الجمهورية الجزائرية الديمقراطية الشعبية
République algérienne démocratique et populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique
جامعة عين تموشنت بلحاج بوشعيب
Université –Ain Temouchent- Belhadj Bouchaib
Faculté des Sciences et de Technologie
Département Math-Informatique

Projet de Fin d'Études

Pour l'obtention du diplôme de Master en : **CYSIA**

Domaine : **Mathématiques et Informatique**

Filière : **Informatique**

Spécialité : **Cyber sécurité et intelligence artificielle**

Thème

Deep Learning et l'Aide au Diagnostic Médical (Application à la Classification d'image médicale)

Présenté par :

1) **Melle** Bensafi Basma
2) **Melle** Soussi Amina Nour El Houda

Devant le jury composé de :

Dr Bendiabdallah Med Hakim	MCA	UAT.B.B (Aïn Témouchent)	Président
Dr Bedad Fatima	MCB	UAT.B.B (Aïn Témouchent)	Examinatrice
Dr Benomar Mohamed Lamine	MCA	UAT.B.B (Aïn Témouchent)	Encadrant

Année Universitaire 2024/2025

Remerciement

Avant toute chose, nous élevons nos louanges vers Dieu Tout-Puissant, Maître des destinées, pour Ses innombrables bienfaits, Sa miséricorde infinie et la force qu'Il nous a insufflée tout au long de ce chemin. Sans Sa grâce, ce mémoire n'aurait pu aboutir.

Nous exprimons notre plus profonde gratitude à notre encadreur, Monsieur Ben Omar Mohamed Lamine, pour la richesse de ses conseils, la justesse de ses orientations, ainsi que pour sa disponibilité, sa patience et son dévouement exemplaire. Son accompagnement a constitué un véritable pilier tout au long de cette aventure académique. Qu'il trouve ici l'expression de notre estime la plus sincère.
Nos pensées les plus affectueuses et reconnaissantes vont à nos familles, véritables havres de soutien et d'amour inconditionnel. Leur présence, leurs encouragements constants, leur confiance et leurs sacrifices silencieux ont été pour nous une source inépuisable de courage, d'équilibre et de sérénité.

Nous remercions également du fond du cœur nos camarades et amis, pour leur précieuse collaboration, leur entraide spontanée, leur bienveillance et leur énergie positive, qui ont contribué à rendre ce parcours plus riche humainement et intellectuellement.

Enfin, nos remerciements s'adressent à toutes celles et ceux qui, de près ou de loin, ont marqué ce projet par un geste, un mot, un regard ou un soutien discret. À toutes ces âmes généreuses, nous adressons nos salutations respectueuses et notre reconnaissance la plus profonde.

Dédicace

À mon cher père,

Pour sa sagesse, sa patience et son soutien inébranlable, qui ont toujours été pour moi une boussole dans les moments d'incertitude.

À ma mère,

Ma force et ma vie, ton amour inconditionnel, ta patience silencieuse et tes prières constantes ont façonné chaque pas de mon parcours. Ce travail t'est dédié, car sans toi, rien n'aurait été possible.

À ma grande mère,

Ton amour profond et ta sagesse m'ont toujours guidée, tes prières et ta douceur m'ont accompagnée tout au long de ma vie. Grâce à toi, j'ai trouvé mes racines et la force d'avancer. Je dédie ce travail à toi, en hommage à ton grand cœur.

À mes sœurs,

Pour leur soutien constant, leur tendresse et leur présence chaleureuse, qui ont toujours été un appui précieux dans mes moments de doute et de joie.

À mes oncles,

Votre soutien constant, vos conseils précieux et votre présence bienveillante ont toujours été pour moi une source de force et d'inspiration. Je vous dédie ce travail avec toute ma gratitude.

À mes meilleurs amis Djihene, Imene

Vous avez toujours été là, fidèles et présentes, offrant soutien, écoute et joie à chaque étape de ce parcours. C'est grâce à vous que ce chemin a été plus léger et plus riche. Je vous dédie ce travail avec toute ma gratitude.

Et à tous les autres membres de ma famille.

Basma

Dédicace

Je dédie ce travail

À mon cher père,

qui est le pilier de ma vie, dont le soutien, le sacrifice et les encouragements constants me permettent de continuer et de croire en moi. Son amour, sa sagesse et sa confiance en mes capacités ont été une source de motivation inestimable tout au long de mon parcours.

À ma mère,

Dont l'amour inconditionnel, la patience et les prières silencieuses m'ont accompagné tout au long de ce travail. Sa force, sa gentillesse et son sacrifice quotidien ont été pour moi une source inépuisable de courage et d'inspiration.

À mes sœurs et mon frère,

Merci pour votre amour, votre soutien et vos encouragements continus. Merci d'être toujours là pour moi, dans les moments de doute comme dans les moments de réussite.

A mes chers oncles et tantes pour leur amour,

Leurs encouragements et leur précieuse présence dans ma vie. Merci pour votre réconfort, votre gentillesse et votre soutien tout au long du chemin. Votre confiance me donne la force d'avancer.

À mes amis,

Pour leur soutien indéfectible, leur encouragement et leur bonne humeur. Merci d'avoir partagé avec moi les joies et les défis de ce parcours, et d'avoir toujours cru en moi.

Amina Nour El Houda

Résumé

La classification des images médicales a reconnu une évolution notable dans les dernières années , grâce aux avancées rapides dans les techniques d'imagerie médicale et l'apparition de méthodes d'apprentissage profond (deep learning) qui se sont révélées particulièrement à haute efficacité dans le domaine de la vision par ordinateur, qui est caractérisé par la capacité d'extraire de façon automatique des caractéristiques de données sans avoir besoin d'étapes de prétraitement complexes, permettant la construction de représentations hiérarchiques riches de l'information visuelle. Ce développement a contribué à améliorer l'efficacité des systèmes de classification, et les rend plus aptes à faire face à la complexité visuelle qui caractérise de nombreuses conditions médicales, en particulier dans le domaine du diagnostic médical par imagerie, comme la classification de la rétinopathie diabétique. C'est précisément ce que nous avons examiné dans le cadre de notre mémoire. Dès lors, nous avons conçu un système automatique dans l'objectif de classifier effectivement la rétinopathie diabétique basé sur les réseaux d'apprentissage profond, notamment par l'utilisation d'apprentissage par transfert afin d'assurer une performance élevée et une durée de calcul raisonnable. Nous avons proposé trois architectures sur la base des modèles convolutives DenseNet121, ViT-B/16 et YOLOv8, reconnues pour leur efficacité sur le jeu de données ImageNet. Notre contribution consiste a réaliser la classification selon deux stratégies : la première consiste à utiliser le modèle directement sur les images, et la deuxième en cascade qui consiste à classifier les images par étapes (premièrement 2 classes (binaire), ensuite 3 classes et finalement 5 classes). Les expériences ont été conduites sur deux datasets : Le premier dataset utilisé c'est Aptos 2019 composé de 3662 images réparties en cinq (5) classes distinctes. Les résultats obtenus ont montré une nette supériorité, avec une précision maximale de 96,30%. Et le deuxième dataset utilisé c'est DDR (Dataset for Diabetic Retinopathy) composé de 12522 images divisées aussi en cinq (5) classes distinctes. Les résultats obtenus se sont avérés satisfaisants, avec une précision sur les jeux de données atteignant les 93% et 96% respectivement.

Mots-clés : apprentissage profonds, classification, apprentissage par transfert, approche en cascade, image médicale, rétinopathie diabétique.

Abstract

Medical image classification has seen significant progress in recent years, driven by rapid advancements in medical imaging techniques and the emergence of deep learning methods, which have proven highly effective in the field of computer vision. Deep learning is characterized by its ability to automatically extract features from data without requiring complex preprocessing steps, enabling the construction of rich hierarchical representations of visual information. These developments have enhanced the efficiency of classification systems, making them better suited to handle the visual complexity inherent in many medical conditions, particularly in medical imaging diagnostics, such as diabetic retinopathy classification. This is precisely the focus of our research. Accordingly, we designed an automated system to effectively classify diabetic retinopathy using deep learning networks, specifically leveraging transfer learning to ensure high performance and reasonable computational time. We proposed three architectures based on the convolutional models DenseNet121, ViT-B/16, and YOLOv8, which are renowned for their effectiveness on the ImageNet dataset. Our contribution involves classification using two distinct strategies : Direct classification, where the model is applied directly to the images. Cascade classification, where images are classified in stages—first into 2 classes (binary), then 3 classes, and finally 5 classes. The experiments were conducted on two datasets : The first dataset used

is APTOS2019, comprising 3,662 images distributed across five (5) distinct classes. The results demonstrated a clear superiority, achieving a maximum accuracy of 96.30%. The second dataset is the DDR (Dataset for Diabetic Retinopathy), consisting of 12,522 images also divided into five (5) distinct classes. The results were satisfactory, with dataset accuracy reaching 93% and 96% respectively.

Keywords : Deep learning, classification, transfer learning, cascade approach, medical imaging, diabetic retinopathy.

الملخص

شهد تصنيف الصور الطبية تطوراً كبيراً في السنوات الأخيرة، وذلك بفضل التقدم السريع في تقنيات التصوير الطبي وظهور أساليب التعلم العميق التي أثبتت فعاليتها بشكل خاص في مجال الرؤية الحاسوبية، والتي تتميز بالقدرة على استخراج الخصائص تلقائياً من البيانات دون الحاجة إلى خطوات معالجة مسبقة معقدة، مما يتيح بناء تمثيلات هرمية غنية للمعلومات المرئية. وقد ساعد هذا التطور في تحسين كفاءة أنظمة التصنيف، مما يجعلها أكثر قدرة على التعامل مع التعقيد البصري الذي يميز العديد من الحالات الطبية، خاصة في مجال تشخيص التصوير الطبي، مثل تصنيف اعتلال الشبكية السكري. وهذا بالضبط ما درسناه في أطروحتنا. وبالتالي، قمنا بتصميم نظام تلقائي بهدف تصنيف اعتلال الشبكية السكري بشكل فعال بناءً على شبكات التعلم العميق، وذلك باستخدام التعلم النقلي لضمان الأداء العالي ووقت الحساب المعقول. لقد اقترحنا ثلاث بنيات بالإستناد إلى النماذج التلافيفية DenseNet121 و ViT-B/16 و YOLOv8، والتي تشتهر بكفاءتها على مجموعة بيانات .ImageNet تتمثل مساهمتنا في إجراء التصنيف وفقًا لاستراتيجيتين: الأولى تتمثل في استخدام النموذج مباشرةً على الصور، والثانية تتمثل في تصنيف الصور على مراحل (أولًا فئتين (ثنائية)، ثم 3 فئات وأخيرًا 5 فئات). أجريت التجارب على مجموعتي بيانات: مجموعة البيانات الأولى المستخدمة هي Aptos 2019 وتتكون من 3662 صورة مقسمة إلى خمس (5) فئات متميزة. وأظهرت النتائج التي تم الحصول عليها تفوقاً واضحاً، حيث بلغت الدقة القصوى 96.30% والمجموعة الثانية من البيانات المستخدمة هي DDR والتي تتكون من 12522 صورة مقسمة أيضًا إلى خمس (5) فئات مميزة. وكانت النتائج التي تم الحصول عليها مرضية، حيث بلغت دقة مجموعة البيانات 93% و96% على التوالي.

الكلمات المفتاحية : التعلم العميق، التصنيف، التعلم الانتقالي، النهج المتتالي، التصوير الطبي، اعتلال الشبكية السكري.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

— **AI** : Artificial Intelligence (Intelligence Artificielle)

— **ML** : Machine Learning (Apprentissage Automatique)

— **DL** : Deep Learning (Apprentissage Profond)

— **CNN** : Convolutional Neural Network (Réseau de Neurones Convolutifs)

— **DiaCNN** : Diabetic Convolutional Neural Network

— **BiCNN** : Bichannel Convolutional Neural Network (CNN bicanaux)

— **Hybrid-CNN** : Hybrid Convolutional Neural Network

— **ViT** : Vision Transformer

— **YOLO** : You Only Look Once (Modèle de détection d'objets en temps réel)

— **MRI** : Magnetic Resonance Imaging (Imagerie par Résonance Magnétique)

— **OCT** : Optical Coherence Tomography (Tomographie par Cohérence Optique)

— **RGB** : Red Green Blue (Rouge Vert Bleu)

— **DLR** : Deep Learning-based Retinopathy

— **DR** : Diabetic Retinopathy (Rétinopathie Diabétique)

— **NODR** : No Diabetic Retinopathy (Pas de Rétinopathie Diabétique)

— **TP** : Travail Pratique

— **API** : Application Programming Interface (Interface de Programmation d'Applications)

— **AMP** : Automatic Mixed Precision (Précision Mixte Automatique)

— **GPU** : Graphics Processing Unit (Processeur Graphique)

— **ROC** : Receiver Operating Characteristic (Courbe ROC)

— **AUC** : Area Under the Curve (Aire Sous la Courbe)

— **TPR** : True Positive Rate (Taux de Vrais Positifs)

— **FPR** : False Positive Rate (Taux de Faux Positifs)

— **ReLU** : Rectified Linear Unit (Unité Linéaire Rectifiée)

— **LR** : Learning Rate (Taux d'Apprentissage)

— **KNN** : K-Nearest Neighbors (K plus proches voisins)

— **SVM** : Support Vector Machine (Machine à Vecteurs de Support)

— **TP** : True Positive (Vrai Positif)

— **FP** : False Positive (Faux Positif)

— **TN** : True Negative (Vrai Négatif)

— **FN** : False Negative (Faux Négatif)

INTRODUCTION GÉNÉRALE

La rétinopathie diabétique est l'une des complications les plus courantes du diabète et constitue une cause majeure de cécité, touchant environ 30 % des diabétiques si elle n'est pas détectée et traitée précocement. Cette pathologie est causée par une hyperglycémie chronique qui endommage progressivement les microvaisseaux et neurones de la rétine. Les yeux sont particulièrement sensibles à l'atteinte des petits vaisseaux. La détection précoce et la classification précise de la rétinopathie diabétique sont donc essentielles pour une prise en charge efficace de la maladie et pour prévenir les complications visuelles graves.

Traditionnellement, le diagnostic et la classification de la rétinopathie diabétique reposaient sur des évaluations effectuées par des ophtalmologistes, un processus long et coûteux. Ces dernières années, les techniques d'imagerie médicale et les algorithmes d'apprentissage automatique ont considérablement amélioré la classification de la rétinopathie diabétique, fournissant des outils plus efficaces pour la détection précoce et la surveillance de la maladie à partir d'images rétiniennes.

Malgré ces progrès, il existe encore des limites et des défis dans les méthodes de classification, tels que le processus manuel des lésions de la rétinopathie diabétique est lent et chronophage, ce qui ralentit le dépistage et la prise en charge des patients, et le manque d'ophtalmologistes expérimentés. Des fois, le risque de divergences d'interprétation et la difficulté à identifier les anomalies rares des structures normales.
Pour pallier ces défis, les systèmes de diagnostic assisté par ordinateur (CAD) se sont développés dans l'objectif principal d'aider le praticien dans son travail. Avec l'arrivé de l'intelligence artificielle (IA) et le deep learning (DL), ils offrent des solutions prometteuses pour améliorer les performances des systèmes CAD. Le deep learning a démontré un fort potentiel en imagerie médicale, atteignant des performances remarquables en termes de sensibilité et de spécificité. Les techniques de classification d'images basées sur les réseaux de neurones convolutifs (CNN) sont de plus en plus utilisées pour la classification automatique de la rétinopathie diabétique. Ces méthodes permettent d'améliorer la classification précoce de la maladie avec une précision équivalente, voire supérieure, à celle des spécialistes humains, tout en facilitant l'accès au dépistage. Plus dernièrement, des architectures basées sur le mécanisme d'attention, tel que les Transformers, ont également prouvé leur efficacité dans la classification d'images médicales, grâce à leur capacité à capturer des relations globales dans l'image. En parallèle, des modèles conçus pour la détection d'objets, permettent de localiser automatiquement les lésions caractéristiques (microanévrismes, hémorragies, exsudats) directement dans les images du fond d'œil. Cela représente un a tout majeur pour le diagnostic visuel, en complément des approches purement classificatrices.

