter des fonctions associées à l'intelligence humaine, telles que le raisonnement et l'apprentissage ».

a. Classification de l'IA selon la typologie de Russel et Norvig

Dès les débuts de la discipline, plusieurs visions de l'IA ont émergé. Selon la typologie proposée par Russell et Norvig [41], on distingue quatre grandes catégories :

- **Les systèmes qui pensent comme les humains**, selon l'approche cognitive. Dans ce cadre, en 1978 Bellman [43] parle de « l'automatisation d'activités que nous associons à la pensée humaine ».
- **Les systèmes qui agissent comme les humains**, c'est-à-dire capables d'imiter un comportement intelligent observable. En 1990, Kurzweil [44] définit l'IA comme « l'art de créer des machines capables de prendre en charge des fonctions exigeant de l'intelligence, quand elles sont réalisées par des humains ».
- **Les systèmes qui pensent rationnellement**, en s'appuyant sur la logique formelle. En 1985, Charniak et McDermott [45] définissent cette approche comme « l'étude des facultés mentales à travers des modèles informatiques ».
- **Les systèmes qui agissent rationnellement**, sont représentés par une approche qui se concentre sur la capacité d'un agent à maximiser son efficacité dans un environnement donné (approche des agents intelligents).

b. Classification de l'IA suivant le niveau de raisonnement

Il s'agit d'une classification **conceptuelle** de l'IA, selon son niveau de raisonnement [46, 47, 48].

1. **IA étroite :** L'IA étroite, également appelée IA faible, est conçue pour effectuer des tâches spécifiques dans un domaine limité.

C'est la forme la plus courante d'IA utilisée aujourd'hui, alimentant des applications telles que les assistants virtuels, les systèmes de reconnaissance d'images et les algorithmes de recommandation. Elle ne peut pas généraliser ses connaissances à des domaines différents : C'est **l'IA actuelle.**

2. **IA générale :** C'est une IA capable de raisonner, apprendre, comprendre et s'adapter à n'importe quelle tâche intellectuelle humaine, avec un niveau de performance équivalent ou supérieur à celui d'un humain. Cette IA **n'existe pas encore**, elle est à ce stade, encore hypothétique.
3. **Super intelligence :** Elle dépasse l'intelligence humaine dans tous les domaines, scientifiques, artistiques, émotionnels, sociaux, etc. Elle est purement théorique à ce stade.

L'intelligence artificielle est aujourd'hui au cœur de nombreuses applications : la reconnaissance d'écriture ou d'images, la traduction automatique, le diagnostic médical assisté, les systèmes de recommandation, ou encore l'optimisation des ressources dans les réseaux de télécommunications. Ces exemples montrent que l'IA n'est plus un simple concept théorique, mais une technologie transversale aux implications pratiques majeures [48].

2.3 Introduction à l'apprentissage automatique

L'apprentissage automatique ou le Machine Learning (ML) ou est une branche de l'intelligence artificielle, qui regroupe des méthodes permettant aux machines d'apprendre à partir de données, sans être explicitement programmées pour chaque tâche. Le principe fondamental est de construire un modèle statistique à partir d'un jeu de données d'apprentissage, puis de l'utiliser pour faire des prédictions ou classer de nouvelles données [49-52].

2.3.1 Apprentissage supervisé

Dans ce cadre, le modèle apprend à prédire une sortie à partir d'une entrée, en s'appuyant sur un ensemble de données étiquetées.

L'objectif est de minimiser l'erreur entre les prédictions et les sorties attendues. C'est le paradigme le plus utilisé en classification et en régression.

2.3.2 Apprentissage non supervisé

Dans ce cadre, les données ne sont pas annotées. L'algorithme cherche à découvrir des structures sous-jacentes, comme des regroupements (clustering) ou des corrélations, sans guidance extérieure. C'est le cas des algorithmes comme K-means, l'analyse en composantes principales (PCA) ou les auto-encodeurs.

2.3.3 Apprentissage semi supervisé

Ce type d'apprentissage, combine l'apprentissage supervisé et non supervisé, en utilisant à la fois des données étiquetées et non étiquetées pour entraîner des modèles d'intelligence artificielle (IA) sur des tâches de classification et de régression.

2.3.4 Apprentissage par renforcement

Dans cette approche, un agent apprend à interagir avec un environnement. À chaque action, il reçoit une récompense ou une punition, où son but est de maximiser une fonction de récompense cumulative. Ce type d'apprentissage est particulièrement utilisé dans les jeux, la robotique ou la gestion dynamique des ressources.

2.4 Introduction aux réseaux de neurones fondamentaux

Les réseaux de neurones artificiels, sont inspirés du fonctionnement du cerveau humain, suivant les neurones biologiques.

2.4.1 Neurone biologique

Le neurone est une cellule composée d'un corps cellulaire et d'un noyau. Le corps cellulaire se ramifie pour former ce que l'on nomme les dendrites. Celles-ci sont parfois si nombreuses, que l'on parle alors d'arborisation dendritique. C'est par les dendrites que l'information est acheminée de l'extérieur vers le soma, corps du neurone [53, 54].

L'information traitée par le neurone chemine ensuite le long de l'axone, pour être transmise aux autres neurones. La jonction entre deux neurones est réalisée par la synapse (figure 2. 1).

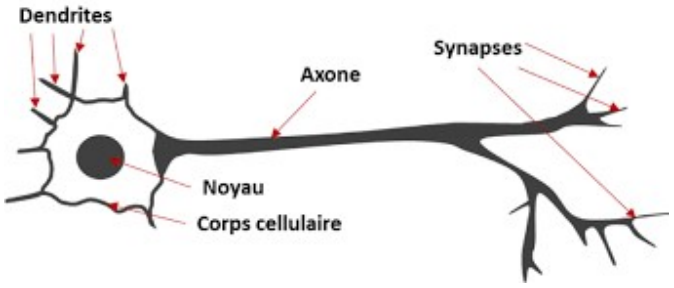


Figure 2.1 : Représentation d'un neurone biologique [53]

2.4.2 Neurone artificiel

Le neurone artificiel (figure 2.2) est un modèle mathématique simplifié du neurone biologique. Ils sont conçus pour recevoir une ou plusieurs entrées numériques x_m pour les pondérer par des coefficients appelés poids w_m et produire une sortie $\hat{y}_1$ (2.1) en appliquant une fonction d'activation non linéaire f, à la somme pondérée des entrées. La sortie de chaque neurone est ensuite transmise à d'autres neurones de la couche suivante, ce qui permet aux réseaux de neurones, de modéliser des relations complexes entre les données [53].

$$\hat{y} = f\left(b + \sum_{i=1}^{m} W_i E_i\right) \quad 2.1$$

Où, b représente le biais.

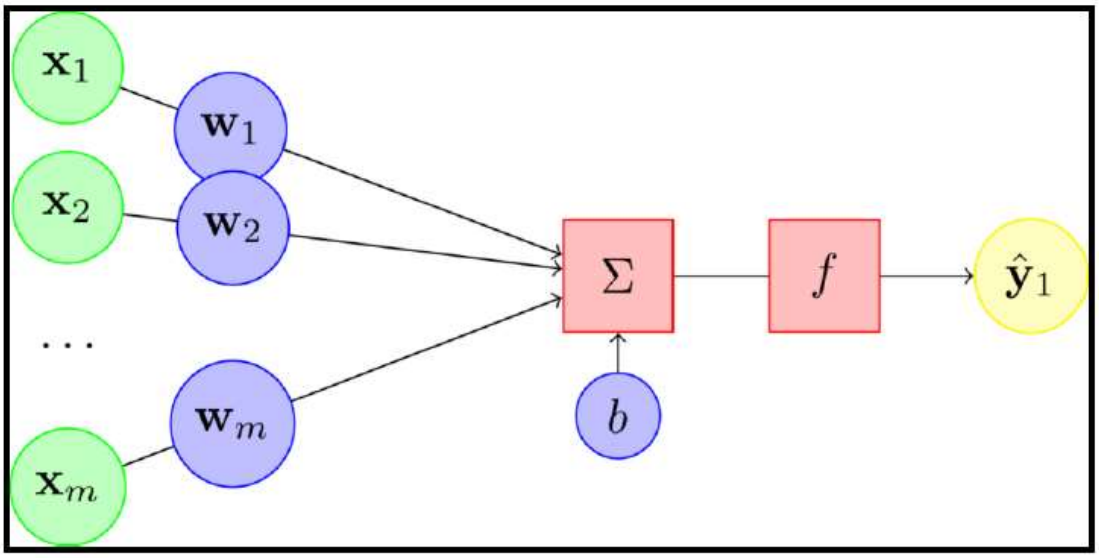


Figure 2.2 : Réseau de neurones artificiel simple [53]

Ce réseau simple correspond au perceptron monocouche, un réseau à propagation avant, composé de neurones à seuil seulement, avec deux couches (entrée et sortie) entièrement interconnectées [53, 54].

2.4.3 Perceptron multicouche

Dans ce réseau (figure 2.3), les neurones de la première couche reçoivent toutes les informations entrées, ceux de la deuxième reçoivent toutes les sorties des neurones de la première couche, et ainsi de suite jusqu'au neurone de sortie qui reçoit celles de la dernière couche [53, 55] .

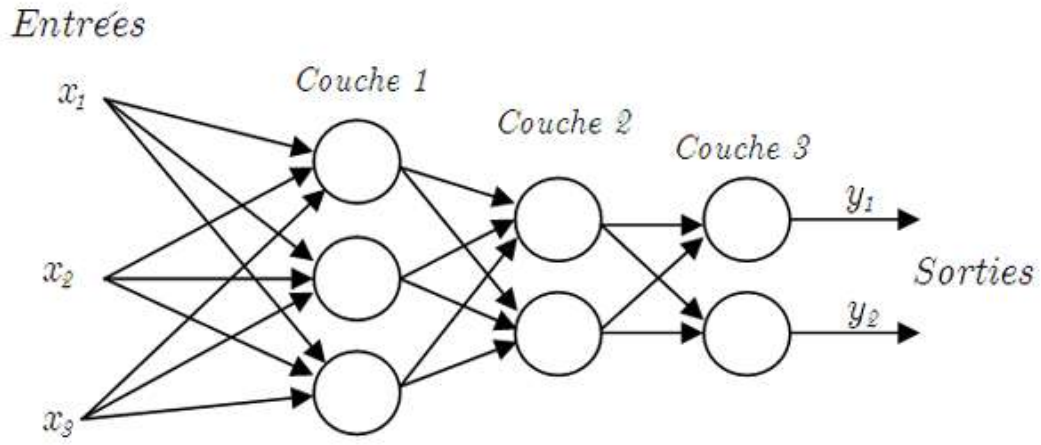


Figure 2.3 : Perceptron multicouche [53]

2.4.4 Fonctions d'activation

Une fonction d'activation est un élément fondamental des réseaux de neurones artificiels. Utilisée pour introduire de la non-linéarité dans le modèle, elle permet de déterminer si un neurone doit être activé ou non, selon les entrées. Plusieurs fonctions d'activation réalisent le rôle cité [49, 53].

a. Fonction rampe

C'est une fonction linéaire, où la sortie correspond à l'entrée. L'équation 2.2, représente la fonction rampe.

$$S = f(x) = x, \forall x \qquad 2.2$$

b. Fonction identité

La fonction échelon est exprimée par la relation 2.3. La sortie ne prend que deux valeurs : 0 pour un signal négatif et 1 pour un signal positif.

$$f(x) = \begin{cases} 1 \; si \; x \geq 0 \\ 0 \; si \; x < 0 \end{cases} \qquad 2.3$$

c. Fonction Sigmoïde

C'est une fonction non-linéaire, dérivable, continue et symétrique par rapport à l'axe des y (figure 2.4). Elle produit une courbe sigmoïde f(x) définie par la formule 2.4.

$$f(x) = \frac{1}{1+e^{-x}} \qquad 2.4$$

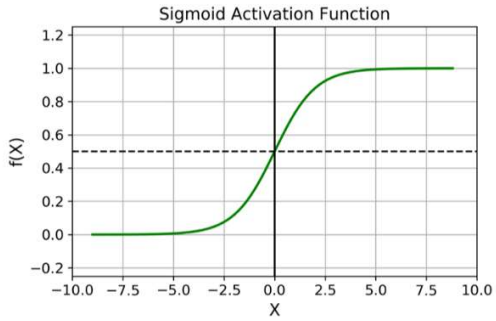


Figure 2.4 : Fonction sigmoïde [49]

d. Fonction tangente hyperbolique

Similaire à la fonction sigmoïde (figure 2.5), Tanh a une forme de S (sigmoïde), mais associe les valeurs d'entrée à une plage comprise entre -1 et 1. La seule différence entre cette fonction et la fonction sigmoïde est que Tan(h) est centrée sur le zéro.

Elle est exprimée par l'équation 2.5.

$$tanh(x) = \frac{1-e^{-2x}}{1+e^{-2x}} \qquad 2.6$$

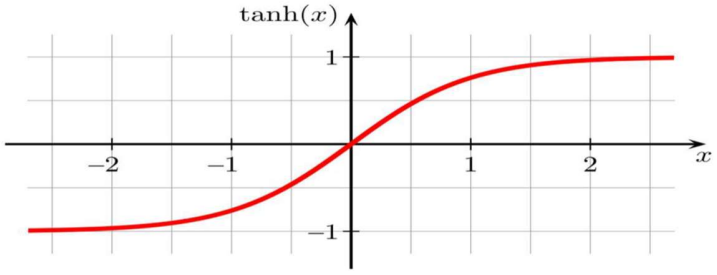


Figure 2.5 : Tangente hyperbolique [49]

e. Fonction softmax

La fonction Softmax (2.6) permet de transformer un vecteur réel, en un vecteur de probabilités. On l'utilise souvent, dans la couche finale d'un modèle de classification, notamment pour les problèmes multiclasses.

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \quad 2.7$$

Où x_i est le vecteur d'entrée

f. Fonction Relu

Le fonctionnement de base de la fonction ReLU (figure 2.6) est simple : elle émet directement la valeur d'entrée si celle-ci est positive, transmet un zéro, si l'entrée est négative ou nulle. Ce simple mécanisme de seuillage (2.8), introduit une non-linéarité essentielle dans le réseau neuronal.

$$f(x) = \begin{cases} x \; si \; x > 0 \\ 0 \leq 0 \end{cases} \quad 2.8$$

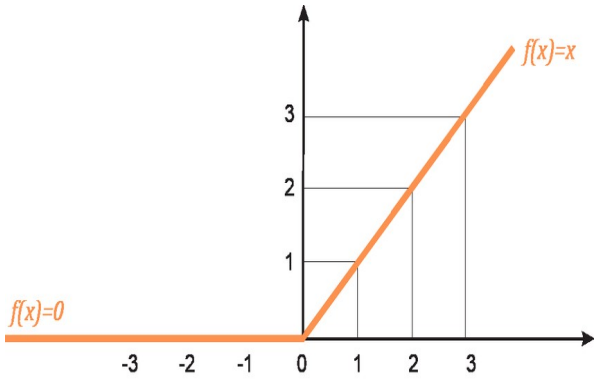


Figure 2.7 : Fonction Relu [49]

2.5 Introduction à l'apprentissage profond

L'apprentissage profond (Deep Learning) est une sous-catégorie du Machine Learning (figure 2.8) qui utilise des réseaux de neurones artificiels à plusieurs couches successives, pour extraire automatiquement des représentations hiérarchiques à partir des données [56-61].

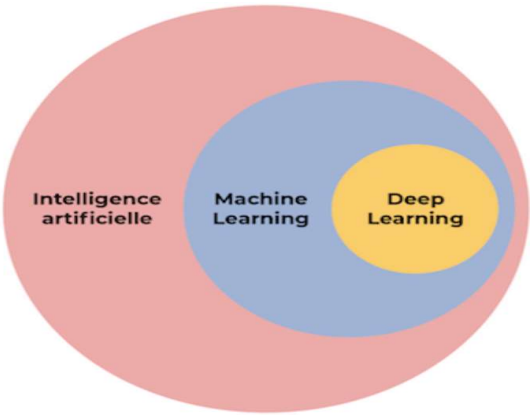


Figure 2.8 : Sous branches de l'IA [57]

Le deep learning permet à une machine d'apprendre à effectuer des tâches complexes, en s'entraînant sur de grandes bases de données [56, 57-59].

2.5.1 Fonctionnement du deep learning

Le modèle du deep learning doit être entraîné à l'aide d'un grand nombre d'images, pour réaliser des systèmes précis. Prenons l'exemple d'une base de données, représentant des chiens. Ce type de réseau neuronal tire son apprentissage des pixels contenus, dans les images reçues. Il peut classer des groupes de pixels, en fonction des caractéristiques du chien ; telles que les griffes, les oreilles et les yeux, indiquant la présence de l'animal dans l'image. Ces caractéristiques, sont déterminées dans ce cas, à partir d'une suite d'opérations de convolution [60].

2.5.2 Différence entre l'apprentissage automatique et l'apprentissage profond

Dans l'apprentissage machine (automatique) un programmeur doit effectuer une extraction de caractéristiques numériques comme, la surface, le diamètre, le volume, etc. avant la classification. Dans l'apprentissage profond il suffit juste d'avoir des données gigantesques, pour apprendre des modèles (figure 2.9).

Les approches basées sur cet apprentissage, permettent l'extraction automatique des caractéristiques, dans les différentes couches du réseau, pour être apprises par le modèle de lui-même [61].

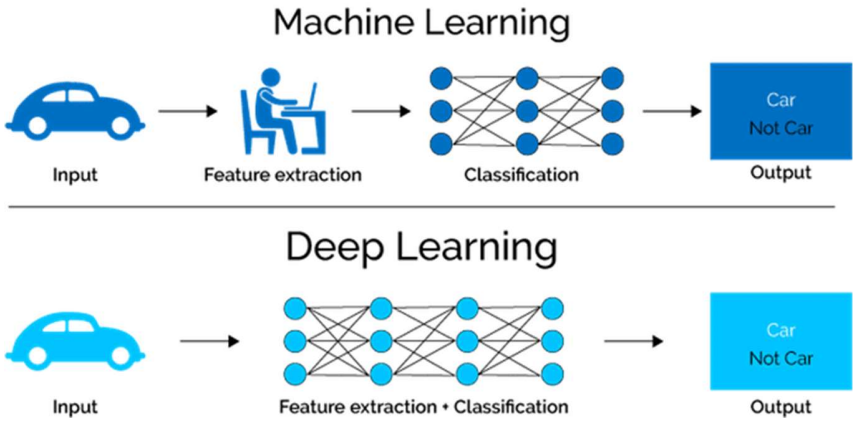


Figure 2.9 : Différence entre l'apprentissage automatique et l'apprentissage profond [61]

2.6 Réseaux de neurones dans l'apprentissage profond

Nous décrivons dans ce qui suit, les principaux modèles des réseaux de neurones dans l'apprentissage profond, les CNN (détaillés) et les RNN (présentés brièvement).

2.6.1 Réseaux de neurones récurrents

Ce sont des réseaux avec des boucles, qui permettent à l'information, de se maintenir. Dans la figure 2.10, un segment du réseau neuronal ; "A" avec une entrée Xt, fournit une valeur ht. Cette boucle permet de transmettre des informations, d'une étape du réseau à l'autre [60, 62].

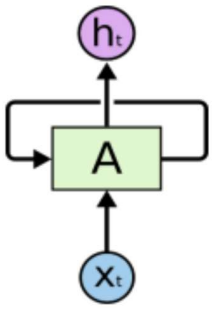


Figure 2.10 : Boucle d'un réseau neuronal récurrent [62]

Un réseau de neurones récurrent est considéré comme des copies multiples, du même réseau ; chacune transmettant un message à son successeur (figure 2.11).

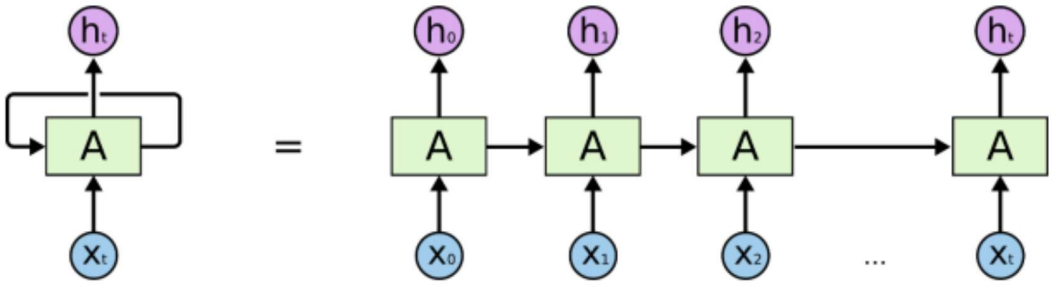


Figure 2.10 : Réseau neuronal récurrent [62]

2.6.2 Réseaux de Neurones Convolutifs

Les réseaux de neurones convolutifs (Convolutional Neural Networks, CNN) constituent une architecture fondamentale du Deep Learning, particulièrement performante pour l'analyse d'images et de signaux structurés. Leur origine remonte aux années 1980 suivant les travaux de Fukushima sur le « Neocognitron », inspiré des recherches de Hubel et Wiesel sur la vision biologique [63]. LeNet-5, développé par Yann LeCun dans les années 1990 [64], fut l'un des premiers CNN utilisés avec succès pour la reconnaissance de chiffres manuscrits, ouvrant la voie à une généralisation de ces architectures, dans de nombreuses applications. Par la suite, des variantes plus profondes comme celles utilisées par Ciresan ont battu des records sur des bases telles que MNIST, CIFAR-10 ou ImageNet [65].

Afin de mieux comprendre la structure et le fonctionnement de ces réseaux, nous allons maintenant détailler leurs composants principaux.

Un réseau neuronal convolutif est composé de plusieurs couches, comme le montre la figure 2. 11.

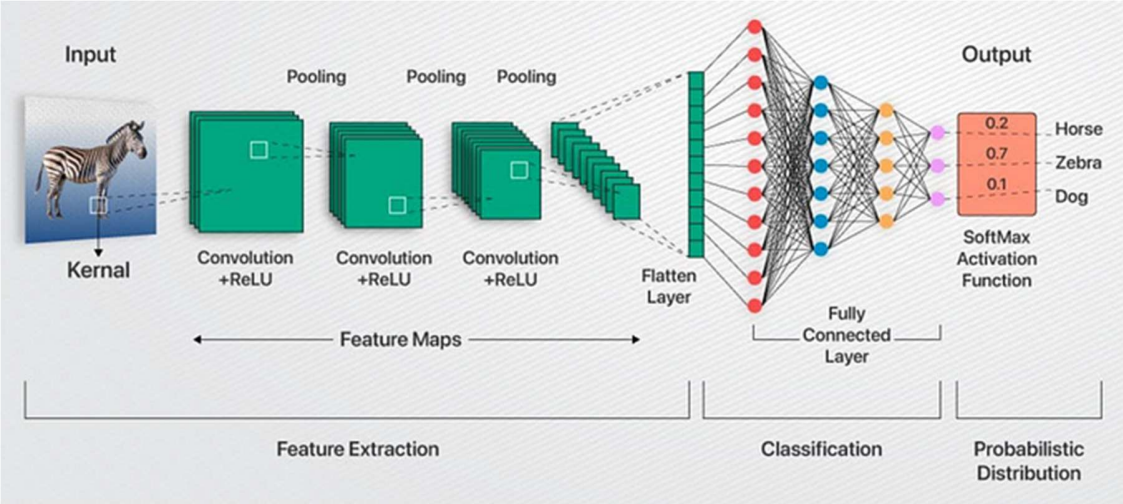


Figure 2.11: Exemple d'un CNN [66]

a. Couche de convolution

La couche de convolution est l'une des composantes les plus fondamentales d'un réseau neuronal. Son but est d'extraire les caractéristiques dans les images reçues en entrée, pour produire une carte de caractéristiques.

Le principe consiste à déplacer un filtre sur l'image, en calculant le produit de convolution entre ce filtre et chaque portion de l'image balayée. Les valeurs du filtre sont multipliées par celles des pixels de la fenêtre traitée, produisant un seul nombre pour chaque position. Les valeurs obtenues pour toutes les positions, sont ensuite combinées pour former une carte de caractéristiques [67-70].

Plusieurs filtres sont utilisés, pour apprendre différents types de motifs, comme la texture et la forme. La figure 2.12 montre un exemple du processus de convolution.

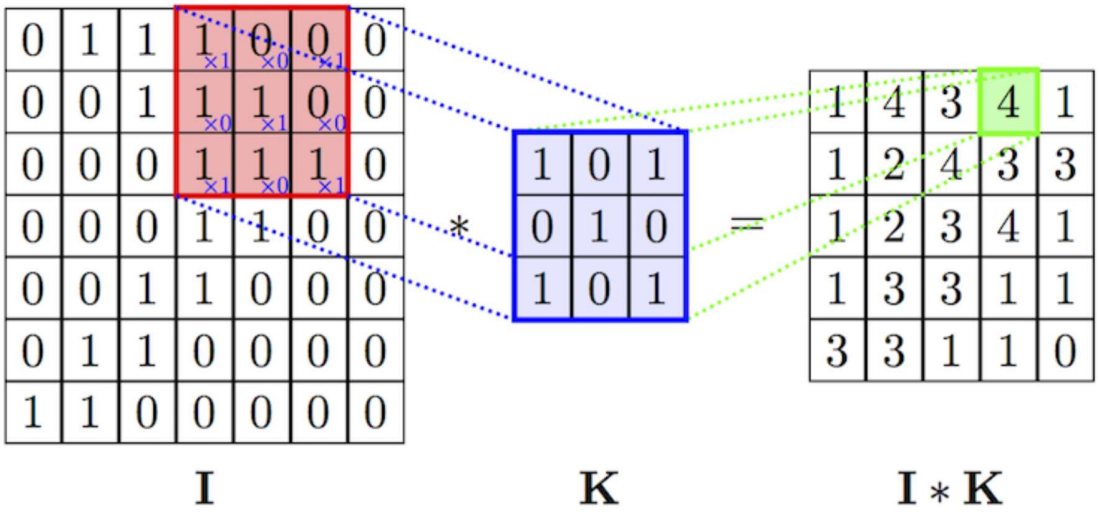


Figure 2.12 : Exemple d'une convolution 2D [69]

Trois hyperparamètres déterminent le volume de la couche de convolution :

- La profondeur de la couche « K (nombre de noyaux de convolution) ;
- Le pas « S » contrôle le chevauchement des descripteurs ;
- Le zéro padding « P » : permet de contrôler la dimension spatiale du volume de sortie.

b. Couche de pooling

L'opération de Pooling est une technique qui permet de réduire la taille d'une image, pour extraire une valeur unique d'une région, tout en préservant les caractéristiques importantes [70, 71]. La valeur extraite dépend du type de regroupement utilisé.

