République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université – Ain Témouchent - Belhadj Bouchaib
Faculté des Sciences et de la Technologie
Département de Mathématiques et Informatique

Projet de Fin d'Études

Pour l'obtention du diplôme de Master en : Informatique
Domaine : Mathématiques et Informatique
Filière : Informatique
Spécialité : Cyber Sécurité et Intelligence Artificielle

Thème

Segmentation Sémantique des Images Satellitaires pour la Surveillance Environnementale

Présenté Par :
Melle LECHLECHE Maroua
Melle MAKNI Fatima Zohraa

Devant le jury composé de :

Mme BOUHALOUAN D	MCB	UAT.B.B (Ain Temouchent)	Présidente
Mme BARNOUSSI M	MAA	UAT.B.B (Ain Temouchent)	Examinatrice
Mr MESSAOUDI M.A	MCB	UAT.B.B (Ain Temouchent)	Encadrant

Année Universitaire 2024/2025

Dédicace

À celle qui fut ma lumière dans les moments sombres, à celle qui a veillé tard pour moi, priant en silence et souriant malgré la fatigue...
À toi, ma chère maman, je te dois tant. Ton amour et ton dévouement sont les piliers de mon parcours.
À toi, mon père, symbole de patience et de force, merci pour ton soutien constant et tes précieux conseils. Tu as toujours cru en moi.
À mes chers frères, Imad Eddine et Mohamed, qu'Allah vous garde toujours à mes côtés, soutien fidèle et bras solides dans chaque étape de ma vie.
À mon grand-père bien-aimé Belghezzel Ali, ton cœur généreux, tes paroles sages et tes prières ont toujours été une source de réconfort et d'inspiration.
À toi, Lechleche Maroua, ma binôme dans ce projet, pour tous les moments de fatigue, de stress et de joie que nous avons partagés, merci du fond du cœur. Ce succès, nous le vivons ensemble : il est le fruit de nos efforts communs.
À la famille de mon père et à celle de ma mère, merci pour l'amour, les prières et la bienveillance. Vous êtes les racines profondes de tout ce que je suis.
À mes amis les plus chers, merci pour les sourires partagés, les épaules sur lesquelles j'ai pu m'appuyer, et les moments inoubliables...
Et tout particulièrement à Meriem, ta présence, ton écoute et ta gentillesse ont été inestimables tout au long de ce chemin.

Fatima

Dédicace

« Les fruits de l'effort se cueillent avec la prière d'une mère, le sourire d'un père,
la patience d'un frère, la tendresse des sœurs et la fidélité d'un ami. »
Je dédie ce travail, humble fruit de persévérance et de passion,
à ceux qui ont été les lumières de mon chemin.
À mes chers parents, Lechleche Saïd et Bouhadjela Fatna,
vous êtes le fondement de tout ce que j'ai accompli.
Votre amour inconditionnel, vos prières silencieuses et vos sacrifices discrets
ont été les piliers de ma réussite.
À mes sœurs adorées, Soulef, Safaa et Aïcha,
merci pour votre affection douce et votre présence rassurante.
À mon frère Oussama,
ta discrétion est une force, ta fierté un moteur.
À mon fidèle ami Maloum Rihem,
pour ton soutien indéfectible, ton écoute sincère et ta présence constante.
À ma binôme et amie Makni Fatima Zohraa,
avec qui j'ai partagé le travail, les idées, les défis et les sourires.
À vous tous, je dédie ces pages avec tout mon respect,
mon amour et ma gratitude.

Maroua

Remerciements

Louange à Dieu, Seigneur des mondes, qui nous a accordé la force, la patience et la détermination pour mener à bien ce travail. Nous Lui demandons de le rendre sincèrement dédié à Sa cause, et de nous accorder la réussite dans tout ce qui est bien. Nous exprimons notre plus sincère gratitude à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce mémoire. Nous tenons à témoigner notre profonde reconnaissance à notre encadrant, Monsieur MESSAOUDI M.A, pour son soutien constant, ses conseils avisés et son suivi rigoureux. Nos remerciements s'adressent également à l'ensemble des enseignants du département de Mathématiques et Informatique pour la qualité de leur enseignement et leur dévouement tout au long de notre formation. Nous exprimons notre profonde reconnaissance aux membres du jury, Madame BOUHALOUAN D. ainsi que Madame BARNOUSSI M., pour l'honneur qu'elles nous font en évaluant ce travail, ainsi que pour leurs remarques pertinentes et enrichissantes. Nos sincères remerciements vont également à nos collègues et amis pour leur appui moral, leur disponibilité et l'esprit de solidarité dont ils ont toujours fait preuve. Enfin, nous exprimons tout notre amour et notre gratitude à nos familles, véritables piliers de notre réussite, pour leurs encouragements constants, leur soutien inconditionnel et leurs prières, qui nous ont permis d'atteindre cet objectif.

Résumé

Ce projet s'intéresse à la **segmentation sémantique** des images satellitaires à l'aide d'architectures de *deep learning*. Cette tâche consiste à attribuer une étiquette à chaque pixel d'une image afin d'identifier automatiquement les différents éléments présents dans une scène (sols nus, végétation, zones urbaines, plans d'eau, etc.). Après une étude approfondie des approches classiques de segmentation et des principes du deep learning, plusieurs architectures ont été implémentées et comparées.

Les expérimentations menées sur une base d'images satellitaires annotées ont permis d'évaluer les performances de chaque modèle à l'aide de métriques telles que la précision (*accuracy*), le *Dice coefficient* et l'analyse des matrices de confusion. Les résultats montrent que les modèles intégrant des *backbones* pré-entraînés offrent une segmentation plus fine et généralisable, notamment dans les zones à forte hétérogénéité visuelle.

Ce travail démontre l'intérêt croissant de l'intelligence artificielle dans l'exploitation des données de télédétection, et ouvre la voie à des systèmes automatisés d'aide à la décision dans des domaines critiques tels que l'environnement, la sécurité et la gestion des risques.

Mots-clés : segmentation sémantique, images satellites, deep learning, réseaux convolutifs, transfert learning, télédétection.

Abstract

This project focuses on the semantic segmentation of satellite images using deep learning architectures. The task involves assigning a label to each pixel in an image to automatically identify the different elements present in a scene (bare soil, vegetation, urban areas, water bodies, etc.). Following an in-depth study of classical segmentation approaches and deep learning principles, several architectures were implemented and compared.

Experiments conducted on an annotated satellite image dataset enabled the evaluation of each model's performance using metrics such as accuracy, Dice coefficient, and confusion matrix analysis. The results show that models integrating pre-trained backbones provide finer and more generalizable segmentation, especially in visually heterogeneous areas.

This work highlights the growing importance of artificial intelligence in the processing of remote sensing data and paves the way for automated decision-support systems in critical fields such as environmental monitoring, security, and risk management.

Keywords : semantic segmentation, satellite images, deep learning, transfer learning, remote sensing .

الملخص

يركز هذا المشروع على التقسيم الدلالي للصور الساتلية باستخدام بنى التعلم العميق. تتمثل هذه المهمة في إعطاء تسمية لكل بكسل في الصورة لتحديد العناصر المختلفة الموجودة في المشهد تلقائيًا (مثل الأراضي العارية، الغطاء النباتي، المناطق الحضرية، المسطحات المائية، وغيرها). بعد دراسة معمقة للمقاربات التقليدية للتقسيم ومبادئ التعلم العميق، تم تنفيذ ومقارنة عدة نماذج

سمحت التجارب المنجزة على قاعدة بيانات من الصور الساتلية الموسومة بتقييم أداء كل نموذج باستخدام مؤشرات مثل الدقة ، وتحليل مصفوفات الالتباس. أظهرت النتائج أن النماذج التي تستخدم نواة مدربة مسبقًا تحقق Dice ، ومعامل(accuracy) .تقسيمًا أكثر دقة وقدرة على التعميم، خاصة في المناطق التي تتميز بتنوع بصري كبير

يثبت هذا العمل الأهمية المتزايدة للذكاء الاصطناعي في استغلال بيانات الاستشعار عن بعد، ويمهد الطريق لتطوير أنظمة آلية لدعم اتخاذ القرار في مجالات حيوية مثل البيئة، والأمن، وإدارة المخاطر

الكلمات المفتاحية:التجزئة الدلالية، الصور الساتلية، التعلم العميق، الشبكات الالتفاف، مقاييس التقييم ، الاستشعار عن بعد التعلم بالنقل

Liste des abréviations

AI	Intelligence Artificielle
ANN	Artificial Neural Network (Réseau de neurones artificiels)
CNN	Convolutional Neural Network (Réseau de neurones convolutifs)
DL	Deep Learning (Apprentissage profond)
DNN	Deep Neural Network
FCN	Fully Convolutional Network
GIS	Geographic Information System
GPU	Graphics Processing Unit
IoU	Intersection over Union
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
OHEM	Online Hard Example Mining
ReLU	Rectified Linear Unit
RGB	Red Green Blue (Couleurs primaires numériques)
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
TNR	True Negative Rate
TPR	True Positive Rate
U-Net	Architecture CNN pour la segmentation sémantique
U-Net++	Variante avancée du U-Net
VGG	Visual Geometry Group (réseau convolutif VGG)
IoT	Internet of Things
AE	Autoencoder
BCE	Binary Cross Entropy
CE	Cross Entropy
DICE	Dice Similarity Coefficient
DLv3+	DeepLabV3+
TPU	Tensor Processing Unit
mIoU	Mean Intersection over Union
SatSeg	Satellite Segmentation (abréviation de la plateforme développée)

Table des matières

Dédicace **1**

Dédicace **2**

Remerciements **3**

Résumé **4**

Liste des abréviations **6**

Introduction Générale **13**

1 Télédetection **15**
1.1 Introduction à la Télédétection 15
1.2 Définition 15
1.3 Les Etapes de Télédétection 16
1.3.1 Source d'énergie ou d'illumination (A) 16
1.3.2 Rayonnement et atmosphère (B) 16
1.3.3 Interaction avec la cible (C) 16
1.3.4 Enregistrement de l'énergie par le capteur (D) 17
1.3.5 Transmission, réception et traitement (E) 17
1.3.6 Interprétation et analyse (F) 17
1.3.7 Application (G) 17
1.4 Les Domain d'application de La Télédétection 17
1.4.1 Eaux de surface 17
1.4.2 Nappes phréatiques : aquifère de surface et proche de la surface . . . 17
1.4.3 Qualité de l'eau 17
1.5 Les Rayonnement électromagnétique 18
1.5.1 Définition 18
1.5.2 Les ondes électromagnétiques 18
1.5.3 Rayonnement et énergie 19
1.5.4 Le spectre électromagnétique 19
1.6 Systèmes d'observation : 21
1.6.1 Les capteurs 21
1.6.2 Types d'acquisition 21
1.6.3 Modes d'acquisition 22
1.6.4 Résolution 22
1.7 Image Satellitaire 22
1.7.1 Introduction 22
1.7.2 Définition 23

1.7.3 Traitement numérique de l'image satellite . . . 23
1.8 Conclusion : . . . 23

2 La Segmentation des Images **25**
2.1 Introduction . . . 25
2.2 Notions fondamentales de l'image numérique . . . 25
2.2.1 Définition d'une image . . . 25
2.2.2 Définition d'une image numérique . . . 25
2.2.3 Types d'images . . . 26
2.2.4 Propriétés caractéristiques d'une image . . . 26
2.3 Introduction à la segmentation d'image . . . 27
2.3.1 Définition . . . 27
2.3.2 Types de segmentation . . . 27
2.3.3 Comparaison des approches . . . 29
2.3.4 Domaines d'application . . . 29
2.4 Approches classiques de segmentation . . . 29
2.4.1 Segmentation Basée sur les contours . . . 29
2.4.2 Segmentation Basée sur les régions . . . 30
2.4.3 Méthodes Par seuillage . . . 30
2.4.4 Classification des pixels . . . 31
2.4.5 Classification supervisée et non supervisée . . . 31

3 Deep Learning **33**
3.1 Historique . . . 33
3.2 Introduction au deep learning : . . . 34
3.3 Domaines d'application . . . 34
3.4 Fondamentaux du Deep Learning . . . 34
3.4.1 Définition . . . 34
3.4.2 Neurones biologiques vs artificiels . . . 35
3.4.3 Perceptron et limites . . . 35
3.4.4 Réseaux de neurones Multicouche (MLP) . . . 36
3.5 Architectures des Réseaux Neuronaux . . . 40
3.5.1 Réseaux de neurones convolutifs (CNN) . . . 40
3.5.2 Pooling, Flattening, Classification . . . 42
3.5.3 Réseaux de Neurones Récurrents (RNN, LSTM, GRU) . . . 44
3.5.4 Autoencodeurs (AE et VariationalAE) : . . . 46
3.5.5 Normalisation dans les réseaux de neurones (BatchNorm, LayerNorm) : . . . 47
3.5.6 Techniques de régularisation : . . . 48
3.5.7 Apprentissage par transfert (Transfer Learning) : . . . 49
3.6 Architectures Avancées pour la Vision par Ordinateur : . . . 49
3.6.1 U-Net : Structure en U pour la segmentation : . . . 49
3.6.2 Fully Convolutional Networks (FCN) : . . . 51
3.6.3 SegNet : encodeur-décodeur . . . 52
3.6.4 VGG, ResNet, EfficientNet : . . . 52
3.6.5 U-Net++ et DeepLabV3+ : . . . 54
3.6.6 Comparaison des architectures [46]. . . . 56
3.7 Entraînement et Évaluation d'un Réseau de Neurones . . . 56
3.7.1 Préparation des données (train, val, test) . . . 56
3.7.2 Fonctions de perte (loss functions) : . . . 57

3.7.3 Optimiseurs : SGD, Adam, RMSprop . . . 58
3.7.4 Problèmes courants : . . . 58
3.7.5 Métriques d'évaluation . . . 58

4 Implémentation et discussion des résultats 61
4.1 Environnements et outils de développement . . . 61
4.1.1 Google Colab . . . 61
4.1.2 Python : . . . 62
4.1.3 Les bibliothèques utilisées . . . 63
4.2 Base de données utilisée . . . 64
4.3 Notre démarche pour la segmentation sémantique en utilisant le Deep Learning . . . 66
4.3.1 Chargement des données . . . 66
4.3.2 Préparation des données . . . 67
4.3.3 Prétraitement . . . 67
4.3.4 Entraînement . . . 68
4.3.5 Métriques d'évaluation : . . . 70
4.3.6 Architecture du modèle utilisér : . . . 70
4.4 Les résultats obtenus de notre approche : . . . 76
4.4.1 Analyse comparative des performances des architectures de segmentation sémantique . . . 76
4.4.2 Analyse des matrices de confusion pour les modèles de segmentation 78
4.4.3 Comparaison avec notre modèle CNN : . . . 81
4.5 Les défis auxquels nous avons été confrontés . . . 84
4.5.1 Préparation des données . . . 84
4.5.2 Déséquilibre des classes . . . 84
4.5.3 Choix et réglage des architectures . . . 84
4.5.4 Accès aux ressources sur Google Colab . . . 84
4.5.5 Optimisation du compromis biais-variance . . . 85
4.6 Interface graphique : . . . 85

Conclusion 94

Table des figures

1.1 Les étapes de Télédétection 16
1.2 Nature et propagation d'une onde électromagnétique 18
1.3 Le spectre électromagnétique 20

2.1 Reconnaissance et segmentation simultanées à l'aide de TextonBoost . . . 28
2.2 Détection des points clés des personnes et segmentation à l'aide de Mask . . 28
2.3 Résultats de segmentation panoptique 29

3.1 Biological (top) vs. artificial (bottom) neurons 35
3.2 perceptron 36
3.3 Fonctions d'activation classiques (fixes) avec leur gamme et leur continuité . 40
3.4 Exemple de convolution 2-D sans basculement du noyau. 41
3.5 Effet du zero-padding sur la taille des réseaux convolutifs. 42
3.6 Max pooling introduit une invariance. 43
3.7 Composants typiques d'un couche CNN. 43
3.8 Architecture RNN 44
3.9 Schéma fonctionnel d'une "cellule" de réseau récurrent LSTM. 45
3.10 Schéma général d'un encodeur automatique 47
3.11 architecture U-net 50
3.12 Fully convolutional networks 51
3.13 Architecture SegNet 52
3.14 Architecture VGG. 53
3.15 Residual learning : a building block. 53
3.16 Une fonction résiduelle F plus profonde pour ImageNet. 54
3.17 (a) L'architecture UNet++ est composée d'un encodeur et d'un décodeur . 55
3.18 DeepLabv3 en adoptant une structure encodeur-décodeur 56

4.1 Logo de Google Colab 62
4.2 Logo de Python 62
4.3 fonction de load_data() 66
4.4 fonction de DataGenerator 67
4.5 fonction de preprocess image mask 68
4.6 fonction de loss 68
4.7 Structure générale de segmentation sémantique par apprentissage profond . 71
4.8 Évolution de la loss, du Dice coefficient et de l'accuracy pour le modèle U-Net sur les ensembles d'entraînement et de validation. 76
4.9 Évolution de la loss, du Dice coefficient et de l'accuracy pour le modèle VGG16 sur les ensembles d'entraînement et de validation. 77
4.10 Évolution de la loss, du Dice coefficient et de l'accuracy pour le modèle VGG19 sur les ensembles d'entraînement et de validation. 77

4.11 Évolution de la loss, du Dice coefficient et de l'accuracy pour le modèle ResNet50 sur les ensembles d'entraînement et de validation. 78
4.12 Matrice de confusion du modèle U-Net 79
4.13 Matrice de confusion du modèle VGG16 79
4.14 Matrice de confusion du modèle VGG19 80
4.15 Matrice de confusion du modèle ResNet50 81
4.16 Rapport de classication 81
4.17 Les résultats de notre segmentation sémantique sur les données de validation de U-Net. 82
4.18 Les résultats de notre segmentation sémantique sur les données de validation de VGG16. 83
4.19 Les résultats de notre segmentation sémantique sur les données de validation de VGG19. 83
4.20 Les résultats de notre segmentation sémantique sur les données de validation de ResNet. 84
4.21 connexion à la plateforme SatSegment 85
4.22 Interface graphique pour la segmentation sémantique d'images satellites à l'aide du modèle U-Net 86
4.23 Interface graphique pour la segmentation sémantique d'images satellites à l'aide du modèle U-Net mode sombre 86
4.24 Interface graphique de SatSegment affichant les performances d'entraînement des variantes de U-Net 87
4.25 Évaluation des performances du modèle U-Net Simple – Précision et Matrice de Confusion 87
4.26 Évaluation détaillée par classe des performances du modèle U-Net Simple sur des images satellitaires 88
4.27 Comparaison visuelle des résultats de segmentation obtenus par différents modèles U-Net sur une image satellitaire urbaine 88
4.28 Comparaison des courbes de perte (Loss) entre différents modèles U-Net selon le backbone utilisé 89
4.29 Comparaison des courbes de perte (dice) entre différents modèles U-Net selon le backbone utilisé 89
4.30 Comparaison des courbes de perte (Accuracy) entre différents modèles U-Net selon le backbone utilisé 90
4.31 Comparaison des courbes de perte (matrice-confusion) entre différents modèles U-Net selon le backbone utilisé 90
4.32 Documentation de l'interface SatSegment pour la segmentation sémantique d'images satellites 91
4.33 Rapport de notre projet de fin d'études sur Segmentation Sémantique des Images Satellitaires pour la Surveillance Environnementale 91
4.34 présentation de la plateforme SatSegment 92
4.35 Accès aux paramètres de l'utilisateur avec l'option de déconnexion 92

Liste des tableaux

3.1 Comparaison des architectures CNN pour la segmentation 56

4.1 Codes des classes et leurs significations 65
4.2 Architecture du modèle U-Net avec fonctions d'activation 71
4.3 Architecture du modèle U-Net basé sur VGG16 avec Dropout 73
4.4 Architecture du modèle U-Net basé sur VGG19 avec Dropout 74
4.5 Architecture du modèle U-Net avec ResNet50 75
4.6 Comparaison des performances des modèles de segmentation 82

Introduction Générale

Depuis plusieurs décennies, la planète est confrontée à une transformation rapide de son environnement sous l'effet combiné de l'activité humaine et des phénomènes naturels. L'urbanisation massive, la déforestation, l'extension des terres agricoles, les incendies, les inondations ou encore le changement climatique ont profondément modifié les écosystèmes terrestres.
Dans ce contexte, la capacité à observer, analyser et comprendre ces changements à grande échelle est devenue essentielle, tant pour la recherche scientifique que pour la prise de décision dans les domaines de l'aménagement du territoire, de la gestion des ressources ou de la protection de l'environnement.
C'est dans cette perspective que la télédétection s'est imposée comme un outil précieux et incontournable. Grâce aux satellites et aux capteurs embarqués, elle permet l'acquisition d'images couvrant de vastes étendues géographiques, à intervalles réguliers, et avec une diversité de résolutions et de spectres. Ces images satellitaires constituent une source d'information extrêmement riche, permettant d'observer l'évolution de la surface terrestre au fil du temps, d'identifier des phénomènes invisibles à l'œil nu, et de cartographier précisément l'occupation du sol.

Cependant, l'exploitation efficace de ces images nécessite des traitements complexes. Face à la grande quantité de données générées quotidiennement et à la diversité des scènes observées, les approches manuelles ou classiques atteignent rapidement leurs limites. Le besoin d'automatiser l'analyse des images satellites est donc devenu un enjeu majeur dans le domaine du traitement d'images. Parmi les tâches fondamentales de cette analyse, la segmentation d'image joue un rôle central. Elle consiste à diviser l'image en régions homogènes, selon certains critères, dans le but de faciliter la compréhension de son contenu. Appliquée aux images satellitaires, la segmentation sémantique permet d'attribuer une étiquette à chaque pixel – par exemple, «eau», «forêt», «bâtiment», ou «sol nu».

Néanmoins, les méthodes traditionnelles de segmentation, basées sur des techniques de seuillage, de contours ou de régions, montrent rapidement leurs limites dans le cas des images satellitaires. Leur manque de robustesse, notamment en présence de bruit, de variations d'illumination ou de structures complexes, réduit leur efficacité dans les applications réelles. C'est dans ce cadre que les progrès récents en intelligence artificielle, et plus particulièrement en apprentissage profond (Deep Learning), ont ouvert de nouvelles perspectives. Les réseaux de neurones convolutifs (CNN), capables d'apprendre automatiquement des représentations à partir de grandes quantités d'images, ont révolutionné le domaine du traitement d'images, en particulier pour les tâches de reconnaissance d'objets, de classification et de segmentation.

Ce mémoire s'inscrit dans cette dynamique. Il a pour objectif de concevoir et d'évaluer des méthodes de segmentation sémantique d'images satellites basées sur le Deep Learning. En s'appuyant sur des architectures performantes comme U-Net ou DeepLabV3+, l'objectif est d'automatiser la classification des pixels dans les images, et de générer des cartes thématiques détaillées permettant une meilleure interprétation des scènes observées. Ce travail vise à démontrer l'intérêt de ces approches dans le cadre d'applications concrètes liées à la télédétection.

Afin d'atteindre cet objectif, ce mémoire est structuré en quatre chapitres complémentaires, qui couvrent à la fois les aspects théoriques et pratiques du projet.

Le premier chapitre est consacré à la télédétection. Il en présente les fondements théoriques, les principes de fonctionnement, les différents types de capteurs et de satellites, ainsi que les principales caractéristiques des images acquises (résolution, spectres, fréquence, etc.). Il introduit également les traitements de base appliqués à ces images, et constitue ainsi le socle nécessaire à la compréhension du travail effectué.

Le deuxième chapitre traite de la segmentation d'images. Il expose les notions fondamentales liées à l'image numérique, les différentes approches de segmentation (classique, par seuillage, par contours, par régions), et met l'accent sur la segmentation sémantique, d'instance et panoptique. Ce chapitre s'intéresse également aux domaines d'application de la segmentation, en particulier dans le cadre des images de télédétection.

Le troisième chapitre est dédié au Deep Learning. Il présente les concepts clés de l'apprentissage profond, en particulier les réseaux de neurones artificiels et les réseaux convolutifs. Il décrit ensuite les principales architectures utilisées pour la segmentation d'images, notamment U-Net, ResNet et DeepLabV3+, en expliquant leur structure, leur fonctionnement et leur intérêt dans le contexte de notre projet.

Enfin, le quatrième chapitre présente la mise en œuvre expérimentale du projet. Il décrit le jeu de données utilisé, les étapes de préparation et de traitement des images, l'implémentation des modèles sur la plateforme Google Colab, ainsi que les résultats obtenus. Ce chapitre inclut également une analyse comparative des performances, les difficultés rencontrées, et une discussion sur les perspectives d'amélioration.

À travers ce mémoire, nous espérons apporter une contribution utile à l'automatisation de l'analyse des images satellitaires, en montrant que les modèles de Deep Learning peuvent significativement améliorer la qualité, la précision et la rapidité de la segmentation sémantique. Ce travail s'inscrit dans une volonté d'exploiter pleinement le potentiel des données de télédétection au service de la connaissance, de la surveillance et de la gestion durable de notre environnement

Chapitre 1

Télédetection

Introduction

La télédétection est une technologie moderne qui permet d'observer et d'analyser la surface terrestre à distance, sans contact direct, grâce à l'utilisation de capteurs embarqués sur des satellites ou des plateformes aériennes. Cette discipline, au carrefour de plusieurs domaines scientifiques, repose essentiellement sur l'acquisition et l'interprétation d'images obtenues par le biais du rayonnement électromagnétique.

Dans ce chapitre, nous allons explorer les principes fondamentaux de la télédétection spatiale, en mettant en lumière son fonctionnement, ses principales étapes, ainsi que ses domaines d'application. Une attention particulière sera portée au rôle du rayonnement électromagnétique et aux différents types de capteurs utilisés pour collecter les données. Ce cadre théorique est indispensable pour comprendre les caractéristiques des images satellites et leur utilité dans les projets de traitement d'images, notamment dans le contexte de la segmentation sémantique abordée dans ce mémoire.

1.1 Introduction à la Télédétection

La télédétection trouve ses origines théoriques dans la combinaison de deux inventions anciennes : la montgolfière et la photographie. Toutefois, c'est véritablement avec l'essor de la photographie aérienne, particulièrement au cours du XXe siècle et plus intensément pendant la Seconde Guerre mondiale, qu'elle prend son envol, stimulée par des objectifs militaires et stratégiques. L'année 1957 marque un tournant décisif : le lancement de Spoutnik, premier satellite artificiel, symbolise l'entrée de la télédétection dans son ère moderne. Depuis, de nombreux pays comme les États-Unis, le Canada, la France, l'ex-URSS (devenue la Russie), la Chine, le Japon et l'Inde ont développé leurs propres programmes spatiaux dédiés à la télédétection. Aujourd'hui, des dizaines de satellites d'observation terrestre orbitent autour de la planète, fournissant en continu une quantité massive d'images utilisées non seulement à des fins militaires, mais également, et de plus en plus, pour des applications civiles.[1].

1.2 Définition

La télédétection désigne l'ensemble des méthodes et des outils permettant de collecter des informations, qu'elles soient qualitatives ou quantitatives, sur un objet ou une zone à distance, sans contact direct. Cette acquisition se fait principalement à l'aide de capteurs embarqués sur des satellites, mais aussi sur des avions ou des ballons, en exploitant les propriétés du rayonnement électromagnétique.[2].

1.3 Les Etapes de Télédétection

De manière plus détaillée, on peut schématiser la télédétection comme un ensemble de 7 étapes clés :

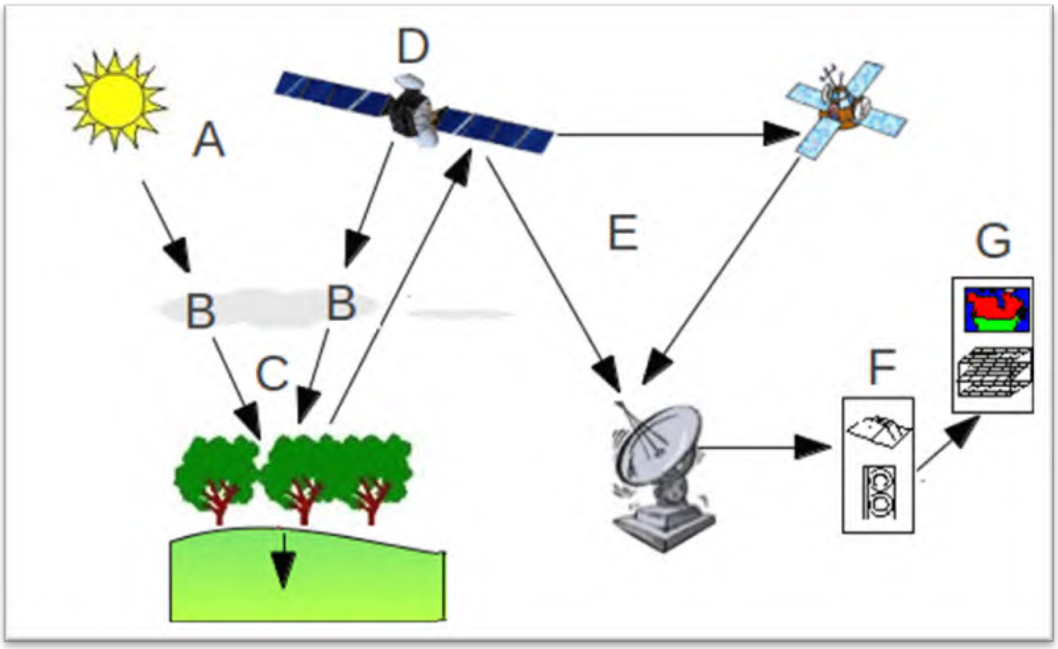


FIGURE 1.1 – Les étapes de Télédétection

1.3.1 Source d'énergie ou d'illumination (A)

À l'origine de tout processus de télédétection se trouve nécessairement une source d'énergie pour illuminer la cible. Le plus souvent, voire dans la presque totalité des cas, cette source d'énergie est le soleil. Mais le satellite lui-même peut être source d'énergie : c'est le cas pour le domaine de la télédétection radar.

1.3.2 Rayonnement et atmosphère (B)

Durant son parcours « aller » entre la source d'énergie et la cible, le rayonnement interagit avec l'atmosphère. Une seconde interaction se produit lors du trajet « retour » entre la cible et le capteur.

1.3.3 Interaction avec la cible (C)

Une fois parvenue à la cible, l'énergie interagit avec la surface de celle-ci. La nature de cette interaction dépend des caractéristiques du rayonnement et des propriétés de la surface. Chaque objet géographique émet ou réfléchi un rayonnement dans les diverses fréquences du spectre électromagnétique. Cette caractéristique s'appelle le comportement spectral. En télédétection, on suppose que tout objet ou classe d'objet sur la surface terrestre possède sa propre « empreinte digitale » dans le spectre électromagnétique (la signature spectrale), en fonction de la longueur d'onde du rayonnement qui est réfléchi ou émis par lui-même. Ainsi, une parcelle de canne à sucre aura des signatures différentes en fonction de son stade végétatif et de son niveau de maturation.

1.3.4 Enregistrement de l'énergie par le capteur (D)

Une fois l'énergie diffusée ou émise par la cible, elle doit être captée à distance par un capteur qui n'est pas en contact avec la cible mais embarqué à bord d'un satellite ou d'un avion par exemple, pour être enfin enregistrée sous format numérique.

1.3.5 Transmission, réception et traitement (E)

Cette information enregistrée par le capteur est transmise, souvent par des moyens électroniques, à une station de réception généralement située au sol où l'information est transformée en images (numériques ou photographiques).

1.3.6 Interprétation et analyse (F)

Une interprétation visuelle et/ou numérique de l'image traitée est ensuite nécessaire pour extraire l'information que l'on désire obtenir sur la cible.

1.3.7 Application (G)

La dernière étape du processus consiste à utiliser l'information extraite de l'image pour mieux comprendre la cible, c'est-à-dire la portion d'espace étudiée (une ville, une zone inondée, une forêt, etc...) pour nous en faire découvrir de nouveaux aspects ou pour aider à résoudre un problème particulier. Ces sept étapes couvrent le processus de la télédétection, du début à la fin.[3].