Dans notre projet de mémoire de fin d'études, l'objectif principal est d'élaborer un système automatique efficace de la rétinopathie diabétique à partir d'images du fond d'œil selon ses différents stades (saine, légère, modérée, sévère, prolifération) en utilisant des architectures convolutionnelles de deep learning, notamment l'apprentissage par transfert (TL), afin de résoudre les problèmes de la classification, aidant au diagnostic médical.
Dans ce but, nous avons utilisé les deux plateformes Google Colab et Kaggel en exploitant le langage Python et ses bibliothèques afin d'améliorer l'efficacité de l'entraînement grâce à leurs GPU gratuits, à l'accès facile aux datasets.
L'approche repose sur :

- Utilisation de trois modèles pré-entrainé (DenseNet121, ViT-B/16 et YOLOv8).

- Utilisation de stratégie en cascade et directe, dans l'objectif d'affiner et simplifier la tâche de classification progressivement via la division de problème en nombreux étapes hiérarchiques, premièrement une classification binaire, ensuite une classification à 3 classes, et enfin une classification finale à 5.
- Améliorer la généralisation des modèles.
- Évaluer la performance des modèles.

Des expérimentations sont menées sur deux jeux de données (dataset), disponibles en ligne, sur Kaggle. Le premier dataset utilisé c'est Diabetic Retinopathy 224x224 Gaussian Filtered contenant 3662 images du fond d'œil. Le deuxième dataset connu sous le nom de DDR (Dataset for Diabetic Retinopathy), elle est composée de 12522 images du fond d'œil. Chaque dataset étant associé à cinq (5) stades selon la gravité de la rétinopathie diabétique (saine, légère, modérée, sévère, prolifférative).

Ce mémoire est organisé en quatre (4) chapitres. Le premier chapitre présente des notions de base sur l'imagerie médicale en général et plus particulièrement sur les images rétiniennes. Le deuxième chapitre est consacré aux maladies oculaires, nous exposons ces notions fondamentales ainsi que les caractéristiques de ces maladies. Dans le troisième chapitre, nous abordons la classification d'images de façon générale, en présentant quelques approches de classification et quelques travaux sur la classification d'images rétiniennes. Finalement, le quatrième chapitre est consacré à la partie résultats et expérimentation, où nous détaillons les méthodes proposées, les bases de données utilisées et nous terminons avec une discussion et une analyse sur les modèles réalisés. Une conclusion générale est déterminée à la fin du mémoire, rappelant ainsi le contexte du travail avec des perspectives de recherches futures.

Chapitre 1 : Imagerie Médicale

Chapitre 1

Imagerie Médicale

1.1 Introduction

L'imagerie médicale est l'une des branches essentielles de la médecine moderne. Elle est utilisée en complément d'un examen clinique et d'autres bilans biologiques pour poser des diagnostics précis et suivre l'évolution des maladies [1]. Elle joue un rôle crucial dans l'évolution des soins et la recherche médicale, en utilisant différentes techniques permettant d'examiner et de visualiser l'intérieur du corps d'un organisme vivant sans nécessiter de chirurgie. Grâce à ces technologies, les experts de la santé peuvent obtenir des images précises des organes, des os et des tissus afin de diagnostiquer, surveiller et traiter un large éventail de maladies.

1.2 Image Numérique

1.2.1 Définition

Une image numérique est la représentation d'une image bidimensionnelle (dessin, icône, photographie, film, etc.) sous la forme d'un ensemble fini de valeurs numériques appelées d'images ou pixels. Chaque pixel a une valeur unique qui définit sa couleur et sa luminosité et tous forment l'image complète. Il existe trois types de couleur dans d'images numériques : image noir et blac , image niveau de gris , image en couleur [2].

1.2.2 Image noire et blanc

Appelé une image binaire aussi, est un type d'image qui ne prend que des pixels avec uniquement deux valeurs 0(noir) et 1(blanc) (voir Figure 1.1), il suffit 1 bit par pixel pour coder l'image.

FIGURE 1.1 – Image binaire un seuil de valeur 100 [3]

1.2.3 Image niveau de gris

Dans une image en niveaux de gris, la couleur d'un pixel choix entre le noir et le blanc peut varier à différentes échelles. Ils sont des images de profondeur de 8 bits, c'est à dire la valeur de chaque pixel est comprise ente 0 (pour le noir absolu) a 255 (pour le blanc pur) et les valeurs intermédiaires sont déterminées l'aspect de l'image [4] (voir Figure 1.2).

FIGURE 1.2 – Image niveau de gris [3]

1.2.4 Image en couleur

Est obtenue en mélangeant trois couleurs primaires : rouge, vert, bleu (RVB ou RGB en anglais) (voir Figure 1.3), chaque couleur utilise 8 bits, donc (256*256*256 couleurs).
Les types des images présentent quelques inconvénients comme :

- L'existence d'un grand nombre de systèmes RVB [Vandenbroucke, 2000 [5]]
- Les coordonnées et les composantes trichromatiques peuvent prendre des valeurs négatives [Benomar, 2018 [6]].
- L'existence de couleurs non reproductible par addition des trois spectres [Benomar, 2018 [6]].
- Deux stimuli de couleur peuvent avoir la même chrominance avec des composantes trichromatiques (R, G, B) différentes, à cause des composantes trichromatiques qui sont liées à la luminance [Benomar, 2018 [6]].

FIGURE 1.3 – Image en couleur [3]

IL existe d'autres espaces couleurs tel que :

1. L'espace colorimétrique CIE 1976 Lab
   Communément appelé CIELAB, est un espace colorimétrique spécifiquement conçu pour calculer la distance euclidienne entre deux points, ce qui permet d'évaluer la différence calorimétrique entre deux stimuli de couleur [7]. Ces paramètres sont :
   - **L*** représente la luminance, varie de 0 (noir) à 100 (blanc) ;
   - **a*** représente la valeur sur l'axe vert → rouge ;

- **b*** représente la valeur sur l'axe bleu → jaune.

Il est basé sur un motif de couleurs opposées. Dans l'espace, il est représenté par une sphère légèrement aplatie (voir figure 1.4)

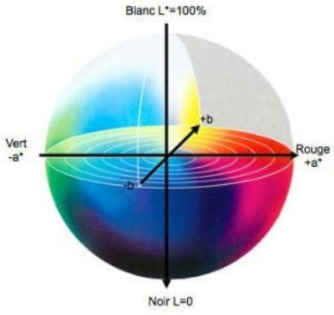


FIGURE 1.4 – Représentation spatiale du modèle L*a*b [8]

2. Espace couleur HSL
   Le modèle HSL (Hue, Saturation, Luminance) ou en français TSL (Teinte, Saturation, Luminosité), basé sur l'oeuvre du peintre Albert H. Munsell (créateur de l'Atlas Munsell), est un modèle de représentation dit "naturel", c'est-à-dire proche de la perception physiologique de la couleur par l'oeil humain [9]. L'un des principaux avantages des couleurs HSL par rapport aux couleurs RGB est que les couleurs complémentaires sont opposées les unes aux autres, ce qui rend l'ensemble Système très intuitif. Le modèle HSL décompose la couleur selon des critères physiologiques :

   - **La teinte (H)** correspond à la perception des couleurs.
   - **La saturation(S)** décrit la pureté, la vivacité ou la matité d'une couleur.
   - **La luminance (L)** fait référence à la quantité de lumière qu'une couleur possède, elle montre si la couleur apparaît claire ou foncée.

   Voici une représentation graphique du modèle HSL. La teinte est représentée par un cercle chromatique, la luminosité et la saturation sont représentées par deux axes (voir Figure 1.5).

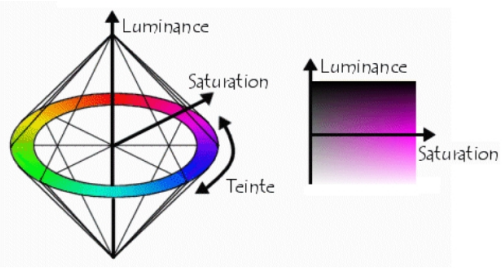


FIGURE 1.5 – Représentation spatiale du modèle HSL [9]

1.3 Les technologies d'imagerie médicale

1.3.1 Définition

L'imagerie médicale est définie comme la matérialisation, sous forme d'images, d'informations anatomiques ou fonctionnelles in vivo de parties du corps humain, obtenues pour répondre à un besoin médical à l'aide de techniques appropriées, tout en minimisant les risques et coûts pour le patient. La

notion de technologie d'imagerie médicale englobe l'utilisation de diverses techniques et des nouvelles méthodes afin d'aider les experts de la santé à repérer des zones qui ne sont pas perceptibles à l'œil nu, parmi les types les plus utilisées en médecine on note les méthodes tomographiques (TDM) basées sur des rayons X, les méthodes échographiques, les méthodes de résonance magnétique (IRM) et a la fin les méthodes optiques [Bihan, 2018 [10]].

1.3.2 Radiographie

La radiographie médicale est un terme complet qui englobe plusieurs types de tests repose sur l'utilisation d'un type de rayonnement à une énergie plus élevée nommé des rayons X, il vise à produire des images statiques après la fin de l'exposition au rayonnement [11]. L'opération est effectuée en faisant passer les rayons X à travers le corps, dont une partie est absorbée ou dispersée par des structures internes, tandis que le rayonnement restant se déplace vers un détecteur qui enregistre l'image pour une évaluation ultérieure [12] (voir figure 1.6).

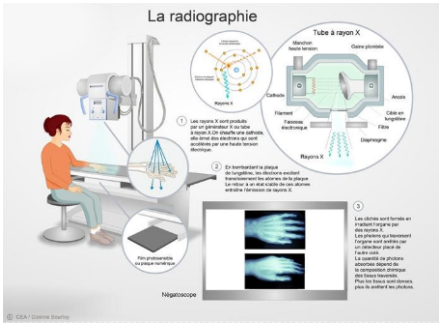


FIGURE 1.6 – Les étapes de photographie utilisant la radiographie [13]

1.3.3 Scanner

Le scanner médical, pareillement appelé tomodensitométrie (TDM), est une modalité d'imagerie incontournable en médecine moderne qui utilise des rayonnements ionisants pour produire des images détaillées en coupe internes du corps humain telle que des vues précises des os, des organes, des vaisseaux sanguins et des tissus mous (voir figure 1.7) grâce à des traitements informatiques avancées et qui offrant ainsi une meilleure résolution que les radiographies classiques qui permet aux médecins d'établir des diagnostics et d'orienter les traitements de nombreuses maladies [14].

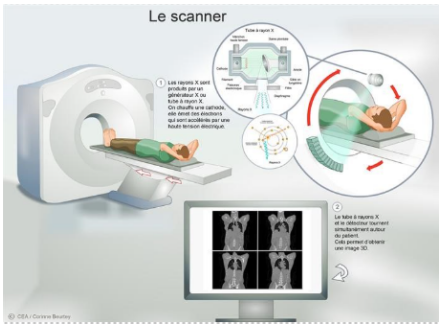


FIGURE 1.7 – Les étapes d'utilisation le scanner [13]

1.3.4 Imagerie par Résonance Magnétique (IRM)

L'IRM est considéré comme une méthode cruciale et efficace qui repose sur un puissant champ magnétique et des ondes radio pour générer des images des coupes anatomiques du corps humain sous différents angles en haute résolution et sans recourir aux rayonnements ionisants à l'inverse d'autres

méthodes d'imagerie comme la radiographie ou le scanner (voir Figure 1.8). Son fonctionnement est basé sur l'interaction entre le champ magnétique et les protons présents dans les molécules d'eau du corps, lorsqu'une onde radio est émise, elle perturbe l'alignement des protons, et lors de leur retour à l'état initial, un signal est capté, traitées par des logiciels et transformé en images en 2D ou 3D de l'intérieur du corps avec une grande précision par un ordinateur. Grâce à son développement, l'IRM a amplement ouvert de diverses perspectives diagnostiques et de recherche notamment pour l'étude d'organes complexes comme le cerveau, la moelle épinière et les muscles, améliorant ainsi la compréhension de nombreuses pathologies [15].

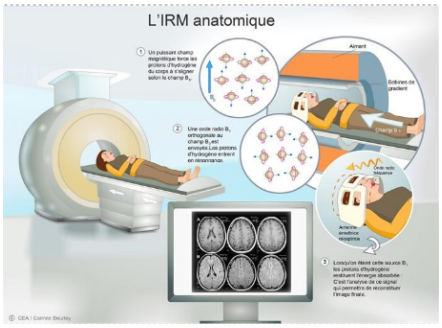


FIGURE 1.8 – Les étapes de l'imagerie par l'IRM [13]

1.3.5 Échographie

L'échographe est un appareil qui se compose d'une sonde émettant des ondes vers les tissus et réceptionnant celles qu'ils renvoient. Selon leur densité, les tissus traversés font écho différemment, plus le tissu est dense, plus l'écho est important [Mersaoui et Fouiri, 2018 [16]]. Son principe repose sur l'émission d'ondes sonores à haute fréquence également nommé des ultrasons à l'aide d'un transducteur inséré dans le corps ou placé sur la peau, ces ondes se propagent à travers les tissus et sont partiellement réfléchies lorsqu'elles rencontrent des structures de densités différentes, les échos ainsi générés sont captés , puis analysés par un ordinateur afin d'obtenir des images dynamiques (voir Figure 1.9) au contraire aux rayons X et à l'IRM, elle n'émet aucun rayonnement ionisant, ce qui la rend particulièrement sûre et adaptée à un large éventail de patients, y compris les femmes enceintes et les nourrissons [17].

FIGURE 1.9 – Exemple d'une image échographie [Mersaoui et Fouiri, 2018 [16]].

1.3.6 Microscopie

La microscopie est une discipline fondamentale permettant d'explorer le monde microscopique en s'appuyant sur divers principes physiques et un ensemble de techniques afin d'observer les divers objets de très petites dimensions qui ont invisibles à l'œil nu, en les élargissant grâce à des instruments appelés microscopes pour obtenir des images détaillées d'échantillons de petite taille. Son principe

repose sur l'interaction entre une onde et l'échantillon observé. Une onde, qu'elle soit lumineuse, électronique ou autre, est envoyée sur la préparation ou émise par celle-ci, également est captée par un objectif qui la concentre et passe par un oculaire qui crée une image observable. Cette image est observée soit à l'œil nu, soit photographiée, soit enregistrée par caméra et stockée sur ordinateur pour retraitement [Semmame, 2020 [18]] (voir Figure 1.10).

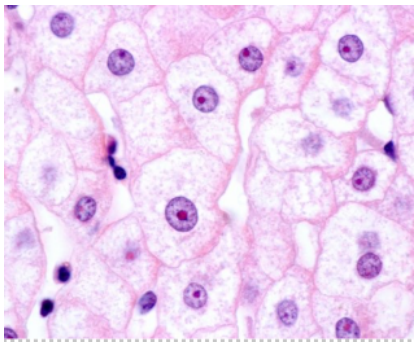


FIGURE 1.10 – Exemple d'une micrographie de cellules hépatiques (hépatocytes) réalisée avec un microscope optique [19]

1.3.7 Tomographie par Cohérence Optique (OCT)

On utilise couramment le terme "OCT" pour désigner la Tomographie par Cohérence Optique (voir Figure 1.11), une avancée majeure en imagerie non invasive qui utilise des ondes lumineuses infrarouges pour observer avec une grande précision les tissus biologiques en coupe transversale. L'OCT a été commercialisée à partir de 1996 pour l'imagerie de la rétine, quelques années seulement après la première démonstration en laboratoire. Elle constitue désormais une technique standard et incontournable en ophtalmologie [Dubois, 2023 [20]]. Le principe de cette méthode repose sur l'interférométrie optique qui utilise des ondes lumineuses pour créer des images avec une résolution élevée en capturant la lumière réfléchie par différentes couches d'un tissu et analyse les franges d'interférence crées. Après cela, ces données sont traitées pour générer des images en 2D ou 3D, qui permettent d'examiner des structures intérieures avec une précision micrométrique. En ophtalmologie, l'OCT est couramment employée pour l'examen de la rétine, ce qui permet de rendre également de diagnostiquer et de suivre des maladies comme la rétinopathie diabétique ou l'œdème maculaire diabétique (voir Figure 1.12), en détectant des anomalies sans nécessiter de procédure invasive [21].

