République Algérienne Démocratique Et Populaire
Ministère de L'enseignement Supérieur et de la Recherche Scientifique
Université 20 août 1955-Skikda
Département d'informatique

MÉMOIRE

En vue de l'obtention du diplôme de
Master en informatique
Spécialité : Génie Logiciel Avancée et application

Thème

La reconstruction 3D en utilisant Autoencoder et U-Net

Réalisé par :
Rehai Hanane
Mallem Ikram

Encadré par :
Bougamouza Fateh

juin 2025

Remerciements

« Tout d'abord, Alhamdulillah, nous remercions Allah pour Sa grandeur et pour nous avoir donné l'opportunité, la force et le courage de réaliser ce travail. Par la suite, Nous tenons à remercier spécialement notre encadreur, Dr. Fateh Bougamouza, qui nous a honorés en acceptant notre encadrement et pour les efforts, les directives et les conseils qu'elle nous a donnés pour rédiger le mémoire de fin d'études. Notre gratitude cordiale est infinie envers les professeurs du département d'informatique qui ont contribué à notre formation au cours des cinq dernières années d'études. Grâce à eux, nous avons construit le premier chemin vers l'avenir à partir de leurs connaissances et de leur expérience scientifique. Nous remercions également les membres du jury qui ont accepté d'évaluer notre travail, et nous attendons avec impatience leurs conseils supplémentaires. Enfin, nos sincères remerciements seraient incomplets sans attribuer une partie à nos familles et à nos amis solidaires. Un merci spécial à nos parents, qui nous ont soutenus émotionnellement et nous ont aidés spirituellement tout au long de notre vie. »

Dédicase

À mes chers parents

Je dédie ce mémoire à mes chers parents qui ont été toujours à mes côtés et m'ont toujours soutenu tout au long de ces longues années d'études. En signe de reconnaissance, qu'ils trouvent ici l'expression de ma profonde gratitude pour tout ce qu'ils ont consenti d'efforts et de moyens pour me voir réussir dans mes études.

À toute ma famille

Aucune expression ne saurait exprimer mon respect et ma considération pour votre soutien et encouragements. Je vous dédie ce travail en reconnaissance de l'amour que vous m'offrez quotidiennement et votre bonté exceptionnelle. Que Dieu le Tout Puissant vous garde et vous procure santé et bonheur.

À mon binôme

Je souhaite exprimer ma sincère reconnaissance envers mon binôme pour son assistance inestimable tout au long de ce travail.

- Hanane Rehai

Dédicace

À mes chers parents

À ma chère mère et à mon cher père, Pour leur amour inconditionnel, leur soutien sans faille, leurs sacrifices silencieux et leurs prières constantes. Vous êtes la source première de ma force et de ma persévérance.

À ma sœur, Pour son affection, son encouragement et sa présence rassurante tout au long de ce parcours.

À mon encadrant(e), Pour ses conseils judicieux, sa disponibilité, son encadrement rigoureux et sa confiance en mes capacités.

À tous les enseignants qui m'ont transmis le savoir depuis mes premières années d'école jusqu'à l'université, Je vous rends hommage pour votre dévouement et votre rôle essentiel dans la construction de mon parcours académique.

À ma famille élargie, Pour leur soutien moral et leurs encouragements constants.

À mes amis fidèles, Pour leur amitié sincère, leur entraide précieuse, et les moments inoubliables partagés durant cette aventure.

Ce mémoire vous est dédié, avec toute ma gratitude et mon respect.

- Ikram Mallem

Résumé

Ce travail traite de la problématique de la reconstruction 3D à partir d'images 2D, en examinant les approches classiques telles que la stéréovision et la Structure from Motion, tout en mettant en évidence leurs principales limitations : sensibilité au bruit, dépendance à la texture et complexité de calcul élevée. Pour surmonter ces contraintes, nous proposons un modèle basé sur un autoencodeur inspiré de l'architecture U-Net, intégrant des connexions de saut (skip-connections). Le modèle est entraîné sur des images extraites de vidéos, puis testé via un processus de reconstruction itératif, visant à révéler progressivement la structure de profondeur cachée. Ce mécanisme permet d'évaluer la capacité du modèle à préserver les caractéristiques essentielles de la scène au fil des reconstructions. Les résultats initiaux sont prometteurs, bien qu'une perte progressive d'information soit observée, comme l'indiquent la dégradation des métriques SSIM et PSNR. Ces observations confirment le potentiel des autoencodeurs avec skip-connections pour extraire des représentations structurées, tout en ouvrant la voie à des améliorations futures à travers des architectures plus robustes (GANs, Transformers), susceptibles d'améliorer la stabilité et la généralisation du modèle.

Mots-clés : Reconstruction 3D, Autoencodeur, U-Net, Connexions de saut, SSIM, PSNR, Deep Learning, Structure from Motion, Stéréovision.

Abstract

This work addresses the problem of 3D reconstruction from 2D images by exploring classical approaches such as stereo vision and Structure from Motion, while highlighting their main limitations : noise sensitivity, texture dependency, and high computational complexity. To overcome these challenges, we propose a model based on an autoencoder inspired by the UNet architecture, incorporating skip-connections. The model is trained on images extracted from videos, then evaluated through an iterative reconstruction process aimed at progressively revealing the hidden depth structure. This mechanism assesses the model's ability to preserve the essential features of the scene across successive reconstructions. Initial results are promising, although a gradual loss of information is observed, as indicated by the decline in SSIM and PSNR metrics. These findings confirm the potential of skip-connection autoencoders to extract structured representations, while paving the way for future improvements using more robust architectures such as GANs and Transformers, which could enhance model stability and generalization.

Keywords : 3D Reconstruction, Autoencoder, U-Net, Skip Connections, SSIM, PSNR, Deep Learning, Structure from Motion, Stereo Vision.

ملخص

يتناول هذا العمل مشكلة إعادة البناء ثلاثي الأبعاد انطلاقًا من صور ثنائية الأبعاد، من خلال دراسة الأساليب التقليدية مثل الرؤية الستيريو وتقنية Motion، from Structure مع تسليط الضوء على أبرز قيود هذه الأساليب، والمتمثلة في الحساسية للضوضاء، والاعتماد الكبير على وجود نسيج في الصورة، والتعقيد الحسابي العالي. لتجاوز هذه التحديات، نقترح نموذجًا يعتمد على الـ Autoencoder مستوحًى من بنية U-Net، ويعتمد على توصيلات تخطٍ (Skip-connections). يتم تدريب النموذج على صور مستخرجة من مقاطع فيديو، ثم يتم اختباره عبر عملية إعادة بناء تكرارية تهدف إلى الكشف التدريجي عن البنية العميقة المخفية للمشهد. تُستخدم هذه الآلية لقياس قدرة النموذج على الحفاظ على الخصائص الأساسية للمشهد عبر مراحل إعادة البناء المتعاقبة. وقد أظهرت النتائج الأولية أداءً واعدًا، رغم تسجيل فقدان تدريجي للمعلومات، كما يتضح من تراجع مؤشري SSIM وPSNR. تؤكد هذه النتائج قدرة Autoencoders المدعّمة بـ Skip-connections على استخراج تمثيلات هيكلية، وتفتح المجال لتحسينات مستقبلية من خلال اعتماد معماريات أكثر قوة، مثلGANs وTransformers، لتعزيز استقرار النموذج ورفع قدرته على التعميم.

الكلمات المفتاحية : إعادة البناء ثلاثي الأبعاد، التشفير التلقائي، U-Net، التوصيلات التخطية، SSIM، PSNR، التعلم العميق، Motion، from Structure الرؤية الستيريو.

Table des Matière

Remerciements I

Dédicase II

Résumé IV

Abstract V

ملخص VI

Table des figure 5

Liste des tableaux 7

Liste des abréviations 8

Introduction Générale 9

1 Fondements de l'Intelligence Artificielle et du Deep Learning 12

1.1 Introduction : 12

1.2 L'intelligence Artificielle 12

1.2.1 Types et Typologies d'Intelligence Artificielle 13

1.2.1.1 L'Intelligence Artificielle faible 13

1.2.1.2 L'Intelligence Artificielle Générale 13

1.2.1.3 Intelligence Artificielle Forte 13

1.3 L'apprentissage automatique(Machine Learning) 13

1.3.1 Types d'apprentissage automatique 14
1.3.1.1 Apprentissage supervisé 14
1.3.1.2 Apprentissage non supervisé 15
1.3.1.3 Apprentissage semi-supervisé 15
1.3.1.4 Apprentissage par renforcement 16
1.4 Apprentissage Profond 17
1.4.1 Historique 17
1.4.2 Les Réseaux de Neurones 18
1.4.2.1 Réseaux de neurones convolutifs : 19
1.4.2.2 Réseaux de neurones récurrents : 23
1.4.2.3 Réseaux Long Short-Term Memory 24
1.4.2.4 Unité Récurrente à Portes : 25
1.4.2.5 Les Auto-encodeurs 25
1.4.2.6 Les réseaux antagonistes génératifs 27
1.4.2.7 Les Transformers 27
1.5 Conclusion 28

2 De la 2D à la 3D : Approches Classiques et Approches par Deep Learning 30

2.1 Introduction 30
2.2 Les approches classiques de la conversion 2D vers 3D 30
2.2.1 Estimation de la Profondeur 30
2.2.2 La stéréovision 31
2.2.2.1 La stéréovision dynamique 32
2.2.2.2 La stéréovision statique 33
2.2.3 La structure from motion 33
2.2.4 Multi-View Stereo 34
2.2.4.1 Méthodes traditionnelles de Multi-View Stereo 34
2.2.5 Avantages et inconvénients des méthodes classiques 35
2.3 La conversion 2D vers 3D utilisant le Deep Learning 35
2.3.1 Estimation monoculaire de la profondeur par Deep Learning 36

2.3.2 Utilisation des CNN pour la reconstruction 3D à partir d'images 2D . . 36
2.3.2.1 Génération de la carte de profondeur 37
2.3.2.2 Projection en 3D à partir de la carte de profondeur 38
2.3.2.3 Rôle exact des CNN dans cette tâche 39
2.3.2.4 Avantages de l'utilisation des CNN pour la reconstruction 3D 39
2.3.3 Autoencodeurs dans la Reconstruction 3D 39
2.3.3.1 Les Skip Connections dans les Autoencodeurs 40
2.3.3.2 Applications récentes des autoencodeurs dans la reconstruction 3D 40
2.3.4 Les Réseaux Antagonistes Génératifs dans la Reconstruction 3D . . . 41
2.3.5 État de L'Art sur La conversion 2D vers 3D 42
2.4 Conclusion 43

3 Implémentation d'un Système de Reconstruction 3D avec U-Net 45

3.1 Introduction 45
3.2 Description du Système 45
3.3 Architecture Générale du Système 46
3.4 Préparation de la Base de Données 46
3.4.1 Collecte et extraction des données brutes 46
3.4.2 Prétraitement des images et constitution des paires d'entraînement . . . 47
3.4.3 Séparation des données en ensembles d'entraînement et de test 48
3.4.4 Sauvegarde des données préparées 48
3.5 Présentation du modèle Autoencoder classique 48
3.5.1 Architecture détaillée du modèle 49
3.5.1.1 Analyse des résultats 50
3.5.1.2 Calcul des métriques SSIM et PSNR 51
3.5.1.3 Analyse des indices SSIM et PSNR 51
3.5.1.4 Visualisation graphique 52
3.5.2 conclusion 52
3.6 Présentation du modèle Autoencoder avec U-Net 53

3.6.1 Objectif du modèle : 53
3.6.2 Principe des Skip Connections : 53
3.6.3 Présentation des variantes architecturales testées 53
3.6.4 Analyse comparative des variantes 54
3.6.5 Architecture détaillée du modèle retenu 54
3.7 Entraînement du modèle 57
3.8 Test et Génération des Reconstructions 58
3.8.1 Calcul des métriques SSIM et PSNR 59
3.8.1.1 Visualisation graphique 59
3.8.1.2 Tableau récapitulatif 60
3.8.1.3 Analyse détaillée des tendances du SSIM et du PSNR 61
3.9 Génération des vidéos de reconstruction itérative 61
3.10 Proposition d'amélioration 62
3.11 Outils d'implémentation 62
3.11.1 Colab 62
3.11.2 Google Drive 63
3.11.3 Les bibliotheques utilisées 63
3.11.3.1 TensorFlow / Keras 63
3.11.3.2 NumPy 63
3.11.3.3 OpenCV (cv2) 64
3.11.3.4 Matplotlib 64
3.11.3.5 os 64
3.11.3.6 tqdm 65
3.12 Conclusion 65

Conclusion Générale **66**

References **67**

Table des figure

1.1 L'Intelligence artificiel , Machine Learning ,Deep Learning 14

1.2 Un ensemble de formation labellisé pour l'apprentissage supervisé (par exemple, la classification de spam) 15

1.3 Régression 15

1.4 Apprentissage par renforcement 16

1.5 Structure d'un réseau de neurone à couche unique (perceptron) 18

1.6 Exemple d'architecture CNN 20

1.7 Exemple de principe du filtre convolutionnel 21

1.8 Principe de la fonction ReLul 21

1.9 Exemple de principe du Pooling 22

1.10 Principe de la couche entièrement connectée (fc) 22

1.11 Exemple montrant l'étiquette codée de la couche de sortie CNN 23

1.12 Architecture de RNN 24

1.13 Architecture d'un auto-Encodeur 26

2.1 Architecture d'un réseau profond pour l'estimation monoculaire de la profondeur. 31

2.2 Principe de la stéréovision : deux caméras capturent une même scène sous des angles légèrement différents pour reconstruire la profondeur par triangulation. . 32

2.3 Diagramme illustrant le principe de la méthode Structure from Motion (SfM). . 33

2.4 Architecture d'un réseau CNN pour l'estimation de la profondeur à partir d'une image RGB unique. 37

2.5 Visualisation de la projection inverse d'une carte de profondeur en points 3D. . 38

2.6 Schéma de l'architecture d'un autoencodeur utilisé pour l'estimation de la profondeur 40

3.1 Architecture generale du système 46

3.2 Comparaison entre image original premiere reconstruction et derniere reconstruction (image 1) 50

3.3 Comparaison entre image original premiere reconstruction et derniere reconstruction (image 2) 50

3.4 Évolution du SSIM et du PSNR sur les 15 itérations(image 1) 52

3.5 Évolution du SSIM et du PSNR sur les 15 itérations(image 2) 52

3.6 Architecture complète de l'Autoencodeur avec connexions de saut 57

3.7 Comparaison entre image original premiere reconstruction et derniere reconstruction (image 1) 59

3.8 Comparaison entre image original premiere reconstruction et derniere reconstruction (image 2) 59

3.9 Évolution du SSIM et du PSNR sur les 15 itérations (image 1) 60

Liste des tableaux

3.1 Architecture complète de l'Autoencodeur Classique 49

3.2 Comparaison des variantes de l'architecture 54

3.3 Architecture complète du modèle Autoencoder avec Skip Connections 55

3.4 Valeurs moyennes du SSIM et du PSNR sur les 15 itérations pour la première image (image 1) 60

Liste des abréviations

AI Intelligence Artificielle

AGI General Artificial Intelligence

ML Apprentissage Automatique (Machine Learning)

DL Apprentissage Profond (Deep Learning)

RL Apprentissage par Renforcement(Reinforcement Learning)

RNA Réseaux de Neurones Artificiels

CNN Convolutional Neural Network

GAN Generative Adversarial Network

AE Autoencoder

RNA Réseaux de Neurones Artificiels

SfM Structure from Motion

ReLU Rectified Linear Unit

3D Trois Dimensions

2D Deux Dimensions

RNN Réseaux de neurones récurrents

2D Deux Dimensions

DNN Deep Neural Network

MVS Multi-View Stereo

SSIM Structural Similarity Index

DNN Deep Neural Network

LSTM Les réseaux Long Short-Term Memory

Introduction Générale

Au cours de la dernière décennie, le domaine de l'intelligence artificielle (IA) a connu une croissance exponentielle, bouleversant de nombreux secteurs scientifiques, industriels et sociaux. Parmi les sous-domaines les plus dynamiques de l'IA, l'apprentissage profond (deep learning) s'est imposé comme une technologie incontournable grâce à sa capacité à traiter de grandes quantités de données, à extraire automatiquement des caractéristiques complexes, et à produire des résultats d'une précision impressionnante.

Dans ce contexte technologique en pleine évolution, le domaine de la vision par ordinateur (computer vision) occupe une place stratégique, car il vise à permettre aux machines de percevoir, analyser et comprendre le contenu visuel du monde réel, qu'il s'agisse d'images ou de vidéos. L'un des défis majeurs dans ce domaine est la reconstruction tridimensionnelle (3D) à partir de données bidimensionnelles (2D), qui consiste à recréer la perception de la profondeur à partir d'images planes. Cette tâche est essentielle dans des applications telles que la réalité virtuelle, la robotique autonome, le diagnostic médical par imagerie, ou encore la cartographie intelligente.

Les méthodes classiques de reconstruction 3D, comme la stéréovision, la Structure-from-Motion (SfM), ou encore la vision multi-vues (MVS), bien qu'efficaces dans certains contextes contrôlés, présentent plusieurs limitations : forte sensibilité au bruit et aux variations de lumière, dépendance à des textures riches, nécessité d'une calibration rigoureuse, et complexité computationnelle élevée.

Ces contraintes ont ouvert la voie à de nouvelles approches basées sur les réseaux de neurones profonds, notamment les autoencodeurs, qui permettent d'apprendre à partir de données non structurées et de capturer des représentations latentes pertinentes. Grâce à leur capacité de compression et de reconstruction, ces architectures permettent de traiter l'information visuelle de manière plus robuste et flexible, même dans des environnements complexes.

Dans ce travail, nous proposons une méthode innovante qui s'appuie sur un autoencodeur convolutif amélioré par des connexions de saut (skip connections), inspiré de l'architecture U-Net. Cette approche permet de générer de manière itérative une séquence d'images simulant un déplacement en profondeur, à partir d'une seule image 2D en entrée. Le système est conçu pour maintenir la cohérence visuelle et structurelle entre les images générées tout en conservant la fidélité des détails à travers les itérations successives.

Pour mettre en œuvre cette idée, nous avons suivi une démarche expérimentale structurée, incluant la collecte et le prétraitement de données vidéo, la génération de paires d'images pour

l'apprentissage, la conception et l'entraînement du modèle autoencodeur, ainsi que l'évaluation des performances via des métriques objectives telles que SSIM (Structural Similarity Index) et PSNR (Peak Signal-to-Noise Ratio). Le système final a également été testé dans des scénarios de reconstruction répétée afin de mesurer la stabilité et la qualité visuelle au fil des générations.

Ce mémoire est structuré en trois chapitres qui abordent de manière progressive les aspects théoriques, méthodologiques et expérimentaux de notre travail.

Le premier chapitre est consacré à une introduction générale à l'intelligence artificielle, au machine learning, et plus particulièrement à l'apprentissage profond. Nous y présentons les concepts fondamentaux, les types de réseaux de neurones (CNN, autoencodeurs, GANs...), ainsi que leurs applications dans le domaine de la vision par ordinateur.

Le deuxième chapitre explore la problématique de la reconstruction 3D à partir d'images 2D. Nous y analysons les méthodes classiques, comme la stéréovision et la structure-from-motion, puis nous présentons les approches modernes basées sur le deep learning, notamment l'estimation de la profondeur à l'aide de réseaux convolutifs et autoencodeurs.

Le troisième chapitre est consacré à la conception, la mise en œuvre et l'expérimentation d'un système de reconstruction 3D basé sur un autoencodeur amélioré par des skip connections, inspiré de l'architecture U-Net. Ce modèle est entraîné de manière itérative afin de révéler progressivement la profondeur cachée d'une image 2D. Les performances sont évaluées à l'aide des métriques SSIM et PSNR, et des variantes architecturales ont été testées pour identifier la plus efficace. Ce chapitre inclut également une proposition d'amélioration du système ainsi qu'une évaluation des outils utilisés.

À travers cette étude, nous visons à démontrer l'efficacité des techniques de deep learning, en particulier des autoencodeurs avec skip connections, dans le contexte de la reconstruction 3D. Ce travail constitue une base solide pour de futures recherches orientées vers des reconstructions plus réalistes et robustes, en intégrant par exemple des GANs ou des techniques multimodales.

Chapitre 1

Fondements de l'Intelligence Artificielle et du Deep Learning

Chapitre 1

Fondements de l'Intelligence Artificielle et du Deep Learning

1.1 Introduction :

Ce premier chapitre vise à poser les fondements théoriques nécessaires à la compréhension du sujet de ce mémoire. Nous y introduisons les concepts clés de l'intelligence artificielle (IA), du machine learning (ML) et de l'apprentissage profond (DL), en mettant en lumière leurs inter-relations et leurs évolutions historiques. Ce cadre conceptuel est indispensable pour appréhender les techniques modernes de reconstruction 3D qui seront abordées dans les chapitres suivants. En particulier, l'attention est portée sur les architectures de réseaux de neurones, tels que les CNN, les RNN, les autoencodeurs, les GANs, et les transformeurs, qui constituent l'ossature des modèles récents utilisés en vision par ordinateur.

1.2 L'intelligence Artificielle

L'intelligence artificielle (IA) est une discipline scientifique qui se concentre sur la création de machines capables de réaliser des tâches nécessitant une forme d'intelligence humaine. Elle est définie par Marvin Minsky comme « la science qui consiste à faire faire aux machines des choses qui nécessiteraient de l'intelligence si elles étaient réalisées par des hommes ».

L'intelligence artificielle (IA) repose sur des fondements mathématiques et des algorithmes basés sur les probabilités et les statistiques, visant à reproduire certaines fonctions cognitives humaines telles que l'apprentissage et la prise de décision. Elle englobe plusieurs domaines (voir figure 1.1), dont le Machine Learning (ML), qui permet aux machines d'apprendre à partir de données pour améliorer leurs performances sans être explicitement programmées. À son tour, le Deep Learning (DL) constitue une sous-catégorie du Machine Learning. Il s'inspire du fonctionnement du cerveau humain à travers l'utilisation de réseaux de neurones artificiels profonds, permettant de traiter et d'interpréter de grandes quantités d'informations complexes.

1.2.1 Types et Typologies d'Intelligence Artificielle

Les différents niveaux d'Intelligence Artificielle (IA) peuvent être classés en trois catégories principales :

1.2.1.1 L'Intelligence Artificielle faible

L'IA faible (en anglais Narrow Artificial Intelligence), désigne des systèmes conçus pour accomplir une tâche spécifique sans conscience ni compréhension. Elle ne simule qu'un aspect limité de l'intelligence humaine. Par exemple, les assistants vocaux comme Siri ou Alexa, les filtres anti-spam des emails, ou encore les systèmes de recommandation relèvent de l'IA faible. Ces systèmes ne peuvent pas effectuer de tâches en dehors de ce pour quoi ils ont été programmés.

1.2.1.2 L'Intelligence Artificielle Générale

L'AGI(en anglais General Artificial Intelligence) fait référence à une forme hypothétique d'IA capable de comprendre, apprendre et appliquer ses connaissances de manière autonome dans divers domaines, à l'instar de l'intelligence humaine. Contrairement à l'IA faible, l'IAG n'est pas limitée à des tâches spécifiques et pourrait s'adapter à de nouvelles situations sans intervention humaine. Des projets comme Gato de DeepMind illustrent les efforts actuels pour développer des systèmes aux capacités plus générales .

1.2.1.3 Intelligence Artificielle Forte

L'IA forte(en anglais Super Artificial Intelligence)est une notion théorique d'une IA possédant une conscience de soi, des sentiments et une compréhension réelle, au-delà de la simple simulation de l'intelligence humaine. Elle implique que la machine ne se contente pas d'exécuter des tâches intelligentes, mais qu'elle comprend et éprouve des expériences subjectives. Actuellement, aucune IA forte n'existe, et son développement soulève des questions philosophiques et éthiques majeures .