1.4 Les Domain d'application de La Télédétection

Opérationnel (O) ou expérimental (E)

1.4.1 Eaux de surface

- Inventaire des lacs et étangs (O)
- Délimitation des vagues d'inondation (O)
- Inventaire de la végétation flottante (O)
- Détection des nappes d'hydrocarbures (O)
- Mesures de temps de déplacement des colorants (O)
- Surveiller l'avance ou la récession des glaciers (O)
- Délimitation des terres humides (O)
- Cartographie des champs de neige (O/E)
- Inventaire des terres irriguées (O/E)
- Études de la morphologie des rives et des rivières (O/E)
- Relation des schémas de drainage à la lithologie, à la structure et à l'histoire géomorphologique (O/E)

1.4.2 Nappes phréatiques : aquifère de surface et proche de la surface

- Délimitation des limites de l'aquifère (O)
- Inférence de la porosité et de la perméabilité primaires et secondaires
- Estimations de la profondeur et de l'épaisseur saturée de la nappe phréatique (O)
- Délimitation des zones probables de recharge et de décharge ; relations entre la topographie et l'hydrologie

1.4.3 Qualité de l'eau

- Mesure de la température de l'eau (O)

— Mesure de la couleur des aquarelles, de la turbidité et de la productivité (O/E)
— Surveiller l'eutrophisation des lacs et des estuaires
— Déterminer la source, l'étendue et la dispersion des polluants (O/E)
— Mesure des polluants fluorescents (O/E) [4].

1.5 Les Rayonnement électromagnétique

1.5.1 Définition

Le rayonnement électromagnétique désigne l'ensemble des radiations émises par une source, qu'il s'agisse du Soleil, de la surface terrestre ou océanique, de l'atmosphère, ou même du capteur satellitaire lui-même. Ces radiations se propagent sous forme d'ondes électromagnétiques ou de particules.

1.5.2 Les ondes électromagnétiques

Une onde électromagnétique est constituée de deux champs oscillants : un champ électrique et un champ magnétique, vibrant à la même fréquence. Ces deux champs sont perpendiculaires l'un à l'autre, et leur propagation s'effectue dans une direction orthogonale à ces deux champs (voir figure ci-dessous). La vitesse de propagation de l'onde dépend du milieu traversé. Dans le vide, cette vitesse est constante et égale 3×10^8 m/s.

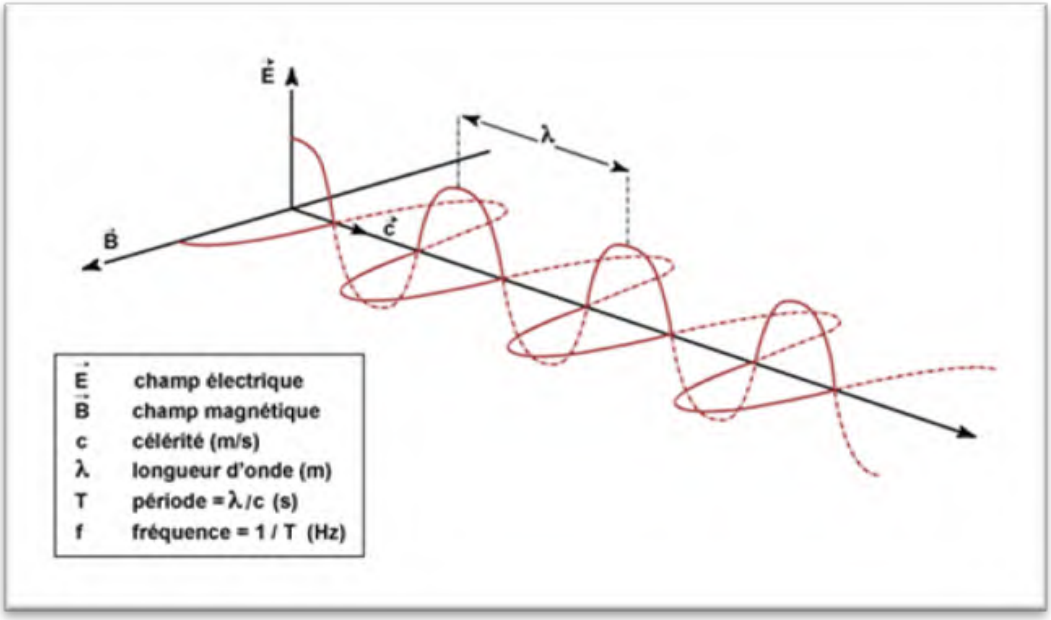


FIGURE 1.2 – Nature et propagation d'une onde électromagnétique

Une onde électromagnétique est caractérisée par plusieurs grandeurs physiques :

• **La longueur d'onde** (λ) : elle exprime le caractère oscillatoire périodique de l'onde dans l'espace.

C'est la longueur d'un cycle d'une onde, la distance séparant deux crêtes successives. Elle est mesurée en mètre ou en l'un de ses sous-multiples, les ondes électromagnétiques utilisées en télédétection spatiale ayant des longueurs d'onde relativement courtes : le nanomètre 1 nm = 10-9 mètre.
le micromètre 1 m = 10-6 mètre.
le centimètre 1 cm = 10-2 mètre.

• **La période (T)** : elle représente le temps nécessaire pour que l'onde effectue un cycle. L'unité est la seconde.
• **La fréquence (ν)** : inverse de la période, elle traduit le nombre de cycles par unité de temps.
Elle s'exprime en Hertz (Hz) - un Hz équivaut à une oscillation par seconde - ou en multiples du Hertz, les ondes électromagnétiques utilisées en télédétection spatiale ayant des fréquences très élevées : le kilohertz 1 kHz = 103 Hz
le mégahertz 1 MHz = 106 Hz
le gigahertz 1 GHz = 109 Hz
Longueur d'onde et fréquence sont inversement proportionnelles et unies par la relation suivante :
λ = c / v
où :
-λ : longueur d'onde de l'onde électromagnétique.
- c : vitesse de la lumière (3.108m.s-1).
- : la fréquence de l'onde.

Par conséquent, plus la longueur d'onde est petite, plus la fréquence est élevée, et réciproquement.

1.5.3 Rayonnement et énergie

Les transferts d'énergie transmis par le rayonnement électromagnétique entre le Soleil et le système Terre-océan-atmosphère ne s'effectuent pas de manière continue, mais par étapes discrètes. Cette énergie est transportée sous forme de paquets, appelés photons, qui sont des particules élémentaires immatérielles. Chaque photon transporte une quantité d'énergie, appelée quantum, qui est directement proportionnelle à la fréquence de l'onde électromagnétique considérée : plus la fréquence est élevée, plus l'énergie du photon est importante.

La relation suivante permet de calculer l'énergie d'un photon en fonction de la fréquence de l'onde :

$$E = h\nu$$

où :
— E : énergie de l'onde électromagnétique (en joules),
— ν : fréquence de l'onde (en hertz),
— h : constante de Planck, égale à $6{,}625 \times 10^{-34}\,\mathrm{J \cdot s}$.
Ainsi, les rayonnements électromagnétiques de courte longueur d'onde ou de fréquence élevée véhiculent davantage d'énergie que les rayonnements de grande longueur d'onde (basse fréquence).

1.5.4 Le spectre électromagnétique

Le spectre électromagnétique représente la répartition des ondes électromagnétiques en fonction de leur longueur d'onde, de leur fréquence ou bien encore de leur énergie (figure ci-dessous).

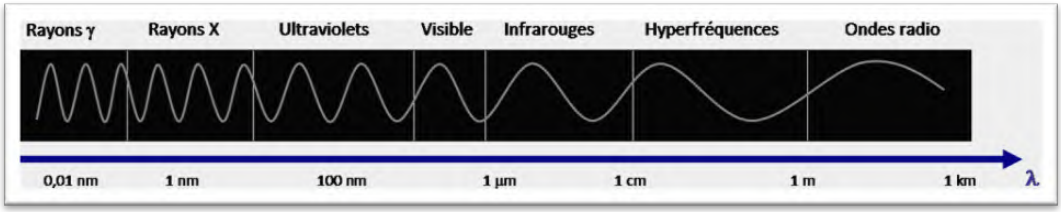


FIGURE 1.3 – Le spectre électromagnétique

En partant des ondes les plus énergétiques, on distingue successivement :

— • **Les rayons gamma (γ)** : Ils proviennent des radiations émises par des substances radioactives. Dotés d'une énergie extrêmement élevée, ces rayonnements peuvent traverser aisément la matière, ce qui les rend particulièrement dangereux pour les cellules vivantes. Leur longueur d'onde est très courte, comprise entre 10^{-14} m (soit un centième de milliardième de millimètre) et 10^{-12} m (un milliardième de millimètre).

— • **Les rayons X** : Ce sont des rayonnements électromagnétiques très énergétiques capables de traverser, avec plus ou moins de facilité, les matériaux. Bien qu'un peu moins nocifs que les rayons gamma, ils restent dangereux. Ils sont largement utilisés dans plusieurs domaines, notamment en médecine pour réaliser des radiographies, dans l'industrie pour le contrôle des bagages (comme dans les aéroports), ainsi qu'en recherche scientifique, notamment pour l'analyse de la matière via le rayonnement synchrotron. Leur longueur d'onde se situe entre 10^{-12} m (un milliardième de millimètre) et 10^{-8} m (un cent millième de millimètre).

— • **Les ultraviolets** : Ces rayonnements possèdent une énergie relativement élevée et peuvent être dangereux pour la peau. Heureusement, une grande partie de ce rayonnement est absorbée par la couche d'ozone, qui joue un rôle essentiel en protégeant les cellules vivantes. Leur longueur d'onde varie entre 10^{-8} m (un cent millième de millimètre) et 4×10^{-7} m (quatre dixièmes de millième de millimètre).

— • **Le domaine visible** : Il représente une bande très restreinte du spectre électromagnétique, que l'œil humain est capable de percevoir. C'est dans cette plage que l'intensité du rayonnement solaire est la plus élevée (environ 0,5 μm). Elle permet également de distinguer toutes les couleurs de l'arc-en-ciel, allant du bleu au rouge. Ce domaine s'étend de 4×10^{-7} m (lumière bleue) à 8×10^{-7} m (lumière rouge).

— • **L'infrarouge** : Ce rayonnement est émis par tout corps ayant une température supérieure au zéro absolu ($-273\,°$C). En télédétection, certaines bandes spectrales de l'infrarouge sont utilisées pour déterminer la température des surfaces terrestres, des océans ainsi que des nuages. Le domaine infrarouge s'étend sur une plage de longueurs d'onde allant de 8×10^{-7} m jusqu'à 10^{-3} m.

— • **Les ondes radar ou hyperfréquences** : Cette partie du spectre est utilisée pour détecter le rayonnement émis par la surface terrestre, s'apparentant ainsi à la télédétection en infrarouge thermique. Elle est aussi exploitée par des capteurs actifs tels que les radars. Un capteur radar émet son propre rayonnement électro-

magnétique et, en analysant le signal rétrodiffusé, il peut localiser et identifier des objets, ainsi que mesurer leur vitesse de déplacement s'ils sont en mouvement. Cette méthode fonctionne indépendamment de la couverture nuageuse, de jour comme de nuit. Le domaine des hyperfréquences couvre des longueurs d'onde allant du centimètre jusqu'au mètre.

— • **Les ondes radio** : Ce domaine, qui couvre les longueurs d'onde les plus étendues du spectre électromagnétique, correspond aux ondes de plus basses fréquences, allant de quelques centimètres à plusieurs kilomètres. Faciles à générer et à capter, les ondes radio sont largement utilisées pour transmettre des informations, notamment en radio, télévision et téléphonie. Par exemple, la bande FM des radios a des longueurs d'onde d'environ un mètre, tandis que celles des téléphones mobiles se situent autour de 10 centimètres. Alors que l'œil humain ne perçoit le rayonnement que dans une très fine portion du spectre le domaine visible, compris entre 0,4 μm et 0,7 μm les capteurs satellitaires exploitent une gamme beaucoup plus large du spectre électromagnétique.

En télédétection spatiale, on utilise principalement trois fenêtres spectrales :

— Le domaine du visible ;
— Le domaine des infrarouges, comprenant l'infrarouge proche (IR proche), moyen (IR moyen) et thermique (IR thermique) ;
— Le domaine des micro-ondes ou hyperfréquences (qui ne sera pas détaillé ici, bien qu'il joue un rôle crucial notamment en télédétection RADAR).

Par ailleurs, certains capteurs, moins nombreux, sont capables de mesurer l'énergie du rayonnement ultraviolet. Ces capteurs sont principalement employés en astronomie pour l'étude des atmosphères planétaires ou pour quantifier la quantité d'ultraviolets atteignant la surface terrestre. In télédétection aérienne, the UV proche, comprising between 250 and 350 nm, is exploited in océanographic applications, notamment to identify and cartographier the hydrocarbures nappes.[5].

1.6 Systèmes d'observation :

Ils représentent à la fois les instruments de mesures du rayonnement électromagnétique (capteurs), et les systèmes de télédétection (plates formes).

1.6.1 Les capteurs

Les capteurs sont des instruments capables de capter le rayonnement émis ou réfléchi par la Terre, puis de le convertir en un signal exploitable pour stocker l'information. Ils se distinguent selon leur type d'acquisition (passif ou actif), leur mode d'acquisition et leur résolution.

1.6.2 Types d'acquisition

Capteurs passifs

Le Soleil est la source d'énergie utilisée en télédétection passive. Cette énergie peut être soit réfléchie (notamment dans le domaine visible), soit absorbée et réémise (comme dans le cas de la fluorescence ou de l'infrarouge thermique). Les capteurs passifs mesurent donc l'énergie naturellement disponible.

Les capteurs actifs

Ils produisent leur propre énergie pour éclairer la cible, puis mesurent le signal rétrodiffusé en retour. Ces capteurs, souvent des radiomètres fonctionnant dans le domaine des hyperfréquences, ont l'avantage de pouvoir effectuer des mesures à tout moment, indépendamment de la lumière du jour ou de la saison.

1.6.3 Modes d'acquisition

Le mode d'acquisition d'un capteur détermine la manière dont les données sont enregistrées en balayant la surface terrestre. Il peut être :

- **À balayage électronique :** ce type d'acquisition est utilisé par exemple par le satellite SPOT, qui dispose d'une barrette linéaire de détecteurs balayant une bande de terrain de 60 km de large. On parle alors de *balayage ligne*.
- **À balayage mécanique :** dans ce cas, un miroir rotatif est utilisé pour balayer la surface. C'est le cas du satellite LANDSAT, qui couvre une bande de 185 km de large. Ce mode est qualifié de *balayage par miroir*.

1.6.4 Résolution

La résolution est une mesure de la capacité d'un système optique à distinguer des signaux proches, que ce soit spatialement ou spectralement. En télédétection, on distingue quatre types principaux de résolution :

1. Résolution radiométrique : Elle correspond au seuil de sensibilité du radiomètre, c'est-à-dire la plus faible intensité de rayonnement (réfléchi ou émis) que le capteur est capable de détecter dans une bande spectrale donnée. Elle détermine le nombre de niveaux de gris possibles dans une image. (Caloz, 1992)
2. Résolution spectrale : Il s'agit de la plus petite largeur de bande spectrale $\Delta\lambda$ dans laquelle le capteur peut mesurer une intensité suffisante. Pour les radiomètres des satellites d'observation terrestre, elle est généralement de l'ordre de 0,1 μm, tandis que pour les radiomètres à haute résolution spectrale, elle atteint environ 10 nm. (Caloz, 1992)
3. Résolution temporelle : Elle correspond à la fréquence avec laquelle un capteur peut revisiter exactement la même zone. Elle dépend de l'orbite du satellite. Par exemple, si un satellite passe tous les 5 jours au-dessus d'un même point, sa résolution temporelle est de 5 jours. Cette résolution est essentielle pour le suivi des phénomènes évolutifs dans le temps (études multi-dates).
4. Résolution spatiale : Elle définit la plus petite unité de surface au sol que le capteur peut détecter. Par exemple, pour le capteur ETM+ de Landsat, elle est de 30 mètres, ce qui signifie que chaque pixel de l'image couvre une surface de 30 m $\times$ 30 m au sol.[6].

1.7 Image Satellitaire

1.7.1 Introduction

L'observation de la Terre depuis l'espace constitue un objectif de longue date pour l'humanité. Aujourd'hui, les images satellitaires représentent une source précieuse d'informations détaillées et fiables sur l'occupation des sols et les phénomènes se déroulant à la surface terrestre.

1.7.2 Définition

Une image satellitaire correspond à la représentation des mesures d'énergie captées dans des bandes spectrales spécifiques (visible, proche infrarouge, infrarouge thermique, vapeur d'eau, hyperfréquence, etc.) pour chaque point observé. La résolution spatiale définit l'unité minimale de surface au sol mesurée par un capteur, chaque mesure étant associée à un pixel. Une image est composée de lignes, elles-mêmes formées de pixels, et chaque canal spectral produit une image distincte. À l'état brut, l'image ne contient pas d'informations géophysiques directement exploitables : elle se compose de valeurs radiométriques exprimées en octets, nécessitant divers prétraitements afin de les convertir en données interprétables.[7].

1.7.3 Traitement numérique de l'image satellite

1. Prétraitement

A - Corrections radiométriques : Ces corrections visent à réduire les perturbations induites par l'atmosphère et les capteurs. Deux images d'une même région acquises à des dates différentes peuvent présenter des valeurs numériques variées, en raison de l'élévation solaire (dépendant de la saison), des conditions atmosphériques ou des changements dans l'occupation du sol. Les corrections radiométriques permettent de compenser ces variations, notamment pour les analyses multi-temporelles. Les satellites d'observation passent à des heures solaires fixes, et les métadonnées enregistrent l'élévation solaire et l'azimut, facilitant l'harmonisation des images.

B - Corrections géométriques : Elles visent à éliminer les distorsions systématiques dues à la prise de vue (forme de la Terre, déplacement du satellite, technologie du capteur) et à permettre une superposition correcte sur une carte. Elles peuvent être réalisées à partir des paramètres orbitaux enregistrés ou par l'utilisation de lois de déformation basées sur des points de référence connus.

2. Traitement

Rehaussement de l'image : Ce procédé améliore la qualité visuelle de l'image pour faciliter l'interprétation humaine. Il met en valeur les contrastes et les détails, en tenant compte des capacités du système visuel humain, notamment la différenciation des niveaux de gris et la perception des couleurs.

Transformations d'images : Ces opérations consistent à manipuler plusieurs bandes spectrales simultanément afin de produire des images dérivées mettant en évidence des caractéristiques spécifiques. Cela inclut des opérations arithmétiques (addition, soustraction, multiplication, division), les rapports spectraux (ratios de bandes) et l'analyse en composantes principales (ACP), permettant une représentation optimisée de l'information multispectrale.[8].

1.8 Conclusion :

La télédétection représente aujourd'hui un outil incontournable pour l'acquisition d'informations précises sur la surface terrestre, sans contact direct. Ce chapitre a permis d'explorer ses principes fondamentaux, ses systèmes d'observation, ainsi que les traitements numériques appliqués aux images satellitaires. Ces éléments offrent un socle de connaissances indispensables pour aborder des problématiques avancées, notamment en lien avec l'analyse automatique des données issues de l'espace. En comprenant les propriétés physiques des images et les spécificités des capteurs, il devient possible d'adapter efficacement les méthodes de traitement aux besoins croissants en matière de surveillance environ-

nementale, de gestion des ressources ou encore de prévention des risques. Les chapitres suivants approfondiront ces enjeux à travers des techniques de segmentation d'images et des approches issues de l'intelligence artificielle.

Chapitre 2

La Segmentation des Images

2.1 Introduction

La segmentation d'image est une étape cruciale en traitement et analyse d'images, notamment en vision par ordinateur, qui vise à décomposer une image en régions homogènes et significatives. Avant de pouvoir exploiter efficacement des images numériques, il est indispensable de comprendre leur nature, leurs caractéristiques fondamentales et les différentes représentations possibles. Ce chapitre présente d'abord les notions essentielles liées à l'image numérique, ses formats et ses propriétés, qui conditionnent les méthodes de segmentation applicables. Puis, il introduit la segmentation d'image sous ses différentes formes — sémantique, d'instance, panoptique — en mettant en lumière leurs spécificités, leurs défis et leurs domaines d'application. Enfin, nous explorerons les approches classiques de segmentation basées sur les contours et les régions, qui, malgré leur ancienneté, constituent un socle conceptuel solide et une référence pour les techniques modernes. Cette étude permettra de poser les bases nécessaires à la compréhension et à la mise en œuvre des méthodes avancées présentées dans les chapitres suivants.

2.2 Notions fondamentales de l'image numérique

2.2.1 Définition d'une image

L'image est une représentation d'une personne ou d'un objet par la peinture, sculpture, le dessin, la photographie, le film...etc. C'est aussi un ensemble structuré d'informations qui, après l'affichage sur l'écran, ont une signification pour l'œil humain. Elle peut être décrite sous la forme d'une fonction (x, y) de brillance analogique continue, définie dans un domaine borné, tel que x et y sont les coordonnées spatiales d'un point de l'image et I est une fonction d'intensité lumineuse et de couleur. Sous cet aspect, l'image est inexploitable par la machine, ce qui nécessite sa numérisation[9].

2.2.2 Définition d'une image numérique

Une image numérique est définie comme une fonction bidimensionnelle, notée f(x,y)f(x,y), où xx et yy représentent les coordonnées spatiales, et l'amplitude de ff en tout point (x,y)(x,y) correspond à l'intensité ou au niveau de gris de l'image à cet endroit. Lorsque les coordonnées xx, yy et les valeurs d'amplitude de ff sont toutes des quantités finies et discrètes, on parle alors d'image numérique. Formellement, une image numérique peut être représentée comme un tableau 2D d'entiers, où chaque élément f(i,j)f(i,j) correspond à la valeur d'un pixel situé aux coordonnées (i,j)(i,j). Par exemple, pour une image en niveaux de gris sur 8 bits, les valeurs des pixels varient entre 0 et 255. Cette discrétisation permet

de traiter et de manipuler l'image à l'aide d'outils informatiques, comme le traitement d'image ou la vision par ordinateur.[10].

2.2.3 Types d'images

— **Image binaire** : Une image binaire est une image numérique où chaque pixel ne peut prendre que deux valeurs, généralement 0 (noir) et 1 (blanc), représentant respectivement l'absence ou la présence d'un objet ou d'une caractéristique. Ce type d'image est souvent utilisé pour simplifier l'analyse en ne conservant que les informations essentielles, comme les contours ou les régions d'intérêt. Les images binaires sont courantes en traitement d'images pour des tâches telles que la segmentation, la reconnaissance de formes ou la morphologie mathématique, où les opérations comme l'érosion, la dilatation ou l'extraction de squelettes peuvent être appliquées efficacement. Elles sont également utilisées dans des applications pratiques comme la lecture de codes-barres, la détection de caractères (OCR) ou l'analyse médicale

— **Image en niveaux de gris** : Une image en niveaux de gris attribue à chaque pixel une intensité lumineuse variant généralement entre 0 (noir) et 255 (blanc). Ce type d'image est courant en photographie monochrome et en imagerie médicale, où les nuances de gris permettent de distinguer des détails fins.

— **Image en couleurs (RGB, multispectral)** :Les images en couleurs utilisent plusieurs canaux pour représenter les teintes. Le modèle RGB (Rouge, Vert, Bleu) combine trois canaux pour reproduire une large gamme de couleurs. Les images multispectrales, quant à elles, capturent des données dans plusieurs longueurs d'onde (visible ou non), utiles en télédétection ou en imagerie scientifique. Ces images offrent une richesse d'informations pour l'analyse et l'interprétation visuelle.Ces types d'images servent de base à de nombreuses applications en traitement d'images, de la simple visualisation à l'analyse complexe. [11].

2.2.4 Propriétés caractéristiques d'une image

Une image numérique est définie par plusieurs propriétés fondamentales qui déterminent sa qualité et son apparence.

— **Pixel** :Un pixel est l'élément de base d'une image numérique ; il représente la plus petite unité d'information visuelle. Chaque pixel possède une valeur, souvent une intensité lumineuse (niveau de gris) ou une couleur (en RGB par exemple).

— **Résolution** :La résolution d'une image fait référence au nombre total de pixels qu'elle contient, souvent exprimée en largeur × hauteur (ex. : 1920×1080), ce qui influence directement le niveau de détail visible. dimensions :Les dimensions spatiales indiquent la taille de l'image en nombre de pixels, et elles sont essentielles pour déterminer la quantité d'information que l'image peut transmettre.

— **Contraste** :Le contraste désigne la différence d'intensité entre les zones claires et sombres d'une image, jouant un rôle majeur dans la perception visuelle des détails.
— **Luminance** :La luminance, quant à elle, correspond à la luminosité perçue d'un pixel, influencée par la source lumineuse et les caractéristiques de l'œil humain.

— **Bruit** : est une perturbation aléatoire des valeurs de pixels, causée par des facteurs

comme le capteur ou de faibles conditions d'éclairage ; il peut dégrader la qualité visuelle et rendre l'interprétation difficile. Des techniques de filtrage ou de restauration sont alors nécessaires pour atténuer son impact.

— **Histogramme** : L'histogramme d'une image représente la répartition statistique des niveaux de gris ou des intensités lumineuses dans l'image. Il permet d'analyser rapidement la dynamique de contraste, la luminosité globale, et la distribution des tons sombres ou clairs. Cet outil est essentiel pour des opérations de traitement comme l'égalisation d'histogramme, qui améliore la visibilité des détails dans des images trop sombres ou trop claires. En exploitant l'histogramme, on peut ajuster le rendu visuel pour mieux faire ressortir les informations pertinentes.

— **Contours** :Les contours correspondent aux transitions brusques d'intensité dans une image, marquant les limites entre objets ou zones homogènes. Leur détection est une étape cruciale dans l'analyse d'image, utilisée pour extraire la forme des objets et segmenter les scènes.

— **Régions** : quant à elles, sont des zones continues partageant des caractéristiques similaires (couleur, texture, intensité). La segmentation en régions permet d'isoler des objets ou parties spécifiques d'une image pour une analyse plus approfondie. Des méthodes comme la détection de contours (ex. : Canny) ou le regroupement de pixels homogènes sont utilisées à cet effet.[12].

2.3 Introduction à la segmentation d'image

2.3.1 Définition

La segmentation d'image est un processus fondamental en vision par ordinateur qui consiste à diviser une image en régions homogènes et significatives basées sur des caractéristiques telles que l'intensité, la texture ou la couleur. une bonne segmentation devrait produire des régions uniformes et homogènes, avec des intérieurs simples et des frontières précises et non irrégulières. Les techniques de segmentation peuvent être classées en plusieurs catégories, notamment le clustering guidé par l'espace de mesure, les méthodes de croissance de région, et les techniques de division et fusion. Ces méthodes visent à équilibrer les propriétés souhaitées, comme l'uniformité des régions et la précision des frontières, bien qu'elles soient souvent ad hoc en l'absence d'une théorie unifiée de la segmentation.[13].

2.3.2 Types de segmentation

— **1.Segmentation sémantique** : La segmentation sémantique attribue une étiquette de classe à chaque pixel d'une image, sans distinction entre les instances d'un même objet (ex. : tous les pixels "voiture" sont regroupés). Historiquement, des méthodes comme les champs aléatoires conditionnels (CRF) ou TextonBoost (Shotton et al., 2009) étaient utilisées, mais les réseaux entièrement convolutionnels (FCN, Long et al., 2015) et leurs améliorations (U-Net, PSPNet) dominent désormais, avec des architectures comme UPerNet (Xiao et al., 2018) intégrant des pyramides de caractéristiques pour une précision accrue.

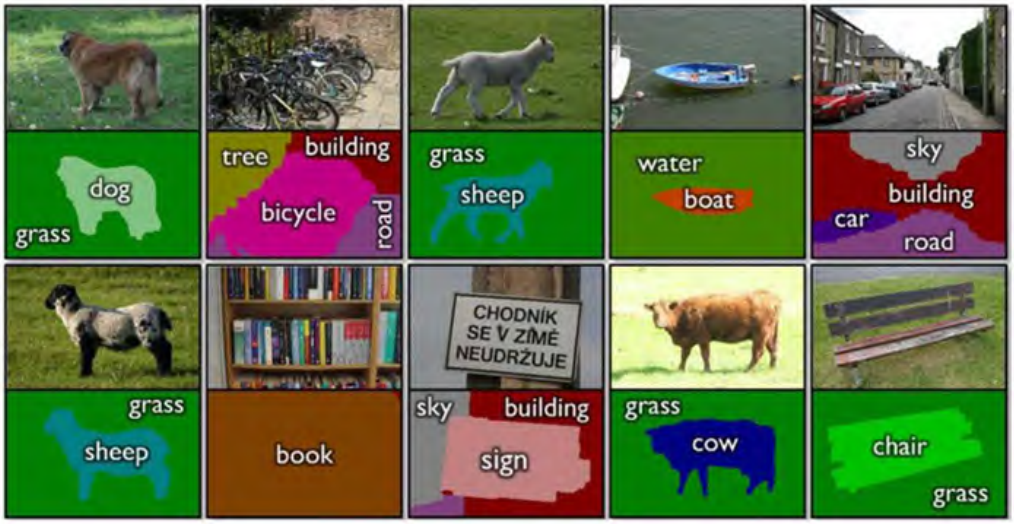


FIGURE 2.1 – Reconnaissance et segmentation simultanées à l'aide de TextonBoost

— **2.Segmentation d'instances** :Cette approche identifie et délimite chaque objet individuel, même s'ils appartiennent à la même classe (ex. : séparation des voitures individuelles). Des méthodes comme Mask R-CNN (He et al., 2017) combinent des propositions de région (RPN) avec un branchement supplémentaire pour prédire des masques binaires par instance, permettant également la détection de points clés (ex. : pose humaine). Les benchmarks comme COCO évaluent ces modèles via des métriques d'IoU (Intersection over Union).

FIGURE 2.2 – Détection des points clés des personnes et segmentation à l'aide de Mask

— **3.Segmentation panoptique** : Introduite par Kirillov et al. (2019), cette tâche unifie la segmentation sémantique ("choses" comme le ciel) et d'instance ("objets" comme les voitures). Elle utilise des métriques hybrides (PQ, Panoptic Quality) et des architectures comme Panoptic FPN (Kirillov et al., 2019) pour labelliser exhaustivement tous les pixels d'une image.

FIGURE 2.3 – Résultats de segmentation panoptique

2.3.3 Comparaison des approches

— **Précision** : La segmentation d'instance offre une localisation fine, tandis que la sémantique est plus rapide mais moins discriminante.
— **Complexité** : Les méthodes panoptiques combinent les deux, mais nécessitent des jeux de données annotés riches (ex. : COCO, Cityscapes).
— **Applications** : Le choix dépend du besoin : analyse de scènes (sémantique), suivi d'objets (instance), ou compréhension holistique (panoptique).

2.3.4 Domaines d'application

— **Médical** :Segmentation de tumeurs (Kamnitsas et al., 2016) ou d'organes via des réseaux 3D
— **Édition photo intelligente** :Complétion de scènes (Hays et Efros, 2007) ou "photo pop-up" 3D (Hoiem et al., 2005a).
— **Robotique/Conduite autonome** :Compréhension d'environnements urbains (Cityscapes).
— **Surveillance** : cEstimation de pose humaine (OpenPose, Cao et al., 2017) ou modélisation 3D (DensePose, Güler et al., 2018).[14].

2.4 Approches classiques de segmentation

2.4.1 Segmentation Basée sur les contours

— **Détection de bords** : La segmentation basée sur les contours repose sur la détection des discontinuités dans une image pour délimiter les objets. Parmi les méthodes les plus utilisées, on trouve les opérateurs de détection de contours tels que Sobel et Prewitt, qui calculent des gradients pour identifier les variations d'intensité. L'algorithme de Canny, plus sophistiqué, optimise la détection en minimisant le bruit tout en préservant la localisation précise des contours. Il combine un lissage gaussien, le calcul du gradient, une suppression des non-maxima et un seuillage avec hystérésis pour produire des contours fins et connectés. Ces méthodes sont essentielles pour extraire les structures géométriques des images, bien qu'elles puissent être sensibles au bruit et aux textures complexes.Canny, J. (1986). A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence, (6), 679-698.

— **Contours actifs** :Les contours actifs sont des méthodes de segmentation dynamiques qui ajustent automatiquement une courbe pour épouser les contours des objets dans une image. Les Snakes, introduits par Kass et al., minimisent une énergie combinant des termes internes (régularité de la courbe) et externes (attraction

vers les gradients de l'image). Bien que simples à implémenter, ils sont sensibles à l'initialisation et peinent à gérer les changements topologiques. Les Level Sets, basés sur une représentation implicite de la courbe, résolvent ces limitations en permettant des divisions ou fusions naturelles de contours. Ils utilisent une fonction de niveau évoluant selon une vitesse dépendant de la courbure et des gradients de l'image, offrant une meilleure robustesse et précision, notamment pour des structures complexes comme en imagerie médicale. L'étude comparée montre que les Level Sets surpassent les Snakes en reproductibilité et précision, avec moins d'intervention manuelle.[15].