Dans ce cadre, les types de regroupement les plus courants sont le max pooling pour extraire la valeur la plus élevée et le pooling moyen, qui extrait la valeur moyenne d'une région (figure 2.13).

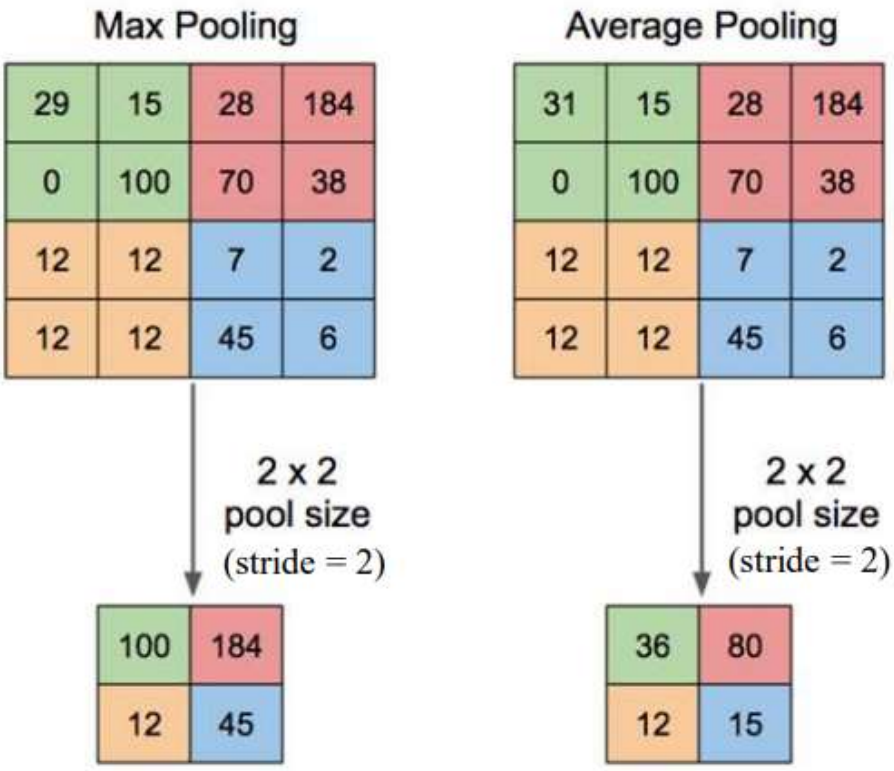


Figure 2.13 : Exemple du pooling [71]

c. Couche Flatten

C'est la couche d'aplatissement qui sert à transformer une matrice de dimensions, NxN en un vecteur de Nx1 (figure 2.14).

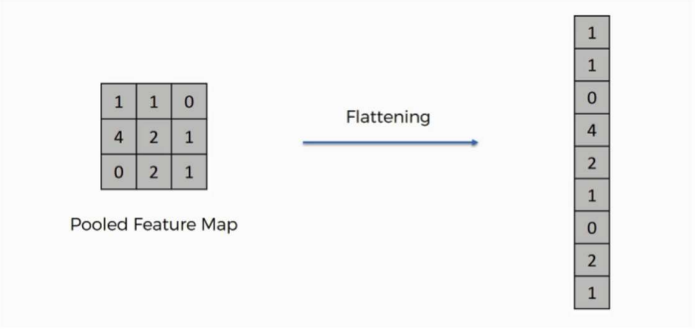


Figure 2.14 : Exemple d'aplatissement [71]

d. Couche Relu

Son rôle est d'introduire une non-linéarité dans le modèle, en appliquant une fonction non linéaire Relu, aux sorties de la couche précédente, remplaçant ainsi les valeurs négatives par des zéros.

La non-linéarité est importante, car elle permet au modèle d'apprendre des relations plus complexes, entre les caractéristiques. Sans la non-linéarité, la sortie du modèle serait une combinaison linéaire des entrées, ce qui limite sa capacité à modéliser des problèmes complexes [71].

e. Couche entièrement connectée

Après plusieurs couches de convolution et de pooling, le raisonnement de haut niveau dans le réseau neuronal se fait via des couches totalement connectées (figure 2.15).

La dernière couche du réseau convolutif est capable de faire une classification spécifique, en combinant toutes les caractéristiques détectées par les couches précédentes, dans les données d'entrée [71].

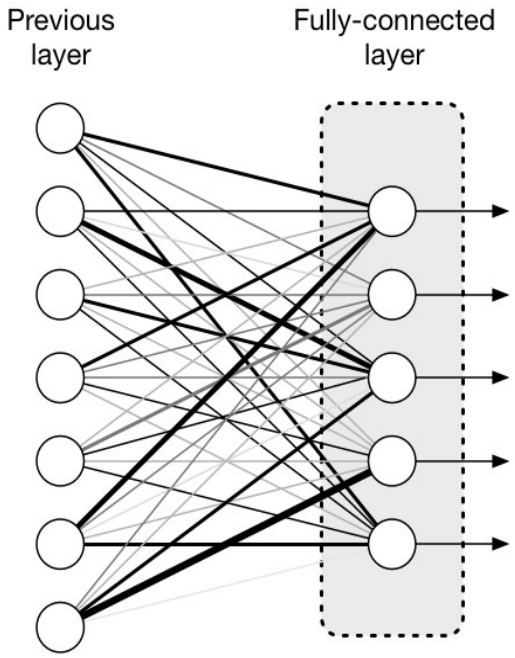


Figure 2.15 : Exemple d'une couche entièrement connectée [71]

2.7 Techniques de régularisation et généralisation

Les techniques de régularisation permettent au réseau de neurones, d'éviter le sur-apprentissage, pour stabiliser l'entraînement [72].

- **Dropout** : pendant l'entraînement, certains neurones sont désactivés de manière aléatoire, ce qui permet d'éviter que le modèle ne devienne trop dépendant d'un neurone spécifique et favorise ainsi la généralisation.
- **Régularisation L1 et L2 (Lasso et Ridge)** : Appliquées aux poids de la couche dense du bloc de classification, ces régularisations ajoutent une pénalité à la fonction de perte en fonction de l'amplitude des poids. La régularisation L1 (norme L1 des poids) tend à rendre les poids de petite valeur nuls, favorisant ainsi la parcimonie du modèle. La régularisation L2 (carré de la norme L2 des poids) pénalise les poids de grande valeur, les empêchant de devenir excessivement grands et réduisant ainsi, la complexité du modèle.
- **Early Stopping** : Il consiste à arrêter l'entraînement dès que les performances du modèle, commencent à se dégrader sur les données de validation. Cela empêche le modèle de continuer à apprendre des bruits et des modèles spécifiques à partir des données d'entraînement.
- **Batch normalisation** : Cette opération normalise la sortie d'une couche précédente, en redistribuant les activations de telle sorte que la moyenne de sortie, soit proche de zéro et que l'écart type soit proche de 1. Ceci stabilise le processus d'apprentissage.
- **Gestion du déséquilibre de classes (Class Weighting) :** Bien que non directement une technique de régularisation au sens strict, la pondération des classes lors de l'entraînement contribue également à la généralisation, en empêchant le modèle de biaiser ses prédictions vers la classe majoritaire. Cela assure que le modèle apprend des caractéristiques pertinentes pour toutes les classes.

2.8 Paramètres d'entraînement et fonctions d'optimisation

Le Deep Learning repose sur des réseaux de neurones artificiels organisés en plusieurs couches, où chaque neurone est connecté à ceux des couches précédentes et suivantes, via des poids synaptiques. Ces poids sont ajustés automatiquement, pendant l'apprentissage pour minimiser l'erreur entre la sortie réelle et la sortie attendue [72, 73].

Il s'appuie principalement sur deux mécanismes :

- La **propagation avant (forward propagation)** : le signal traverse les couches du réseau pour produire une sortie.
- La **rétropropagation de l'erreur (backpropagation)** : l'erreur entre la sortie attendue et la sortie obtenue, est propagée en arrière pour mettre à jour les poids synaptiques, selon la descente de gradient.

2.8.1 Paramètres d'entraînement

Les paramètres d'entraînement clefs [72, 73] dans l'apprentissage sont considérés par :

- Le taux d'apprentissage (learning rate) permet de déterminer la taille des ajustements effectués sur les poids du modèle, à chaque étape de l'entraînement.
- La fonction de perte (loss function) : mesure l'écart entre les prédictions du modèle et les étiquettes réelles de l'ensemble d'entraînement.

Il existe de nombreuses fonctions de perte différentes, qui sont choisies en fonction du type de problème de l'apprentissage automatique. Nous citons dans ce cas, deux fonctions de perte couramment utilisées.

1. Entropie croisée binaire

Elle est utilisée pour les problèmes de classification binaire. L'entropie binaire $\mathcal{L}(\mathrm{Y}, \widehat{\mathrm{Y}})$ correspond à l'équation (2.9).

$$\mathcal{L}(\mathrm{Y}, \widehat{\mathrm{Y}}) = -\left[\mathrm{Y}\log(\widehat{\mathrm{Y}}) + (1-\mathrm{Y})\log(1-\widehat{\mathrm{Y}})\right] \quad 2.9$$

Avec,

Y classe réelle,
$\widehat{Y}$ classe prédite.

2. Entropie croisée catégorielle

Elle est utilisée pour les problèmes de classification multi-classes. Cette entropie mesure la différence entre les distributions de probabilités prédites Y_i et, les distributions de probabilités réelles $\hat{Y}_i$(2.10).

$$Loss = -\sum_{i=1}^{C} Y_i \, log\left(\hat{Y}_i\right) \quad 2.10$$

2.8.2 Fonctions d'optimisation

Les fonctions d'optimisation permettent de minimiser la fonction de perte, en utilisant la descente de gradient ou ses variantes [74].

1. Descente du gradient

Cette méthode [48] a pour but de trouver le minimum local de la fonction coût, en faisant plusieurs itérations, afin d'ajuster les poids (2.11).

$$W_j = W_i - \tau \frac{\delta J}{\delta W_i} \quad 2.11$$

Avec,

W_i, ancien poids,
W_j nouveau poids,
τ taux d'apprentissage.
J fonction coût.

2. Descente du gradient stochastique

La descente de gradient consiste à trouver un minimum local, étant donné la fonction dans tous les échantillons du dataset, ce qui peut être relativement lent si le dataset est large. La SGD (Stochastic Gradient Descent) quant à elle, prend un échantillon au hasard (d'où le nom stochastique) et applique l'algorithme du gradient, depuis cet échantillon pour ajuster les poids.

3. Descente de gradient stochastique avec momentum

Elle consiste à ajouter un momentum β à la descente de gradient stochastique régulière (2.12). Ce Momentum conserve la mise à jour à chaque itération et, l'ajoute à l'itération suivante [48].

$$W_j = \beta W_i - \tau \frac{\delta J}{\delta W_i} \quad 2.12$$

4. Estimation du moment adaptatif

Il s'agit d'une extension de la descente du gradient stochastique. Dans cet algorithme d'optimisation, Le moment adaptatif ADAM, met à jour les poids W (2.13). La fonction permet de calculer les taux d'apprentissage adaptatifs, pour chaque paramètre.

$$W_t = W_t - \tau \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \quad 2.13$$

Avec,

$\hat{m}_t$ la moyenne des gradients,
$\hat{v}_t$ la moyenne des carrés des gradients,
ϵ Terme de stabilité numérique.

2.9 Apprentissage par transfert

L'apprentissage par transfert est une technique issue du machine learning, permettant de réutiliser les connaissances acquises sur une tâche initiale (source), pour améliorer les performances sur une tâche différente mais connexe (cible). Plutôt que d'entraîner un modèle à partir de zéro, le transfert utilise un modèle pré-entraîné, souvent sur un grand ensemble de données génériques (ex. : ImageNet), puis l'adapte à un nouveau domaine, réduisant ainsi les besoins en données et en ressources de calcul [75].

Les méthodes traditionnelles du machine learning, supposent que les données d'entraînement et de test appartiennent au même espace de caractéristiques et suivent la même distribution. En revanche, l'apprentissage par transfert permet d'outrepasser cette contrainte, en tirant parti de l'expérience accumulée sur des tâches précédentes, pour améliorer la généralisation [76].

La réalisation d'un modèle basé sur l'apprentissage par transfert [77], suit les étapes ci-dessous :

- **Sélection d'un modèle pré-entraîné** : Choix d'un modèle ayant été entraîné sur une tâche proche (ex. : ImageNet).
- **Adaptation du modèle** par la suppression de la dernière couche spécifique à la tâche initiale et l'ajout de nouvelles couches pour la tâche cible.
- **Gel des couches** : Certaines couches peuvent être gelées (non mises à jour) pour conserver les représentations apprises.
- **Entraînement sur la tâche cible** par l'ajustement des poids restants et des nouvelles couches avec un faible taux d'apprentissage.
- **Évaluation et ajustement** par la surveillance de la performance sur les données de validation et l'ajustement des hyperparamètres.

2.10 Fine-Tuning : Affinage technique des modèles pré-entraînés

Le Fine-Tuning représente une extension du transfert learning, où non seulement de nouvelles couches sont ajoutées à un modèle pré-entraîné, mais certaines couches internes sont également réentraînées, pour mieux adapter le modèle à la tâche cible. Ce processus est particulièrement utile lorsque les données disponibles pour la nouvelle tâche, sont limitées mais que des performances élevées sont attendues.

Son principe consiste à réentraîner partiellement ou totalement, un réseau de neurones pré-entraîné. Techniquement, cela se fait en dégelant ("unfreezing") une ou plusieurs couches du réseau et en réajustant leurs poids, selon une stratégie adaptée. Cette méthode repose sur l'hypothèse que les premières couches d'un réseau (près de l'entrée) apprennent des représentations génériques (ex : bords, textures), tandis que les couches plus profondes apprennent des représentations spécifiques à la tâche [78-80].

2.10.1 Fine tuning total

La couche fully-connected du réseau pré-entraîné, est remplacée par un classifieur adapté au nouveau problème et initialisé aléatoirement. Toutes les couches sont par la suite, entraînées sur les nouvelles images.

2.10.2 Fine tuning partiel

La dernière couche fully-connected, est remplacée par le nouveau classifieur initialisé aléatoirement et, les paramètres de certaines couches, du réseau pré-entraîné sont fixés. Ainsi, en plus du classifieur, les couches non-fixées, seront entraînées sur les nouvelles données, qui correspondent généralement aux plus hautes du réseau.

Le Fine-Tuning représente une approche fine et puissante pour adapter des modèles génériques à des contextes très spécifiques, tout en réduisant les ressources de calcul. Il permet d'obtenir des résultats de haute précision même sur des tâches spécialisées, en particulier lorsqu'il est combiné à des techniques de régularisation et d'optimisation avancées.

2.11 Conclusion

Dans ce chapitre, nous avons exploré les fondements théoriques de l'intelligence artificielle et ses principales composantes, en particulier le machine learning et l'apprentissage profond. Nous avons mis en lumière les différents modules des réseaux de neurones convolutifs. Des techniques avancées comme le Fine-Tuning, ont également été discutées, soulignant leur importance croissante dans les applications modernes, notamment en classification et optimisation de systèmes intelligents.

L'ensemble de ces notions, forme un socle solide pour comprendre et mettre en œuvre des solutions d'IA performantes dans divers contextes, y compris celui des réseaux de télécommunications.

Ces fondements théoriques désormais posés, nous allons aborder dans le chapitre suivant, la phase conceptuelle de ce projet, en nous intéressant à la structuration de l'approche pratique, à la préparation des données issues des réseaux réels, ainsi qu'à l'élaboration de la solution de classification basée sur les réseaux de neurones convolutifs pré-entraînés.

Chapitre 3 Etude conceptuelle pour la prédiction des cellules LTE

3.1 Introduction

En s'appuyant sur les bases théoriques exposées au chapitre précédent, ce chapitre a pour but de définir le cadre conceptuel du projet de fin d'études. Nous commencerons par présenter le contexte professionnel et technique dans lequel s'inscrit cette recherche, en exposant également les limites des approches classiques utilisées pour gérer le déséquilibre du réseau LTE. À partir de cette analyse, les motivations ayant conduit à l'adoption d'une approche reposant sur le Deep Learning, seront expliquées. Plus spécifiquement, l'intérêt du transfert d'apprentissage pour prédire automatiquement l'état d'une cellule LTE à partir de ses indicateurs de performance, sera mis en avant.

Ce travail vise à démontrer que l'intelligence artificielle peut offrir une solution fiable, rapide et adaptable aux méthodes traditionnelles, souvent chronophages et difficiles à généraliser à une large échelle.

3.2 De la complexité des réseaux LTE à la nécessité de l'apprentissage profond

Les réseaux mobiles LTE présentent une complexité croissante en raison de la densification des cellules, de la diversité des équipements utilisateurs (UE) et de la variabilité du trafic selon les zones géographiques et les périodes temporelles. Les méthodes traditionnelles de supervision, reposant sur des règles fixes ou des seuils empiriques, montrent leurs limites en termes d'adaptabilité, de précision et de scalabilité [2]. Dans ce contexte, l'intelligence artificielle (IA), et plus particulièrement l'apprentissage profond (Deep Learning), émerge comme une solution puissante pour capturer les schémas complexes et dynamiques des données de réseau. Le Deep Learning permet d'extraire automatiquement des caractéristiques pertinentes à partir de données brutes sans recourir à une ingénierie manuelle, ce qui est particulièrement utile dans des environnements où les corrélations entre les paramètres, sont non linéaires et multidimensionnelles [81].

Les réseaux de neurones convolutifs (CNN) sont à l'origine conçus pour des données visuelles, mais leur usage s'est étendu à d'autres domaines en transformant des données numériques en images ou représentations visuelles. Cela permet d'exploiter leur capacité à apprendre des représentations hiérarchiques et discriminantes, même à partir de données 'réseau' transformées, comme les KPI LTE convertis en heatmaps ou graphiques matriciels [82].

3.3 Approche adoptée

L'approche globale de notre solution, illustrée par le synoptique de la figure 3.1, résume les grandes étapes conceptuelles et opérationnelles de la classification du réseau cellulaire.

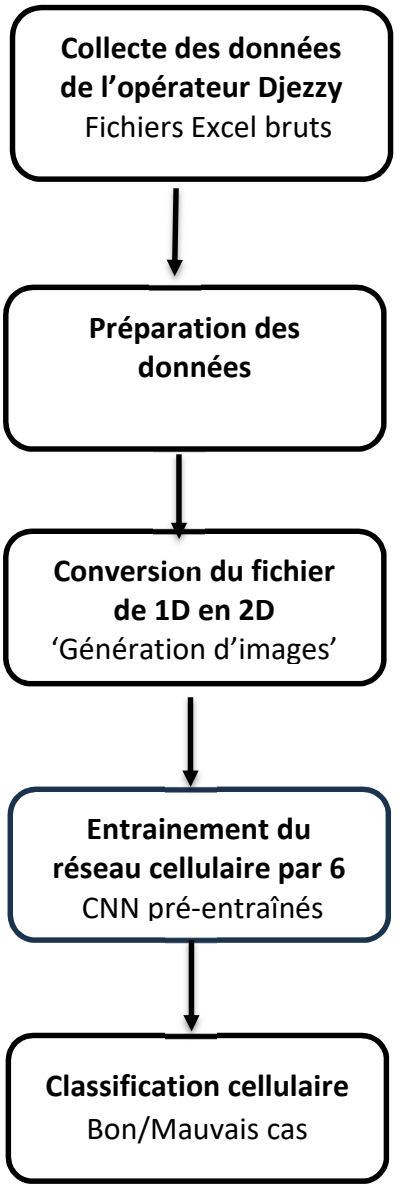


Figure 3.1 : Synoptique de l'approche adoptée

Dans le cadre de ce projet, l'objectif principal est de classifier l'état d'une cellule LTE, en fonction de ses indicateurs de performance (KPI), afin de prédire l'opportunité d'activer le mécanisme de Mobility Load Balancing (MLB). Cette classification est accomplie suivant plusieurs variantes des CNN pré-entrainés, comme le VGG16, MobilenetV1, MobilenetV2, MobilenetV3 small, EfficientnetB0 et Resnet50.

La tâche adoptée est formalisée comme un problème de classification binaire, où chaque cellule est catégorisée dans l'une des deux classes suivantes :

- Classe 1 (Bad Case) : la cellule se trouve en situation de surcharge ou de déséquilibre de charge, ce qui indique que l'activation du MLB est recommandée.
- Classe 0 (Good Case) : la cellule fonctionne dans des conditions normales ou ne présente pas de surcharge critique justifiant le déclenchement du MLB ; aucune action n'est requise.

L'entrée du modèle est une image synthétique, générée à partir de la combinaison de plusieurs KPI clés, (comme, PRB, Time Advance (TA), Throughput et User Equipment (UE) Count). La sortie du modèle est une probabilité qui indique la classe prédite, pour l'état de la cellule. Cette approche permet au modèle d'apprendre à reconnaître des motifs complexes dans les données, pour prendre une décision éclairée.

3.4 Nature des données d'entrée

L'efficacité d'un modèle d'apprentissage profond repose intrinsèquement sur la qualité et la pertinence des données d'entrée. Dans le cadre de ce projet, les données proviennent des plateformes de supervision réseau Huawei (U2020/MAOS) [83], qui permettent l'extraction de fichiers bruts contenant les indicateurs de performance clés (KPIs) des cellules LTE. Ces données, initialement sous forme tabulaire (CSV, Excel), nécessitent des étapes rigoureuses de nettoyage, de transformation et de normalisation, avant de pouvoir être exploitées efficacement comme entrées pour un modèle de réseau de neurones convolutif (CNN).

Les KPIs sont des mesures fondamentales pour l'évaluation continue de la performance et de la santé du réseau LTE [22, 82]. Ils permettent d'identifier les problèmes de congestion, de saturation des ressources et de dégradation de la qualité de service. Chez Djezzy, plusieurs KPIs spécifiques sont suivis, afin de garantir une gestion optimale du trafic [4, 5, 84]. Les indicateurs clés utilisés dans cette étude sont décrits ci-dessous.

1. **Nombre d'UE**

Ce KPI (3.1) mesure le nombre d'utilisateurs (appareils connectés) actifs sur une cellule à un instant donné.

Il est essentiel pour évaluer la charge utilisateur et la pression sur les ressources du réseau. Un nombre élevé des UEs **(User Equipment)**, peut indiquer un potentiel de congestion si les ressources allouées sont insuffisantes.

UE = Nombre total d'équipements connectés 3.1

2. **Trafic**

Le trafic **(Traffic)** quantifie le volume de données échangées (en uplink et downlink) sur le réseau au cours d'un intervalle de temps défini (3.2). Il est crucial pour analyser la demande en bande passante et identifier les périodes de surcharge réseau.

$$\boldsymbol{Trafic} = \frac{\boldsymbol{Données(Mbps)}}{\textbf{Temps}} \quad 3.2$$

3. **PRB**

Le PRB **(Physical Resource Block)** est un indicateur clé de l'utilisation des ressources physiques radio. Chaque PRB (3.3) représente une unité minimale d'allocation de ressources dans le domaine fréquentiel et temporel. Le suivi de l'utilisation des PRBs permet d'évaluer l'efficacité de l'allocation des ressources et de détecter les goulots d'étranglement.

$$PRB = \frac{\text{PRB utilisées}}{\text{PRB disponibles}} \times 100 \quad 3.3$$

4. **Débit**

Le débit **(Throughput)** mesure la quantité de données transférées par unité de temps (souvent en bits par seconde). C'est un indicateur (3.4) direct de la performance du réseau et, de la qualité d'expérience utilisateur. Une diminution du débit, peut être le symptôme d'une congestion réseau ou d'un manque de capacité.

$$Debit = \frac{\text{Volume de données}}{\text{Durée}} \quad 3.4$$

5. **TA**

Le Time Advance (TA) est une mesure qui indique la distance entre l'équipement utilisateur (UE) et la station de base. Il est utilisé par le réseau pour synchroniser les transmissions.
Un TA élevé peut suggérer que l'indicateur UE est éloigné de la cellule, ce qui peut impacter la qualité du signal et la performance. Ce paramètre est représenté par l'équation 3.5.

$$TA = Distance \times Vitesse\ de\ propagation \quad (3.5)$$

Ces KPIs jouent un rôle central dans la gestion et l'optimisation continue du réseau LTE chez Djezzy. En exploitant les données issues de plateformes comme Huawei U2020 [85], il est possible de surveiller en temps réel la performance du réseau, d'optimiser l'utilisation des ressources et d'anticiper les problèmes de congestion pour garantir une expérience utilisateur de haute qualité.