FIGURE 1.11 – Appareil de Tomographie par Cohérence Optique (OCT) utilisé pour l'examen rétinien [22]

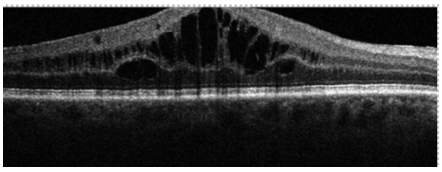


FIGURE 1.12 – Image OCT d'une rétine avec œdème maculaire diabétique (kystes noirs dans les couches rétiniennes) [23]

1.4 Conclusion

Le diagnostic correct des maladies est la nécessité la plus fondamentale avant le traitement, grâce aux progrès technologiques rapides dans les sciences médicales, qui ont à leur tour offert divers moyens d'imagerie médicale, allant de l'imagerie par tomodensitométrie à des techniques plus avancées telles que la tomographie par ordinateur, les ultrasons ou l'imagerie par résonance magnétique, etc., fournissant ainsi un outil précieux pour l'examen et le traitement précis des patients. Le chapitre suivant traite de l'anatomie de l'œil et des différentes maladies oculaires, en mettant particulièrement l'accent sur la rétinopathie diabétique, ses causes, ses symptômes et ses diagnostiques.

Chapitre 2 : Maladies ophtalmologiques

Chapitre 2

Maladies ophtalmologiques

2.1 Introduction

Les maladies oculaires englobent un ensemble d'affections qui affectent les structures de l'œil et peuvent altérer temporairement ou définitivement la vision. Le manque de diagnostic et de traitement à temps entraîne la perte de vision pour des millions de personnes à travers le monde. Parmi les pathologies les plus courantes, on retrouve la cataracte, le glaucome, la dégénérescence maculaire liée à l'âge (DMLA), ainsi que la rétinopathie diabétique, qui est une complication grave du diabète. Ces maladies peuvent être d'origine génétique, dégénérative, infectieuse.

2.2 L'œil

2.2.1 Définition

L'œil est l'organe principal du système visuel, qui capte les images et les transforme en signal électrique vers le nerf optique. Ce signal est ensuite « traduit » par le cerveau, au niveau du cortex visuel (voir Figure 2.1), qui nous renvoie l'image traitée et permet ainsi l'interprétation de notre environnement [24].

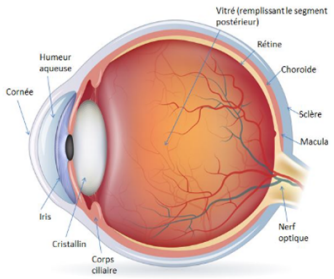


FIGURE 2.1 – Montrant l'anatomie du globe oculaire [25]

2.3 Fond d'œil

2.3.1 Définition

Le fond d'œil est une évaluation ophtalmologique visant à analyser les structures de l'œil à l'arrière du cristallin. Effectué par un médecin ophtalmologiste à l'aide d'instruments d'optique tels que des lunettes et un microscope, le fond d'œil est essentiel pour vérifier la santé de la rétine, en particulier dans certaines maladies telles que le diabète ou l'hypertension [26].

2.3.2 Déroulement d'un examen de fond d'œil

Chaque visite de contrôle réalise par un examen simple du fond d'œil est rapide, peut être réalise selon les techniques suivantes [26] :

— Ophtalmoscope : sans contact avec l'œil, En utilisant une petite lumière qui traverse vers la pupille (voir Figure 2.2).

FIGURE 2.2 – Image montrant utilisation d'ophtalmoscope [27]

— Biomicroscope ou lamp a fente : lorsque le spécialiste a besoin d'une vision complète de fond de l'œil. Pour examiner la rétine, il utilisera un verre à trois miroirs posés directement sur votre œil (voir Figure 2.3).

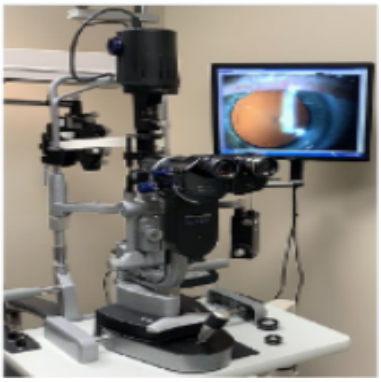


FIGURE 2.3 – Image montrant la machine de Biomicroscope [28]

2.4 Maladies oculaires

2.4.1 La cataracte

Est une maladie qui affecte l'œil et provoque une chute progressive de la vision pouvant aller jusqu'à la cécité (voir Figure 2.4). C'est un phénomène normal après 55 ans [29].

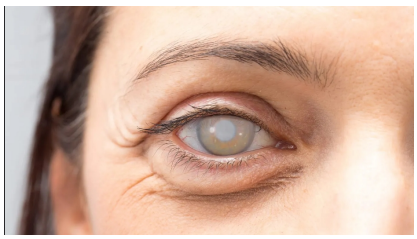


FIGURE 2.4 – Image montrant la maladie de la cataracte [30]

2.4.2 Le glaucome

Est une maladie oculaire sûrement grave caractérisée par des lésions du nerf optique et des cellules nerveuses qui le composent (voir Figure 2.5). Des modifications du champ visuel, d'abord périphérique puis vers le centre de la vision, sont associées à ces lésions du nerf optique [31].

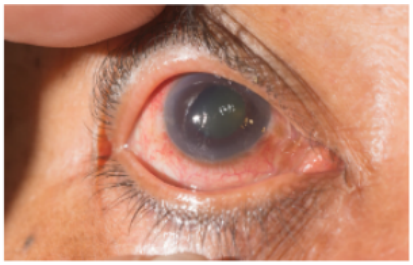


FIGURE 2.5 – Image montrant la maladie du glaucome [32]

2.4.3 Les maladies infectieuses

Regroupent un certain nombre de pathologies dont la gravité varie en fonction de la structure de l'œil (voir Figure 2.6). Des bactéries, des virus, des champignons ou des parasites peuvent infecter l'œil et les tissus qui l'entourent [33].

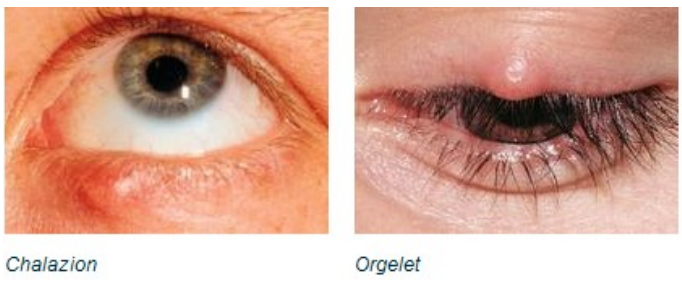


FIGURE 2.6 – Image montrant un type du maladie infectieuse [33]

2.4.4 La rétinopathie diabétique (RD)

Est une complication du diabète sucré. Elle est causée par une glycémie élevée à long terme, qui entraîne des modifications des vaisseaux sanguins de la rétine (voir Figure 2.7). Pouvant conduire à une perte de vision progressive, voir à la cécité [34].

FIGURE 2.7 – Image montrant la maladie de la rétinopathie diabétique [35]

Après avoir défini quatre types de maladies oculaires et il y a d'autres types, nous avons choisi de travailler sur la détection de la rétinopathie diabétique dans notre projet pour plusieurs raisons. Tout d'abord de nombreux patients diabétiques pertent la vision si elle n'est pas diagnostiquée et traitée à temps. De plus, les avancées technologiques dans l'imagerie médicale, comme la tomographie en cohérence optique (OCT) et la rétinographie, offrent des moyens plus précis et accessibles pour diagnostic cette maladie. Enfin, améliorer les méthodes de dépistage de cette maladie contribue à développer la prise en charge des patients diabétiques et à diminuer la charge sur les systèmes de santé.

2.5 La maladie de diabète

2.5.1 Définition

Le diabète est une maladie dans laquelle le corps ne produit pas d'insuline ou ne l'utilise pas correctement. L'insuline est une hormone qui transforme le sucre, les glucides en énergie nécessaire. Le diabète se caractérise par une hyperglycémie chronique, un excès de sucre dans le sang et donc un taux de glucose trop élevé, la maladie survient lorsque le pancréas ne secrète plus d'insuline. Il y a deux types principaux de diabète : type 1 et type 2, ils nécessitent une prise en charge sérieuse. Il n'y a pas de "petits diabètes" [36]

2.5.2 Les Complications

Une glycémie mal contrôlée peut entraîner des complications graves :

- Cécité,
- Atteintes des nerfs,
- Les yeux,
- Les reins,
- Risques d'amputations.
- D'accident vasculaire cérébral.

Ces complications aggravent le diabète et tendent à faire baisser l'espérance de vie des personnes atteintes. La majorité des complications liées au diabète peuvent être évitées, réduites si le diabète est dépisté et traite précocement [36].

2.6 La rétinopathie diabétique

2.6.1 Définition

La rétinopathie diabétique est une complication fréquente du diabète sucré qui compromet le fonctionnement de la rétine. C'est une pathologie qui apparaît lorsque les vaisseaux sanguins de la rétine se détériorent [37] qui touche 50 des patients diabétiques de type 2. Cette pathologie peut accélérer la survenue d'autres maladies des yeux comme la cataracte, et même conduire à la cécité en l'absence d'un traitement adapte [38].

2.6.2 Les classifications de la rétinopathie diabétique

2.6.2.1 La rétinopathie non proliférante

La rétinopathie non proliférante ou bien appelé la rétinopathie diabétique simple (voir Figure 2.8) débute par un stade de rétinopathie diabétique non proliférante qui provoqué une augmentation de la perméabilité capillaire, des microanévrismes, des hémorragies, des exsudats, une ischémie maculaire et un œdème maculaire [39]. L'examen ophtalmologique présente des signes précoces de troubles rétiniens, y compris des microanévrismes et des hémorragies rétiniennes. Les microanévrismes sont des lésions arrondies qui se distinguent par leur taille et leur couleur. Les microanévrismes sont souvent localisés dans le pôle postérieur de la rétine et peuvent être associés à des hémorragies intrarétiniennes. De plus, la création de taches cotonneuses et un aspect diffus de la rétine de couleur grise indiquent des lésions microvasculaires plus graves, qui, selon la quantité d'anomalies observées, peuvent indiquer un risque de progression [34].

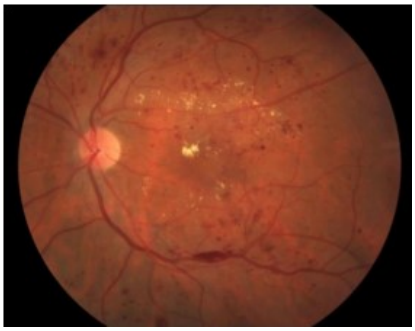


FIGURE 2.8 – Exemple d'une image de fond d'œil dans la rétinopathie diabétique non proliférante [39]

2.6.2.2 rétinopathie proliférante

La rétinopathie diabétique augmenté en plusieurs étapes, dont la plus grave est la rétinopathie diabétique proliférative (RDP) (voir Figure 2.9). Elle touche 15 à 20 pourcent des personnes diabétiques. Sur le diabétique rétinopathie proliférante se développe avec une ischémie rétinienne sévère, formant une néovascularisation : des vaisseaux sanguins anormaux fragiles pénètrent dans le vitré et provoquent une hémorragie dans la cavité, ce qui augmente considérablement le risque de perte de vision. D'autres complications comprennent la prolifération fibreuse. Des parties séparées de la rétine peuvent se décomposer, une augmentation de la pression intraoculaire, souvent constatée dans le cas d'un glaucome, menace le nerf optique [34].

FIGURE 2.9 – : Exemple d'une image de fond d'œil dans la rétinopathie diabétique proliférante [39]

2.6.3 Les symptômes de la RD

Au début la rétinopathie ne s'accompagne d'aucun symptôme visuel. La chute de la vision est souvent progressive et lent. Mais elle peut être brutale lors de l'hémorragie de la vitre dans le cas d'une rétinopathie proliférative [40].
Les symptômes qui apparaissent généralement sont :

- Vision trouble.,
- Des microanévrismes.,
- Perte de vision lente au fil du temps.,
- Perte soudaine de la vision.

2.6.4 Diagnostic de la RD

Le diagnostic de rétinopathie diabétique repose essentiellement sur un examen ophtalmologique approfondis. L'examen de routine devrait comprendre l'acuité visuelle, il faudra éliminer la possibilité d'œil cette à ophtalmoscopie indirecte associé à biomicroscopie utilisant lampe à fente et la photographie du fond d'œil (voir Figure 2.10). Il devrait révéler des lésions susceptibles d'être des signes majeurs de maladie (néovaisculatisation, œdème, hémorragie, décollement de la rétine). L'angiographie à la fluorescéine peut être indiqué pour la détection des fuites vasculaires et des anomalies de la circulapart, la tomographie en cohérence optique OCT pourrait être réalisée pour évaluer l'épaisseur rétinienne ou la présence de traction vitréo-maculaire. Ces tests peuvent aider à confirmer le diagnostic et à clarifier la stratégie de réponse [34].

FIGURE 2.10 – Diagnostic de la rétinopathie diabétique utilisant biomicroscopie (lampe à fente)[41]

2.6.5 Le traitement de la RD

Le traitement de la rétinopathie diabétique demandé une collaboration entre l'ophtalmologue et le diabétologue. Diagnostiquée à temps, la maladie peut être traitée efficacement, lorsque des damages diabétiques sont découvertes, elles doivent être stabilisées. Le traitement utilisé est la photocoagulation au laser (voir Figure 2.11). Un contrôle lourd de la glycémie ralentit la progression

de la rétinopathie, pour la rétinopathie proliférante compliquée ou à haut risque de complication, une photocoagulation panrétinienne au laser, des médicaments anti-VEGF (anti-vascular endothelial growth factor) et parfois une vitrectomie. Dans certains cas de rétinopathie non proliférative sérieux, la photocoagulation panrétinienne au laser peut être utilisée. Rétinopathie diabétique proliférative avec risque élevé d'hémorragie intravitréelle, néovascularisation pré-rétinienne étendue, doit être traité par photocoagulation panrétinienne au laser [42].

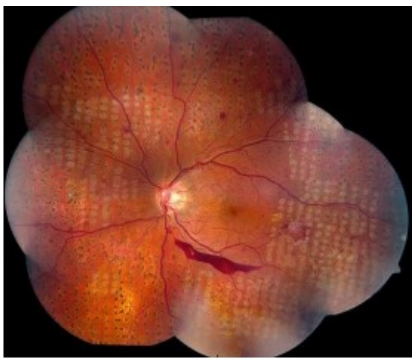


FIGURE 2.11 – Exemple d'une de rétinophotolographie d'impacts de photo coagulation laser [43]

2.7 Conclusion

Les maladies oculaires constituent un enjeu majeur de santé publique, pouvant entraîner une perte de vision en a cause de l'absence d'un diagnostic précoce. La rétinopathie diabétique occupe une place préoccupante, chez les patients diabétiques et représente une cause majeure de cécité. Grâce aux avancées en imagerie médicale et en intelligence artificielle, le dépistage et le diagnostic de cette pathologie peuvent être améliorés plus rapide et efficace. La prévention, le suivi régulier sont essentiels pour limiter l'impact de la rétinopathie diabétique et améliorer la qualité de vie des patients atteints de diabète. Dès lors, dans le prochain chapitre, nous abordons les concepts généraux de la classification d'images particulièrement la machine learning, le deep learning, quelques approches de classification et en nous parlons de certains travaux liés à la classification d'images rétiniennes.