2.4.2 Segmentation Basée sur les régions

- **Croissance de région (Region Growing)** :La croissance de région est une méthode de segmentation qui agrège progressivement des pixels ou voxels similaires à partir d'un point de départ (seed) selon un critère d'homogénéité. une approche automatique et basée sur la texture est proposée pour la segmentation du foie en tomodensitométrie (CT). Les caractéristiques de texture, extraites via des matrices de co-occurrence 3D (GLCM), permettent de différencier le foie des tissus adjacents malgré des niveaux de gris similaires.[16].
- **Division et fusion (Split and Merge)** : la méthode de division et fusion (Split and Merge) une technique de segmentation d'images basée sur les régions. Cette approche consiste à diviser une image en sous-régions homogènes selon des critères prédéfinis, puis à fusionner les régions adjacentes similaires pour former des segments plus grands. L'avantage principal de cette méthode réside dans sa capacité à prendre en compte les informations spatiales et à s'adapter aux variations locales de l'image. Cependant, elle peut être sensible au choix des critères de similarité et des seuils, ce qui peut conduire à une sur-segmentation ou à une sous-segmentation si les paramètres ne sont pas optimisés. Dans le contexte de l'analyse quantitative des fibres de collagène, cette technique pourrait être utilisée pour identifier et isoler des structures fibrillaires spécifiques.[17].

2.4.3 Méthodes Par seuillage

- **Seuillage global (Otsu)** : Le seuillage global d'Otsu est une méthode automatique de binarisation d'image largement utilisée dans le traitement d'images. Proposée par Nobuyuki Otsu en 1979, cette technique consiste à déterminer un seuil optimal à partir de l'histogramme des niveaux de gris de l'image, de manière à séparer les pixels en deux classes distinctes : le fond et les objets. Le seuil est choisi de façon à maximiser la variance inter-classes, c'est-à-dire la séparation entre les distributions de niveaux de gris des deux groupes, ce qui équivaut à minimiser la variance intra-classe. L'un des grands avantages de la méthode d'Otsu est qu'elle est non paramétrique et non supervisée, ne nécessitant ni hypothèse préalable sur les distributions de pixels ni intervention humaine. Elle est particulièrement efficace lorsque l'histogramme de l'image présente deux pics distincts (bimodalité), mais peut être limitée en présence de bruit ou d'images mal contrastées.[18].
- **Seuillage adaptatif** : e seuillage adaptatif est une technique avancée de traitement d'image qui permet de segmenter une image en deux classes (par exemple, "sombre" et "clair") en tenant compte des variations locales d'éclairage. Contrairement au seuillage global qui utilise une valeur de seuil fixe pour toute l'image, le seuillage adaptatif calcule un seuil spécifique pour chaque pixel en fonction de son voisinage. Cette approche est particulièrement utile pour les images présentant

des variations d'illumination spatiales ou temporelles, comme dans les flux vidéo en temps réel.[19].

2.4.4 Classification des pixels

K-means :Le K-means est une méthode couramment utilisée pour la classification des pixels dans des applications comme la segmentation d'images. Son principe repose sur la partition des pixels en un nombre prédéfini de clusters (K) en minimisant la variance intra-cluster. Chaque pixel est représenté par ses caractéristiques, telles que l'intensité lumineuse ou les valeurs des canaux de couleur (RVB), et est assigné au cluster dont le centre (moyenne des pixels du cluster) est le plus proche. Bien que simple et efficace, le K-means présente des limites, notamment sa sensibilité aux initialisations aléatoires des centres et sa tendance à produire des clusters de forme sphérique, ce qui peut ne pas convenir à des structures complexes dans les images. Des améliorations, comme l'utilisation de distances adaptées ou l'intégration de contraintes spatiales, ont été proposées pour surmonter ces limitations.[20].

2.4.5 Classification supervisée et non supervisée

— **La classification supervisée** : l'utilisateur définit des classes thématiques en sélectionnant des échantillons de référence (zones d'entraînement) pour chaque catégorie d'intérêt. Des algorithmes comme le maximum de vraisemblance, les k-plus proches voisins (kNN), ou les machines à vecteurs de support (SVM) sont ensuite utilisés pour attribuer chaque pixel à l'une des classes prédéfinies. Cette méthode est efficace lorsque les classes sont bien connues et que des données de référence sont disponibles.

— **La classification Non supervisée** :ne nécessite pas d'échantillons préalables. Elle repose sur des algorithmes de clustering (comme les méthodes des k-moyennes ou la classification hiérarchique) pour regrouper les pixels en classes spectrales basées sur leur similarité statistique. Ces classes sont ensuite interprétées a posteriori par l'utilisateur. Cette approche est utile pour explorer des données inconnues ou pour identifier des structures naturelles dans l'image.

Les deux méthodes présentent des avantages et des limites : la classification supervisée offre une précision thématique mais dépend de la qualité des échantillons, tandis que la classification non supervisée est plus flexible mais peut produire des résultats moins précis sans validation terrain. Des approches hybrides combinant les deux sont souvent employées pour optimiser l'analyse.[21].

Conclusion

La segmentation d'image est un processus fondamental qui permet d'extraire des informations structurées et exploitables à partir d'images complexes. Ce chapitre a permis d'établir un cadre théorique en définissant les concepts clés liés aux images numériques, ainsi qu'en présentant les différents types de segmentation et leurs spécificités. Les approches classiques exposées, bien qu'ayant connu une certaine limite face à la complexité croissante des données, restent essentielles pour comprendre les mécanismes sous-jacents à la détection et la séparation des objets dans une image. Elles servent aussi de point de départ pour l'évolution vers des méthodes plus sophistiquées basées sur l'apprentissage profond, qui seront abordées dans les chapitres suivants. Ainsi, cette introduction à la segmentation d'image fournit les fondations indispensables pour appréhender les techniques contemporaines et leurs applications multiples dans divers domaines scientifiques et industriels.

Chapitre 3

Deep Learning

Introduction

Depuis les débuts de l'intelligence artificielle, l'idée de permettre aux machines d'apprendre à partir de l'expérience humaine a suscité un vif intérêt. Le *Deep Learning*, ou apprentissage profond, incarne cette vision en utilisant des réseaux de neurones artificiels profonds pour apprendre automatiquement des représentations complexes à partir de données brutes.

Ce chapitre vise à explorer les fondements, les architectures et les méthodes d'entraînement du *deep learning*, en mettant en évidence son impact sur des domaines variés tels que la vision par ordinateur, le traitement du langage naturel, ou encore les systèmes de recommandation.

De ses origines biologiques à ses applications les plus modernes, ce chapitre propose une immersion progressive dans l'univers des réseaux neuronaux profonds.

3.1 Historique

Certains des premiers RN (McCulloch et Pitts, 1943) n'apprenaient pas du tout. Hebb (1949) a publié des idées sur l'apprentissage non supervisé. Les décennies suivantes ont été marquées par des NAS peu profondes non supervisées et des NAS supervisées (p. ex., Rosenblatt, 1958). Les premières NNs supervisées étaient essentiellement des variantes de régresseurs linéaires datant de deux siècles (Gauss, Legendre).

Les réseaux d'apprentissage profond ont vu le jour dans les années 1960, lorsque Ivakhnenko et Lapa (1965) ont publié le premier algorithme général d'apprentissage pour des perceptrons multicouches à avance profonde supervisés. Leurs unités ont des fonctions d'activation polynomiales combinant additions et multiplications dans les polynômes de Kolmogorov-Gabor. Ivakhnenko (1971) a déjà décrit un réseau profond avec 8 couches formées par la "méthode de traitement des données en groupe", encore populaire dans le nouveau millénaire. Un ensemble d'entraînement de vecteurs d'entrée avec des vecteurs de sortie cibles correspondants, les couches sont progressivement cultivées et entraînées par analyse de régression, puis taillées à l'aide d'un ensemble de validation séparé, où la régularisation est utilisée pour éliminer les unités superflues. Le nombre de couches et d'unités par couche peut être appris en fonction du problème.

Comme les NNs profonds ultérieurs, les réseaux d'Ivakhnenko ont appris à créer des représentations hiérarchiques, distribuées et internes des données entrantes. De nombreuses méthodes non neuronales ultérieures d'intelligence artificielle et d'apprentissage machine apprennent également des représentations de données hiérarchiques de plus en plus abstraites. Par exemple, les méthodes de reconnaissance syntaxique des motifs (Fu, 1977) telles

que l'induction grammaticale permettent de découvrir des hiérarchies de règles formelles pour modéliser les observations.[22].

3.2 Introduction au deep learning :

Depuis l'Antiquité, les humains rêvent de créer des machines capables de penser. Avec l'arrivée des ordinateurs programmables, l'idée que ces machines puissent devenir intelligentes a émergé bien avant leur concrétisation. L'intelligence artificielle est aujourd'hui un domaine en plein essor, utilisé pour automatiser des tâches, comprendre le langage ou les images, diagnostiquer des maladies et appuyer la recherche scientifique. Les premiers succès de l'IA portaient sur des problèmes formels et bien définis, faciles à décrire mathématiquement mais difficiles pour les humains. À l'inverse, les tâches intuitives simples pour les humains (comme reconnaître un visage ou une voix) se sont révélées bien plus complexes à formaliser pour les machines. Pour résoudre ces problèmes intuitifs, une solution a émergé : permettre aux ordinateurs d'apprendre à partir de l'expérience. C'est l'approche de l'apprentissage profond, où les machines construisent une hiérarchie de concepts de plus en plus complexes à partir de données brutes, évitant ainsi aux programmeurs de devoir définir toutes les règles à l'avance. Les approches traditionnelles basées sur des bases de connaissances formelles, comme le projet Cyc, ont montré leurs limites : elles peinent à capturer la complexité et l'ambiguïté du monde réel. À l'inverse, l'apprentissage automatique (machine learning) permet aux machines d'extraire des modèles à partir des données. Cependant, ces méthodes dépendent fortement de la représentation des données : de simples algorithmes (comme la régression logistique ou naïve Bayes) peuvent être efficaces si les bonnes caractéristiques (features) sont fournies. Mais pour des données complexes comme des images, il est difficile de savoir à l'avance quelles caractéristiques extraire. C'est là que l'apprentissage profond excelle : il apprend automatiquement les représentations pertinentes directement à partir des données brutes (comme les pixels d'une image), sans intervention humaine pour définir les bonnes caractéristiques.[23].

3.3 Domaines d'application

- **Vision par ordinateur** :L'objectif de la vision par ordinateur est d'apprendre aux machines à voir et à percevoir les images de la même manière que les humains.
- **T**raitement de la langue naturelle : la traduction linguistique en temps réel et de haute précision, la reconnaissance vocale et des assistants virtuels
- **Systèmes de recommandation** : Ces systèmes basés sur l'apprentissage profond alimentent l'internet avec des flux de contenu tels que facebook, de la musique comme spotify, de la vidéo comme youtube et netflix, ainsi que de la publicité en ligne ciblée et des recommandations d'achat (Amazon).[24].
- **Images satellites** : à l'analyse d'images satellites, notamment pour la classification de l'occupation du sol, la détection d'objets, le suivi du changement climatique et la segmentation sémantique.

3.4 Fondamentaux du Deep Learning

3.4.1 Définition

L'apprentissage profond permet aux modèles informatiques composés de plusieurs couches de traitement d'apprendre des représentations de données avec plusieurs niveaux d'abstraction. Ces méthodes ont considérablement amélioré l'état de la technique dans la reconnaissance vocale, la reconnaissance d'objets visuels, la détection d'objets et de

nombreux autres domaines tels que la découverte de médicaments et la génomique. L'apprentissage profond découvre une structure complexe dans de grands ensembles de données en utilisant l'algorithme de rétropropagation pour indiquer comment une machine doit modifier ses paramètres internes qui sont utilisés pour calculer la représentation dans chaque couche à partir de la représentation dans la couche précédente. Les réseaux convolutifs profonds ont permis des percées dans le traitement des images, de la vidéo, de la parole et de l'audio, tandis que les réseaux récurrents ont mis en lumière des données séquentielles telles que le texte et la parole.[25].

3.4.2 Neurones biologiques vs artificiels

Aneuron dans le cerveau humain est une cellule biologique qui traite l'information. Le neurone biologique est représenté en haut de la fig. 2. Il existe trois éléments de base d'un neurone biologique. Les dendrites reçoivent des signaux xi, i = 1,2, ,n comme une entrée d'autres neurones. Un corps cellulaire (nu cleus) contrôle l'activité des neurones, et les axones transmettent des signaux aux neurones voisins. La longueur de l'axone peut être plusieurs fois ou même des dizaines de milliers de fois plus longue que le corps cellulaire. Les axones sont divisés en plusieurs branches près de leur extrémité, qui sont connectées aux dendrites d'autres neurones. Il existe des millions de neurones massivement connectés (environ 1011), ce qui est approximativement égal au nombre d'étoiles dans la Voie lactée (Brunak et Lautrup, 1990). Chaque neurone est connecté à des milliers de neurones voisins, et ces neurones sont organisés en couches successives dans le cortex cérébral. Un tel réseau de neurones massivement parallèle communique par un très court train d'impulsions, en millisecondes, et a une capacité qui inclut le parallèle [26].

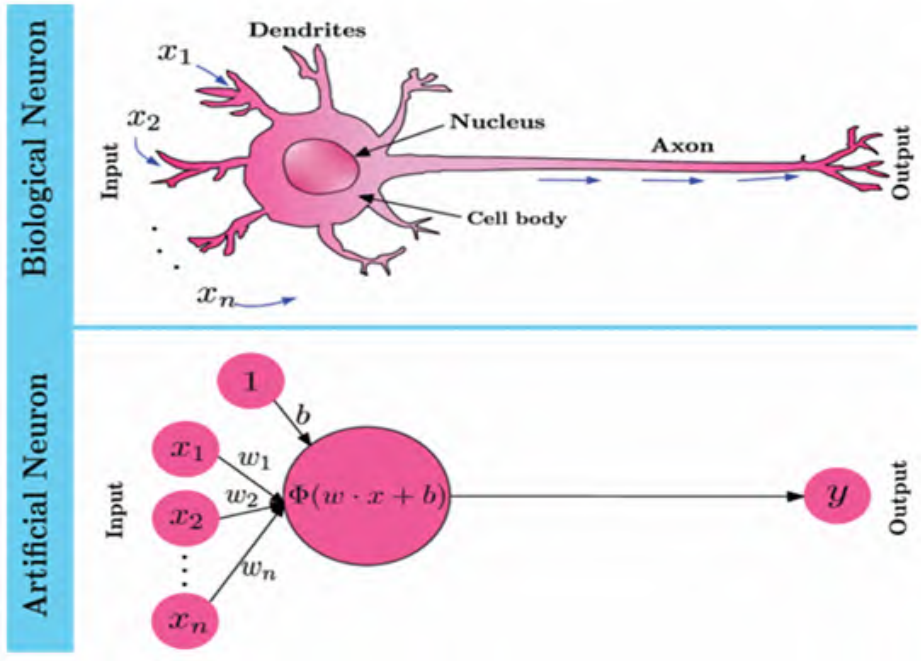


FIGURE 3.1 – Biological (top) vs. artificial (bottom) neurons

3.4.3 Perceptron et limites

Le perceptron est un modèle (et une machine) créé à la fin des années 1950 par le psychologue américain Frank Rosenblatt de l'Université de Cornell. Il s'appuie sur la com-

préhension contemporaine des mécanismes du fonctionnement d'un neurone isolé (figure 3-a) et leur modélisation mathématique proposée quelques années plus tôt par McCulloch et Pitts (McCulloch 1943). Comme illustré sur la figure 3-b, le neurone « mathématique » calcule la somme des signaux amonts (X1, X2, X3...), chacun étant affublé d'un poids spécifique relatif à son importance (W1, W2, W3...). Si la somme de ces activités, une fois pondérées, est inférieure à un certain seuil prédéfini, le neurone reste inactif. En revanche, si cette somme dépasse ce seuil, le neurone s'active (comme illustré sur la figure 3-b). A ce neurone formel, Rosenblatt ajoute une procédure d'apprentissage à partir de l'estimation de l'erreur commise lors de l'essai ce qui permet l'ajustement des poids synaptiques par correction d'erreur (Rennard 2013).

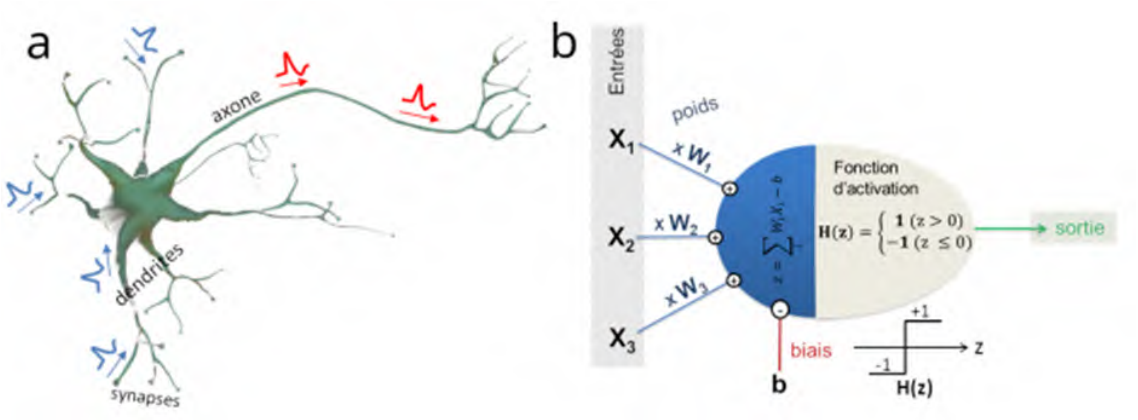


FIGURE 3.2 – perceptron

illustre deux représentations : (a) un neurone artificiel isolé, et (b) un perceptron simple doté de trois entrées X_1, X_2 et X_3, associées à trois poids synaptiques W_1, W_2, W_3 ainsi qu'un biais b. Le perceptron constitue l'un des premiers modèles d'apprentissage supervisé, dans lequel les paramètres sont ajustés afin de minimiser les erreurs sur un ensemble d'apprentissage. Une fois les poids et le biais optimisés, le modèle devient capable d'effectuer des tâches de classification. Cependant, malgré les promesses initiales de son inventeur Frank Rosenblatt, cette machine apprenante s'est rapidement heurtée à des limites majeures : elle n'est capable de résoudre que des problèmes linéairement séparables, ce qui restreint considérablement son champ d'application. Ce n'est qu'avec l'émergence des réseaux de neurones multicouches et l'introduction de l'algorithme de descente de gradient que ces limitations ont pu être surmontées, ouvrant ainsi la voie à des architectures plus complexes et performantes [27].

3.4.4 Réseaux de neurones Multicouche (MLP)

Un perceptron multicouches, appelé *Multi-Layered Perceptron* (MLP), est constitué d'unités de calcul élémentaires appelées **neurones**.

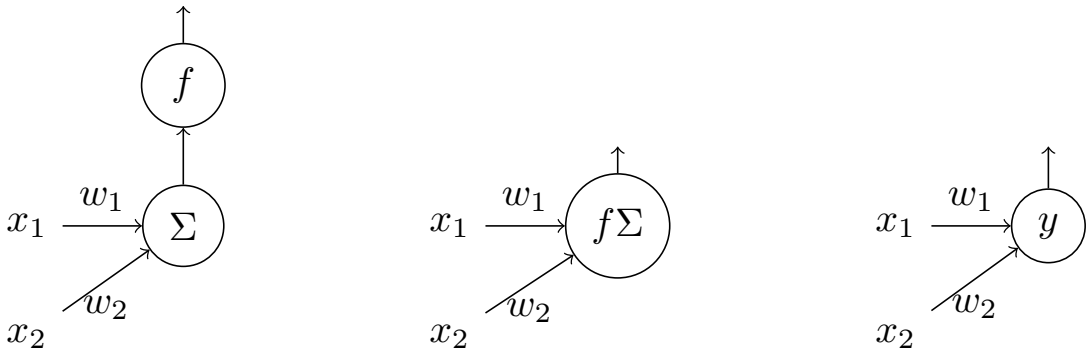
Un neurone possède des entrées, qui sont des variables à valeurs réelles, notées $x_1, x_2, \ldots, x_n$, et une sortie, notée y. Chaque entrée est associée à un poids, noté w_i.

Le calcul effectué par un neurone consiste à multiplier la valeur de chaque entrée x_i par son poids w_i et d'en faire la somme. Le résultat de cette somme est additionné à une constante b, appelée le *biais*.

Ce résultat constitue lui-même l'entrée d'une fonction non linéaire f, appelée *fonction d'activation*. Le résultat final du calcul constitue la sortie du neurone, notée $\hat{y}$, et appelée *activation du neurone*.

$$\hat{y} = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{3.1}$$

Représentation graphique



— **L**es neurones sont parfois représentés graphiquement.
— **L**es neurones sont parfois représentés graphiquement.
— **L**a représentation de gauche détaille les différentes étapes du calcul.
— **C**elle du milieu condense la somme et la fonction d'activation en un seul nœud.
— **I**l est souvent pratique de définir une variable qui prend pour valeur l'activation d'un neurone, comme dans la figure de droite.

Interconnexions

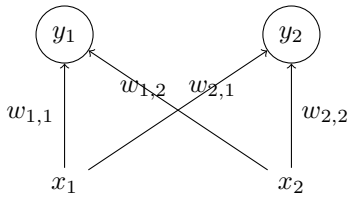
Les neurones peuvent être interconnectés pour constituer un réseau de neurones (RN).

Deux modes de connexion sont possibles :

neurones peuvent partager des entrées.

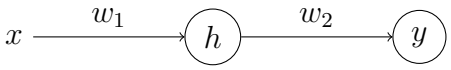
la sortie d'un neurone peut constituer l'entrée d'un autre neurone.

Partage d'entrée :



$$y_1 = f(w_{1,1}x_1 + w_{2,1}x_2 + b_1)$$
$$y_2 = f(w_{1,2}x_1 + w_{2,2}x_2 + b_2)$$

Composition :

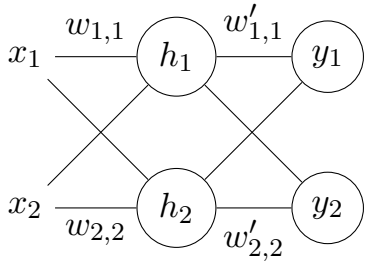


La sortie d'un neurone constitue l'entrée d'un autre.

$$y = f_2(w_2 \times h + b_2)$$
$$h = f_1(w_1 x + b_1)$$

La notion de couche

Dans un réseau de neurones (RN), les neurones sont généralement organisés en couches. Les sorties des neurones d'une couche constituent les entrées des neurones de la couche suivante.



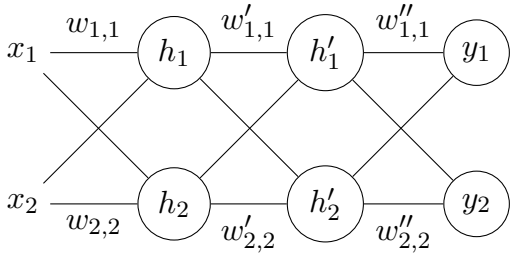
Lorsque chaque neurone d'une couche est connecté à tous les neurones de la couche suivante, le réseau est dit **entièrement connecté**.

Les calculs réalisés par le réseau sont les suivants :

— x_1 et x_2 constituent la **couche d'entrée** (ce ne sont pas des neurones).
— y_1 et y_2 constituent la **couche de sortie**.
— h_1 et h_2 constituent une **première couche cachée**.

Le nombre de couche constitue la profondeur du réseau.

Fonctions d'activation :



Les fonctions d'activation jouent un rôle crucial dans les réseaux de neurones artificiels (RNA) en introduisant des non-linéarités, ce qui permet aux réseaux d'apprendre des modèles complexes. Elles déterminent si un neurone doit être activé ou non, influençant ainsi la propagation des signaux à travers le réseau. **Caractéristiques Principales**

1. **Non-linéarité** : Essentielle pour capturer des relations complexes dans les données.

2. **Efficacité computationnelle** : Doivent être simples à calculer pour optimiser les performances.
3. **Problèmes de gradients** : Les fonctions comme la sigmoïde et la tangente hyperbolique peuvent causer des gradients faibles (*vanishing gradient*), tandis que ReLU et ses variantes atténuent ce problème.
4. **Plage de valeurs** : Une plage finie (bornée) améliore la stabilité pendant l'entraînement.
5. **Différentiabilité** : Nécessaire pour la rétropropagation, sauf pour certaines fonctions comme ReLU qui sont différentiables presque partout[28].

Types de Fonctions d'Activation

Classiques (Fixes)

— **Sigmoïde** : Plage $[0, 1]$, utile pour les probabilités mais souffre du problème de *vanishing gradient*.

$$\Phi(x) = \frac{1}{1 + e^{-x}}.$$

Tangente hyperbolique (tanh) : Plage $[-1, 1]$, centrée autour de zéro.

$$\Phi(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}.$$

ReLU (Rectified Linear Unit) : Simple et efficace, mais peut causer des « neurones morts » (*dying ReLU*).

$$\Phi(x) = \begin{cases} 0 & \text{si } x \leq 0, \\ x & \text{si } x > 0. \end{cases}$$

Variantes de ReLU : Leaky ReLU, PReLU, ELU, etc., améliorent ReLU en gérant mieux les valeurs négatives.

Fonctions oscillatoires : Comme $\sin(x)$, utiles pour capturer des motifs périodiques.

$$\mathbb{E}(xm) = x\mathbb{E} = xP(X \leq x).$$

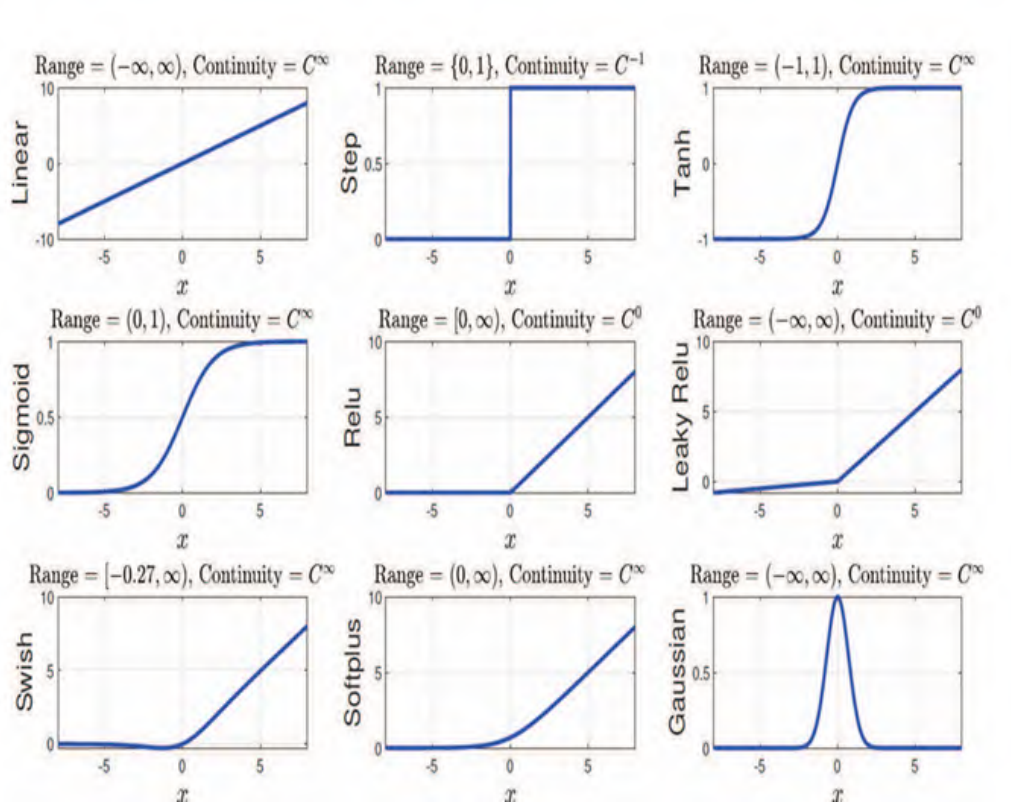


FIGURE 3.3 – Fonctions d'activation classiques (fixes) avec leur gamme et leur continuité

- **Adaptatives :**
  - **Paramétriques :** Introduisent des paramètres apprenables (ex. Swish, Mish).
  - **Ensemble :** Combinaison de plusieurs fonctions pour améliorer les performances.
  - **Stochastiques/Probabilistes :** Ajoutent du bruit pour une meilleure exploration.
  - **Fractionnaires :** Basées sur le calcul fractionnaire pour plus de flexibilité [29].

3.5 Architectures des Réseaux Neuronaux

3.5.1 Réseaux de neurones convolutifs (CNN)

Principe et motivation : Les réseaux de neurones convolutifs (CNN) sont une architecture spécialisée de réseaux de neurones conçue pour traiter des données structurées en grille, comme les images (2D) ou les séries temporelles (1D). Leur motivation principal repose sur trois idées clés : :

- **1. Interactions parsimoniousness (sparse interactions))** : Contrairement aux réseaux entièrement connectés, où chaque neurone est lié à tous les neurones de la couche précédente, les CNN utilisent des noyaux (filtres) de petite taille qui ne se connectent qu'à une région locale de l'input. Cela réduit le nombre de paramètres et améliore l'efficacité statistique.
- **2. Partage des paramètres (parameter sharing)** : Un même filtre est appliqué sur toute l'image, ce qui permet de détecter une caractéristique (comme une bordure) indépendamment de sa position.
- **3. Invariance translationnelle (equivariance)** : Grâce à la convolution, le

réseau devient insensible aux translations locales (par exemple, un motif détecté reste reconnu même s'il est légèrement déplacé)..

Ces propriétés font des CNN des modèles particulièrement adaptés à la vision par ordinateur, où les images présentent une structure spatiale hiérarchique (des pixels aux contours, puis aux formes complexes).

Fonctionnement d'un CNN : filtre, padding, stride

Un CNN applique des opérations de convolution successives :

1. **Filtres (kernels)** :
   — Un filtre est une petite matrice de poids (par exemple 3×3 ou 5×5) qui balaie l'image en multipliant ses valeurs par les pixels locaux..
   — Chaque filtre extrait une caractéristique spécifique (bords, textures, motifs).
   — Plusieurs filtres sont utilisés en parallèle pour capturer différentes features.

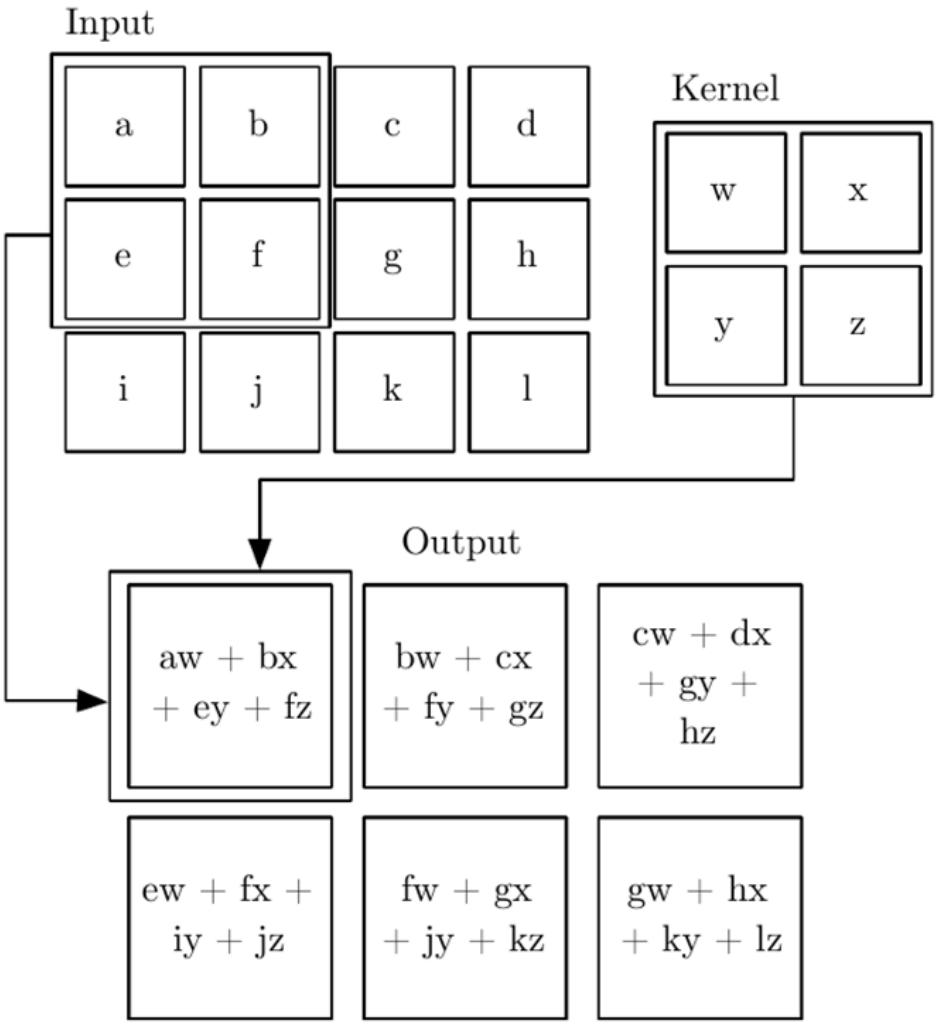


FIGURE 3.4 – Exemple de convolution 2-D sans basculement du noyau.