3.4 Préparation des données

La préparation des données est effectuée par plusieurs traitements, dont le filtrage des données, l'encodage des classes cibles et la normalisation des KPI.

3.4.1 Nettoyage et filtrage des données

Le prétraitement des données brutes, essentiel pour garantir la robustesse et la fiabilité du modèle, commence par une série d'opérations de nettoyage et de filtrage [86]. Ces étapes visent à éliminer les incohérences et à ne conserver que les données les plus pertinentes, pour l'analyse. Ce prétraitement considère les opérations ci-dessous :

- **Gestion des valeurs aberrantes et manquantes :** Suppression des doublons, des lignes incomplètes et des entrées incohérentes (telles que les valeurs négatives, les zéros aberrants ou les cellules vides).
- **Harmonisation des identifiants :** Standardisation des formats des identifiants clés du réseau, comme les identifiants d'eNodeB et de Cell ID, pour assurer la cohérence et la traçabilité des données.
- **Filtrage temporel :** Conservation exclusive des données correspondant aux périodes de forte activité du réseau (par exemple, les heures de pointe). Cette approche garantit que l'analyse se concentre sur les scénarios les plus représentatifs et critiques de charge réseau.

- **Exclusion de périmètre :** Retrait des cellules ou des secteurs qui ne sont pas éligibles ou configurés pour l'activation du mécanisme de Mobility Load Balancing (MLB), ou qui sont considérés comme hors du périmètre d'étude.

3.4.2 Encodage des classes cibles

L'annotation des classes cibles [86], pierre angulaire de notre approche de la classification supervisée, a été menée avec une rigueur particulière pour assurer la fiabilité de la "vérité terrain".

Contrairement aux approches initiales basées sur des règles heuristiques simples, ou une annotation purement manuelle, nous avons entrepris une révision et une consolidation approfondies de l'ensemble des données brutes via un traitement sur Excel. Cette démarche a permis de constituer une nouvelle base de données labellisée, où chaque enregistrement de KPI est explicitement associé à sa classe (Good Case ou Bad Case). Cette labellisation affinée a permis de définir avec précision les conditions menant à un déséquilibre nécessitant le MLB.

L'objectif final est de formaliser la tâche en un problème de classification binaire, avec les deux classes suivantes :

- Classe 1 (Bad Case) : Cette classe indique une situation où la cellule présente une surcharge ou un déséquilibre significatif, justifiant l'activation immédiate du mécanisme de Mobility Load Balancing (MLB).
- Classe 0 (Good Case) : Cette classe représente un état normal de la cellule, où les indicateurs de performance sont stables et aucune action d'équilibrage de charge n'est requise.

Ce processus de labellisation soignée est crucial, car il transforme la tâche en un problème de classification binaire supervisée, permettant au modèle d'apprendre directement des exemples étiquetés.

3.4.3 Normalisation des KPI

Les indicateurs de performance clés (KPIs) sont intrinsèquement exprimés dans des unités de mesure hétérogènes, où ils présentent des échelles de valeurs variées [9, 84]. Pour garantir que chaque indicateur contribue équitablement à la représentation visuelle des données et pour éviter que les KPIs ayant de grandes amplitudes, ne dominent le processus d'apprentissage, une étape de normalisation est indispensable.

Nous avons opté pour la méthode de **Min-Max Scaling**. Cette technique transforme toutes les valeurs des KPI de manière linéaire afin de les ramener dans un intervalle prédéfini, typiquement entre [0,1]. La formule de normalisation est représentée par l'équation 3.6.

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad 3.6$$

Où X est la valeur originale du KPI, Xmin est la valeur minimale observée pour ce KPI dans l'ensemble des données et Xmax est la valeur maximale.

Cette normalisation est cruciale non seulement pour standardiser l'entrée des réseaux de neurones, mais aussi pour s'assurer que l'intensité visuelle dans les images générées reflète de manière proportionnelle l'importance relative de chaque KPI, sans biais lié à leur échelle d'origine.

3.5 Génération d'images à partir des KPI

Afin de rendre les données tabulaires des KPI exploitables par un réseau de neurones convolutif (CNN), chaque ligne de données correspondant à l'état d'un secteur LTE à un instant t, est transformée en une image synthétique [4, 87]. Cette étape est cruciale car elle permet d'exploiter la puissance des CNN, intrinsèquement conçus pour l'analyse de données visuelles. Dans ce cadre, un ensemble de neuf indicateurs de performance clés normalisés, est sélectionné pour former la base de chaque image. Ces valeurs numériques représentatives de divers aspects de la performance cellulaire (tels que l'utilisation des ressources radio, les débits ou le comportement des utilisateurs sur différentes fréquences), sont agencées pour créer une matrice de 3×3 pixels. Chaque valeur de KPI normalisée, comprise entre 0 et 1, est ensuite convertie en une intensité de pixel en niveaux de gris (valeurs entre 0 et 255).

Le processus de transformation conceptuel est le suivant :

a. **Sélection des KPI** : Identification et extraction des neuf KPI les plus pertinents pour caractériser l'état d'une cellule, choisis pour leur capacité à révéler des dynamiques de surcharge ou de déséquilibre.
b. **Mise en forme matricielle** : Organisation des neuf valeurs de KPI normalisées dans une structure matricielle de 3×3, où la position de chaque KPI peut être prédéfinie pour faciliter l'interprétation des motifs spatiaux par le CNN.

c. **Conversion en image en niveaux de gris** : Représentation visuelle de cette matrice sous forme d'image monochome (niveaux de gris), où l'intensité de chaque pixel est directement proportionnelle à la valeur normalisée du KPI correspondant.
d. **Préparation pour le modèle** : Bien que l'image intrinsèque soit compacte (3×3 pixels), elle est ensuite redimensionnée à la résolution d'entrée standard attendue par les modèles CNN pré-entrainés (généralement 224×224 pixels) et convertie en trois canaux (RGB) par duplication des canaux de niveaux de gris, via le pipeline de données d'apprentissage profond. Cette standardisation est essentielle pour l'application du transfert d'apprentissage [87].

La conversion des données tabulaires des indicateurs de performance (KPI) en images visuelles, telle que détaillée précédemment, représente une approche méthodologique intéressante, offrant plusieurs bénéfices significatifs pour la classification de l'état des cellules LTE [5, 88]. Cette transformation des KPI en images, offre plusieurs avantages conceptuels :

- **Standardisation de l'entrée** : Elle permet de convertir des données tabulaires hétérogènes en un format visuel uniforme, rendant possible l'utilisation directe de modèles CNN pré-entraînés.
- **Exploitation des relations implicites** : L'agencement spatial des KPI dans l'image peut permettre au CNN de détecter des corrélations complexes et des motifs non linéaires, entre les indicateurs sans nécessiter une ingénierie de caractéristiques manuelle intensive.
- **Compatibilité avec le transfert d'apprentissage** : En adoptant un format visuel, même de nature synthétique, le modèle peut efficacement exploiter les caractéristiques génériques de bas niveau apprises sur des corpus d'images variés (comme ImageNet), réduisant ainsi le besoin en un grand volume de données étiquetées spécifiques au domaine.

3.6 Répartition du dataset et gestion du déséquilibre de classes

Une fois la base de données d'images synthétiques constituée et labellisée, elle est systématiquement répartie en trois sous-ensembles distincts pour les phases d'entraînement, de validation et de test du modèle. Cette division est fondamentale pour évaluer correctement la capacité de généralisation du modèle et pour éviter le surapprentissage. La répartition est la suivante :

- **70 % pour l'entraînement :** utilisé pour l'ajustement des poids internes du modèle.

- **15 % pour la validation :** employé pour le réglage fin des hyperparamètres du modèle et le suivi de sa convergence au cours de l'entraînement, sans influence sur les poids du réseau.
- **15 % pour le test final :** réservé pour une évaluation impartiale et finale des performances du modèle sur des données totalement inédites, simulant ainsi sa performance en conditions réelles.

Par ailleurs, compte tenu de la nature des données de réseau LTE, où les "Good Cases" (cellules normales) sont intrinsèquement plus fréquents que les "Bad Cases" (cellules en déséquilibre), un déséquilibre de classes est inévitable. Pour éviter que le modèle ne biaise ses prédictions en faveur de la classe majoritaire et ne compromette la détection des cellules réellement congestionnées, des techniques de rééquilibrage sont appliquées [88]. Plus spécifiquement, nous avons recours à une pondération des classes (class weighting), au moment de l'entraînement. Cette méthode assigne un poids plus élevé aux exemples de la classe minoritaire (les "Bad Cases") lors du calcul de la fonction de perte, forçant ainsi le modèle à accorder une plus grande importance à la détection correcte de ces cas critiques. Cette stratégie permet d'assurer que le modèle soit exposé de manière adéquate aux exemples de "Bad Cases", améliorant ainsi sa capacité à les identifier correctement.

3.7 Exploration des architectures de réseaux de neurones convolutifs choisis

Le choix d'une architecture de réseau de neurones convolutif (CNN) est une décision stratégique, qui impacte directement la performance, la complexité computationnelle et la capacité de déploiement d'une solution d'apprentissage profond. Face à la diversité des architectures existantes, plusieurs modèles reconnus pour leurs performances en classification d'images, sont considérés et analysés du point de vue de leur pertinence, pour la classification des états de cellules LTE. L'objectif est d'identifier l'architecture offrant le meilleur compromis, entre la robustesse de la classification, l’efficacité des ressources et le potentiel d'intégration opérationnelle.

Les architectures majeures explorées conceptuellement dans le cadre de ce projet, incluent six modèles.

3.7.1 Modèle VGG16

Le VGG16 est un modèle de réseaux de neurones convolutif proposé par Simonyan et al [90] de l'Université d'Oxford dans l'article intitulé « réseaux convolutionnels très profonds pour la reconnaissance d'images à grande échelle ».

Cette architecture classique (figure 3.2) reconnue pour sa simplicité structurelle, est basée sur 13 de couches de convolution de taille (3x3), 5 couches de max pooling et 3 couches entièrement connectées [90, 91, 92]. Sa profondeur importante lui confère une grande capacité à extraire des caractéristiques hiérarchiques complexes, souvent utilisée comme base robuste pour le transfert d'apprentissage.

Toutes les couches cachées, sont équipées de la non-linéarité de rectification (ReLU). Les cartes de caractéristiques (maps features) de la couche finale, sont représentées sous forme de vecteurs avec des valeurs scalaires, suivant la fonction de classification soft-max.

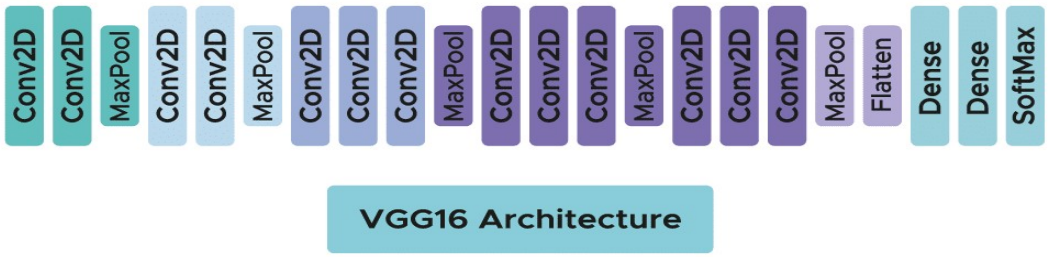


Figure 3.2 : Architecture de VGG16 [91]

Le réseau VGG16 a atteint une précision de 92,7% dans le top 5 des tests dans ImageNet [75], qui est un jeu de données de plus de 14 millions d'images, appartenant à 1000 classes.

3.7.2 EfficientNet

Cette famille d'architectures représente une avancée majeure en matière d'efficacité, offrant des performances de pointe avec un nombre de paramètres et de FLOPs (Floating Point Operations), considérablement réduits par rapport à des modèles plus anciens.

EfficientNet (figure 3.3) introduit une mise à l'échelle composée, qui optimise de manière uniforme la profondeur, la largeur et la résolution du réseau [93, 94].

Théoriquement, ce modèle propose un excellent rapport précision/coût computationnel, le rendant attractif pour des tâches où la performance est critique. Néanmoins, sa complexité interne et ses techniques d'optimisation spécifiques, peuvent rendre son fine-tuning plus délicat sur des jeux de données plus petits ou moins diversifiés, et son architecture moins intuitive à modifier.

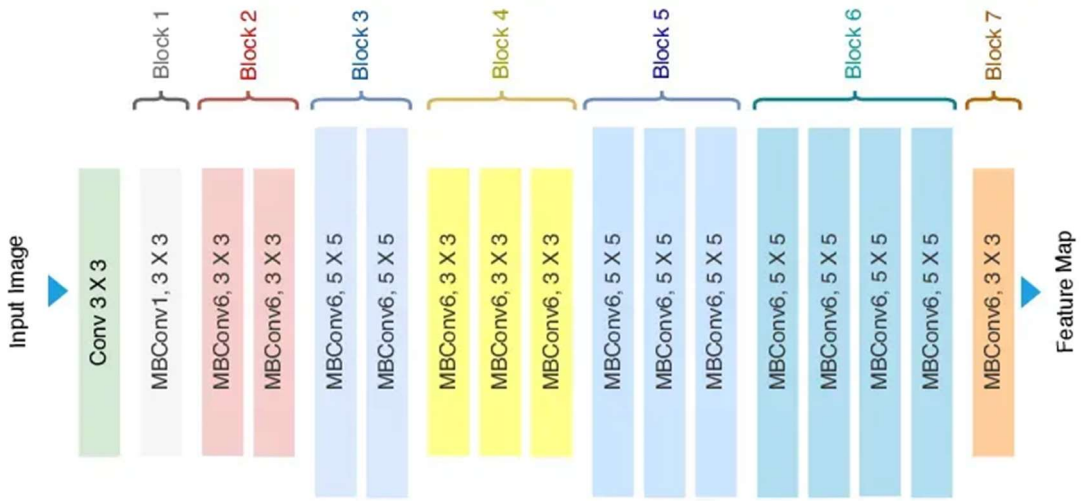


Figure 3.3 : Architecture du réseau efficientnet [93]

EfficientNet utilise des couches 'Mobile Inverted Bottleneck (MBConv)', qui sont une combinaison de convolutions séparables en profondeur et de blocs résiduels inversés.

3.7.3 MobileNet

Développé spécifiquement pour les applications embarquées et mobiles, MobileNet se distingue par son accent sur l'efficacité computationnelle et la légèreté [95].

Ce modèle se distingue par son approche optimisée basée sur les **convolutions séparables en profondeur (depthwise separable convolutions)** [96]**.** Cette technique clé (figure 3.4) décompose la convolution standard en deux étapes distinctes :

- **Convolution Depthwise :** Un filtre spatial indépendant est appliqué séparément à chaque canal d'entrée. Cette opération réduit significativement la complexité en traitant la profondeur des données séparément de leur dimension spatiale.
- **Convolution Pointwise :** Une convolution 1×1 est ensuite réalisée pour combiner linéairement les canaux résultant de la phase depthwise. Cette étape permet de fusionner les informations des différents canaux de manière efficace et avec un coût computationnel réduit.

Cette décomposition architecturale permet à MobileNet, de réduire considérablement le nombre de paramètres et les opérations de calcul nécessaires, tout en préservant des performances élevées en classification. À titre de comparaison, MobileNet contient typiquement environ 4,2 millions de paramètres, ce qui représente une réduction massive par rapport à VGG16.

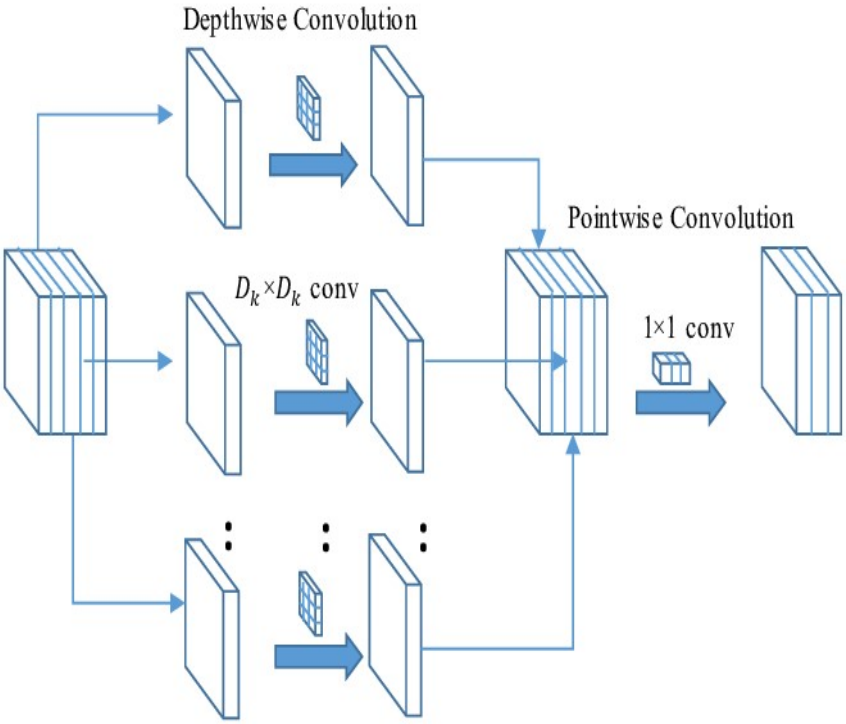


Figure 3.4 : Bloc de Convolution séparable en profondeur [96]

Les différentes variantes de MobileNet (comme MobileNetV1, V2 ou V3) offrent divers points de compromis, permettant de choisir le modèle le mieux adapté aux contraintes spécifiques de légèreté et de vitesse [95].

a. MobileNet V1

C'est la première version de MobileNet (figure 3.5), lancée par Google en 2017 [97, 98]. Elle utilise des couches de convolution profonde, pour extraire des caractéristiques à partir d'images d'entrée, suivies d'une réduction de dimension pour réduire la taille du modèle. Il introduit également deux hyperparamètres contrôlables :

- α (width multiplier) : réduit le nombre de canaux,
- ρ (resolution multiplier) : réduit la taille de l'image d'entrée.

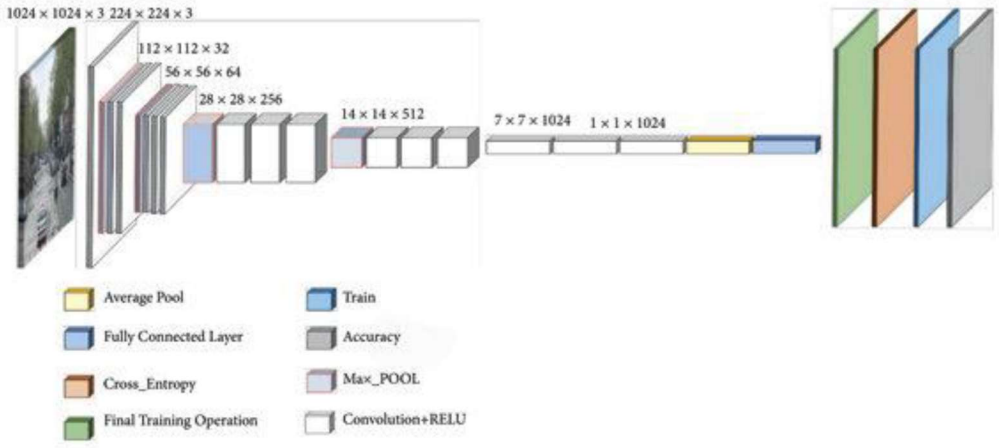


Figure 3.5 : Architecture de MobileNetV1 [97]

MobileNetV1 est une architecture composée de 28 couches dont 13 couches de convolution en profondeur et, 13 couches de convolution ponctuelle.

b. MobileNetV2

Cette version de MobileNet (figure 3.6) a été lancée en 2018 [98]. Elle utilise une technique appelée "inverted residual with linear bottleneck", où une couche est d'abord projetée dans un espace de plus grande dimension, puis recontractée, pour améliorer la précision et l'efficacité du modèle par rapport à la version précédente.

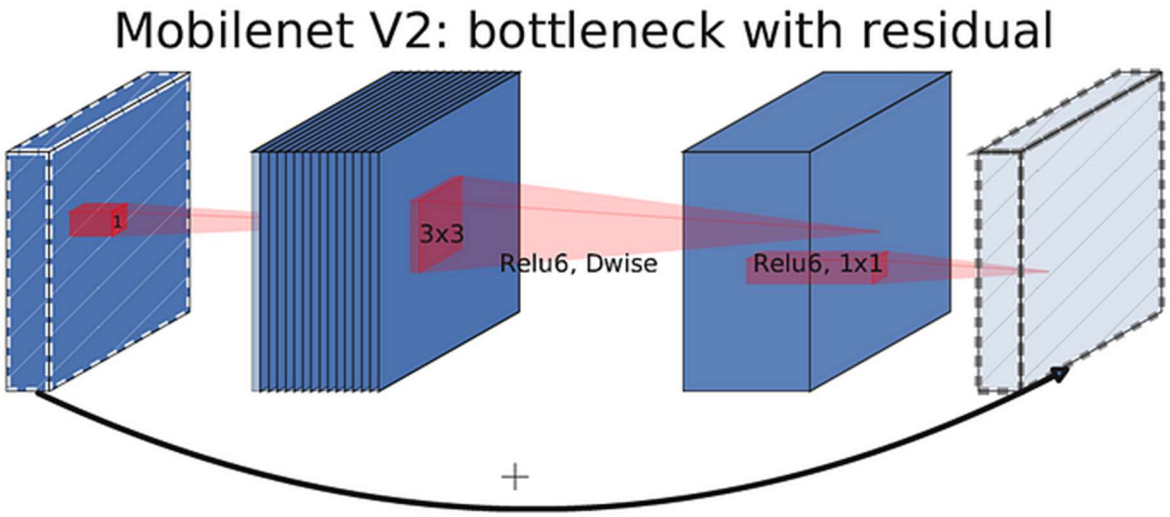


Figure 3.6 : Architecture de MobileNetV2 [98]

c. MobileNet V3 small

Cette version (figure 3.7) présente une architecture ultra-optimisée pour les systèmes embarqués. Créée en 2019, ce modèle introduit des blocs SE (squeeze and excitation), pour améliorer la qualité des représentations apprises par un réseau de neurones convolutifs [98].

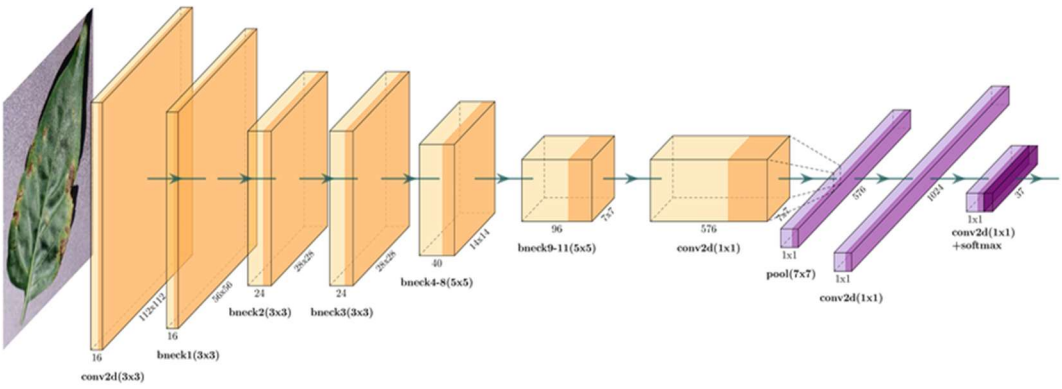


Figure 3.7 : Architecture de MobileNetV3 small [98]

3.7.4 Resnet

Le ResNet (Residual Network) est un type de réseau de neurones convolutifs profonds qui a été développé par Microsoft en 2015 [99, 100]. Il est conçu pour résoudre le problème de la dégradation de la précision des réseaux profonds, lorsque la profondeur augmente. L'idée des blocs résiduels (figure 3.8) a été introduite, pour traiter le problème du gradient de disparition. Une méthode connue sous le nom de connexions sautées dans ce réseau, y est appliquée. La connexion de saut contourne certains niveaux entre les activations des couches de liaison aux couches suivantes. Cela crée un bloc restant. Ces blocs restants sont empilés pour générer des 'resnets'.

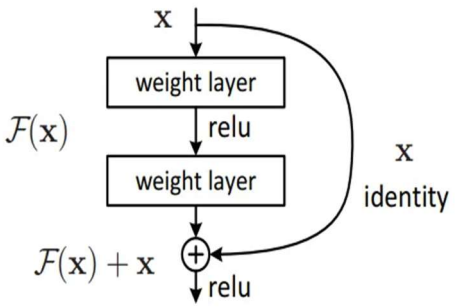


Figure 3.8 : Bloc résiduel [100]

Ce réseau facilite l'apprentissage en ajoutant les connexions résiduelles. Dans ce cadre la sortie résiduelle y est déterminée suivant l'équation 3.1.

$$y=F(x)+x \qquad 3.1$$

Où : F(x) est la transformation (résiduelle) et x l'entrée initiale (skip connection).

Plusieurs variantes de Resnet existent, dans ce projet, c'est le modèle Resnet50 (figure 3.9) qui est adopté. Il est composé d'une couche de convolution initiale, de 4 étages (stages) de blocs résiduels (avec bottleneck) et d'une couche de classification fully connected (FC).