Chapitre 3 : Analyse Automatique des images médicales

Chapitre 3

Analyse Automatique des images médicales

3.1 Introduction

L'analyse automatique des images médicales est une discipline riche et variée dans le domaine médical, car elle permet de détecter les changements et les schémas pathologiques avec une plus grande précision que le diagnostic traditionnel basé sur les observations humaines, ce qui annule leurs tentatives de comprendre précisément l'image. Par conséquent, des techniques avancées et précises sont apparues dans l'analyse qui réduisent la charge de travail des professionnels de la santé. Développé grâce à l'intégration de technologies avancées telles que l'apprentissage automatique et l'intelligence artificielle (IA). Ces développements sont importants, en les utilisant, les ordinateurs peuvent analyser et extraire des informations de base à partir d'images médicales (rayons X, IRM, etc.). Cela améliore la précision du diagnostic, accélère la détection des maladies et réduit les erreurs humaines.

3.2 Intelligence Artificielle (IA)

3.2.1 Définition

L'intelligence artificielle (IA) est un canal de l'informatique qui cherche à simuler l'intelligence humaine qui prendre des décisions et résoudre des problèmes nécessite de la compréhension, de l'apprentissage et le raisonnement de la raison. C'est essentiellement la capacité d'un système informatisé à montrer des capacités cognitives. Elle a pour but de rendre la machine indéterminable de l'être humain et englobe l'apprentissage automatique et l'apprentissage profond (voir Figure 3.1) [44].

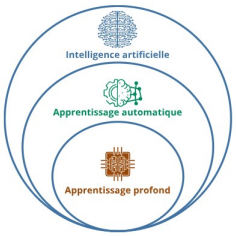


Figure 3.1 – Relation entre IA et ML et DL [45]

L'intelligence artificielle commence par la construction de systèmes experts. Ces systèmes experts sont des systèmes très complexes dont la construction nécessite des milliers d'ingénieurs qui enseignent aux ordinateurs comment accomplir des tâches grâce à une programmation rigoureuse et approfondie règle. Ils consistent généralement en une base de connaissances (contenant des objets, des attributs et conditions) et un moteur d'inférence qui peut utiliser des faits et des règles pour générer des nouveaux faits , jusqu'à ce que la réponse à la question soit obtenue (voir Figure 3.2) qui montre l'architecture de système expert [46].

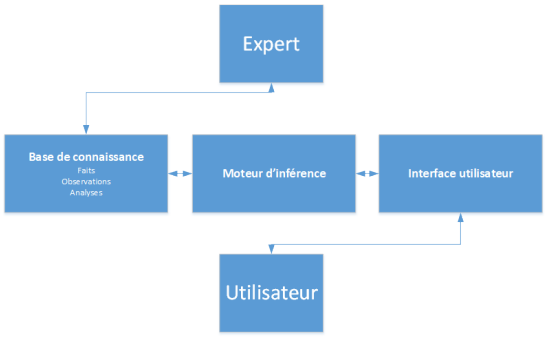


FIGURE 3.2 – L'architecture de système expert [47]

Bien que faciles à mettre en œuvre, ces systèmes experts se révèlent particulièrement difficiles à appliquer à des tâches complexes. Il est presque impossible de résoudre un tel problème à l'aide d'un ensemble de règles prédéfinies. C'est pourquoi l'avènement de l'apprentissage automatique permet aux systèmes informatiques d'apprendre à partir de données plutôt que de s'appuyer uniquement sur une base de connaissances explicite [46].

3.2.2 Apprentissage Automatique (Machine Learning)

3.2.2.1 Définition

L'apprentissage automatique est un sous-domaine de l'intelligence artificielle, ou le terme fait référence à la capacite des systèmes informatiques à trouver indépendamment des solutions aux problèmes en reconnaissant les modèles dans les bases de données. En d'autres termes : la Machine Learning (ML) permet aux systèmes informatiques de reconnaitre des modèles sur la base d'algorithmes et d'ensemble de données existantes et de développer des concepts de solutions adéquates. Par conséquent, dans la Machine Learning, la connaissance artificielle est générée sur la base de l'expérience [Khadidja et al., 2020 [48]]. Il intégré des techniques telles que : l'apprentissage supervisé, l'apprentissage non supervisé et l'apprentissage par renforcement (voir Figure 3.3).

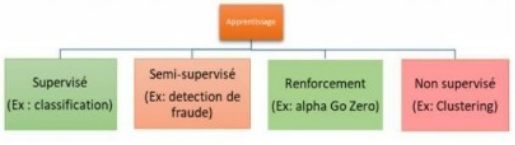


FIGURE 3.3 – Différents types d'apprentissage automatique [Saidj, 2022 [49]]

3.2.2.2 Apprentissage supervisé

L'apprentissage supervisé est l'un des principaux types d'apprentissage machine qui consiste à utiliser un ensemble de données étiquetées pour entraîner un modèle afin de réaliser des prédictions ou des classifications sur de nouveaux exemples. Dans ce type d'apprentissage, l'ensemble de données est divisé en un ensemble de données d'entrainement, qui permet au modèle d'apprendre au fil du

temps, et un ensemble de test, qui évalue la performance du modèle. Le processus d'apprentissage est basé sur une fonction de coût, qui mesure l'écart entre les prévisions du modèle et les valeurs réelles. L'algorithme ajuste ensuite ses paramètres pour réduire cette erreur [50].
Ces tâches d'apprentissage peuvent être classées en deux types principaux (voir Figure 3.4) :

-La classification est une tâche consistant à choisir une classe (valeur) parmi toutes celles possibles, Exemple : un algorithme classifiant une tumeur comme « bénigne » ou « maligne » [Matteis et al., 2022 [51]]. Parmi les algorithmes de classification les plus courants : les machines à vecteurs de support (SVM), les arbres de décision, les k-plus proches voisins (k-NN) ..ect.

-La regression est utilisée lorsque la sortie à prédire peut prendre des valeurs continues, il s'agit d'une variable réelle [Matteis et al., 2022 [51]], par exemple estimer le taux de rotation d'une entreprise. Parmi les algorithmes de régression les plus courants : La régression linéaire, la régression logistique,.. ect .

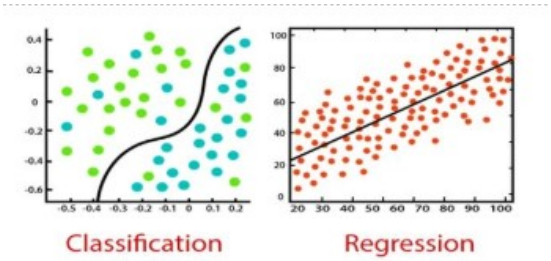


FIGURE 3.4 – Différence visuelle entre Classification et Régression [Anberi et al., 2024 [52]]

Régression linéaire
La régression linéaire est un algorithme statistique utilisé pour modéliser la relation entre deux variables. Elle suppose qu'il existe une relation linéaire entre la variable dépendante (les sorties) et une ou plusieurs variables indépendantes (les entrées) (voir Figure 3.5). Son objectif principal est de trouver la droite la plus appropriée pour décrire cette relation. La meilleure droite est déterminée en réduisant la somme des écarts quadratiques entre les valeurs réelles et les valeurs prédites. Cette approche permet également de faire des prédictions pour de nouvelles valeurs peuvent être prévues en se basant sur les données disponibles [53].
La régression linéaire est exprimée selon la formule suivante :

$$y = aX + b$$

ou X est la variable indépendante, y est la variable dépendante, a est coefficient directeur (pente), b est Constante ou point d'intersection avec l'axe Y [53].

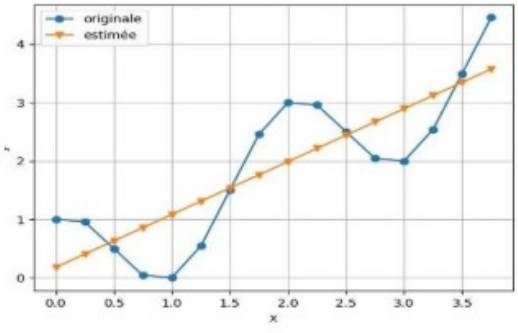


FIGURE 3.5 – Régression linéaire : approximation des données [Saidj, 2022 [49]]

Régression logistique

La régression logistique, également connue sous le nom de modèle Logit, qui est l'un des modèles statistiques les plus couramment utilisés pour estimer la probabilité qu'un échantillon appartienne à une catégorie ou à un événement particulier. Ce modèle permet de traiter plusieurs variables indépendantes, qu'elles soient continues ou catégorielles. Elle est utilisée lorsque la variable cible est binaire (binary classification), c'est-à-dire qu'elle ne prend que deux valeurs (0 ou 1). Elle peut également être généralisée pour inclure une classification à plusieurs catégories, connue sous le nom de régression logistique multinomiale (Multinomial Logistic Regression). Il dépend de l'équation de la régression linéaire, mais utilise la fonction sigmoïde (Sigmoid) pour convertir les valeurs dans une plage comprise entre 0 et 1 (voir Figure 3.6), ce qui la rend particulièrement adapté à l'estimation des probabilités dans des fonctions de classification binaires [54].

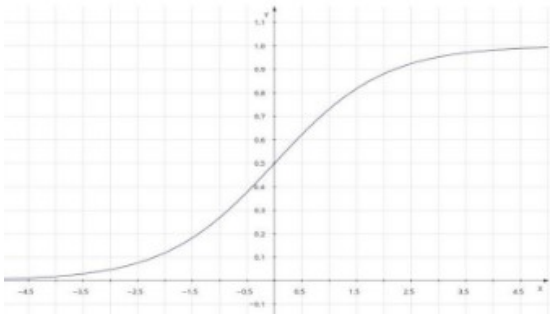


FIGURE 3.6 – Fonction sigmoïde utilisée en régression logistique [Saidj, 2022 [49]]

Les k-plus proches voisins (k-NN)

L'algorithme de classification K-NN notamment dit (en anglais K-Nearest Neighbours) est un algorithme d'apprentissage automatique très important, simple mais robuste, et non paramétrique[1] . Il se base sur le principe de la "similarité des points voisins" (voir Figure 3.7). KNN fonctionne en identifiant les k exemples de formation les plus proches d'une entrée donnée et prédit ensuite la classe ou la valeur en fonction de la classe majoritaire ou de la valeur moyenne de ces voisins [Anberi et al., 2024 [52]]. La performance de KNN dépend de la sélection de la métrique k et de la distance appropriée. Il est utilisé aussi bien pour les tâches de classification que de régression.

– Régression des K-voisins les plus proches : elle prédit les valeurs continues en faisant la moyenne des sorties des k voisins les plus proches [Anberi et al., 2024 [52]].

– Classification des K-voisins les plus proches : les points de données sont classés en fonction de la classe majoritaire de leurs k voisins les plus proches [Anberi et al., 2024 [52]].

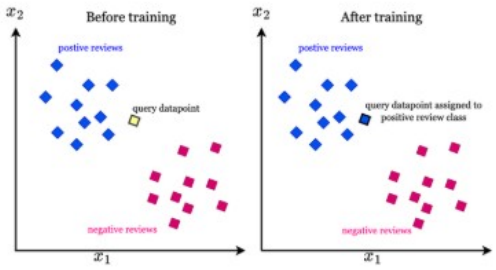


FIGURE 3.7 – K-Plus Proches Voisins (K-NN) Algorithme de classification : applications et exemples concrets [Errouche et Merrouche, 2024 [55]]

1. un algorithme qui ne fait aucune hypothèse sur la structure et la distribution des données.

3.2.2.3 Apprentissage non supervisé

L'apprentissage automatique non supervisé est un type d'apprentissage où le système est entraîné sans aucune données étiquetée ou annotée. En d'autres termes, l'algorithme apprend à identifier des modèles et des relations au sein des données de manière autonome, sans aucune connaissance ou orientation préalable. Cette caractéristique en fait un outil puissant d'analyse de données, en particulier dans les cas où des données étiquetées sont soit indisponibles, soit trop coûteuses à obtenir [56] .On peut distinguer quelques-unes des algorithmes les plus courantes :

– Algorithmes de clustering : regroupent des points de données similaires en fonction de leurs caractéristiques (voir Figure 3.8). Par exemple, si vous avez un ensemble de données pour les achats des clients, les algorithmes clustering peuvent vous aider à identifier des groupes de clients ayant des habitudes d'achat similaires [56].

– Algorithmes de réduction dimensionnelle : réduisent le nombre de variables dans un ensemble de données tout en conservant un maximum d'informations que possible. Cela peut être particulièrement utile lorsque vous travaillez avec de grands ensembles de données contenant de nombreuses variables, car cela peut faciliter leur analyse [56].

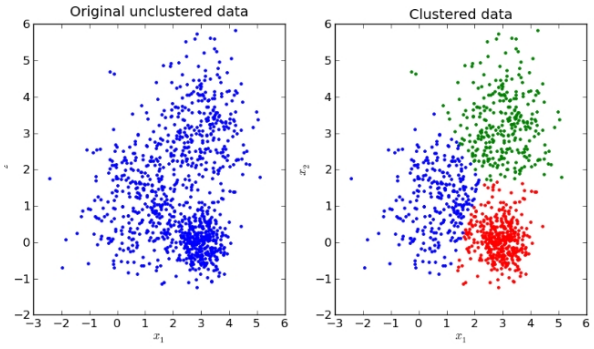


FIGURE 3.8 – Exemple de clustering [57]

3.2.2.4 Apprentissage semi-supervisé

Ce type de d'apprentissage automatique est un mélange d'apprentissage supervisé et non supervisé. Typiquement, une petite quantité de données étiquetées est utilisée pour guider le modèle, tandis que la majorité des données sont non étiquetées [58]. Son objectif est de développer une fonction capable de prédire précisément la variable de sortie en se basant sur les variables d'entrée. Contrairement à l'apprentissage supervisé, l'algorithme est formé sur un ensemble de données contenant à la fois des données étiquetées et non étiquetées. Cette approche s'avère particulièrement bénéfique lorsque de nombreuses données non étiquetées sont disponibles, mais que l'étiquetage complet de toutes ces données est trop coûteux ou difficile [Anberi et al., 2024 [52]].

3.2.2.5 Apprentissage par renforcement

L'apprentissage par renforcement est une branche de l'apprentissage automatique qui permet aux agents logiciels et aux machines d'évaluer automatiquement le comportement optimal dans un contexte ou un environnement particulier pour améliorer son efficacité. Il repose sur le principe de la récompense et de pénalité, où le modèle (Agent - Agent) apprend à prendre des décisions optimales en interagissant avec l'environnement et en expérimentant différentes stratégies pour atteindre l'objectif souhaité. Cela se fait par essais et erreurs, alors que l'agent tente d'exécuter des actions afin d'obtenir la récompense la plus élevée possible à long terme [Anberi et al., 2024 [52]].

3.2.3 Apprentissage Profond (Deep Learning)

En machine Learning, l'extraction des caractéristiques et la classification sont faites de manière séparée. Les données sont ensuite utilisées pour transmettre les caractéristiques, ce qui permettra à un modèle qui sera capable de faire des prédictions. Cette extraction de caractéristiques est extrêmement fatigant, prend beaucoup de temps et nécessite la connaissance d'un expert surtout quand on a un grand nombre de données. D'où l'intervention de l'apprentissage profond. Le deep learning consiste à apprendre automatiquement des caractéristiques à partir d'un jeu de données étiquetées (voir Figure 3.9). En d'autres termes, la machine est approvisionnée en données brutes et la détection des attributs est effectuée de manière automatique [59].

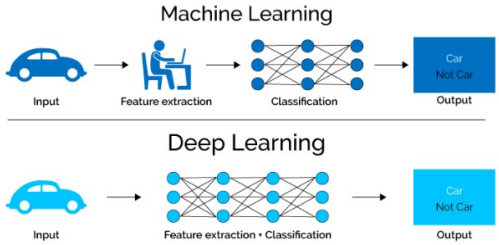


FIGURE 3.9 – deroulement du machine learning et du deep learning [59]

Les réseaux de neurones artificiels sont utilisés dans le Deep learning pour créer des algorithmes capables d'imiter les actions du cerveau humain. Les réseaux sont composés de dizaines de couches de neurones, Chaque couche de neurones dans les réseaux est responsable de recevoir et interpréter les informations de la couche précédente (voir Figure 3.10).