2. **Padding** :
   Pour éviter la réduction de la taille de l'image après convolution, on ajoute souvent des zéros autour de l'input (zero-padding

Types courants :
Valid : Pas de padding (la sortie est plus petite).
Same : Padding pour conserver la taille d'origine.

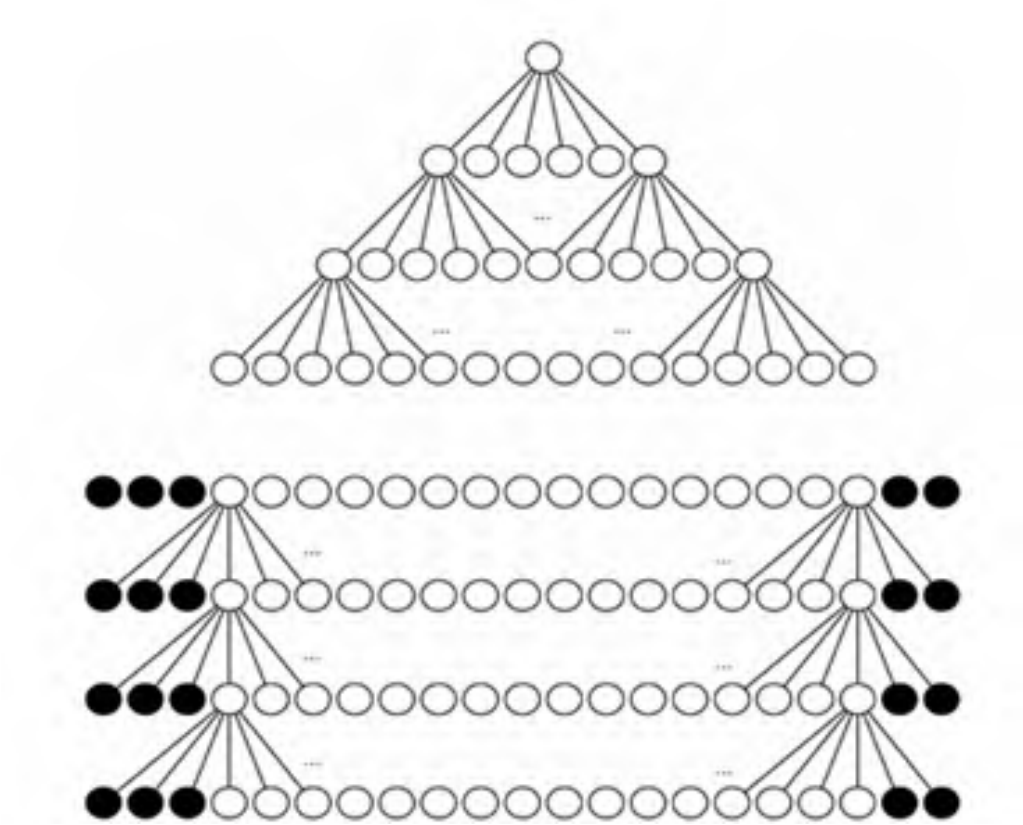


FIGURE 3.5 – Effet du zero-padding sur la taille des réseaux convolutifs.

~~3.~~ **Stride** :
- Do Le stride définit le pas de déplacement du filtre (par défaut 1).
- Un stride > 1 (ex : 2) réduit la résolution spatiale, ce qui diminue la complexité calculatoire.

3.5.2 Pooling, Flattening, Classification

1. **Pooling (sous-échantillonnage)** :
   - Réduit la dimension spatiale tout en gardant les infos importantes.
   - Types : **Max pooling**, **Average pooling**.
   - Avantages :
     Invariance aux petites translations.
     Réduction des calculs et du risque de surapprentissage.

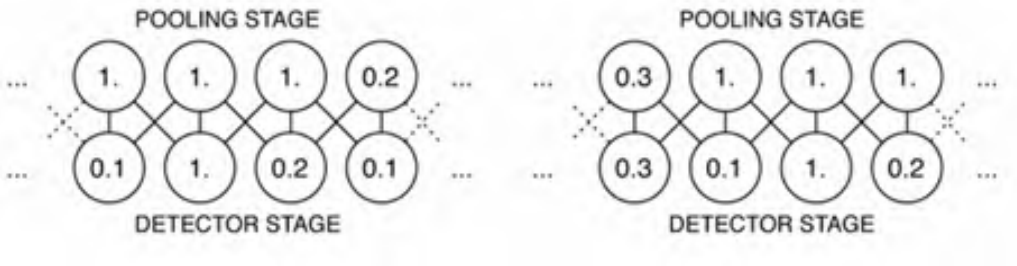


FIGURE 3.6 – Max pooling introduit une invariance.

2. **Flattening(mise à plat)** :
   — o Après plusieurs couches de convolution/pooling, les feature maps 3D (hauteur × largeur × canaux) sont transformées en un vecteur 1D.
   — o Permet de connecter les features à un réseau fully connected pour la classification.
3. **Classification (couches denses)** :
   — Le vecteur aplati est traité par des couches entièrement connectées (Dense).
   — La dernière couche utilise une activation softmax (pour multi-classes) ou sigmoid (binaire) pour prédire la classe.

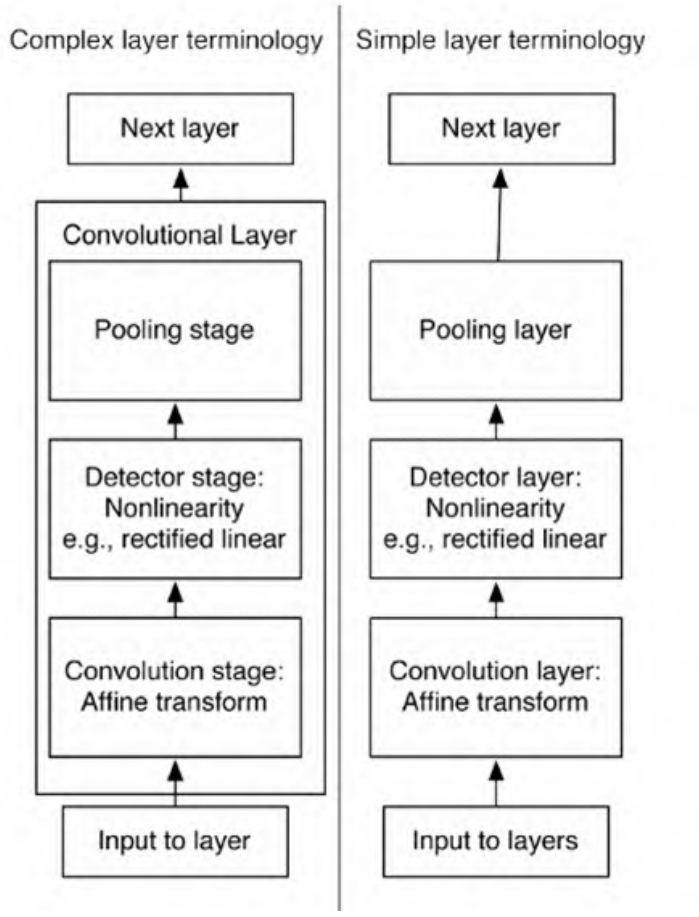


FIGURE 3.7 – Composants typiques d'un couche CNN.

Exemple d'architecture :

Input → Conv → ReLU → Pooling → Conv → ReLU → Pooling → Flatten → Dense → Softmax [30].

3.5.3 Réseaux de Neurones Récurrents (RNN, LSTM, GRU)

1 Réseaux de Neurones Récurrents (RNN) :

Principe : Les RNN sont conçus pour traiter des séquences de données (texte, séries temporelles, etc.) en maintenant une mémoire interne (état caché) qui capture l'information des entrées passées.

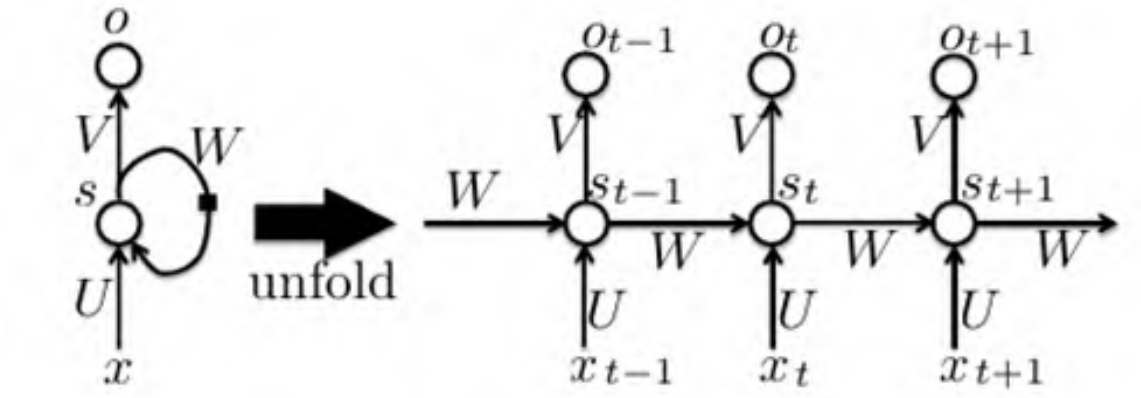


FIGURE 3.8 – Architecture RNN

La figure 3.8 illustre Gauche : circuit d'un réseau récurrent ordinaire avec une récurrence cachée-à-cachée, représenté comme un circuit, avec des matrices de poids U, V et W pour les trois types de connexions différents (entrée-vers-caché, caché-vers-sortie et caché-vers-caché, respectivement). Chaque cercle indique un vecteur complet d'activations.Droite : le même réseau vu comme un graphe de flux déroulé dans le temps, où chaque nœud est maintenant associé à un instant temporel particulier.

- **Avantages :**
  - — Modélisation de dépendances temporelles courtes.
  - — Partage des paramètres (poids WhWh et WxWx fixes pour tous les pas de temps).
- **Limitations :**
  - — Problème de gradients disparaissants/explosifs : Difficulté à capturer des dépendances à long terme (gradients mal propagés sur de nombreux pas de temps).
  - — Mémoire à court terme : L'état caché a du mal à retenir des informations ancienne

2 Long Short-Term Memory (LSTM) :

Principe : Les LSTM améliorent les RNN avec une mémoire à long terme via une architecture à portes (gates) qui contrôlent le flux d'information..

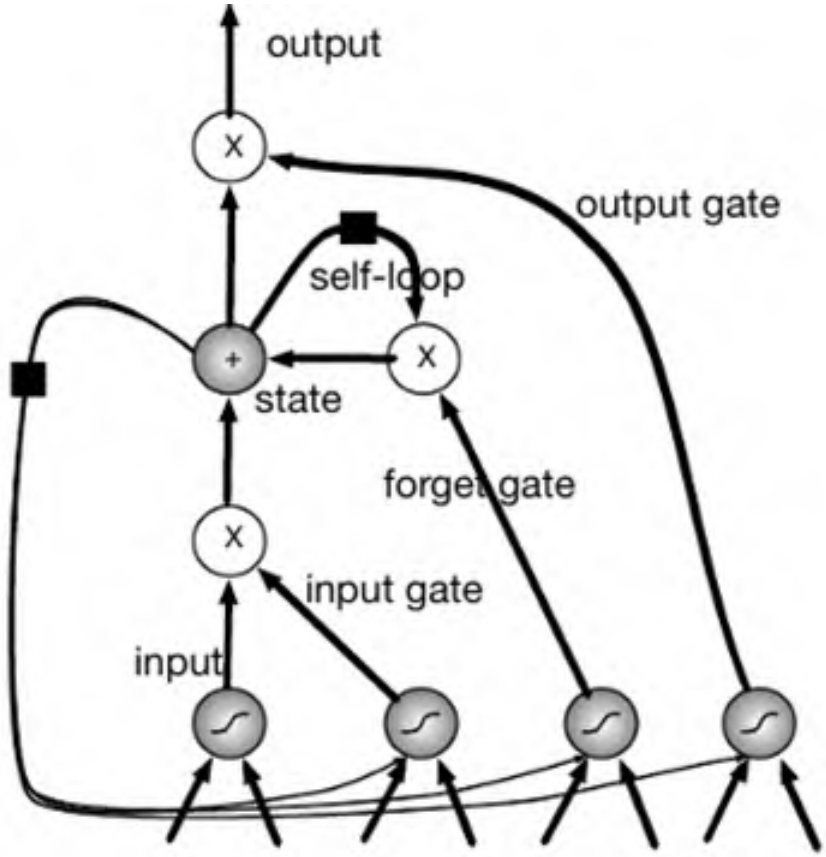


FIGURE 3.9 – Schéma fonctionnel d'une "cellule" de réseau récurrent LSTM.

La figure 3.9 illustre le fonctionnement interne d'une cellule mémoire dans un réseau de neurones récurrent de type LSTM (Long Short-Term Memory). Contrairement aux architectures récurrentes classiques, les cellules LSTM sont connectées de manière récurrente entre elles, remplaçant les unités cachées standards. Une caractéristique en entrée est d'abord traitée par une unité de neurone classique, puis elle est conditionnellement intégrée à l'état interne de la cellule si la porte d'entrée (sigmoïde) le permet. L'unité d'état est dotée d'une boucle de rétroaction linéaire (auto-connexion), dont la persistance est contrôlée par une porte d'oubli. La sortie de la cellule peut quant à elle être modulée, voire bloquée, par une porte de sortie. Toutes les portes utilisent une fonction d'activation sigmoïde, tandis que l'unité d'entrée peut appliquer n'importe quelle fonction d'écrasement, comme la tangente hyperbolique (`tanh`) ou la sigmoïde. Par ailleurs, l'unité d'état peut également fournir une entrée supplémentaire aux différentes portes. Le carré noir représenté dans le schéma correspond à un délai d'une unité de temps, reflétant la dynamique temporelle propre aux réseaux LSTM.

- **Avantages :**
  — Capture des dépendances à très long terme (ex : phrases de 100 mots).
  — Résiste aux gradients disparaissants grâce à la cellule mémoire.).
- **Applications :**
  — Traduction automatique, reconnaissance vocale, génération de texte Traduction automatique, reconnaissance vocale, génération de texte .

3 Gated Recurrent Units (GRU) :

Principe : Version simplifiée des LSTM avec deux portes (pas de cellule mémoire séparée).

- **Avantages :**
  — Moins de paramètres que les LSTM (calculs plus rapides).
  — Performances comparables pour de nombreuses tâches.
- **Cas d'usage :** Modèles légers pour des séquences moyennement longues (ex : analyse de sentiments).[31].

3.5.4 Autoencodeurs (AE et VariationalAE) :

Autoencodeurs (AE) : Les autoencodeurs (AE) sont des architectures neuronales non supervisées utilisées pour apprendre des représentations compactes et significatives des données. Ils se composent de deux parties principales :

— **Encodeur :** Une fonction ff qui transforme l'entrée xx en une représentation latente $f(x) \rightarrow h$
— **Décodeur** Une fonction gg qui reconstruit l'entrée à partir du code latent $g(h) \rightarrow \hat{x}$

L'objectif est de minimiser l'erreur de reconstruction L(x,x), souvent mesurée par l'erreur quadratique ou l'entropie croisée. Les AE sont utilisés pour la réduction de dimension, le débruitage (Denoising AE), ou comme pré-entraînement pour des tâches supervisées.

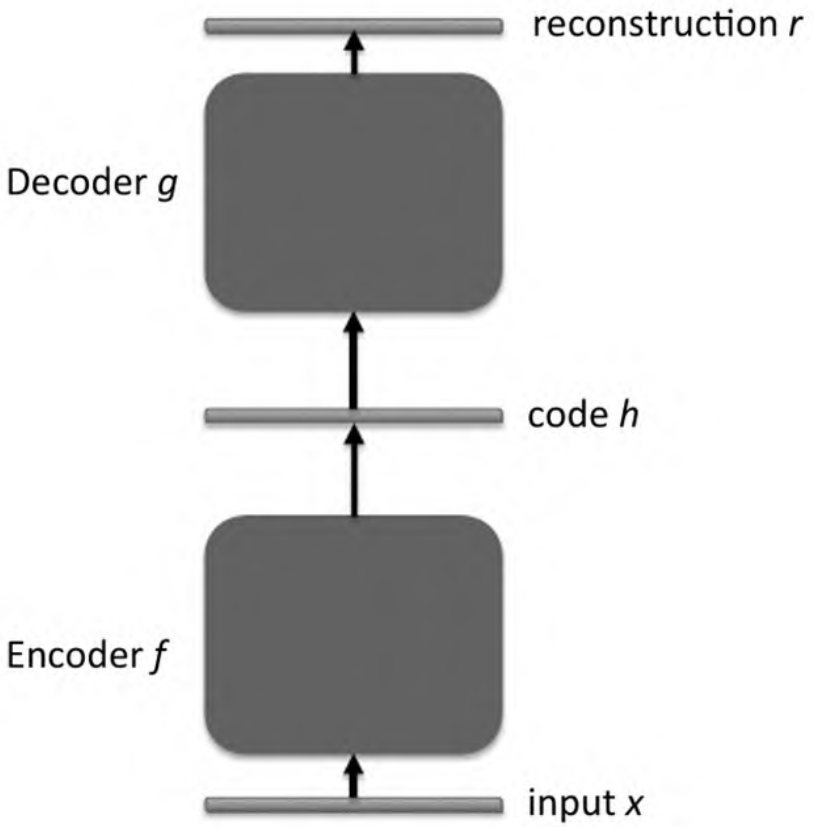


FIGURE 3.10 – Schéma général d'un encodeur automatique

Variational Autoencoders (VAE) : étendent cette idée dans un cadre probabiliste (Kingma Welling, 2014). Au lieu de produire une représentation déterministe h, l'encodeur d'un VAE apprend une distribution $q(h \mid x)$.(généralement Gaussienne) caractérisée par une moyenne et une variance (σ^2) Le décodeur génère ensuite x à partir de h échantillonné via $p(x \mid h)$. Le VAE optimise une borne inférieure de l'évidence (ELBO), combinant l'erreur de reconstruction et un terme de régularisation qui force $q(h \mid x)$ à ressembler à un prior $p(h)$(e.g., N(0,I)N(0,I)) :

$$\mathcal{L} = \mathbb{E}_{q(h|x)} \left[\log p(x \mid h)\right] - \mathrm{KL}\left(q(h \mid x) \,\|\, p(h)\right).$$

Cela permet aux VAE de générer de nouvelles données en échantillonnant hp(h)hp(h) puis en passant par le décodeur.[32].

3.5.5 Normalisation dans les réseaux de neurones (BatchNorm, LayerNorm) :

La normalisation est une technique essentielle pour stabiliser et accélérer l'entraînement des réseaux de neurones profonds. Deux méthodes principales dominent :

Batch Normalization (BatchNorm) : BatchNorm normalise les activations d'une couche en calculant la moyenne et la variance par mini-lot pendant l'entraînement. Elle réduit le Internal Covariate Shift (variation des distributions d'entrée entre les couches), permettant des taux d'apprentissage plus élevés et une meilleure stabilité. BatchNorm introduit des paramètres apprenables (,) par canal pour préserver la capacité expressive du modèle. En phase d'inférence, elle utilise des statistiques globales (moyenne/variance mobiles). Efficace pour les CNN, elle est moins adaptée aux RNN ou aux petits lots.[33].

Layer Normalization (LayerNorm) : LayerNorm normalise par exemple (sur toutes les dimensions des features d'une couche) plutôt que par canal et lot. Contrairement à BatchNorm, elle est indépendante de la taille du lot, ce qui la rend idéale pour les RNN (e.g., Transformers) et les tâches séquentielles. Elle utilise des statistiques calculées sur les activations d'une seule couche, ce qui simplifie son application aux modèles récurrents et aux lots de taille 1.[34].

Comparaison :

- **BatchNorm** excelle en vision par ordinateur (CNN) avec des lots suffisamment grands.

- **LayerNorm** est préférée pour les architectures séquentielles (RNN, Transformers) et les petits lots.

Ces méthodes améliorent la convergence et la généralisation, mais leur choix dépend de l'architecture et des contraintes (taille des lots, parallélisme). D'autres variantes comme InstanceNorm (pour le style transfer) ou GroupNorm (pour les petits lots) ciblent des cas d'usage spécifiques.

3.5.6 Techniques de régularisation :

La régularisation est essentielle pour améliorer la généralisation des réseaux de neurones et éviter le surapprentissage (overfitting). Parmi les techniques couramment utilisées, on trouve :

- **Dropout :** cette méthode consiste à désactiver aléatoirement une fraction des neurones pendant l'entraînement, ce qui empêche le réseau de trop dépendre de neurones spécifiques et favorise une représentation plus robuste des données.[35].

- **Early Stopping :** Cette approche consiste à interrompre l'entraînement lorsque la performance sur un ensemble de validation commence à se dégrader, évitant ainsi la suroptimisation sur les données d'entraînement [36].

- **Data Augmentation :** l'augmentation des données (par rotations, translations, etc.) enrichit artificiellement le jeu d'entraînement, améliorant la capacité du modèle à généraliser en exposant le réseau à davantage de variations des données d'origine.[37].

Ces techniques, combinées ou utilisées séparément, permettent d'obtenir des modèles plus performants sur des données non vues lors de l'apprentissage.

3.5.7 Apprentissage par transfert (Transfer Learning) :

L'apprentissage par transfert (Transfer Learning) est une technique de machine learning qui permet de réutiliser des connaissances acquises dans un domaine source (généralement avec de grandes quantités de données) pour améliorer les performances dans un domaine cible (souvent avec peu de données). Contrairement à l'apprentissage traditionnel, qui suppose que les données d'entraînement et de test suivent la même distribution, le transfer learning relâche cette hypothèse, ce qui le rend particulièrement utile dans des domaines comme la vision par ordinateur, le traitement du langage naturel (NLP) et la bioinformatique (Tan et al., 2018). Les approches courantes incluent :

- **Le fine-tuning :** Réutiliser un modèle pré-entraîné (ex. ResNet, BERT) et l'adapter au nouveau domaine en réentraînant partiellement ses couches.

- **L'extraction de caractéristiques :** Utiliser les couches basses d'un réseau neuronal comme extracteurs de features fixes.

- **Les méthodes adverses :** Aligner les distributions des domaines source et cible via des réseaux antagonistes (GANs).[38].

3.6 Architectures Avancées pour la Vision par Ordinateur :

3.6.1 U-Net : Structure en U pour la segmentation :

L'U-Net, est une architecture de réseau neuronal convolutif (CNN) spécialement conçue pour la segmentation d'images biomédicales. Sa structure symétrique en U combine une voie contractante (encodeur) pour capturer le contexte global et une voie expansive (décodeur) pour localiser précisément les objets. L'encodeur utilise des couches de convolution et de max-pooling pour réduire la résolution spatiale tout en augmentant la profondeur des caractéristiques. Le décodeur restaure la résolution via des transpositions de convolution (up-conv) et des connexions résiduelles (skip connections) qui fusionnent les caractéristiques haute résolution de l'encodeur avec les cartes up-samplées, améliorant ainsi la précision des contours cette architecture, optimisée pour les petits jeux de données grâce à une augmentation robuste des données (déformations élastiques), a établi l'état de l'art sur des défis comme la segmentation de structures neuronales en microscopie électronique (ISBI 2012). Son efficacité repose sur sa capacité à segmenter des images de grande taille via une stratégie de tuilage (overlap-tile) et son absence de couches fully connected, réduisant ainsi la complexité computationnelle.

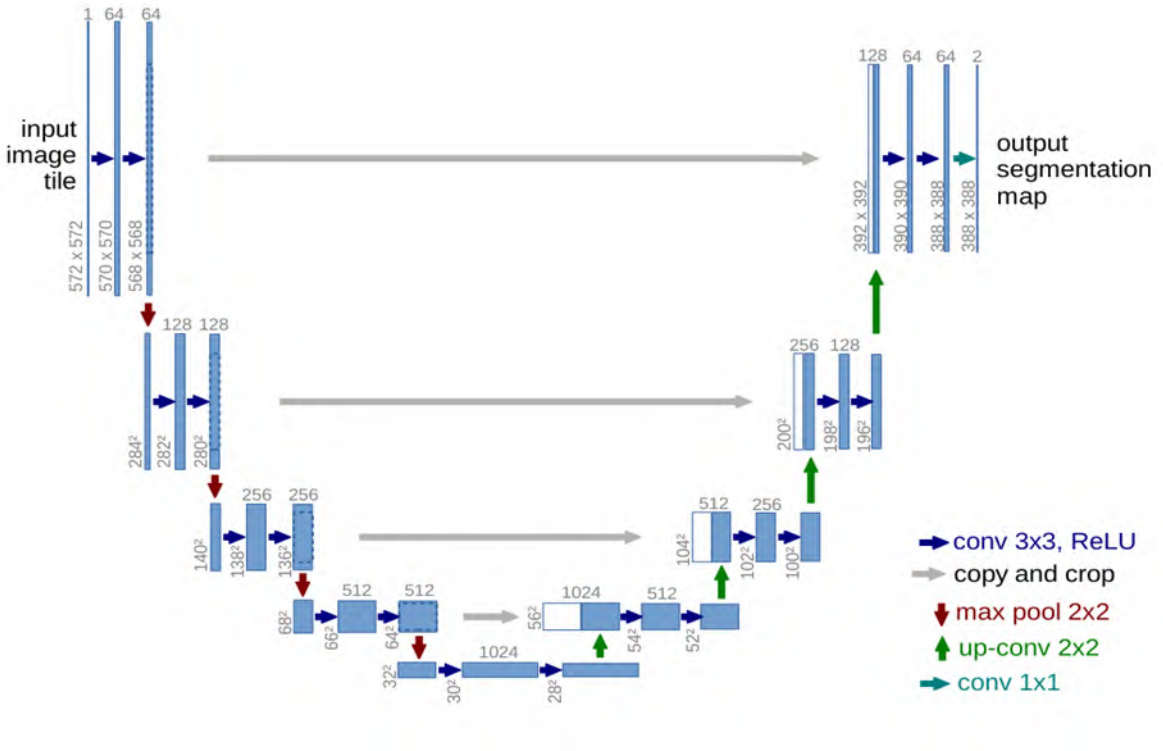


FIGURE 3.11 – architecture U-net

a) Encodeur (Partie Descendante) :

• **Fonction :** Extraire des caractéristiques hiérarchiques (e.g., contours, textures) via des couches convolutives et de pooling.

• **Opérations : Convolutions :** Apprentissage de motifs locaux (e.g., filtres 3x3 suivis de ReLU). **Pooling :** (max ou average) : Réduction de la résolution spatiale (par 2) pour capturer des informations globales.

b) Décodeur (Partie Montante) :

• **Fonction :** Reconstruire la segmentation à haute résolution à partir des caractéristiques compressées.

• **Opérations : Transposed Convolution (ou upsampling) :** Augmente la résolution spatiale. **Convolutions :** Affinement des caractéristaires upsamplées.

c) Connexions Skip : Fusion des cartes de caractéristiques de l'encodeur avec celles du décodeur (même résolution) pour restaurer les détails fins.

• **Rôle :** Combler le fossé entre informations locales (encodeur) et globales (décodeur).

• **Avantage :** Évite la perte de détails spatiaux due au pooling, crucial pour la segmentation précise (e.g., contours d'organes).[39].

3.6.2 Fully Convolutional Networks (FCN) :

Les Fully Convolutional Networks (FCN) sont une avancée majeure dans le domaine de la segmentation sémantique, permettant une prédiction dense (pixel à pixel) à partir d'images de taille arbitraire. Contrairement aux réseaux de convolution classiques utilisés pour la classification, qui incluent des couches entièrement connectées (fully connected) produisant des sorties de taille fixe, les FCN remplacent ces couches par des opérations convolutionnelles. Cette modification permet de conserver la nature spatiale des données et de générer des cartes de segmentation de mêmes dimensions que l'image d'entrée. Les FCN utilisent des couches de déconvolution (ou transposées de convolution) pour suréchantillonner les cartes de caractéristiques produites par les couches profondes, souvent trop grossières en raison des opérations de pooling. Une innovation clé est l'introduction de connexions skip, qui combinent des informations provenant de couches profondes (riches en sémantique) avec des couches plus superficielles (précises spatialement). Par exemple, la fusion des prédictions de pool4 (stride 16) et pool3 (stride 8) améliore la finesse des segmentations. Les FCN sont entraînés de bout en bout (end-to-end) avec une perte calculée au niveau des pixels, éliminant le besoin de prétraitements complexes (comme les superpixels) ou de post-traitements (comme les CRFs). Ils ont établi des records sur des benchmarks comme PASCAL VOC, NYUDv2 et SIFT Flow, tout en étant efficaces en calcul (inférence en quelques centaines de millisecondes). Leur flexibilité et leur performance en font une base pour de nombreuses architectures modernes de segmentation (U-Net, DeepLab, etc.).[40].

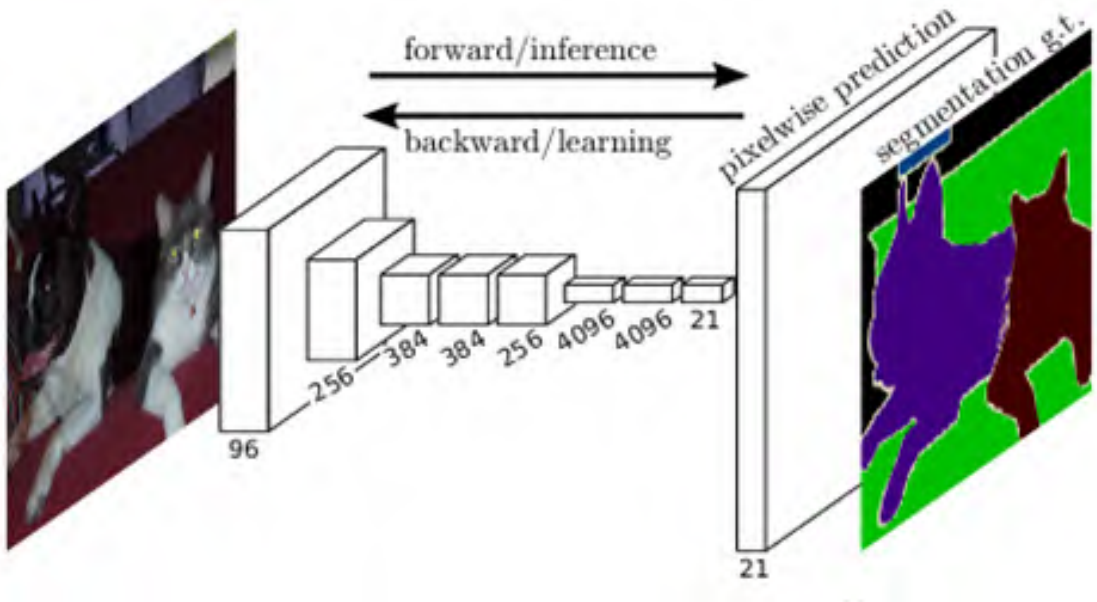


FIGURE 3.12 – Fully convolutional networks

3.6.3 SegNet : encodeur-décodeur

SegNet est une architecture de réseau neuronal convolutif profonde de type encodeur-décodeur, spécialement conçue pour la segmentation sémantique d'images.

L'encodeur : inspiré de VGG16, utilise des couches convolutives pour extraire des caractéristiques hiérarchiques, suivies d'opérations de max-pooling avec mémorisation des indices de pooling pour réduire la résolution spatiale.

Le décodeur : novateur, utilise ces indices pour effectuer un upsampling non linéaire, reconstruisant ainsi une segmentation précise tout en minimisant la mémoire requise. Contrairement à d'autres méthodes comme FCN.

SegNet évite d'apprendre l'upsampling et privilégie des filtres convolutifs entraînables pour densifier les cartes de caractéristiques. Cette approche équilibre précision et efficacité, notamment pour des applications en temps réel comme la compréhension de scènes routières ou intérieures.