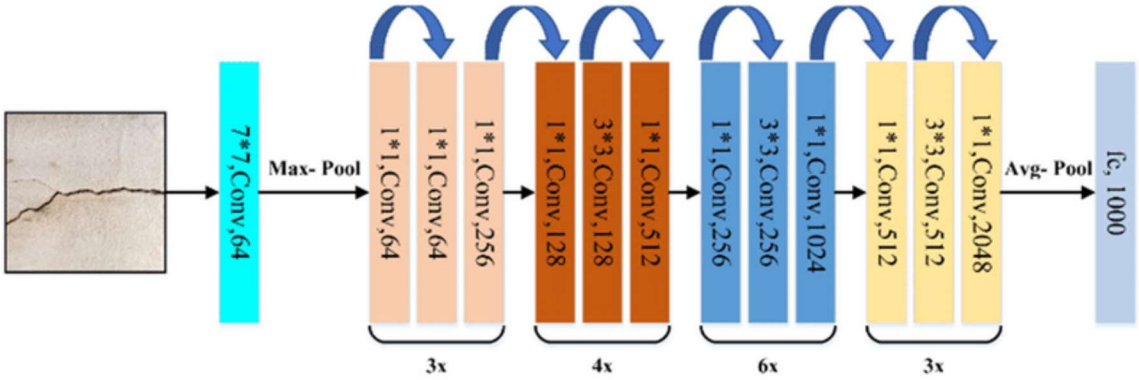


Figure 3.9 : Modèle Resnet50 [100]

3.7.4 Adaptation des modèles sélectionnés à la classification MLB

Pour adapter les modèles choisis à la classification binaire de l'état des cellules LTE (Good/Bad Case), la méthodologie du transfert d'apprentissage (transfer learning) y est considérée [101]. Cette approche, qui consiste à réutiliser un modèle pré-entraîné sur un grand jeu de données généraliste (comme ImageNet), permet de bénéficier des caractéristiques visuelles de haut niveau déjà apprises par le réseau, tout en adaptant ses couches finales à la tâche spécifique.

Les étapes principales de cette adaptation sont les suivantes :

- Réutilisation du corps du modèle pré-entraîné : La majeure partie du CNN entraînée sur ImageNet et reconnue pour sa capacité à extraire des caractéristiques pertinentes à partir d'images, est conservée.
- Suppression et remplacement de la coyche fully connected : La couche finale de la classification originale du modèle (conçue pour les 1000 classes d'ImageNet) est supprimée. Elle est remplacée par un nouveau bloc de classification, spécifiquement conçu pour l'étude explorée. Ce bloc se compose :

- ❖ D’une couche Global Average Pooling (GAP), qui réduit la dimensionnalité des cartes de caractéristiques, en calculant la moyenne de chaque carte.
- ❖ D’une couche Dropout dont le taux est un hyperparamètre optimisé.
- ❖ D’une couche dense (Fully Connected) avec une fonction d'activation ReLU (Rectified Linear Unit), et intégrant des régularisations L1 et L2, dont les coefficients sont également des hyperparamètres optimisés.
- ❖ D’une couche de sortie dense avec une fonction d'activation Sigmoïde, qui produit une probabilité unique comprise entre 0 et 1, indiquant l'appartenance aux classes considérées.

Pour affiner chacun des modèles sélectionnés aux spécificités des images de KPI LTE, une stratégie de fine-tuning est appliquée [78]. Les couches convolutives initiales du modèle sont conservées, afin d’exploiter les caractéristiques de bas niveau. Par la suite, les dernières couches du corps du modèle, ainsi que toutes les couches du bloc de classification ajouté, sont rendues entraînables. Cela permet au modèle d'adapter les caractéristiques de plus haut niveau et les poids de classification, aux motifs spécifiques présents dans les données étudiées.

3.7.5 Paramètres d'entraînement

Les paramètres d’entraînement [72, 74] utilisés dans la réalisation su système adopté, sont accordés par :

- **Batch (lot)** : Le lot représente un seul échantillon de la base des données.
- **Batch-Size** : La taille du lot est le nombre d'échantillons avec lesquels, les entrées du réseau de neurones sont alimentées en un seul pas.
- **Learning rate** : Le taux d'apprentissage est un paramètre très important utilisé dans l'optimiseur, pour indiquer la taille du pas. Ce taux d'apprentissage est généralement réglé entre 0,01 et 0,0001.
- **Epoch** : Epoch représente le cycle de passage de tous les échantillons de la base de données, à travers le réseau de neurones dans le processus de d'entraînement.
- **Pas par ’epoch’** : il correspond au nombre de la taille du lot pendant une époque. Il est calculé en divisant la longueur de l'ensemble de données, par la taille du lot utilisée.

Ces divers paramètres combinés renforcent la robustesse du modèle et sa capacité à maintenir des performances élevées, sur des données non observées, ce qui est essentiel pour une application fiable dans des environnements dynamiques, comme les réseaux LTE.

3.8 Métriques d'évaluation et critères de performance

Pour évaluer de manière rigoureuse la performance l'approche considérée, il est impératif d'utiliser un ensemble de métriques appropriées, allant au-delà de la simple précision globale (accuracy). En effet, dans le contexte de classes déséquilibrées, où les "Bad Cases" (classes minoritaires) sont intrinsèquement moins fréquents que les "Good Cases" (classes majoritaires), l'accuracy seule peut être trompeuse. Des métriques spécifiques sont nécessaires (définies dans les sous sections suivantes), pour garantir une évaluation juste de la capacité du modèle à détecter correctement la classe minoritaire, qui est d'un intérêt critique pour l'activation du MLB.

3.7.1 Matrice de confusion

La matrice de confusion est un tableau (figure 3.10) qui offre une vue d'ensemble des performances d'un modèle de classification, en présentant le nombre de prédictions correctes et incorrectes pour chaque classe [102].

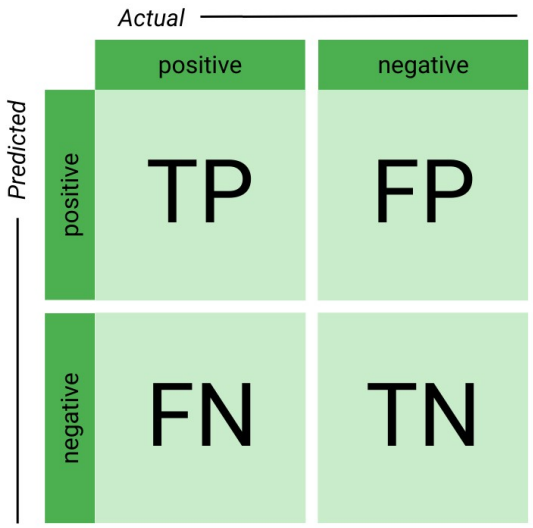


Figure 3.10 : Matrice de confusion [102]

- **Vrais positifs (TP) :** Nombre de "Bad Cases" (classe 1) correctement identifiés comme tels.
- **Vrais Négatifs (TN) :** Nombre de "Good Cases" (classe 0) correctement identifiés comme tels.

- **Faux Positifs (FP) :** Nombre de "Good Cases" (classe 0) qui ont été incorrectement classés comme "Bad Cases" (classe 1). C'est une erreur de type I.
- **Faux Négatifs (FN) :** Nombre de "Bad Cases" (classe 1) qui ont été incorrectement classés comme "Good Cases" (classe 0). C'est une erreur de type II, souvent la plus coûteuse dans le contexte de la détection de problèmes.

3.8.2 Précision

La précision mesure la fiabilité des prédictions positives du modèle [102]. Elle est calculée comme le pourcentage de vrais positifs parmi toutes les prédictions que le modèle réalise comme positives. Ce paramètre est exprimé par la formule 3.2.

$$\frac{TP}{TP+} \qquad 3.2$$

Une précision élevée indique que lorsque le modèle prédit un "Bad Case", il est très probable que ce soit réellement un "Bad Case".

3.8.3 Rappel

Le rappel (Recall / Sensibilité) mesure la capacité du modèle à identifier tous les cas positifs réels [102]. Il est calculé comme le pourcentage de vrais positifs, parmi tous les cas positifs réels présents dans l'ensemble de données (3.3).

Un rappel élevé est crucial dans notre contexte, car il indique que le modèle est capable de détecter une grande proportion des "Bad Cases" réels, minimisant ainsi les cas manqués, où le MLB doit être activé.

$$\frac{TP}{TP+FN} \qquad 3.3$$

3.8.4 F1-score

Le F1-score (3.4) est la moyenne harmonique de la précision et du rappel [102]. Il offre une métrique équilibrée, particulièrement utile lorsque l'on recherche un équilibre entre précision et rappel, ou en présence de classes déséquilibrées.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Reca} \qquad 3.4$$

Un F1-score élevé indique que le modèle a à la fois un rappel élevé (il trouve la plupart des cas pertinents) et une précision élevée (ses prédictions positives sont fiables).

3.8.5 Accuracy

L'accuracy (précision globale) représente le pourcentage de prédictions correctes sur le total des prédictions effectuées par le modèle [102]. Cette précision est exprimée par l'équation 3.5.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad 3.5$$

Bien qu'elle soit une métrique de base, elle est moins informative seule dans les scénarios de classes déséquilibrées, car un modèle prédisant toujours la classe majoritaire, pourrait afficher une accuracy élevée tout en étant incapable de détecter la classe minoritaire d'intérêt.

3.8.6 Courbe ROC et AUC

La **courbe ROC** est une représentation graphique de la performance d'un modèle de classification binaire, montrant le taux de vrais positifs (True Positive Rate ou Rappel) en fonction du taux de faux positifs (False Positive Rate), pour différents seuils de classification. L'**AUC** (Area Under the Curve) mesure l'aire totale sous cette courbe ROC. L'AUC fournit une métrique unique et synthétique de la capacité discriminante du modèle, c'est-à-dire sa capacité à distinguer les classes positives des classes négatives. Une valeur d'AUC proche de 1 indique une excellente performance de la classification, indépendamment du seuil de décision choisi, ce qui est particulièrement pertinent pour évaluer la robustesse d'un modèle [102].

3.9 Conclusion

Ce chapitre a présenté la méthodologie proposée pour la classification de l'état des cellules LTE, en vue de l'optimisation du Mobility Load Balancing (MLB), suivant les modèles des réseaux convolutifs pré-entraînés choisis.

Les étapes fondamentales, depuis la formulation précise de la tâche de classification binaire jusqu'à la préparation minutieuse des données, incluant leur nettoyage, leur normalisation et leur transformation en images synthétiques, ont été explorées.

L’étude conceptuelle a également abordé les stratégies du fine tuning pour les différentes architectures étudiées, en adaptation avec les données LTE. Enfin, les paramètres d'entraînement adéquats, ainsi que les métriques d'évaluation pertinentes ont été définis, pour garantir la robustesse et la fiabilité du modèle.

L'ensemble de cette démarche méthodologique, ancrée dans les principes de l'apprentissage profond et adaptée aux spécificités des réseaux LTE, vise à établir une base solide pour le développement d'une solution intelligente. Le prochain chapitre s'attachera à présenter et à analyser les résultats expérimentaux obtenus, validant ainsi l'efficacité de l'approche proposée.

Chapitre 4 Mise en œuvre du système de prédiction cellulaire LTE et résultats expérimentaux

4.1 Introduction

Ce chapitre se concentre sur la mise en œuvre pratique d'une solution d'automatisation de la prédiction Good/Bad Case pour le réseau LTE de Djezzy, en utilisant les techniques du deep learning. L'objectif principal est de concevoir, entraîner et évaluer six architectures des réseaux de neurones convolutifs pré-entraînés sur une base de données réelle non publique. Le système développé exploitera les données KPI (Key Performance Indicator) extraites des plateformes Huawei OSS (U2020 et PRS).

La démarche suivie selon l'approche adoptée (chapitre 3) pour mettre en œuvre les six modèles sélectionnés par, mobilenetV1, mobilenetV2, mobilenetV3 small, efficientnetB0, VGG16 et Resnet50, comprend plusieurs étapes clés :

- **Préparation des données :** elle consiste à effectuer le nettoyage, la normalisation et la transformation des données tabulaires, en images pour les rendre compatibles avec un réseau de neurones convolutionnel (CNN).
- **Configuration de l'environnement de développement :** cet environnement concerne l'utilisation de ressources locales (ordinateur personnel) et de plateformes cloud comme Google Colab et Kaggle, pour bénéficier de leurs capacités GPU, essentielles au Deep Learning.
- **Conception et Fine-Tuning partiel des modèles développés :** cette réalisation est basée sur la sélection rigoureuse des hyperparamètres et l'application de stratégies de régularisation pour prévenir le sur-apprentissage.
- **Entraînement et validation des modèles adoptés :** ce traitement est mis en œuvre par des techniques d'optimisation et d'évaluation statistique pour mesurer la performance de la classification binaire.
- **Analyse des résultats :** elle est obtenue par l'évaluation critique des performances du modèle, l'identification de ses forces et faiblesses, et la proposition de pistes d'amélioration.

Ce chapitre vise à démontrer la faisabilité de l'approche IA, pour automatiser la détection des cas critiques dans le cadre du Mobility Load Balancing (MLB) du réseau LTE de Djezzy.

4.2 Environnement de développement

Le développement et l'entraînement du modèle de classification ont été réalisés en combinant des ressources matérielles locales et des plateformes cloud, afin de tirer parti des avantages spécifiques de chaque environnement.

4.2.1 Matériel utilisé

L'environnement matériel local, utilisé pour le prétraitement des données et les premières expérimentations, ont été réalisées sur un ordinateur portable personnel Asus, avec les caractéristiques suivantes :

- Processeur : AMD Ryzen 5 4600H with Radeon Graphics, cadencé à 3.00 GHz (6 cœurs physiques / 12 threads).
- Mémoire vive (RAM) : 16,0 Go (dont 15,4 Go utilisable).
- Type de système : Système d'exploitation 64 bits, processeur x64.
- Carte graphique intégrée : AMD Radeon Graphics.
- Système d'exploitation : Windows 11 Professionnel.

Cet environnement a été essentiel pour la manipulation initiale des fichiers de données volumineux, la préparation des jeux de données, ainsi que les étapes de transformation des données tabulaires en images, un format adapté aux réseaux de neurones convolutionnels.

4.2.2 Langage de programmation

Le langage de programmation Python (version 3.9) a été choisi pour l'intégralité de ce projet, en raison de sa position dominante et de son écosystème robuste, pour l'intelligence artificielle et le Deep Learning. Effectivement, Python [103] offre une syntaxe claire et lisible, ce qui accélère le développement. Sa force réside également dans son vaste ensemble de bibliothèques spécialisées, couvrant la manipulation de données, la visualisation, le Machine Learning et le Deep Learning, ce qui en fait un choix idéal pour ce type de projet. La grande communauté et les nombreuses ressources disponibles, ont également été un atout majeur.

4.2.3 Bibliothèques et outils logiciels

Pour assurer le bon déroulement du projet, plusieurs bibliothèques et outils logiciels ont été sélectionnés, en fonction de leurs fonctionnalités spécifiques et de leur robustesse prouvée dans le domaine du traitement des données, du Machine Learning et du Deep Learning.

Ces outils ont permis une gestion efficace des données, une expérimentation rapide, ainsi qu'une modélisation performante.

1. **Pandas**

Cette bibliothèque est incontournable pour la manipulation des données tabulaires. Elle offre des structures de données flexibles (DataFrame, Series) permettant de nettoyer, filtrer, agréger et transformer les KPI extraits des fichiers Excel. Pandas facilite également l'import/export des données, ce qui a été essentiel pour le prétraitement avant la création des jeux de données d'entraînement [104].

2. **NumPy**

Utilisé principalement pour le traitement numérique, NumPy fournit des tableaux multidimensionnels performants et des fonctions mathématiques optimisées. Dans ce projet, NumPy a été exploité pour réaliser des opérations matricielles sur les données normalisées, préparer les matrices d'entrée, et effectuer des calculs statistiques indispensables avant la modélisation [105].

3. **Matplotlib**

Cet outil de visualisation a été utilisé pour générer des graphiques illustrant l'évolution des KPI, comme le PRB, le trafic et débit, sur différentes périodes. Par ailleurs, Matplotlib a servi à tracer les courbes d'apprentissage du modèle (précision, perte) pendant les phases d'entraînement et de validation, permettant ainsi de suivre la convergence et de détecter d'éventuels phénomènes de surapprentissage [106].

4. **Scikit-learn**

Bien que l'approche principale repose sur le Deep Learning, Scikit-learn a joué un rôle clé dans plusieurs étapes. Cette bibliothèque a été utilisée pour effectuer la normalisation Z-Score des données avant leur entrée dans le réseau de neurones, garantir une échelle cohérente entre les variables, et calculer des métriques d'évaluation standards (matrice de confusion, rapport de classification, précision, rappel, F1-score). Ces métriques ont permis d'évaluer la qualité du modèle et de comparer différentes configurations [107].

5. **TensorFlow / Keras**

Le framework TensorFlow [108], associé à son API de haut niveau Keras, a constitué la colonne vertébrale de la modélisation.

Keras [109] a offert une interface simple et modulaire pour construire, entraîner et évaluer les modèles adaptés à notre tâche de classification Good/Bad Case. Les fonctions d'import de modèles pré-entraînés, de personnalisation des couches, et la gestion avancée des callbacks ont permis de mettre en œuvre efficacement le Fine-Tuning partiel.

6. Keras Tuner

L'optimisation des hyperparamètres est cruciale pour maximiser la performance du modèle. Keras Tuner [110], outil spécialisé pour cette tâche, a été utilisé pour rechercher automatiquement la meilleure combinaison des paramètres d'entraînement (taux d'apprentissage, nombre de couches gelées, taille de batch, etc.) via une recherche aléatoire (Random Search). Cette étape a amélioré la robustesse et la généralisation du modèle.

7. Outils d'expérimentation

Pour l'entraînement et l'expérimentation des modèles de Deep Learning, deux environnements cloud basés sur Jupyter Notebooks [111], ont été principalement utilisés : Google Colab et Kaggle Notebooks. Ces plateformes offrent un accès gratuit à des ressources GPU, indispensables pour traiter les grands volumes de données et exécuter efficacement les phases d'entraînement.

a. Google Colab

Google Colab [112] est une plateforme très populaire offrant un accès gratuit à des GPU, comme Tesla T4 ou P100. Elle permet d'exécuter des notebooks Python dans un environnement cloud, avec une intégration facile de bibliothèques relatives au Deep Learning. Cette plateforme offre un accès facile, de nombreuses ressources d'aide et une exécution fluide avec GPU. Cependant, les sessions sont limitées à environ 12 heures consécutives, d'où la nécessité de reconnecter le notebook après l'interruption.

b. Kaggle Notebooks

Kaggle [113] propose également un environnement de notebooks en ligne avec accès gratuit à des GPU Tesla P100, particulièrement apprécié pour les compétitions et projets de Data Science. Les sessions pouvant durer jusqu'à 12 heures, avec une limite d'utilisation cumulée d'environ 30 heures d'exécution par semaine, ce qui est suffisant pour des phases d'entraînement prolongées. L'intégration avec les datasets Kaggle, facilite le chargement direct des données. Cependant, comparé à Colab, les ressources GPU sont parfois moins flexibles.

La combinaison de ces deux environnements a permis d'optimiser les temps d'entraînement : Google Colab pour des exécutions rapides et interactives et Kaggle pour des sessions plus longues et plus stables, grâce à sa limite hebdomadaire étendue.

Cette dualité garantit ainsi un bon équilibre entre l'accessibilité, la performance et la flexibilité dans la gestion des ressources GPU nécessaires au projet.

4.3 Extraction des données de supervision depuis le 'PRS'

L'extraction des données de supervision, a nécessité l'exploration de la plateforme PRS Huawei, avec un choix de 5 indicateurs KPI.

4.3.1 Présentation de la plateforme PRS Huawei

Le PRS (Performance Report System) est un module logiciel intégré à l'OSS Huawei [85], utilisé par Djezzy pour la supervision, l'analyse et le rapport des performances réseau LTE.

Grâce à son interface graphique intuitive et ses nombreuses fonctionnalités de filtrage, le 'PRS' permet aux équipes d'ingénierie, d'accéder à une grande variété d'indicateurs clés de performance KPI (Key Performance Indicators) collectés directement depuis les équipements radio (eNodeB), via le serveur U2020.

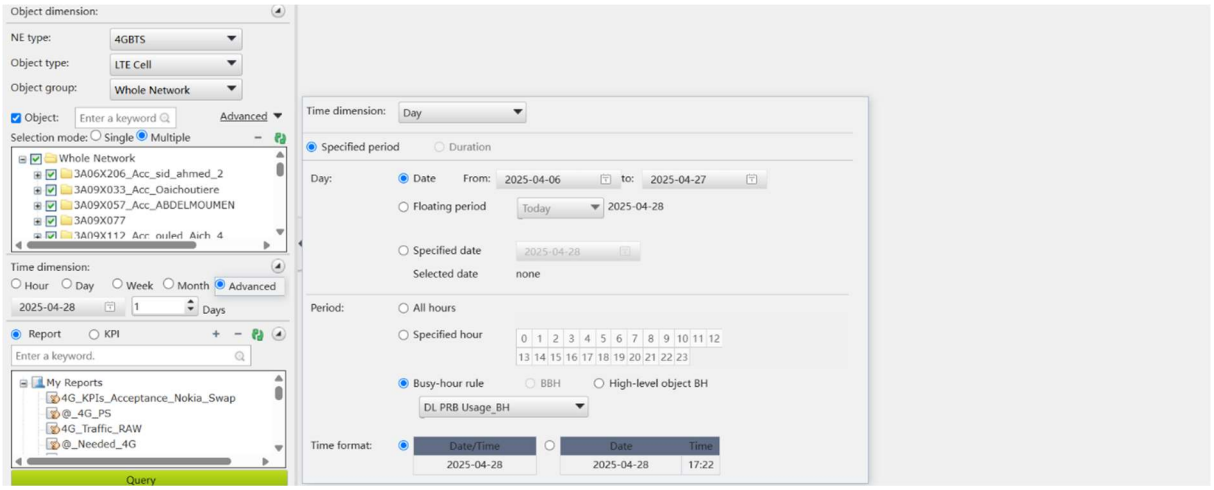


Figure 4.1 : Capture d'écran de l'interface PRS

4.3.2 Types de KPI disponibles

Le 'PRS' offre l'accès à **plusieurs centaines de KPI** couvrant :

- La qualité de service (QoS).
- Le trafic utilisateur.
- L'utilisation des ressources radio (PRB, RRC, ERAB, etc.).

- Les indicateurs de mobilité
- Les alarmes et dégradations

Cependant, dans le cadre de cette étude orientée MLB (Mobility Load Balancing), sous la supervision du service 'TRAFIC', nous avons limité l'extraction aux KPI suivants :

- **Taux de ressources physiques radio (PRB) : il r**eprésente le taux d'utilisation des ressources physiques radio par cellule).
- **Débit (Throughput) : il indique l'**Indicateur du débit moyen ou maximal en downlink, par cellule.
- **Trafic du volume de données :** il représente le volume total de données transporté sur la cellule pendant la période.
- **Nombre d'utilisateurs connectés (Number of Connected Users (UE)) :** il correspond au nombre moyen d'utilisateurs simultanément connectés par cellule.
- **Timing Advance (TA) :** il indique la mesure approximative de la distance moyenne des utilisateurs à la station de base.

Ces choix sont directement liés aux critères utilisés, dans le processus de décision du MLB chez Djezzy.

4.3.3 Mode de collecte et fréquence de mise à jour

Le système PRS reçoit les compteurs des 'eNodeB', via le serveur U2020 (tableau 4.1). Les compteurs sont mis à jour automatiquement toutes les 15 minutes (période dite Quarter Hour).

Le système PRS consolide ces compteurs, en statistiques exploitables sous différents formats et niveaux de granularité.

Type de statistique	**Période**	**Utilisation typique**
Statistiques horaires	Données agrégées par heure	Suivi de la charge horaire et des Busy Hours
Statistiques quotidiennes	Moyenne ou somme par jour	Analyse de la performance journalière
Statistiques mensuelles	Consolidation sur le mois	Étude de la tendance à long terme

Tableau 4.1 : Paramétrage des compteurs

Dans notre cas une extraction globale sur l'ensemble du réseau LTE Djezzy, a été effectuée sur une période de 6 mois, où chaque fichier contient 15 jours, afin de disposer d'un large jeu de données brut.

4.4 Traitement et préparation des données sous Excel

Les données collectées, sont présentées à l'état brut, ce qui a nécessité plusieurs étapes, en vue de leur préparation.

4.4.1 Objectif du traitement manuel

Avant d'entraîner un modèle de Deep Learning pour la classification Good/Bad Case, il a été nécessaire de créer un jeu de données propre et labellisé. Cette étape de préparation a été réalisée manuellement sous Excel, à partir des fichiers d'extraction brute issus du PRS Huawei, contenant les KPI de l'ensemble du réseau LTE de Djezzy. L'objectif était de passer de données brutes, à une base finale sectorisée et labellisée, comportant, pour chaque secteur et chaque fréquence LTE, les valeurs des KPI clés ainsi qu'un label binaire (Good ou Bad Case), utilisé ensuite pour l'entraînement du modèle.

4.4.2 Format initial des données PRS (Fichier brut)

Les données extraites du système PRS sont fournies sous forme de fichiers Excel (.xls/.xlsx). Chaque fichier couvre 15 jours de données et plusieurs fichiers, ont été extraits afin de couvrir une période totale de 6 mois. Chaque feuille Excel contient les KPI pour toutes les cellules du réseau LTE (figure 4.2).