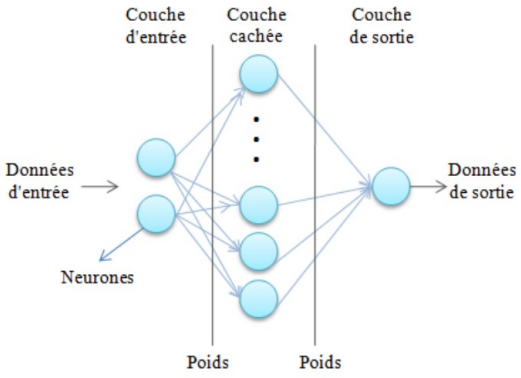


FIGURE 3.10 – Structure de modèle de réseau de neurones artificiels [Ferchichi, 2017 [60]]

Il existe plusieurs modèles de Deep Learning, parmi lesquels on peut citer :

3.2.3.1 Convolutional Neural Networks (CNN)

Les réseaux convolutifs sont une forme particulière de réseau neuronal multicouches conçu pour traiter les données telles que des images, de l'audio ...etc. Ces réseaux de neurones artificiels sont capables de catégoriser les informations des plus simples aux plus complexes. C'est un réseau composé de deux parties, la première partie est la partie convolution et la deuxième est la partie classification. Les réseaux convolutifs sont caractérisé par leurs premières couches convolution. La couche de convolution occupe une place centrale dans ce type de réseau. La première partie applique généralement trois types d'opération a d'extraire les informations juste. Ces opérations sont la convolution, le pooling et la fonction d'activation [61] (voir Figure 3.11).

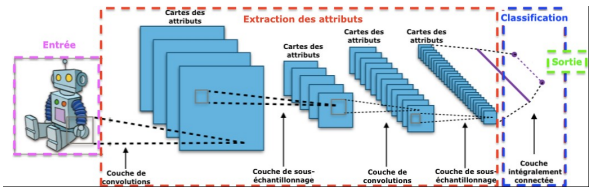


FIGURE 3.11 – Architecture standard d'un réseau à convolutions [61]

1. Couche de convolution
   Cette couche est l'élément de base d'un CNN. Son objectif est de repérer toutes les caractéristiques présentes dans les images reçues en entrée. Pour cela, on réalise un filtrage par convolution. Nous définissons une taille de fenêtre qui s'étendra sur toute l'image (voir Figure 3.12). Au début de la convolution, la fenêtre sera tout en haut à gauche de l'image, puis elle se déplacera d'un certain nombre de cases vers la droite, et lorsqu'elle arrivera à la fin de l'image, elle se déplacera d'un pas vers le bas, et ainsi de suite jusqu'à ce que le filtre couvre toute l'image [62].

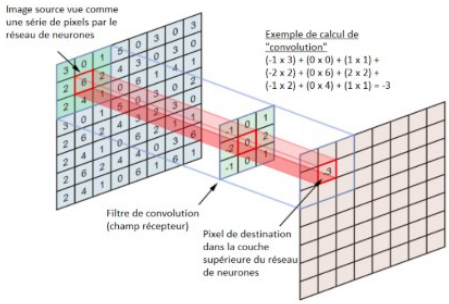


FIGURE 3.12 – Opération de convolution appliquée à la reconnaissance d'images [63]

2. Couche pooling
   C'est la deuxième étape essentielle dans l'élaboration d'un réseau de neurones convolutifs. Une couche de pooling est une couche additionnelle entre deux couches de convolution.Il existe plusieurs techniques telles que le Maxpooling , l'AveragePooling...etc. Le Maxpooling est le plus utilisé. ; Cela revient à prendre la valeur la plus élève de la fenêtre et à exclure les autres (voir Figure 3.13). Une méthode efficace pour travailler avec de grandes images, car elle permet de réduire la taille des images tout en conservant leurs caractéristiques pertinentes [63].

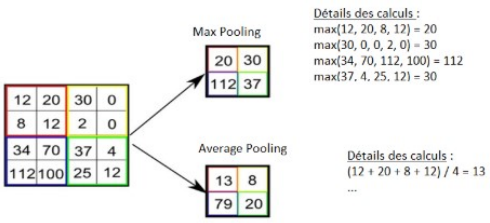


FIGURE 3.13 – Illustration des techniques de Max pooling & Average pooling [63]

3. Couches entièrement connectées
   Ils représentent les parties catégorielles de chaque neurone qui se connectent à tous les neurones, tel que la relation entre la couche supérieure et la couche inférieure (voir Figure 3.14). Après avoir reçu les vecteurs en entrée, les couches entièrement connectées appliquent des combinaisons linéaires en séquence. Ensuite, il y a une fonction d'activation pour produire la sortie neuronale. Cette sortie devient alors l'entrée de la couche suivante de neurones [62].

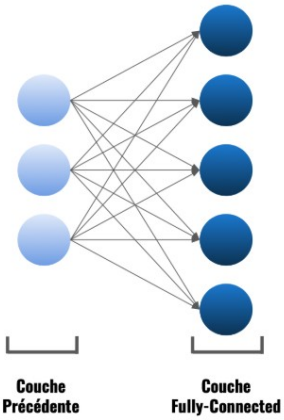


FIGURE 3.14 – Illustration de la couche entièrement connectée [64]

4. Fonction d'activation
   Cela permet d'augmenter l'efficacité du traitement en insérant des couches qui effectueront des opérations mathématiques entre les couches de traitement. Il existe de nombreux types de fonctions d'activation (voir Figure 3.15), telles que la fonction ReLU, sigmoïde, tanh, softmax... etc. La fonction la plus couramment utilisée en pratique est la fonction ReLU (voir Figure 3.16), elle renvoie une valeur positive [65]. La définition est la suivante :

$$F(x) = \begin{cases} x, & \text{si } x \geq 0 \\ 0, & \text{si } x < 0 \end{cases}$$

   Ce qui signifie : Si x est positif ou nul, alors $F(x) = x$, sinon $F(x) = 0$.

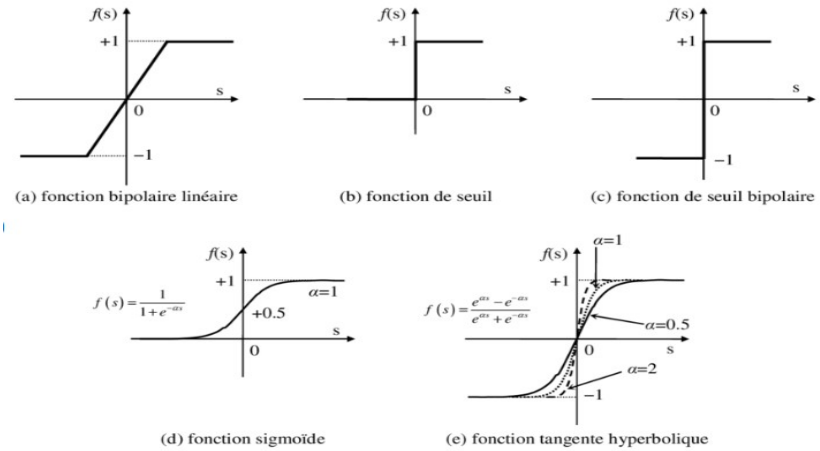


FIGURE 3.15 – Fonctions d'activation d'un neurone artificiel [Bechouche, 2013 [66]]

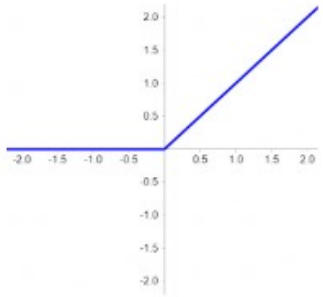


FIGURE 3.16 – exemple de fonctions d'activation ReLu [67]

3.2.3.2 Les types d'architectures CNN

Actuellement, les algorithmes les plus couramment utilisés parmi les techniques innovantes de l'intelligence artificielle (IA) sont les CNNs.
CNN commence les études neurobiologiques, ils ont établi les bases de nombreux modèles cognitifs. Au fil des décennies, différentes tentatives ont été menés pour améliorer les performances des réseaux neuronaux convolutionnels (CNN), de nombreuses architectures bien connues existent. Les CNN se trouvent dans de nombreux types d'architectures, conviennent à différentes tâches telles que la classification d'images, la segmentation et même la détection d'objets. Parmi celles-ci, les architectures CNN les plus largement reconnues(voir Figure 3.17) :

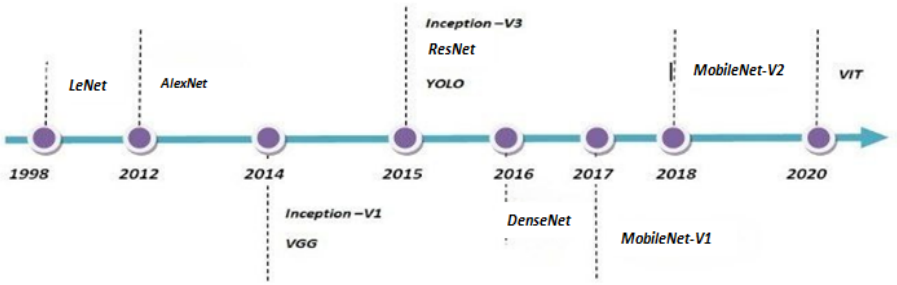


FIGURE 3.17 – évolutive des architectures de CNN

1. LeNet-5 [LeCun et al.,1998 [68]] se compose de 3 couches entièrement connectées et de 2 couches convolutionnelles, ainsi que d'une couche de pooling et de poids entraînables (voir Figure 3.18). Le modèle contient environ 60 000 paramètres. Année de publication : 1998, Créé par Yann LeCunn en adaptant un style d'arrière-plan à l'architecture du réseau neuronal convolutionnel de Fukushima. LeNet-5 sert de modèle de référence pour les CNN contemporains, car tous les CNN affilié à la structure d'empilement de couches convolutionnelles et de pooling, se terminant par une ou plusieurs couches entièrement connectées.

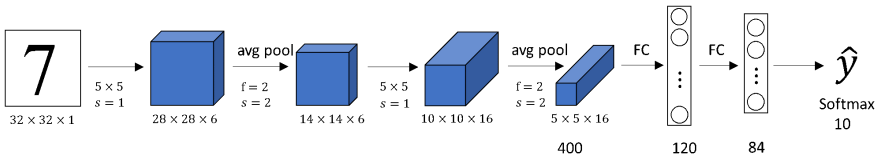


FIGURE 3.18 – Architecture de LeNet-5 [69]

2. AlexNet [Krizhevsky et al., 2017 [70]] est l'architecture CNN la plus connue, elle a gagné le défi ImageNet en 2012. Les auteurs ont utilisé deux techniques de régularisation pour réduire le sur apprentissage, le dropout (Comprend la désactivation des nombres de neurones randomisés au niveau de l'entraînement) et l'augmentation des données (Cela inclut la génération de données de formation supplémentaires à partir d'exemples existants). Il se compose de cinq couches convolutionnelles, de trois couches de pooling et de trois Entièrement connecté et 60 millions de paramètres (voir Figure 3.19)

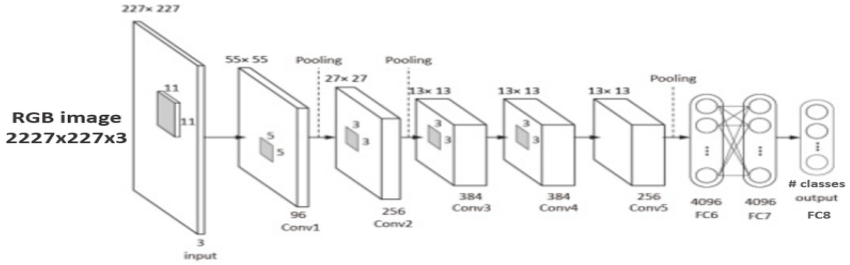


FIGURE 3.19 – Architecture d'AlexNet [Khvostikov et al., 2018 [71]]

3. VGG [Simonyan et all., 2014 [72]] Créé au laboratoire de recherche du Visual Geometry Group (VGG) de l'Université d'Oxford, VGG a obtenu la 2e place au concours ImageNet 2014. Il existe deux variantes : VGG16, qui se compose de 16 couches pondérées, et VGG19, comportant 19 couches pondérées. Ces couches intègrent des couches de regroupement. Contrairement à d'autres réseaux qui utilisent des tailles de filtre variables pour chaque couche, toutes les couches convolutionnelles de VGG utilisent de petits filtres mesurant 3x3. Il utilise principalement Maxpooling. Après les couches convolutionnelles et de regroupement, le modèle comprend plusieurs couches entièrement connectées pour la classification, généralement suivies d'une couche softmax pour la classification multi-classe. Pour introduire la non-linéarité dans le modèle, la fonction ReLU sert de fonction d'activation (voir Figure 3.20).

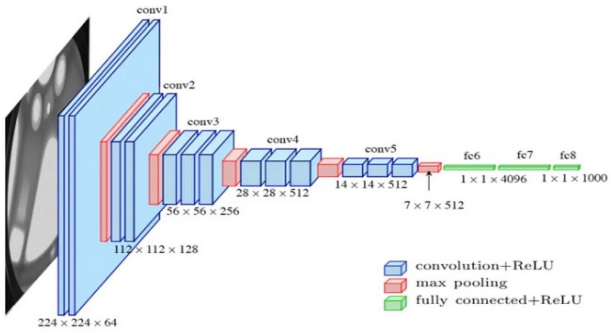


FIGURE 3.20 – Architecture de VGG [73]

4. Inception-v3 [Szegedy et al., 2016 [74]], il introduit par Christian Szegedy et ses collègues en 2015, est une itération améliorée d'Inception-V1 et V2, qui comportait 24 millions de paramètres et fonctionnait sur 48 couches. L'objectif principal d'Inception-V3 était de minimiser les dépenses de calcul des réseaux profonds tout en conservant leurs capacités de généralisation. Elle ont remplacé les grands filtres (5x5 et 7x7) par des filtres asymétriques plus petits (1x7 et 1x5), et ils ont converti la convolution 5x5 en une combinaison de deux opérations de convolution 3x3. (voir Figure 3.21)

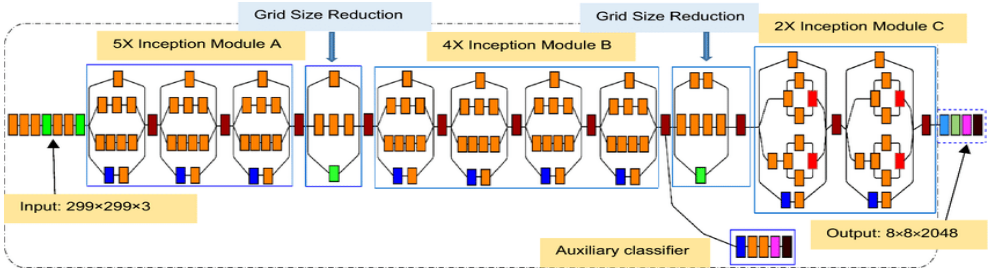


FIGURE 3.21 – Architecture de Inception-v3 [Singh et Vishwakarma, 2021[75]]

5. ResNet [He et al., 2016 [76]], appelé Deep Residual Networks, représente une architecture CNN développée pour relever le défi de la classification d'images complexes. Ce concept a été introduit dans l'article de 2015 intitulé Deep Residual Learning for Image Recognition par Kaiming He et all. La méthodologie consiste à utiliser des blocs résiduels pour améliorer la formation des réseaux profonds, permettant aux informations de contourner des couches spécifiques au lieu de traverser chaque couche dans l'ordre. Cela est accompli grâce à l'incorporation de connexions de saut ou de connexions résiduelles, qui combinent l'entrée d'origine avec les sorties des couches intermédiaires. Les composants fondamentaux de ResNet sont constitués de blocs de conv et d'identité avec la mise en œuvre de nombreuses connexions de saut (voir Figure 3.14). Le réseau est capable de progresser même lorsque plusieurs couches n'ont pas encore commencé le processus d'apprentissage (voir Figure 3.22).