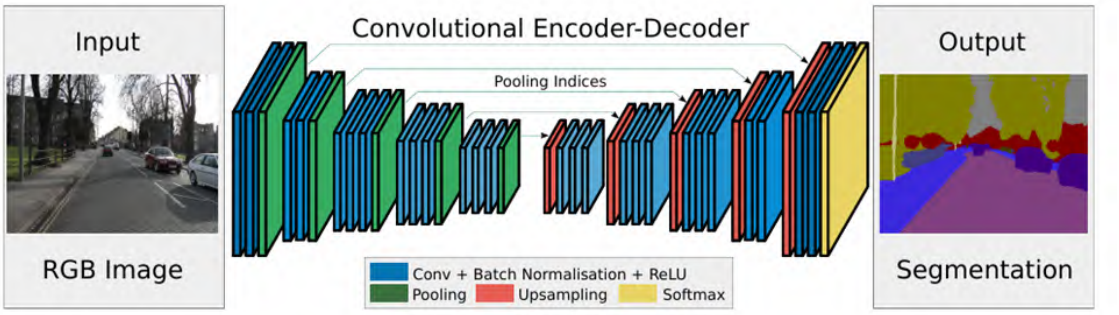


FIGURE 3.13 – Architecture SegNet

La figure 3.13 illustre l'architecture SegNet. Il n'y a pas de couches entièrement connectées, elle est donc uniquement convolutionnelle. Un décodeur suréchantillonne son entrée en utilisant les indices de pooling transférés depuis son encodeur pour produire une ou des cartes de caractéristiques creuses. Il effectue ensuite une convolution avec un banc de filtres entraînables pour densifier la carte de caractéristiques. Les cartes de caractéristiques finales en sortie du décodeur sont envoyées vers un classifieur soft-max pour une classification pixel par pixel.[41].

3.6.4 VGG, ResNet, EfficientNet :

VGG : Le modèle VGG (Visual Geometry Group) est une architecture de réseau de convolution profond (CNN) proposée par Karen Simonyan et Andrew Zisserman de l'Université d'Oxford en 2014. Cette architecture se distingue par son utilisation de couches convolutives empilées avec de petits filtres 3×3, permettant de capturer des motifs complexes tout en réduisant le nombre de paramètres par rapport à des filtres plus grands. Les réseaux VGG (notamment VGG-16 et VGG-19, avec 16 et 19 couches pondérées respectivement) ont démontré des performances exceptionnelles sur des tâches de classification d'images, comme le dataset ImageNet, où ils ont atteint des erreurs top-5 de 7,3

Une particularité de VGG est sa simplicité structurelle : des blocs répétés de convo-

lutions suivis de max-pooling, et des couches fully-connected en fin de réseau. Bien que gourmand en calcul en raison de sa profondeur, VGG a servi de base pour de nombreuses applications en vision par ordinateur, grâce à sa généralisation efficace comme extracteur de caractéristiques. Les poids pré-entraînés de VGG restent largement utilisés pour le transfer learning.[42].

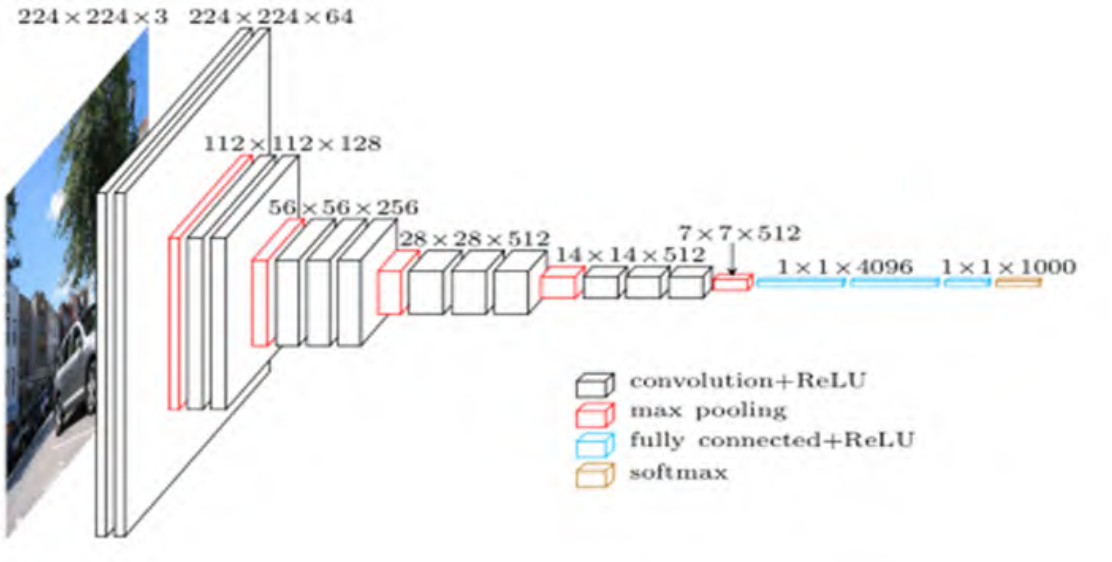


FIGURE 3.14 – Architecture VGG.

ResNet (Réseau Résiduel) : révolutionne l'apprentissage profond en permettant l'entraînement de réseaux de neurones extrêmement profonds (jusqu'à 152 couches ou plus) sans rencontrer le problème de dégradation des performances. L'innovation clé réside dans les connexions résiduelles (ou "shortcuts"), qui permettent au réseau d'apprendre une fonction résiduelle F(x)=H(x)xF(x)=H(x)x au lieu de la transformation directe H(x)H(x)

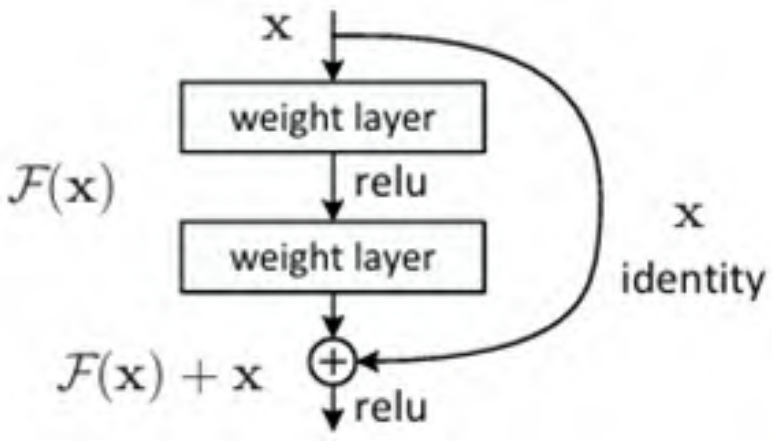


FIGURE 3.15 – Residual learning : a building block.

Ces connexions identité facilitent la circulation du gradient durant la rétropropagation, ce qui stabilise l'entraînement et améliore la précision. RésNet a remporté plusieurs

compétitions (ILSVRC 2015, COCO) et est devenu une pierre angulaire en vision par ordinateur, grâce à sa capacité à combiner profondeur et efficacité.

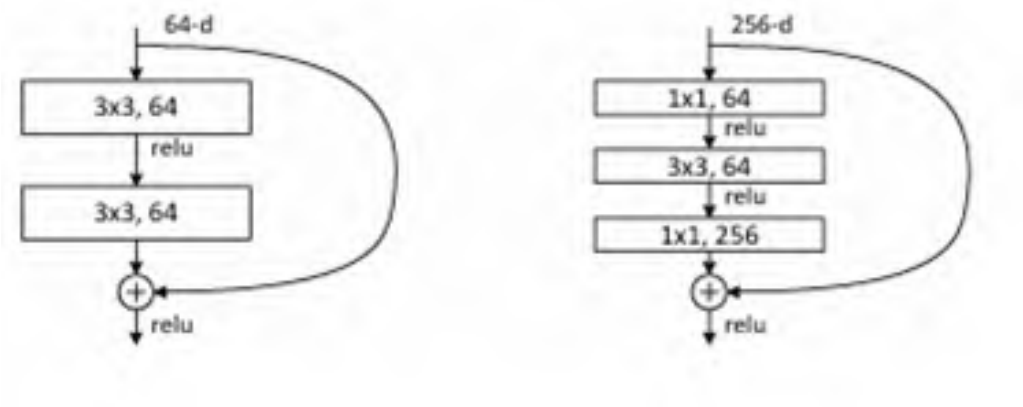


FIGURE 3.16 – Une fonction résiduelle F plus profonde pour ImageNet.

La figure 3.16 illustre À gauche : Un bloc de construction (appliqué sur des cartes de caractéristiques 56×56), similaire à celui de la Fig. 3 pour ResNet-34. À droite : Un bloc de construction de type « bottleneck » (goulot d'étranglement) utilisé pour ResNet-50/101/152.[43].

3.6.5 U-Net++ et DeepLabV3+ :

U-Net++ : est une architecture avancée pour la segmentation d'images médicales, proposée comme une amélioration de l'U-Net classique. Elle se distingue par l'introduction de connexions imbriquées et denses (nested dense skip pathways) entre l'encodeur et le décodeur, réduisant ainsi l'écart sémantique entre les cartes de caractéristiques des deux sous-réseaux. Contrairement à l'U-Net original où les connexions skip relient directement les couches de l'encodeur au décodeur, U-Net++ enrichit progressivement les caractéristiques de l'encodeur via des blocs de convolution denses avant leur fusion avec celles du décodeur, facilitant l'apprentissage. De plus, U-Net++ intègre une supervision profonde (deep supervision), permettant soit de combiner les prédictions multi-échelles pour une meilleure précision, soit de sélectionner une branche spécifique pour un compromis vitesse/précision. Évaluée sur des tâches variées (segmentation de nodules pulmonaires, noyaux cellulaires, foie et polypes), U-Net++ surpasse U-Net et sa version élargie (wide U-Net) avec un gain moyen de 3,9 points d'IoU, tout en offrant une flexibilité pour l'optimisation du temps d'inférence via l'élagage du modèle.

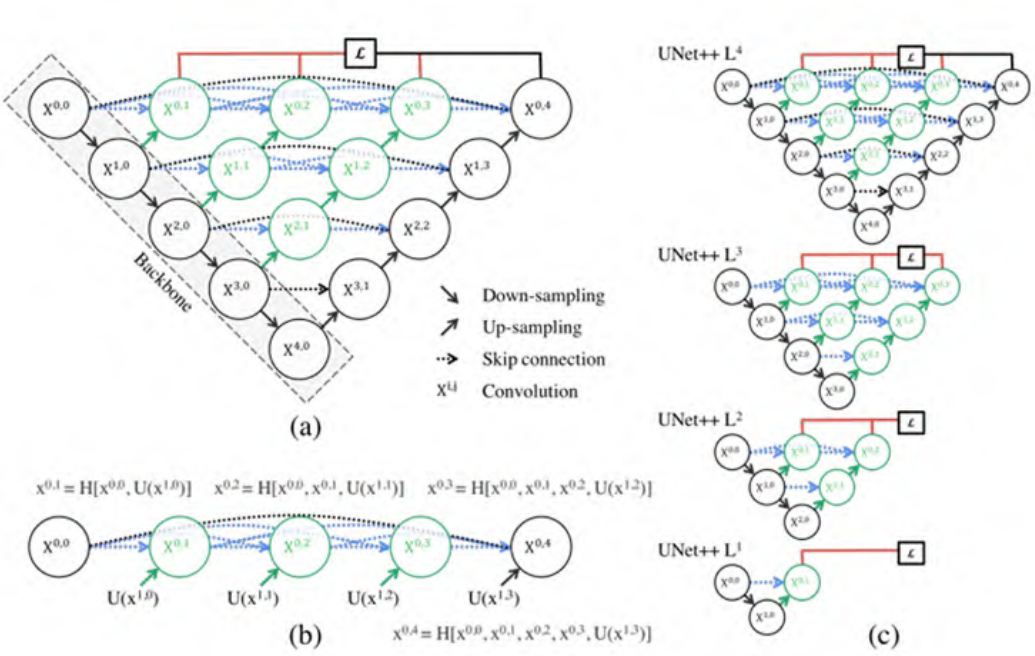


FIGURE 3.17 – (a) L'architecture UNet++ est composée d'un encodeur et d'un décodeur .

La figure 3.17 illustre L'idée principale derrière UNet++ est de réduire l'écart sémantique entre les cartes de caractéristiques de l'encodeur et du décodeur avant leur fusion. Par exemple, l'écart entre les niveaux (X00, X13) est comblé par un bloc de convolution dense à trois couches. Dans le schéma, les éléments en noir représentent l'U-Net original, tandis que le vert et le bleu illustrent les blocs de convolution denses sur les chemins skip, et le rouge indique la supervision profonde (deep supervision). Ces composants (rouge, vert, bleu) distinguent UNet++ de l'U-Net classique.(b) Analyse détaillée du premier chemin skip de UNet++.(c) UNet++ peut être élagué (pruned) lors de l'inférence si le modèle est entraîné avec supervision profonde, permettant ainsi un compromis entre précision et rapidité.[44].

DeepLabv3+ : est une architecture avancée pour la segmentation sémantique d'images, proposée par Google Research. Elle combine les avantages des modules de **pyramide spatiale atrous (ASPP)** pour capturer des informations contextuelles multi-échelles et une **structure encodeur-décodeur** pour restaurer les contours précis des objets. L'encodeur, basé sur DeepLabv3, utilise des convolutions atrous pour ajuster la résolution des caractéristiques extraites, tandis que le décodeur affine les prédictions en fusionnant des caractéristiques de bas niveau (issues de l'encodeur) avec des caractéristiques sémantiques riches. DeepLabv3+ intègre également des **convolutions séparables depthwise** (inspirées de Xception) pour réduire la complexité calculatoire sans sacrifier la performance. Évalué sur des benchmarks comme PASCAL VOC 2012 et Cityscapes, le modèle atteint des scores mIOU record sans post-traitement, tout en offrant une flexibilité pour équilibrer précision et vitesse via le paramétrage des résolutions de sortie. Une implémentation open source est disponible dans TensorFlow.

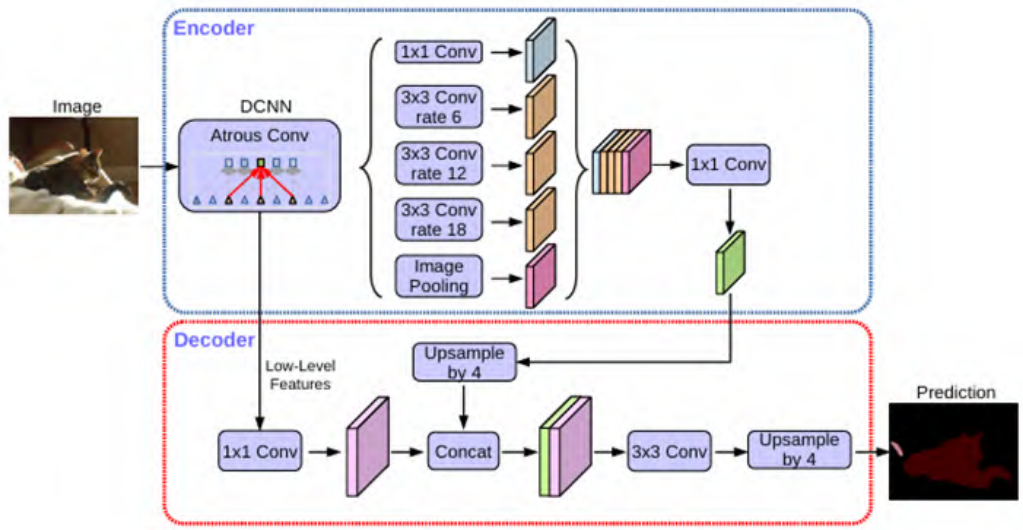


FIGURE 3.18 – DeepLabv3 en adoptant une structure encodeur-décodeur

La figure 3.2 illustre l'architecture générale d'un DeepLabv3. Le module encodeur est chargé de capturer des informations contextuelles à différentes échelles grâce à l'utilisation de convolutions atrous (ou dilatées) appliquées à plusieurs taux de dilatation. Cette stratégie permet d'élargir le champ réceptif sans perte de résolution spatiale. Quant au module décodeur, bien que relativement simple, il joue un rôle essentiel dans l'affinement des contours et la restitution précise des frontières des objets dans la carte de segmentation finale [45].

3.6.6 Comparaison des architectures [46].

Modèle	Type	Backbone	mIoU	Caractéristiques clés
U-Net	Encoder-Decoder	Custom CNN	~75–80%	Simplicité, efficace avec peu de données
U-Net++	Encoder-Decoder imbriqué	VGG, ResNet	~78–85%	Connexions skip imbriquées, meilleure capture des détails
SegNet	Encoder-Decoder	VGG16	~70–75%	Upsampling basé sur indices, nombre de paramètres réduit
DeepLabV3+	Encoder-Decoder avec convolutions dilatées	Xception-71	89.0%	ASPP, convolutions atrous, backbone puissant

TABLE 3.1 – Comparaison des architectures CNN pour la segmentation

3.7 Entraînement et Évaluation d'un Réseau de Neurones

3.7.1 Préparation des données (train, val, test)

Les données sont divisées comme suit : La préparation des données est une étape cruciale pour entraîner, valider et évaluer un modèle de deep learning. Les données sont généralement divisées en trois ensembles distincts :

— **Train** : Utilisé pour ajuster les paramètres du modèle (typiquement 70–80% des données).

— **Validation** : Permet de régler les hyperparamètres (ex : taux d'apprentissage, architecture) et d'éviter le surapprentissage (10–15% des données).

— **Test** : Évalue la performance finale du modèle sur des données jamais vues (10–15%).

3.7.2 Fonctions de perte (loss functions) :

1. Entropie croisée (Cross-Entropy Loss) : L'entropie croisée est une fonction de perte standard largement utilisée dans les tâches de classification, y compris la segmentation sémantique. Elle mesure la différence entre la distribution prédite par le modèle et la vérité terrain. En contexte de déséquilibre de classes, une version pondérée (weighted cross-entropy) peut être utilisée pour accorder plus d'importance aux classes rares, réduisant ainsi le biais vers la classe majoritaire. Cependant, son efficacité est limitée dans les cas de déséquilibres extrêmes, car elle traite chaque pixel indépendamment et ne prend pas en compte la structure spatiale globale de la prédiction.

$$\mathcal{L}_{\text{CE}} = -\sum_{i} y_i \log(\hat{y}_i)$$

Cette fonction est utilisée pour la classification et peut être pondérée en cas de déséquilibre de classes.

2. Dice Loss : La fonction de perte basée sur le coefficient de Dice est conçue pour maximiser l'overlap entre la prédiction et la vérité terrain. Elle est particulièrement adaptée aux problèmes de segmentation fortement déséquilibrés car elle mesure directement la qualité de la prédiction globale pour chaque classe. Le Dice Loss est différentiable et efficace pour entraîner des réseaux convolutifs sur des structures de petite taille telles que les lésions cérébrales ou les tumeurs. Sa forme normalisée permet de stabiliser l'apprentissage même en présence de très faibles volumes cibles

$$\mathcal{L}_{\text{Dice}} = 1 - \frac{2\sum_i y_i \hat{y}_i}{\sum_i y_i + \sum_i \hat{y}_i}$$

Elle maximise l'overlap entre la vérité terrain et la prédiction, particulièrement utile pour les classes rares.

3. IoU Loss (Jaccard Loss :) La perte basée sur l'intersection sur union (IoU), aussi appelée Jaccard loss, mesure le ratio entre l'intersection et l'union des ensembles de pixels prédits et réels. Bien qu'elle soit conceptuellement proche du Dice Loss, elle pénalise davantage les faux positifs et faux négatifs, ce qui peut la rendre plus robuste dans certaines configurations. Elle est utile dans les cas de segmentation avec bruit ou ambiguïté, mais sa dérivabilité peut être plus complexe à gérer dans certains frameworks de deep learning. Des variantes comme le Generalized Dice Loss introduisent des pondérations par classe pour corriger l'effet des déséquilibres extrêmes.[47].

$$\mathcal{L}_{\text{IoU}} = 1 - \frac{\sum_i y_i \hat{y}_i}{\sum_i y_i + \sum_i \hat{y}_i - \sum_i y_i \hat{y}_i}$$

3.7.3 Optimiseurs : SGD, Adam, RMSprop

— **SGD (Stochastic Gradient Descent)** : C'est l'optimiseur le plus basique, qui met à jour les paramètres du modèle en utilisant le gradient de la fonction de coût par rapport aux paramètres, avec un pas d'apprentissage fixe. Bien que simple, SGD peut être lent et sensible au bruit des gradients, surtout lorsque les données sont mal conditionnées.

— **RMSprop** : Proposé pour résoudre certains problèmes de SGD, RMSprop adapte le pas d'apprentissage pour chaque paramètre en normalisant les gradients par une moyenne mobile des carrés des gradients passés. Cela permet d'atténuer les oscillations dans les directions de forte courbure et d'accélérer la convergence dans les directions plates.

— **Adam(Adaptive Moment Estimation)** : Combinaison des idées de RMSprop et de la méthode des moments, Adam calcule des moyennes mobiles à la fois des gradients (premier moment) et de leurs carrés (deuxième moment). Il inclut également un terme de biais pour corriger les estimations initiales. Adam est largement utilisé en raison de sa robustesse et de sa capacité à adapter automatiquement les pas d'apprentissage pour chaque paramètre.
Ces optimiseurs diffèrent par leur complexité et leur efficacité, avec Adam étant souvent préféré pour sa performance générale, tandis que RMSprop et SGD restent pertinents dans des contextes spécifiques ou pour des problèmes bien conditionnés.[48].

3.7.4 Problèmes courants :

En apprentissage automatique, deux problèmes majeurs affectent fréquemment les performances des modèles : le sur-apprentissage (overfitting) et le sous-apprentissage (underfitting).

— **Sur-apprentissage (Overfitting)** :Le modèle apprend trop bien les données d'entraînement, y compris le bruit et les détails non pertinents, ce qui nuit à sa capacité à généraliser à de nouvelles données. Cela se manifeste par une erreur d'entraînement très faible mais une erreur de test élevée. Des techniques comme la régularisation (L1/L2), l'arrêt précoce (early stopping) ou l'augmentation des données peuvent aider à réduire ce phénomène.

— **Sous-apprentissage (Underfitting)** : À l'inverse, le modèle est trop simple pour capturer les motifs sous-jacents des données, résultant en une mauvaise performance aussi bien sur l'ensemble d'entraînement que sur de nouvelles données. Cela peut être résolu en augmentant la capacité du modèle (par exemple, en ajoutant des couches dans un réseau de neurones) ou en améliorant l'ingénierie des caractéristiques (feature engineering).[49]

3.7.5 Métriques d'évaluation

1.Précision, rappel et F1-score : L'évaluation des performances d'un modèle d'apprentissage supervisé repose sur un ensemble de métriques permettant de quantifier la qualité des prédictions. Pour les tâches de classification, les métriques les plus couramment utilisées sont la précision, le rappel et le F1-score.

— **La précision (ou "precision") :** mesure la proportion de prédictions positives correctes parmi toutes les prédictions positives Précision (Precision)

$$\text{Précision} = \frac{TP}{TP + FP}$$

— **Le rappel (ou "recall") :** indique la proportion de cas positifs correctement identifiés parmi tous les cas réellement positifs.

$$\text{Rappel} = \frac{TP}{TP + FN}$$

— **Le F1-score :** combine ces deux indicateurs en une moyenne harmonique, particulièrement utile en cas de déséquilibre des classes. F1-score = 2 × (Précision × Rappel) / (Précision + Rappel)[50][51][52].

$$F1 = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

2.Courbe ROC, AUC, IoU, mIoU

— **Courbe ROC (Receiver Operating Characteristic)** : trace la relation entre le taux de vrais positifs (TPR) et le taux de faux positifs (FPR).[52].

— **AUC (Area Under the Curve)** : résume la performance globale d'un classifieur.[53].

— **IoU (Intersection over Union)** :est principalement utilisée pour la segmentation sémantique, calculée comme :[54].

$$\text{IoU} = \frac{|\text{Prédiction} \cap \text{Réel}|}{|\text{Prédiction} \cup \text{Réel}|}$$

— **mIoU (mean IoU)** : est la moyenne de l'IoU sur toutes les classes du dataset, utile dans les tâches multiclasse.[55].

Conclusion

Le deep learning a révolutionné le domaine de l'intelligence artificielle en offrant des modèles capables d'apprendre des représentations hiérarchiques et abstraites à partir de vastes volumes de données. À travers ce chapitre, nous avons examiné ses fondements historiques et biologiques, les structures neuronales élémentaires, les architectures avancées (CNN, RNN, AE, etc.) ainsi que les techniques modernes d'entraînement et d'évaluation. L'efficacité du deep learning, en particulier dans des tâches complexes comme la segmentation d'images ou la traduction automatique, repose sur sa capacité à combiner puissance de calcul, grandes bases de données, et innovations algorithmiques. Malgré ses défis (surapprentissage, coût computationnel), il constitue aujourd'hui une technologie incontournable pour l'IA moderne et continue d'évoluer vers des modèles plus performants, efficaces et interprétables.

Chapitre 4

Implémentation et discussion des résultats

Introduction

Ce chapitre est consacré à la mise en œuvre pratique des modèles de segmentation sémantique étudiés dans les chapitres précédents. Après avoir sélectionné les architectures pertinentes, notamment U-Net, U-Net avec VGG16/VGG19 et ResNet50, nous avons procédé à leur implémentation sur un jeu de données d'images satellitaires, annotées pour différentes classes d'intérêt environnemental.

L'objectif principal de cette phase expérimentale est de comparer les performances de ces modèles selon des critères quantitatifs et qualitatifs, en mettant en lumière leurs capacités à délimiter finement les différentes régions d'intérêt. Nous présentons dans un premier temps l'environnement de développement utilisé, les choix techniques opérés (prétraitement des données, fonction de perte, métriques, etc.), puis les résultats obtenus lors des phases d'entraînement, de validation et de test.

Une attention particulière est portée à l'analyse des résultats par classes, à la visualisation des prédictions et aux limites observées lors des expériences. Ce chapitre se conclut par une discussion critique sur les performances observées, en lien avec les objectifs du mémoire.

4.1 Environnements et outils de développement

4.1.1 Google Colab

Google Colaboratory, souvent raccourci en "Colab" est un service cloud, offert par Google (gratuit), base sur Jupyter Notebook et destine a la formation à la recherche dans l'apprentissage automatique. Cette plateforme permet d'entrainer des modèles de Machine Learning directement dans le cloud. Sans donc avoir besoin d'installer quoi que ce soit sur notre ordinateur à l'exception d'un navigateur. Cet environnement nous permet d'écrire et d'exécuter du code Python dans votre navigateur, avec aucune configuration requise, accès gratuit aux GPU, partage facile.[56]

FIGURE 4.1 – Logo de Google Colab

Avantages de Google Colab :

- **Gratuit et accès GPU/TPU** : Utilisation gratuite de GPU (T4, P100) et TPU pour entraîner des modèles rapidement.
- **Prêt à l'emploi** : Aucune installation nécessaire ; bibliothèques comme Pas d'installation requise, avec de nombreuses bibliothèques préinstallées (TensorFlow, Keras, OpenCV, PyTorch...).
- **Intégration Google Drive** : Sauvegarde automatique, accès direct aux fichiers et partage facile..
- **Cloud multiplateforme** : Accessible depuis n'importe quel appareil, sans dépendre d'un PC puissant
- **Collaboration en temps réel** :Édition simultanée d'un notebook à plusieurs, comme sur Google Docs.
- **Compatible GitHub :**Chargement/sauvegarde direct depuis GitHub pour le versionnage du code.
- **Historique et récupération :** Suivi des modifications et retour à des versions précédentes.
- **Installation de packages personnalisés** :Possibilité d'installer n'importe quelle bibliothèque via !pip install

4.1.2 Python :

Python est le langage de programmation open source le plus utilisé dans le domaine de l'intelligence artificielle et surtout en Deep Learning vu qu'il contient un nombre important de bibliothèques performantes et utiles pour la vision artificielle et l'utilisation des réseaux de neurones. Il est conçu pour optimiser la productivité des programmeurs en orant des outils de haut niveau et une syntaxe simple à utiliser depuis quelques années. Notre modéle est fait avec le langage de Python dans le Cloud[57]

Notre modèle a été entièrement implémenté en Python et exécuté sur la plateforme Google Colab dans le cloud.

FIGURE 4.2 – Logo de Python

4.1.3 Les bibliothèques utilisées

Dans ce projet, plusieurs bibliothèques ont été utilisées pour la manipulation des données, la création du modèle de segmentation, l'entraînement, l'évaluation et la visualisation des résultats :

2.3.1 TensorFlow

TensorFlow est une bibliothèque open source développée par Google, dédiée au calcul numérique à grande échelle et à l'apprentissage automatique. Elle est conçue pour fonctionner efficacement sur divers types de matériels (CPU, GPU, TPU), ce qui en fait une solution flexible pour l'entraînement et l'inférence de modèles profonds.
Utilisation : Backend principal pour entraîner le modèle de segmentation sémantique.[58]

2.3.2 Keras (intégré à TensorFlow)

Keras est une interface de haut niveau intégrée à TensorFlow, facilitant la conception, l'entraînement et l'évaluation de réseaux neuronaux. Son API intuitive permet une modélisation rapide avec des abstractions prêtes à l'emploi.
Utilisation : Construction de l'architecture U-Net, gestion de l'entraînement.[59]

2.3.3 NumPy

NumPy est la bibliothèque fondamentale pour le calcul scientifique en Python. Elle permet la manipulation efficace de tableaux multidimensionnels et l'exécution de fonctions mathématiques.
Utilisation : Traitement numérique des images et des masques.[60]

2.3.4 OpenCV (cv2)

OpenCV est une bibliothèque open source dédiée au traitement d'images et à la vision par ordinateur.
Utilisation : Chargement et transformation des images et masques.[61]

2.3.5 Matplotlib

Matplotlib est une bibliothèque de visualisation puissante pour créer des graphiques statiques ou interactifs.
Utilisation : Visualisation des images d'entrée, masques, et sorties du modèle.[62]

2.3.6 Scikit-learn

Scikit-learn est une bibliothèque dédiée à l'apprentissage automatique.
Utilisation : Division du dataset en ensembles d'entraînement et de validation (`train_test_split`).[63]

2.3.7 Glob

`glob` est un module standard Python permettant de rechercher des fichiers à l'aide de motifs génériques.
Utilisation : Récupération automatique des chemins d'images et de masques.[64]

2.3.8 OS

`os` est un module standard permettant d'interagir avec le système d'exploitation.
Utilisation : Gestion des chemins de fichiers et des répertoires.[65]

2.3.9 to_categorical (Keras utils)

`to_categorical` de `tensorflow.keras.utils` convertit des étiquettes en vecteurs one-hot.
Utilisation : Formatage des masques pour une classification multiclasse.[66]

4.2 Base de données utilisée

Nous avons utilisé dans ce travail l'ensemble de données **EarthVQA**, conçu pour entraîner des modèles d'intelligence artificielle à répondre à des questions visuelles (*Visual Question Answering – VQA*) à partir d'images satellites à haute résolution.

Le dataset contient un grand nombre de paires *image-question-réponse*, couvrant divers domaines d'observation de la Terre comme les structures urbaines, les forêts, l'agriculture ou les plans d'eau. Il a été proposé pour encourager le développement de modèles capables d'interpréter et d'analyser des scènes complexes issues de l'imagerie satellitaire.

Les statistiques de téléchargement et la description complète du jeu de données sont accessibles en ligne [1].

Images satellites

— **Nombre d'images :** 6 000
— **Résolution :** 0,3 mètre par pixel
— **Source :** Nanjing, Changzhou et Wuhan (Chine)
— **Types de zones :** urbaines et rurales (bâtiments, routes, forêts, cultures)

Questions & réponses

— Plus de 208 000 paires question/réponse
— Chaque image est accompagnée de 20 à 50 questions

3.1.3 Masques sémantiques (annotations pixel par pixel)

— Chaque pixel d'une image satellite est associé à une étiquette de classe
— **Classes typiques :**

1. EarthVQA dataset :
`https://forms.office.com/pages/responsepage.aspx?id=DQSIkWdsW0yxEjajBLZtrQAAAAAAAAAAAAN_ _4KEQ7ZUOTdESFVLSFozMEZCMlVQVEczU1g3OTZDVy4u&route=shorturl`

TABLE 4.1 – Codes des classes et leurs significations

Code de classe	**Étiquette de la classe**
0	Fond / *Background*
1	Bâtiments / *Building*
2	Routes / *Road*
3	Eau / *Water*
4	Terrain nu / *Barren*
5	Forêt / *Forest*
6	Zones agricoles / *Agricultural*
7	Aire de jeux / *Playground*
8	Étang / *Pond*

4.3 Notre démarche pour la segmentation sémantique en utilisant le Deep Learning

4.3.1 Chargement des données

— **Source des données** : Le jeu de données utilisé est stocké dans Google Drive. Il est structuré en plusieurs dossiers :
`train/images_png`,
`train/masks_png`,
`val/images_png`
`val/masks_png`.

— **Montage du Drive** : L'accès aux données se fait à travers Google Colab après le montage de Google Drive (le montage n'est pas montré dans le code, mais il est implicite via les chemins `/content/drive/`).