1.3 L'apprentissage automatique(Machine Learning)

L'apprentissage automatique est un sous-domaine de IA qui s'intéresse à la construction d'algorithmes qui, pour être utile, s'appuie sur un ensemble d'exemples de certains phénomènes. Ces exemples peuvent provenir de la nature, être fabriqués à la main par l'homme ou générés par un autre algorithme. L'apprentissage automatique peut également être défini comme le processus de résolution d'un problème pratique par :
1) la collecte d'un ensemble de données
2) la construction algorithmique d'un modèle statistique basé sur cet ensemble de données.
Ce modèle statistique est censé être utilisé d'une manière ou d'une autre pour résoudre le

problème pratique. Une définition un peu plus générale :[L'apprentissage automatique est le domaine d'étude qui donne aux ordinateurs la possibilité d'apprendre sans être explicitement programmés.[1]

Et une autre plus technique : [on dit qu'un programme informatique apprend de l'expérience E en ce qui concerne une tâche T et une mesure de performance P, si sa performance sur T, mesurée par P, s'améliore avec l'expérience E.[2]

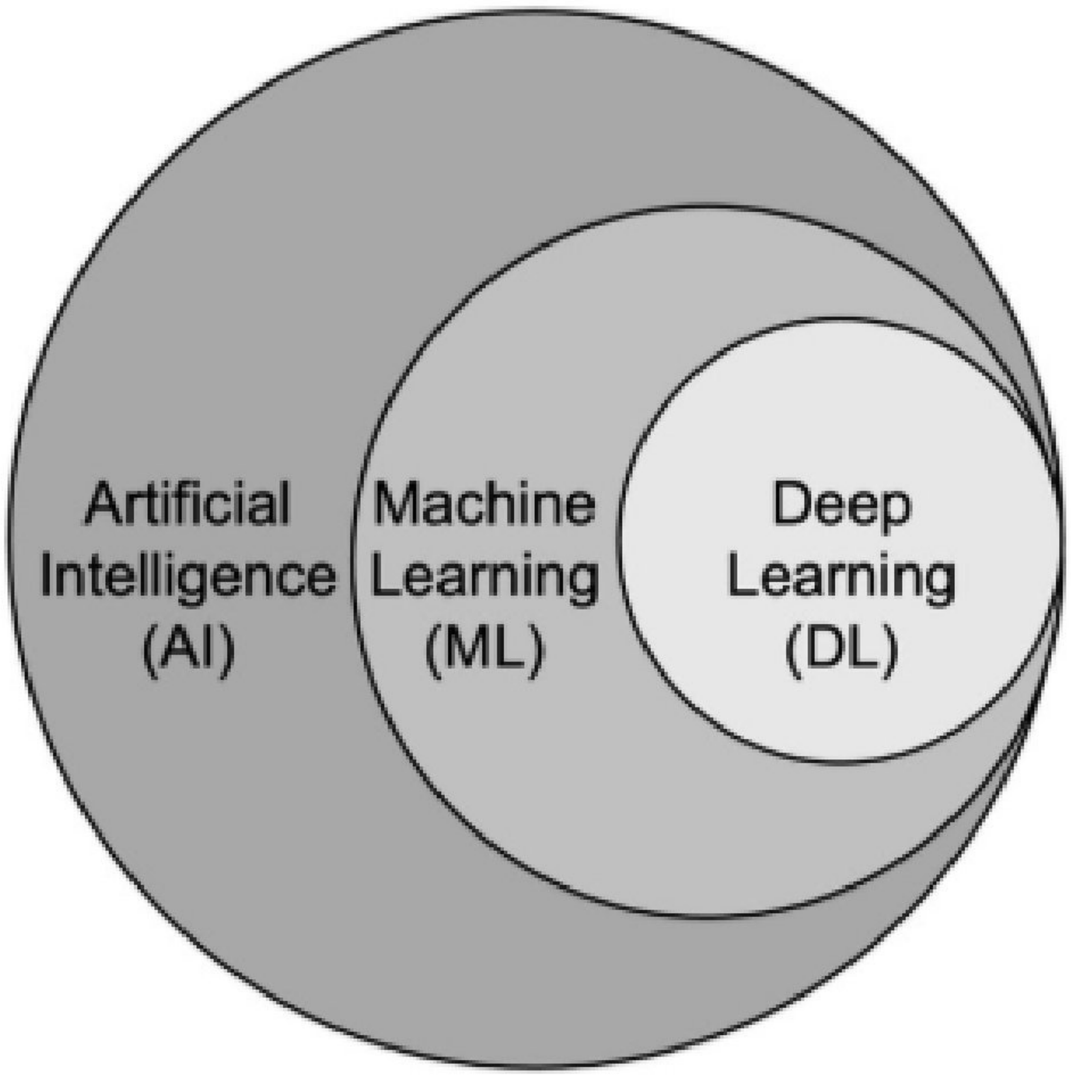


Fig. 1.1 : L'Intelligence artificiel , Machine Learning ,Deep Learning

1.3.1 Types d'apprentissage automatique

Il existe de nombreux types de systèmes d'apprentissage automatique. Dans ce qui suit, nous les classons selon qu'ils nécessitent ou non une supervision humaine (supervisés, non supervisés, semisupervisée, et apprentissage de renforcement).

1.3.1.1 Apprentissage supervisé

Dans l'apprentissage supervisé, les données de formation que l'on fournie à l'algorithme comprennent les solutions, appelées étiquètes (ou labels en anglais). Une tâche d'apprentissage supervisée typique est la classification. Le filtre anti-spam en est un bon exemple : il est formé avec de nombreux exemples d'e-mails avec leur classe (spam ou ham), et il doit apprendre à classer les nouveaux e-mails, comme présenté dans la figure 1.2

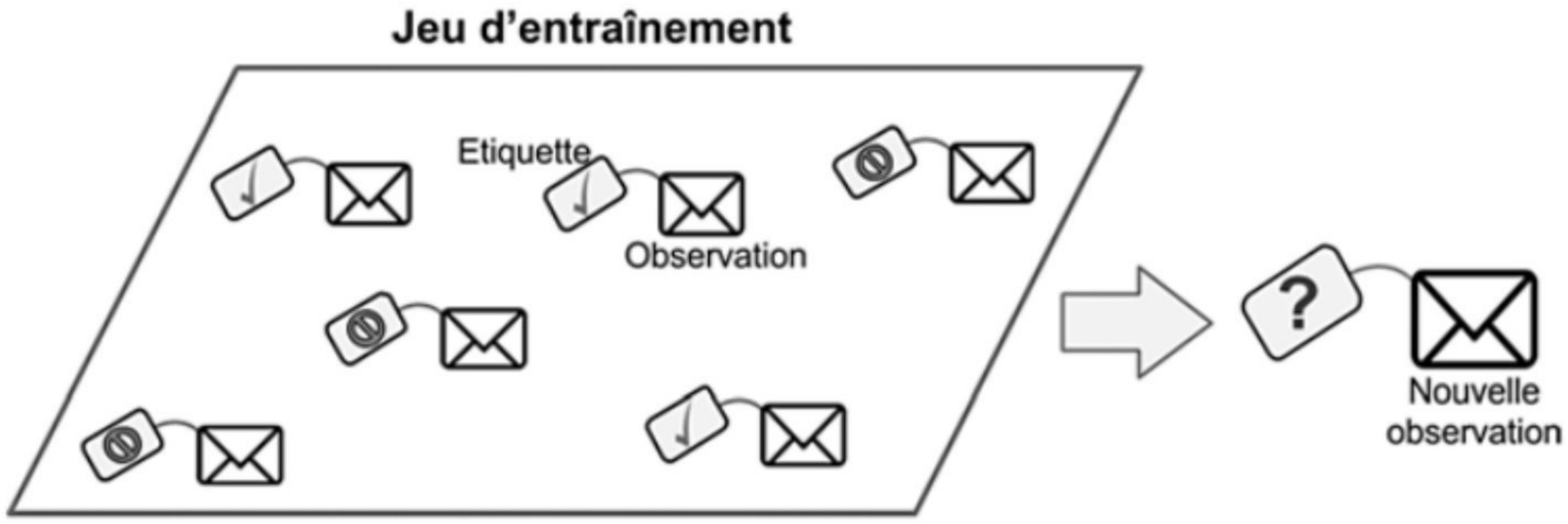


Fig. 1.2 : Un ensemble de formation labellisé pour l'apprentissage supervisé (par exemple, la classification de spam)

Uneautre tâches typique consiste à prédire une valeur numérique cible, tel que le prix d'une voiture, en fonction d'un ensemble de caractéristiques (kilométrage, âge, marque, etc.) appelées prédicteurs. Ce type de tâche est appelé régression. Pour entraîner le système, nous devons lui donner de nombreuxexemplesdevoitures, y compris leurs prédicteurs et leurs étiquettes (c'est à-dire leurs prix) la figure 1.3 décrit une tache de régression.

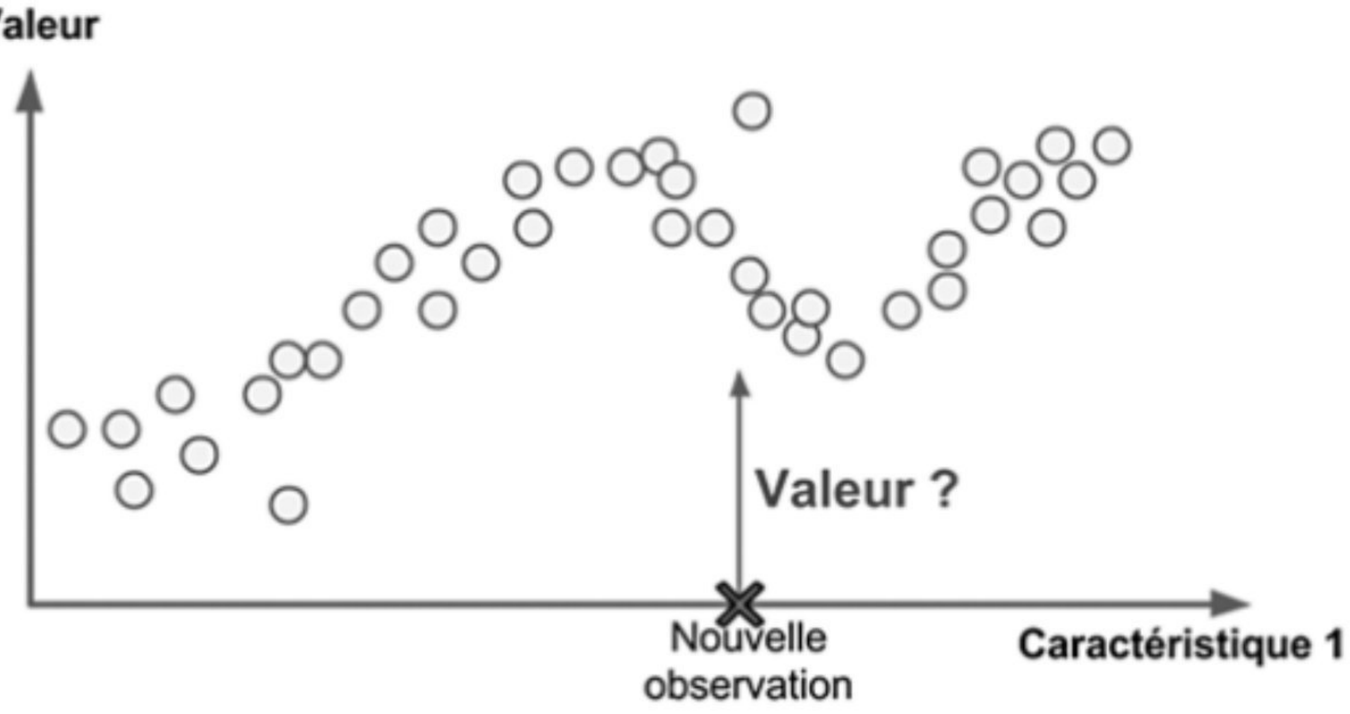


Fig. 1.3 : Régression
[3]

1.3.1.2 Apprentissage non supervisé

L'apprentissage non supervisé traite des données non étiquetées. Le modèle cherche à découvrir des structures ou des motifs cachés dans les données sans supervision explicite.

1.3.1.3 Apprentissage semi-supervisé

L'apprentissage semi-supervisé est une méthode d'apprentissage automatique qui exploite à la fois un petit ensemble de données étiquetées et un grand ensemble de données non étiquetées

pour entraîner un modèle. Elle vise à améliorer la performance de l'apprentissage supervisé tout en réduisant le coût lié à l'annotation manuelle des données. Cette approche s'avère particulièrement utile dans les domaines où l'obtention de labels précis est coûteuse ou difficile, comme en vision par ordinateur ou en traitement du langage naturel.[4]

1.3.1.4 Apprentissage par renforcement

L'apprentissage par renforcement (Reinforcement Learning, RL) est un paradigme d'apprentissage automatique dans lequel un agent apprend à prendre des décisions en interagissant avec un environnement. À chaque étape, l'agent observe l'état actuel de l'environnement, sélectionne une action, reçoit une récompense (positive ou négative) et observe le nouvel état résultant de cette action. L'objectif de l'agent est de maximiser la somme cumulative des récompenses reçues au fil du temps.[5]

Ce processus est souvent modélisé à l'aide de processus de décision de Markov (MDP), qui fournissent un cadre mathématique pour la prise de décision séquentielle dans des environnements incertains. Une caractéristique clé de l'apprentissage par renforcement est le compromis entre l'exploration (essayer de nouvelles actions pour découvrir leurs effets) et l'exploitation (utiliser les connaissances actuelles pour maximiser la récompense).

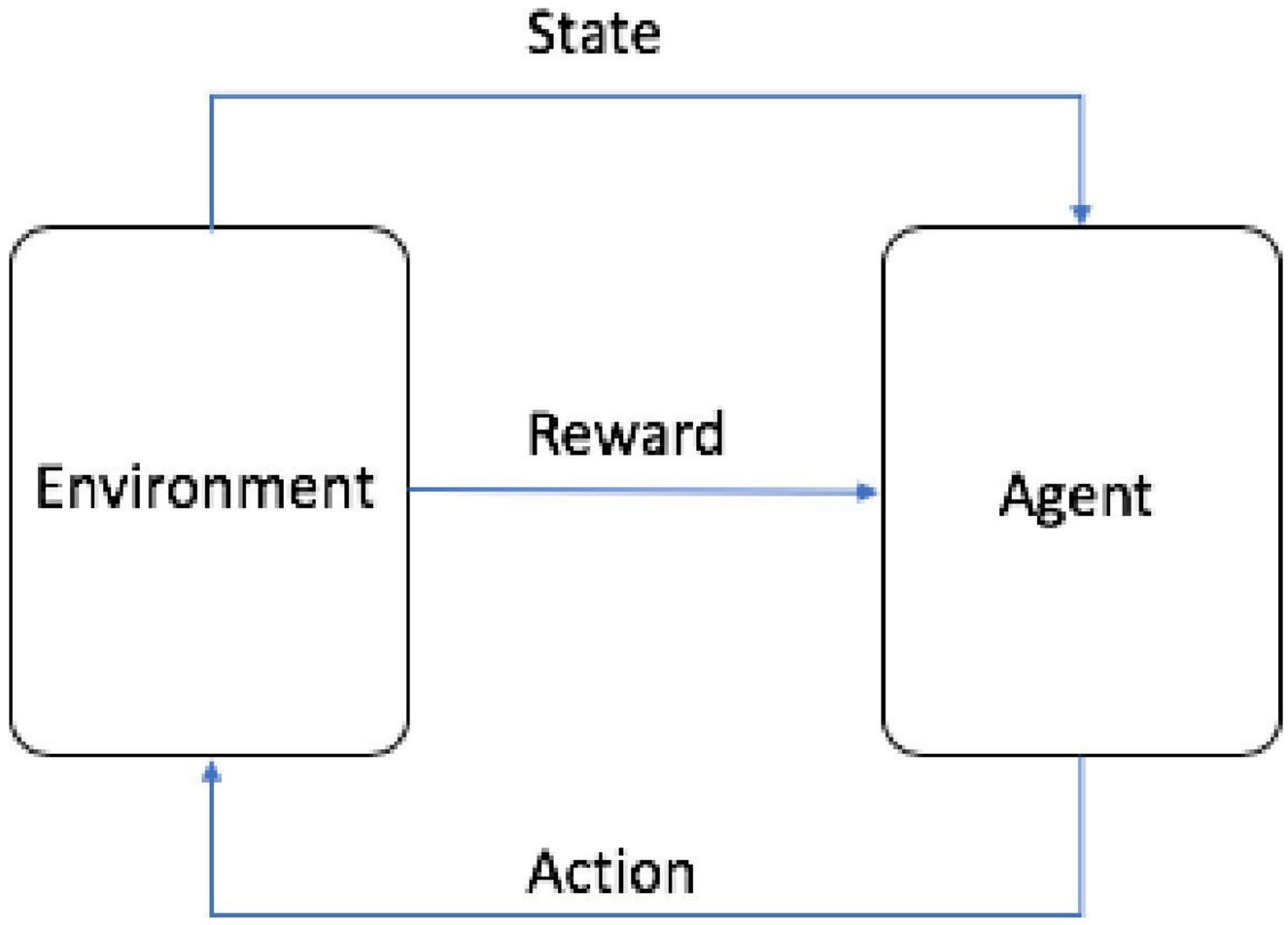


Fig. 1.4 : Apprentissage par renforcement
[6]

Le programme AlphaGo de DeepMind est un bon exemple d'apprentissage par renforcement : il a fait la une des journaux en mars 2016 lorsqu'il a battu le champion du monde Lee Sedol au jeu de Go. Il a appris sa politique gagnante en analysant des millions de parties, puis en jouant de nombreuses parties contre lui-même.

1.4 Apprentissage Profond

L'apprentissage profond (*Deep Learning*) est un domaine avancé de l'apprentissage automatique (*Machine Learning*), doté de capacités puissantes d'analyse et d'extraction d'informations. Il est particulièrement adapté à des applications complexes telles que la vision par ordinateur, la bioinformatique, et le traitement du langage naturel.[7]

Ce type d'apprentissage est reconnu pour offrir des performances significativement supérieures à celles des algorithmes traditionnels de *Machine Learning*, notamment dans les contextes où le traitement de volumes massifs de données est nécessaire.

Une des caractéristiques essentielles du *Deep Learning* réside dans sa capacité à apprendre automatiquement des représentations complexes des données, sans intervention humaine directe. Cela le rend à la fois très flexible, autonome et puissant, ouvrant la voie à des avancées majeures dans plusieurs domaines scientifiques et industriels.

1.4.1 Historique

L'histoire de l'apprentissage profond s'étend sur plusieurs décennies, marquée par des contributions fondamentales :

- **1943** : *Walter Pitts* et *Warren McCulloch* proposent le premier modèle mathématique du neurone artificiel, posant les bases des réseaux de neurones.
- **1958** : *Frank Rosenblatt* développe le *perceptron*, l'un des premiers modèles d'apprentissage supervisé basé sur les neurones artificiels.
- **1974** : *Paul Werbos* introduit l'algorithme de rétropropagation du gradient, permettant l'entraînement efficace des réseaux multicouches.
- **1986** : *David Rumelhart*, *Geoffrey Hinton* et *Ronald Williams* popularisent la rétropropagation, ouvrant la voie à l'entraînement des réseaux profonds.
- **Années 1990** : *Yann LeCun* développe les réseaux convolutifs (*CNN*), appliqués avec succès à la reconnaissance de chiffres manuscrits.
- **2012** : *AlexNet*, conçu par *Alex Krizhevsky*, *Ilya Sutskever* et *Geoffrey Hinton*, remporte la compétition *ImageNet*, déclenchant un regain d'intérêt massif pour le *Deep Learning*.
- **2014–2015** : Émergence des *GAN* (Generative Adversarial Networks) par *Ian Goodfellow*, et développement des *Autoencoders variational* (*VAE*), ouvrant la voie à la génération de contenu réaliste.
- **2017** : Google introduit le modèle *Transformer*, base des modèles de langage modernes comme BERT et GPT.
- **2018–2020** : Progrès spectaculaires des modèles de traitement du langage naturel (*NLP*) comme *BERT*, *GPT-2*, puis *GPT-3*, capables de comprendre et de générer du texte de manière cohérente.
- **2021–2023** : Déploiement à grande échelle de l'intelligence artificielle dans la médecine, la finance, la conduite autonome et les arts numériques. Apparition de modèles multimodaux comme *CLIP* et *DALL·E*.

- **2024** : Avancées dans l'optimisation des grands modèles de langage (LLM) avec des versions plus efficaces, personnalisables et moins coûteuses en énergie.
- **2025** : L'apprentissage profond s'oriente vers l'intégration de plusieurs modalités (texte, image, vidéo, audio) dans des systèmes unifiés. L'intelligence artificielle générative est de plus en plus utilisée pour la simulation 3D, la médecine de précision, et l'éducation interactive personnalisée.

1.4.2 Les Réseaux de Neurones

Les réseaux de neurones sont des outils très utilisés pour la classification, l'estimation,la prédiction et la segmentation[8] Un réseau de neurones, ou « réseau neuronal », possède une ar chitecture inspirée du cerveau humain, organisée en neurones et synapses, et se présente comme un ensemble de nœuds (appelés aussi neurones formels ou unités) interconnectés entre eux.

Un réseau est structuré en couches : une couche d'entrée, une ou plusieurs couches cachées, et une couche de sortie. Il peut être constitué d'une seule couche cachée,auquel cas il est appelé perceptron. Dans ce type de réseau, chaque neurone de la couche reçoit toutes les données en entrée, effectue un traitement, puis transmet le résultat à un neurone de sortie unique. La valeur produite constitue la sortie finale du réseau

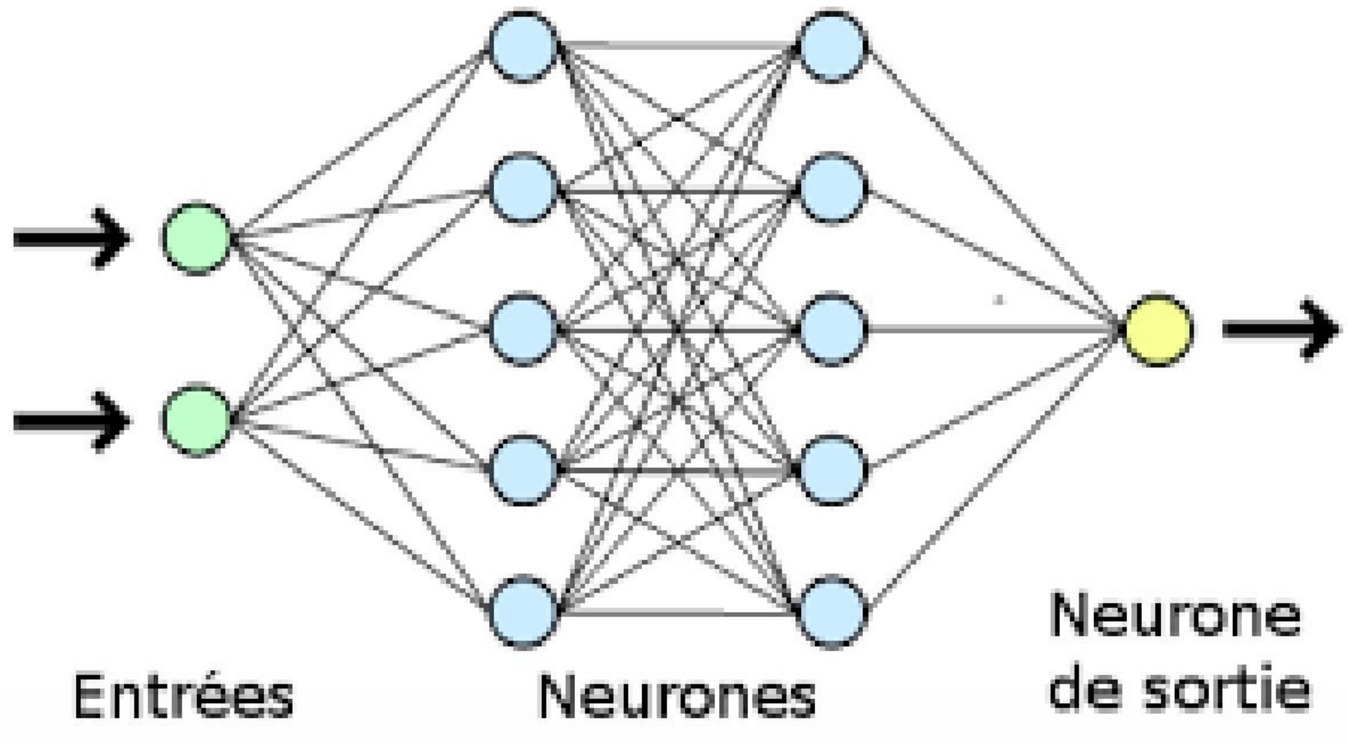


Fig. 1.5 : Structure d'un réseau de neurone à couche unique (perceptron) [9]

Les réseaux de neurones artificiels (RNA) trouvent leur inspiration dans le fonctionnement des réseaux de neurones biologiques présents dans le cerveau humain. En effet, le principe fondamental repose sur la tentative de modéliser, de manière simplifiée, la manière dont les neurones biologiques traitent, transmettent et apprennent des informations. Chaque neurone artificiel imite le comportement d'un neurone biologique en recevant des signaux d'entrée (analogues aux signaux synaptiques), en les pondérant, puis en produisant une sortie en fonction d'une fonction d'activation.

Cette analogie biologique a permis le développement de modèles capables d'apprendre à partir de données, de reconnaître des motifs complexes, et de généraliser sur de nouveaux cas. Bien que les réseaux artificiels soient encore loin de reproduire la complexité et la plasticité du cerveau humain, leur architecture modulaire et leur capacité d'apprentissage distribué incarnent l'esprit des systèmes neuronaux naturels, tout en les adaptant aux besoins computationnels modernes.

Les différents types de réseaux de neurones

Il existe différentes sortes de réseaux neuraux. En règle générale, les réseaux de neurones sont classés en fonction du nombre d'épaisseurs qui séparent l'entrée de données de la sortie du résultat, chacun conçu pour résoudre des tâches spécifiques en fonction de leur architecture et de leur fonctionnement. Voici quelques exemples de réseaux de neurones couramment utilisés :

1.4.2.1 Réseaux de neurones convolutifs :

Un réseau de neurones convolutifs (*Convolutional Neural Network*, CNN est un algorithme d'apprentissage en profondeur largement utilisé en vision par ordinateur pour des tâches telles que la classification et la détection d'objets, en raison de sa structure inspirée du cortex visuel.

Contrairement aux réseaux d'apprentissage en profondeur classiques, les CNN intègrent des couches spécifiques telles que les couches de convolution et les couches de mise en commun (pooling). Ces couches introduisent un chaînage partiel des connexions, ce qui permet de réduire le nombre total de paramètres tout en favorisant le partage des caractéristiques communes entre les différentes régions de l'image.

Malgré leurs nombreux avantages, les CNN présentent certains défis, notamment :

- Le risque de surapprentissage (*overfitting*) lorsque le volume de données d'entraînement est limité.
- Une complexité de calcul élevée, surtout dans les architectures profondes.

Les architectures CNN modernes sont basées sur des blocs convolutifs optimisés, capables de produire des structures plus profondes tout en étant moins exigeantes en termes de ressources de calcul et de stockage.