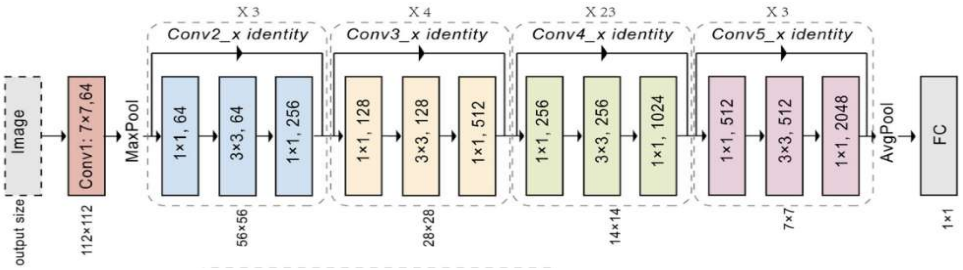


FIGURE 3.22 – Architecture de ResNet [77]

6. DenseNet [Gao Huang et al., 2016 [78]] est une architecture de réseau neuronal convolutif (CNN), introduite par Gao Huang et al. dans un article publié initialement sur arXiv en août 2016, ensuite présenté officiellement à la conférence CVPR en 2017. Cette architecture repose sur des blocs denses et des couches de transition, les couches convolutives de chaque bloc sont combinées pour former plusieurs blocs, et chaque couche reçoit toutes les sorties précédentes en sortie. Ces variantes populaires : DenseNet-121 (voir Figure 3.23), DenseNet-169, DenseNet-201, Le chiffre indique le nombre total de couches dans le modèle.

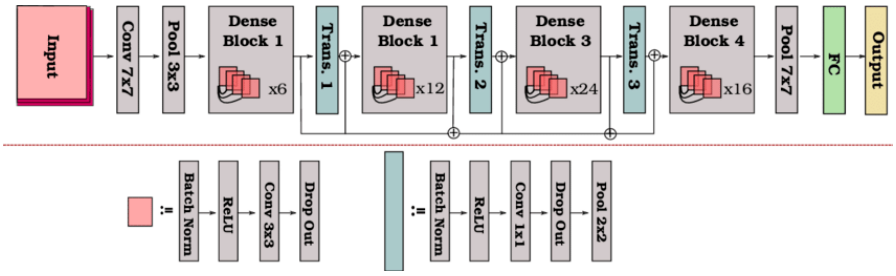


FIGURE 3.23 – Une illustration schématique de l'architecture DenseNet-121 [Radwan, 2019 [79]]

3.2.3.3 Autres approches du Deep Learning

1. Transformers / ViT (Vision Transformer) : Les Transformers ont initialement été développés pour des tâches de traitement du langage naturel (NLP). Introduits par Vaswani et al. en 2017 [80], il est caractérisé par leur mécanisme innovant de self-attention. Le succès des Transformers dans le NLP a motivé leur adaptation à d'autres domaines, notamment la vision par ordinateur. [Dosovitskiy et al., 2020 [81]] ont proposé le Vision Transformer (ViT), un modèle qui applique les mêmes principes que les Transformers textuels, mais sur des images. Dans ViT l'image est divisée en séquences et préserve la structure spatiale grâce au codage positionnel. Ces patchs capturent les relations globales en utilisant une architecture Transformer self-attention. Un token de classification sont ajoutés et traités pour effectuer des prédictions finales (voir Figure 3.24).

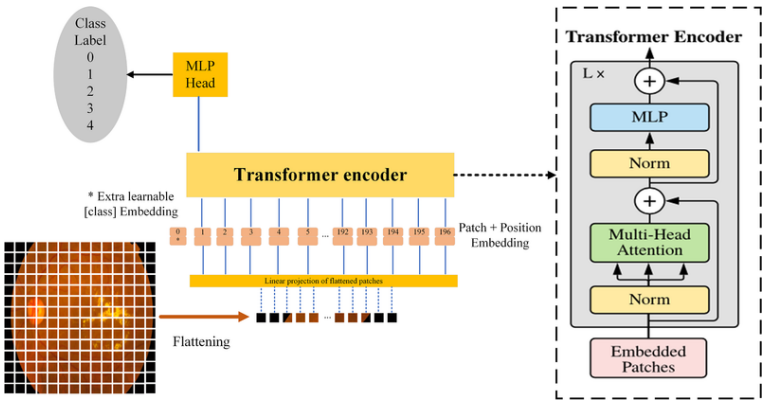


FIGURE 3.24 – Schéma de l'architecture ViT [Wu et al., 2021 [82]]

2. YOLO (You Only Look Once) [Redmon et al., 2016 [83]] est une famille de systèmes de détection d'objets en temps réel basés sur des réseaux de neurones convolutifs, initialement introduits en 2015 par Joseph Redmon et ses collègues. Il applique une passe avant à l'image entière, la divise en grilles et prédit les cadres de délimitation et les probabilités pour chaque région (voir Figure 3.25). Chaque itération de YOLO apporte des améliorations en termes de vitesse et de précision. Les principales versions de YOLO sont les suivantes : YOLOv1 (2015), YOLOv2 (2016), YOLOv3 (2018), YOLOv4 (2020), il ya d'autres version proposé par Ultralytics comme YOLOv5(2020) et YOLOv8(2023).

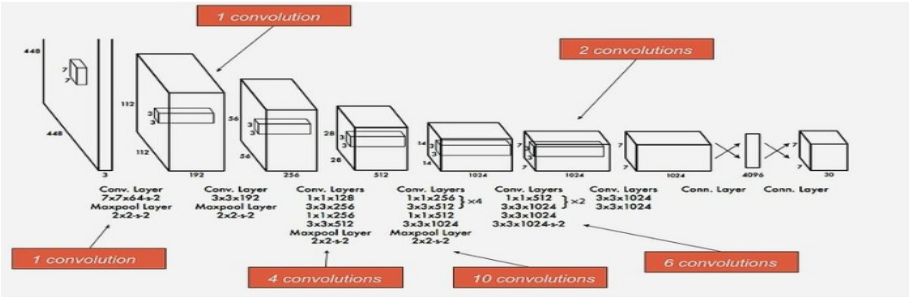


FIGURE 3.25 – Schéma de l'architecture YOLO [Shenoda, 2023 [84]]

3.2.3.4 Transfert learning

L'apprentissage par transfert englobe un ensemble de techniques qui facilitent l'application des connaissances acquises en traitant des problèmes particuliers pour résoudre différents problèmes. En règle générale, ll est important de noter que les modèles utilisés dans ce domaine demandent un temps de calcul important et des ressources importantes. Cependant, en employant des modèles pré-entraînés tels que une base, l'apprentissage par transfert permet la création rapide de modèles très performants et la résolution efficace de problèmes complexes [85] (voir Figure 3.26).

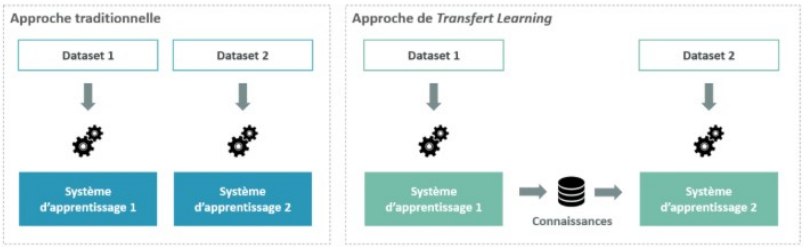


FIGURE 3.26 – Illustration du tranfert learning [85]

L'apprentissage par transfert vous permet de réutiliser des modèles pré-entraînés pour de nouvelles tâches. Les principales technologies sont :

1. Feature Extraction : nous utilisons des modèles pré-entraînés comme extracteurs de fonctionnalités et formons uniquement le classificateur final.
2. Fine-Tuning : certaines couches du modèle sont recyclées en fonction de nouvelles données pour mieux s'adapter.
3. Adaptation Domain-Specific : adapter les modèles à des données différemment distribuées (par exemple, des images médicales).
4. Multi-Task Learning : le modèle apprend plusieurs tâches simultanément pour une meilleure généralisation.
5. Few-Shot Learning / Meta-Learning : apprenez rapidement avec une petite quantité de données.

Rendre les poids de l'ancien modèle entraînables peut aider à spécialiser le nouveau modèle pour le nouveau jeu de donnés disponible [85].

3.2.3.5 Le sur-ajustement (Overftting)

Le surajustement se produit lorsqu'un modèle se concentre tellement sur les détails et le bruit des données d'entraînement qu'il perd sa capacité à se généraliser à de nouvelles données (voir Figure 3.27). Cela se traduit par une erreur de formation plus faible mais une erreur de validation/test plus élevée. Il y a plusieur causes de overfitting :

- Le modèle est trop complexe en termes de couches, de neurones et de paramètres.
- Les données ne sont pas suffisantes ou ne sont pas représentatives.
- Le modèle mémorise au lieu de généraliser, ce qui entraîne une surcharge d'entraînement.
- Il manque des mesures de régularisation (Dropout)

L'inverse est le sous-apprentissage (underfitting), qui se produit lorsque les données d'entraînement peuvent encore être améliorées. Lorsque les erreurs de prédiction des ensembles de données de formation et de validation sont élevées et la différence entre les deux est très faible et le modèle est dit sous-ajusté [86].

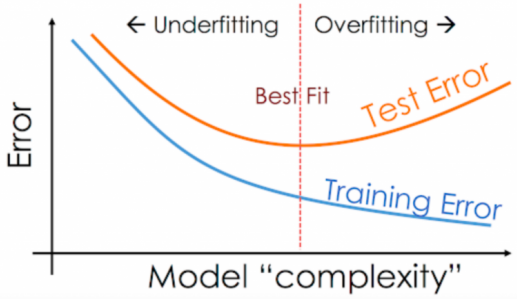


FIGURE 3.27 – Diagramme represente Overfitting et Underfitting [87]

Dans le cas où il y aurait peu d'exemples d'apprentissage, le modèle peut parfois apprendre des bruits ou les performances se dégradent en raison de la présence de détails indésirables dans les exemples d'apprentissage du modèle sur les nouveaux exemples (test et validation) [86]. Il existe plusieurs techniques pour lutter contre ce phénomène dans le processus d'apprentissage, il est important de prendre des mesures telles que :

- Utilisation de Data Augmentation ;
- Utilisation de Dropout dans le modèle afin désactiver les neurones d'une manière aléatoire pendant le processus d'apprentissage ;

- Ajustement des hyperparamètres (nombre de couche, type de fonction d'activation, learning rate, optimiseur [SGD, Adam,...], batch-size, nombre d'époque,...) ;
- KFold Cross Validation ;
- Arrêt précoce (EarlyStopping) ;
  ... etc.

3.3 Classification d'images rétiniennes

3.3.1 Définition

La classification d'images consiste à attribuer des étiquettes aux images selon des catégories prédéfinies. Cette procédure implique l'analyse de l'image d'entrée pour obtenir les classifications appropriées. Dans le domaine médical, notamment pour le diagnostic de maladies oculaires comme la rétinopathie diabétique et le glaucome, la segmentation d'images joue un rôle important. Si la segmentation manuelle est laborieuse et requiert une expertise, de nombreuses propositions ont été faites pour la segmentation automatique des vaisseaux sanguins dans les images rétiniennes, visant à améliorer la précision et l'efficacité des diagnostics médicaux grâce aux technologies d'apprentissage automatique et de vision par ordinateur.

3.3.2 Types de classifications

3.3.2.1 Classification Binaire (Binary Classification)

Aussi appelée classification binomiale, est une transformation de données conçue pour attribuer des membres d'un ensembles en deux groupes disjoints (voir Figure 3.28), selon que les éléments ont ou non des attributs/ fonctionnalité donnée. De plus, l'étiquette Y aura deux valeurs possibles 0 ou 1. Par exemple, un modèle d'apprentissage automatique qui classe la maladie de tumeur, la classe (étiquette) 1 pour dire qu'il s'agit d'une tumeur maligne et la valeur 0 pour les tumeurs bénignes [88].

x	y
X_1	t_1
X_2	t_2
X_3	t_1
X_4	t_2
X_5	t_1

FIGURE 3.28 – Exemple de classification Binaire [89]

3.3.2.2 Classification Multi-classes

Tant que le nombre d'étiquettes possibles est supérieur à 2 (voir Figure 3.29), on parle de classification multi-classe. La reconnaissance d'objets dans les images, le diagnostic médical, la détection du type de véhicule dans les images, la reconnaissance des émotions faciales sont des exemples de classification multi-classes [90].

X	y
X_1	t_2
X_2	t_3
X_3	t_4
X_4	t_1
X_5	t_3

Multi-class Classification

FIGURE 3.29 – Exemple de classication multi-classes [89]

3.3.2.3 Classification Multi-label

Il s'agit d'une amélioration par rapport à la classification traditionnelle dans laquelle les catégories ne s'excluent pas mutuellement, permettant à chaque individu d'appartenir à une ou plusieurs classes à la fois(voir Figure 3.30). Un grand nombre d'applications nécessitent ce type de classification actue [91].

X	y
X_1	$[t_2, t_5]$
X_2	$[t_1, t_2, t_3, t_4]$
X_3	$[t_3]$
X_3	$[t_2, t_4]$
X_3	$[t_1, t_3, t_4]$

Multi-label Classification

FIGURE 3.30 – Exemple de classication multi-label [89]

3.3.3 État de l'art

Cette section est dédiée à la présentation de quelques travaux dans le domaine de la classification d'images de la rétinopathie diabétique. Nous avons étudié des approches basée sur les modèles de deep learning (apprentissage profond).

Il existe de nombreux ouvrages et études différentes dans la littérature sur la Classification d'images de la rétinopathie diabétique. Nous avons sélectionné certains articles pour notre recherche.

Dans ce cadre [Pao et al., 2020 [92]] ont proposé une technique d'utilisation des (CNN) bicanaux combinant des images d'entropie et un prétraitement par UnsharpMasking dans le but d' améliorer la détection de la rétinopathie diabétique. ils ont utilisé APTOS 2019 de Kaggle comme une base de donnees dans cette étude, prises à l'aide de la photographie du fond d'œil dans diverses conditions d'imagerie (images couleur RVB, sous format png), il contient un grand ensemble d'images de la rétine (35 126 images) sont classées en cinq niveaux de gravité : pas de RD, légère, modérée, sévère, proliférante. Dans un premier temps, Application d'extraction du composant vert (meilleure visibilité des vaisseaux et des lésions) et conversion en niveau de gris et appliquer unsharpmasking. Après

redimensionnees les images a 100x100 pixels, ensuite calcul local de l'entropie (nxn = 9x9) pour représenter l'hétérogénéité et Les valeurs sont normalisées entre 0 et 1 pour l'entrée au CNN.
Le modèle bicanal CNN a donné des performances de détection pour la rétinopathie diabétique avec une précision de 87,83 %, une sensibilité de 77,81 % et une spécificité de 93,88 %. Ces résultats surpassent ceux d'une approche basée sur un seul canal (précision de 86,10 %, sensibilité de 73,24 % et spécificité de 93,81 %). De plus, l'AUC de la courbe ROC a atteint 0,93, une amélioration significative par rapport à 0,87 pour le CNN entraîné sur des photographies originales.

Dans ce cadre [Shoaib et al., 2024 [93]] ont proposé un modèle personnalisé nommé DiaCNN pour le diagnostic de la rétinopathie diabétique, basé sur le transfer learning. et comparé à InceptionResNetv2, Inceptionv3. Le dataset utilisé est ODIR (Ocular Disease Intelligent Recognition), comprenant des images de fond d'œil annotées pour détecter la rétinopathie diabétique. il contient des images étiquetées avec des classes représentant différentes étapes de la rétinopathie diabétique, allant de l'absence de la maladie à des stades plus avancés. Le dataset comprend 5000 photographies en couleur du fond d'œil, réparties en ensembles d'entraînement et de test . Dans un premier temps, ils ont applique redimensionnement des images,et Augmentation des données et la normalisation.
Pour la phase de test de modèle DiaCNN, la Précision de 98,3 %, pour le modèle InceptionResNetv2,la Précision de 97,5 %, pour le modèle Inceptionv3,la Précision de 97,5%.