— **Alignement des paires image/masque** : Une fonction `load_data()` est utilisée pour vérifier la correspondance entre les noms des images et des masques, évitant ainsi les erreurs d'appariement.

```python
def load_data(img_path, mask_path):
    image_files = sorted(glob(os.path.join(img_path, '*')))
    mask_files = sorted(glob(os.path.join(mask_path, '*')))

    image_basenames = {os.path.splitext(os.path.basename(f))[0]: f for f in image_files}
    mask_basenames = {os.path.splitext(os.path.basename(f))[0]: f for f in mask_files}

    common_basenames = sorted(list(image_basenames.keys() & mask_basenames.keys()))

    filtered_image_files = [image_basenames[name] for name in common_basenames]
    filtered_mask_files = [mask_basenames[name] for name in common_basenames]

    if len(image_files) != len(filtered_image_files):
        print(f"Warning: Found {len(image_files)} images and {len(mask_files)} masks.")
        print(f"Using {len(filtered_image_files)} matching pairs.")

    return filtered_image_files, filtered_mask_files
```

FIGURE 4.3 – fonction de load_data()

Cette fonction charge les chemins des images et des masques, et filtre pour ne garder que les paires valides ayant le même nom de base.

4.3.2 Préparation des données

— **Partitionnement** : Le jeu de données est divisé en trois sous-ensembles via `train_test_split` :
  — Entraînement : 80%
  — Validation : 10%
  — Test : 10%
— **Générateur de données** :La classe DataGenerator gère le chargement par batch, le prétraitement, et le mélange aléatoire pour éviter le surapprentissage.

```python
class DataGenerator(tf.keras.utils.Sequence):
    def __init__(self, img_paths, mask_paths, batch_size, shuffle=True):
        if len(img_paths) != len(mask_paths):
            raise ValueError("Image/mask count mismatch.")
        self.img_paths = img_paths
        self.mask_paths = mask_paths
        self.batch_size = batch_size
        self.shuffle = shuffle
        self.indexes = np.arange(len(self.img_paths))
        if self.shuffle:
            np.random.shuffle(self.indexes)
```

FIGURE 4.4 – fonction de DataGenerator

4.3.3 Prétraitement

— **Redimensionnement** : Les images et masques sont redimensionnés à `256×256` pixels à l'aide de `cv2.resize`.
— **Normalisation** : Les valeurs des pixels sont normalisées dans l'intervalle [0, 1] en les divisant par 255.
— **Gestion des masques** :
  — Les masques en niveaux de gris sont convertis en catégories discrètes (valeurs de 0 à 8) à l'aide de `np.where`.
  — Un encodage one-hot est appliqué via `to_categorical` pour s'adapter à la perte multi-classe.

```python
def preprocess_image(img_path):
    img = cv2.imread(img_path)
    img = cv2.resize(img, (IMG_WIDTH, IMG_HEIGHT))
    img = img / 255.0
    return img

def preprocess_mask(mask_path):
    mask = cv2.imread(mask_path, cv2.IMREAD_GRAYSCALE)
    mask = cv2.resize(mask, (IMG_WIDTH, IMG_HEIGHT), interpolation=cv2.INTER_NEAREST)
    mask = np.where(np.isin(mask, list(range(NUM_CLASSES))), mask, 0)
    return mask
```

FIGURE 4.5 – fonction de preprocess image mask

4.3.4 Entraînement

— **Optimiseur** : L'optimiseur Adam est utilisé avec un taux d'apprentissage de 1×10^{-4}.
— **Fonction de perte** : Une fonction personnalisée basée sur la moyenne entre la `categorical crossentropy` et la `Dice loss` est utilisée pour gérer le déséquilibre entre les classes et améliorer la précision globale.

```python
def dice_coefficient(y_true, y_pred, smooth=1.0):
    y_true_f = tf.cast(tf.reshape(y_true, [-1]), tf.float32)
    y_pred_f = tf.cast(tf.reshape(y_pred, [-1]), tf.float32)
    intersection = tf.reduce_sum(y_true_f * y_pred_f)
    return (2. * intersection + smooth) / (tf.reduce_sum(y_true_f) + tf.reduce_sum(y_pred_f) + smooth)

def dice_loss(y_true, y_pred):
    return 1.0 - dice_coefficient(y_true, y_pred)

def combined_loss(y_true, y_pred):
    ce_loss = tf.keras.losses.categorical_crossentropy(y_true, y_pred)
    return 0.5 * ce_loss + 0.5 * dice_loss(y_true, y_pred)
```

FIGURE 4.6 – fonction de loss

Formule mathématique de combined-loss : Soient :
— $y \in \mathbb{R}^{H \times W \times C}$: **le masque vrai** one-hot (ground truth)
— $\hat{y} \in \mathbb{R}^{H \times W \times C}$: **la prédiction du modèle** (softmax sur chaque pixel)
— H, W : hauteur et largeur de l'image
— C : nombre de classes

1. Entropie croisée catégorielle (Categorical Crossentropy)

Pour chaque pixel (i, j), l'entropie croisée est définie par :

$$\mathrm{CE}(y, \hat{y}) = -\sum_{i=1}^{H}\sum_{j=1}^{W}\sum_{c=1}^{C} y_{ijc} \cdot \log(\hat{y}_{ijc})$$

où $y_{i,j,c}$ est la valeur réelle (one-hot) et $\hat{y}_{i,j,c}$ la prédiction pour la classe c.
On prend généralement la moyenne sur tous les pixels de l'image :

$$\mathrm{CE}_{\mathrm{mean}} = -\frac{1}{H \cdot W}\sum_{i=1}^{H}\sum_{j=1}^{W}\sum_{c=1}^{C} y_{ijc} \cdot \log(\hat{y}_{ijc})$$

2. Dice Loss multiclasse

Le *Dice coefficient* pour chaque classe c est :

$$\mathrm{Dice}_c = \frac{2\sum_{i,j} y_{i,j,c} \cdot \hat{y}_{i,j,c}}{\sum_{i,j} y_{i,j,c} + \sum_{i,j} \hat{y}_{i,j,c} + \epsilon}$$

$$\mathrm{Dice}_{\mathrm{mean}} = \frac{1}{C}\sum_{c=1}^{C} \mathrm{Dice}_c$$

La Dice Loss est alors :

$$\mathrm{DiceLoss} = 1 - \mathrm{Dice}_{\mathrm{mean}}$$

Fonction de perte combinée

On combine souvent les deux pertes précédentes pour obtenir une perte plus robuste :

$$\mathrm{CombinedLoss}(y, \hat{y}) = 0.5 \cdot \mathrm{CE}_{\mathrm{mean}} + 0.5 \cdot \mathrm{DiceLoss}$$

Soit :

$$\mathcal{L}_{\mathrm{combined}} = 0.5 \cdot \left(-\frac{1}{HW}\sum_{i,j,c} y_{ijc} \log(\hat{y}_{ijc}) \right) + 0.5 \cdot \left(1 - \frac{1}{C}\sum_{c=1}^{C} \frac{2\sum_{i,j} y_{ijc}\hat{y}_{ijc}}{\sum_{i,j} y_{ijc} + \sum_{i,j} \hat{y}_{ijc} + \epsilon} \right)$$

où α et β sont des coefficients d'équilibrage (par exemple $\alpha = 0.5$, $\beta = 0.5$).

Callbacks utilisés

- **ModelCheckpoint** : sauvegarde automatique du meilleur modèle basé sur le Dice coefficient.
- **EarlyStopping** : arrêt anticipé de l'entraînement si le Dice coefficient n'évolue plus.
- **ReduceLROnPlateau** : réduction du taux d'apprentissage si stagnation du Dice.

4.3.5 Métriques d'évaluation :

— **Dice Coefficient** : mesure de la similarité entre les prédictions et les masques réels.

— **Accuracy** : taux global de pixels correctement classés.

— **Visualisation** : la fonction `visualize_predictions()` permet de comparer les prédictions avec les masques de vérité terrain.

4.3.6 Architecture du modèle utilisér :

Modèle 1 (U-Net simple) :

Le modèle utilisé est une version simple et personnalisée du réseau U-Net, une architecture convolutive conçue pour la segmentation d'images. Il se compose de quatre parties principales :

Encodeur (partie descendante)

— Composé de 4 blocs de convolution : chaque bloc applique deux convolutions suivies d'une activation ReLU, puis d'un `MaxPooling`.
— La taille spatiale est progressivement réduite tandis que le nombre de filtres augmente : $64 \rightarrow 512$.
— Fonction principale : extraire les caractéristiques contextuelles globales de l'image.

Bottleneck

— Il s'agit du point le plus profond du réseau.
— Deux convolutions sont appliquées avec un nombre élevé de filtres (1024).
— Ce bloc capture des représentations riches et abstraites des données.

Décodeur (partie montante)

— Composé de 4 blocs de dé-convolution : chaque bloc utilise un `Upsampling` suivi d'une `Conv2DTranspose`, puis concatène les caractéristiques avec celles de l'encodeur correspondant (skip connections).
— Permet une reconstruction progressive de la carte de segmentation.
— Les détails spatiaux sont récupérés grâce aux concaténations avec les couches de l'encodeur.

Couche de sortie

— Une couche `Conv2D` avec `NUM_CLASSES = 9` filtres et une activation `softmax`.
— Elle produit une carte de segmentation finale, où chaque pixel est assigné à une catégorie.

TABLE 4.2 – Architecture du modèle U-Net avec fonctions d'activation

Étape	Type de couche	Taille d'entrée	Activation
Entrée	-	256 × 256 × 3	-
Encodeur - Bloc 1	Conv2D(64) ×2 + MaxPooling2D	256 × 256 × 3	ReLU
Encodeur - Bloc 2	Conv2D(128) ×2 + MaxPooling2D	128 × 128 × 64	ReLU
Encodeur - Bloc 3	Conv2D(256) ×2 + MaxPooling2D	64 × 64 × 128	ReLU
Encodeur - Bloc 4	Conv2D(512) ×2 + MaxPooling2D	32 × 32 × 256	ReLU
Bottleneck	Conv2D(1024) ×2	16 × 16 × 512	ReLU
Décodeur - Bloc 1	Conv2DTranspose(512) + concat + Conv2D(512) ×2	16 × 16 × 1024	ReLU
Décodeur - Bloc 2	Conv2DTranspose(256) + concat + Conv2D(256) ×2	32 × 32 × 512	ReLU
Décodeur - Bloc 3	Conv2DTranspose(128) + concat + Conv2D(128) ×2	64 × 64 × 256	ReLU
Décodeur - Bloc 4	Conv2DTranspose(64) + concat + Conv2D(64) ×2	128 × 128 × 128	ReLU
Sortie	Conv2D(9, 1×1)	256 × 256 × 64	Softmax

Entraînement par CNN :

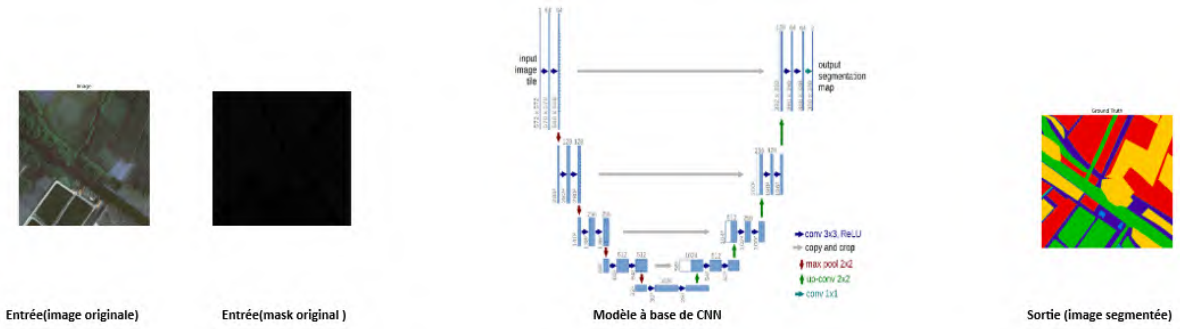


FIGURE 4.7 – Structure générale de segmentation sémantique par apprentissage profond

Modèle 2 (U-Net avec VGG16)

Le second modèle repose sur l'architecture U-Net, mais utilise cette fois le réseau pré-entraîné **VGG16** comme encodeur (*backbone*). Cette approche s'appuie sur le *transfert learning*, ce qui permet au modèle de bénéficier des caractéristiques préalablement apprises sur ImageNet, un large corpus d'images généralistes.
L'architecture complète se divise également en quatre parties principales :

Encodeur (partie descendante - VGG16)

- Le cœur de l'encodeur est constitué du modèle **VGG16**, sans sa partie *fully connected* (`include_top=False`) et avec des poids pré-entraînés sur ImageNet.
- Les sorties intermédiaires des blocs `block1_conv2`, `block2_conv2`, `block3_conv3`, `block4_conv3` et `block5_conv3` sont extraites pour être utilisées dans les connexions d'encodage/décodage.
- Ce *backbone* extrait des caractéristiques riches et hiérarchiques tout en réduisant progressivement la taille spatiale de l'image.
- Une option permet de geler les poids du VGG16 afin de réaliser un transfert learning partiel (`freeze_encoder=True`).

Bottleneck

- Le bloc `block5_conv3` agit comme goulot d'étranglement.
- Il capture les représentations les plus abstraites et discriminantes de l'image.

Décodeur (partie montante)

- Il se compose de quatre blocs de déconvolution (`Conv2DTranspose`) suivis de concaténations avec les sorties correspondantes du VGG16 (*skip connections*).
- Chaque bloc est suivi de deux convolutions 3×3 avec activation ReLU et d'une couche de Dropout pour régularisation.
- Cette phase permet une reconstruction progressive de la carte de segmentation.

Couche de sortie

- Une couche de convolution 1×1 avec 9 filtres (correspondant au nombre de classes), suivie d'une activation `softmax`, permet de générer la carte de segmentation finale.
- Chaque pixel est ainsi affecté à l'une des 9 classes possibles.

TABLE 4.3 – Architecture du modèle U-Net basé sur VGG16 avec Dropout

Étape	**Type de couche**	**Taille d'entrée**	**Activation**
Entrée	-	$256 \times 256 \times 3$	-
Encodeur	VGG16 pré-entraîné (jusqu'à `block5_conv3`)	$256 \times 256 \times 3$	ReLU
Bottleneck	Sortie de `block5_conv3`	$16 \times 16 \times 512$	ReLU
Décodeur - Bloc 1	TransConv(512) + concat + Conv2D(512) $\times$ 2 + Dropout(0.3)	$16 \times 16 \times 512$	ReLU
Décodeur - Bloc 2	TransConv(256) + concat + Conv2D(256) $\times$ 2 + Dropout(0.3)	$32 \times 32 \times 512$	ReLU
Décodeur - Bloc 3	TransConv(128) + concat + Conv2D(128) $\times$ 2 + Dropout(0.2)	$64 \times 64 \times 256$	ReLU
Décodeur - Bloc 4	TransConv(64) + concat + Conv2D(64) $\times$ 2 + Dropout(0.1)	$128 \times 128 \times 128$	ReLU
Sortie	Conv2D(9, 1×1) + Softmax	$256 \times 256 \times 64$	Softmax

Modèle 3 (U-Net avec VGG19)

Le deuxième modèle implémente une architecture U-Net en utilisant **VGG19** comme encodeur pré-entraîné sur ImageNet. Cette stratégie combine les avantages de la structure en U (skip connections et reconstruction progressive) avec la puissance d'extraction de caractéristiques d'un réseau profond et pré-entraîné.
L'architecture est divisée en quatre parties principales :

- **Encodeur (partie descendante)** :
  - L'encodeur est basé sur *VGG19*, un réseau convolutionnel profond composé de blocs de convolutions suivis de couches de max pooling.
  - Les sorties de certains blocs intermédiaires sont extraites (`block1_conv2`, `block2_conv2`, ..., `block5_conv4`) afin de permettre les connexions par saut (*skip connections*).
  - Il permet une extraction hiérarchique de caractéristiques de bas niveau (textures, bords) jusqu'à des caractéristiques plus abstraites.

- **Bottleneck** :
  - La couche la plus profonde du réseau est représentée par la sortie du dernier bloc convolutionnel de VGG19 (`block5_conv4`), avec un grand nombre de filtres (512).
  - Elle capture des représentations riches et compressées de l'image d'entrée.

- **Décodeur (partie montante)** :
  - Le décodeur est construit de manière symétrique à l'encodeur, en appliquant :
    - *Upsampling* via `Conv2DTranspose`,
    - *Concaténation* avec les cartes d'activation de l'encodeur correspondantes,
    - Deux convolutions 3×3 avec activation `ReLU`.
  - Des *couches de dropout* sont ajoutées après chaque bloc de décodage afin de réduire le surapprentissage.

— Cette phase permet la reconstruction progressive de la carte de segmentation, tout en récupérant les détails spatiaux grâce aux connexions par saut.

— **Couche de sortie** :
  — Une couche `Conv2D 1×1` avec `NUM_CLASSES = 9` filtres et une activation `softmax` est utilisée.
  — Elle génère une carte de segmentation dans laquelle chaque pixel est affecté à une classe.

TABLE 4.4 – Architecture du modèle U-Net basé sur VGG19 avec Dropout

Étape	**Type de couche**	**Taille d'entrée**	**Activation**
Entrée	-	$256 \times 256 \times 3$	-
Encodeur - Bloc 1	VGG19 : `block1_conv2`	$256 \times 256 \times 3$	ReLU
Encodeur - Bloc 2	VGG19 : `block2_conv2`	$128 \times 128 \times 64$	ReLU
Encodeur - Bloc 3	VGG19 : `block3_conv4`	$64 \times 64 \times 128$	ReLU
Encodeur - Bloc 4	VGG19 : `block4_conv4`	$32 \times 32 \times 256$	ReLU
Bottleneck	VGG19 : `block5_conv4`	$16 \times 16 \times 512$	ReLU
Décodeur - Bloc 1	Conv2DTranspose(512) + concat + Conv2D(512) $\times$ 2	$16 \times 16 \times 512$	ReLU
Décodeur - Bloc 2	Conv2DTranspose(256) + concat + Conv2D(256) $\times$ 2	$32 \times 32 \times 512$	ReLU
Décodeur - Bloc 3	Conv2DTranspose(128) + concat + Conv2D(128) $\times$ 2	$64 \times 64 \times 256$	ReLU
Décodeur - Bloc 4	Conv2DTranspose(64) + concat + Conv2D(64) $\times$ 2	$128 \times 128 \times 128$	ReLU
Sortie	Conv2D(9, 1×1)	$256 \times 256 \times 64$	Softmax

Modèle 4 : U-Net avec ResNet50

Le deuxième modèle repose sur une version avancée du réseau U-Net, où l'encodeur classique est remplacé par le modèle **ResNet50 pré-entraîné sur ImageNet**, servant de backbone. Cette approche permet de bénéficier de l'apprentissage préalable de caractéristiques visuelles pertinentes sur une large base de données. L'architecture conserve la structure typique en quatre parties :

— **Encodeur (partie descendante)** :
  — Utilise les couches convolutives du ResNet50 comme encodeur, avec des blocs résiduels (*residual blocks*) permettant une meilleure propagation du gradient.
  — Les *skip connections* sont extraites à différentes profondeurs du réseau : `conv1`, `conv2`, `conv3`, `conv4`.
  — La taille spatiale est progressivement réduite (de 256×256 à 8×8), tandis que la profondeur des canaux augmente (de 64 à 2048).
  — Fonction principale : apprendre des représentations contextuelles hiérarchiques riches.

— **Bottleneck** :
  — Correspond à la sortie finale du ResNet50 (niveau `conv5`), contenant des représentations denses et abstraites.
  — Cette couche constitue le cœur informationnel de l'architecture.

— **Décodeur (partie montante)** :
  — Composé de 5 blocs de dé-convolution : chaque bloc applique une couche `Conv2DTranspose` (ou un `Upsampling`), puis concatène avec les sorties correspondantes de l'encodeur (*skip connections*).
  — Chaque bloc est suivi de deux couches `Conv2D` avec activation ReLU et d'une couche de `Dropout` pour la régularisation.
  — Permet une reconstruction progressive de la carte de segmentation, en intégrant à la fois les détails locaux et le contexte global.

— **Couche de sortie** :
  — Une couche finale `Conv2D(9, 1×1)` avec une activation `Softmax` est utilisée.
  — Elle génère la carte de segmentation finale, où chaque pixel est classifié dans l'une des 9 classes cibles.

TABLE 4.5 – Architecture du modèle U-Net avec ResNet50

Étape	**Type de couche**	**Taille d'entrée**	**Activation**
Entrée	-	$256 \times 256 \times 3$	-
Encodeur - Bloc 1	ResNet50 `conv1_relu`	$128 \times 128 \times 64$	ReLU
Encodeur - Bloc 2	`conv2_block3_out`	$64 \times 64 \times 256$	ReLU
Encodeur - Bloc 3	`conv3_block4_out`	$32 \times 32 \times 512$	ReLU
Encodeur - Bloc 4	`conv4_block6_out`	$16 \times 16 \times 1024$	ReLU
Bottleneck	`conv5_block3_out`	$8 \times 8 \times 2048$	ReLU
Décodeur - Bloc 1	Conv2DTranspose(512) + concat + Conv2D(512)×2 + Dropout	$16 \times 16 \times 2048$	ReLU
Décodeur - Bloc 2	Conv2DTranspose(256) + concat + Conv2D(256)×2 + Dropout	$32 \times 32 \times 1024$	ReLU
Décodeur - Bloc 3	Conv2DTranspose(128) + concat + Conv2D(128)×2 + Dropout	$64 \times 64 \times 512$	ReLU
Décodeur - Bloc 4	Conv2DTranspose(64) + concat + Conv2D(64)×2 + Dropout	$128 \times 128 \times 256$	ReLU
Décodeur - Bloc 5	Conv2DTranspose(32) + Conv2D(32)×2	$256 \times 256 \times 128$	ReLU
Sortie	Conv2D(9, 1×1)	$256 \times 256 \times 32$	Softmax

4.4 Les résultats obtenus de notre approche :

4.4.1 Analyse comparative des performances des architectures de segmentation sémantique

U-Net simple :

Les courbes du modèle U-Net montrent une convergence progressive avec une diminution continue de la loss et une amélioration constante de l'accuracy et du dice score. Bien que les performances soient globalement correctes, l'écart entre les courbes d'entraînement et de validation devient visible en fin d'apprentissage, ce qui peut suggérer un léger surapprentissage (overfitting). Toutefois, la stabilité des courbes indique une capacité raisonnable de généralisation.

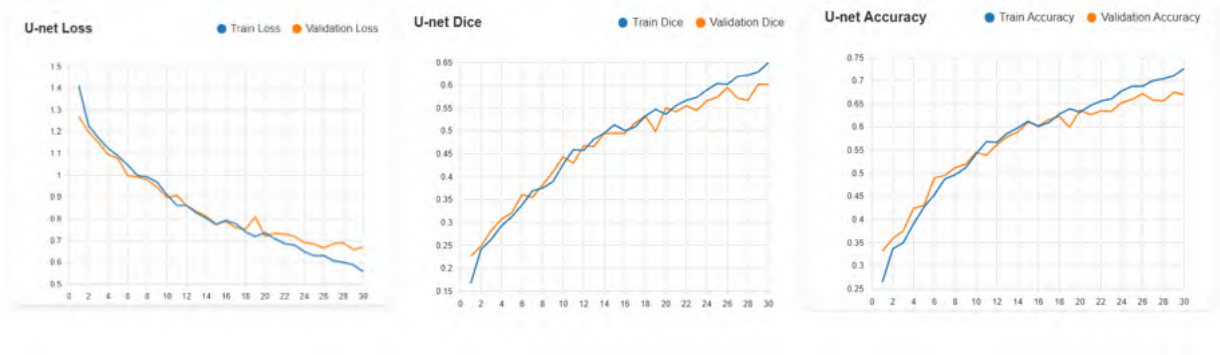


FIGURE 4.8 – Évolution de la loss, du Dice coefficient et de l'accuracy pour le modèle U-Net sur les ensembles d'entraînement et de validation.

VGG16 :

Les performances du modèle basé sur VGG16 montrent une amélioration rapide de la dice coefficient et de l'accuracy dès les premières époques, accompagnée d'une forte diminution de la loss. Cela témoigne de la puissance de la profondeur du réseau et de ses capacités d'extraction de caractéristiques. On observe également une bonne stabilité entre les courbes d'entraînement et de validation, signe d'une bonne généralisation. Cela suggère que VGG16 est bien adapté à la tâche de segmentation considérée, surtout lorsqu'il est utilisé comme backbone d'un U-Net modifié

FIGURE 4.9 – Évolution de la loss, du Dice coefficient et de l'accuracy pour le modèle VGG16 sur les ensembles d'entraînement et de validation.

VGG19 :

Le modèle VGG19 présente un comportement similaire à VGG16 avec toutefois un légère instabilité en début d'entraînement, visible par une variation importante des courbes. Néanmoins, le modèle atteint rapidement une bonne performance en termes de dice score et d'accuracy. Il est possible que la profondeur supplémentaire de VGG19 entraîne une convergence plus lente ou une sensibilité accrue à l'initialisation des poids. En fin d'apprentissage, les courbes montrent une bonne cohérence entre les données d'entraînement et de validation.

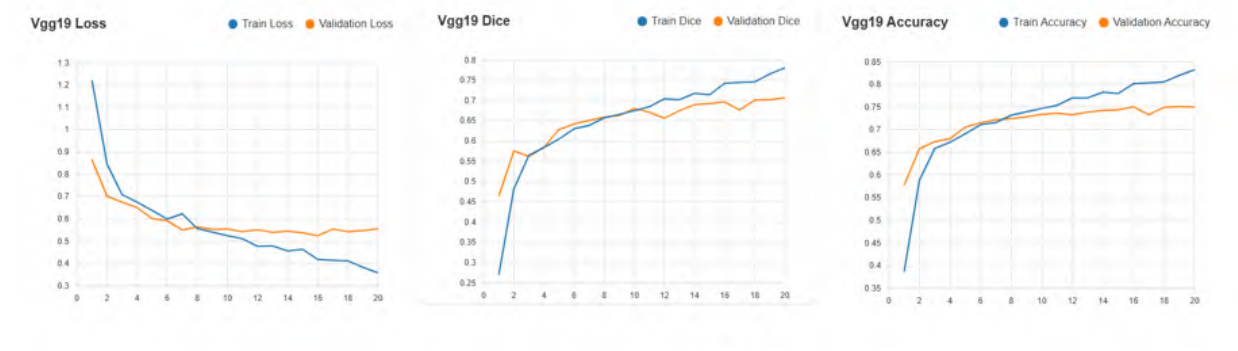


FIGURE 4.10 – Évolution de la loss, du Dice coefficient et de l'accuracy pour le modèle VGG19 sur les ensembles d'entraînement et de validation.

ResUNet50 :

Le modèle ResUNet se distingue par une phase initiale d'instabilité marquée dans la loss d'entraînement, possiblement due à la complexité de l'architecture résiduelle. Cependant, après cette phase transitoire, le modèle montre une amélioration rapide et une stabilisation des métriques. On remarque également une hausse significative de la dice coefficient et de l'accuracy, surpassant visiblement les autres modèles. Cela confirme que l'utilisation des blocs résiduels permet d'améliorer la transmission des gradients et de faciliter l'apprentissage dans les réseaux profonds, tout en réduisant le risque de dégradation des performances.

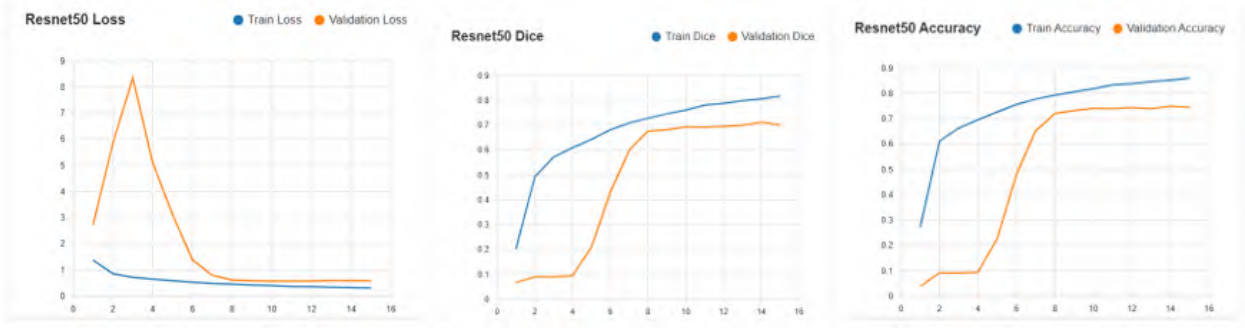


FIGURE 4.11 – Évolution de la loss, du Dice coefficient et de l'accuracy pour le modèle ResNet50 sur les ensembles d'entraînement et de validation.

4.4.2 Analyse des matrices de confusion pour les modèles de segmentation

Les matrices de confusion présentées permettent une évaluation fine des performances des différentes architectures (**U-Net**, **VGG16**, **VGG19**, **ResNet50**) en termes de classification des pixels pour chaque classe dans la tâche de segmentation sémantique. Chaque ligne représente les pixels réels d'une classe, tandis que chaque colonne représente les prédictions faites par le modèle.

1. U-Net

Le modèle **U-Net** montre une bonne performance générale, notamment pour les classes 0, 1 et 2, avec une concentration marquée des prédictions sur la diagonale. Cependant, des confusions sont visibles pour certaines classes adjacentes (ex. classes 4, 5 et 6), ce qui pourrait s'expliquer par une faible séparation sémantique entre ces classes dans les données ou un manque de profondeur dans l'architecture pour capturer des caractéristiques fines. L'équilibre entre les classes semble également affecter les performances pour les classes minoritaires.

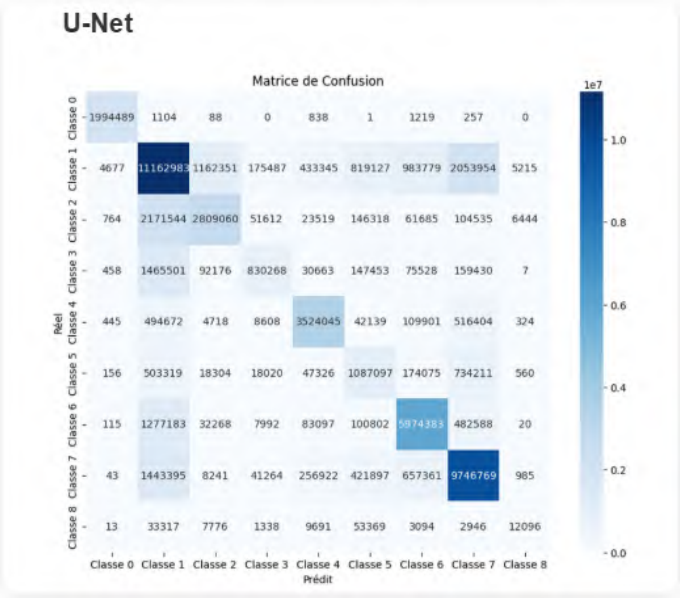


FIGURE 4.12 – Matrice de confusion du modèle U-Net

2. VGG16

L'architecture **VGG16** améliore visiblement la précision de la segmentation, avec une diagonale plus marquée et moins de dispersion dans la matrice de confusion. Les classes sont mieux séparées les unes des autres, ce qui témoigne de meilleures capacités d'extraction de caractéristiques. Le modèle semble particulièrement performant sur les classes 0, 1, 2 et 6, ce qui pourrait être dû à une meilleure représentation des contextes spatiaux dans les couches convolutives profondes.

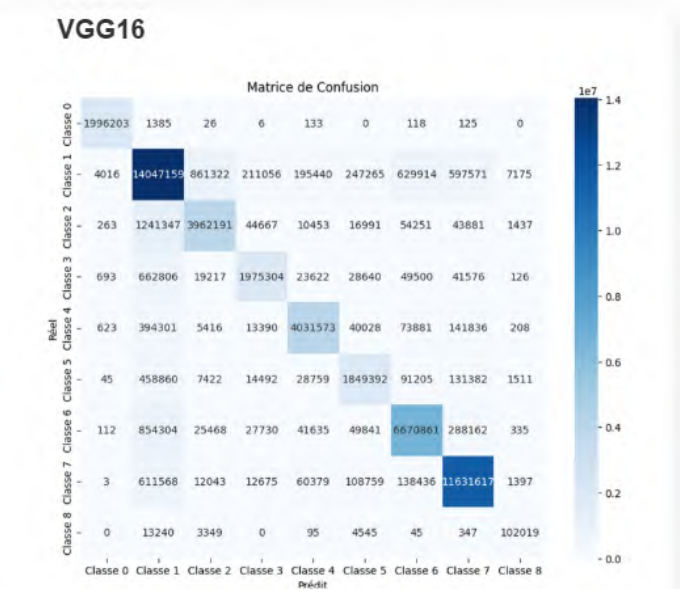


FIGURE 4.13 – Matrice de confusion du modèle VGG16

3. VGG19

VGG19 présente des résultats comparables à VGG16, avec cependant légèrement plus de confusion inter-classes dans certaines zones, notamment entre les classes 3, 5 et 6. Cela pourrait résulter de l'augmentation de la profondeur du réseau, qui rend l'apprentissage plus difficile sans ajustement fin des hyperparamètres. Néanmoins, la structure générale de la matrice reste cohérente et la majorité des prédictions se situent toujours sur la diagonale.