Les réseaux de neurones convolutifs, tels qu'illustrés à la figure 1.6, intègrent généralement plusieurs couches de convolution, de regroupement (pooling) et de normalisation. Ces couches permettent d'extraire progressivement des caractéristiques de bas niveau (bords, textures) vers des caractéristiques de plus haut niveau (formes, objets complexes) à partir de l'entrée, tout au long du processus d'inférence.

Les représentations ainsi extraites sont ensuite transmises à des couches finales, telles qu'un perceptron multicouche ou des couches entièrement connectées (fully connected layers), qui assurent la classification des données en un nombre fini de catégories prédéfinies.[10][11]

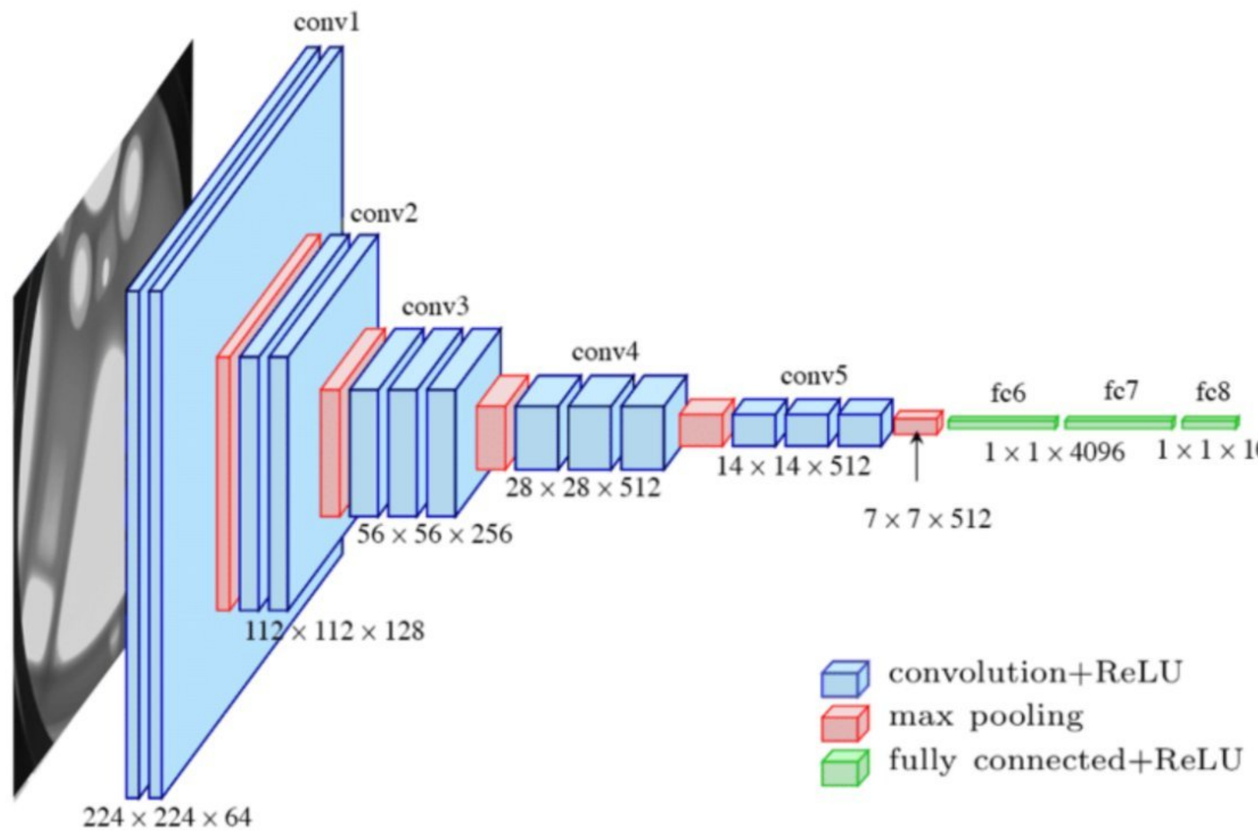


Fig. 1.6 : Exemple d'architecture CNN

Une architecture CNN typique est constituée d'un empilement de couches de traitement, chacune jouant un rôle spécifique dans l'extraction et l'interprétation des caractéristiques des données d'entrée :

- Couche d'entrée du CNN
- Couche de convolution
- Couche de mise en commun (Pooling)
- Couche entièrement connecté
- Couche Logistique ou Softmax
- Couche de sortie (output layer)

Couche d'entrée du CNN

La couche d'entrée dans CNN doit contenir des données décrivant l'image. Les données d'image sont représentées par une matrice tridimensionnelle qui en général doit être remodelée en une seule colonne (représentation vectorielle).

Couche de convolution

La couche de convolutionest parfois appelée couche d'extraction de caractéristiques, car les caractéristiques de l'image sont extraites dans cette couche. Tout d'abord, une partie de l'image est connectée à la couche Convo pour effectuer une opération de convolution et calculer le produit scalaire entre le champ récepteur (c'est une région locale de l'image d'entrée ayant la même taille que celle du filtre) et le filtre comme le montre la figure 1.7. Le résultat de l'opération est un entier unique du volume de sortie. Ensuite, nous faisons glisser le filtre sur le champ récepteur suivant de la même image d'entrée par une foulée et refaisons la même opération. Cette opération est répétée par le même processus encore et encore jusqu'à ce que toute l'image soit parcourue.

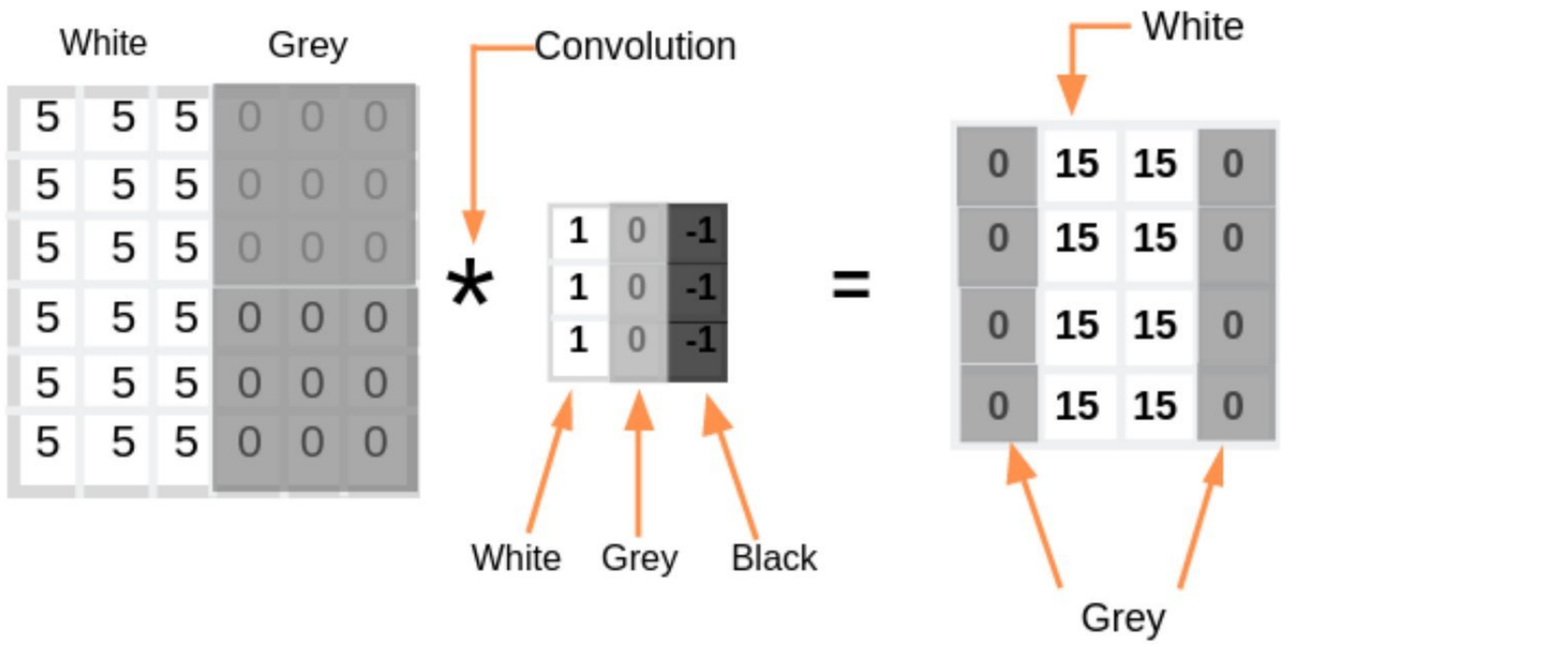


Fig. 1.7 : Exemple de principe du filtre convolutionnel [12]

La couche Convo contient également l'activation ReLU voir figure 1.8 pour que toutes les valeurs négatives soient mises à zéro.

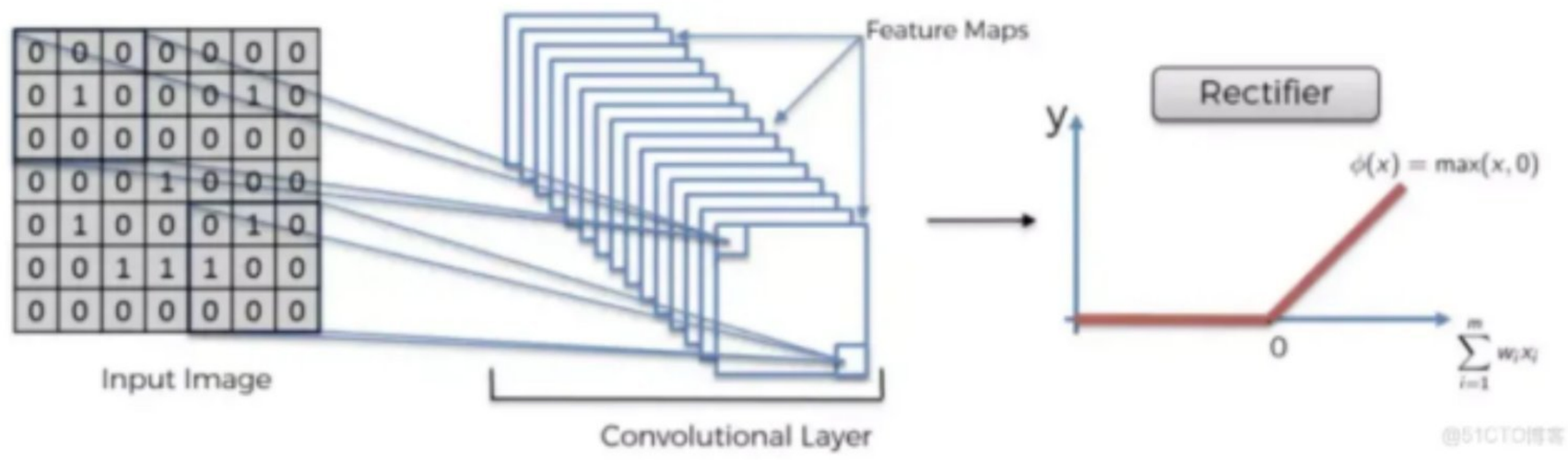


Fig. 1.8 : Principe de la fonction ReLul

Couche de mise en commun (Pooling)

La couche de Pooling est utilisée pour réduire le volume spatial de l'image d'entrée après la convolution. Elle est utilisée entre deux couches de convolution. Si nous appliquons FC (Fully Connected) après la couche Convo sans appliquer le pooling ou le pooling maximum, le calcul sera coûteux. Ainsi, la mise en commun maximale est le seul moyen de réduire le volume spatial de l'image d'entrée en codant l'information.

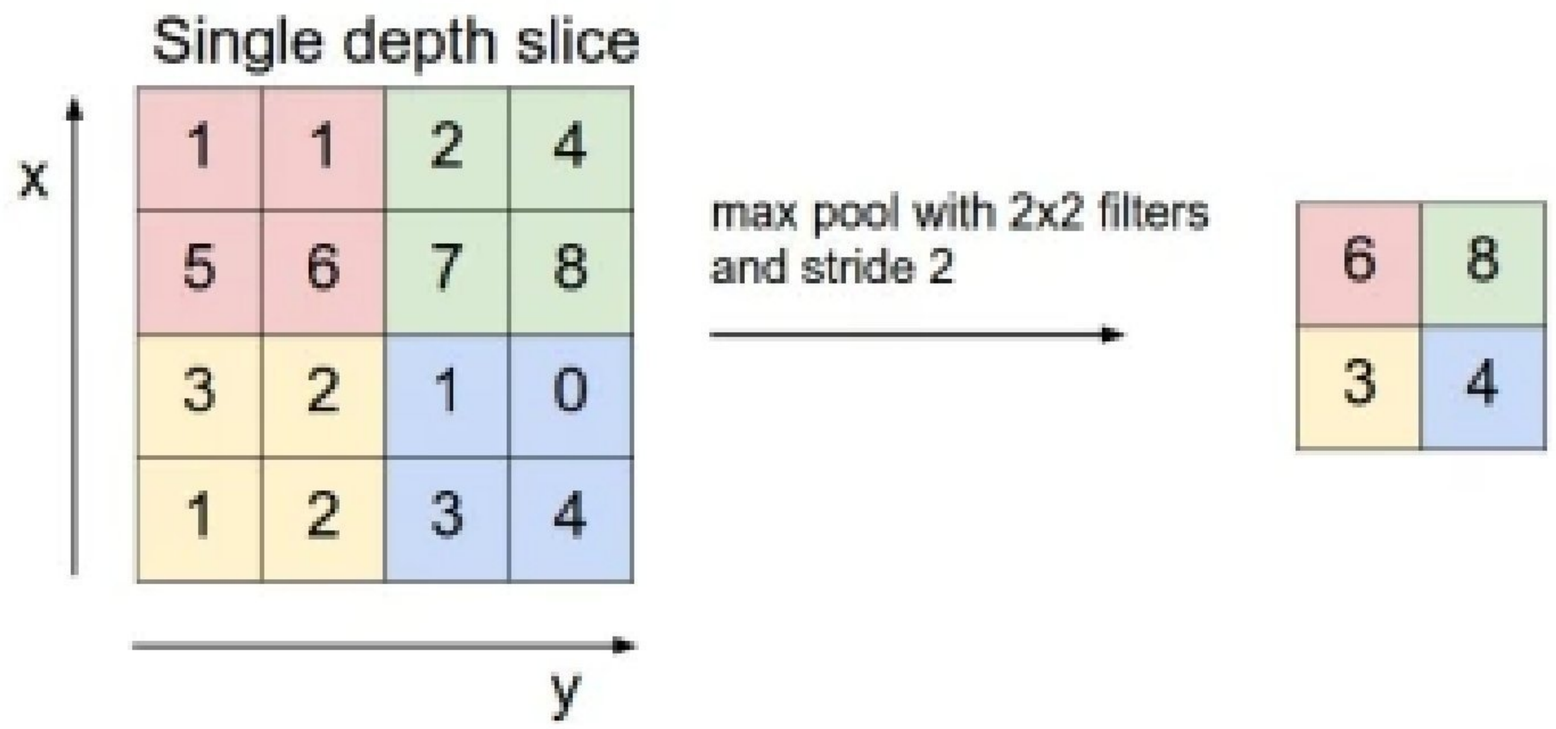


Fig. 1.9 : Exemple de principe du Pooling
[13]

Couche entièrement connecté

Une couche entièrement connectée (Fully Connected) implique des poids,des biais et des neurones. Il connecte les neurones d'une couche aux neuronesd'uneautrecouche.Il est utilisé pour classer les images entre différentes catégories par formation.

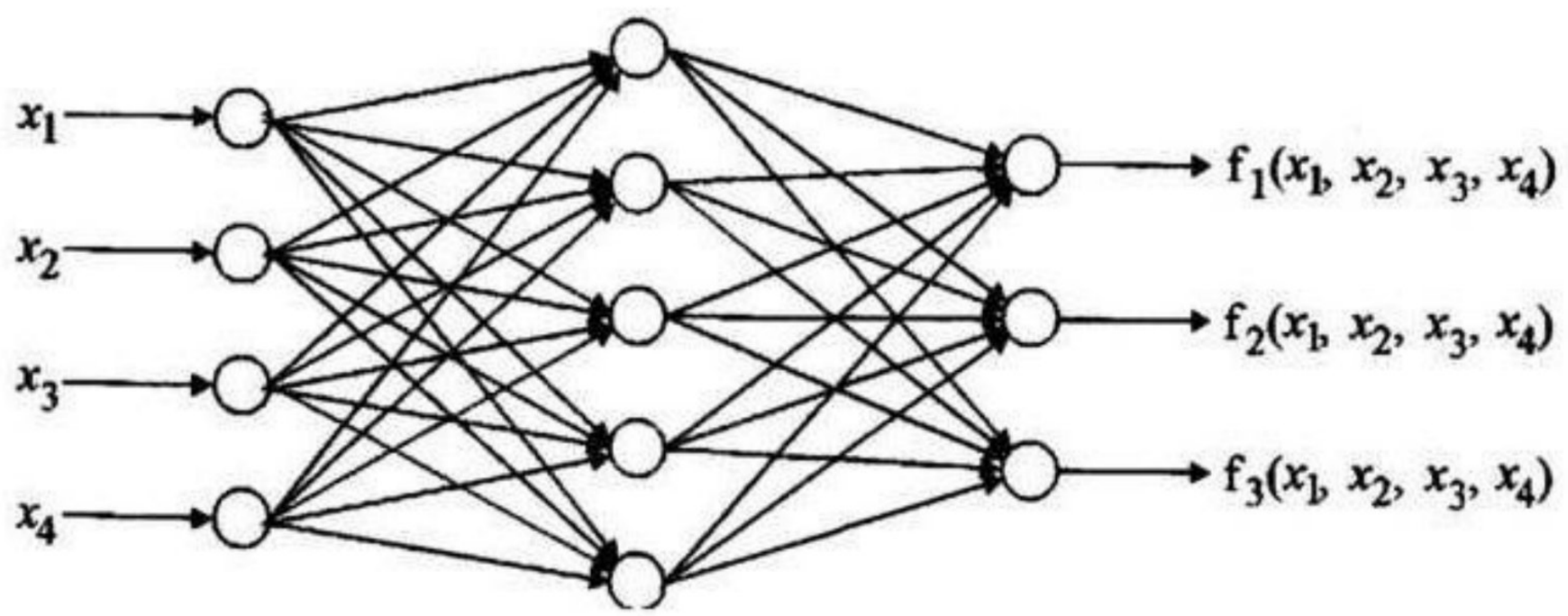


Fig. 1.10 : Principe de la couche entièrement connectée (fc)
[14]

Couche Logistique ou Softmax

Softmax ou couche logistique est la dernière couche de CNN. Elle réside à la fin de la couche FC. La logistique est utilisée pour la classification binaire et Softmax est pour la multi classification.

Couche de sortie (output layer)

La couche de sortie contient l'étiquette qui est sous forme codée comme le montre(la figure 1.11.)

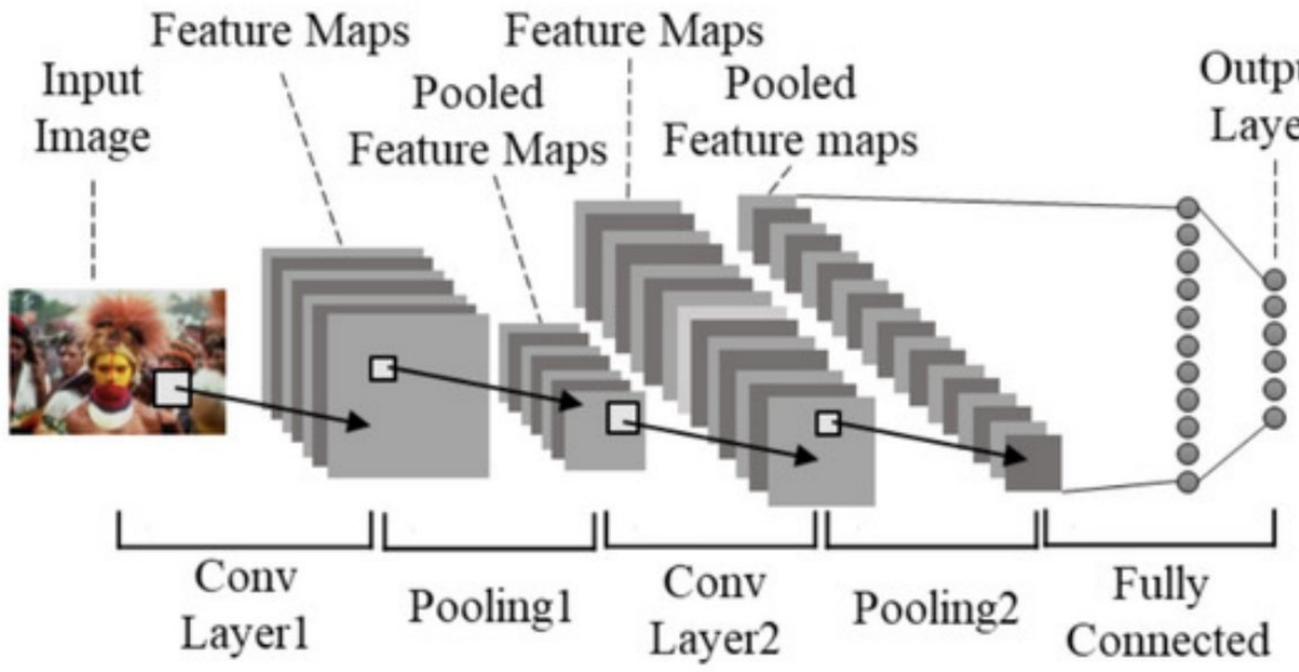


Fig. 1.11 : Exemple montrant l'étiquette codée de la couche de sortie CNN [15]

Avantages des CNN dans le Traitement des Images

Les réseaux de neurones convolutifs (CNN) se distinguent des réseaux de neurones traditionnels par plusieurs caractéristiques qui les rendent particulièrement efficaces pour le traitement des images :

- Connectivité locale : Chaque neurone d'une couche de convolution est connecté uniquement à une région locale de la couche précédente, appelée *champ récepteur*. Cette approche exploite la corrélation spatiale locale présente dans les images, permettant au réseau de capturer efficacement les motifs locaux
- Partage des poids : Les filtres de convolution sont appliqués de manière uniforme sur toute l'image, ce qui signifie que les mêmes paramètres (poids) sont utilisés pour détecter une caractéristique spécifique à différents endroits. Cette technique réduit considérablement le nombre de paramètres à apprendre, rendant le modèle plus efficace et moins sujet au surapprentissage.
- Invariance aux translations : Grâce aux opérations de *pooling* (regroupement), les CNN acquièrent une certaine invariance aux translations et aux distorsions mineures des objets.

1.4.2.2 Réseaux de neurones récurrents :

Les Réseaux de Neurones Récurrents (RNN) sont une variante importante des réseaux neuronaux, largement utilisés dans le traitement du langage naturel. Ils sont appelés ”récurrents” car ils effectuent la même tâche pour chaque élément d'une séquence, la sortie dépendant des calculs précédents [16]

Une autre façon de concevoir les RNN est qu'ils possèdent une ”mémoire” qui capture des

informations sur ce qui a été calculé jusqu'à présent. En théorie, les RNN peuvent utiliser des informations dans des séquences arbitrairement longues, mais en pratique, ils se limitent à considérer quelques étapes en arrière. Les RNN sont une classe de réseaux de neurones qui permettent aux prédictions antérieures d'être utilisées comme entrées, par le biais d'états cachés. Ils sont représentés de la manière illustrée dans la (Figure 1.12).

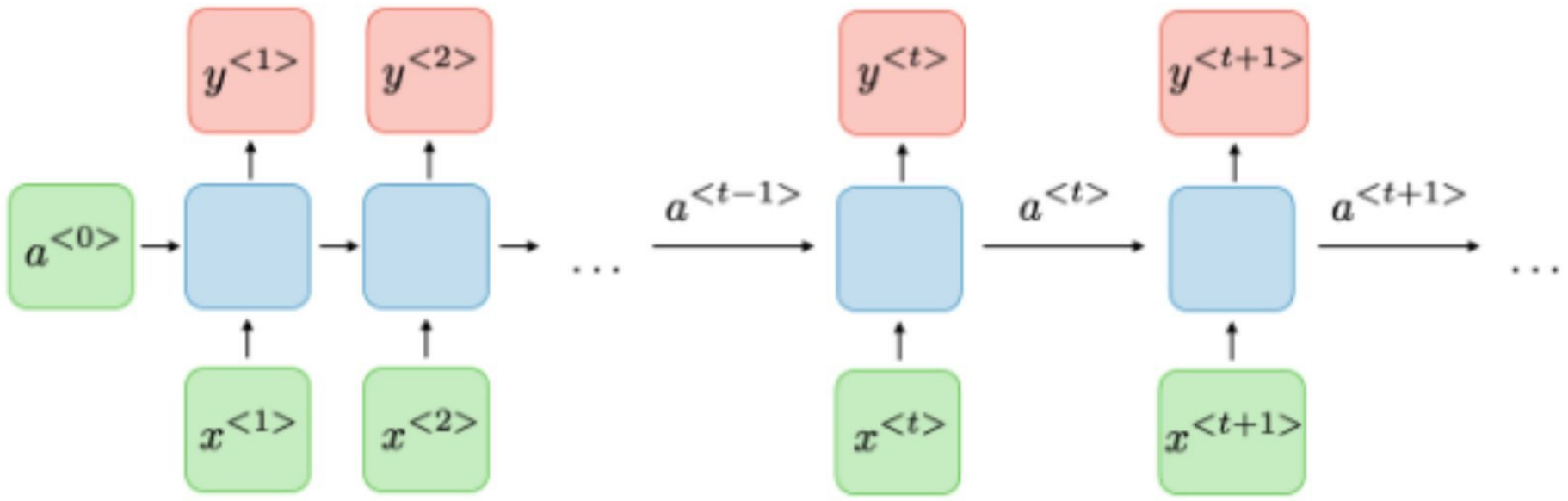


Fig. 1.12 : Architecture de RNN
[17]

1.4.2.3 Réseaux Long Short-Term Memory

Les réseaux Long Short-Term Memory (LSTM) ont été introduits par Hochreiter et Schmidhuber en 1997 [18] pour résoudre le problème du gradient qui disparaît dans les réseaux neuronaux récurrents (RNN). Cette architecture permet de modéliser efficacement les dépendances à long terme dans les séquences de données.