Dans un autre travail sur les images de la rétine, [KHalifa et al., 2019 [94]] ont conçu un réseau basé sur transfert Learning d'apprentissage profond (AlexNet, ResNet18, SqueezeNet, GoogleNet, VGG16 et VGG19), dans le but de détecter la rétinopathie diabétique par transfert profond des modèles d'apprentissage profond. La base de données utilise c'est APTOS 2019 Blindness Detection, en provenance de l'Asia Pacific Tele-Ophthalmology Society, elle contient un grand ensemble d'images de la rétine sont classées en cinq niveaux de gravité : pas de RD, légère, modérée, sévère, proliférante. (images couleur RVB, 224x224 pixels sous format png). Les choix viennent des modèles AlexNet, VGG16, VGG19, ResNet18, SqueezeNet et GoogleNet. Toutes ces architectures ont des couches relativement moins nombreuses qui diminuent le temps d'entraînement et la complexité des calculs. Les architectes tels que Xception et DenseNet.
Les métriques révèlent qu'AlexNet a réalisé les meilleures performances, avec une précision de 96,23 %, un rappel de 95,42 % et un score F1 de 95,82 %. VGG16 et VGG19 ont également présenté des résultats compétitifs avec des précisions respectives de 95,19 % et 94,64 %. SqueezeNet a obtenu des performances plus modestes avec une précision de 82,80 %.

[Aswathi et al., 2021 [95]] ont présenté une technique utilisation les models de deep learning et transfer learning (VGG19, InceptionV3, MobileNet, ResNet et NASNet) pour détecter et classifier la rétinopathie diabétique. Ils ont utilise jeu de données Messidor, il contient un grand ensemble d'images de la rétine (1200 images) de fond d'œil, capturées sous divers angles et éclairages, Images rgb. Un clinicien a évalué chaque image en fonction de la gravité de la rétinopathie diabétique sur une échelle de 0 à 4. Les images ont été prétraitées à l'aide des techniques CLAHE et Power law Transformation pour améliorer le contraste et la netteté. Les architectures CNN utilisées sont InceptionV3, ResNet50et VGG19 pour la classification binaire, et MobileNet et NASNet pour la classification multi-classes. InceptionV3 a été choisi comme modèle principal en raison de ses meilleures performances sur la rétinopathie diabétique. L'architecture, entraînée sur ImageNet, comprend des couches convolutionnelles, de pooling, des modules Inception et des dropouts pour

éviter le surapprentissage, garantissant un équilibre optimal du flux d'information.
Classification binaire testée avec VGG19, ResNet et InceptionV3en utilisant les optimiseurs Adadelta, Adam et SGD ,l'architecture inceptionV3 prè-entrainée a donné les meilleurs resultats parmi les modeles testés. La précision des modèles est similaire entre 48,75 % et 51,25 % pour la classification multi-classes.

Dans les années récentes, l'utilisation des réseaux de neurones profonds a permis d'atteindre des performances élevées dans la classification automatique des cellules de la rétine, ce qui nous a amenés à étudier l'approche de [Alwakid et al., 2023 [96]] ont proposé une technique utilisation les models de deep learning pour détecter et classifier la rétinopathie diabétique. ils ont utilisé un ensemble de données de kaggle dans cette étude, la base e donneés utiliseé est DDR 12255 images en RGB et également classes en 5 classes, de dimension varaible . Ces images sont ensuite prétraitées (Redimensionnement, normalisation et augmentation des donnée …), l'utilisation DenseNet121 et utilisation des hyperparamètres taux d'apprentissage, taille du batch, nombre d'époques, fonctions d'activation et de perte.
le résultat principal est le DenseNet121 qui est efficace pour la détection et la classification de la rétinopathie diabétique, c'est précision 79,67%.

Dans le même contexte, [Mohanty et al., 2023 [97]] ont proposé des modèle hybride VGG16-XGBoost basé sur l'apprentissage profond pour détecter et classifier la rétinopathie diabétique. Ils ont utilisé APTOS 2019 Blindness Detectionde kaggle, il contient un grand ensemble d'images de la rétine (3663 images) de fond d'œil, capturées sous divers angles et éclairages. Avec extension (.png) et une dimension de 1836x1216 pixels. Un clinicien a évalué chaque image en fonction de la gravité de la rétinopathie diabétique sur une échelle de 0 à 4 : pas de RD, légère, modérée, sévère, proliférante. Ces images sont redimensionnées en 224x224 pixels pour l'adaptation aux réseaux de neurones convolutifs. Apres appliquer la normalisation et l'augmentation de données (rotation, zoom , …), utilise VGG16 pour l'extraction des caractéristiques et le classifieur XGBoost pour classer les images.
Modèle hybride VGG16-XGBoost a atteint une précision de 79,50% dans la détection et la classification de la RD.

C'est dans ce cadre [Benabdessalam et al., 2023 [98]] ont proposé une technique de comparaison de 6 architectures CNN (MobileNetV2, DenseNet-121, EfficientNetB5, ResNet101, Inception-V3 et VGG19) dans le but de fournit une évaluation objective pour aider au choix des modèles les plus performants pour détecter la maladie. La base de données utilisée dans cette étude est APTOS 2019 à partir de Kaggle d'images réelles de la rétine (3662 images) de fond d'œil en RVB, de taille 224x224 pixels sous format png. Un clinicien a évalué chaque image en fonction de la gravité de la rétinopathie diabétique sur une échelle de 0 à 4 : pas de RD, légère, modérée, sévère, proliférante.
Dans un premier temps, redimensionnées les images, après un prétraitement est réalisé pour extraire les caractéristiques appliquant un filtre CLAHE aux images pour améliorer les contrastes. Par la suite, faire la classification basée sur les caractéristiques extraites. Ces architectures présentent une performance globale solide pour la classe 0, ont obtenu des valeurs élevées de précision, de rappel et de F1-score 0.95%. Dans la classe 1 les architectures MobileNetV2, DenseNet-121 présentant une meilleure performance tel que MobileNetV2 avec une précision 0.93%, rappel 0.87%, F1-score 0.88% et DenseNet-121 avec une précision 0.92%, rappel 0.85%, F1-score 0.92% et les autres architectures ont également obtenu de bonnes performances, mais légèrement inférieures.

Pour la classe 2 les architectures MobileNetV2, DenseNet-121 présentant des performances supérieures tel que MobileNetV2 avec une précision 0.93%, rappel 0.93%, F1-score 0.93% et DenseNet-121 avec une précision 0.90%, rappel 0.96%, F1-score 0.92% et les autres architectures ont montré une légère dégradation des performances, avec des précisions variantes entre 0,87% pour EfficientNetB5, 0,88% pour InceptionV3, 0,84% pour ResNet-101 et 0,89% pour VGG19.
Dans la classe 3, toutes les architectures ont affiché des performances relativement faibles par rapport aux autres stades de gravité. Les précisions varient entre 0,75% pour VGG19, 0,7% pour ResNet-101, 0,72% pour InceptionV3, 0,87% pour DenseNet-121, 0,73% pour MobileNetV2 et 0,68% pour EfficientNetB5. Dans la classe 4, MobileNetV2 a obtenu la meilleure précision avec 0,88%, suivie de près par DenseNet-121 avec une précision de 0,85% et les autres architectures ont affiché des performances légèrement inférieures, avec des précisions de 0,81% pour InceptionV3, 0,78% pour ResNet-101 et 0,83% pour VGG19.

Aussi dans ce cadre que [Alyoubi et al., 2021 [99]] ont proposé une technique utilisent une architecture personalisé CNN512 fusionné avec YOLOv3 dans le but de détecter la rétinopathie diabétique (localisaé précisement des lésions). Ils ont utilisé une base de données dans cette étude, DDR à partir de Kaggle d'images réelles de la rétine de fond d'œil en RVB, de taille varaible sous format png. Un clinicien a évalué chaque image en fonction de la gravité de la rétinopathie diabétique sur une échelle de 0 à 4 : pas de RD, légère, modérée, sévère, proliférante. Dans un premier temps, Application de l'augmentation des données pour obtenir les meilleurs résultats, CNN512 construit sur mesure pour classer les images da la rétinopathie diabétique , et YOLOv3 adapté pour la localisation des lésions rétinopathie diabétique. Les performances en précision (CNN512+YOLOv3) est 89%. Dans cette partie nous avons présenté différents travaux de la littérature relatifs au domaine de

notre projet. Afin de simplifier et résumer ces différentes méthodes, le tableau 3.1 représente un récapitulatif sur l'ensemble des méthodes résumées ci-dessus.

Auteurs	**Dataset**	**Méthodologies**	**Résultats**
[Pao et al., 2020 [92]]	APTOS 2019 Blindness Detection (35 126 images)	utilisation des (CNN) bicanaux combinant des images d'entropie et un prétraitement par UnsharpMasking pour améliorer la détection de la rétinopathie diabétique.	Accuracy=87.8% Sensitivity=77.8%
[Shoaib et al., 2024 [93]]	ODIR(Ocular Disease Intelligent Recognition),	L'étude propose un modèle personnalisé nommé DiaCNN pour le diagnostic de la rétinopathie diabétique. et comparé à VGG16, ResNet50 et DenseNet121.,	precision DiaCNN= 98,3 % InceptionResNetv2= 97.5% Inceptionv3= 97.5%
[KHalifa et al., 2019 [94]]	APTOS 2019	Les modèles de transfert d'apprentissage profond (AlexNet, ResNet18, SqueezeNet, GoogleNet, VGG16 et VGG19) pour améliorer la détection et la classification de la rétinopathie diabétique .	Précision : AlexNet=96.23% VGG16=95.19% VGG19=94.64% ResNet18=93.75% SqueezeNe=82.8% GoogleNet=94.92%
[Aswathi et al., 2021 [95]]	Messidor, il contient (1200 images) de fond d'œil	utilisation des modèles CNN pré-entraînés et prétraitement des images (CLAHE, Powerlaw) pour améliorer la classification automatique de la rétinopathie diabétique	(VGG19, InceptionV3, MobileNet, ResNet, NASNet) ont obtenu des précisions similaires en classification multi-classes, allant de 48,75 % à 51,25 %
[Alwakid et al., 2023 [96]]	DDR (Dataset for Diabetic Retinopathy)	Utilisation de DenseNet121 pour l'analyse d'images médicales et la classification de la rétinopathie diabétique.	précision = 79,67%
[Mohanty et al., 2023 [97]]	APTOS 2019 Blindness Detection	Proposition un modèle hybride VGG16-XGBoost, combine entre architecture VGG16 pour extraction des données et le classifieur XGBoost.	Accuracy : VGG16-XGBoost=79.5%

[Benabdesalam et al., 2023 [98]]	APTOS 2019 Blindness Detection	Comparaison de six architectures CNN (MobileNetV2, DenseNet-121, EfficientNetB5, ResNet101, Inception V3 et VGG19) pour la classification des images de fond d'œil.	MobileNetV2 Précision=0.88% DenseNet-121 Précision=0.92% EfficientNetB5 Précision=0.778% ResNet-101 Précision=0.81% VGG19 Précision=0.856% InceptionV3 Précision=0.842%
[Alyoubi et al., 2021 [99]]	DDR (Dataset for Diabetic Retinopathy)	Utilise une architecture personalisé CNN512 fusionné avec YOLOv3 pour détecter la rétinopathie diabétique.	précision=89%

TABLE 3.1 – Récapitulatif de certaines études proposées dans la littérature pour la classification des images de la Rétinopathie Diabétique

3.4 Conclusion

La classification des images joue un rôle essentiel dans de plusieurs domaines, de la reconnaissance faciale à la médecine. Porté par le développement et les progrès continus de l'intelligence artificielle. L'apprentissage automatique et la classification d'images offrent des solutions de plus en plus complexes pour l'analyse et l'interprétation visuelle. Dans le dernier chapitre, nous montrerons le modèle proposé, le langage utilisé, lanalyse des resultats avec la partie discussion.

Chapitre 4 : Resultas et Experimentations

Chapitre 4

Resultas et Experimentations

4.1 Introduction

Le diagnostic de diverses maladies ophtalmologiques repose en grande partie sur l'examen du fond d'œil. Cette analyse est particulièrement centrée sur les vaisseaux sanguins de la rétine qui sont essentiels pour le maintien de la santé humaine et servent d'indicateurs clés dans le diagnostic de maladies graves comme une perte de vision sévère, la cataracte, le glaucome, Les maladies infectieuses et La rétinopathie diabétique. Il est difficile de différencier les différentes catégories de cellules rétiniennes normales et anormales, ce qui demande de l'expérience et des compétences. Ainsi, Dans le cadre de notre travsil, nous avons propose un système de classification automatique et de detection de la rétinopathie diabétique dans les images du font d'œil. Ce système peut identier cinq (5) classes différentes de font d'œil avec une haute précision. L'approche est basée sur le transfert learning en utilisant des architectures de Deep Learning.

4.2 Outils utilisés

Pour la réalisation de notre système, nous avons utilisé deux plateforme différentes :
- Google Colab intégré à Google Drive, qui permet de stocker, partager et accéder facilement aux notebooks. Dans notre cas, on a utilise pour la préparation des datasets.
- Kaggle qui donne un accès immédiat aux GPU gratuitement, sans avoir besoin de passer par des demandes d'autorisation comme sur Colab Pro et une mémoire plus grande par rapport à la RAM de colab.

4.2.1 Google Colab (Colaboratory)

Google Colab est un service interactif hébergé de Jupyter Notebook basé sur le cloud qui ne nécessite aucune configuration pour être utilisé. Elle fournit un environnement de développement complet sur python, avec de plusieures bibliothèques célèbres pré-installées telles que TensorFlow, Keras...etc. En outre, elle offre un accès gratuit aux ressources de calcul puissantes mais limiter en matiere de temps, notamment des GPU et des TPU [100] (voir La Figure 4.1).
Colab a été lancé en 2017 par Google Research, particulièrement dans le but de démocratiser l'accès aux outils d'intelligence artificielle et d'apprentissage automatique, à la science des données et à l'éducation [100].

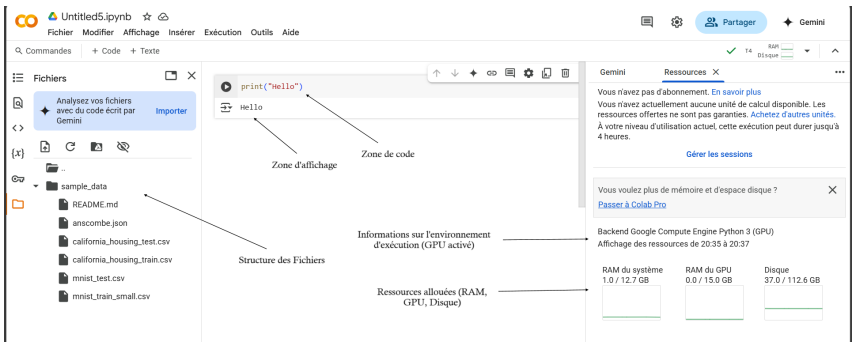


FIGURE 4.1 – Interface de Google Colab

4.2.2 Kaggle

Kaggle est une plateforme communautaire en ligne dédiée à la data science et au l'apprentissage automatique. Elle permet aux utilisateurs atteindre tous les progrès de science des données gratuits publiés par la communauté et offre un environnement de codage nommé Kaggle Notebooks (voir Figure 4.2), qui fonctionne directement dans le navigateur sans exiger d'installation. Kaggle fournit également des ressources puissantes comme des GPU et TPU gratuits pour entraîner des modèles plus rapidement. Cette plateforme est célèbre pour ses compétitions où des plus grandes entreprises de science des données du monde, telles que Walmart ou Facebook, proposent des concours réels en échange de récompenses. De plus, elle permet d'encourager le partage de connaissances entre ses membres en permettant d'écrire et de partager du code, et d'héberger des ensembles de données [101].