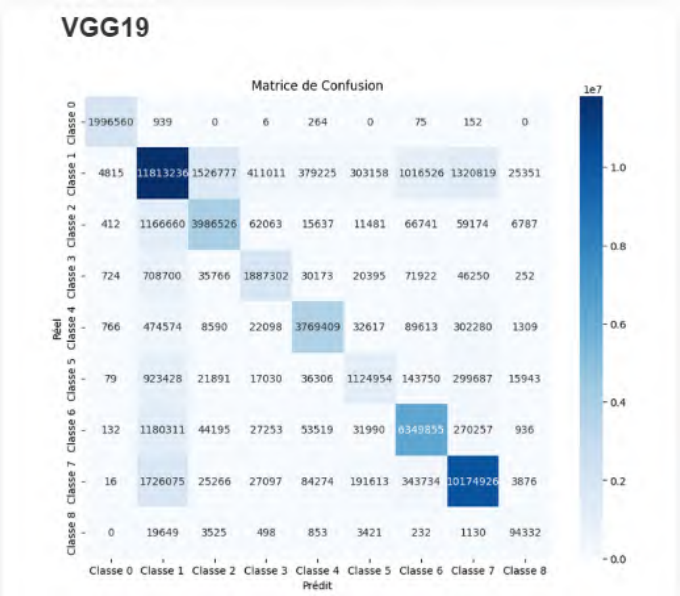


FIGURE 4.14 – Matrice de confusion du modèle VGG19

4. ResNet50

La matrice de confusion de **ResNet50** montre une bonne répartition des prédictions avec des performances très compétitives, notamment sur les classes 0, 1, 2 et 7. On observe cependant une plus grande dispersion pour les classes 4 et 5, ce qui pourrait indiquer une certaine sensibilité à la variabilité intra-classe. L'utilisation des blocs résiduels semble aider à mieux préserver les gradients dans les couches profondes, mais pourrait nécessiter un ajustement plus fin ou un entraînement plus long pour améliorer la stabilité.

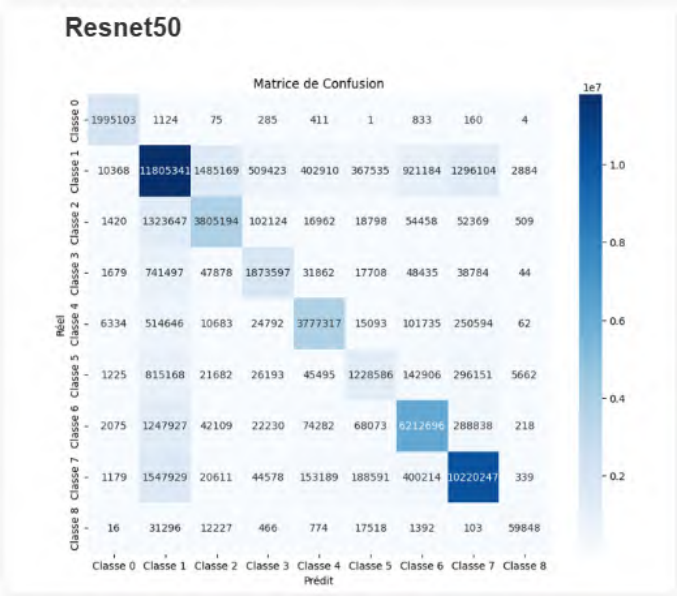


FIGURE 4.15 – Matrice de confusion du modèle ResNet50

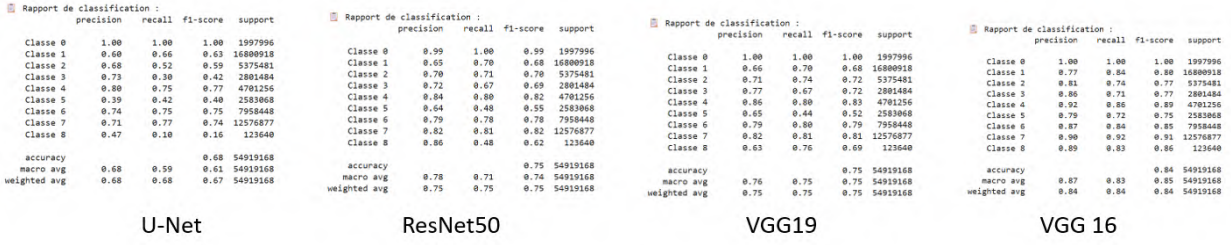


FIGURE 4.16 – Rapport de classication

La Figure 4.16 montre les performances de nos quatre modèles sur les données de test, où l'on trouve les statistiques pour chaque type de cellules (9 types : classe 0 jusqu'à 8) ainsi que la moyenne des performances pour chaque modèle.

4.4.3 Comparaison avec notre modèle CNN :

Le tableau comparatif des métriques révèle que VGG16 obtient les meilleurs résultats en termes de perte et de Dice sur les deux ensembles (train et validation), ce qui témoigne de sa capacité à extraire efficacement les caractéristiques pertinentes. ResNet50, bien que légèrement moins performant en termes de perte, surpasse les autres modèles en Dice de validation (0.8043), confirmant son aptitude à segmenter des structures complexes. VGG19, malgré sa profondeur, affiche des performances moindres, probablement dues à un surapprentissage ou à des hyperparamètres non optimisés.

TABLE 4.6 – Comparaison des performances des modèles de segmentation

Modèle	**Loss**	**Val_Loss**	**Dice**	**Val_Dice**	**Accuracy**	**Val_Accuracy**
VGG16	0.12888	0.15493	0.8854	0.7322	0.9673	0.7668
VGG19	0.33574	0.5548	0.7449	0.6824	0.8228	0.7265
U-Net Simple	0.5574	0.6690	0.7493	0.7887	0.8293	0.7657
ResNet50	0.32044	0.5824	0.8705	0.8043	0.8745	0.7402

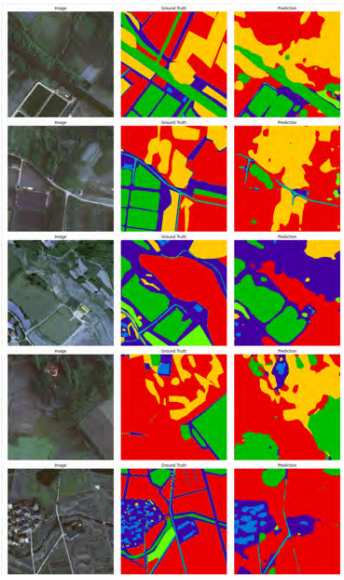


FIGURE 4.17 – Les résultats de notre segmentation sémantique sur les données de validation de U-Net.

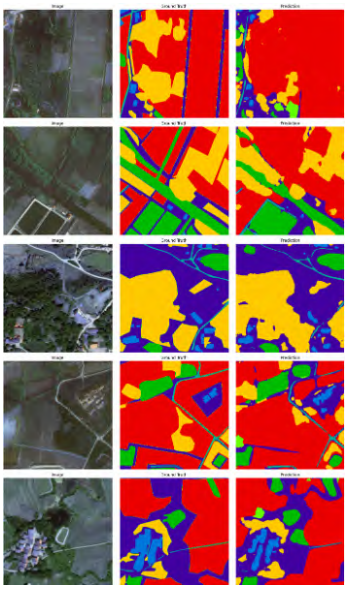


FIGURE 4.18 – Les résultats de notre segmentation sémantique sur les données de validation de VGG16.

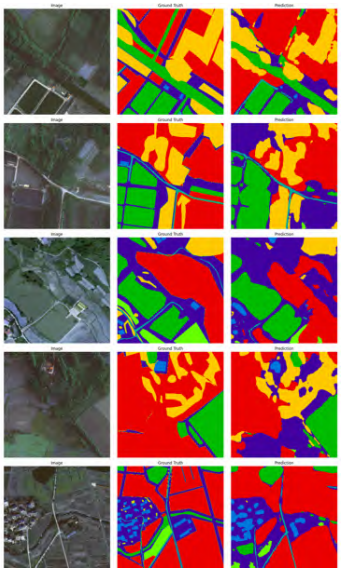


FIGURE 4.19 – Les résultats de notre segmentation sémantique sur les données de validation de VGG19.

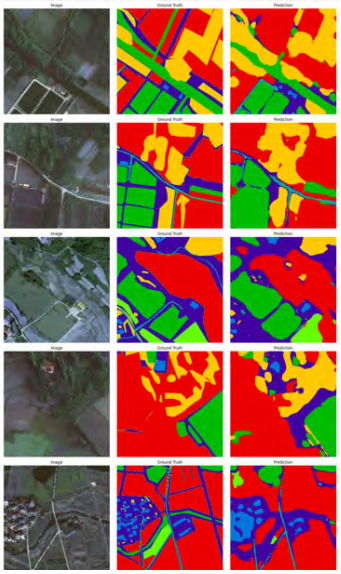


FIGURE 4.20 – Les résultats de notre segmentation sémantique sur les données de validation de ResNet.

4.5 Les défis auxquels nous avons été confrontés

La réalisation de ce travail de recherche a été jalonnée de nombreux défis techniques et méthodologiques.

4.5.1 Préparation des données

L'acquisition d'un jeu de données pertinent et la qualité des annotations associées ont constitué un défi majeur. Trouver un *dataset* adapté à notre problématique spécifique n'a pas été immédiat.

4.5.2 Déséquilibre des classes

Le déséquilibre marqué entre les différentes classes a significativement affecté les performances des modèles. Les classes rares étaient souvent mal prédites, entraînant une diminution de la qualité globale de la segmentation. Pour atténuer cet effet, des techniques telles que l'augmentation de données ciblée et la pondération de la fonction de perte ont été explorées.

4.5.3 Choix et réglage des architectures

L'adaptation des architectures (U-Net, VGG16, VGG19, ResNet50) aux caractéristiques des données a nécessité de nombreux ajustements. Le réglage des hyperparamètres, comme le taux d'apprentissage, le nombre d'époques ou la taille du *batch*, s'est révélé complexe, notamment pour les modèles profonds qui demandaient un entraînement plus long et plus sensible aux paramètres.

4.5.4 Accès aux ressources sur Google Colab

L'utilisation de Google Colab, bien qu'elle ait facilité l'accès à un environnement GPU, a également imposé des restrictions importantes, notamment en termes de durée de

session, de disponibilité des ressources, et de capacité mémoire. Ces limitations ont restreint l'exploration de configurations plus ambitieuses, comme l'ajustement fin des hyperparamètres, l'utilisation de réseaux plus profonds ou l'intégration de techniques d'apprentissage avancées telles que le *learning rate scheduler*, le *fine-tuning* ou le *ensemble learning*.

4.5.5 Optimisation du compromis biais-variance

Trouver un équilibre entre sous-apprentissage et surapprentissage s'est avéré complexe. Certaines architectures simples (comme U-Net) peinaient à capturer des détails fins, tandis que les architectures plus complexes risquaient de sur-ajuster les données d'entraînement. Des techniques telles que le *dropout*, l'*early stopping* ou la régularisation L2 ont été mobilisées pour améliorer la généralisation.

4.6 Interface graphique :

L'interface de SatSegment met en avant une plateforme intuitive et structurée, conçue pour l'analyse géospatiale à l'aide de techniques avancées d'intelligence artificielle. Elle présente clairement ses principales fonctionnalités telles que l'analyse automatique d'images, la sécurité des données et la génération de rapports intelligents. L'usage de technologies modernes côté frontend (React, TypeScript) et backend (FastAPI, PostgreSQL), ainsi que de bibliothèques IA comme PyTorch et TensorFlow, souligne la robustesse et l'efficacité du système. Cette organisation permet une interaction fluide entre l'utilisateur et les capacités du modèle de segmentation.

Premièrement on lance l'interface :

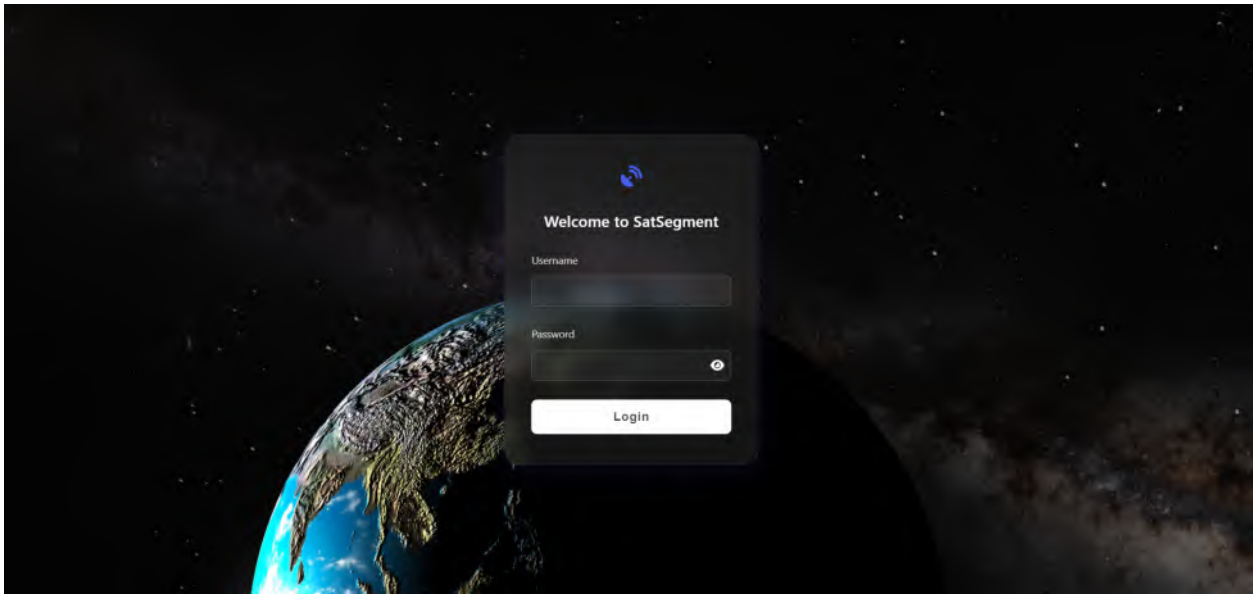


FIGURE 4.21 – connexion à la plateforme SatSegment

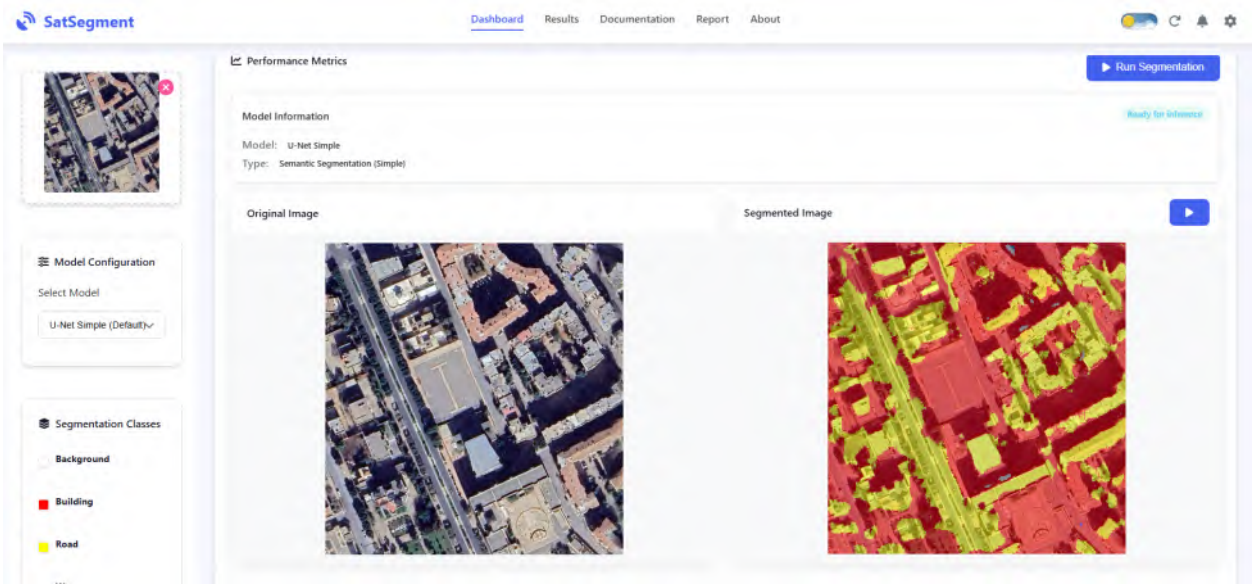


FIGURE 4.22 – Interface graphique pour la segmentation sémantique d'images satellites à l'aide du modèle U-Net

La figure 4.22 ci-dessus illustre une interface graphique interactive nommée SatSegment, développée pour la tâche de segmentation sémantique d'images satellites. Cette interface permet de charger une image satellite (à gauche) et d'exécuter une segmentation automatique (à droite) à l'aide d'un modèle de type U-Net Simple, un réseau de neurones convolutionnels largement utilisé pour les tâches de segmentation pixel à pixel.

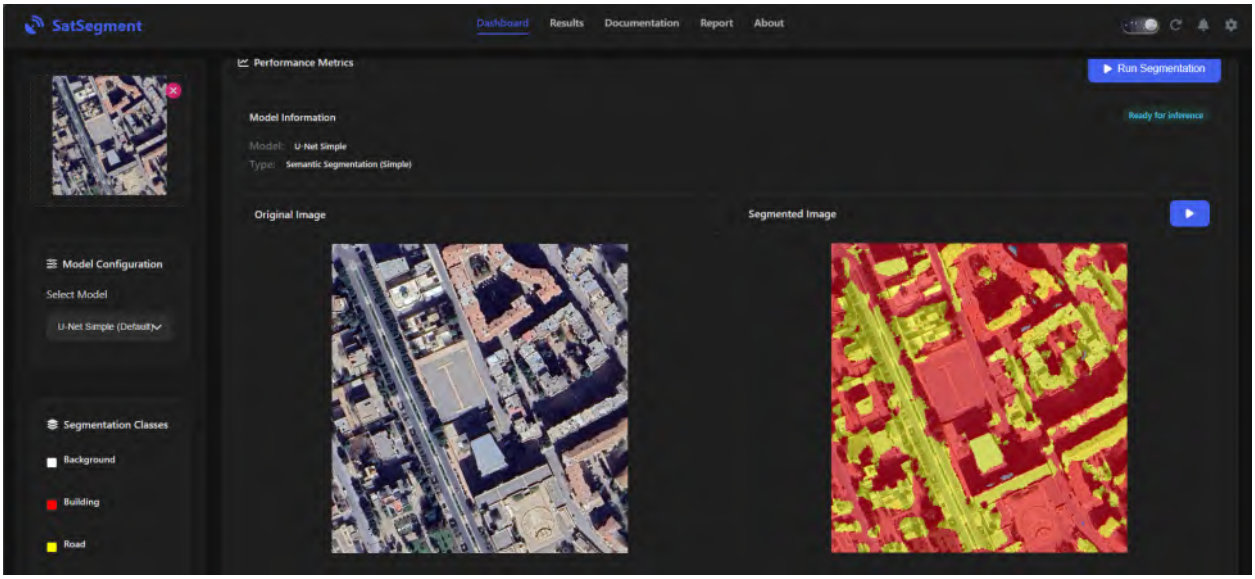


FIGURE 4.23 – Interface graphique pour la segmentation sémantique d'images satellites à l'aide du modèle U-Net mode sombre

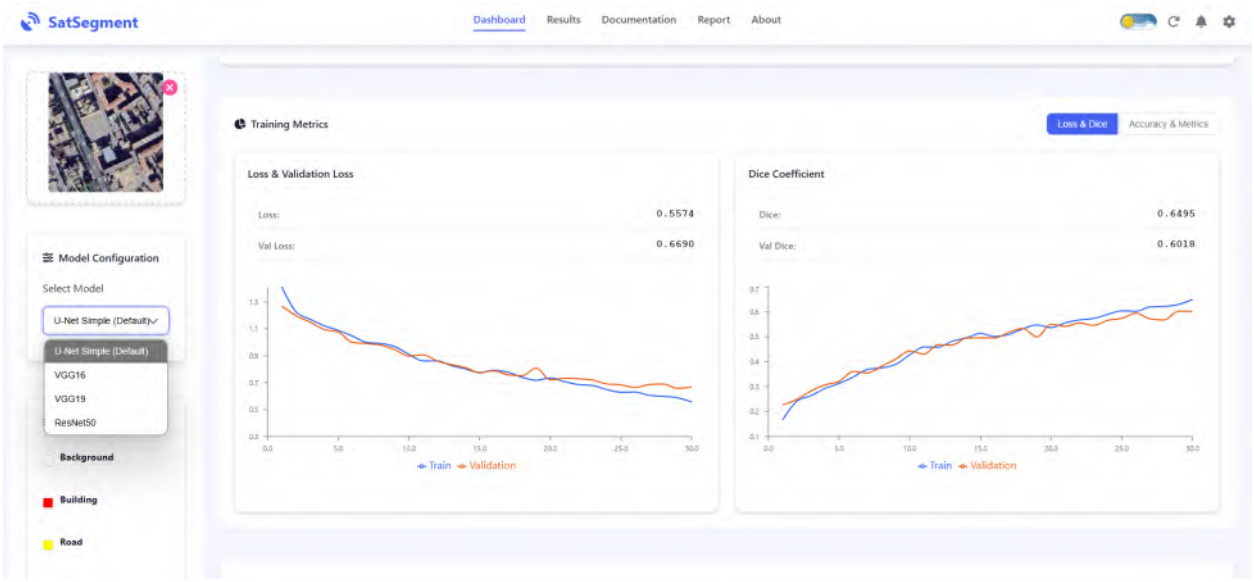


FIGURE 4.24 – Interface graphique de SatSegment affichant les performances d'entraînement des variantes de U-Net

La figure 4.24 fournit des outils interactifs permettant de visualiser les métriques d'entraînement, d'évaluer les performances de différents modèles U-Net Simple (Default),U-Net + VGG16,U-Net + VGG19,U-Net + ResNet50
À gauche : la fonction de perte (Loss) pour l'entraînement (bleu) et la validation (orange).
À droite : le coefficient de Dice, qui mesure le chevauchement entre les masques prédits et les masques réels.

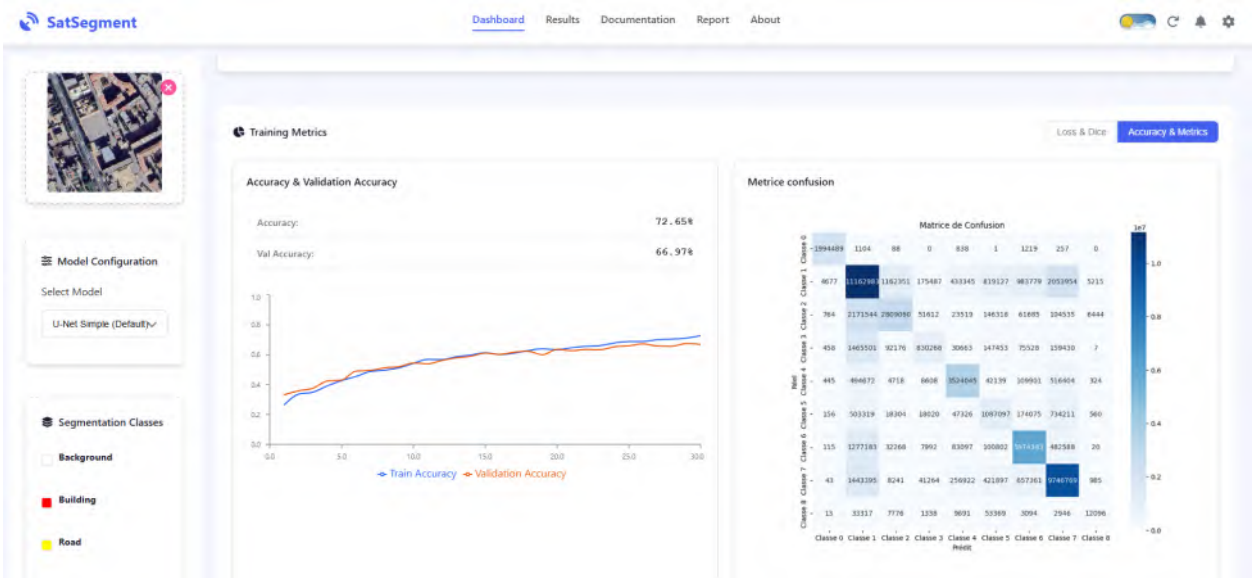


FIGURE 4.25 – Évaluation des performances du modèle U-Net Simple – Précision et Matrice de Confusion

La figure 4.25 présente une autre vue de l'interface de l'outil SatSegment, dédiée à l'analyse quantitative des performances du modèle U-Net Simple (Default) appliqué à la segmentation sémantique d'images satellitaires. Cette vue se concentre sur deux aspects principaux : la précision globale et la matrice de confusion.

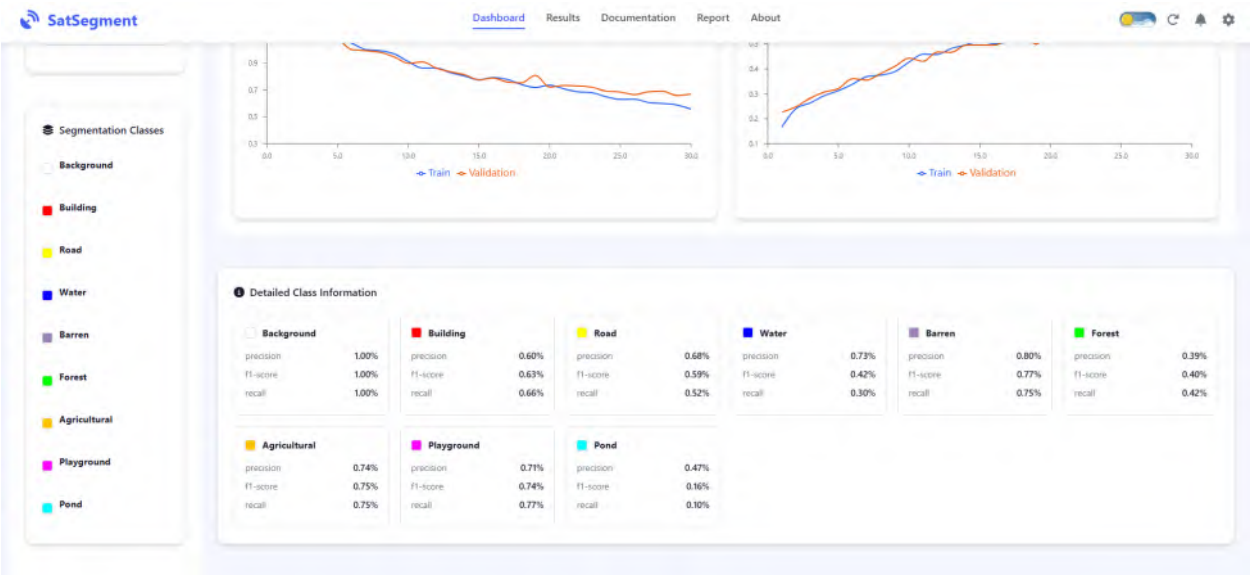


FIGURE 4.26 – Évaluation détaillée par classe des performances du modèle U-Net Simple sur des images satellitaires

La figure 4.26 affiche les métriques de performance par classe pour le modèle de segmentation U-Net Simple, tel qu'intégrées dans l'interface de SatSegment. Les trois métriques évaluées sont :Precision Recall F1-score

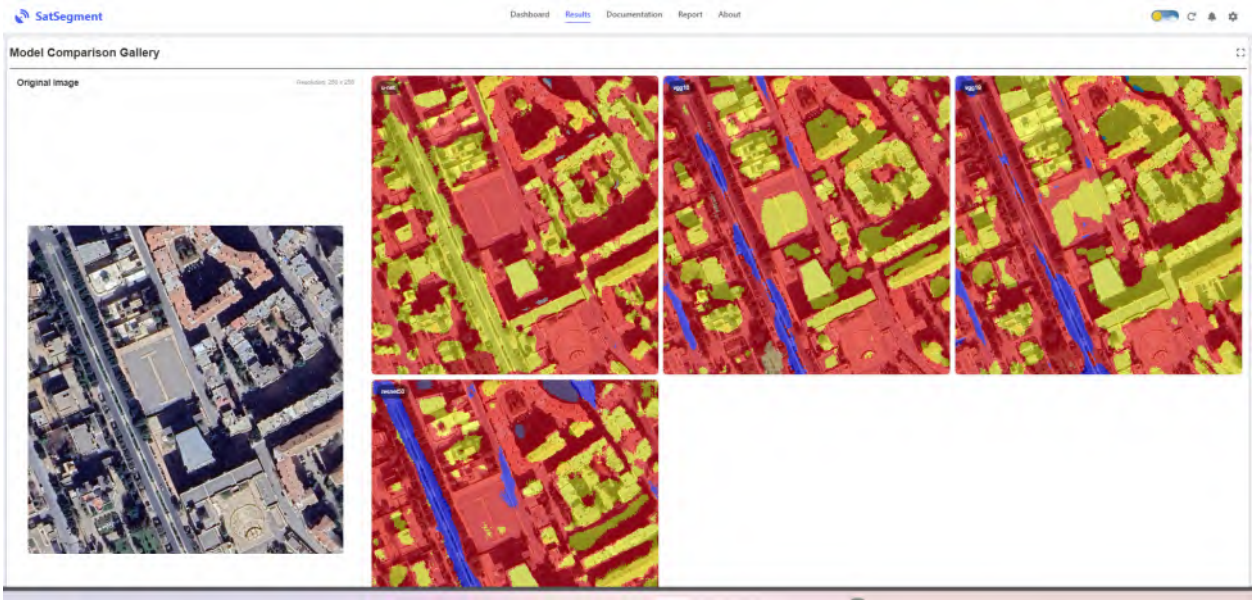


FIGURE 4.27 – Comparaison visuelle des résultats de segmentation obtenus par différents modèles U-Net sur une image satellitaire urbaine

La figure 4.27 illustre une comparaison visuelle des cartes de segmentation générées par quatre variantes du modèle U-Net, appliquées à une image satellite (résolution 256x256) d'un environnement

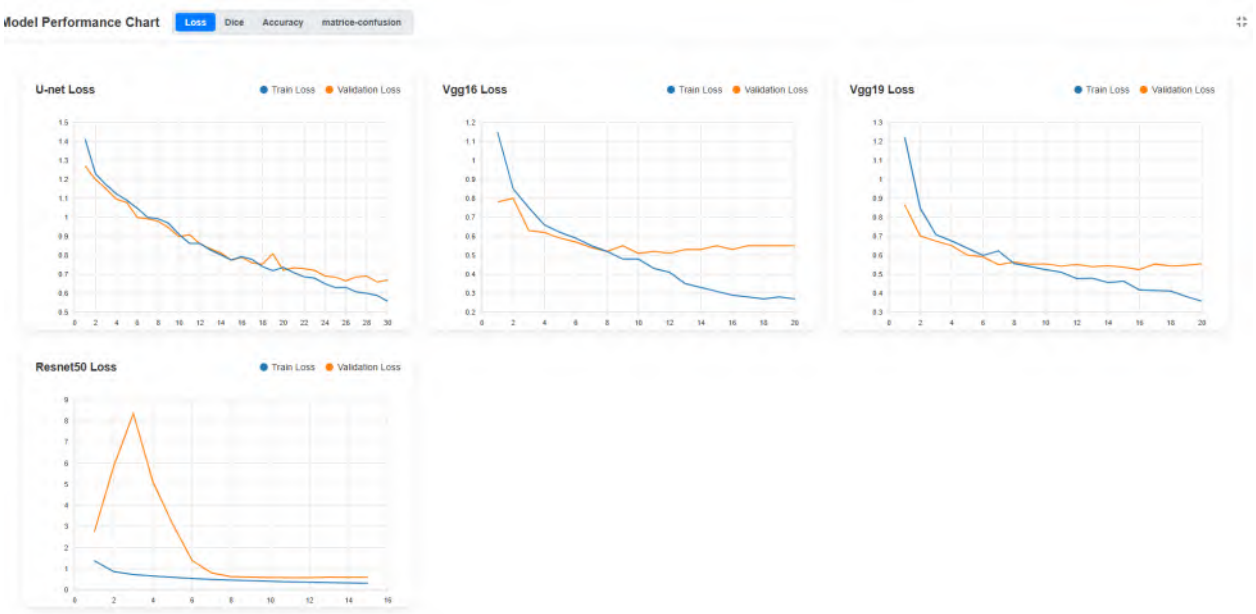


FIGURE 4.28 – Comparaison des courbes de perte (Loss) entre différents modèles U-Net selon le backbone utilisé