Date	eNodeB Name	eNodeB Function Name	Cell Name	LocalCell Id	Cell FDD TDD Indication	Integrity	DL_PRB_Utilization(%)	Average_TA(Km)	4G_User_Throughput(kbps)_vmp(kbit/s)	Cell Traffic Volume (DL+UL)(GB)	L.Traffic.ActiveUser.Avg
02/03/2025	MBTS_A16X1	4A16X113	4A16X113_1	0	CELL_FDD	100%	57,7205	0,5881	8802,5237	132,16	3,4604
02/03/2025	MBTS_A16X1	4A16X113	4A16X113_2	1	CELL_FDD	100%	57,5302	0,6196	6921,3765	114,2695	3,3136
02/03/2025	MBTS_A16X1	4A16X113	4A16X113_3	2	CELL_FDD	100%	42,0053	0,6197	9984,5253	62,7337	1,3541
02/03/2025	MBTS_C25X1	4C25X105	4C25X105_2	1	CELL_FDD	100%	37,4487	0,457	8261,9816	47,9382	0,9999
02/03/2025	MBTS_C25X1	4C25X105	4C25X105_1	0	CELL_FDD	100%	47,184	0,3644	7899,1262	81,8097	1,9345
02/03/2025	MBTS_C25X0	4C25X023	4C25X023_1	0	CELL_FDD	100%	66,1246	0,6138	3620,9069	97,6427	4,1443
02/03/2025	MBTS_C25X0	4C25X023	4C25X023_2	1	CELL_FDD	100%	66,1237	0,5822	4292,1262	115,0165	3,9765
02/03/2025	MBTS_C25X0	4C25X023	4C25X023_3	2	CELL_FDD	100%	25,3873	0,429	12941,8922	40,9852	0,6859
02/03/2025	MBTS_C25X0	4C25X064	4C25X064_3	2	CELL_FDD	100%	43,805	0,4129	6860,5405	52,1489	1,1918
02/03/2025	MBTS_C25X0	4C25X064	4C25X064_2	1	CELL_FDD	100%	63,9576	0,4933	5889,6485	120,3536	3,2631
02/03/2025	MBTS_C25X0	4C25X064	4C25X064_1	0	CELL_FDD	100%	44,915	0,5401	7036,3052	50,8281	1,1835
02/03/2025	MBTS_C25X0	4C25X019	4C25X019_1	0	CELL_FDD	100%	87,1229	0,4682	2444,6421	195,9397	11,9778
02/03/2025	MBTS_C25X0	4C25X019	4C25X019_3	2	CELL_FDD	100%	67,6179	0,5747	6034,1982	157,1964	4,3323
02/03/2025	MBTS_C25X0	4C25X012	4C25X012_1	0	CELL_FDD	100%	59,6642	0,261	7418,0859	132,3123	3,1001
02/03/2025	MBTS_C25X0	4C25X012	4C25X012_2	1	CELL_FDD	100%	77,5108	0,637	3797,8646	115,0211	4,2039
02/03/2025	MBTS_C25X0	4C25X012	4C25X012_3	2	CELL_FDD	100%	38,7454	0,3867	9135,3405	59,4258	1,2762
02/03/2025	MBTS_C25X0	4C25X025	4C25X025_1	0	CELL_FDD	100%	34,0489	0,3998	13347,353	65,397	1,1894
02/03/2025	MBTS_C25X0	4C25X025	4C25X025_2	1	CELL_FDD	100%	65,9944	0,756	5340,562	121,1011	3,5612
02/03/2025	MBTS_C25X0	4C25X025	4C25X025_3	2	CELL_FDD	100%	24,6085	0,5398	12728,3476	32,3343	0,6265
02/03/2025	MBTS_C25X0	4C25X028	4C25X028_1	0	CELL_FDD	100%	57,6976	0,4047	6011,0028	92,3476	2,431
02/03/2025	MBTS_C25X0	4C25X028	4C25X028_3	2	CELL_FDD	100%	68,8571	0,4735	4154,2429	132,4933	4,9916
02/03/2025	MBTS_C25X0	4C25X071	4C25X071_1	0	CELL_FDD	100%	46,9577	0,5101	7533,7674	73,8285	1,6959
02/03/2025	MBTS_C25X0	4C25X071	4C25X071_2	1	CELL_FDD	100%	49,9372	0,7493	6829,7821	73,1483	1,7054
02/03/2025	MBTS_C25X0	4C25X071	4C25X071_3	2	CELL_FDD	100%	34,0557	0,685	7676,2772	39,7892	1,0088

Figure 4.2 : fichier extrait du système PRS

4.4.3 Codification des cellules et identification des secteurs

Chez l'opérateur Djezzy, chaque cellule LTE suit une codification interne, qui permet d'identifier son secteur d'appartenance. Le tableau 4.2 montre un exemple typique d'une cellule LTE.

Nom de la cellule LTE : 4A16X113_1	Code
''4''	4G
''A''	Région centre ("O" pour ouest et "C" pour est)
"16"	Code de la wilaya Alger
"_1"	Fréquence et secteur

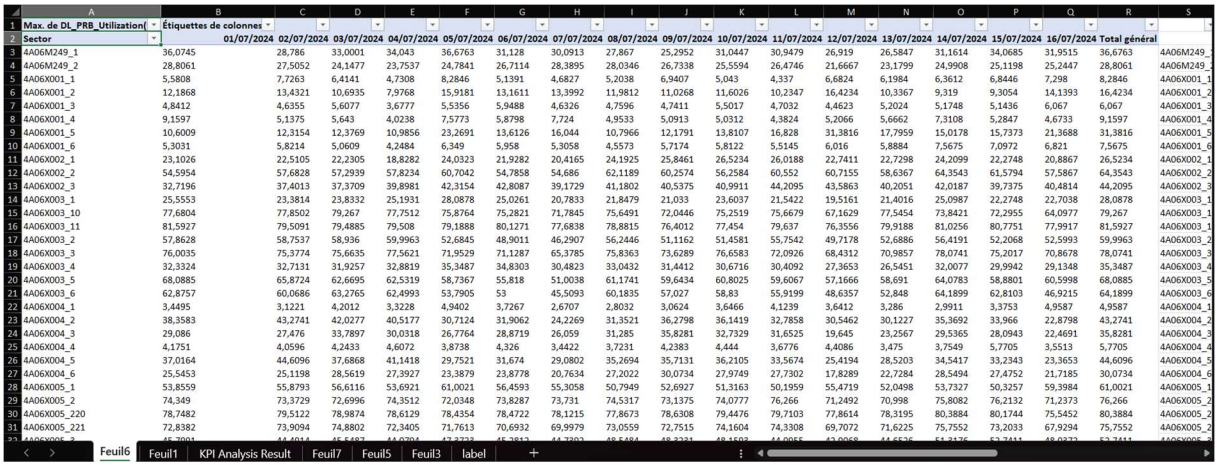
Tableau 4.2 : Codification d'un exemple d'une cellule LTE

4.4.4 Étapes de traitement sous Excel

Le traitement manuel s'est déroulé en plusieurs phases :

1. **Identification des cellules les plus chargées (TCD PRB max)**

Cette identification s'est faite par la création d'un tableau croisé dynamique (TCD) pour représenter les cellules ayant le maximum de PRB utilisé quotidiennement (figure 4.3). Dans ce cadre, les lignes sont indiquées par '**Cell Name**', les colonnes par '**Date**' et la valeur correspond à Max PRB en % (figure 4.3).



Max. de DL_PRB_Utilization(	Étiquettes de colonnes																		
Sector	01/07/2024	02/07/2024	03/07/2024	04/07/2024	05/07/2024	06/07/2024	07/07/2024	08/07/2024	09/07/2024	10/07/2024	11/07/2024	12/07/2024	13/07/2024	14/07/2024	15/07/2024	16/07/2024	Total général		
4A06M249_1	36,0745	28,786	33,0001	34,043	36,6763	31,128	30,0913	27,867	25,2952	31,0447	30,9479	26,919	26,5847	31,1614	34,0685	31,9515	36,6763	4A06M249_1	
4A06M249_2	28,8061	27,5052	24,1477	23,7537	24,7841	26,7114	28,3895	28,0346	26,7338	25,5594	26,4746	21,6667	23,1799	24,9908	25,1198	25,2447	28,8061	4A06M249_2	
4A06X001_1	5,5808	7,7263	6,4141	4,7308	8,2846	5,1391	4,6827	5,2038	6,9407	5,043	4,337	6,6824	6,1984	6,3612	6,8446	7,298	8,2846	4A06X001_1	
4A06X001_2	12,1868	13,4321	10,6935	7,9768	15,9181	13,1611	13,3992	11,9812	11,0268	11,6026	10,2347	16,4234	10,3367	9,319	9,3054	14,1393	16,4234	4A06X001_2	
4A06X001_3	4,8412	4,6355	5,6077	3,6777	5,5356	5,9488	4,6326	4,7596	4,7411	5,5017	4,7032	4,4623	5,2024	5,1748	5,1436	6,067	6,067	4A06X001_3	
4A06X001_4	9,1597	5,1375	5,643	4,0238	7,5773	5,8798	7,724	4,9533	5,0913	5,0312	4,3824	5,2066	5,6662	7,3108	5,2847	4,6733	9,1597	4A06X001_4	
4A06X001_5	10,6009	12,3154	12,3769	10,9856	23,2691	13,6126	16,044	10,7966	12,1791	13,8107	16,828	31,3816	17,7959	15,0178	15,7373	21,3688	31,3816	4A06X001_5	
4A06X001_6	5,3031	5,8214	5,0609	4,2484	6,349	5,958	5,3058	4,5573	5,7174	5,8122	5,5145	6,016	5,8884	7,5675	7,0972	6,821	7,5675	4A06X001_6	
4A06X002_1	23,1026	22,5105	22,2305	18,8282	24,0323	21,9282	20,4165	24,1925	25,8461	26,5234	26,0188	22,7411	22,7298	24,2099	22,2748	20,8867	26,5234	4A06X002_1	
4A06X002_2	54,5954	57,6828	57,2939	57,8234	60,7042	54,7858	54,686	62,1189	60,2574	56,2584	60,552	60,7155	58,6367	64,3543	61,5794	57,5867	64,3543	4A06X002_2	
4A06X002_3	32,7196	37,4013	37,3709	39,8981	42,3154	42,8087	39,1729	41,1802	40,5375	40,9911	44,2095	43,5863	40,2051	42,0187	39,7375	40,4814	44,2095	4A06X002_3	
4A06X003_1	25,5553	23,3814	23,8332	25,1931	28,0878	25,0261	20,7833	21,8479	21,033	23,6037	21,5422	19,5161	21,4016	25,0987	22,2748	22,7038	28,0878	4A06X003_1	
4A06X003_10	77,6804	77,8502	79,267	77,7512	75,8764	75,2821	71,7845	75,6491	72,0446	75,2519	75,6679	67,1629	77,5454	73,8421	72,2955	64,0977	79,267	4A06X003_1	
4A06X003_11	81,5927	79,5091	79,4885	79,508	79,1888	80,1271	77,6838	78,8815	76,4012	77,454	79,637	76,3556	79,9188	81,0256	80,7751	77,9917	81,5927	4A06X003_1	
4A06X003_2	57,8628	58,7537	58,936	59,9963	52,6845	48,9011	46,2907	56,2446	51,1162	51,4581	55,7542	49,7178	52,6886	56,4191	52,2068	52,5993	59,9963	4A06X003_2	
4A06X003_3	76,0035	75,3774	75,6635	77,5621	71,9529	71,1287	65,3785	75,8363	73,6289	76,6583	72,0926	68,4312	70,9857	78,0741	75,2017	70,8678	78,0741	4A06X003_3	
4A06X003_4	32,3324	32,7131	31,9257	32,8819	35,3487	34,8303	30,4823	33,0432	31,4412	30,6716	30,4092	27,3653	26,5451	32,0077	29,9942	29,1348	35,3487	4A06X003_4	
4A06X003_5	68,0885	65,8724	62,6695	62,5319	58,7367	55,818	51,0038	61,1741	59,6434	60,8025	59,6067	57,1666	58,691	64,0783	58,8801	60,5998	68,0885	4A06X003_5	
4A06X003_6	62,8757	60,0686	63,2765	62,4993	53,7905	53	45,5093	60,1835	57,027	58,83	55,9199	48,6357	52,848	64,1899	62,8103	46,9215	64,1899	4A06X003_6	
4A06X004_1	3,4495	3,1221	4,2012	3,3228	4,9402	3,7267	2,6707	2,8032	3,0624	3,6466	4,1239	3,6412	3,286	2,9911	3,3753	4,9587	4,9587	4A06X004_1	
4A06X004_2	38,3583	43,2741	42,0277	40,5177	30,7124	31,9062	24,2269	31,3521	36,2798	36,1419	32,7858	30,5462	30,1227	35,3692	33,966	22,8798	43,2741	4A06X004_2	
4A06X004_3	29,086	27,476	33,7897	30,0318	26,7764	28,8719	26,059	31,285	35,8281	32,7329	31,6525	19,645	23,2567	29,5365	28,0943	22,4691	35,8281	4A06X004_3	
4A06X004_4	4,1751	4,0596	4,2433	4,6072	3,8738	4,326	3,4422	3,7231	4,2383	4,444	3,6776	4,4086	3,475	3,7549	5,7705	3,5513	5,7705	4A06X004_4	
4A06X004_5	37,0164	44,6096	37,6868	41,1418	29,7521	31,674	29,0802	35,2694	35,7131	36,2105	33,5674	25,4194	28,5203	34,5417	33,2343	23,3653	44,6096	4A06X004_5	
4A06X004_6	25,5453	25,1198	28,5619	27,3927	23,3879	23,8778	20,7634	27,2022	30,0734	27,9749	27,7302	17,8289	22,7284	28,5494	27,4752	21,7185	30,0734	4A06X004_6	
4A06X005_1	53,8559	55,8793	56,6116	53,6921	61,0021	56,4593	55,3058	50,7949	52,6927	51,3163	50,1959	55,4719	52,0498	53,7327	50,3257	59,3984	61,0021	4A06X005_1	
4A06X005_2	74,349	73,3729	72,6996	74,3512	72,0348	73,8287	73,731	74,5317	73,1375	74,0777	76,266	71,2492	70,998	75,8082	76,2132	71,2373	76,266	4A06X005_2	
4A06X005_220	78,7482	79,5122	78,9874	78,6129	78,4354	78,4722	78,1215	77,8673	78,6308	79,4476	79,7103	77,8614	78,3195	80,3884	80,1744	75,5452	80,3884	4A06X005_2	
4A06X005_221	72,8382	73,9094	74,8802	72,3405	71,7613	70,6932	69,9979	73,0559	72,7515	74,1604	74,3308	69,7072	71,6225	75,7552	73,2033	67,9294	75,7552	4A06X005_2	

Feuil6 | Feuil1 | KPI Analysis Result | Feuil7 | Feuil5 | Feuil3 | label

Figure 4.3 : TCD /Max PRB

2. **Sélection des 3 cellules les plus chargées par secteur et par jour**

Pour chaque jour, les 3 cellules avec le plus fort taux de PRB par secteur, ont été sélectionnées. Ceci permet de concentrer l'analyse sur les cellules les plus impactées par la charge.

3. Récupération des autres KPI (Throughput, Traffic, UE, TA)

La récupération des KPI établis par : **'Throughput', 'Traffic', (UE) et (TA)** ont été récupérées pour chacune des 3 cellules, suivant excel. Une moyenne des 3 valeurs (par KPI), a été ensuite calculée pour chaque secteur et chaque jour (figure 4.4).

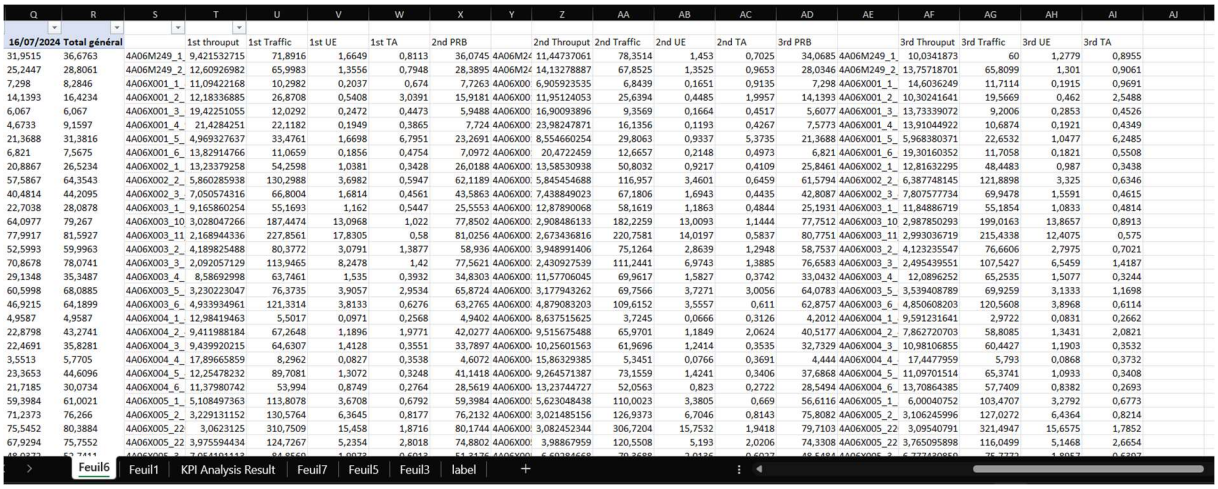


Figure 4.4 : Récupération des autres KPI (Throughput, Traffic, UE, TA)

4. Regroupement par secteur et par fréquence

Ce regroupement s'est fait par la création d'un second **tableau croisé dynamique**, pour obtenir :

- **En lignes** : les secteurs.
- **En colonnes** : les fréquences LTE (L1800, L2100, L2300, L900).
- **En valeurs** : la moyenne des KPI.

5. Calcul des écarts inter-fréquences (deltas).

Pour chaque secteur, des écarts (deltas) ont été calculés entre les fréquences, afin d'analyser les déséquilibres de charge et de performance (essentiel pour la logique MLB). Ces deltas ont concerné : PRB, 'Throughput', 'Traffic', UE et TA.

6. Attribution du label Good/Bad Case

En se basant sur les règles de décision (M Test 1 et M Test 2) validées par le service Trafic, chaque secteur (figure 4.5) a été labellisé manuellement en :

- **Good Case** : nécessite le MLB
- **Bad Case** : pas de déséquilibre (ne nécessite pas le MLB)

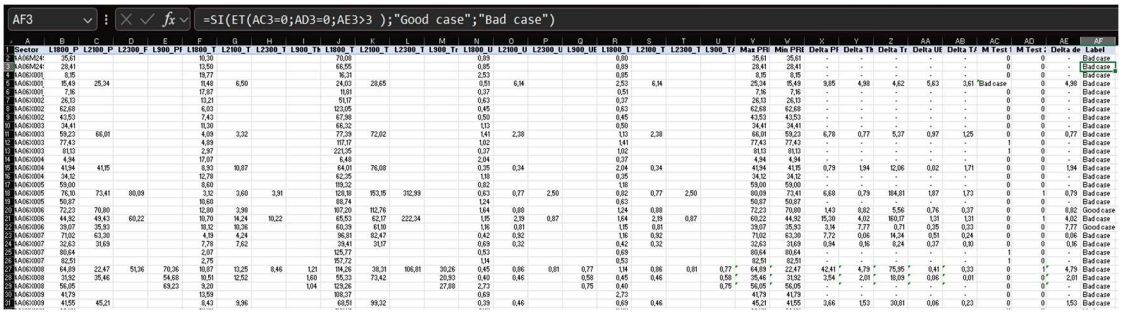


Figure 4.5 : Exemple de labellisation par secteur

4.5 Prétraitement et transformation des données sous Python

Après avoir obtenu la base de données finale labellisée sous Excel, la phase suivante a consisté à préparer ces données, pour les rendre exploitables par un modèle de Deep Learning. Cette étape de prétraitement a été réalisée en Python, en utilisant des bibliothèques spécialisées pour la manipulation de données et le traitement numérique.

4.5.1 Fusion des fichiers Excel

Les données issues du système PRS étaient réparties sur plusieurs fichiers Excel, chacun contenant 15 jours de mesures. Afin de constituer un seul jeu de données global, tous les fichiers ont été fusionnés en un unique 'DataFrame' via Pandas.

4.5.2 Sélection des colonnes utiles

Le jeu de données brut contenait de nombreuses colonnes, mais seules les variables pertinentes pour notre étude ont été conservées. Ces variables incluaient :

- Les KPI liés à l'utilisation des ressources et aux performances (PRB, Throughput, Traffic, UE, TA).
- Les deltas calculés entre les fréquences.
- Le label final « Good Case » ou « Bad Case ».
- L'identifiant du secteur (encodé dans la colonne « Sector »).

Cette étape de filtrage a permis de réduire la dimensionnalité du jeu de données et, de se concentrer uniquement sur les informations nécessaires à la classification.

4.5.3 Gestion des valeurs manquantes

Certaines lignes présentaient des valeurs manquantes, en particulier au niveau des labels (notamment pour les secteurs, où aucun label n'avait pu être attribué manuellement).

Les lignes sans label ont été supprimées, car elles ne pouvaient pas être utilisées pour un apprentissage supervisé. Quant aux valeurs manquantes dans les KPI, elles ont été remplacées par la moyenne de la colonne correspondante, afin d'éviter toute perturbation lors de l'entraînement du modèle.

4.5.4 Encodage des secteurs (One-Hot Encoding)

La variable catégorielle « Sector », qui identifie chaque secteur du réseau, a été encodée en utilisant la méthode 'One-Hot Encoding'. Cette transformation a permis de convertir les identifiants de secteurs, en variables numériques binaire (0/1), rendant l'ensemble du jeu de données pleinement compatible avec le réseau de neurones.

4.5.5 Normalisation des KPI

Pour garantir que toutes les variables d'entrée soient sur une échelle comparable, une normalisation Min-Max a été appliquée aux colonnes KPI et aux deltas. Cette transformation a permis de ramener toutes les valeurs dans l'intervalle [0,1], évitant ainsi que certaines variables ne dominent l'apprentissage du modèle, à cause d'une échelle trop grande.

4.5.6 Équilibrage des classes

L'analyse initiale a révélé un fort déséquilibre entre les classes (tableau 4.3).

.

Classe	Nombre d'échantillons
Bad Case	20 487
Good Case	6 982

Tableau 4.3 : Nombre d'échantillons initial

Pour éviter que le modèle n'apprenne à privilégier la classe majoritaire, un équilibrage par sous-échantillonnage a été réalisé. La classe majoritaire (Bad Case) a été réduite pour avoir le même effectif que la classe minoritaire (Good Case), soit 6 982 exemples dans chaque classe.

4.5.7 Transformation des données en images

Dans le cadre de ce projet, nous avons choisi de transformer chaque observation (ligne du DataFrame) en une image de petite dimension (3×3 pixels). Chaque pixel encode la valeur normalisée d'un KPI ou d'un delta. Ce choix a permis d'exploiter la puissance des réseaux de neurones convolutionnels (CNN) adoptés.

Le processus de génération a produit 13 964 images (6 982 Good Case + 6 982 Bad Case), chacune, stockée au format PNG (figure 4.6).

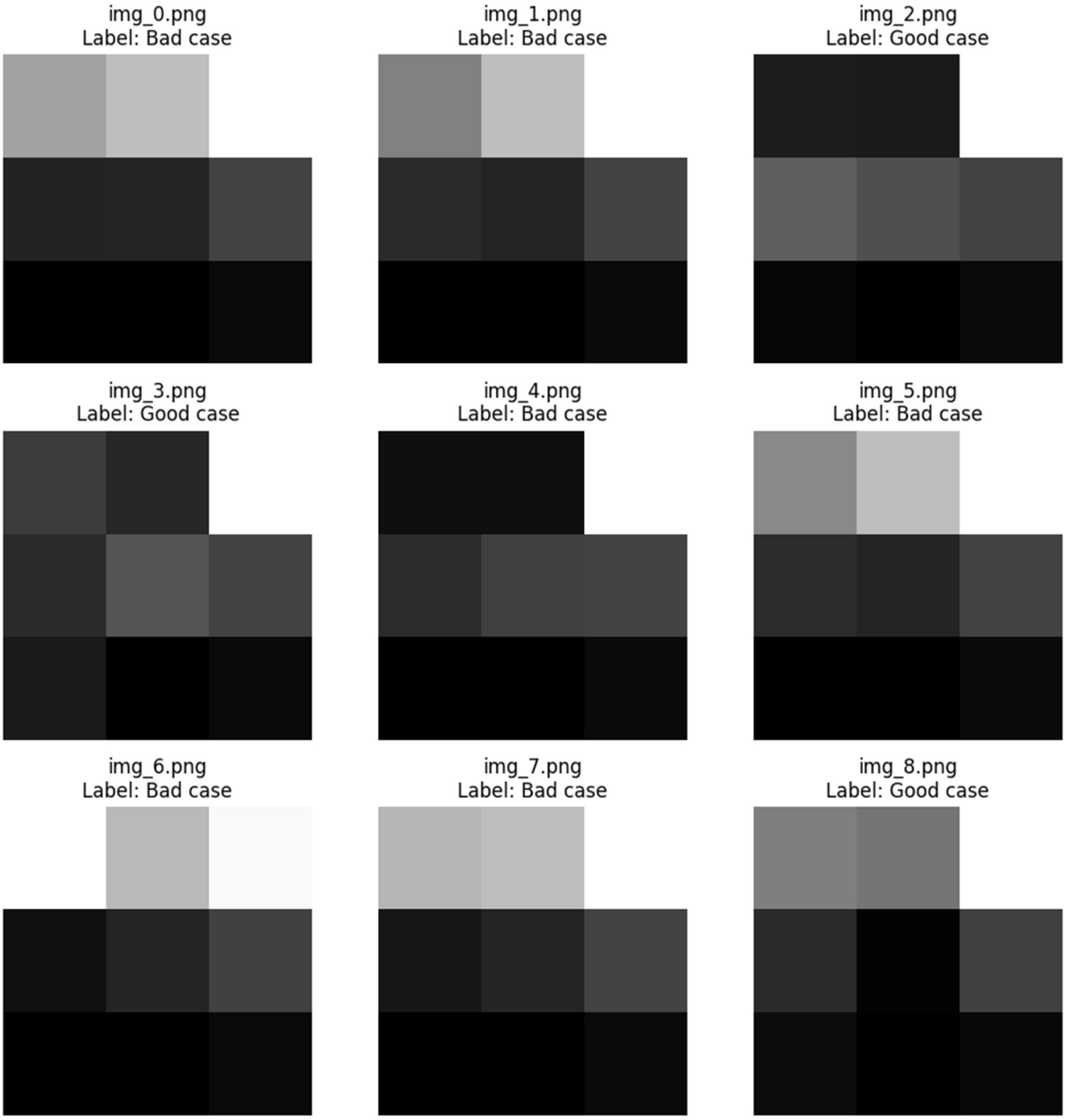


Figure 4.6 : Images converties labellisées

4.5.8 Création du fichier d'annotation

Un fichier CSV **(labels.csv)** a ensuite été généré, pour assurer la correspondance entre chaque image et son label (Good Case ou Bad Case). Ce fichier contient deux colonnes :

- **filename** : Le nom du fichier image (ex : img_0.png, img_1.png, …)
- **Label** : La classe associée (Good Case ou Bad Case)

4.5.9 Organisation finale du dataset

Enfin, les images ont été triées et réparties dans deux sous-dossiers distincts :

- dataset/Good case/
- dataset/Bad case/

Cette organisation facilite leur exploitation ultérieure avec les outils Keras ou TensorFlow, en particulier grâce aux fonctions de génération de lots d'images (ImageDataGenerator, etc).