Chaque cellule LSTM comprend trois portes principales :

- Porte d'oubli (forget gate) : détermine quelles informations de l'état précédent doivent être oubliées.
- Porte d'entrée (input gate) : décide quelles nouvelles informations seront stockées dans la cellule.
- Porte de sortie (output gate) : contrôle quelles informations de la cellule sont utilisées pour la sortie.

Ces portes utilisent des fonctions d'activation telles que la sigmoïde et la tangente hyperbolique pour réguler le flux d'informations. Grâce à cette structure, les LSTM sont particulièrement efficaces pour des tâches telles que la reconnaissance vocale, la traduction automatique et la modélisation de séries temporelles.

1.4.2.4 Unité Récurrente à Portes :

Les Unité Récurrente à Portes, an anglais Gated Recurrent Units(GRU) sont une variante des réseaux de neurones récurrents (RNN), introduite par Cho et al. en 2014. Conçues pour traiter efficacement les données séquentielles, elles visent à résoudre le problème de la disparition du gradient rencontré dans les RNN traditionnels. Les GRU utilisent deux mécanismes de portes : la porte de mise à jour (update gate) et la porte de réinitialisation (reset gate), permettant de contrôler le flux d'informations et de capturer les dépendances à long terme. Comparées aux LSTM, les GRU présentent une architecture plus simple avec moins de paramètres, ce qui les rend plus rapides à entraîner tout en offrant des performances comparables dans des tâches telles que la modélisation du langage, la reconnaissance vocale et la prévision de séries temporelles.[19]

1.4.2.5 Les Auto-encodeurs

Un auto-encodeur (ou « Auto-encoder » en anglais) est un type de réseau de neurones profonds qui est conçu pour apprendre à représenter une donnée d'entrée en utilisant une quantité réduite de données. En d'autres termes, il cherche à compresser la donnée en un format plus compact. Les Auto-encodeurs sont couramment utilisés en apprentissage automatique pour des tâches telles que la compression de données, l'apprentissage de représentations et la détection de motifs .[20]

Fonctionnement et Architecture des Auto-encodeurs

Les auto-encodeurs sont des réseaux de neurones dont le nombre de neurones dansla couche d'entrée est identique à celui de la couche de sortie.ils ont une architecture particulière appelée « architecture bottleneck », où les couches cachées sont plus petites que les couches d'entrée. Cette architecture peut être décomposée en trois parties distinctes comme le montre (la figure 1.13)

- Encodeur : Il est chargé de transformer l'entrée en une représentation dans une space latent de dimension réduite. Cette opération permet de compresser l'entrée en une représentation plus économique.
- Goulot d'étranglement (ou bottleneck) :Cette partie du réseau stocke la représentation en codée de l'entrée,qui sera ensuite transmise au décodeur pour effectuer la reconstruction de l'entrée initiale.[21]
- Décodeur :le décodeur est chargé de reconstruire les données originales à partir de cette représentation compressée.

Il prend en entrée le vecteur latent produit par l'encodeur et tente de générer une sortie aussi proche que possible de l'entrée initiale. Il s'agit d'un processus d'apprentissage non supervisé où le modèle apprend à minimiser l'erreur de reconstruction entre l'entrée et la sortie.

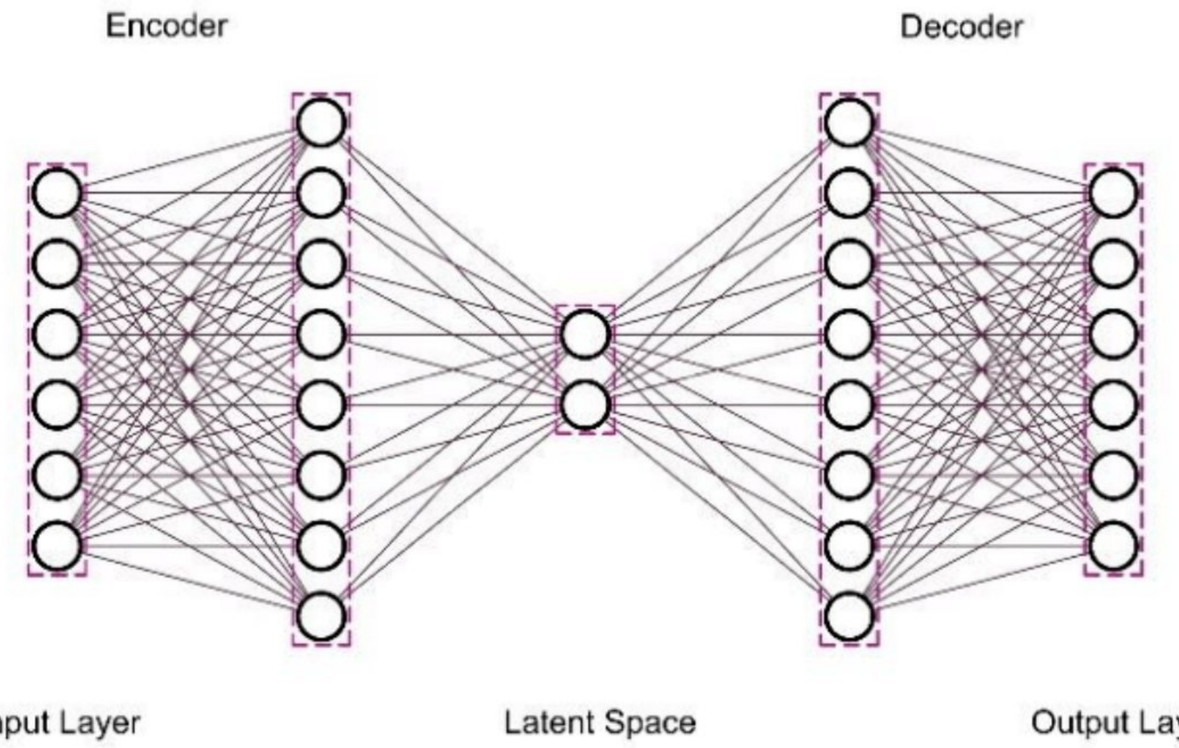


Fig. 1.13 : Architecture d'un auto-Encodeur
[22]

L'objectif principal de l'auto-encodeur est d'obtenir une sortie similaire à l'entrée. Bien que l'architecture du décodeur puisse différer de celle de l'encodeur, dans la plupart des cas, elle est conçue pour être l'image miroir de l'encodeur. Il est important que la dimensionnalité de l'entrée et de la sortie soit la même. Les couches convolutives sont souvent utilisées pour l'encodeur et les couches dé convolutives pour le décodeur en pratique.

Types d'Autoencodeurs

Il existe plusieurs types d'autoencodeurs, chacun répondant à un besoin spécifique selon le contexte d'utilisation.

- Autoencodeur Classique (Vanilla Autoencoder) : Modèle de base composé d'un encodeur et d'un décodeur, utilisé pour la réduction de dimensionnalité et l'extraction de caractéristiques.
- Autoencodeur Débruiteur (Denoising Autoencoder) : Cet autoencodeur est entraîné à reconstruire une entrée propre à partir d'une version bruitée. Cette approche améliore la robustesse du modèle face aux perturbations et permet une meilleure généralisation.
- Autoencodeur Contractif (Contractive Autoencoder) : L'autoencodeur contractif introduit une pénalité dans la fonction de coût pour rendre la représentation latente moins sensible aux petites variations de l'entrée. Cela favorise l'apprentissage de représentations plus robustes.
- Autoencodeur Variationnel (Variational Autoencoder - VAE) Le VAE est un modèle probabiliste qui apprend une distribution sur l'espace latent, permettant la génération de nouvelles données similaires aux données d'entraînement. Il combine des techniques d'inférence variationnelle avec des autoencodeurs.
- Autoencodeur Clairsemé (Sparse Autoencoder) Cet autoencodeur impose une contrainte de parcimonie sur la couche cachée, forçant le modèle à activer uniquement certaines unités. Cela favorise l'apprentissage de représentations plus interprétables et utiles pour

des tâches en aval.

Avantages des autoencodeurs

Les autoencodeurs offrent plusieurs bénéfices dans les domaines de l'apprentissage automatique et du traitement de données, que nous allons présenter ci-dessous.

- Apprentissage non supervisé : n'exige pas de labels pour l'entraînement, ce qui est utile lorsque les données annotées sont rares.
- Réduction de dimensionnalité : capables de compresser les données tout en conservant les informations essentielles.
- Extraction de caractéristiques : apprennent automatiquement des représentations utiles pour d'autres tâches d'apprentissage automatique.
- Débruitage : efficaces pour éliminer le bruit des données, améliorant ainsi la qualité des entrées.
- Détection d'anomalies : peuvent identifier des données inhabituelles en mesurant la différence entre l'entrée et la reconstruction.

1.4.2.6 Les réseaux antagonistes génératifs

Les GAN ont été introduits pour la première fois par Ian Goodfellow [23] et ses collègues en 2014 dans l'article "Generative Adversarial Networks" publié dans la conférence NIPS. De puis leur mise en place, les GAN ont été utilisés dans de nombreuses applications, tels que la génération d'images, la synthèse de voix, la traduction de langues et bien plus encore. Le GAN ou Réseau Antagoniste Génératif (Generative Adversarial Network en anglais) est une méthode de l'apprentissage automatique permettront de créer une cartographie qui transforme une distribution simple et connue, telle qu'une distribution gaussienne, en une distribution plus complexe et spécifique à un domaine de modèles donné. Le principe de base de cette méthode repose sur la confrontation de deux réseaux au sein d'un cadre de travail spécifique pour générer des données qui ressemblent à des données réelles.Ces deux réseaux sont appelés générateur et discriminateur.

1.4.2.7 Les Transformers

Les transformers sont des architectures de réseaux de neurones introduites en 2017 par Vaswani et al. dans l'article Attention Is All You Need. Conçus initialement pour la traduction automatique, ils ont révolutionné le traitement des séquences en remplaçant les mécanismes récurrents par une attention auto-adaptative. Cette approche permet de traiter l'ensemble des éléments d'une séquence en parallèle, améliorant ainsi l'efficacité et la capacité à capturer des dépendances à long terme.

L'architecture standard d'un transformer se compose de deux blocs principaux : un encodeur qui transforme les données d'entrée en représentations contextuelles, et un décodeur qui génère la sortie en se basant sur ces représentations. Au cœur de cette architecture se trouve le

mécanisme d'attention multi-tête, qui permet au modèle de se concentrer sur différentes parties de la séquence pour chaque tâche spécifique.

Les transformers ont été largement adoptés dans divers domaines, notamment le traitement du langage naturel (par exemple, BERT, GPT), la vision par ordinateur (Vision Transformers), la reconnaissance vocale et la bio-informatique. Leur capacité à gérer efficacement de grandes quantités de données séquentielles les rend essentiels pour les modèles de langage de grande taille et d'autres applications avancées de l'intelligence artificielle.

1.5 Conclusion

Pour conclure ce chapitre, nous avons vu les notions importantes autour de l'intelligence artificielle et du deep learning. Ces connaissances sont essentielles pour comprendre les méthodes modernes de traitement d'images. Grâce à ces explications, nous sommes maintenant prêts à passer aux techniques de reconstruction 3D, que nous allons étudier dans le chapitre suivant.

Chapitre 2

De la 2D à la 3D : Approches Classiques et Approches par Deep Learning

Chapitre 2

De la 2D à la 3D : Approches Classiques et Approches par Deep Learning

2.1 Introduction

Dans la continuité des concepts théoriques abordés précédemment, ce chapitre s'attarde sur les différentes méthodes de reconstruction 3D à partir d'images 2D. Il est structuré en deux grandes parties : d'une part, les approches classiques fondées sur des principes géométriques tels que la stéréovision, SfM ou MVS ; et d'autre part, les approches modernes basées sur le deep learning, notamment via les CNN et autoencodeurs. L'objectif est d'offrir une vue comparative entre ces deux paradigmes, en mettant en lumière leurs avantages, limites et domaines d'application.

2.2 Les approches classiques de la conversion 2D vers 3D .

2.2.1 Estimation de la Profondeur

L'estimation de la profondeur est une tâche fondamentale en vision par ordinateur, dédiée à l'extraction d'informations de profondeur 3D à partir d'images 2D. En prédisant la distance de chaque pixel par rapport à la caméra et en construisant une carte de profondeur correspondante, cette technologie ajoute effectivement une troisième dimension aux données visuelles. Elle permet de capturer la structure géométrique et les relations spatiales au sein d'une scène, jouant ainsi un rôle central dans la perception visuelle et l'interaction avec l'environnement [24].

L'objectif de l'estimation de la profondeur est de générer une carte de profondeur $D \in \mathbb{R}^{H \times W}$ à partir d'une image 2D $I \in \mathbb{R}^{H \times W \times 3}$, où chaque valeur de profondeur $d_{i,j} \in D$ représente la distance physique entre un pixel $i_{i,j} \in I$ dans l'image et la caméra [25]. Cependant, cette tâche est par nature mal posée. Une image 2D est une projection du monde 3D, ce qui entraîne une compression ou une perte d'informations de profondeur lors du processus de capture.
La Figure suivante illustre une architecture typique d'un réseau de neurones profond utilisé pour

estimer la profondeur à partir d'une image 2D.

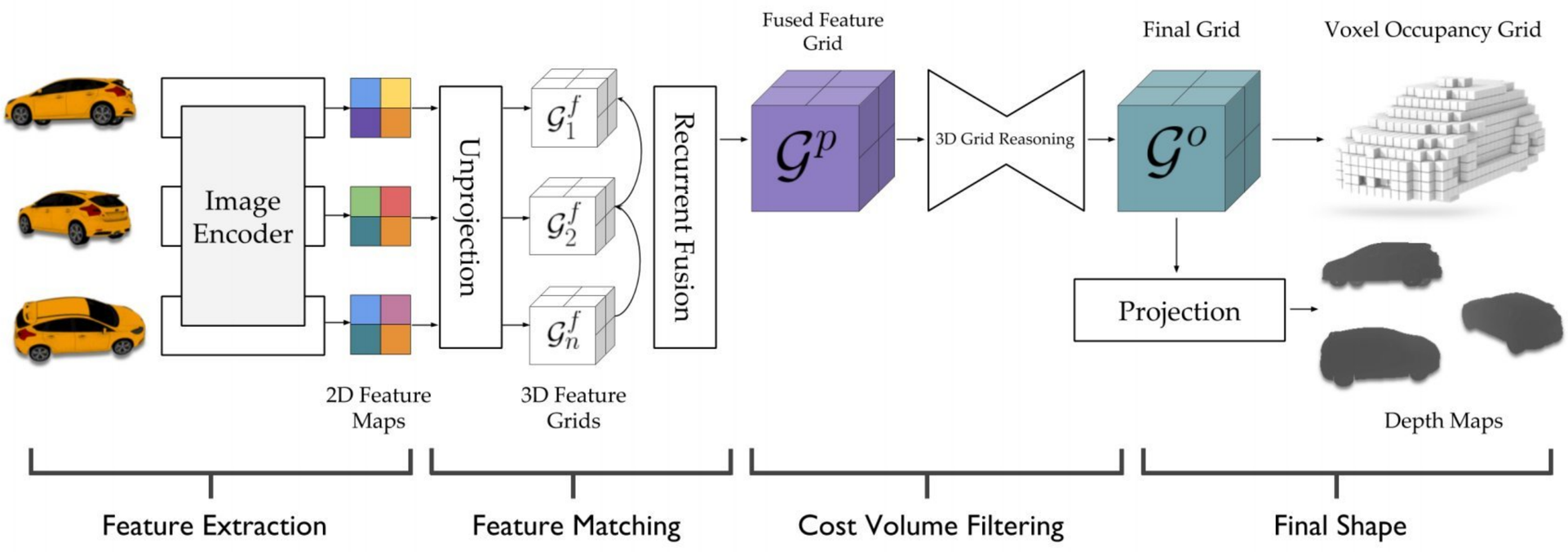


Fig. 2.1 : Architecture d'un réseau profond pour l'estimation monoculaire de la profondeur. [26]

Dans le cas de l'estimation monoculaire de la profondeur, l'absence de parallaxe ou d'informations auxiliaires accentue encore cette ambiguïté, rendant le problème d'autant plus complexe [27].

Malgré ces défis, l'estimation de la profondeur reste indispensable dans de nombreux domaines [10, 11, 28]. En fournissant des informations de profondeur précises, elle permet aux systèmes informatiques de déterminer les positions relatives et les distances des objets dans une scène, établissant ainsi les bases pour une compréhension avancée de l'environnement. Dans la conduite autonome et la robotique, l'estimation de la profondeur améliore la détection d'obstacles, la planification de trajectoires et la perception de l'environnement, augmentant ainsi la fiabilité et la sécurité des systèmes [29]. En réalité augmentée et virtuelle (AR/VR), les données de profondeur permettent une modélisation de scène précise et une interaction réaliste, renforçant l'immersion de l'utilisateur [30]. De plus, en photographie computationnelle et en post-traitement d'images, l'estimation de la profondeur facilite des tâches telles que l'imagerie multifocale, la génération de vidéos 3D et le flou d'arrière-plan [31]. En somme, l'estimation de la profondeur est bien plus qu'un domaine de recherche central en vision par ordinateur : c'est une technologie transformatrice qui permet aux systèmes de vision de percevoir et d'interagir avec le monde en trois dimensions. Ses nombreuses applications ont le potentiel de stimuler l'innovation, en comblant le fossé entre les données visuelles et la compréhension du monde réel.

2.2.2 La stéréovision

La stéréovision est une technique pour déduire des informations tridimensionnelles à partir de deux ou plusieurs images bidimensionnelles représentant des prises de vue d'une même scène sous des angles légèrement différents [32]. On peut aussi dire « la stéréovision consiste à observer une même scène avec deux caméras qui sont éloignées l'une de l'autre et dont on connaît la distance qui les séparent. Connaissant la géométrie exacte du système stéréoscopique

la première étape de reconstruction 3D consiste à mettre en correspondance les deux images. Cette phase réside dans la détermination de couples de points observés dans les deux images, ou dans l'appariement de points d'intérêt» [33]. La stéréovision est une technique de vision fondée sur l'utilisation de plusieurs caméras, en combinant leur différents points de vue, cela permet d'accéder à des informations de profondeur difficilement accessibles lorsqu'une unique vue est utilisée.

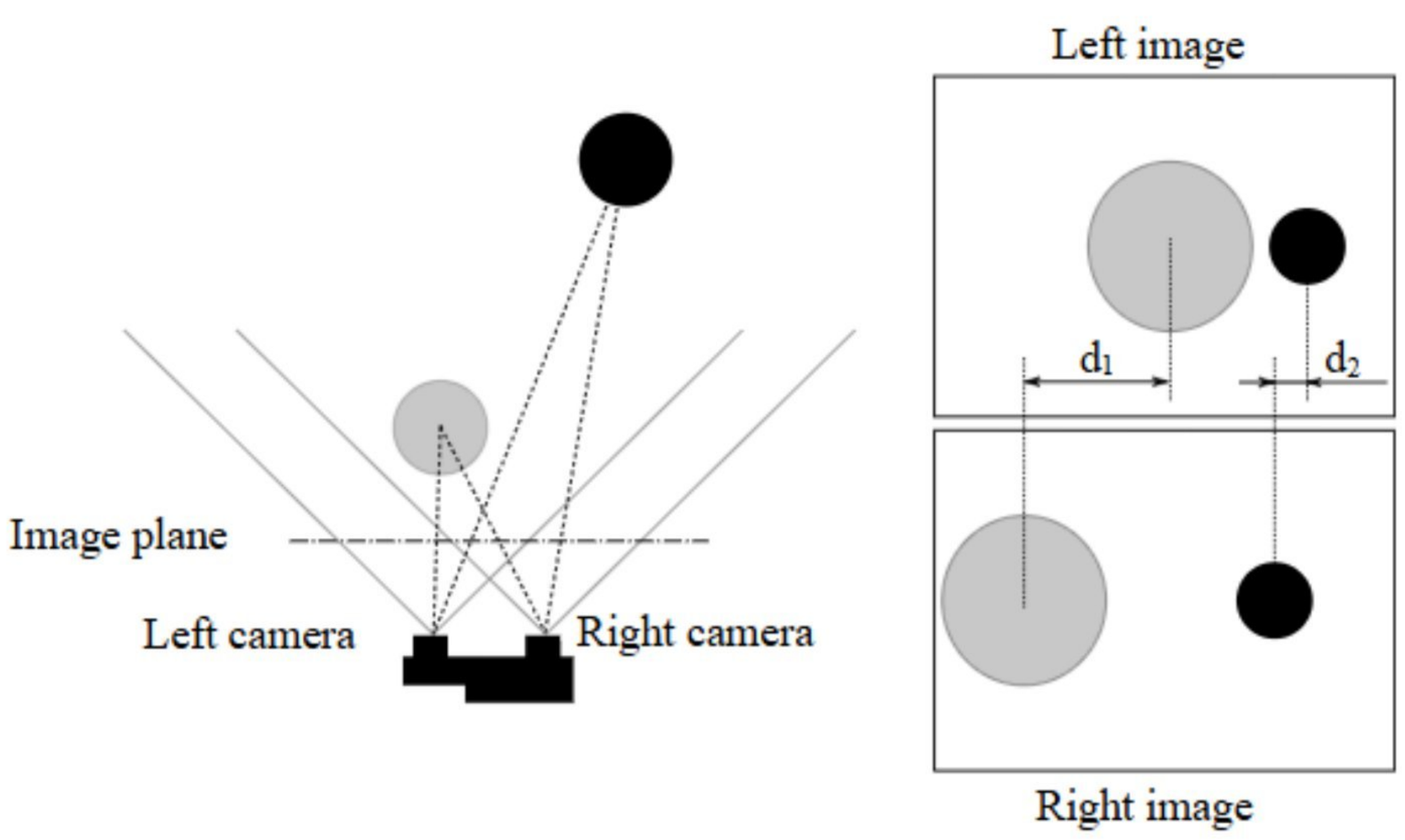


Fig. 2.2 : Principe de la stéréovision : deux caméras capturent une même scène sous des angles légèrement différents pour reconstruire la profondeur par triangulation.
[34]

L'exploitation de cette information permet de bâtir des techniques de détection d'obstacles génériques et robustes. Selon de nombreux auteurs, ces techniques peuvent être rangées en deux catégories : les méthodes par appariements et les méthodes par rectification homographique [35].

Le principe de la stéréovision (stéréoscopie) consiste à reconstruire tridimensionnellement une cible ou toute une scène en utilisant des images prises simultanément (prises de points de vue différents) ou séquentiellement. Plusieurs chercheurs ont étudiés des systèmes stéréoscopiques formés de deux ou plusieurs caméras. Généralement, on peut distinguer deux approches :

2.2.2.1 La stéréovision dynamique

Est une méthode de mesure tridimensionnelle d'une cible qui se déplace entre les prises de vue. Dans ce cas une seule caméra pourrait être suffisante pour effectuer des mesures tridimensionnelles. Plusieurs images prises séquentiellement sont considérées comme des images acquises depuis différents points de vue.

2.2.2.2 La stéréovision statique

Elle impose que les objets à mesurer soient fixes par rapport au référentiel objet. Les techniques de reconstruction tridimensionnelle de cette approche utilisent des images acquises au moyen d'une seule caméra qui se déplace dans la scène ou bien deux ou plusieurs caméras afin de prendre des images simultanément.
Dans toutes les techniques utilisées, les cibles devraient être visibles dans au moins deux images. [36]

2.2.3 La structure from motion

Le principe de structure from motion (SfM, « Structure acquise à partir d'un mouvement ») est une technique d'imagerie par intervalle (en) photogrammétrique destinée à estimer la structure 3D d'un objet à partir d'images 2D. Elle combine la vision par ordinateur et la vue humaine. La SfM désigne le phénomène par lequel une personne (et autres créatures vivantes) peut estimer la structure 3D d'un objet ou d'une scène en mouvement à partir de son champ de vision 2D (rétinien).