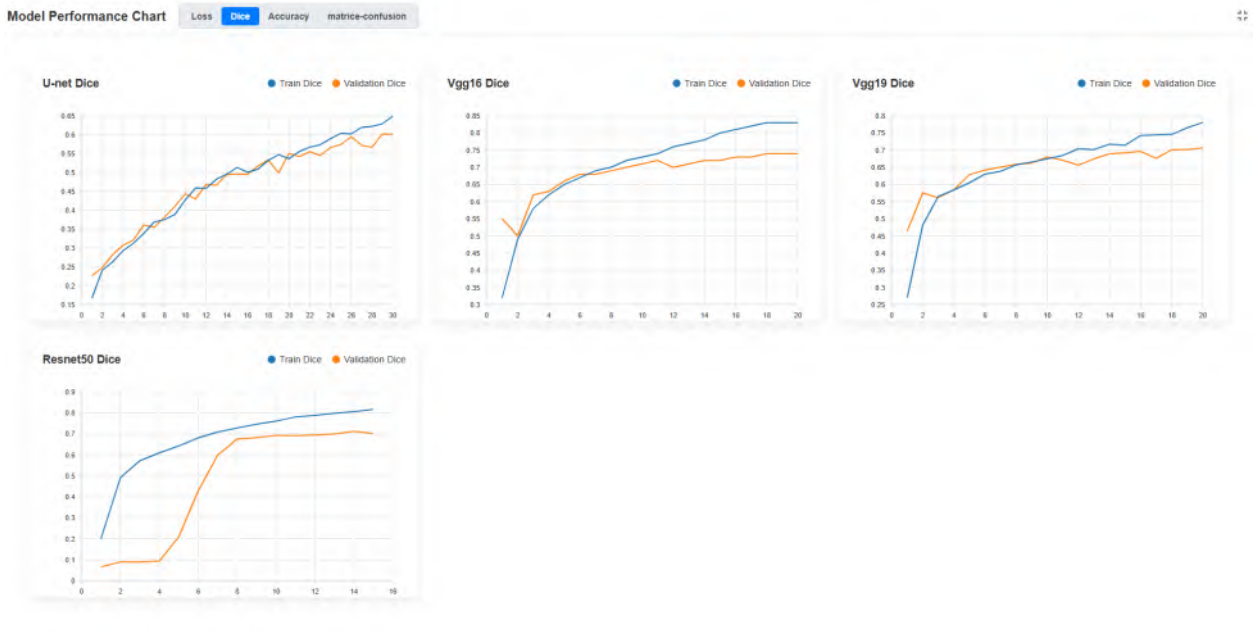


FIGURE 4.29 – Comparaison des courbes de perte (dice) entre différents modèles U-Net selon le backbone utilisé

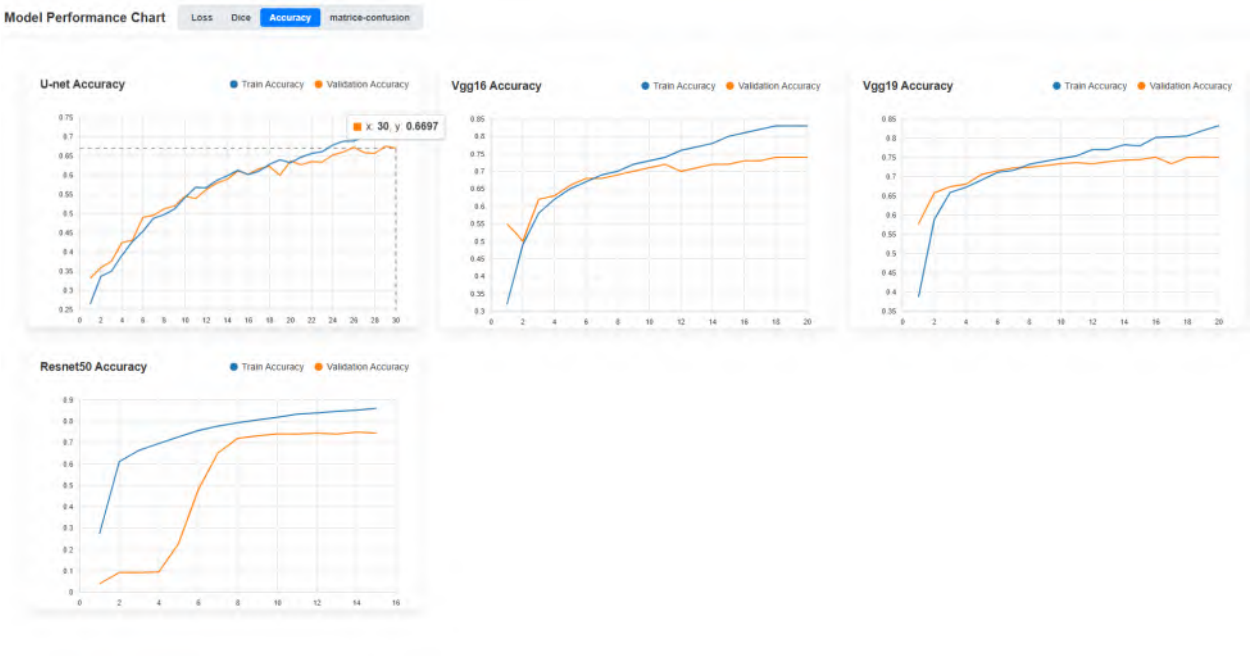


FIGURE 4.30 – Comparaison des courbes de perte (Accuracy) entre différents modèles U-Net selon le backbone utilisé

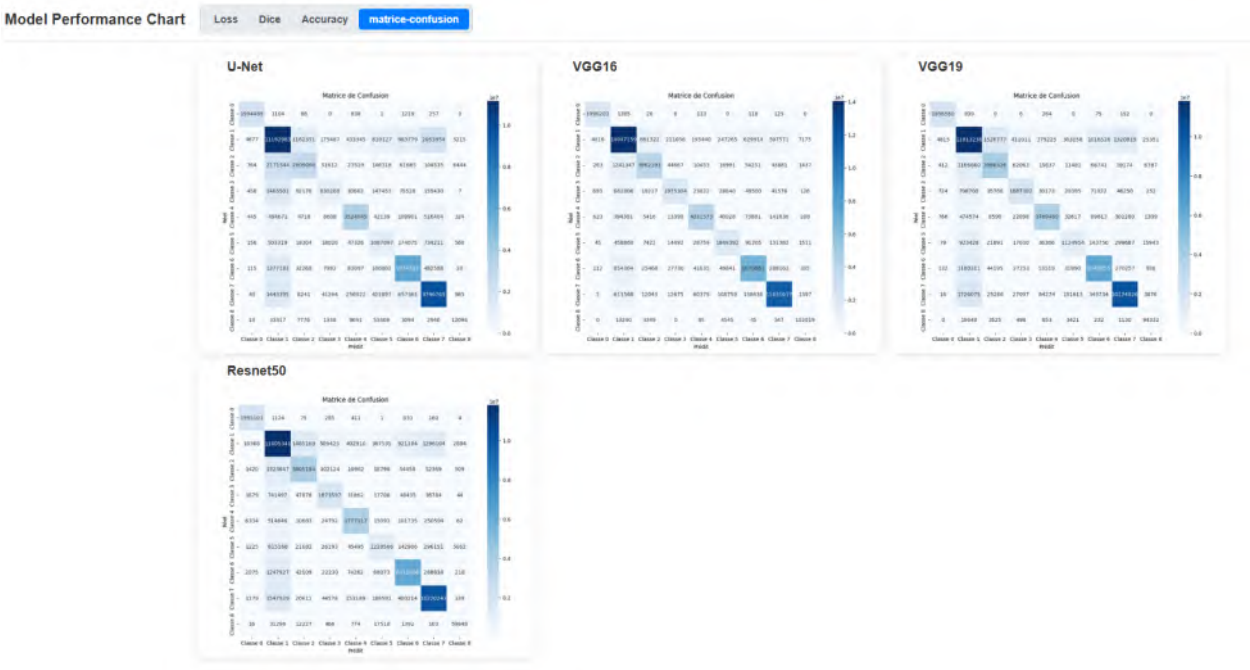


FIGURE 4.31 – Comparaison des courbes de perte (matrice-confusion) entre différents modèles U-Net selon le backbone utilisé

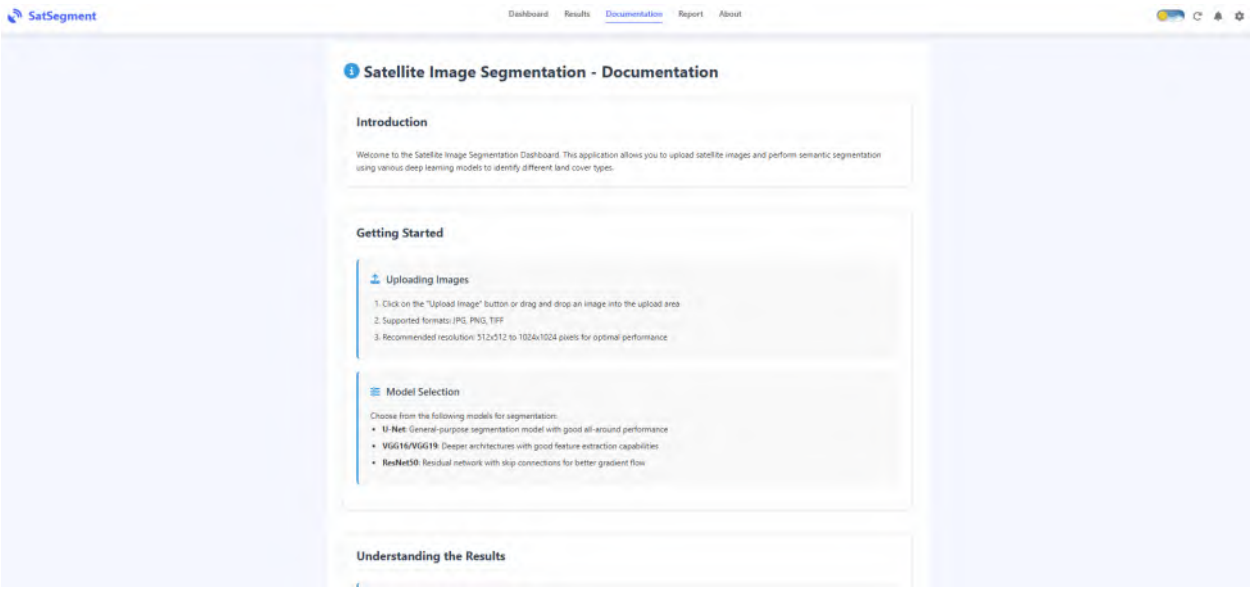


FIGURE 4.32 – Documentation de l'interface SatSegment pour la segmentation sémantique d'images satellites

La figure 4.32 affichée constitue une composante essentielle de l'outil SatSegment, en fournissant à l'utilisateur une introduction claire, un guide de démarrage rapide et une explication des modèles intégrés pour la segmentation sémantique d'images satellites

FIGURE 4.33 – Rapport de notre projet de fin d'études sur Segmentation Sémantique des Images Satellitaires pour la Surveillance Environnementale

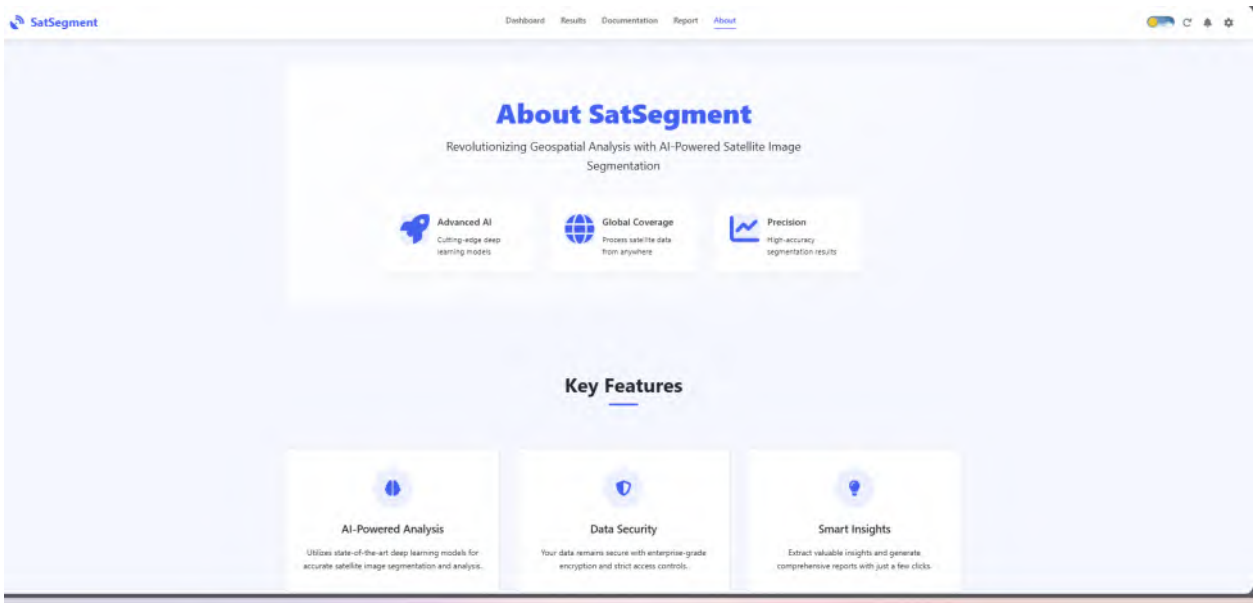


FIGURE 4.34 – présentation de la plateforme SatSegment

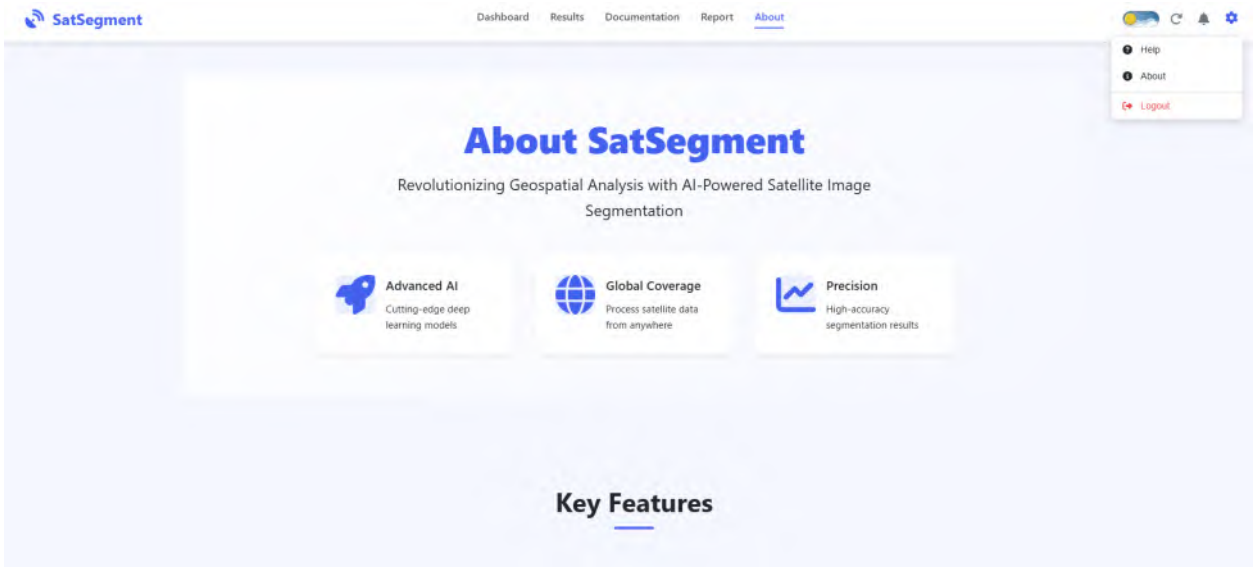


FIGURE 4.35 – Accès aux paramètres de l'utilisateur avec l'option de déconnexion

Conclusion :

Ce mémoire a porté sur l'étude, la conception et l'évaluation de modèles de segmentation sémantique appliqués aux images satellitaires, dans un contexte de surveillance environnementale et stratégique. Face à des problématiques majeures telles que la désertification, les incendies de forêts ou les marées noires, la capacité à analyser rapidement et efficacement les images de télédétection représente un enjeu scientifique, technologique et sociétal crucial.

À travers une revue approfondie des méthodes classiques de traitement d'images, nous avons mis en évidence les limites de ces approches face à la complexité, la variabilité et l'hétérogénéité des scènes capturées par les satellites. L'émergence du Deep Learning, et en particulier des réseaux de neurones convolutifs (CNN), a permis d'introduire des solutions plus performantes, capables d'apprendre des représentations hiérarchiques riches et robustes.

Notre contribution s'est traduite par la mise en œuvre et la comparaison de plusieurs architectures de segmentation sémantique de type encoder-decoder, notamment U-Net, U-Net couplé à VGG16/VGG19, et U-Net avec ResNet50 comme backbone. Ces modèles ont été entraînés et évalués sur un jeu de données représentatif d'images satellitaires multiclasse.

Les résultats obtenus, mesurés à l'aide d'indicateurs rigoureux tels que le *Dice coefficient*, la *précision moyenne* ou les *matrices de confusion*, ont confirmé la supériorité des modèles basés sur l'apprentissage profond. En particulier, U-Net avec VGG16 s'est distingué par sa capacité à préserver les détails spatiaux tout en assurant une segmentation cohérente à l'échelle globale.

En conclusion, ce travail confirme la pertinence et l'efficacité des méthodes de Deep Learning pour la segmentation sémantique d'images satellitaires. Il ouvre également la voie à de futures recherches visant à intégrer des approches encore plus puissantes, telles que les architectures Transformers, l'apprentissage auto-supervisé ou les modèles multi-modaux, pour une meilleure exploitation des données d'observation de la Terre.

Conclusion générale

Dans un contexte mondial marqué par des pressions environnementales croissantes et une nécessité grandissante de surveillance territoriale efficace, les technologies de traitement d'images satellitaires s'affirment comme un levier stratégique incontournable. Le présent mémoire s'est inscrit dans cette dynamique, en s'attachant à étudier, implémenter et évaluer différentes architectures de réseaux neuronaux convolutifs (CNN) appliquées à la tâche de segmentation sémantique d'images satellitaires. L'objectif principal consistait à comparer ces architectures pour identifier celles offrant les meilleures performances dans un cadre de surveillance environnementale automatisée.

La première partie de ce travail s'est concentrée sur les fondements de la télédétection spatiale, en explorant les mécanismes d'acquisition d'images via capteurs satellitaires et les propriétés physiques du rayonnement électromagnétique. Cette introduction était essentielle pour comprendre les spécificités des images satellitaires, notamment leur haute dimensionnalité, leur variabilité spectrale et leur caractère bruité. En effet, ces images diffèrent largement des images classiques utilisées en vision par ordinateur, ce qui nécessite des méthodes de traitement adaptées.

Dans un second temps, nous avons abordé les techniques classiques de segmentation d'images. Bien que historiquement pertinentes, ces approches telles que le seuillage, la détection de contours ou encore la classification des pixels se sont révélées limitées face aux exigences de précision et de robustesse posées par les images satellites. Leur incapacité à gérer efficacement la complexité contextuelle, les formes irrégulières ou les variations d'échelle nous a conduit à justifier le recours à des approches plus modernes, notamment fondées sur l'apprentissage profond.

Le troisième chapitre a ainsi été consacré à une présentation détaillée du Deep Learning et des architectures CNN. Nous y avons étudié les mécanismes internes des réseaux de neurones convolutifs, les différentes fonctions d'activation, les techniques de normalisation et de régularisation, ainsi que les stratégies d'optimisation. Un accent particulier a été mis sur les architectures avancées comme U-Net, SegNet, DeepLabV3+, et U-Net++, reconnues pour leur efficacité dans les tâches de segmentation sémantique. Le transfert learning, quant à lui, s'est révélé comme une méthode stratégique pour exploiter des modèles pré-entraînés sur de larges corpus, puis les adapter à des bases de données spécifiques, comme celles issues de la télédétection.

Dans la quatrième partie, nous avons mis en œuvre plusieurs modèles de segmentation à l'aide de la bibliothèque TensorFlow/Keras dans un environnement Colab. Les architectures implémentées comprenaient le modèle U-Net de base, ainsi que ses variantes combinées avec VGG16, VGG19 et ResNet50. Les modèles ont été entraînés sur une base annotée contenant plusieurs classes sémantiques (eau, végétation, zone urbaine, sol nu, etc.), et évalués selon des métriques standards : Dice coefficient, Accuracy, et matrices de confusion.

Les résultats expérimentaux obtenus ont mis en évidence des différences significatives entre les architectures. Le modèle U-Net simple a offert des performances honorables, mais les versions utilisant des backbones pré-entraînés (notamment U-Net + ResNet50 et U-Net + VGG16) ont démontré une capacité supérieure à généraliser, surtout dans des scènes complexes ou à forte hétérogénéité visuelle. Le modèle U-Net + ResNet50 a atteint un Dice coefficient de 0,8043, témoignant d'une segmentation fine et efficace. À l'inverse, les performances de U-Net + VGG19 ont été légèrement inférieures, ce qui suggère que la profondeur du réseau n'est pas systématiquement synonyme de performance, en particulier lorsque les données d'entraînement sont limitées.

Outre les résultats chiffrés, une interface graphique nommée SatSegment a été développée afin de rendre le système accessible à des utilisateurs non spécialistes. Cette interface permet de charger une image satellite, d'afficher les prédictions du modèle choisi et de visualiser les performances associées. Cette dimension logicielle apporte une plus-value pratique significative au travail, en ouvrant la voie à une intégration dans des systèmes réels de surveillance ou d'aide à la décision.

Malgré les réussites obtenues, ce travail n'est pas exempt de limites. En premier lieu, la taille réduite du jeu de données et le déséquilibre entre les classes ont pu influencer négativement l'apprentissage, en particulier pour les classes minoritaires. Des techniques telles que l'augmentation de données ou l'usage de pondérations de classes auraient pu être davantage exploitées pour atténuer ces biais. Ensuite, les ressources limitées de Google Colab ont freiné l'expérimentation de modèles plus complexes, notamment ceux intégrant des mécanismes d'attention ou des architectures basées sur les transformers, qui sont aujourd'hui à la pointe du domaine.

De plus, les résultats quantitatifs ne fournissent qu'un aperçu partiel de la qualité de la segmentation. Une évaluation qualitative approfondie, impliquant des experts en télédétection, aurait permis de valider l'utilité pratique des segmentations générées. Enfin, l'adaptation du système à des données multi-temporelles ou hyperspectrales pourtant essentielles dans de nombreuses applications réelles reste à explorer.

À l'issue de ce travail, plusieurs perspectives d'amélioration se dessinent. Il serait pertinent d'intégrer des architectures hybrides CNN-transformers telles que SegFormer ou Swin-Unet, qui ont récemment montré leur efficacité pour capturer à la fois des caractéristiques locales et globales. Par ailleurs, des stratégies semi-supervisées ou auto-supervisées pourraient permettre de tirer parti d'images non annotées, souvent abondantes mais sous-exploitées. Sur le plan applicatif, le déploiement du modèle en edge computing c'est-à-dire sur des systèmes embarqués pourrait favoriser un traitement temps réel dans des contextes critiques, comme les catastrophes naturelles ou la surveillance militaire.

En somme, ce mémoire a permis de démontrer que l'intégration de l'intelligence artificielle, et plus particulièrement du deep learning, dans le traitement d'images satellitaires est non seulement possible, mais nécessaire. Elle répond à un besoin urgent de compréhension automatisée de l'environnement, et ouvre la voie à des systèmes d'observation intelligents, évolutifs et autonomes. En conciliant rigueur scientifique, implémentation technique et pertinence applicative, ce travail constitue une contribution concrète à la transformation numérique de la télédétection.

Bibliographie

[1] Université d'Antananarivo & CIRAD. (n.d.). *BIG n°3 – Initiation à la télédétection*. Bulletin de l'information géographique.

[2] Auly, T. (2018, 23 avril). *Télédétection*. Dynamiques Environnementales Info. https://doi.org/10.58079/nhtu

[3] Kergomard, C. (1990). *La télédétection aérospatiale : une introduction*. Cours de télédétection, École Normale Supérieure Paris.

[4] Université Paris 1 Panthéon-Sorbonne. (n.d.). *Rayonnement électromagnétique*. https://e-cours.univ-paris1.fr/modules/uved/envcal/html/rayonnement/1-rayonnement-electromagnetique/index.html

[5] Annecca, G. (s.d.). *Capteurs : Conditionnement des signaux* [Module de Capteur, Licence PRO]. http://dptgeii.iutsd.univ-lorraine.fr/cours/lpsarii/IM/Cours/Capteurs.pdf

[6] Tamim, A. (2015). *Segmentation et classification des images satellitaires* (Thèse de doctorat, Université Mohammed V, Rabat). https://inria.hal.science/tel01242495

[7] Landgrebe, D. (2003). Signal theory methods in multispectral remote sensing. Wiley-Interscience.

[8] André, M. (1987). *Introduction aux techniques de traitement d'images*. Paris : Éditions Eyrolles.

[9] André, M. (1987). Introduction aux techniques de traitement d'images. Paris : Éditions Eyrolles.

[10] Rodrigo, R. (2010). Introduction to digital image processing. CRC Press.

[11] Sonka, M., Hlavac, V., Boyle, R. Image Processing, Analysis, and Machine Vision.

[12] Thyagarajan, K. S. (2005). Digital image processing with application to digital cinema. Routledge. — Chapiter 1

[13] HARALICK, R. M., SHAPIRO, L. G. Image Segmentation Techniques..

[14]]Szeliski, R. (2022). Computer vision : Algorithms and applications (2nd ed.). Springer. https ://szeliski.org/Book/

[15] Winzenrieth, R., Claude, I., Hobatho, M. C., Pouletaut, P., Sebag, G. (2003). Comparaison de deux méthodes de segmentation par contours actifs : les snakes et les level sets pour la segmentation d'IRM de hanche

[16] Gambino, O., Vitabile, S., Re, G. L., La Tona, G., Librizzi, S., Pirrone, R., ... Midiri, M. Automatic Volumetric Liver Segmentation Using Texture Based Region Growing.

[17] Nejim, Z., Navarro, L., Morin, C., Badel, P. (2023). Quantitative analysis of second harmonic generated images of collagen fibers : a review. Research on Biomedical Engineering, 39(1), 273-295.

[18] Otsu, N. (1979). A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man, and Cybernetics, 9(1), 62–66. https ://-doi.org/10.1109/TSMC.1979.4310076

[19] Bradley, D., Roth, G. (2007). Adaptive thresholding using the integral image. Journal of Graphics Tools, 12(2), 13–21. https ://-doi.org/10.1080/2151237X.2007.10129236

[20] Jain, A. K. (2010). Data clustering : 50 years beyond K-meansq. Pattern Recognition Letters, 31, 651-666.

[21]]Richards, J. A., Jia, X. (2006). Remote sensing digital image analysis : An introduction (4th ed.). Springer. https ://doi.org/10.1007/3-540-29711-1

[22] Schmidhuber, J. (2015). Deep learning. Scholarpedia, 10(11), 32832.

[23] Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y. (2016). Deep learning (Vol. 1, No. 2). Cambridge : MIT press.

[24] Benomar M.L, « Deep learning », Cours du module Deep Learning, Université Belhadj Bouchaib – Ain Témouchent, 2024.

[25] LeCun, Y., Bengio, Y. Hinton, G. Deep learning. Nature 521, 436–444 (2015). https ://doi.org/10.1038/nature14539.

[26] Jagtap, A. D., Karniadakis, G. E. (2023). How important are activation functions in regression and classification ? A survey, performance comparison, and future directions. Journal of Machine Learning for Modeling and Computing, 4(1).

[27] Orlianges, J. C., Allegret, O., El Moustakime, Y., Crunteanu STANESCU, A., Carrizales Juarez, R. (2024). Retour vers le perceptron-fabrication d'un neurone synthétique à base de composants électroniques analogiques simples. Les journées de l'interdisciplinarité.

[28] IanGoodfellow, Yoshua Bengio, Aaron Courville, Deep Learning, MITPress, 2016.

[29] Jagtap, A. D., & Karniadakis, G. E. (2023). How important are activation functions in regression and classification ? A survey, performance comparison, and future directions. *Journal of Machine Learning for Modeling and Computing, 4* (1).

[30] Bengio, Y., Goodfellow, I., & Courville, A. (2017). *Deep learning* (Vol. 1, pp. 23–24). Cambridge, MA : MIT Press. Chapitre 9 : Convolutional Networks.

[31] Bengio, Y., Goodfellow, I., & Courville, A. (2017). *Deep learning* (Vol. 1, pp. 23–24). Cambridge, MA : MIT Press. Chapitre 10 : Sequence Modeling.

[32] Bengio, Y., Goodfellow, I., & Courville, A. (2017). *Deep learning* (Vol. 1, pp. 23–24). Cambridge, MA : MIT Press. Chapitre 15 : Linear Factor Models and Auto-Encoders.

[33] Ioffe, S., & Szegedy, C. (2015, June). Batch normalization : Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning* (pp. 448–456).

[34] Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer Normalization. *stat*, *1050*, 21.

[35] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout : A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research, 15*, 1929–1958.

[36] Géron, A. (2017). *Deep Learning avec TensorFlow*. Chapitres 11 et 12 : Regularization and Tuning.

[37] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data, 6*(1), 1–48.

[38] Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., & Liu, C. (2018). A survey on deep transfer learning. *arXiv preprint arXiv :1808.01974*.

[39] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net : Convolutional networks for biomedical image segmentation. In *MICCAI 2015, Part III* (pp. 234–241). Springer.

[40] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net : Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (Vol. 9351, pp. 234–241). Springer. https://doi.org/10.1007/978-3-319-24574-4_28

[41] Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). SegNet : A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(12), 2481–2495. https://doi.org/10.1109/TPAMI.2016.2644615

[42] Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv :1409.1556*.

[43] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *CVPR* (pp. 770–778). https://arxiv.org/abs/1512.03385

[44] Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., & Liang, J. (2018). UNet++ : A nested U-Net architecture for medical image segmentation. *arXiv preprint arXiv :1807.10165*.

[45] Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV* (pp. 801–818). https://arxiv.org/abs/1802.02611

[46] Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., & Terzopoulos, D. (2021). Image segmentation using deep learning : A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. https://arxiv.org/abs/2001.05566

[47] Sudre, C. H., Li, W., Vercauteren, T., Ourselin, S., & Cardoso, M. J. (2017). Generalised Dice overlap as a deep learning loss function for highly unbalanced segmentations. *arXiv preprint arXiv :1707.03237v3*. https://arxiv.org/abs/1707.03237

[48] Bengio, Y., Goodfellow, I., & Courville, A. (2017). *Deep learning*. Cambridge, MA : MIT Press. Chapitre 8 : Optimization.

[49] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. Chapitre 5 : Machine Learning Basics. https://www.deeplearningbook.org/

[50] Powers, D. M. W. (2011). Evaluation : From precision, recall and F-measure to ROC and correlation. *Journal of Machine Learning Technologies. arXiv :2010.16061*

[51] Géron, A. (2022). *Deep Learning avec Scikit-Learn, Keras et TensorFlow* (3e éd.). Pearson. Chapitre 3.

[52] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. Chapitre 5.

[53] Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*. https://doi.org/10.1371/journal.pone.0118432

[54] Rahman, M. A., & Wang, Y. (2016). Optimizing Intersection-Over-Union in deep neural networks for image segmentation. *arXiv preprint arXiv :1608.05471*.

[55] Rezatofighi, H., et al. (2019). Generalized Intersection over Union : A metric and a loss for bounding box regression. In *CVPR 2019*.

[56] Google. (n.d.). Colaboratory FAQ : Resource Limits. https://research.google.com/colaboratory/faq.html#resourcelimits

[57] Python Software Foundation. (2024). The Python Tutorial — Python 3.13.4. https://docs.python.org/3/tutorial/

[58] https://www.tensorflow.org/api_docs

[59] https://keras.io/guides/

[60] https://numpy.org/doc/

[61] https://docs.opencv.org/

[62] https://matplotlib.org/stable/contents.html

[63] https://scikit-learn.org/

[64] https://docs.python.org/3/library/glob.html

[65] https://docs.python.org/3/library/os.html

[66] https://www.tensorflow.org/api_docs/python/tf/keras/utils/to_categorical