À l'issue de ce prétraitement, l'ensemble des données était prêt à être utilisé, pour l'entraînement du modèle de classification via un processus de Fine-Tuning partiel, basé sur les modèles choisis.

4.6 Architecture des modèles testés

Dans cette étude, six architectures CNN pré-entraînées sur ImageNet ont été explorées pour la classification binaire Good/Bad Case, en tirant parti de l'apprentissage par transfert et du fine-tuning partiel. Ces modèles ont été sélectionnés en raison de leur popularité, leur performance reconnue et leur pertinence pour des tâches de reconnaissance d'images.

Pour chaque modèle, une stratégie de fine-tuning partiel a été appliquée, afin de tirer parti des connaissances acquises sur ImageNet, tout en adaptant le réseau aux spécificités des images générées à partir des données LTE. Dans ce cas, les réseaux de neurones considérés par MpbilenetV1 (figure 4.7), MobilenetV2 (figure 4.8), MobilenetV3 small (figure 4.9), EfficientnetB0 (figure 4.10), Resnet50 (figure 4.11) et VGG16 (figure 4.12) présentent le fine tuning suivant :

- Les premières couches, qui extraient des caractéristiques génériques (contours, textures), sont gelées (non entraînables).
- Les couches supérieures, plus spécifiques, sont dégelées pour être réentraînées sur le du jeu de données du réseau LTE transformé en images.

Cette approche permet de réduire les risques de surapprentissage, tout en profitant des connaissances acquises lors de l'entraînement sur ImageNet.

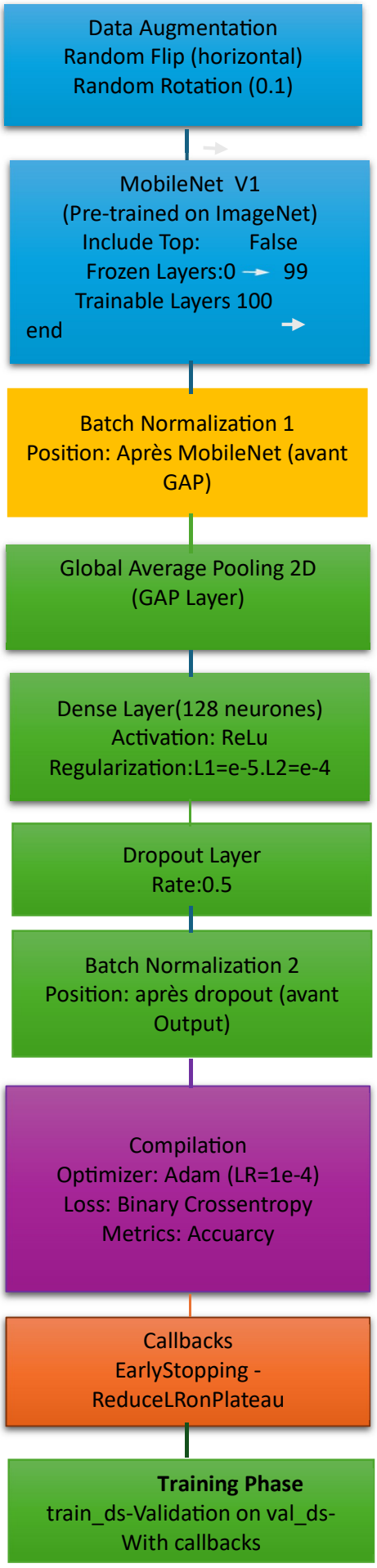


Figure 4.7 MobilenetV1 affiné

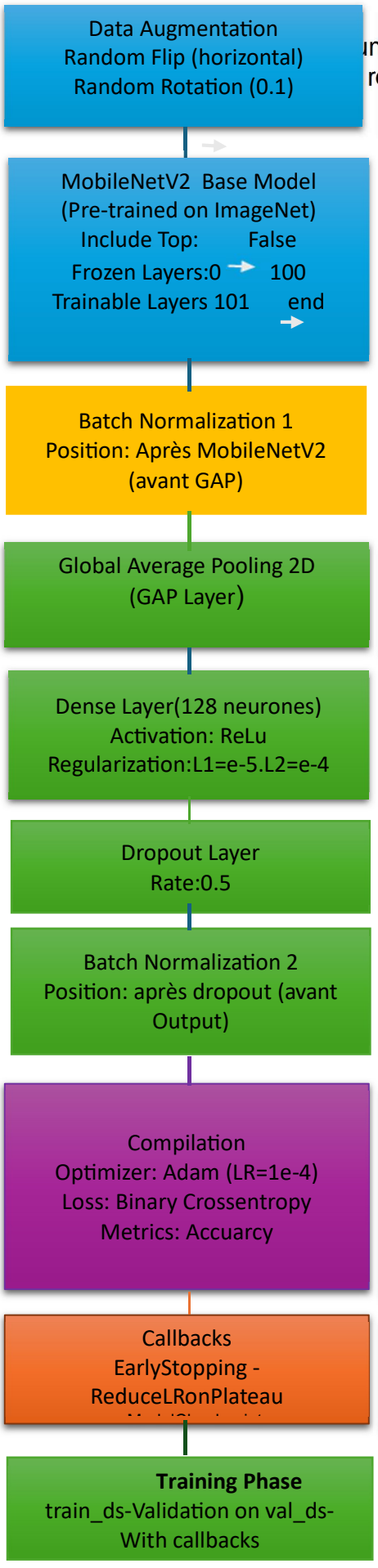


Figure 4.8 : MobilenetV2 affiné

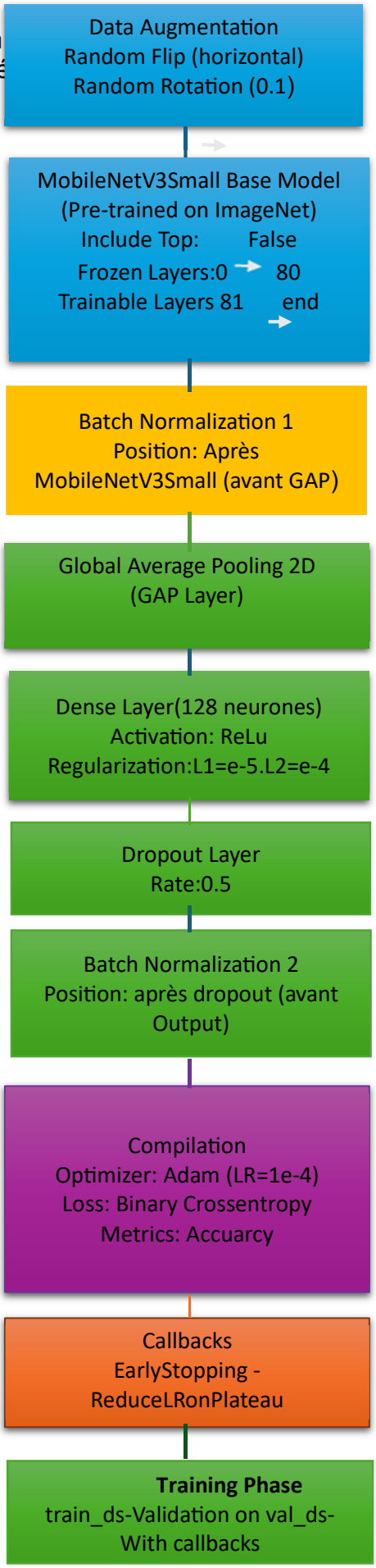


Figure 4.9 : MobilenetV3 affiné

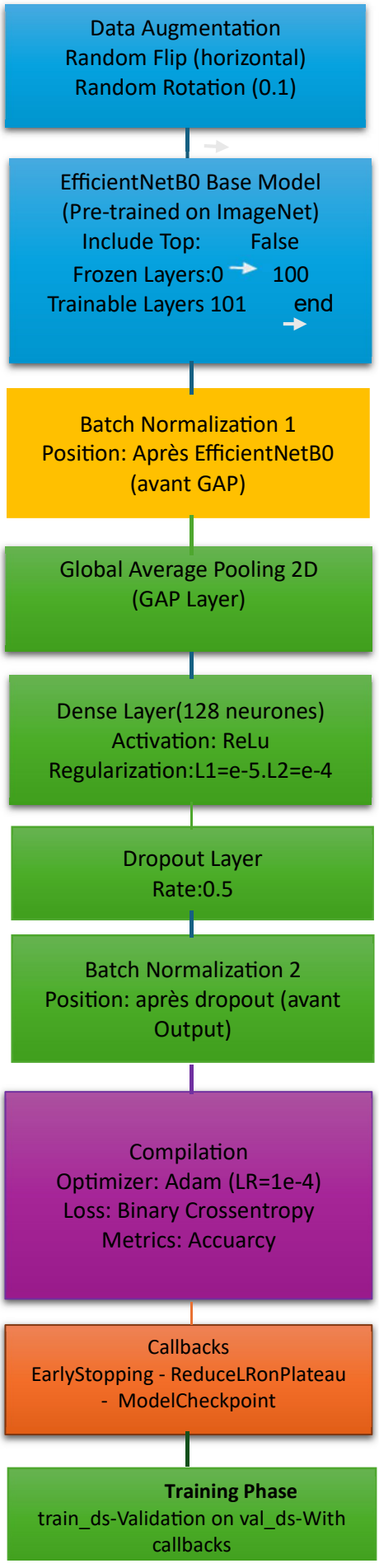


Figure 4.10 : EfficientnetB0 affiné

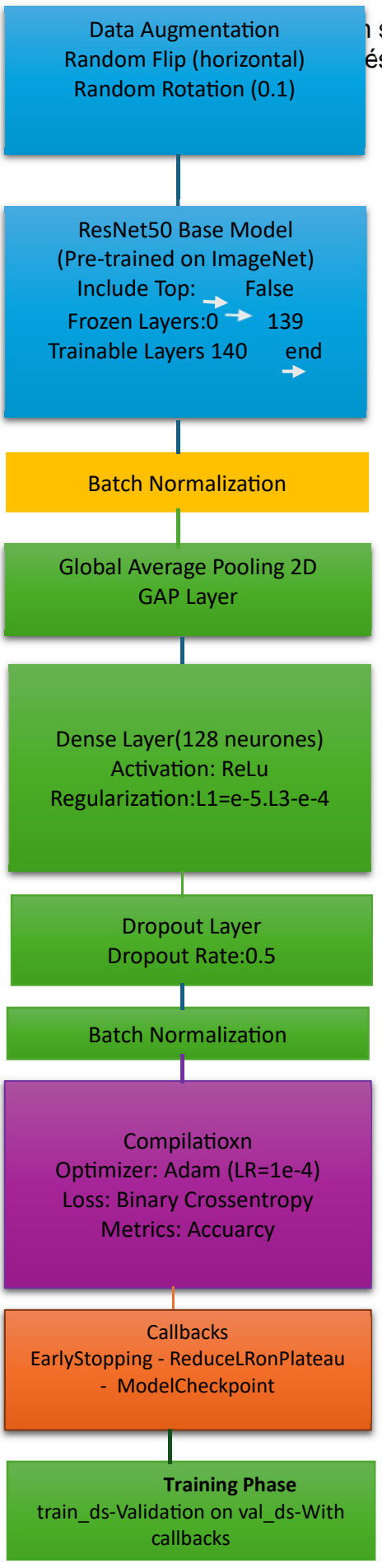


Figure 4.11 : Resnet50 affiné

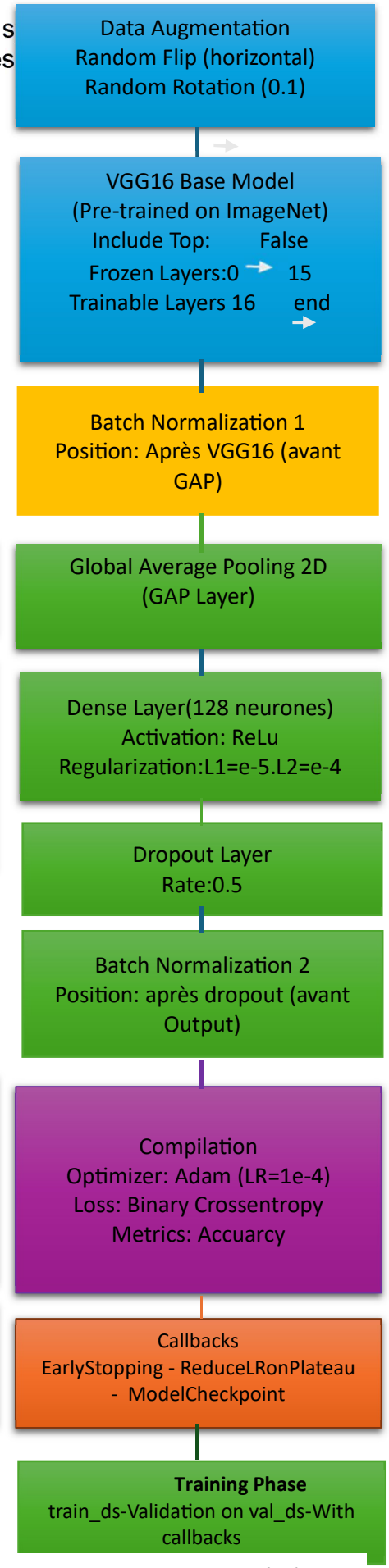


Figure 4.12 : VGG16 affiné

4.7 Stratégie d'entraînement et optimisation

Pour garantir une performance optimale et une bonne généralisation du modèle de classification Good/Bad Case, plusieurs techniques d'entraînement et d'optimisation ont été mises en œuvre tout au long du projet.

4.7.1 Répartition de la base

La base de données a été répartie en trois ensembles ; 70% pour le jeu d'entraînement, 15% pour la validation et 15% pour le test.

4.7.2 Data augmentation

Pour augmenter la diversité des données d'entraînement et limiter le surapprentissage, une augmentation des données a été appliquée via :

- **Des Rotations aléatoires** (petits angles ±10°).
- **Un Renversement horizontal.**
- **De Légers décalages et zooms** (optionnel selon les tests).

Ces transformations artificielles permettent aux modèles, d'être plus robustes face à la variabilité réelle des KPI.

4.7.3 Callbacks d'entraînement

Pour un entraînement plus efficace et stable, plusieurs 'callbacks Keras' ont été utilisés :

- **EarlyStopping** : il permet l'arrêt automatique de l'entraînement dès que la perte de validation ne s'améliore plus pendant un certain nombre d'époques (5). Cela évite le surapprentissage.
- **ReduceLROnPlateau** : cette fonction permet la réduction automatique du taux d'apprentissage (learning rate), lorsque la validation stagne, facilitant la convergence.
- **ModelCheckpoint** : cette fonction sauvegarde automatique des poids du modèle, lorsque la métrique de validation s'améliore, garantissant la meilleure version finale.

4.7.4 Optimisation de l'apprentissage

L'Optimiseur Adam a été choisi pour sa capacité à s'adapter automatiquement, aux gradients et sa rapidité de convergence. Dans ce cas, un taux d'apprentissage initial de 1e-4 a été fixé, puis ajusté dynamiquement grâce à la fonction ReduceLROnPlateau.

La taille des batches a été adaptée en fonction des ressources GPU disponibles, notamment sur Kaggle et Google Colab.

4.7.5 Évaluation pendant l'entraînement

Le jeu de données de validation a été utilisé pour superviser la performance du modèle et détecter le surapprentissage.

Les principales métriques évaluées, sont la précision (accuracy) et la perte (loss) sur les jeux d'entraînement et validation.

4.7.6 Test final

Le jeu de test, non utilisé pendant l'entraînement ni la validation, a permis d'évaluer la performance finale et la capacité de généralisation réelle du modèle.

Sur ce jeu de test, des métriques supplémentaires ont été calculées, telles que la matrice de confusion, et le rapport de classification (precision, recall, F1-score).

4.7.7 Impact sur la détection du MLB

Dans le cadre du MLB (Mobility Load Balancing), la distinction entre la classe **Good case** et la classe **Bad case,** est directement liée à la prise de décision d'un rééquilibrage de charge.

- **Good case :** le secteur nécessite le déclenchement du MLB (action requise) Autrement dit, un good case correspond à un déséquilibre détecté, le MLB doit être déclenché.
- **Bad case** : le secteur ne nécessite pas d'action (pas de déséquilibre, pas de MLB).

4.8 Résultats et interprétation

Cette section présente les principaux résultats obtenus, pour la classification cellulaire LTE, suivant les courbes d'entraînement.

L'analyse des courbes d'apprentissage (accuracy et loss), permet d'observer le comportement dynamique des modèles durant l'entraînement. Elle est essentielle pour détecter la stabilité du modèle, sa capacité de généralisation et d'éventuels phénomènes de surapprentissage. La matrice de confusion est aussi évaluée, mettant en avant la détection des vrais positifs et des vrais négatifs.

4.8.1 Résultats de MobileNet V1

Ce modèle affiche une progression lente, mais régulière de l'accuracy (figure 4.13), atteignant 88 %. Les courbes de loss (figure 4.14) pour l'entraînement et la validation sont très proches, indiquant une bonne convergence sans le surapprentissage.

Cependant, l'amélioration sature rapidement, dès l'époque 20, suggérant une capacité d'apprentissage limitée.

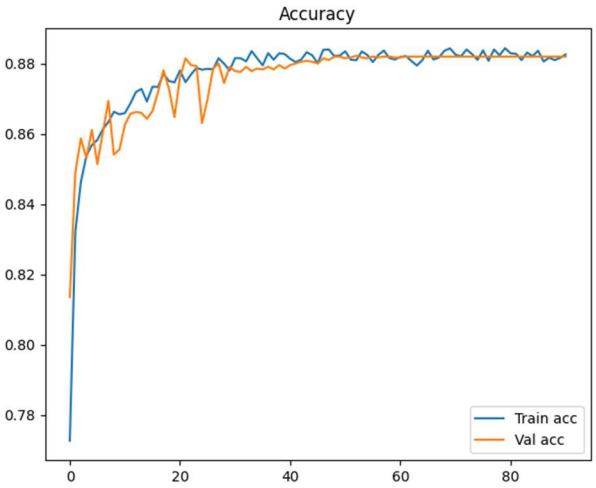


Figure 4.13 : Courbe de l'exactitude de mobilenetV1

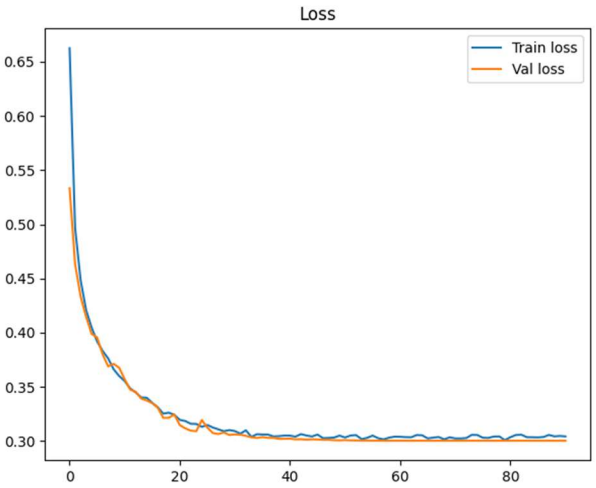


Figure 4.14 : Courbe de la perte de mobilenetV1

Analyse de la matrice de confusion

La matrice de confusion (figure 4.15) présente pour la classe 'good case', 2839 échantillons correctement classés comme Good case (vrais positifs) et 235 échantillons ont été mal classés (faux négatifs) comme Bad case. Pour la classe 'bad case', 767 échantillons ont été correctement détectés comme Bad case (vrais négatifs) et 281 échantillons ont été classés à tort (faux positifs), comme Good case.

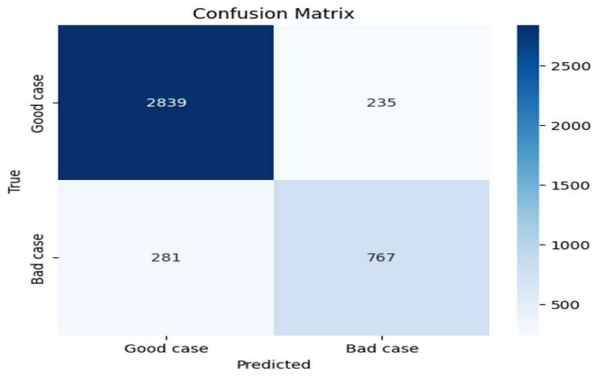


Figure 4.15 : Matrice de confusion de MobileNetV1

Le modèle démontre une bonne capacité à détecter les Good cases. Cependant, 235 'Good cases' ne sont pas correctement détectés (faux négatifs), soit 8 % d'erreurs sur cette classe.

Dans le cas de la classe 'bad case', les performances sont plus faibles, avec un taux de faux positifs élevé (27 % des Bad cases non détectés).

L'analyse détaillée des performances montre que les résultats de la matrice de confusion, sont cohérents avec le rapport de classification obtenu (tableau 4.4).

Classe	Précision	Rappel	F1-score
'Good case'	91%	92%	92%
'Bad case'	77%	73%	75%
Accuracy	**88%**		
Macro Average	**83%**		
weighted average	**88%**		

Tableau 4.5 : Rapport de classification

Le paramètre macro moyenne (Macro Average) évalué à 83 %, montre un déséquilibre de performance entre les deux classes. Le modèle étant plus performant sur la classe majoritaire (Good case).

La moyenne pondérée (**weighted average) évaluée à 87%,** reste proche de l'accuracy, ce qui est attendu vu que la classe Good case est dominante dans le jeu de données.

- **Analyse des erreurs les plus critiques**

Les faux négatifs sur la classe 'Good case' (235 cas) indiquent que le modèle n'a pas détecté certains secteurs en déséquilibre, ce qui pourrait conduire à ne pas déclencher le MLB là où il est pourtant nécessaire. C'est l'erreur la plus pénalisante pour la qualité de service.

Les faux positifs sur la classe 'Bad case' (281 cas) montrent que **l**e modèle a classé à tort des Bad cases, comme Good cases, ce qui risquerait de déclencher des actions MLB inutiles, générant ainsi des déplacements de charge non justifiés, voire des instabilités réseau.

- **Discussion sur les résultats de MobileNet**

Le modèle MobileNet montre de bonnes performances générales (87 % pour l'accuracy). Mais des limites restent visibles, surtout pour la détection des Bad cases.
Dans un contexte où l'objectif prioritaire est de ne pas manquer les Good cases, l'optimisation de la sensibilité (recall) sur cette classe est essentielle.

4.8.2 Résultats de MobileNetV2

MobileNetV2 montre une amélioration continue de la performance (figure 4.16), avec une accuracy de validation de 96 %, légèrement supérieure à celle d'entraînement (94 %). Cela traduit une bonne capacité de généralisation, renforcée par une perte de validation en baisse constante (figure 4.17). Ce modèle s'avère stable et performant, sans signe de surajustement.

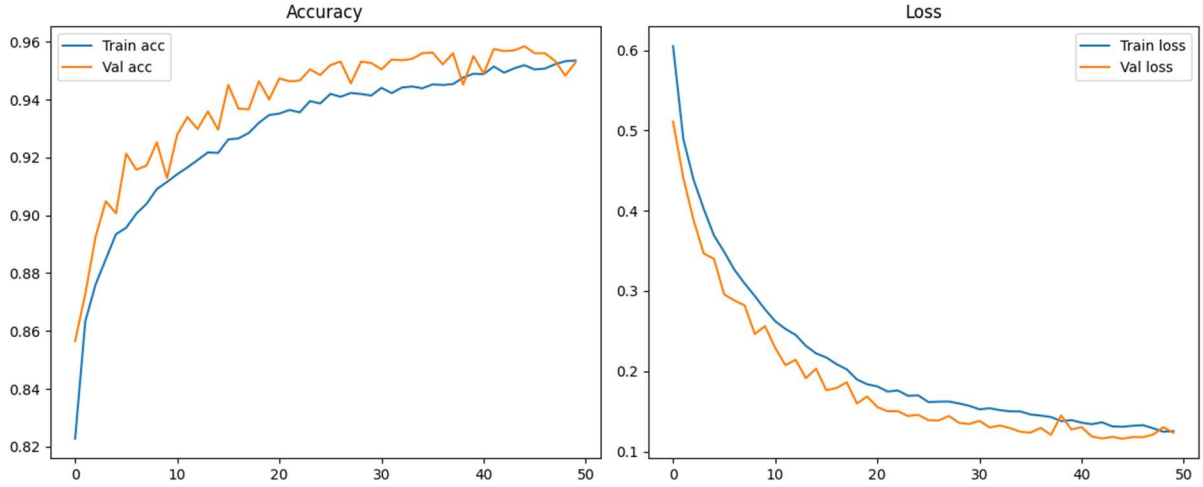


Figure 4.16 : Courbe de l'exactitude de mobilenetV2

Figure 4.17 : Courbe de la perte de mobilenetV2

Analyse de la matrice de confusion

L'évaluation du modèle basé sur l'architecture MobileNetV2 sur le jeu de test a permis d'obtenir la matrice de confusion représentée par la figure 4.19.