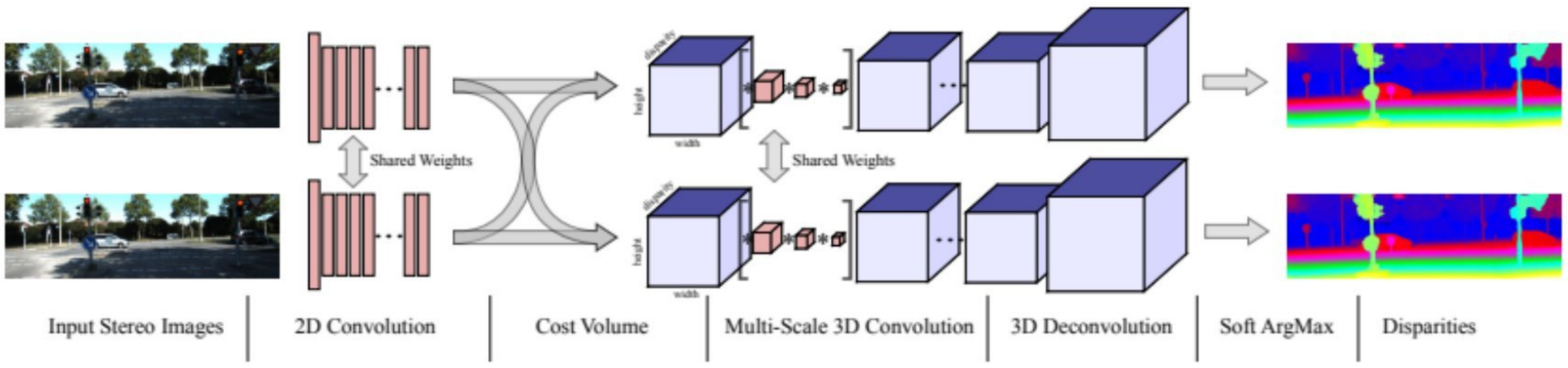


Fig. 2.3 : Diagramme illustrant le principe de la méthode Structure from Motion (SfM).

La création de modèles numériques de surface (MNS) à l'aide de la photogrammétrie permet de quantifier des changements topographiques(Lane et al., 1993). Les récents développements informatiques et dans le traitement de l'image ont permis la mise au point de nouvelles méthodes facilitant la création de modèles topographiques et d'orthophotos. La photogrammétrie digitale « Structure from Motion, (SfM) »(Smith et al., 2015; Snavely et al., 2008) permet la reconstruction d'orthophotos et de MNS à partir d'images de différents points de vue offrant ainsi une nouvelle possibilité pour le suivi de tout mouvement gravitaire. Parallèlement, le développement d'appareils numériques performants (ANP) et peu coûteux facilite la prise de photographies aériennes de qualité à moindres coûts.

2.2.4 Multi-View Stereo

Reconstruire des modèles 3D à partir de plusieurs images 2D prises sous différents angles. En analysant les différences et similitudes entre ces images, les algorithmes MVS peuvent estimer la profondeur (profondeur de la scène) et la géométrie des objets, permettant ainsi de créer une représentation tridimensionnelle du monde réel.

Cette technique joue un rôle crucial dans plusieurs applications, telles que la réalité virtuelle, la navigation autonome, la cartographie 3D, la robotique, et la préservation du patrimoine culturel. On distingue principalement deux grandes catégories de méthodes MVS : les méthodes traditionnelles et les méthodes basées sur l'apprentissage profond.

2.2.4.1 Méthodes traditionnelles de Multi-View Stereo

Avant l'émergence des réseaux de neurones et de l'apprentissage profond, la stéréoscopie multi-vues représentait déjà un domaine fondamental dans la reconstruction 3D, et avait connu des avancéessignificatives. Ces méthodes traditionnelles peuvent être classées en quatre grandes catégories :

1- Méthodes basées sur les voxels :
elles reconstruisent la scène en divisant l'espace 3D en une grille de petits cubes (voxels), et estiment la probabilité d'occupation de chaque voxel.[37][38]

2- Méthodes basées sur la grille (grid-based) :
elles utilisent des structures régulières pour organiser les données spatiales. [39][40]

3- Méthodes basées sur les surfaces :
elles cherchent à reconstruire directement les surfaces visibles.

4- Méthodes basées sur les cartes de profondeur :
elles estiment une carte de profondeur pour chaque image, puis reprojettent les pixels dans l'espace 3D pour créer un nuage de points.

Parmi toutes ces approches, la méthode basée sur les cartes de profondeur est la plus souple et la plus utilisée dans la pratique. Elle consiste à calculer la profondeur de chaque pixel dans chaque image, puis à fusionner ces informations pour générer un modèle 3D. Plusieurs algorithmes bien connus adoptent cette approche, comme Furu [40], Tola [41], Gipuma [42], et COLMAP [43]

Malgré les performances appréciables de ces méthodes traditionnelles, elles présentent certaines limites : des besoins computationnels élevés, une vitesse de traitement relativement lente, et une difficulté à gérer des scènes avec peu de texture ou des surfaces faiblement réfléchissantes.

2.2.5 Avantages et inconvénients des méthodes classiques

Les méthodes classiques de conversion 2D vers 3D, bien qu'efficaces historiquement, présentent un certain nombre d'avantages et de limitations notables. Voici une synthèse de leurs points forts et points faibles :

Avantages :

- Simplicité conceptuelle :Les méthodes comme la stéréovision ou la structure from motion sont fondées sur des principes géométriques bien établis.
- Matériel accessible : Certaines approches (comme SfM) ne nécessitent qu'une caméra unique ou des appareils standards.
- Résultats précis dans des conditions contrôlées : Avec une calibration correcte, ces techniques peuvent produire des modèles 3D de haute précision .
- Bonne compréhension théorique : Leur fonctionnement est bien compris et largement documenté, ce qui facilite leur implémentation.
- Applications validées : Utilisées depuis longtemps en robotique, cartographie, inspection industrielle etc...

Inconvénients :

- Sensible aux conditions environnementales : Faible luminosité, occlusions, surfaces peu texturées peuvent réduire considérablement la qualité des résultats.
- Traitement coûteux : Certaines méthodes (ex : MVS voxel-based) sont très gourmandes en mémoire et temps de calcul .
- Dépendance à la calibration : Une erreur dans la calibration des caméras peut fausser toute la reconstruction.
- Peu robustes face à la variation de scènes : Elles ont du mal à s'adapter à des environnements complexes ou dynamiques.
- Pas de généralisation : Chaque scène ou séquence peut nécessiter un traitement spécifique.

2.3 La conversion 2D vers 3D utilisant le Deep Learning

L'apprentissage profond a profondément transformé la manière dont on aborde la reconstruction 3D. Contrairement aux méthodes classiques basées sur des principes géométriques explicites, les approches modernes s'appuient sur des réseaux neuronaux capables d'apprendre automatiquement des représentations complexes à partir d'images 2D. Cette capacité permet d'estimer la profondeur et de reconstruire des scènes même en présence de conditions difficiles (variations de lumière, occlusions, etc.).

Dans ce qui suit, nous présentons les principales techniques de reconstruction 3D par Deep Learning :

2.3.1 Estimation monoculaire de la profondeur par Deep Learning

La vision par ordinateur vise à permettre aux machines de comprendre des images numériques. L'un des défis majeurs réside dans l'estimation de la profondeur, car une image 2D est une projection d'une scène 3D, ce qui entraîne une perte d'information spatiale.

L'estimation monoculaire de la profondeur consiste à prédire la distance entre la caméra et chaque point visible dans une image unique. Bien que ce soit un problème mal posé (impossible de trianguler à partir d'une seule vue), il existe des indices de profondeur que les modèles peuvent apprendre, tels que la perspective, l'élévation, le flou, les ombrages, les textures, la taille relative, et les occultations [44] :

Ces indices sont subtilement présents dans les images, et les modèles d'apprentissage profond, tels que les réseaux de neurones convolutifs (CNN) ou les modèles génératifs, peuvent les apprendre automatiquement à partir de larges bases de données, produisant ainsi des cartes de profondeur précises[45]

2.3.2 Utilisation des CNN pour la reconstruction 3D à partir d'images 2D

Les CNN ont révolutionné la reconstruction 3D à partir d'images 2D en permettant l'extraction automatique de caractéristiques visuelles. Grâce à une architecture hiérarchique, ils capturent des informations allant des bords et textures aux formes et structures complexes.

Une architecture typique encodeur-décodeur est utilisée pour transformer les représentations visuelles extraites en cartes de profondeur. Contrairement à l'approche classique d'ingénierie de caractéristiques manuelles, les CNN apprennent les structures de la scène et les relations spatiales de manière implicite.

L'objectif final est de produire une représentation 3D réaliste de la scène à partir d'une seule image, ce qui ouvre des applications dans les domaines de la robotique, de la réalité augmentée, et des véhicules autonomes.

Comme illustré dans la Figure 2.4, les CNN adoptent souvent une architecture de type encodeur-décodeur pour effectuer l'estimation de la profondeur à partir d'une image unique. Cette structure permet au réseau d'apprendre des représentations hiérarchiques, allant des caractéristiques locales aux structures globales de la scène.[46]

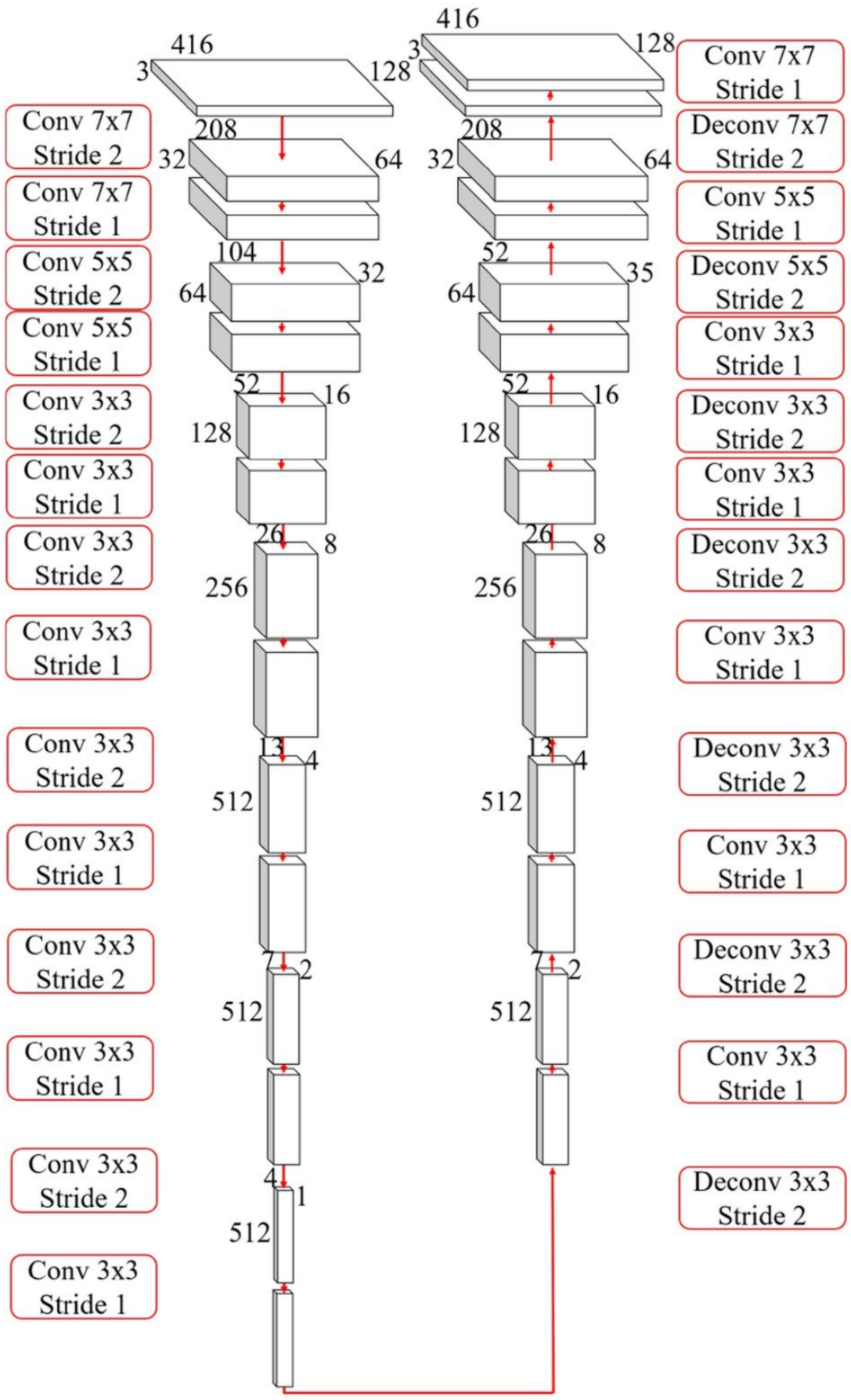


Fig. 2.4 : Architecture d'un réseau CNN pour l'estimation de la profondeur à partir d'une image RGB unique.

[47]

2.3.2.1 Génération de la carte de profondeur

Une fois les caractéristiques extraites, le modèle génère une Depth Map (carte de profondeur). Cette étape est essentielle pour la reconstruction 3D et peut être réalisée de différentes manières :

- Supervisée :en utilisant des données réelles de profondeur comme référence.
- Auto-supervisée ou non supervisée :lorsque les données d'entraînement n'ont pas de vérité terrain, le modèle apprend en utilisant des relations entre les images (par exemple, des images stéréo).

Quelques modèles populaires dans ce domaine incluent :

- Monodepth2 : Ce modèle auto-supervisé génère des cartes de profondeur monoculaires de haute qualité en utilisant une architecture de type encoder-decoder. Il se base sur des images stéréo pour effectuer l'apprentissage sans étiquettes de profondeur. Monodepth2

est particulièrement efficace pour les applications mobiles, car il fonctionne en temps réel avec une seule caméra.[48]

- DenseDepth : Ce modèle utilise une architecture de réseau profond pour estimer des cartes de profondeur à partir d'images monoculaires. Il est entraîné sur des jeux de données de profondeur réels et génère des cartes de profondeur denses et précises.[49]
- MegaDepth : Ce modèle permet de produire des cartes de profondeur à partir d'images monoculaires en s'appuyant sur un large ensemble de données collectées à partir de scènes extérieures, notamment des paysages urbains. MegaDepth a la particularité d'apprendre à partir d'images de très grande échelle et de reconstruire des scènes 3D de manière robuste.[50]

2.3.2.2 Projection en 3D à partir de la carte de profondeur

Une fois la carte de profondeur générée, chaque pixel de l'image peut être projeté dans l'espace 3D grâce à une opération de back-projection, en utilisant les paramètres intrinsèques de la caméra[51]. Cette opération permet de convertir les informations de profondeur en une représentation 3D de la scène, et de générer différentes formes de représentation, telles que :

- un nuage de points (Point Cloud) :une collection de points 3D représentant la scène.
- une maillage polygonale (Mesh) : une représentation de la scène sous forme de surfaces connectées.
- une grille voxelisée (Voxel Grid) :une représentation en 3D utilisant une grille de voxels.

Comme illustré dans la Figure 2.5, la technique de projection inversée est utilisée pour convertir la carte de profondeur en une représentation 3D, permettant ainsi de reconstruire la scène avec une grande précision.

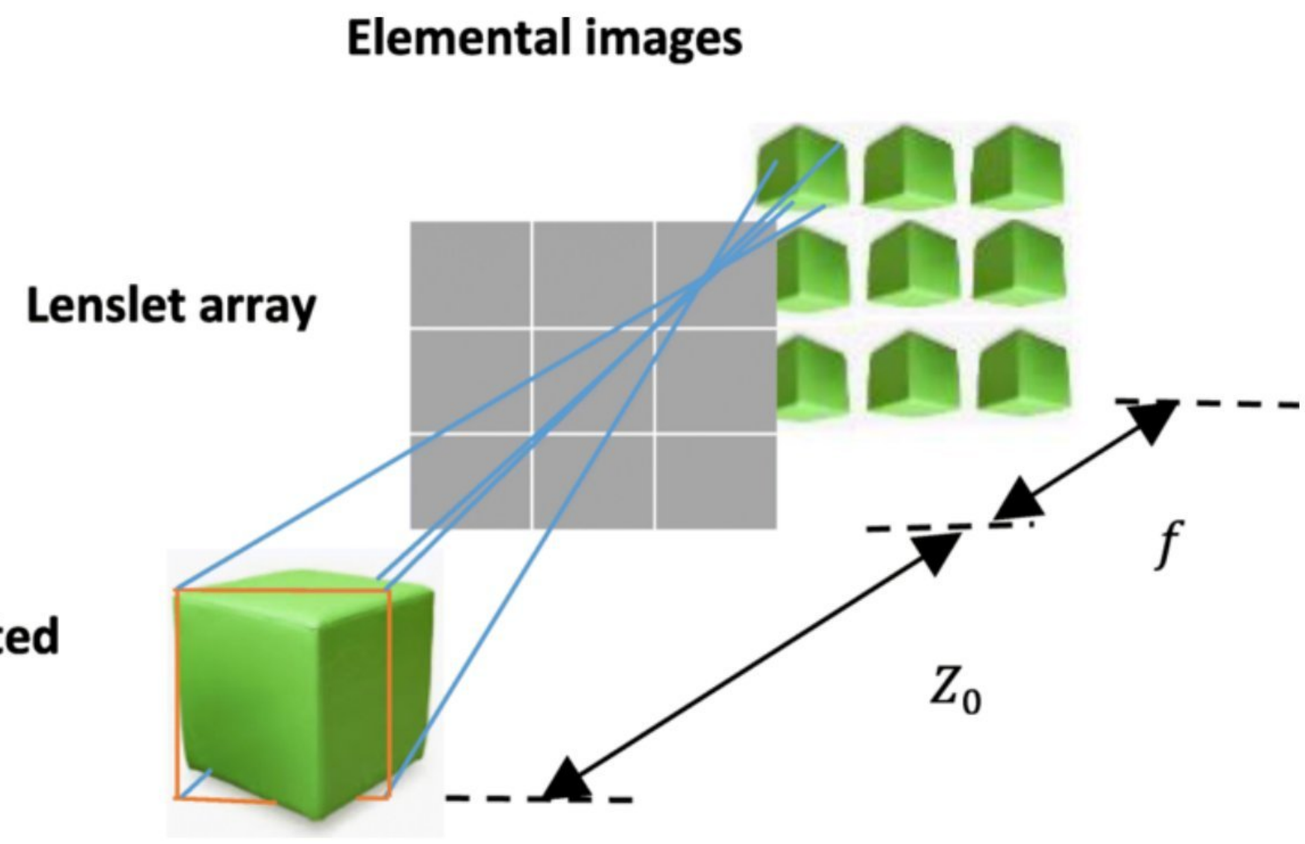


Fig. 2.5 : Visualisation de la projection inverse d'une carte de profondeur en points 3D. [52]

2.3.2.3 Rôle exact des CNN dans cette tâche

Les CNN assurent plusieurs fonctions critiques :

- Analyse du contexte local et global pour détecter les variations fines dans les textures et l'éclairage.
- Apprentissage des motifs visuels liés à la profondeur (comme les objets proches étant plus nets).
- Modélisation de la géométrie implicite à travers de grands jeux de données.
- Génération d'une carte de profondeur dense et précise à l'aide de couches convolutives et d'opérations de pooling successives.

2.3.2.4 Avantages de l'utilisation des CNN pour la reconstruction 3D

- Ne nécessite pas de capteurs coûteux ou de données 3D annotées.
- Fonctionne avec des images monoculaires (caméra simple).
- Possibilité d'inférence en temps réel pour des applications pratiques (AR, robotique, etc.).
- Capacité d'apprentissage automatique des indices de profondeur implicites, permettant une généralisation à de nouvelles scènes.
- Intégration facile dans des systèmes existants grâce à leur compatibilité avec des architectures matérielles courantes (GPU, edge devices, etc.).

2.3.3 Autoencodeurs dans la Reconstruction 3D

Les autoencodeurs jouent un rôle fondamental dans la reconstruction tridimensionnelle (3D) à partir de données bidimensionnelles (2D), grâce à leur capacité à apprendre des représentations latentes compactes et pertinentes. En comprimant l'information dans un goulot d'étranglement (bottleneck), ils permettent de capturer des structures spatiales complexes, facilitant ainsi la prédiction de formes 3D à partir d'images 2D[53]

Des architectures avancées comme les autoencodeurs convolutifs (CAE) ou variationnels (VAE) sont fréquemment utilisées pour modéliser la profondeur, générer des volumes voxelisés ou reconstruire des maillages à partir d'images simples ou stéréo.[54]

En incorporant des couches de décodage adaptées, ces modèles peuvent produire des reconstructions géométriquement cohérentes.

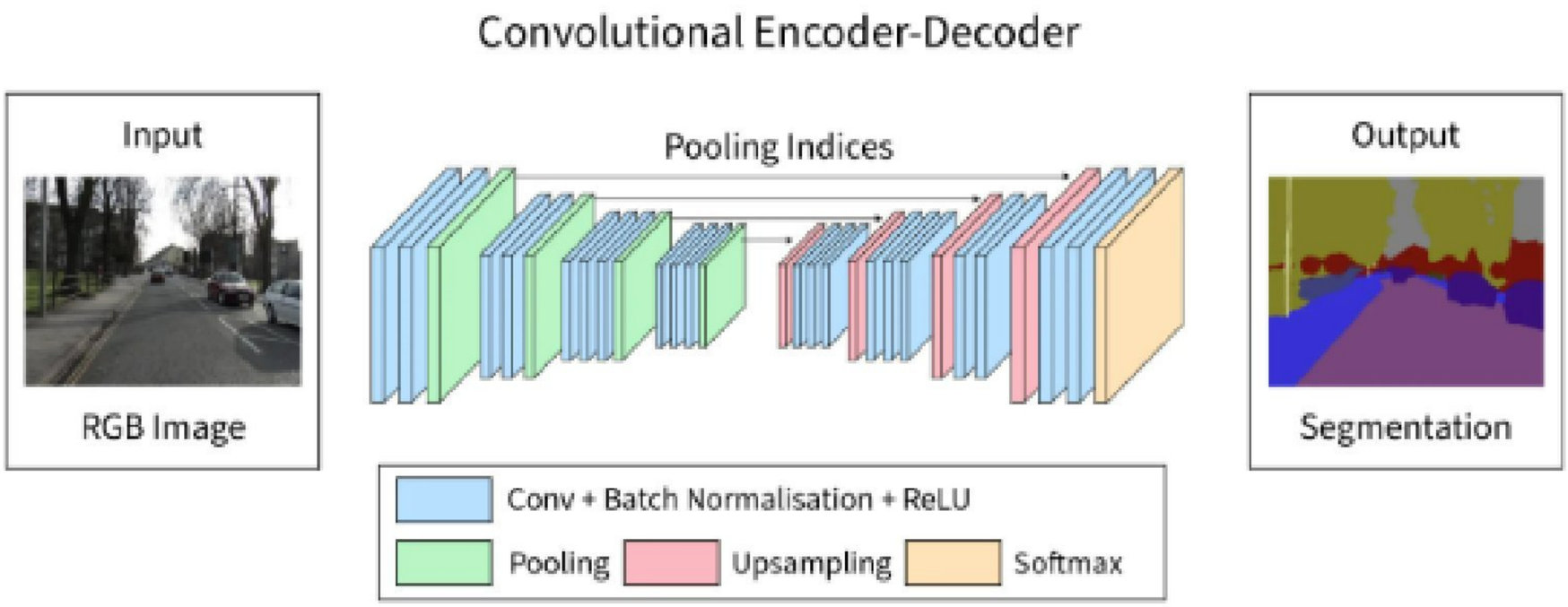


Fig. 2.6 : Schéma de l'architecture d'un autoencodeur utilisé pour l'estimation de la profondeur [55]

2.3.3.1 Les Skip Connections dans les Autoencodeurs

Les *skip connections*, également appelées connexions de saut, sont des mécanismes architecturaux qui permettent de transférer directement l'information entre des couches non consécutives d'un réseau neuronal profond. Contrairement à une architecture classique où l'information circule de manière strictement séquentielle, les skip connections contournent certaines couches en reliant les couches d'entrée à des couches plus profondes.

Dans le contexte des autoencodeurs, cette stratégie est particulièrement utile pour préserver les détails fins de l'image d'entrée lors de la phase de décodage. En transférant les caractéristiques locales extraites par l'encodeur directement vers le décodeur, les skip connections permettent une meilleure reconstruction, notamment en réduisant la perte d'information due à la compression dans le *bottleneck*.

Cette approche a été popularisée par des architectures comme U-Net [56], initialement conçue pour la segmentation d'images biomédicales, mais largement adoptée dans des tâches de reconstruction et de génération. Dans notre projet, l'intégration des skip connections a permis d'améliorer significativement la qualité des images reconstruites, en particulier dans les détails structurels et les textures fines.

2.3.3.2 Applications récentes des autoencodeurs dans la reconstruction 3D

L'évolution des autoencodeurs les place aujourd'hui au cœur des avancées en vision par ordinateur et en modélisation 3D. Leur capacité à apprendre des représentations efficaces a transformé la manière dont les objets et scènes 3D sont reconstruits à partir d'entrées 2D. Voici quelques exemples concrets d'applications récentes :

- Reconstruction 3D à partir de données 2D limitées : Les autoencodeurs, notamment les

autoencodeurs variationnels (VAE), sont utilisés pour générer des modèles 3D de haute qualité à partir d'un nombre limité d'images 2D. Cette approche est particulièrement utile dans des contextes où la collecte de données 3D complètes est difficile ou coûteuse.

- Prédiction de caractéristiques 3D à partir de nuages de points :Des modèles d'autoencodeurs sont appliqués à des nuages de points 3D pour prédire des caractéristiques telles que les normales de surface et les variations de surface. Cette capacité est cruciale pour des applications comme la modélisation 3D de scènes complexes et la réalité augmentée.
- Génération de formes 3D à partir de données volumétriques : Les autoencodeurs génératifs, tels que les autoencodeurs généralisés (GAE), sont utilisés pour la génération de formes 3D volumétriques. Ces modèles apprennent un espace latent structuré qui permet de générer de nouvelles formes en interpolant dans cet espace, offrant ainsi des applications dans la conception assistée par ordinateur et la fabrication additive.
- Réparation de surfaces 3D incomplètes : Des réseaux neuronaux basés sur des autoencodeurs sont employés pour la reconstruction de surfaces 3D à partir de modèles incomplets. En utilisant des informations de courbure et des techniques d'inpainting, ces modèles peuvent remplir les zones manquantes de manière réaliste, ce qui est essentiel pour des applications comme l'impression 3D et la numérisation du patrimoine culturel.

En conclusion, les autoencodeurs constituent une approche robuste et polyvalente pour la reconstruction tridimensionnelle à partir d'images bidimensionnelles, en facilitant l'extraction et l'exploitation de représentations latentes significatives. Le chapitre suivant présentera l'implémentation concrète de cette méthode dans le cadre de notre projet, en détaillant l'architecture adoptée, les choix liés à l'entraînement du modèle, ainsi que l'évaluation des résultats obtenus

2.3.4 Les Réseaux Antagonistes Génératifs dans la Reconstruction 3D

Les Réseaux Antagonistes Génératifs (GANs) sont devenus une solution puissante pour l'estimation de la profondeur à partir d'images RGB uniques. Contrairement aux méthodes classiques nécessitant des paires stéréo ou des données LiDAR, les GANs permettent une estimation non supervisée ou faiblement supervisée de la profondeur, en exploitant leur architecture compétitive pour apprendre des représentations complexes.

Un GAN est composé de deux réseaux :

Le générateur (G) : qui prend une image 2D en entrée et prédit une carte de profondeur correspondante.

Le discriminateur (D) : qui évalue si la carte de profondeur générée est réaliste ou non, en la comparant à des cartes de profondeur réelles (lorsqu'elles sont disponibles) ou en utilisant des contraintes géométriques.

Dans le contexte de la reconstruction 3D, le générateur apprend à projeter chaque pixel de l'image d'entrée dans l'espace tridimensionnel, en générant une carte de profondeur fidèle. Cette carte peut ensuite être utilisée pour reconstruire la scène 3D sous forme de nuage de points, maillage polygonal ou grille voxelisée.

Le processus d'entraînement repose sur une combinaison de fonctions de perte, notamment :

- la perte L1 pour mesurer la différence pixel à pixel entre la carte générée et la vérité terrain,
- la perte SSIM (Structural Similarity Index) pour préserver la structure spatiale de l'image,
- la perte adversariale fournie par le discriminateur, qui pousse le générateur à produire des cartes de profondeur de plus en plus réalistes.

Certains modèles récents, comme Multi-Scale GAN ou DepthGAN, utilisent des générateurs multi-résolution pour capturer à la fois les détails locaux et globaux de la scène. Ces approches hiérarchiques améliorent la précision de l'estimation de la profondeur, même dans des zones à faible texture ou avec des objets fins.

En résumé, les GANs permettent de passer d'une représentation 2D à une compréhension tridimensionnelle de la scène, en apprenant directement la structure spatiale à partir d'images RGB. Cette approche est particulièrement utile pour les applications en vision par ordinateur, robotique, réalité virtuelle et systèmes autonomes.

2.3.5 État de L'Art sur La conversion 2D vers 3D

Dans cette section , nous citons quelque travaux président dans le domaine du La conversion 2D vers 3D.

Godard et al. (2017)

L'équipe de Godard a développé une méthode permettant d'estimer la profondeur à partir d'une seule image 2D en utilisant un réseau neuronal convolutif (CNN). Le modèle apprend à prédire la carte de profondeur pour chaque pixel de l'image 2D donnée, en se basant sur les caractéristiques visuelles extraites.

Résultats : Le modèle a montré une capacité notable à estimer la profondeur dans des scènes complexes, en particulier dans des environnements urbains. Les résultats ont montré une bonne précision de la profondeur, mais le modèle a montré des difficultés dans les scènes très détaillées ou avec des objets en arrière-plan.

Défis : Les performances du modèle étaient dégradées lorsqu'il y avait des objets multiples superposés, ou des variations d'éclairage importantes dans l'image. Ces conditions ont rendu la tâche de reconstruction de la profondeur plus difficile, nécessitant des approches d'optimisation supplémentaires.

Vincent et al. (2008)

Afin de rendre les autoencodeurs plus robustes face aux données bruitées ou corrompues, Vincent et ses collègues ont proposé le Denoising Autoencoder. Ce modèle consiste à corrompre intentionnellement les données d'entrée puis à entraîner le réseau pour qu'il les reconstruise correctement. Cette approche a permis d'améliorer significativement la qualité de reconstruction des données même en présence de bruit. Toutefois, la généralisation du modèle à différents types

de bruit reste un défi majeur, ce qui nécessite souvent des ajustements spécifiques en fonction des applications visées.

Kingma & Welling (2014)

Kingma et Welling ont proposé le Variational Autoencoder (VAE), une approche probabiliste qui combine l'apprentissage profond et les modèles génératifs. Leur objectif était de permettre non seulement la compression et la reconstruction des données, mais également la génération de nouvelles données synthétiques réalistes. Le VAE offre la possibilité de modéliser des distributions latentes continues et de générer des échantillons variés à partir de ces distributions. Néanmoins, il existe un compromis entre la qualité de reconstruction et la diversité des données générées, ce qui constitue un défi important lors de l'entraînement du modèle.

Hinton Salakhutdinov (2006)

Dans une autre contribution majeure, Hinton et Salakhutdinov ont exploré l'utilisation des autoencodeurs pour la réduction de la dimensionnalité des données volumineuses. Leur approche a permis de réduire efficacement le nombre de dimensions tout en conservant une grande partie de l'information initiale, facilitant ainsi les tâches de classification et de visualisation. Cependant, concevoir l'architecture de l'autoencodeur de manière optimale pour chaque type de données demeure un défi à surmonter, notamment dans les contextes où les données sont très hétérogènes.

Zhao et al. (2017)

Zhao et al. ont proposé l'Adversarial Autoencoder (AAE), qui intègre les concepts des réseaux antagonistes génératifs (GAN) avec les autoencodeurs classiques. Cette approche vise à améliorer la qualité de la représentation latente et à renforcer la capacité du modèle à générer de nouvelles données réalistes. Bien que l'AAE permette d'obtenir des résultats prometteurs en termes de qualité de génération et de fidélité des représentations, l'entraînement est particulièrement complexe en raison de la compétition entre les différents réseaux (générateur et discriminateur), ce qui peut rendre le modèle difficile à stabiliser.

2.4 Conclusion

Ce chapitre a mis en évidence la transition progressive des méthodes classiques de reconstruction 3D vers des approches plus modernes et performantes basées sur le deep learning. Tandis que les techniques traditionnelles reposent fortement sur des hypothèses géométriques et une calibration précise, les réseaux neuronaux offrent plus de flexibilité, d'automatisation et de robustesse face aux variations des données. Cette évolution technologique ouvre de nouvelles perspectives pour la reconstruction 3D, qui seront exploitées et implémentées dans notre propre système présenté dans le chapitre suivant.

Chapitre 3

Implémentation d'un Système de Reconstruction 3D avec U-Net

Chapitre 3

Implémentation d'un Système de Reconstruction 3D avec U-Net

3.1 Introduction

Ce dernier chapitre est consacré à la mise en œuvre pratique des connaissances acquises dans les chapitres précédents. Il décrit le processus de conception et de réalisation de notre système de reconstruction 3D à partir d'images 2D extraites de vidéos monoculaires. Nous avons utilisé un modèle basé sur un autoencodeur avec des skip connections, ce qui permet de conserver les informations spatiales importantes et d'améliorer la qualité des reconstructions. Nous expliquons également les étapes de préparation des données, les choix techniques faits, ainsi que les expérimentations réalisées pour évaluer la performance du modèle et analyser les résultats obtenus.

3.2 Description du Système

Notre système de reconstruction repose sur l'utilisation d'un autoencodeur convolutionnel avec connexions de saut (skip connections), ce qui permet de mieux préserver les détails fins lors de la reconstruction. Le système comprend plusieurs étapes principales :

- Extraction des images : À partir de vidéos 2D, on extrait des images à intervalles réguliers pour éviter la redondance.
- Prétraitement : (Redimensionnement et Normalisation)
- Création des paires d'apprentissage : (x_data, y_data) où y représente une image légèrement décalée dans le temps par rapport à x.
- Entraînement du modèle : Un autoencodeur entraîné pour prédire y à partir de x.
- Génération progressive des image 2D
- Reconstruction 3D (Création des vidéos)

3.3 Architecture Générale du Système

La figure 3.1 représente l'architecture globale de notre système :

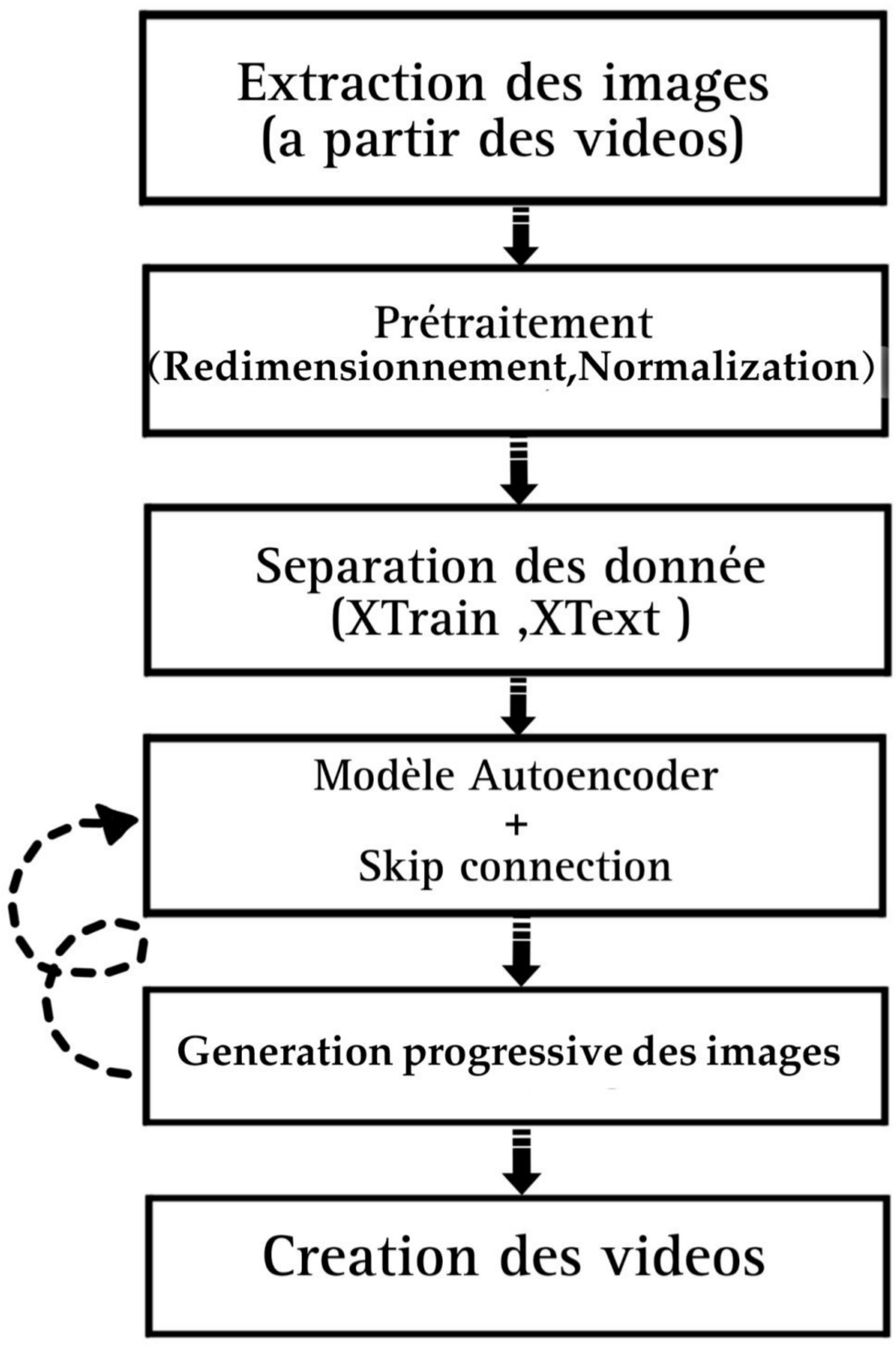


Fig. 3.1 : Architecture generale du système

3.4 Préparation de la Base de Données

3.4.1 Collecte et extraction des données brutes

Dans le cadre de ce projet, la première étape essentielle a été la constitution d'une base de données adaptée pour l'apprentissage du modèle d'autoencodeur destiné à la reconstruction vidéo en 3D. Pour cela, nous avons récupéré 100 vidéos sur la plateforme YouTube. Ces vidéos, sélectionnées soigneusement, portent sur des scènes variées afin de garantir une diversité suffisante des données.

Une fois les vidéos téléchargées, nous avons procédé à leur extraction en images indivi-

duelles (frames). Cette opération a été réalisée à l'aide de la bibliothèque OpenCV, qui permet d'extraire chaque image à une fréquence spécifique pour constituer une séquence ordonnée d'images. Ces images ont été sauvegardées dans un dossier structuré nommé `video3d`. Ce dossier contient 100 sous-dossiers nommés de manière séquentielle de `simple001` à `simple100`. Chaque sous-dossier correspond à une vidéo unique, et à l'intérieur, les images sont numérotées de manière croissante (`001.jpg`, `002.jpg`, etc.), garantissant ainsi un accès chronologique et organisé aux données.

Cette organisation en dossiers multiples est cruciale pour une gestion efficace des données, notamment pour la séparation en ensembles d'entraînement et de test, mais aussi pour permettre un traitement batch lors de la phase d'apprentissage.

3.4.2 Prétraitement des images et constitution des paires d'entraînement

Afin d'adapter ces images à notre modèle de deep learning, plusieurs opérations de prétraitement ont été effectuées :

- **Redimensionnement :** Chaque image extraite a été redimensionnée à une taille fixe de 128x128 pixels, afin de standardiser les entrées du réseau de neurones. Ce choix de taille est un compromis entre la qualité de l'information visuelle et la limitation des ressources computationnelles nécessaires à l'entraînement.
- **Conversion RGB :** Les images ont été converties en mode RGB pour conserver la richesse chromatique, indispensable à la reconstruction fidèle des scènes.
- **Normalisation :** Pour faciliter la convergence du réseau, les valeurs des pixels ont été normalisées en flottants compris entre 0 et 1, ce qui est une pratique standard dans le traitement d'images par réseaux de neurones.

Par ailleurs, pour entraîner notre autoencodeur à prédire une image future à partir d'une image donnée, nous avons constitué des paires d'images (`x_data` et `y_data`) avec un intervalle temporel de 4 images (paramètre `image_step=4`). Concrètement, pour chaque vidéo, l'image numéro `i` (appelée entrée) est associée à l'image numéro `i + 4` (appelée cible). Cela permet au modèle d'apprendre à prédire une version future de la scène, ce qui est une base essentielle dans la reconstruction temporelle en 3D.

La boucle d'extraction suit la logique suivante :

- Pour chaque dossier `simpleXXX`,
- On parcourt les images par pas de 4,
- On récupère l'image courante et l'image distante de 4 frames,
- On ajoute ces images aux listes `x_data` et `y_data`.

Cette méthode a permis d'extraire un total de **6131 paires d'images**, offrant ainsi un corpus riche et varié pour l'entraînement.

3.4.3 Séparation des données en ensembles d'entraînement et de test

Une étape clé en apprentissage automatique est la séparation des données en deux ensembles distincts :

- **Ensemble d'entraînement** : Utilisé pour entraîner le modèle et ajuster ses paramètres.
- **Ensemble de test** : Utilisé pour évaluer la capacité du modèle à généraliser sur des données inconnues.

Pour cela, nous avons sélectionné cinq dossiers spécifiques (`simple020`, `simple003`, `simple038`, `simple060`, `simple004`) qui composent l'ensemble de test. Ces dossiers ont été choisis afin de fournir des exemples suffisamment variés et représentatifs, mais distincts de ceux utilisés pour l'entraînement.

Dans ces dossiers de test, seule la première paire d'images (correspondant au premier intervalle de 4 frames) a été retenue pour valider la capacité de reconstruction du modèle. Cette sélection rigoureuse évite la contamination des données et assure un vrai test de généralisation.

Les autres dossiers, soit 95% de la base, ont été utilisés pour l'entraînement. Ainsi, la séparation finale est la suivante :

- **Ensemble d'entraînement :** 6126 paires d'images (taille `(6126, 128, 128, 3)`)
- **Ensemble de test :** 5 paires d'images (taille `(5, 128, 128, 3)`)

Cette forte disproportion est justifiée par la volonté d'entraîner le modèle sur un maximum de données tout en conservant un échantillon test strict et non biaisé.

3.4.4 Sauvegarde des données préparées

Pour faciliter les phases ultérieures d'entraînement et de test, les tableaux numpy `x_train`, `y_train`, `x_test` et `y_test` ont été sauvegardés sur Google Drive sous forme de fichiers `.npy`. Cette méthode permet de charger rapidement les données en mémoire sans refaire le prétraitement à chaque exécution du programme.

La structure des fichiers est la suivante :

```
/content/drive/MyDrive/mon_projet/x_train.npy
/content/drive/MyDrive/mon_projet/y_train.npy
/content/drive/MyDrive/mon_projet/x_test.npy
/content/drive/MyDrive/mon_projet/y_test.npy
```

Cette organisation garantit reproductibilité, traçabilité et facilite la collaboration en cas de travail partagé.

3.5 Présentation du modèle Autoencoder classique

Avant de concevoir une architecture avancée avec skip connections, nous avons commencé par implémenter un autoencodeur classique, basé sur une structure symétrique d'encodage et de décodage. L'objectif était de tester la capacité de cette architecture de base à reconstruire les

images, afin de poser un point de comparaison avec le modèle amélioré que nous présenterons dans la section suivante.

L'autoencodeur classique repose uniquement sur des couches convolutionnelles et de pooling pour encoder l'image, suivies d'un processus inverse de décodage basé sur des couches de convolution et d'UpSampling.

3.5.1 Architecture détaillée du modèle

L'architecture complète de cet autoencodeur est résumée dans le tableau ci-dessous :

Tab. 3.1 : Architecture complète de l'Autoencodeur Classique

Couche (type)	**Dimension de sortie**	**Paramètres**
Input (Image 128×128×3)	(None, 128, 128, 3)	0
Conv2D (32 filtres)	(None, 128, 128, 32)	896
MaxPooling2D	(None, 64, 64, 32)	0
Conv2D (64 filtres)	(None, 64, 64, 64)	18,496
MaxPooling2D	(None, 32, 32, 64)	0
Conv2D (128 filtres)	(None, 32, 32, 128)	73,856
MaxPooling2D	(None, 16, 16, 128)	0
Conv2D (128 filtres)	(None, 16, 16, 128)	147,584
UpSampling2D	(None, 32, 32, 128)	0
Conv2D (64 filtres)	(None, 32, 32, 64)	73,792
UpSampling2D	(None, 64, 64, 64)	0
Conv2D (32 filtres)	(None, 64, 64, 32)	18,464
UpSampling2D	(None, 128, 128, 32)	0
Conv2D (3 canaux RGB)	(None, 128, 128, 3)	867

Le modèle contient en tout environ 352 955 paramètres, dont la majorité sont concentrés dans les couches convolutives profondes (notamment celles à 128 filtres). Le fonctionnement du modèle se décompose en deux étapes principales :

Phase d'encodage

Dans cette phase, l'image est progressivement réduite en taille à l'aide de couches MaxPooling, tout en augmentant la profondeur (nombre de filtres). Cette compression spatiale permet de capturer les caractéristiques principales de l'image dans un espace latent de dimension plus faible .

Phase de décodage

Le décodage se fait en utilisant des couches UpSampling2D qui restaurent progressivement les dimensions d'origine de l'image. Chaque étape d'agrandissement est suivie par une convolution pour réinterpréter les caractéristiques latentes. les figure suivante présente une comparai-

son entre une image d'entrée et sa première et derniére reconstruction obtenue en utilisant un Autoencoder classique.

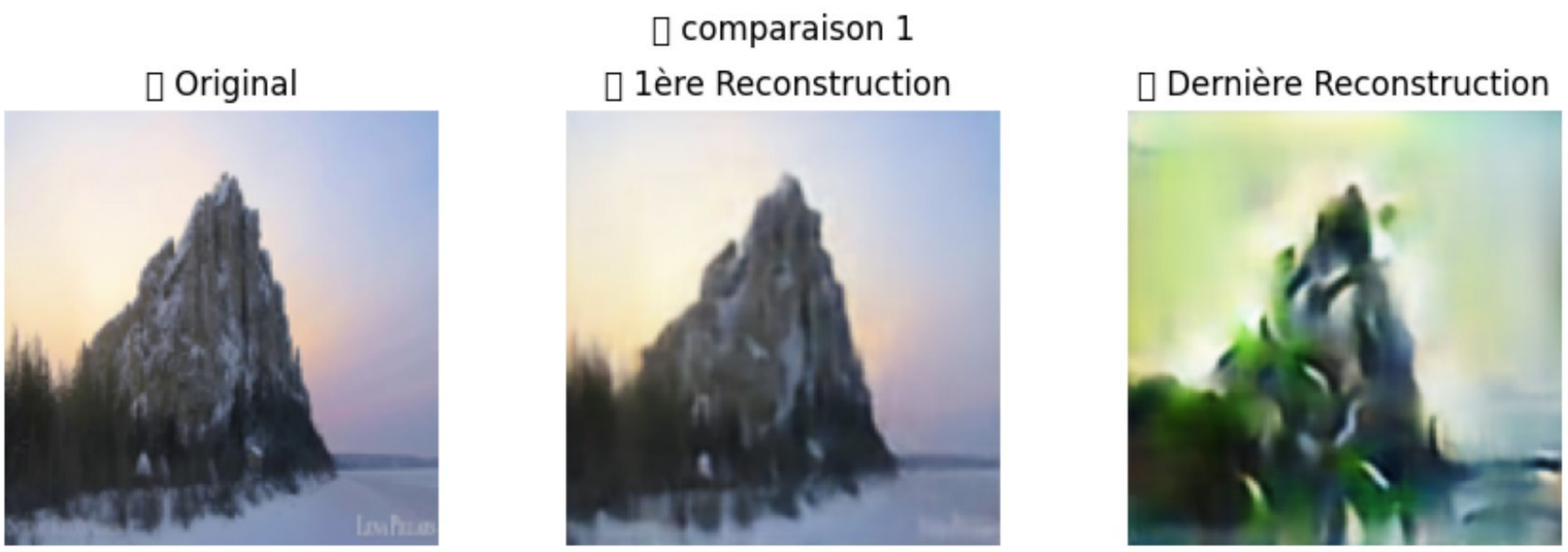


Fig. 3.2 : Comparaison entre image original premiere reconstruction et derniere reconstruction (image 1)

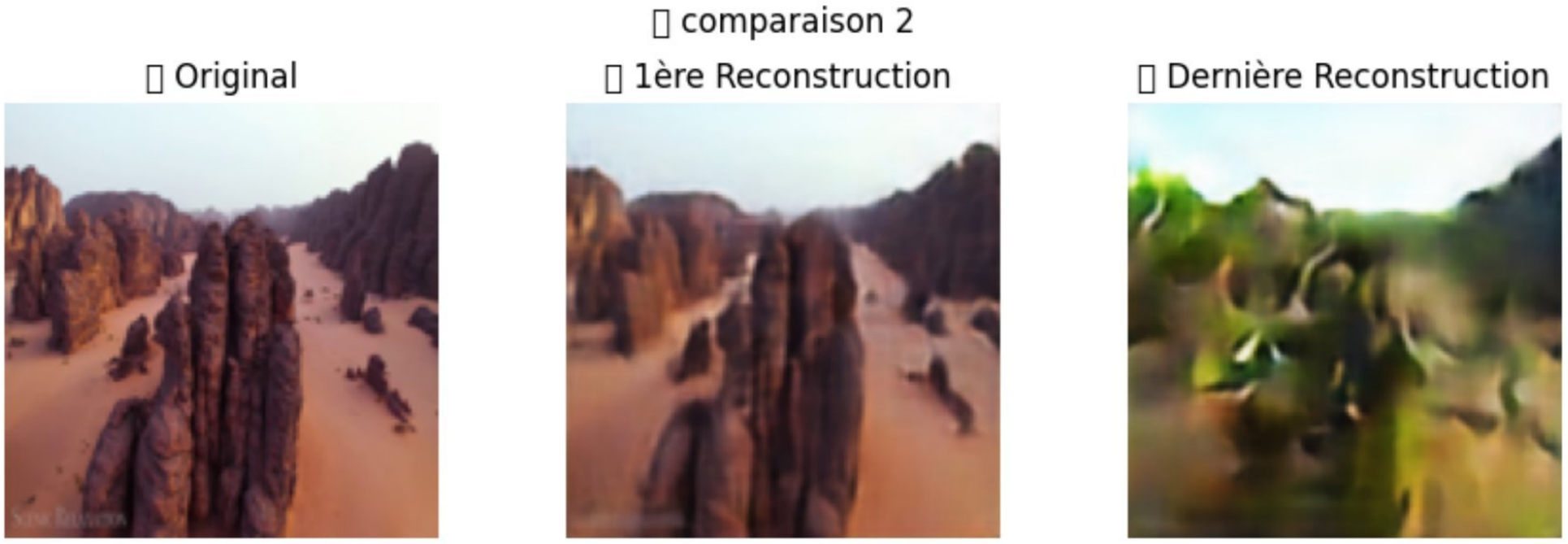


Fig. 3.3 : Comparaison entre image original premiere reconstruction et derniere reconstruction (image 2)

3.5.1.1 Analyse des résultats

Comme illustré dans les figures 3.2 et 3.3, l'autoencodeur classique démontre une capacité limitée à maintenir la qualité des reconstructions à travers les itérations :

Lors de la première reconstruction, l'image produite reste relativement proche de l'originale. Les formes globales sont bien conservées, bien que l'on note déjà une légère perte de netteté et une atténuation des textures fines.