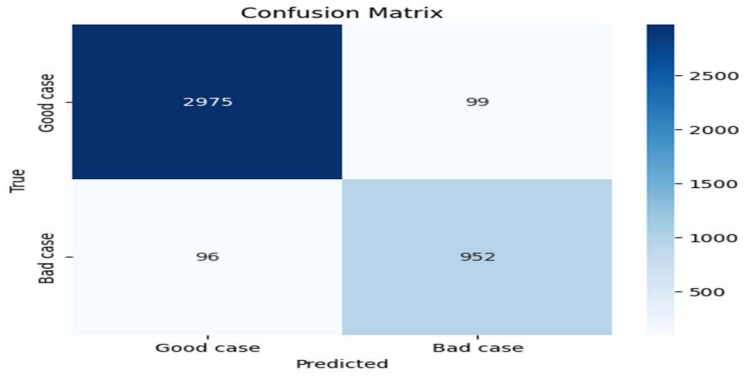


Figure 4.19 : Matrice de confusion MobileNetV2

La matrice de confusion obtenue montre que pour la classe "Good case", 2975 exemples ont été correctement classés (vrais positifs) et 99 exemples ont été mal classés (faux négatifs).

Pour la classe "Bad case", 952 exemples ont été correctement détectés comme Bad case (vrais positifs) et 96 exemples ont été classés à tort (faux positifs).

- **Analyse des performances**

L'analyse des performances (tableau 4.6) montre un taux de classification correct sur l'ensemble du jeu de test.

Classe	Précision	Rappel	F1-score		Nombre de classes
Good case	0.97	0.97	0.97		3074
Bad case	0.91	0.91	0.91		1048
Accuracy				96%	4122

Tableau 4.6 : Rapport de classification

Ces scores indiquent que le modèle est très performant pour détecter les Good cases, avec seulement 3 % de faux négatifs, ce qui est crucial dans le contexte de déclenchement du MLB. Le modèle est également fiable pour détecter les Bad cases, avec seulement 9 % d'erreurs, ce qui limite les faux déclenchements d'actions MLB.

- **Analyse des erreurs critiques**

Les erreurs les plus critiques restent les faux négatifs sur la classe Good case (ici 99 cas), car elles correspondent à des secteurs en surcharge, que le modèle n'a pas identifiés, entraînant un risque de dégradation de la qualité de service. Inversement, les faux positifs sur Bad case (96 cas) entraînent des déclenchements inutiles du MLB, ce qui pourrait perturber inutilement l'équilibre des ressources réseau.

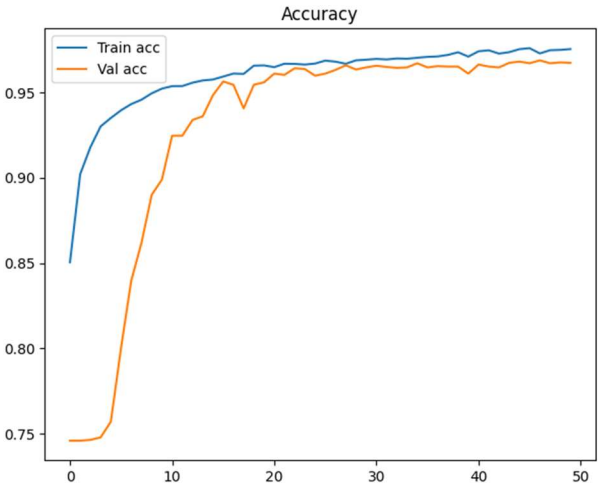
- **Discussion spécifique à MobileNetV2**

Le modèle MobileNetV2 montre un excellent compromis entre la légèreté, la vitesse d'inférence et la précision, avec une eexactitude de 96 % et des performances équilibrées sur les deux classes. Sa capacité à bien détecter les Good cases, tout en gardant un faible taux d'erreurs sur les Bad cases, en fait une solution viable pour un déploiement opérationnel dans le contexte du MLB sur réseau LTE.

4.8.3 Résultats du modèle MobileNetV3Small

MobileNetV3Small atteint rapidement un taux de validation supérieur à 97 %, avec seulement quelques oscillations initiales (figure 4.20). La courbe de perte (figure 4.21) décroît de façon

nette, surtout entre les 10 premières époques, avant de se stabiliser. L'apprentissage est rapide, précis, et montre une excellente adaptation, ce qui est remarquable pour une architecture légère.



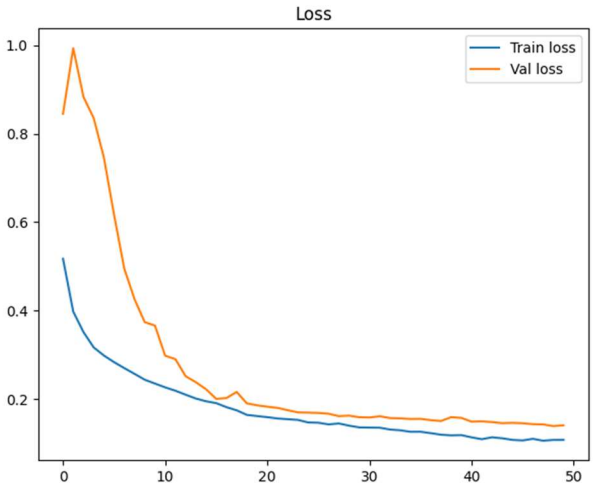


Figure 4.20 : Courbe de l'exactitude de mobilenetV3 small

Figure 4.21 : Courbe de la perte de mobilenetV3 small

Analyse de la matrice de confusion

La matrice de confusion (figure 4.22) montre que le modèle MobileNetV3-Small a réalisé un très bon taux de classification.

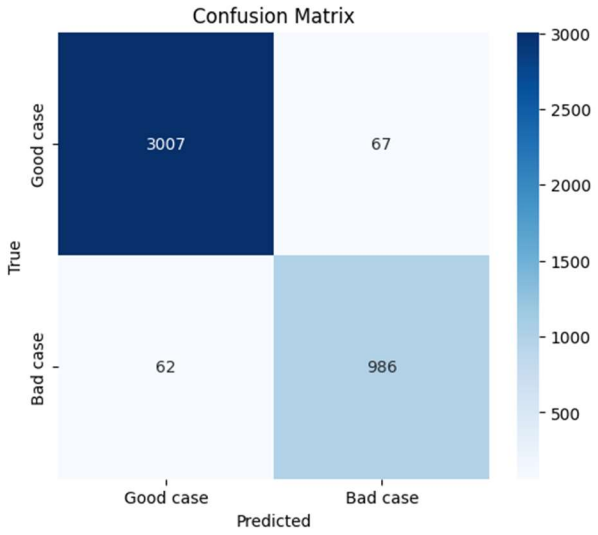


Figure 4.22 : Matrice de confusion de mobilenetV3 small

Dans le cas de la classe "Good case", sur les 3074 cas réels, le modèle en a correctement identifié 3006 (vrais positifs), soit un taux de détection de 98 %, avec seulement 67 erreurs de classification (faux négatifs).

Dans le cas de la classe "Bad case", sur 1048 exemples réels, le modèle a correctement détecté 986 Bad cases (vrais négatifs), avec 62 erreurs (faux positifs).

Le modèle montre une forte capacité de généralisation, avec :

- **Peu de faux négatifs** : cela signifie que le modèle ne surclasse pas à tort des Good case en Bad case, ce qui évite des actions de rééquilibrage injustifiées dans le réseau.
- **Peu de faux positifs :** le modèle parvient à capturer la majorité des Bad cases, ce qui est crucial dans le contexte d'optimisation de la charge réseau. En effet, un Bad case non détecté pourrait entraîner un maintien de la surcharge réseau, ce qui affecterait la qualité d'expérience des utilisateurs.

Discussion spécifique à MobileNetV3-Small

Les résultats obtenus avec MobileNetV3-Small, démontrent que même une architecture légère et optimisée pour le déploiement sur des infrastructures limitées, est capable de fournir d'excellentes performances de classification dans un contexte réseau LTE.

4.8.4 Résultats du modèle EfficientNetB0

Efficientnet présente des courbes proches entre l'entraînement et la validation (figure 4.23), avec une accuracy qui a atteint 96 % pour la validation. L'évolution de la perte (figure 4.24) est fluide et sans fluctuations anormales. Ce modèle démontre une convergence régulière, sans surapprentissage, même sur un faible nombre d'époques (50).

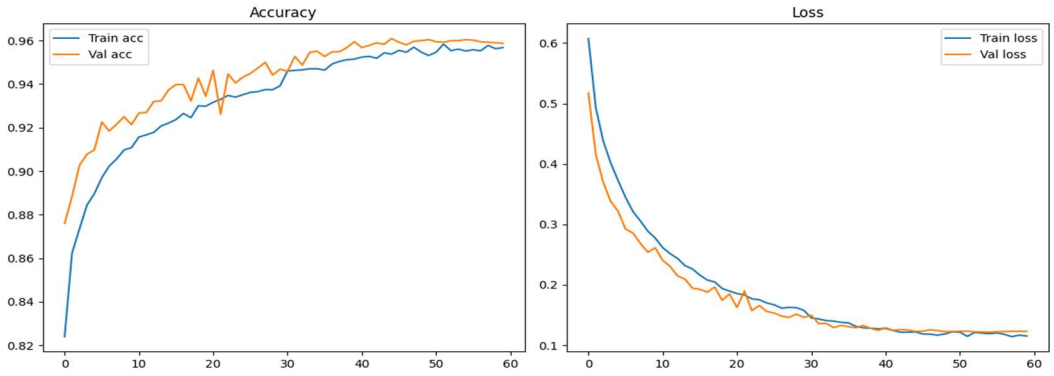


Figure 4.23 : Courbe de l'exactitude d'EfficientNet

Figure 4.24 : Courbe de la perte d'EfficientNet

Analyse de la matrice de confusion

La matrice de confusion (figure 4.25) montre que le modèle efficientnet a réalisé un taux de classification satisfaisant.

Dans le cas de la classe "Good case", sur les 3074 cas réels, le modèle en a correctement identifié 2973, avec 101 erreurs de classification (faux négatifs).

Dans le cas de la classe "Bad case", sur 1048 exemples réels, le modèle a correctement détecté 982 Bad cases, avec 66 erreurs (faux positifs).

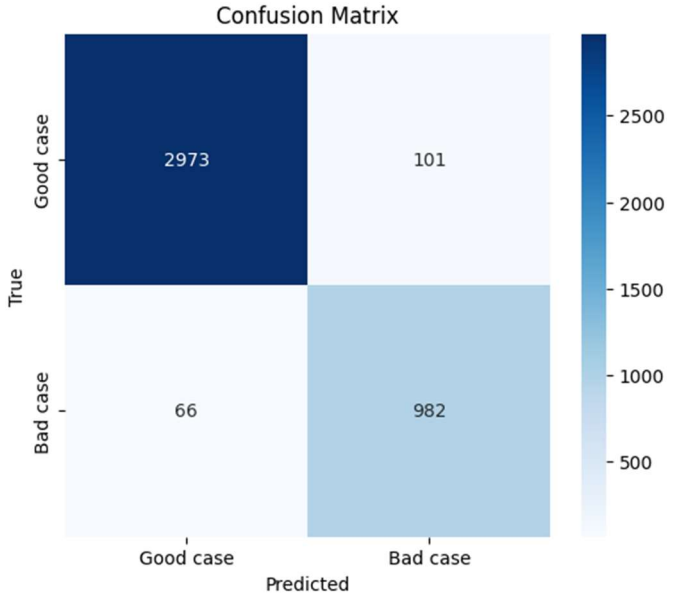


Figure 4.25 : Matrice de confusion

- **Analyse des performances**

L'analyse des performances (tableau 4.7) montre un bon taux de classification sur l'ensemble du jeu de test.

Classe	Précision	Rappel	F1-score	Nombre de classes
'Good case'	98%	977%	97%	**3074**
'Bad case'	91%	94%	92%	**1048**
Accuracy	**96%**			**4122**
Macro Average	**95%**			
weighted average	**96%**			

Tableau 4.7 : Rapport de classification

Le modèle EfficientNet a atteint une précision globale de 96 %, montrant sa capacité à catégoriser les deux classes.

Dans le cas de la classe des 'good cases', il a atteint une précision de 98%, un rappel de 97% et un F1-score de 97% montrant donc, une très bonne capacité à détecter cette classe, avec seulement 3 % de faux négatifs (101 cas).

Dans le cas de la classe des 'bad cases', le modèle a obtenu une précision de 91%, un rappel de 94% et un F1-score de 92%, ce qui montre que les performances sur cette classe, restent très bonnes, avec seulement 6 % d'erreurs.

Avec une macro moyenne de 95%, le modèle indique un bon équilibre entre les deux classes.

La moyenne pondérée de 96%, reflète la forte présence de la classe des 'Good cases' dans le dataset.

- **Analyse des erreurs les plus critiques pour EfficientNet**

Les 101 faux négatifs sur la classe des 'Good cases', qui représentent des secteurs en surcharge non détectés par le modèle, peuvent conduire à un non-déclenchement du MLB, là où il serait nécessaire.

Les 66 faux positifs détectés sur la classe des 'Bad cases', peuvent entraîner des déclenchements MLB injustifiés, mais leur nombre reste relativement faible**.**

- **Interprétation spécifique à EfficientNet**

Le modèle EfficientNet présente un très bon compromis entre la précision et l'efficacité, suivant :

- Une exactitude de 96 %.
- Des F1-scores élevés sur les deux classes.
- Des erreurs critiques (faux négatifs sur Good case) limitées.

Sa légèreté par rapport à des modèles plus lourds comme VGG16 ou ResNet, tout en maintenant une excellente performance, fait d'EfficientNet un candidat très intéressant pour une mise en œuvre en production, surtout si les ressources matérielles sont limitées**.**

4.8.5 Résultats de Resnet50

ResNet50 présente un comportement exemplaire : les courbes d'entraînement et de validation relatives à l'exactitude (figure 4.26), restent proches avec une convergence rapide dès l'époque **10**.

La perte (figure 4.27) chute fortement avant de se stabiliser autour de **0.1**, avec une accuracy relative à la validation proche de **97 %**. Le modèle combine performance et stabilité, sans signe d'overfitting.

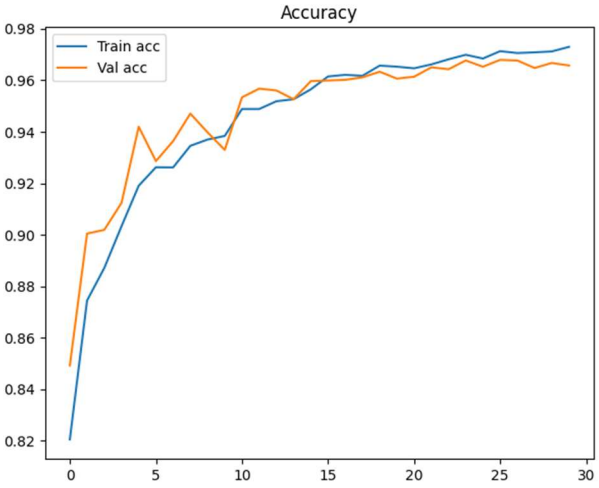


Figure 4.26 : Courbe de l'exactitude de Resnet50

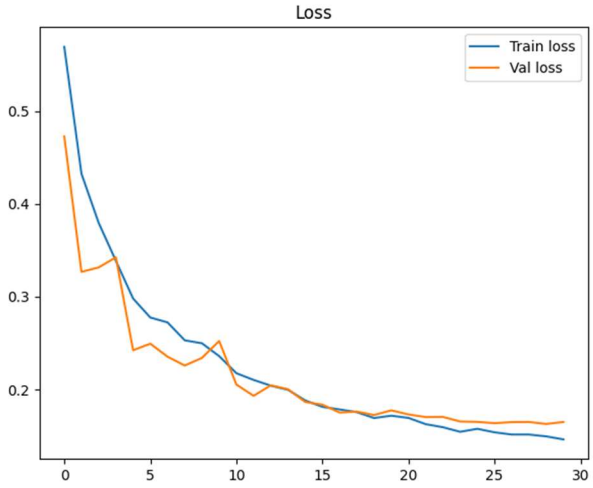


Figure 4.27 : Courbe de la perte de Resnet50

Analyse de la matrice de confusion

Le modèle entraîné avec l'architecture ResNet a été évalué sur le jeu de test, montrant des résultats satisfaisants via la matrice de confusion (figure 4.28).

Dans ce cadre, pour la classe "Good case", 3021 échantillons ont été correctement classés comme Good case (vrais positifs) et 53 échantillons ont été mal classés (faux négatifs).

Pour la classe "Bad case", 975 échantillons ont été correctement détectés comme Bad case (vrais négatifs) et 73 échantillons ont été à tort classés (faux positifs).

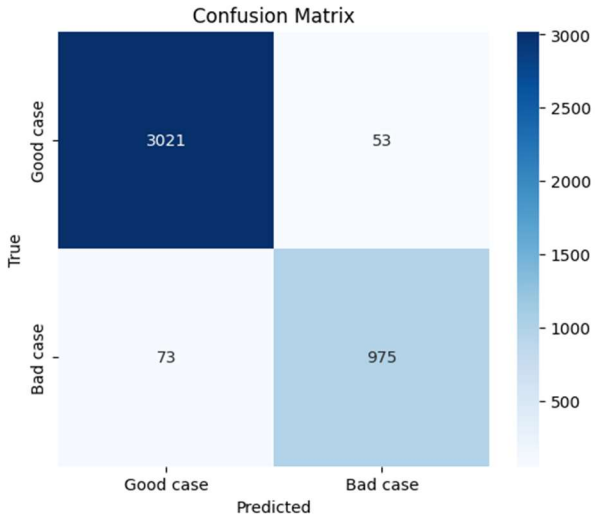


Figure 4.28 : Matrice de confusion

Analyse des performances

Les résultats de la matrice de confusion sont cohérents avec les métriques du rapport de classification indiqué par le tableau 4.8.

Classe	Précision	Rappel	F1-score	Nombre de classes
'Good case'	98%	98%	98%	**3074**
'Bad case'	93%	93%	93%	**1048**
Accuracy	**97%**			**4122**
Macro Average	**95%**			
weighted average	**96%**			

Tableau 4.8 : Rapport de classification

Le modèle a atteint une précision globale de 97 %, ce qui démontre une excellente capacité de généralisation.

Les performances obtenues pour la classe 'good case', indiquent une précision, un rappel et un F1-score de 98%, signifiant que le modèle détecte presque tous les secteurs, nécessitant un déclenchement du MLB, avec seulement 53 faux négatifs (environ 2 % d'erreur).

La détection des Bad cases est également très satisfaisante, avec seulement 7 % d'erreurs de classification.

Avec seulement 53 faux négatifs sur la classe 'Good case', ResNet minimise considérablement le risque de laisser des secteurs en surcharge sans traitement, ce qui est crucial pour la qualité de service réseau.

De même, avec 73 faux positifs sur la classe 'Bad case', le modèle évite la majorité des déclenchements inutiles du MLB, ce qui préserve la stabilité globale du réseau.

Discussion spécifique à ResNet

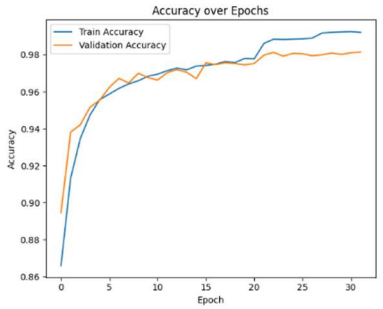
Parmi les modèles testés, ResNet se distingue par sa très haute précision et son excellent équilibre entre les deux classes, avec :

- Une accuracy de 97 %.
- Des F1-scores supérieurs à 93 % pour les deux classes.
- Une très faible proportion d'erreurs critiques, que ce soit pour les Good cases ou les Bad cases.

Ces performances font de ResNet un candidat particulièrement adapté aux environnements où, la précision de la détection des cas de déséquilibre est prioritaire, malgré une complexité et une taille du modèle supérieures par rapport aux architectures plus légères, comme MobileNet.

4.8.6 Résultats de VGG16

VGG16, bien qu'atteignant une exactitude de 98 %, affiche un léger décalage entre les courbes d'entraînement et de validation (figure 4.29). Cela suggère un début de surapprentissage dès l'époque 25, la courbe d'entraînement, continuant de progresser pendant que la validation stagne. Ce comportement est attendu pour un modèle aussi profond et lourd. Avec une faible perte, la courbe de perte (figure 4.30) se stabilise avec une erreur de 1%.



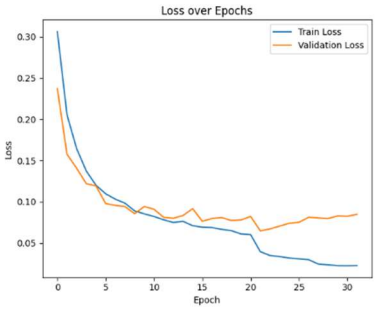


Figure 4.29 : Courbe de l'exactitude de VGG16 **Figure 4.30** : Courbe de la perte de VGG16

- **Analyse des performances**

Les performances globales du modèle VGG16 sont excellentes, comme le montre le rapport de classification du tableau 4.9.

Classe	Précision	Rappel	F1-score	Nombre de classes
'Good case'	99%	99%	99%	**3074**
'Bad case'	96%	96%	96%	**1048**
Accuracy	**98%**			**4122**
Macro Average	**97%**			
weighted average	**98%**			

Tableau 4.9 : Rapport de classification

L'exactitude de 98 % signifie que des échantillons du jeu de test, ont été correctement classés par le modèle. Effectivement, VGG16 a montré d'excellentes performances avec une précision, un rappel et un F1-score de 99% pour la classe 'good case'.

Le paramètre 'macro average', Indique un très bon équilibre entre les deux classes, sans qu'une classe soit trop favorisée.

La moyenne pondérée, présentant un taux de 98%, correspond à de très bonnes performances.

Le modèle détecte presque parfaitement les good cases, avec seulement 1 % d'erreur. Il est aussi très performant sur la détection des 'bad cases', avec seulement 4 % d'erreurs, ce qui est très faible vu la taille du jeu de test.

Avec ces très faibles taux d'erreur, VGG16 présente des garanties solides pour une prise de décision MLB fiable.

- **Discussion spécifique à VGG16**

Le modèle VGG16 offre des performances exceptionnelles, aussi bien sur la détection des Bad cases que des Good cases, suivant, une exactitude de 98 %, un F1-score très élevé et une excellente sensibilité sur les deux classes. Ces résultats positionnent VGG16 comme l'un des modèles les plus précis parmi ceux testés, même s'il reste plus lourd et plus coûteux, en ressources de calcul comparé aux architectures MobileNet ou EfficientNet.

Son utilisation est donc idéale, lorsque la priorité est donnée à la précision, au détriment du temps de l'inférence ou de la légèreté du modèle.

4.9 Analyse comparative des modèles développés

Afin de déterminer le modèle le plus performant pour la classification Good Case / Bad Case dans le cadre de l'optimisation MLB, une étude comparative des architectures des réseaux de neurones développés est effectuée (tableau 4.10).

Modèle	Accuracy	Loss	F1-score	Epochs	Meilleure Val_accuracy
MobileNet	88%	0.1	86%	100	88%
MobileNetV2	95%	0.15	94%	70	95%
MobileNetV3Small	97%	0.13	96%	50	97%
EfficientNetB0	96%	0.1	96%	50	96%
VGG16	98%	0.015	98%	50	98%
ResNet50	98%	0.15	96%	50	95%

Tableau 4.10 : Analyse comparative

À l'issue de cette analyse comparative des résultats (matrices de confusion, métriques de classification et courbes d'apprentissage), plusieurs constats majeurs peuvent être établis :

- ResNet50 s'impose comme le modèle le plus performant, avec une accuracy globale de 97 %, un excellent rappel sur les *Good cases* (98 %) et une grande stabilité d'apprentissage. Il est particulièrement adapté aux cas critiques du MLB, où chaque décision a un impact direct sur la qualité de service.
- MobileNetV3Small et EfficientNetB0 offrent des compromis idéaux pour des environnements contraints (temps réel, ressources limitées) tout en maintenant des performances élevées (F1-score de 96 %, rappel > 97 %).
- MobileNetV2 est une solution équilibrée avec une excellente généralisation, une accuracy de 96 % et une grande légèreté, le rendant pertinent pour un déploiement embarqué.
- VGG16, bien qu'ayant atteint 98 % de précision, souffre de son poids computationnel et d'un début de surapprentissage, limitant son usage aux environnements sans contraintes matérielles.

- MobileNet Standard reste en retrait par rapport aux autres architectures, aussi bien en accuracy (87 %) qu'en capacité de détection des cas critiques (rappel Good case = 92 %).