Au fur et à mesure des reconstructions (après plusieurs itérations), l'image devient de plus en plus floue et déformée. Les contours se brouillent, les couleurs deviennent artificielles, et l'ensemble visuel perd en cohérence. Cela suggère une accumulation progressive d'erreurs, et un manque de capacité du modèle à préserver les détails visuels importants.

Ces résultats soulignent plusieurs limitations clés du modèle classique :

- Pas de mécanisme de récupération des détails, ce qui entraîne une perte irréversible d'information.
- Propagation des erreurs d'une étape à l'autre sans correction.

3.5.1.2 Calcul des métriques SSIM et PSNR

Pour évaluer quantitativement la qualité des images reconstruites par notre Autoencoder, nous avons calculé deux métriques standards en traitement d'image :

- SSIM (Structural Similarity Index Measure) : Cette métrique mesure la similarité perceptuelle entre l'image originale et l'image reconstruite, en se basant sur la structure, la luminance et le contraste. Elle varie entre 0 (aucune similarité) et 1 (images identiques). La formule mathématique du SSIM est donnée par :

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3.1}$$

  μ_x et μ_y sont les moyennes des images x et y.
  σ_x^2 et σ_y^2 leurs variances.
  σ_{xy} la covariance entre les deux images.
  C_1, C_2 des constantes de stabilisation pour éviter les divisions par zéro.

- PSNR (Peak Signal-to-Noise Ratio) : Cette métrique quantifie le rapport entre la puissance du signal original et le bruit introduit par la reconstruction. Plus la valeur est élevée, meilleure est la qualité de la reconstruction.Il est exprimé en décibels (dB) et est défini par :

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \tag{3.2}$$

  MAX_I est la valeur maximale possible pour un pixel (255 pour une image en 8 bits).
  MSE (Mean Squared Error) représente l'erreur quadratique moyenne entre l'image originale et l'image reconstruite.

Les calculs ont été effectués pour chaque image de test et pour chaque itération de reconstruction (au total 15 itérations). Pour cela, les images originales (x_test) ont été converties en format uint8 (valeurs entre 0 et 255) afin d'assurer une comparaison cohérente avec les images reconstruites sauvegardées.

Les fonctions ssim et psnr du module skimage.metrics ont été utilisées pour mesurer la similarité et la qualité des reconstructions, ce qui nous a permis de quantifier précisément la fidélité des images générées par le modèle.

3.5.1.3 Analyse des indices SSIM et PSNR

Les résultats des métriques SSIM et PSNR, calculés sur les 15 itérations de reconstruction pour chaque image test, ont été enregistrés sous forme de matrices. Ces données permettent

d'observer la tendance générale de la qualité de reconstruction au fil des passages successifs dans l'Autoencoder.

3.5.1.4 Visualisation graphique

Les courbes de l'évolution moyenne du SSIM et du PSNR sur les 15 itérations ont été tracées afin d'illustrer l'impact des itérations répétées sur la qualité des reconstructions. Les figure ci-dessous montre une évolution claire des deux indicateurs au fil des itérations.

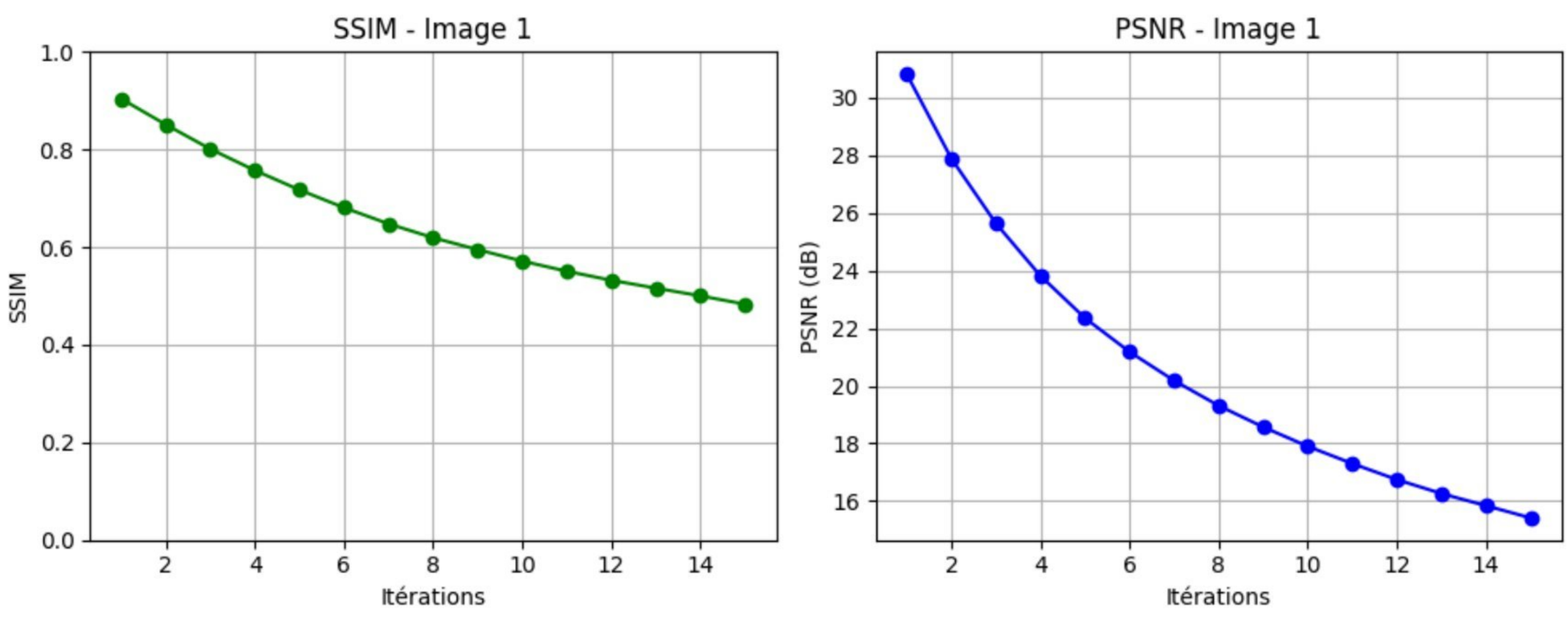


Fig. 3.4 : Évolution du SSIM et du PSNR sur les 15 itérations(image 1)

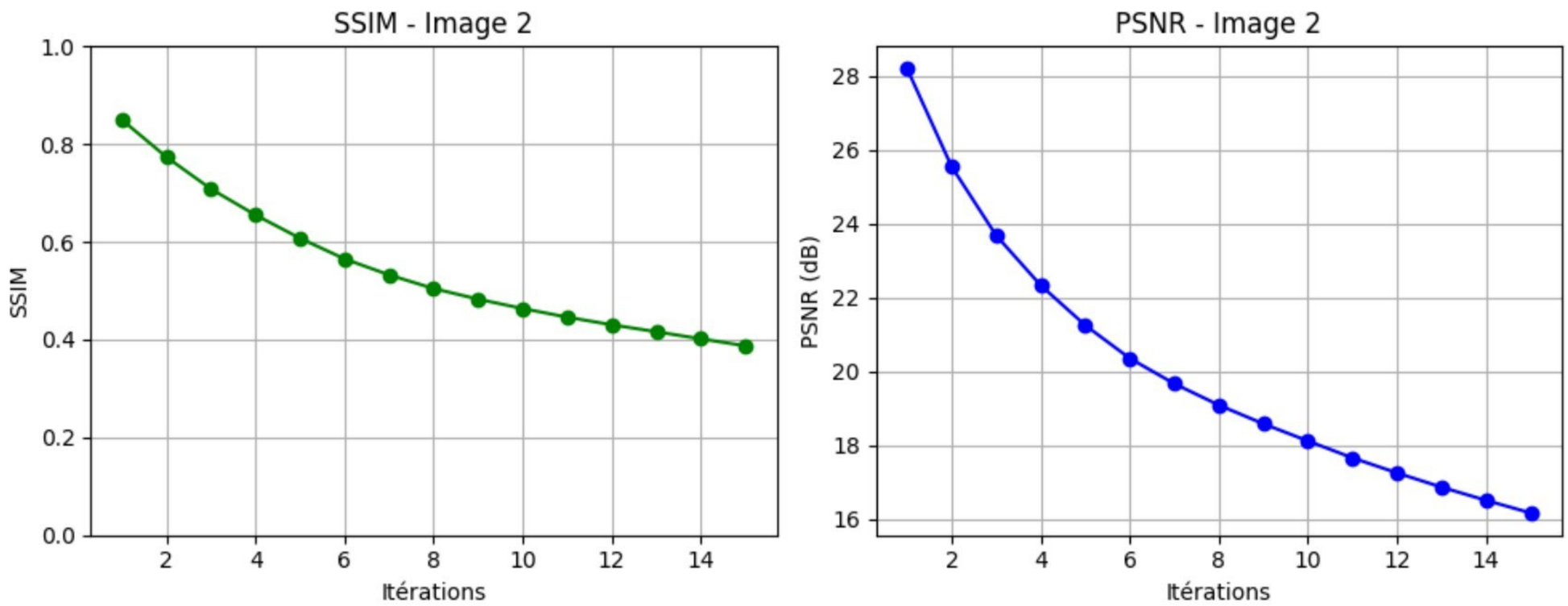


Fig. 3.5 : Evolution du SSIM et du PSNR sur les 15 itérations(image 2)

3.5.2 conclusion

L'autoencodeur classique constitue une base expérimentale pertinente pour étudier le processus de reconstruction d'images. Cependant, les limites observées, notamment en termes de perte de détails et de qualité dégradée au fil des itérations, ont mis en évidence la nécessité d'une approche plus performante. C'est dans cette optique que nous avons introduit une architecture améliorée intégrant des connexions de saut (skip connections), visant à renforcer la fidélité de

la reconstruction en conservant les informations spatiales essentielles. Cette approche sera détaillée dans la section suivante.

3.6 Présentation du modèle Autoencoder avec U-Net

Afin de surmonter les limitations observées avec l'autoencodeur classique, notamment la perte progressive des détails fins lors des reconstructions itératives, nous avons opté pour une architecture améliorée intégrant des connexions de saut (skip connections).

3.6.1 Objectif du modèle :

L'objectif principal de notre Autoencodeur avec Skip Connections est d'améliorer la qualité de la reconstruction en préservant un maximum d'informations spatiales à travers les différentes couches du réseau.

En effet, lorsqu'un autoencodeur classique encode une image via plusieurs couches convolutives et opérations de réduction dimensionnelle (MaxPooling), il tend à perdre certaines informations cruciales liées aux textures fines, aux contours précis et aux petits objets présents dans l'image d'origine.

3.6.2 Principe des Skip Connections :

Les skip connections sont des liaisons directes établies entre les couches correspondantes de l'encodeur et du décodeur. Elles permettent de transférer les caractéristiques de bas niveau extraites dans les premières couches vers les couches de reconstruction, contournant ainsi le goulot d'étranglement (bottleneck).

Ce mécanisme présente plusieurs avantages :

- Il permet au décodeur de bénéficier d'un accès direct aux informations locales perdues dans les représentations compressées.
- Il améliore la fidélité des images reconstruites par rapport aux originales.
- Il facilite l'apprentissage du modèle en réduisant la profondeur effective du chemin de rétropropagation (gradient flow).
- Il assure une reconstruction plus stable et plus détaillée, même après plusieurs itérations successives.

3.6.3 Présentation des variantes architecturales testées

Avant de finaliser l'architecture du modèle U-Net, nous avons testé plusieurs combinaisons de couches afin d'identifier celle qui permet la meilleure qualité de reconstruction.

Les variantes testées sont résumées dans le tableau ci-dessous :

Tab. 3.2 : Comparaison des variantes de l'architecture

Variante	Type de reconstruction (décodeur)	Taille des noyaux (kernel)
V1	Conv2D + UpSampling2D	3×3
V2	Conv2D + UpSampling2D	5×5
V3	Conv2D + Conv2DTranspose	3×3
V4	Conv2D + Conv2DTranspose	5×5

Pour chaque variante, nous avons appliqué le même protocole expérimental basé sur 15 itérations de reconstruction sur une image de test. Les métriques SSIM et PSNR ont été calculées à chaque itération afin de quantifier la fidélité des reconstructions.

3.6.4 Analyse comparative des variantes

L'analyse comparative des différentes variantes architecturales testées nous a permis de tirer plusieurs enseignements importants concernant le choix des couches de décodage et la taille des noyaux :

UpSampling2D est une méthode simple de redimensionnement qui se contente d'agrandir l'image sans créer de nouveaux détails — un peu comme un zoom basé sur interpolation. Elle n'apprend pas de nouvelles structures, ce qui lui permet de préserver fidèlement la forme globale de l'image d'origine sans introduire d'artéfacts. Cette stabilité est particulièrement avantageuse dans le contexte de reconstructions itératives, où la fidélité visuelle est prioritaire.

À l'inverse, Conv2DTranspose est une méthode "apprenante" qui génère des détails lors du redimensionnement. Bien qu'elle soit utile dans des contextes comme la génération d'images ou la segmentation, son comportement devient problématique lorsque le modèle est appliqué plusieurs fois sur la même image : à chaque itération, de nouveaux détails artificiels s'accumulent, ce qui entraîne une dégradation progressive de l'image (bruit, flou, artefacts).

En ce qui concerne la taille des noyaux, nous avons constaté que les noyaux 3×3 offrent un meilleur compromis entre précision et stabilité. Les noyaux 5×5, bien qu'ils capturent une zone plus large, ont tendance à sur-généraliser et à lisser les détails fins, ce qui nuit à la fidélité de la reconstruction.

En conclusion , la combinaison Conv2D (3×3) + UpSampling2D s'est révélée la plus efficace dans notre contexte : elle assure une reconstruction fidèle, sans amplification des erreurs à travers les itérations, tout en maintenant une bonne conservation des détails locaux.

3.6.5 Architecture détaillée du modèle retenu

Suite à l'analyse comparative des différentes variantes de l'architecture, nous avons retenu la combinaison Conv2D (3×3) + UpSampling2D, qui a démontré la meilleure stabilité et fidélité de reconstruction selon les indicateurs SSIM et PSNR.

Nous avons ainsi adopté une architecture inspirée du modèle U-Net, largement utilisé dans les domaines du traitement d'images médicales et de la segmentation sémantique. Ce modèle repose sur une structure symétrique d'encodage et de décodage, enrichie par des connexions croisées entre les couches correspondantes, ce qui le rend particulièrement adapté à notre objectif de reconstruction fidèle.

Le tableau suivant résume, couche par couche, les dimensions des sorties et le nombre de paramètres :

Tab. 3.3 : Architecture complète du modèle Autoencoder avec Skip Connections

Layer (type)	**Output Shape**	**Param #**	**Connected to**
input_layer_1 (InputLayer)	(None, 128, 128, 3)	0	-
conv2d_7 (Conv2D)	(None, 128, 128, 64)	1,792	input_layer_1[0]
max_pooling2d_3 (MaxPooling2D)	(None, 64, 64, 64)	0	conv2d_7[0][0]
conv2d_8 (Conv2D)	(None, 64, 64, 128)	73,856	max_pooling2d_3[0][0]
max_pooling2d_4 (MaxPooling2D)	(None, 32, 32, 128)	0	conv2d_8[0][0]
conv2d_9 (Conv2D)	(None, 32, 32, 256)	295,168	max_pooling2d_4[0][0]
max_pooling2d_5 (MaxPooling2D)	(None, 16, 16, 256)	0	conv2d_9[0][0]
conv2d_10 (Conv2D)	(None, 16, 16, 512)	1,180,160	max_pooling2d_5[0][0]
up_sampling2d_3 (UpSampling2D)	(None, 32, 32, 512)	0	conv2d_10[0][0]
concatenate (Concatenate)	(None, 32, 32, 768)	0	up_sampling2d_3[0][0], conv2d_9[0][0]
conv2d_11 (Conv2D)	(None, 32, 32, 256)	1,769,728	concatenate[0][0]
up_sampling2d_4 (UpSampling2D)	(None, 64, 64, 256)	0	conv2d_11[0][0]
concatenate_1 (Concatenate)	(None, 64, 64, 384)	0	up_sampling2d_4[0][0], conv2d_8[0][0]
conv2d_12 (Conv2D)	(None, 64, 64, 128)	442,496	concatenate_1[0][0]
up_sampling2d_5 (UpSampling2D)	(None, 128, 128, 128)	0	conv2d_12[0][0]
concatenate_2 (Concatenate)	(None, 128, 128, 192)	0	up_sampling2d_5[0][0], conv2d_7[0][0]
conv2d_13 (Conv2D)	(None, 128, 128, 64)	110,656	concatenate_2[0][0]
conv2d_14 (Conv2D)	(None, 128, 128, 3)	1,731	conv2d_13[0][0]

Encoder :

Les couches de convolution (Conv2D) utilisent des kernels de taille 3×3, ce qui permet d'extraire efficacement les caractéristiques locales à différentes échelles. Ces couches apprennent des caractéristiques à différents niveaux, tandis que les couches de sous-échantillonnage (MaxPooling2D) réduisent la résolution spatiale afin de capturer les structures globales de l'image.

Decoder :

les couches de suréchantillonnage (UpSampling2D) augmentent progressivement la résolution. À chaque étape, on concatène les sorties du décodeur avec les sorties correspondantes de l'encodeur via des Concatenate, réalisant ainsi les skip connections.

Skip Connections

trois connexions sont établies :

- Entre conv2d_9 (Encoder) et up_sampling2d_3 (Decoder)
- Entre conv2d_8 et up_sampling2d_4
- Entre conv2d_7 et up_sampling2d_5

La figure 3.6 illustre l'architecture complète du modèle Autoencodeur avec connexions de saut. Elle reprend les différentes couches et les connexions de type "skip" mentionnées dans le tableau 3.3.

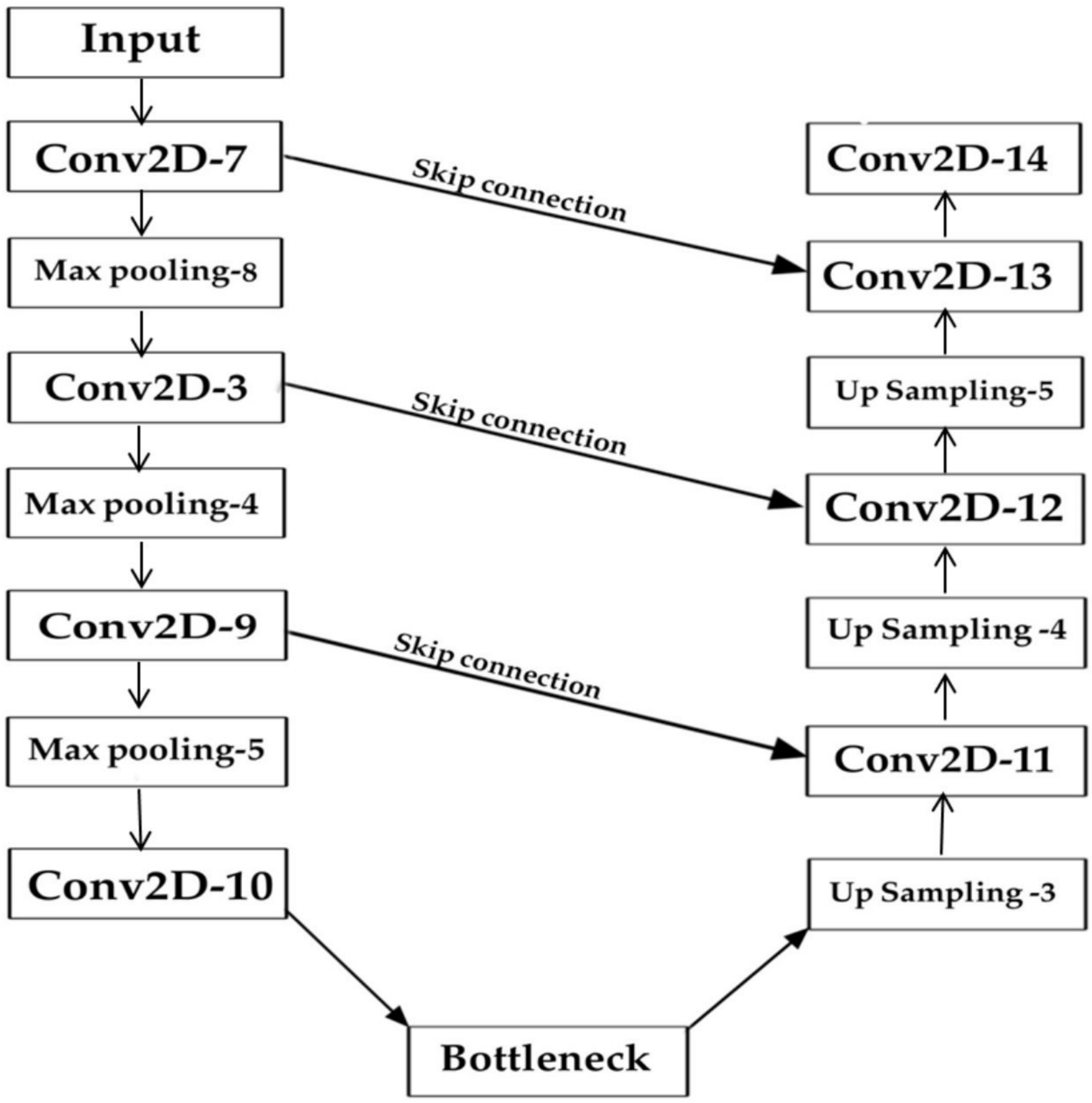


Fig. 3.6 : Architecture complète de l'Autoencodeur avec connexions de saut

3.7 Entraînement du modèle

L'entraînement de notre autoencodeur amélioré (avec Skip Connections) s'est effectué sur les images issues de notre base de données, redimensionnées en 128×128 pixels et normalisées dans l'intervalle [0,1].

Les images d'entrée (x_train) et les cibles (y_train) sont extraites automatiquement à l'aide d'un générateur de données personnalisé, qui lit les images depuis la structure de dossiers sur Google Drive sans les charger toutes en mémoire. Cette méthode permet un entraînement plus efficace et moins gourmand en ressources, notamment lors du traitement de grands ensembles d'images.

Le modèle est entraîné sur :

- époques : 250
- taille de lot (batch size) : 16
- optimiseur : Adam .
- Le choix de la fonction de perte est fondamental : au lieu d'utiliser une perte classique

comme la MSE, nous avons opté pour une perte basée sur l'indice SSIM (Structural Similarity Index) afin de maximiser la similarité structurelle entre les images originales et reconstruites.

Durant l'entraînement, une fonction de rappel (ModelCheckpoint) est utilisée pour sauve garder automatiquement les poids du modèle chaque 5 époques. Les modèles sont enregistrés sur Google Drive pour garantir leur accessibilité à tout moment.

À la fin de l'entraînement, le modèle est sauvegardé dans un fichier .h5 et utilisé pour l'étape d'inférence sur les images de test. Cette configuration garantit une stabilité d'entraînement, une perte progressive maîtrisée, et une capacité du modèle à capturer les structures fines des images.

Pour mieux visualiser l'évolution des performances du modèle durant l'entraînement, nous présentons ci-dessous le graphique de la fonction de perte en fonction du nombre d'époques. Ce suivi permet d'observer la convergence du modèle, ainsi que la capacité de généralisation en comparant la perte sur les données d'entraînement et de validation.

3.8 Test et Génération des Reconstructions

Après avoir entraîné notre Autoencoder sur les données d'entraînement, nous avons procédé à la phase de test afin d'évaluer la qualité des images reconstruites à partir des images originales de test. Cette étape est fondamentale pour vérifier la capacité du modèle à généraliser et à produire des reconstructions fidèles aux images d'entrée, sans avoir vu ces dernières pendant l'entraînement.

Pour cela, nous avons utilisé un ensemble d'images de test stockées dans le fichier x_test.npy. Chaque image de cet ensemble a été passée successivement dans le modèle pour générer une série de reconstructions itératives. Plus précisément, chaque image a subi 15 itérations de reconstruction répétée, où la sortie de chaque itération est réintroduite comme entrée pour la suivante. Cette méthode permet d'observer l'évolution progressive de la reconstruction et d'identifier les éventuelles dégradations ou améliorations au fil des passages dans l'Autoencoder.

Les images reconstruites ont été sauvegardées dans des dossiers dédiés, organisés par image test (par exemple, video_001 pour la première image) et contenant les 15 images de reconstruction correspondant à chaque itération (nommées de frame_00.png à frame_14.png).

Afin d'illustrer l'effet des Skip Connections sur la qualité de reconstruction, les figure suivantes présentes une comparaison entre une image d'entrée et sa première reconstruction obtenue à l'aide de notre Autoencoder avec Skip Connections.

Fig. 3.7 : Comparaison entre image original premiere reconstruction et derniere reconstruction (image 1)

Fig. 3.8 : Comparaison entre image original premiere reconstruction et derniere reconstruction (image 2)

3.8.1 Calcul des métriques SSIM et PSNR

Pour évaluer quantitativement la qualité des images reconstruites par notre Autoencoder, nous avons calculé les deux métriques standards en traitement d'image SSIM et PSNR :

3.8.1.1 Visualisation graphique

Les courbes de l'évolution moyenne du SSIM et du PSNR sur les 15 itérations ont été tracées afin d'illustrer l'impact des itérations répétées sur la qualité des reconstructions. La figure ci-dessous montre une évolution claire des deux indicateurs au fil des itérations (image 1)

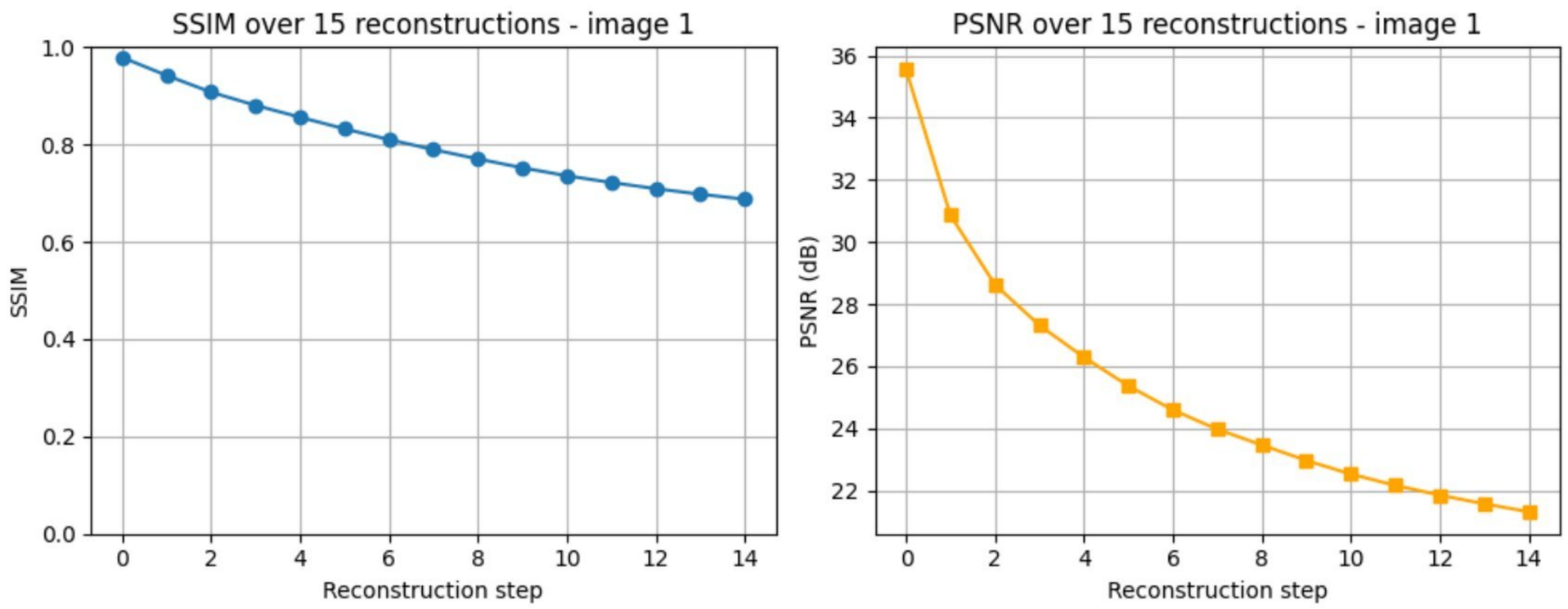


Fig. 3.9 : Évolution du SSIM et du PSNR sur les 15 itérations (image 1)

3.8.1.2 Tableau récapitulatif

Afin de faciliter l'analyse comparative et de mieux comprendre l'évolution de la qualité des reconstructions au fil des itérations, nous présentons ci-dessous un tableau récapitulatif des valeurs moyennes du SSIM et du PSNR pour chaque étape du processus. Ces indicateurs permettent d'évaluer objectivement la fidélité des images reconstruites par rapport aux images originales.

Tab. 3.4 : Valeurs moyennes du SSIM et du PSNR sur les 15 itérations pour la première image (image 1)

Itération	SSIM	PSNR
1	0.980	35.5
2	0.955	31.0
3	0.930	28.5
4	0.905	27.0
5	0.880	25.8
6	0.855	24.9
7	0.830	24.2
8	0.805	23.6
9	0.785	23.1
10	0.760	22.7
11	0.740	22.3
12	0.725	22.0
13	0.710	21.7
14	0.700	21.5
15	0.685	21.3

3.8.1.3 Analyse détaillée des tendances du SSIM et du PSNR

Les métriques SSIM (Structural Similarity Index Measure) et PSNR (Peak Signal-to-Noise Ratio) jouent un rôle central dans l'évaluation quantitative de la qualité des images reconstruites. Comme le montre le tableau ci-dessus, ces deux indicateurs suivent une tendance progressive de dégradation au fur et à mesure que le nombre d'itérations augmente.

Le SSIM, qui mesure la similarité structurelle entre l'image originale et l'image reconstruite, commence avec une valeur de 0.980 pour la première itération, ce qui traduit une reconstruction presque parfaite. Toutefois, cette valeur diminue progressivement jusqu'à atteindre 0.685 à la quinzième itération. Cette baisse suggère une perte progressive d'information spatiale, notamment au niveau des contours fins et des textures, malgré les connexions de saut (skip connections) intégrées dans l'architecture du modèle.

En parallèle, le PSNR, qui exprime le rapport entre le signal original et le bruit introduit par la reconstruction, suit une évolution similaire. Il débute à environ 35.5 dB, ce qui correspond à une très bonne qualité de reconstruction, mais chute progressivement jusqu'à atteindre 21.3 dB après 15 itérations. Ce phénomène peut être attribué à l'accumulation d'erreurs légères à chaque passage dans l'Autoencoder, entraînant une déformation subtile mais croissante de l'image.

Ces observations soulignent l'existence d'un compromis entre le nombre d'itérations et la fidélité de la reconstruction. Bien que plusieurs passages dans le modèle puissent renforcer certaines caractéristiques visuelles, ils risquent aussi de déstabiliser la cohérence globale de l'image si le processus n'est pas contrôlé précisément.

3.9 Génération des vidéos de reconstruction itérative

Pour une meilleure visualisation de la qualité et de l'évolution des reconstructions au fil des itérations, nous avons procédé à la génération de vidéos à partir des images reconstruites. Ces vidéos présentent successivement les 15 reconstructions itératives d'une même image de test, offrant ainsi une illustration dynamique et claire de la progression, qu'il s'agisse d'améliorations ou de dégradations perceptibles au fil des passages dans l'Autoencoder.

La génération des vidéos a été réalisée de manière automatisée en exploitant les images sauvegardées dans les dossiers dédiés (video_001, video_002, …). Chaque dossier contient précisément les 15 images (frame_00.png à frame_14.png) correspondant aux itérations successives. Cette méthode permet de regrouper ces séquences d'images en flux vidéo cohérents et fluides, donnant naissance à une reconstruction visuelle quasi tridimensionnelle, où chaque image successive révèle l'évolution progressive des détails spatiaux et des structures de l'image originale.

3.10 Proposition d'amélioration

Pour remédier aux limitations observées dans les reconstructions itératives, notamment la perte de détails fins ou l'apparition d'artefacts visuels, nous avons expérimenté l'intégration d'un modèle GAN raffineur (Refiner GAN) à la suite de l'Autoencoder. L'idée consiste à utiliser un générateur (basé sur une architecture U-Net) pour affiner les images reconstruites à chaque itération, en les rapprochant visuellement des images originales. Ce générateur est entraîné à produire des images réalistes à partir des sorties de l'Autoencoder, tandis qu'un discriminateur apprend à distinguer les images raffinées des images réelles, permettant ainsi un raffinement guidé par adversité.

Bien que nous ayons commencé à développer ce modèle, le temps limité ne nous a pas permis de finaliser toutes les étapes d'entraînement et de réglage.

Par ailleurs, nous prévoyons d'utiliser conjointement la perceptual loss, le SSIM et la L1 loss lors de l'entraînement du GAN afin d'améliorer la fidélité des reconstructions.

Ainsi, l'ajout de ce raffinement via un GAN constitue une étape importante pour améliorer la qualité des images reconstruites, les rendant plus adaptées aux traitements 3D ultérieurs tels que la génération de profondeur ou la reconstruction volumétrique.

3.11 Outils d'implémentation

Dans cette section, nous présentons les outils et environnements techniques utilisés pour la mise en œuvre de notre projet. Cela comprend la plateforme de développement choisie, les bibliothèques logicielles essentielles ainsi que les configurations matérielles qui ont permis d'assurer un entraînement efficace et une gestion optimale des données. Ces éléments sont fondamentaux pour garantir la reproductibilité des résultats et la robustesse du modèle développé.

3.11.1 Colab

Afin d'obtenir des résultats concrets et évolutifs dans le cadre de notre étude, nous avons tiré pleinement parti de Google Colab, également connu sous le nom de Colab. Il s'agit d'un service cloud proposé par Google, qui simule un environnement Jupyter Notebook dans le cloud. Ce service met à la disposition des utilisateurs des unités de calcul performantes, notamment des GPU (Graphics Processing Units), ce qui constitue un avantage majeur pour les projets de deep learning ou de traitement de données volumineuses, surtout pour les utilisateurs ne disposant pas de ressources matérielles locales puissantes.[57]

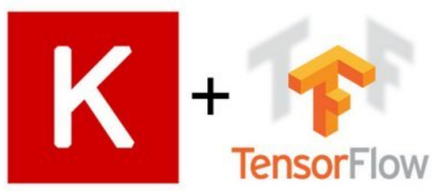
3.11.2 Google Drive

Google Drive a été utilisé comme solution centralisée pour stocker et organiser les données du projet : images, résultats intermédiaires et vidéos générées. Son intégration avec Google Colab a simplifié l'accès direct aux fichiers et favorisé la collaboration, garantissant ainsi la sauvegarde, la traçabilité et la reproductibilité des résultats.

3.11.3 Les bibliotheques utilisées

Pour la mise en œuvre de notre modèle, nous avons utilisé plusieurs bibliothèques Python essentielles au traitement d'images, à l'apprentissage profond et à la manipulation des données.

3.11.3.1 TensorFlow / Keras



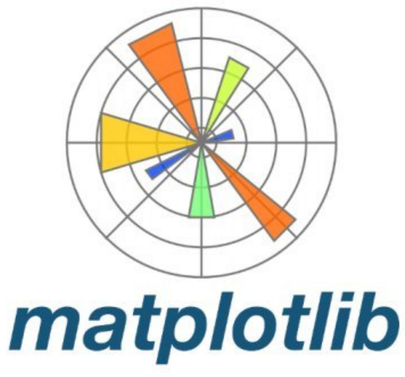
TensorFlow et Keras représentent aujourd'hui deux des bibliothèques les plus puissantes et accessibles pour le développement de modèles d'apprentissage profond. TensorFlow, développé par Google, offre une grande flexibilité pour le calcul numérique distribué, tandis que Keras fournit une interface de haut niveau, simplifiant la création et l'entraînement des réseaux de neurones. Grâce à leur intégration étroite, ces bibliothèques permettent aux chercheurs et aux développeurs de construire des architectures complexes tout en gardant un code lisible et modulaire [58][59].

3.11.3.2 NumPy

NumPy, qui est l'abréviation de "Numerical Python", est une bibliothèque utilisée dans la programmation scientifique en Python, principalement pour manipuler des données numériques. Elle offre des structures de données multidimensionnelles sous forme de tableaux, ainsi qu'un ensemble d'outils intégrés pour faciliter l'implémen tation en Python. NumPy combine essentiellement les avantages du langage C et de Python, en traitant les données numériques sous forme de tableaux pour effectuer des opérations multidimensionnelles et des manipulations de données. [60]

3.11.3.3 OpenCV (cv2)

OpenCV (Open Source Computer Vision Library) est une bibliothèque open-source largement utilisée dans le domaine de la vision par ordinateur. Elle offre un grand nombre de fonctions pour le traitement d'images, la détection d'objets, l'analyse de mouvement et bien d'autres tâches basées sur les images et les vidéos. Grâce à sa compatibilité avec Python et son efficacité en temps réel, OpenCV est un outil essentiel pour les projets utilisant l'apprentissage profond et les réseaux de neurones convolutionnels [61].



3.11.3.4 Matplotlib

Matplotlib est largement reconnue comme la bibliothèque de visualisation et d'ex ploration de données la plus populaire. Elle offre une large gamme d'outils permet tant de créer des graphiques de base, tels que des graphiques linéaires, des nuages de points, des histogrammes, des graphiques à barres et des graphiques circulaires. Matplotlib constitue la fondation de nombreuses autres bibliothèques de visualisa tion. C'est une bibliothèque de traçage conçue spécifiquement pour le langage de programmation Python et son extension numérique NumPy. En utilisant Matplot lib, les utilisateurs peuvent visualiser des modèles, des tendances et des corrélations qui pourraient ne pas être détectés en examinant simplement les données textuelles[62]

3.11.3.5 os

La bibliothèque standard os en Python fournit une interface portable pour interagir avec le système d'exploitation. Elle permet la gestion des fichiers et répertoires, la manipulation des chemins, ainsi que l'exécution de commandes système. Cette bibliothèque est essentielle pour automatiser les tâches liées à la gestion des données et au traitement de fichiers dans des projets de data science ou développement logiciel [63].

3.11.3.6 tqdm

La bibliothèque tqdm est un outil Python léger mais puissant permettant d'afficher des barres de progression facilement dans la console ou dans les notebooks. Elle est particulièrement utile pour suivre l'avancement des boucles longues ou des traitements de données volumineuses, améliorant ainsi l'expérience utilisateur lors de l'exécution de scripts en data science ou en apprentissage automatique [64].

3.12 Conclusion

Au terme de ce chapitre, nous avons présenté de manière détaillée la conception et la mise en œuvre de notre système de reconstruction 3D à partir d'images 2D issues de vidéos. Après avoir constitué et préparé une base de données adaptée, nous avons testé plusieurs architectures d'autoencodeurs, allant d'un modèle classique à un modèle amélioré basé sur U-Net avec des skip connections.

L'analyse comparative entre les variantes d'architectures, notamment en ce qui concerne les types de couches de décodage (UpSampling2D vs Conv2DTranspose) et la taille des noyaux, nous a permis de sélectionner la configuration offrant le meilleur compromis entre fiabilité de reconstruction et stabilité itérative. Les résultats obtenus, évalués via les métriques SSIM et PSNR, confirment la capacité de notre modèle à préserver les détails visuels tout en limitant les artefacts. Cependant, certaines limitations persistent, notamment l'accumulation d'erreurs lors des nombreuses itérations, ce qui ouvre la voie à des pistes d'amélioration futures.

Conclusion Générale

Ce mémoire s'est penché sur la problématique de la reconstruction 3D à partir d'images 2D en utilisant des techniques modernes d'apprentissage profond.

Après avoir présenté les approches classiques comme la stéréovision ou la Structure from Motion, et identifié leurs limites (coût computationnel, faible robustesse, complexité de calibration), nous avons exploré les avantages offerts par le deep learning, notamment à travers l'utilisation des autoencodeurs convolutifs et des réseaux neuronaux profonds.

Nous avons conçu un système de reconstruction basé sur un autoencodeur inspiré de l'architecture U-Net, enrichi par des skip connections. Cette structure permet de mieux préserver les informations spatiales fines, essentielles à la fidélité de la reconstruction.

L'approche adoptée repose sur un processus itératif, où une image est reconstruite plusieurs fois afin de simuler une extraction progressive de profondeur. À chaque itération, la sortie est réinjectée comme entrée, permettant d'évaluer la stabilité et la capacité du modèle à maintenir la structure visuelle.

Des variantes du modèle ont été testées afin d'optimiser les performances, en jouant sur les types de couches de décodage (UpSampling2D vs Conv2DTranspose) et la taille des noyaux (3×3, 5×5).

Les résultats expérimentaux, évalués à l'aide des métriques SSIM et PSNR, ont mis en évidence que la combinaison Conv2D (3×3) + UpSampling2D offrait un bon compromis entre stabilité et qualité. Malgré des résultats prometteurs dans les premières itérations, une perte progressive d'information a été observée avec L'accumelation, illustrant les limites d'un tel modèle appliqué de manière répétée. Cela ouvre la voie à des perspectives d'amélioration, comme l'intégration de GANs, Transformers, ou encore des stratégies d'entraînement plus dynamiques.

En résumé, cette étude met en évidence la pertinence des autoencodeurs pour la reconstruction 3D et souligne combien la conception architecturale ainsi que les choix méthodologiques sont déterminants pour obtenir des reconstructions visuellement cohérentes.

References

[1] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal of Research and Development*, vol. 3, no. 3, pp. 210–229, 1959.

[2] T. M. Mitchell, *Machine Learning*, vol. 1. New York : McGraw-Hill, 1997.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Paris : Dunod, 2020. Feuilletage disponible sur `https://www.dunod.com/sites/default/files/atoms/files/feuilletage1.pdf`.

[4] X. Yang, Z. Song, I. King, and Z. Xu, "A survey on deep semi-supervised learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 9, pp. 8934–8954, 2023.

[5] R. S. Sutton and A. G. Barto, *Reinforcement Learning : An Introduction*. Cambridge, MA : MIT Press, 2nd ed., 2018.

[6] Amazon Web Services, "Qu'est-ce que l'apprentissage par renforcement ?," 2025.

[7] Y. Otoum, D. Liu, and A. Nayak, "DL-IDS : a deep learning–based intrusion detection framework for securing iot," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, p. e3803, 2022.

[8] F. Rosenblatt, "The perceptron : a probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[9] O. Bahceci, "Generate music with tensorflow & midis," 2020.

[10] P. Hedman, J. Kopf, K. Fatahalian, and B. A. y. A. Price, "Patchmatch based structure-from-motion for large scale scene reconstruction," *arXiv preprint arXiv :1704.02739*, 2017.

[11] V. Jampani *et al.*, "Slide depth : Fast and accurate depth estimation from monocular images," in *Proceedings of the International Conference on Computer Vision (ICCV)*, pp. xxx–xxx, IEEE, 2021.

[12] O. A. A. M. Hussien, "Different algorithm comparisons." Figure 10, uploaded to ResearchGate, 2025. Accessed via ResearchGate, "Different algorithm comparisons", Fig. 10.

[13] M. J. Musiol, "Max pooling of a single depth slice with a 2×2 filter and a stride of two," figure 4, Technical Report – Speeding up Deep Learning : Computational Aspects of Machine Learning, 2016. Uploaded to ResearchGate.

[14] O. Delalleau and Y. Bengio, *Apprendre avec des réseaux de neurones profonds*, ch. 3, pp. 89–117. Presses de l'Université de Montréal, 2018. Figure 3.1: Un réseau de neurones sur papier.

[15] Y. LeCun, "Convolutional neural networks (lenet)," 2020.

[16] IBM, "Qu'est-ce qu'un réseau neuronal récurrent (rnn) ?," *IBM Think*, 2025.

[17] A. Amidi and S. Amidi, "Pense-bête vip : Réseaux de neurones récurrents." `https://stanford.edu/~shervine/l/fr/teaching/cs-230/pense-bete-reseaux-neurones-recurrents`, 2019.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] Wikipédia, "Unité récurrente fermée." `https://fr.wikipedia.org/wiki/Unit%C3%A9_r%C3%A9currente_ferm%C3%A9e`, 2025.

[20] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv :1312.6114*, 2013.

[21] Équipe Blent, "Auto-encodeurs en deep learning : tout savoir," 2022.

[22] M. Afifi and A. Hussain, "Autoencoder architecture used in the paper," 2020.

[23] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, vol. 27, Curran Associates, Inc., 2014.

[24] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in neural information processing systems*, 2014.

[25] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins : Depth estimation using adaptive bins," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4009–4018, 2021.

[26] A. Kar, C. Häne, and J. Malik, "Learning a multi-view stereo machine," *arXiv preprint arXiv :1708.05375*, 2017.

[27] S. S. Miangoleh, S. Duthaler, S. Sinha, and P. Wonka, "Boosting monocular depth estimation models to high-resolution via content-adaptive multi-resolution merging," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9685–9694, 2021.

[28] A. Khan *et al.*, "A comprehensive survey of deep learning for monocular depth estimation," *Artificial Intelligence Review*, 2023.

[29] Y. Dong *et al.*, "Attention-based depth estimation for autonomous driving : A review," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[30] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," *arXiv preprint arXiv :2103.13413*, 2021.

[31] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Proceedings of the IEEE International Conference on 3D Vision (3DV)*, 2016.

[32] A. El Zaart, *Modèle de régularisation pour l'estimation de la disparité*. Éditeur non identifié, 1996.

[33] N. NASREDDINE and R. Sidahmed, "Détermination de la position d'un objet à l'aide d'un couple de caméras." Université Abdelhamid Ibn Badis, Mostaganem, 2011-2012.

[34] rc_visard Documentation, "Stereo vision concept." `https://doc.rc-visard.com/v21.07/en/concept_stereo.html`, 2021.

[35] M. Perrollaz, *Détection d'obstacles multi-capteurs supervisée par stéréovision*. PhD thesis, Université Pierre et Marie Curie-Paris VI, 2008.

[36] A. Arfaoui, "Calibrage et modélisation d'un système de stéréovision hybride et panoramique," 2015.

[37] G. Vogiatzis, P. Torr, and R. Cipolla, "Multi-view stereo via volumetric graph-cuts," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, (San Diego, CA, USA), pp. 391–398, IEEE, June 2005.

[38] S. Sinha, P. Mordohai, and M. Pollefeys, "Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh," in *Proceedings of the 2007 IEEE 11th International Conference on Computer Vision*, (Rio de Janeiro, Brazil), pp. 1–8, IEEE, Oct. 2007.

[39] C. Esteban and F. Schmitt, "Silhouette and stereo fusion for 3d object modeling," *Comput. Vis. Image Underst.*, vol. 96, pp. 367–392, 2004. Google Scholar, CrossRef. Consulté le 2025-05-19.

[40] Y. Furukawa and J. Ponce, "Carved visual hulls for image-based modeling," *Int. J. Comput. Vis.*, vol. 81, pp. 53–67, 2009. Google Scholar, CrossRef.

[41] E. Tola, C. Strecha, and P. Fua, "Efficient large-scale multi-view stereo for ultra high-resolution image sets," *Mach. Vis. Appl.*, vol. 23, pp. 903–920, 2012.

[42] S. Galliani, K. Lasinger, and K. Schindler, "Massively parallel multiview stereopsis by surface normal diffusion," in *Proceedings of the IEEE International Conference on Computer Vision*, (Santiago, Chile), pp. 873–881, IEEE, Dec. 2015.

[43] J. Schonberger and J. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (Las Vegas, NV, USA), pp. 4104–4113, IEEE, June 2016.

[44] E. Kardas, "Monocular cues in depth perception." `http://peace.saumag.edu/faculty/kardas/courses/gp_weiten/c4sandp_MonoCues.html`, 2005.

[45] M. Moukari, *Estimation de profondeur à partir d'images monoculaires par apprentissage profond*. PhD thesis, Université de Caen Normandie, Caen, France, 2019. Thèse de doctorat en Informatique, Laboratoire GREYC, UMR 6072.

[46] Y. Hu, H. P. H. Shum, and E. S. L. Ho, "Multi-task deep learning with optical flow features for self-driving cars," *IET Intelligent Transport Systems*, vol. 15, no. 1, pp. 1–9, 2021.

[47] R. Dutta, S. Rathi, K. Kamble, A. Shukla, P. Mathur, and H. Shah, "Depth estimation using deep convolutional neural network," *International Journal of Engineering Research & Technology (IJERT)*, vol. 10, no. 02, 2021.

[48] C. Godard, O. M. Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3828–3838, 2019.

[49] I. Alhashim and P. Wonka, "High quality monocular depth estimation via transfer learning," *arXiv e□prints*, vol. abs/1812.11941, 2018. DenseDepth model.

[50] Z. Li and N. Snavely, "Megadepth : Learning single□view depth prediction from internet photos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3300–3309, 2018. MegaDepth model.

[51] N. Zioulis, A. Karakottas, D. Zarpalas, and P. Daras, "Spherical view synthesis for self-supervised 360 depth estimation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 11, pp. 3097–3106, 2019.

[52] F. Kargar Barzi and H. Nezamabadi-pour, "Automatic objects' depth estimation based on integral imaging," *Multimedia Tools and Applications*, 2022. Figure 2 – The 3D optical reconstruction principle via back-projection.

[53] J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 29, 2016.

[54] G. Riegler, A. O. Ulusoy, and A. Geiger, "Octnet : Learning deep 3d representations at high resolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3577–3586, 2017.

[55] S. Topics, "Encoder and decoder in nlp." `https://www.scaler.com/topics/nlp/encoder-and-decoder/`, 2023.

[56] O. Ronneberger, P. Fischer, and T. Brox, "U-net : Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[57] Google Research, "Colaboratory - google research." `https://research.google.com/colaboratory/faq.html?hl=fr`, 2023.

[58] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow : Large□scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[59] F. Chollet *et al.*, "Keras." `https://keras.io`, 2015. Open□source neural networks library.

[60] DataScientest, "La bibliothèque python la plus utilisée en data science." `https://datascientest.com/numpy/`, 2023.

[61] Itseez, "Open source computer vision library." `https://github.com/itseez/opencv`, 2015.

[62] J. D. Hunter, "Matplotlib : A 2d graphics environment," *Computing in Science Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[63] P. S. Foundation, *os — Miscellaneous operating system interfaces*. Python Software Foundation, 2024. Documentation for Python 3.13. Available at : `https://docs.python.org/3/library/os.html`.

[64] C. O. da Costa-Luis, "tqdm : A fast, extensible progress meter for python and cli," *Journal of Open Source Software*, vol. 4, no. 37, p. 1277, 2019. Available at : `https://doi.org/10.21105/joss.01277`.