4.10 Conclusion

En conclusion, L'étude réalisée pour la prédiction cellulaire LTE, a nécessité plusieurs expérimentations, pour atteindre les résultats obtenus, suivant des techniques de régularisation pour éviter le surapprentissage, un équilibre de classes, pour aboutir à une précision satisfaisante et une fonction d'optimisation ADAM, pour un entraînement efficace.

Les résultats obtenus sont largement satisfaisants, avec certains compromis à adopter. Dans ce cadre, le choix du modèle dépendra fortement des contraintes opérationnelles :

- Pour un déploiement dans un environnement critique avec une tolérance zéro sur les erreurs : ResNet50 est recommandé.
- Pour un déploiement embarqué ou edge computing, MobileNetV3Small ou EfficientNetB0 sont les candidats idéaux.
- Enfin, MobileNetV2 constitue un excellent compromis entre la performance et l'efficience pour une intégration simple et fiable du système de détection MLB.

Conclusion Générale

Dans un contexte de croissance rapide du trafic mobile en Algérie, la gestion dynamique des ressources radio dans les réseaux LTE, est devenue un enjeu majeur pour les opérateurs comme Djezzy. Face au phénomène du déséquilibre intercellulaire, où certaines cellules sont surchargées tandis que d'autres restent sous-utilisées, l'optimisation du Mobility Load Balancing (MLB) devient essentielle pour garantir la qualité du service et une meilleure répartition du trafic. Le travail réalisé dans ce projet, s'est inscrit dans cette optique, pour la prédiction du réseau cellulaire LTE, suivant six modèles des réseaux de neurones convolutifs pré-entraînés adaptés.

Tout au long de ce mémoire, nous avons exploré les différentes facettes de cette problématique. La première partie a permis de dresser un état des lieux du réseau LTE et des défis spécifiques, liés au déséquilibre de charge. La deuxième partie a présenté les concepts fondamentaux de l'intelligence artificielle, en mettant l'accent sur l'apprentissage profond et les réseaux de neurones convolutifs (CNN), qui se sont révélés être des outils puissants pour la classification supervisée des réseaux LTE.

Sur le plan conceptuel, un processus de traitement de données adapté a été défini, allant de l'extraction des KPIs à la transformation des données tabulaires en images, en passant par le fine-tuning partiel de plusieurs modèles CNN pré-entraînés. Ces modèles ont été entraînés sur plusieurs hyperparamètres, comme les techniques de régularisation (dropout de 0.5, L1 et L2de l'ordre de 10^{-5}), ou la fonction d'optimisation ADAM avec un taux d'apprentissage initial de 10^{-3}, pour éviter le surapprentissage et permettre la stabilité du système.

L'analyse comparative des résultats obtenus avec les différentes architectures (MobileNet, MobileNetV2, MobileNetV3Small, EfficientNetB0, VGG16 et ResNet50) a permis de dégager plusieurs constats majeurs :

- **ResNet50** s'est imposé comme le modèle le plus performant, avec une **exactitude e 97 %**, des **F1-scores supérieurs à 93 %**, et une excellente capacité à détecter les **Good Cases**, c'est-à-dire les secteurs en déséquilibre, nécessitant un rééquilibrage.
- **MobileNetV3Small** et **EfficientNetB0** ont également montré de bonnes performances (**96 % pour l'exactitude**), tout en offrant une grande légèreté et une rapidité d'inférence adaptées à des environnements contraints.

- **MobileNetV2** s’est distingué par son équilibre entre la **simplicité**, la **vitesse** et la **précision**, le rendant pertinent pour des déploiements embarqués.
- Enfin, **VGG16**, bien qu'ayant atteint **98 % pour l’exactitude**, a montré des signes de surapprentissage et nécessite des ressources matérielles importantes, limitant son utilisation pour des applications en temps réel.

Au-delà de ces performances, ce travail ouvre la voie vers une future **intégration du système de prédiction dans une chaîne SON (Self-Organizing Network)** chez Djezzy. Concrètement, cela consisterait à :

1. **Collecter en temps réel les KPIs** du réseau via les plateformes Huawei (U2020, MAOS),
2. **Prétraiter et convertir ces données** en images,
3. **Effectuer une prédiction immédiate de l’état de chaque cellule**,
4. **Générer des recommandations MLB automatiques**,
5. **Déclencher dynamiquement des actions de rééquilibrage** via le système SON existant.

Cette approche permettrait de réduire les délais de réaction, d’améliorer la qualité d’expérience des utilisateurs et de maximiser l’utilisation des ressources radio disponibles, en répondant de manière proactive aux variations du trafic.

Plusieurs axes de développement peuvent être également envisagés :

- **Élargir la base de données** avec des périodes plus longues et des zones géographiques variées,
- **Explorer des architectures plus récentes** (EfficientNetV2, RegNet, etc.),
- **Optimiser l’inférence en temps réel**, pour rendre le système directement déployable sur les serveurs SON,
- **Procéder à une validation terrain**, en intégrant le modèle dans l’environnement de production chez Djezzy.

Ce projet a démontré la faisabilité et l’efficacité de l’apprentissage profond pour la détection automatisée des déséquilibres de charge dans les réseaux LTE. Il représente une avancée vers la mise en place d’une gestion plus intelligente, autonome et réactive du trafic mobile en Algérie, en s’inscrivant pleinement dans la dynamique de transformation digitale des opérateurs.

Bibliographie

[1] ARPCE. Plan National des Fréquences LTE en Algérie, Autorité de Régulation de la Poste et des Communications Électroniques, rapport annuel, https://www.arpce.dz/fr/file/o3x8z4, 2023, consulté le 29 avril 2025.

[2] S. Sesia, I. Toufik, and M. Baker, LTE - The UMTS Long Term Evolution : From Theory to Practice, Wiley, 2011.

[3] Philippe Godlewski, Philippe Martins, Marceau Coupechoux. Concepts Cellulaires et Paramètres Radio, ENST, Département Informatique et Réseaux, France, https://marceaucoupechoux.wp.imt.fr/files/2018/02/conceptscell.pdf, consulté le 29 avril 2025.

[4] E. Boukari, L. Zemerli. Deep learning structure RNN pour la prédiction de la congestion dans les réseaux de télécommunications. Mémoire de fin d'étude en électronique, USTHB, Algérie 2024.

[5] M. Reguiegue, A. Chaabane. Etude et dimensionnement d'un réseau mobile 4G LTE à la ville d'El-Oued, mémoire de master en télécommunications, université d'El oued, Algérie, 2021.

[6] D. Hocine, A. Missoum. Analyse et optimisation du réseau d'accès radio LTE, mémoire de master en télécommunications, université de Blida 1, Algérie, 2020.

[7] C. T. Kuo, M. Médard, A. E. Özdaglar. Completion Time Minimization and Robust Power Control in Wireless Packet Networks, Actes de la Conférence Internationale IEEE sur les Communications (ICC), Dresde, Allemagne, pp. 1–6, juin 2009, https://ieeexplore.ieee.org/document/5198837, consulté le 25 avril 2025.

[8] G. Bellik, Y. Farez. Etude et optimisation des ressources d'un réseau LTE, mémoire de master en électronique, université de Tizi Ouzou, Algérie, 2018.

[9] H. Ayad, R. A. Mekidiche. Optimisation de la couverture radio 4 G LTE des opérateurs de télécommunications, mémoire de master en télécommunications, université de Tlemcen, Algérie, 2018.

[10] F. Rabahi. Optimisation du handover pour les réseaux LTE par l'intelligence artificielle, mémoire de master en télécommunications, université d'Aïn Temouchent, Algérie, 2016.

[11] A. Lounnas, L. SILI. Étude du canal de transmission des liaisons FHN dans le réseau de télécommunication 3G et 4G, mémoire de master en télécommunications et réseaux, université de Tizi Ouzou, Algérie, 2016.

[12] Cable Free, « LTE Interfaces », CableFree: Wireless Excellence, https://www.cablefree.net/wirelesstechnology/4glte/lte-interfaces, consulté le 18 mai 2025.

[13] Z. Polgár. LTE system radio interface and system architecture, cours en télécommunications, université technique de Cluj-Napoca, Roumanie, https://users.utcluj.ro/~dtl/TCSTI/Cursuri/Curs_STI_01_e.pdf, 2025, consulté le 20 mai 2025.

[14] David Tse and Pramod Viswanath, Fundamentals of Wireless Communication, Cambridge University Press, 2005.

[15] ETSI TS. Digital cellular telecommunications system, technical specification, https://www.etsi.org/deliver/etsi_ts/133100_133199/133108/17.02.00_60/ts_133108v170200p.pdf, consulté le 20 mai 2025.

[16] A. Pérez, LTE et LTE-Advanced : Les interfaces radioélectriques du réseau mobile 4G, Dunod, Paris, France, 2014.

[17] 3GPP TS 36.101. E-UTRA; User Equipment (UE) radio transmission and reception, Section 5 (Operating Bands and Channel Arrangement), Tables 5.5-1 & 5.6.1-1, https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2411, consulté le 20 janvier 2025.

[18] Journal Officiel de la République algérienne, Plan national des fréquences, sections relatives aux bandes GSM/3G/4G, JORADP, https://www.joradp.dz/FTP/jo-francais/2021/F2021072.pdf, N° 72, pages 5-23, septembre 2021, consulté le 20 avril 2025.

[19] Communiqué ARPCE. Adjudication par appel à la concurrence n° 01/2025 pour l'octroi de trois licences d'établissement et d'exploitation d'un réseau de communications électroniques mobiles 5G, https://www.arpce.dz/fr/pub/n8j9p0?utm, mai 2025, consulté le 20 juin 2025.

[20] nPerf. Carte de couverture 5G/4G/3G Djezzy, Algérie, https://www.nperf.com/fr/map/DZ/-/1226.Djezzy/signal, consulté le 11 juin 2025.

[21] Djezzy, Djezzy teste avec succès la 5G à Alger, Oran et Annaba en partenariat avec Huawei et Nokia, https://www.djezzy.dz/djezzy/actualite-et-nouveautes/communiques-de-presse/djezzy-teste-avec-succes-la-5g-a-alger-oran-et-annaba/, consulté le 12 juin 2025.

[22] H. Holma, A. Toskala. LTE for UMTS Evolution to LTE-Advanced, Wiley, 2011.

[23] M. Mathuranathan, Orthogonal Frequency Division Multiplexing (OFDM) Basics, https://www.gaussianwaves.com/2012/10/orthogonal-frequency-division-multiplexing-ofdm-basics/, consulté le 20 avril 2025.

[24] Ixia. SC-FDMA Single Carrier FDMA in LTE, https://support.ixiacom.com/sites/default/files/resources/whitepaper/sc-fdma-indd.pdf, Nov. 2009, consulté le 22 avril 2025.

[25] D. Tse, P. Viswanath. Fundamentals of Wireless Communication, Cambridge University Press, 2005.

[26] S. S. Mishra, J. S. Roy. Comparison of Performances between SC-FDMA and OFDMA Systems under Different Subcarrier Mapping Schemes, ASEAN Engineering Journal, vol. 13, N°2, pages 19–23, 2023.

[27] P. Kryszkiewicz. Power Amplifier-Aware Transmit Power optimization for OFDM and SC-FDMA Systems,, arXiv. https://arxiv.org/abs/2501.11994, https://doi.org/10.48550/arXiv.2501.11994, pages 1-6, 2025.

[28] Venkatesh, G. K. & Rao, P. V. Performance analysis of a novel method for fast handovers in TDD and FDD for long term evolution. International Journal of Engineering and Technology, vol. 7 N°1.9, pages 115–118, 2018.

[29] D. Chandra, S. Yusnita, D. B. Sitepu, A. Mursydan, D. Meidelfi. LTE Network Area Coverage on FDD and TDD Technology. International Journal of Advanced Science Computing and Engineering, vol. 2, N°1, pages 21–33, 2020.

[30] A. Pérez. LTE et LTE-Advanced : Les interfaces radioélectriques du réseau mobile 4G, Dunod, Paris, France, 2014.

[31] Huawei. Intra-RAT Mobility Load Balancing, 21 06 2024, https://www.huawei.com/en/psirt/vul-response-process, juin 2024, consulté le le 01 février 2025.

[32] T. Zaknoune, S. Ould-Dris. Optimisation Radio du Réseau 4G, mémoire de master en télécommunications, Université de Tizi Ouzou, 2018.

[33] A. Ozovehe, O. U.Okereke, E. C. Anene, A. U. Usman. Busy Hour Traffic Congestion Analysis in Mobile Macrocells. Nigerian Journal of Technology, vol. 36, N° 4, 1265–1270, 2017.

[34] Opensignal. Mobile Network Experience Report Algeria, Mars 2023, https://www.opensignal.com/reports/2023/03/algeria/mobile-network-experience, consulté le 12 juin 2025.

[35] Huawei Support, Basic Concepts of Load Balancing – NE40E-F V800R022C00SPC600 Feature Description, https://support.huawei.com, consulté le 10 juin 2025.

[36] Huawei Support, Configuring a Traffic Load Balancing Mode – AR300/AR700 CLI Guide," https://support.huawei.com, consulté le 10-juin-2025.

[37] LTE-SON Blogspot, Mobility Load Balancing Optimization in LTE, https://lte-son.blogspot.com/2018/10/mobility-load-balancing-optimization.html, consulté le 12 juin 2025.

[38] Huawei Support, Configuring a Load Balancing Mode for Unknown Unicast Traffic, https://support.huawei.com, consulté le 12 juin 2025.

[39] Bamidele Moses Kuboye. Comparative Analysis of Scheduling Algorithms Performance in a Long Term Evolution Network, Journal of Computer Science Research, vol. 3, N°4, pages 20-25, 2021.

[40] PakTechPoint, “Mobility Load Balancing (MLB) in LTE,” https://paktechpoint.com/mobility-load-balancing-in-lte-mlb-feature, Consulté le 13 avril 2025.

[41] S. Russell, P. Norvig. Artificial Intelligence: A Modern Approach, 4th ed., Pearson, 2021.

[42] ISO/IEC, Information technology – Vocabulary – Part 28: Artificial intelligence – Basic concepts and expert systems, Genève, Suisse, 1995.

[43] R. Bellman. An Introduction to Artificial Intelligence: Can Computers Think? Boyd & Fraser, San Francisco, 1978.

[44] R. Kurzweil. The Age of Intelligent Machines, MIT Press, 1990.

[45] E. Charniak, D. McDermott. Introduction to Artificial Intelligence, Addison-Wesley, 1985.

[46] E. Rich, K. Knight, Artificial Intelligence, McGraw-Hill, 1991.

[47] A. Benmansour, S. Houari. L’aide de l’intelligence artificielle à la prise de décision, Mémoire de fin d’études en Électronique, Université de Tlemcen, Algérie, 2017.

[48] A. Mackworth, R. Goebel. Computational Intelligence: A Logical Approach, Oxford University Press, 1998.

[49] T. Hastie, R. Tibshirani, J. Friedman. The Elements of Statistical Learning, Springer series in statistics, springer, 2009.

[50] R. Sutton, A. Barto. Reinforcement Learning: An Introduction, MIT Press, 2018.

[51] A. Shalev-Shwartz, S. Ben-David. Understanding Machine Learning: From Theory to Algorithms, 2014, Cambridge university press, 2014.

[52] A. Géron. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, O'Reilly, 2019.

[53] C. TOUZET. Les réseaux de neurones artificiels introduction au connexionnisme, cours, exercices et travaux pratiques. EC2, Collection de l’EERIE, N. Giambiasi, juillet 1992.

[54] Astronoo. Neurone formel : Comprendre notre univers, https://astronoo.com/fr/articles/neurone-formel.html, consulté le 10 janvier 2025.

[55] D. Etiemble, F. Auzanneau. Introduction aux réseaux de neurones, techniques de l’ingénieur, 2023.

[56] M. Nielsen. Neural Networks and Deep Learning, http://neuralnetworksanddeeplearning.com, consulté le 25 avril 2025.

[57] F. Heep. Qu'est-ce que l'intelligence artificielle (IA) ? https://s-peers.com/fr/wiki/was-ist-artificial-intelligence/, consulté le 20 janvier 2025.

[58] N.Neeta, K.V Santosh, R. Balasbramanian. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions, SN computer science, Springer, vol. 2, N°6, pages 1-20, 2021.

[59] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.

[60] K. Dahmane. Analyse d'images par méthodes de deep learning appliquée au contexte routier en conditions météorologiques dégradées, thèse de doctorat en électronique et systèmes, université Clermont, Auvergne, France, 2020.

[61] T.K. Senthil kumar. Machine learning model vs deep learning-case study approach, jigsaw, https://www.jigsawacademy.com/machine-learning-model-vs-deep-learningcase-study-approach/, 2020, consulté le 20 janvier 2025.

[62] Pensée artificielle. Architecture RNN, https://penseeartificielle.fr/tout-pour-biendebuter-en-deep-learning-5/architecture-rnn-2/, consulté le 20 janvier 2025.

[63] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, Biological Cybernetics, vol. 36, pages 193–202, 1980.

[64] Y. LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner, « Gradient-based learning applied to document recognition », Proceedings of the IEEE, Vol. 86, N°11, pages 2278 – 2324, 1998.

[65] D. Ciresan, U. Meier, J. Masci, L. M. Gambardella, J.. Schmidhuber. Flexible, High Performance Convolutional Neural Networks for Image Classification, Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, 1237-1242, Switzerland, 2011.

[66] A. Kothiya. Understanding "convolution" operations in CNN - Analytics Vidhya - Medium. Medium. https://medium.com/analytics-vidhya/convolution-operations-in-cnn-deep-learning-compter-vision-128906ece7d3, juin 2021, consulté le 20 janvier 2025.

[67] M. Mazir. Convolutional AI Neural Network pour la détection du port du masque, mémoire de Master en électronique, Université de Bejaia, 2022.

[68] B. El Habib. Les réseaux de neurones convolutifs. https://datasciencetoday.net/index.php/en-us/deep-learning/173-les-reseaux-de-neurones-convolutifs, consulté le 20 janvier 2025.

[69] T. keldenish. CNN et Couche de Convolution, qu'est-ce que c'est ? Inside Machine Learning, 2021, https://inside-machinelearning.com/cnn-couche-deconvolution/, consulté le 20 janvier 2025.

[70] M. BEN LAZREG. Recherche de l'information dans les réseaux de neurones convolutifs pré-entraînés, mémoire de master en génie de la production automatisée, école de technologie supérieure du Québec, Canada, 2020.

[71] Open classrooms. Découvrez les différentes couches d'un CNN, https://openclassrooms.com/fr/courses/4470531-classez-et-segmentez-des-donneesvisuelles/5083336-decouvrez-les-differentes-couches-dun-cnn, consulté le 20 janvier 2020.

[72] R. Tavenard. Régularisation — Introduction au Deep Learning, https://rtavenar.github.io/deep_book/fr/content/fr/regularization.html, consulté le 20 janvier 2025.

[73] X. Glorot, Y. Bengio. Understanding the difficulty of training deep feedforward neural networks, Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010), vol. 9, pages 249–256, 2010.

[74] R. Travenard. Optimisation, https://rtavenar.github.io/deep_book/fr/content/fr/optim.html, consulté le 20 janvier 2025.

[75] A. Krizhevsky, I. Sutskever, G.. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks, Advances in Neural Information Processing Systems vol. 25, pages 1106-1114, 2012.

[76] P. Marcelino. Transfer learning from pre-trained models, https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751, consulté le 20 janvier 2025.

[77] Data Scientist. Transfer learning : qu'est-ce que c'est ? https://datascientest.com/transfer-learning, consulté le 20 janvier 2025.

[78] A. Zhang, Z. Lipton, M. Li, A. Smola. Dive into deep learning, fine tuning, http://d2l.ai/chapter_computer-vision/fine-tuning.html, consulté le 20 janvier 2025.

[79] R. Chelghom, I. Ameur. Classification de données médicales par transfert learning, Mémoire de Master en informatique, université de Constantine 2, 2022.

[80] M. A. Briki. Classification des données médicales par fine-tuning, mémoire de master en électronique, université de Tlemcen, 2021.

[81] A. Bhorkar, K. Zhang, J. Wang, "DeepAuto: A Hierarchical Deep Learning Framework for Real-Time Prediction in Cellular Networks, thèse de doctorate n télécommunications, 2019.

[82] Zhou, Y., Wang, B., Wang, L., Zhang, R., & Gao, X. (2017). Time Advance Based User Positioning for LTE Networks. IEEE Communications Letters, vol. 21, N°3, 580–583. doi:10.1109/LCOMM.2016.2640000.

[83] Huawei Technologies Co., Ltd., MAOS System User Guide – Multi-vendor Access Operation System, 2018, https://support.huawei.com, consulté le 20 janvier 2025.

[84] Rania Behloul & Rym Bardadi, Application du Machine Learning dans la prédiction des KPI de mobilité en LTE, Mémoire de Master en télécommunications, USTHB, 2020.

[85] Huawei Technologies Co., Ltd., U2020 Product Documentation, Technical report, 2022.

[86] Alhassan Mumuni, Fuseini Mumuni. Automated data processing and feature engineering for deep learning and big data applications: a survey, pages 1-38, arXiv:2403.11395, https://doi.org/10.1016/j.jiixd.2024.01.002, https://arxiv.org/pdf/2403.11395, 2024, consulté le 20 avril 2025.

[87] H. A. Alenizy & J. Berri. Transforming tabular data into images for classification with CNNs, Scientific reports, vol. 15, pages 1-14, https://doi.org/10.1038/s41598-025-01568-0, 2025.

[88] F. Fanelli, Francesca. ML Optimization of Cell-Range Overshooting Detection in Real LTE Networks, mémoire de master en télécommunications, école Polytechnique de Torino, 2020.

[89] M. R. Tanhatalab, H. Yousefi, H. M. Hosseini, M. M. Bonab, V. Fakharian .H. Abarghouei. Deep RAN: A Scalable Data-driven Platform to Detect Anomalies in Live Cellular Network Using Recurrent Convolutional Neural Network, arXiv:1911.04472, pages 1-6, https://arxiv.org/pdf/1911.04472, consulté le 20 avril 2025.

[90] K. Simonyan et A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv :1409.1556, pages 1-14, https://doi.org/10.48550/arXiv.1409.1556, 2015, consulté le 20 avril 2025.

[91] D. (Daniel), « VGG : en quoi consiste ce modèle ? Daniel vous dit tout ! », DataScientist. https://datascientest.com/quest-ce-que-le-modele-vgg, consulté le 25 juin 2025.

[92] O. More. VGG-16 : convolutional neural network», 2020, https://medium.com/@ommore524/vgg-16-convolution-neural-networkbae747a7494a, consulté le 20 mars 2025.

[93] Keras. EfficientNet B0 to B7, https://keras.io/api/applications/efficientnet/, consulté le 20 janvier 2025.

[94] Mingxing Tan and Quoc V. Le, « EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, https://arxiv.org/abs/1905.11946, 2019, consulté le 7 mai 2025.

[95] N. Klinger. MobileNet : Efficient Deep Learning for Mobile Vision, https://viso.ai/deep-learning/mobilenet-efficient-deep-learning-for-mobile-vision, consulté le 25 juin 2025.

[96] Delwar Hossain, Masudul Haider Imtiaz, Tonmoy Ghosh, Viprav Raju, Edward Sazonov.Real-Time Food Intake Monitoring Using Wearable Egocnetric Camera, 42nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) in

conjunction with the 43rd Annual Conference of the Canadian Medical and Biological Engineering Society, pages 4191-4195, 2020.

[97] A. G. Howard, M. Zhu. B. Chen, D. Kalenichenko. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, 2017, https://arxiv.org/abs/1704.04861,2017, consulté le 6 mai 2025.

[98] Andrii Polukhin. MobileNet Architectures, https://medium.com/@pandrii000/mobilenet-architectures-17fe7406d794, 2022, consulté le 20 janvier 2025.

[99] S. Bangar. Resnet Architecture Explained Medium. *Medium*. https://medium.com/@siddheshb008/resnet-architecture-explained-47309ea9283d, juillet 2022, consulté le 20 janvier 2025.

[100] I. Azeem. Understanding ResNet Architecture: A Deep Dive into Residual Neural Network, https://medium.com/@ibtedaazeem/understanding-resnet-architecture-a-deep-dive-into-residual-neural-network-2c792e6537a9, novembre 2023, consulté le 20 Janvier 2025.

[101] D. Skiöld & S. Arora, Investigating the Use of Machine Learning in KPI Prediction, Mémoire de Master, KTH, 2022.

[102] Complex systems and AI. Métriques pour la classification, https://complex-systemsai.com/analyse-des-donnees/metriques-pour-la-classification, consulté le 20 janvier 2025.

[103] Python. https://docs.python.org/3/, consulté le 20 janvier 2025.

[104] Pandas. Python Data Analysis Library, https://pandas.pydata.org/, consulté le 20 Janvier 2025.

[105] NumPy. https://numpy.org/, consulté le 20 janvier 2025.

[106] Matplotlib. Visualization with Python, https://matplotlib.org/, consulté le 20 Janvier 2025.

[107] Scikit-learn. Machine learning in Python, scikit-learn documentation, https://scikit-learn.org/stable/, consulté le 20 Janvier 2025.

[108] TensorFlow. https://www.tensorflow.org/, consulté le 20 janvier 2025.

[109] Keras. Deep Learning for humans, https://keras.io/, consulté le 20 Janvier 2025.

[110] Keras. Keras Tuner, https://keras.io/keras_tuner/, consulté le 20 avril 2025.

[111] Jupyter. https://jupyter.org/, consulté le 20 Janvier 2025.

[112] Google colab. https://colab.research.google.com/#, consulté le 20 janvier 2025.

[113] Kaggle. https://www.kaggle.com/code, consulté le 20 janvier 2025.