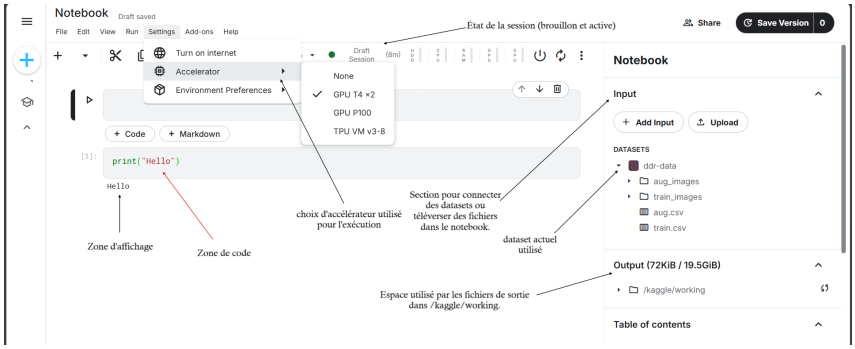


FIGURE 4.2 – Interface de kaggle

4.2.3 Langage de programmation (Python)

Python est un langage de programmation adapté à différentes utilisations grâce à des bibliothèques spécialisées, ce qui le rend utilisé dans divers contextes l'intelligence artificielle, le développement web, surtout dans le domaine de la data science et du machine learning. Python est un langage open source, il est multiplateforme, orienté objet et peut fonctionner sur un différents de systèmes d'exploitation. Il dispose d'une immense bibliothèque de modules prêts à l'emploi et d'une communauté très active, ce qui facilite l'apprentissage et l'exécution de projets complexes. [102].

4.2.4 Les bibliothèques

- TensorFlow

  TensorFlow est une bibliothèque open-source d'apprentissage automatique et de calcul numérique, développé par Google Brain et a été lancée officiellement en 2015 sous licence

Apache 2.0. Elle est utilisée pour les réseaux de neurones profonds et les tâches d'analytique prédictive. Également, cette technologie simplifie et accélère l'entraînement à les développeurs et déployer des modèles à grande échelle, y compris la classification, la reconnaissance d'images, et le traitement du langage naturel (TAL).
Le fonctionnement de TensorFlow repose sur la notion de représenter les calculs sous forme de graphes de flux de données, où les nœuds effectuent des opérations mathématiques et les données appelée tenseurs (Tensor) circulent entre ces nœuds [103].

FIGURE 4.3 – Logo de TensorFlow [104]

- Keras

  Keras est de servir d'interface de haut niveau (API) d'apprentissage profond écrite en Python conçue pour les êtres humains, elle est capable de fonctionner sur TensorFlow ou PyTorch. Cette librairie se concentre sur la rapidité de débogage, l'élégance et la concision du code, la maintenabilité et la capacité de déploiement. Son fonctionnement est de créer des modèles de deep learning en utilisant des abstractions de haut niveau, facilitant ainsi le processus de développement tout en maintenant la flexibilité nécessaire pour des tâches complexes [105].

FIGURE 4.4 – Logo de keras [106]

- Pytorch

  Pytorch est une bibliothèque logicielle d'apprentissage automatique Python open source basée sur Torch développée par Meta. PyTorch vous permet d'effectuer les calculs tensoriels nécessaires à l'apprentissage en profondeur. L'optimisation de ces calculs est réalisée par un processeur (CPU) ou une unité de traitement graphique (GPU). La création de PyTorch remonte à l'équipe de recherche de Facebook [107].

FIGURE 4.5 – Logo de pytorch [108]

- Numerical Python (Numpy)

  NumPy est une bibliothèque open source fondamentale pour le calcul scientifique en Python, qui fournit un objet tableau multidimensionnel, divers objets dérivés (notamment des tableaux masqués et des matrices), ainsi qu'un ensemble de routines pour des opérations rapides sur les tableaux, afin d'effectuer des fonctions mathématiques, logiques, statistiques et bien plus encore. NumPy est amplement utilisé dans les domaines en détresse des calculs numériques intensifs, comme la science des données, le machine learning et le traitement d'images, grâce à son aptitude à manipuler utilement de larges quantités de données[109].

FIGURE 4.6 – Logo de Numpy [110]

- Scikit-learn

  est une bibliothèque Python gratuite pour l'apprentissage automatique. Il comprend des fonctionnalités pour l'estimation de forêts aléatoires, la régression logistique et les algorithmes de classification.Les bibliothèques Python NumPy, SciPy et Matplotlib jouent un rôle clé dans l'analyse et la visualisation des données, et elles sont largement utilisées pour cela [111].

FIGURE 4.7 – Logo de Scikit-learn [111]

- Matplotlib

  est une bibliothèque permettant de tracer et de visualiser des données sous forme graphique . En plus, il propose une API orientée objet qui facilite l'intégration de graphiques dans vos applications à l'aide d'outils d'interface graphique populaires tels que Tkinter, wxPython, Qt ou GTK [112].

FIGURE 4.8 – Logo de Matplotlib [113]

4.3 Dataset utilisés

Les ensembles de données que nous allons utiliser pour réaliser notre classification sont des dataset disponible en ligne, dans kaggle [114]. Le premier dataset utilisé c'est Diabetic Retinopathy 224x224 Gaussian Filtered [115] (voir Figure 4.9). Ces images sont constituées d'images de la rétinien filtrées gaussiennes pour détecter la rétinopathie diabétique, et sont redimensionnées en 224x224 pixels, cet ensemble de données est composée de 3662 images, issu de la compétition APTOS 2019 Blindness Detection. Les images sont classées en 5 classes selon la sévérité de la RD : No_DR (saine) de 1805 images, Mild (légère) de 370 images, modérée de 999 images, sévère de 193 images et prolifération_DR de 295 images.

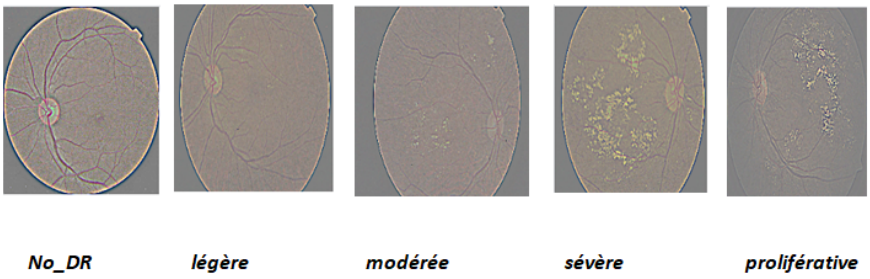


FIGURE 4.9 – Visualisation de différente classe de dataset Aptos 2019 Gaussian Filtered [115]

Et pour le deuxième dataset connu sous le nom de DDR (Dataset for Diabetic Retinopathy) [116], provenant de 147 hôpitaux en Chine. Il est composé de 13673 images du fond d'œil pour la classification des stades de la RD, qui ont été acquises à l'aide de Biomicroscope (lamp à fente). L'ensemble de données est organisé dans les cinq (5) classes suivantes selon la sévérité de la RD (voir Figure 4.10) : aucune (No_DR) de 6266, Mild (légère) de 630, modérée de 4477 images, sévère de 236 images et prolifération de 913 images. Il existe une sixième catégorie qui indique les images de mauvaise qualité, mais nous n'avons pas pris. Nous avons utilisé data qui contient 12522 images .

FIGURE 4.10 – Visualisation de différente classe de dataset DDR [116]

4.4 Méthodes proposées

Dans le cadre de notre projet, nous avons élaboré et comparé plusieurs architectures de classification à base de réseaux neuronaux profonds afin de diagnostiquer la rétinopathie diabétique à partir d'images du fond d'œil en utilisant deux stratégies (Directe et Cascade), la première consiste a utiliser les modeles directement sur les images, et la deuxième en cascade consiste a classifier les images par étape. Nous avons principalement utilisé les architectures DenseNet121, ViT-B/16 et YOLOv8, Cela constitue un moyen efficace de réutiliser des modèles pré-entraînés sur de grandes bases de données telles qu'ImageNet [Deng et al., 2009 [117]] en appliquant des méthodes d'apprentissage par transfert. L'avantage de cette stratégie est qu'elle permet de réduire le temps de formation tout en améliorant les performances, notamment pour des domaines complexes comme la vision médicale. Voici les principales caractéristiques de notre approche par rapport à la configuration des architectures et modèles :

• Tous les modèles utilisés sont pré-entraînés sur la base de données ImageNet, qui se compose de millions d'images réparties dans 1 000 catégories, ce qui leur donne une capacité initiale à extraire des caractéristiques visuelles générales.

• Afin d'adapter les modèles (DenseNet121, ViT-B/16, YOLOv8) à notre tâche de classification en cinq classes (No_DR, Mild, Moderate, Severe, Proliferate_DR), nous avons retiré les couches supérieures des architectures (include_top=False) pour ajouter un classifieur personnalisé (voir Figure 4.11). Ainsi, après la partie Features extraction, nous avons appliqué une couche GlobalAveragePooling2D, suivie d'une ou plusieurs couches Dense et d'une couche Dropout pour éviter le surapprentissage ce qui représente la partie classification de chaque modèle.

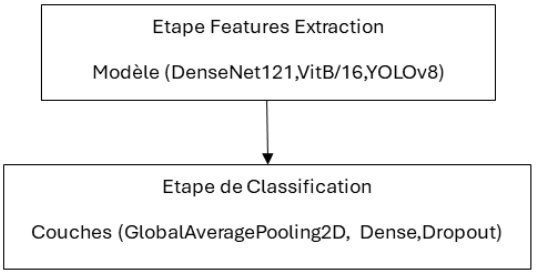


FIGURE 4.11 – Schéma représente les étapes de l'extraction et la classification

• L'apprentissage est effectué après équilibrage des données, en augmentant les images (rotation, mise en miroir, mise à l'échelle, éclairage, etc.) ou en suréchantillonnant selon les classes disponibles dans dataset.

• Les images ont été redimensionnées à une taille standard (224×224 pixels) pour assurer une compatibilité avec les modèles pré-entraînés.

• Les expériences ont été menées sur deux bases de données :

- APTOS 2019 filtrée par un filtre gaussien.

- DDR, utilisée directement avec un équilibrage par augmentation.

4.4.1 Modèle 1 : Architecture DenseNet121

Dans le premier modèle nous avons exploité l'architecture DenseNet121 pré-entraîné (voir chapitre 3) en deux data différentes :

a. Application sur Dataset-1 (Aptos 2019)

a.1 - Approche directe

Dans cette approche, Comme structure de la partie classification du modèle (voir Figure 4.12), nous avons dégelé les 30 dernières couches pour un fine-tuning, et une couche de GlobalAveragePooling2D afin de diminuer les dimensions des cartes de caractéristiques générées et le nombre de paramètres avant la couche de classification, après nous avons ajouté une couche dense de 256 neurones avec une fonction d'activation ReLU. Ensuite, une couche de BatchNormalization pour de stabiliser l'apprentissage puis une couche de Dropout de 50% a été intégrée pour éviter le sur-apprentissage (overfitting). A la fin une seule couche dense ayant 5 neurones avec fonction d'activation "Softmax" afin d'obtenir les probabilités d'appartenance pour chaque classe.

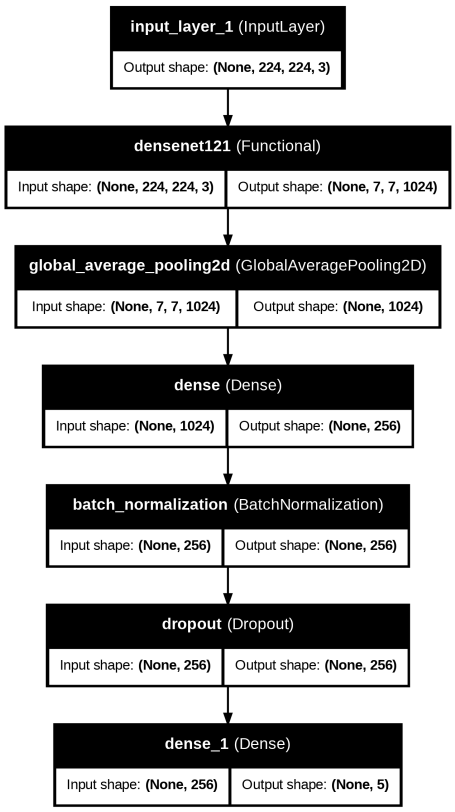


FIGURE 4.12 – Architecture de l'approche directe du modèle 1 (dataset 1)

• Chargement des données : Dans cette étape, nous avons chargé et préparé les images du dataset APTOS 2019 prétraitées avec un filtre gaussien en utilisant un fichier CSV contenant les informations sur les images, telles que les identifiants des images et les étiquettes de diagnostic, est chargé à l'aide de la bibliothèque pandas (voir Figure 4.13). Ils sont de taille 224x224 pixels, afin d'éviter les problèmes de mémoire et ce qui est compatible avec les exigences du modèle DenseNet121.

```
# ajouté une colonne pour classification du 5 classe
df = pd.read_csv(r'../input/diabetic-retinopathy-224x224-gaussian-filtered/train.csv')

diagnosis_5class_dict = {
    0: 'No_DR',
    1: 'Mild',
    2: 'Moderate',
    3: 'Severe',
    4: 'Proliferate_DR',
}
diagnosis_dict = {
    0: 'No_DR',
    1: 'Mild',
    2: 'Moderate',
    3: 'Severe',
    4: 'Proliferate_DR',
}
# Ajout de la colonne texte pour les 3 classes
df['type_5class'] = df['diagnosis'].map(diagnosis_5class_dict.get)

df['type'] = df['diagnosis'].map(diagnosis_dict.get)
print(df['type_5class'].value_counts())
df.head()
```

```
type_5class
No_DR              1805
Moderate            999
Mild                370
Proliferate_DR      295
Severe              193
Name: count, dtype: int64
```

	id_code	diagnosis	type_5class	type
0	000c1434d8d7	2	Moderate	Moderate
1	001639a390f0	4	Proliferate_DR	Proliferate_DR
2	0024cdab0c1e	1	Mild	Mild
3	002c21358ce6	0	No_DR	No_DR
4	005b95c28852	0	No_DR	No_DR

FIGURE 4.13 – Chargé les images apartir de fichier CSV

• Data augmentation : Après le chargement du dataset, nous avons constaté un déséquilibre dans la distribution du nombre d'images par classe, comme illustré dans la Figure 4.14, ce qui provoque des problèmes de sur-apprentissage.

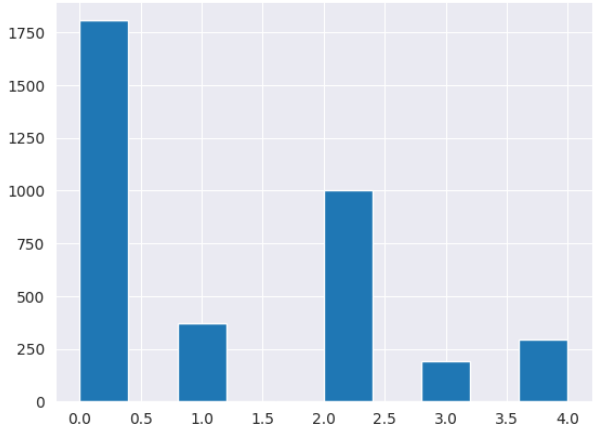


FIGURE 4.14 – Visualisation la distribution des classes de dataset Aptos 2019

Pour résoudre ce problème, nous avons utilisé le principe de oversampling qui consiste à dupliquer des images des classes minoritaires (Figure 4.15). L'objectif est d'équilibré les classes du dataset sans introduire de nouvelles images artificielles par transformation. Ce choix nous a également permis de conserver le même type de prétraitement utilisé dans l'approche suivante (en cascade), afin de pouvoir comparer équitablement les deux approches. Une augmentation légére pour rendre le modèle plus robuste et généralisable.

```
#dupliqué les images pour les classes minoritaires afin de equilibré dataset
df_class1 = df[df['type_5class'] == 'Mild']
df_class2 = df[df['type_5class'] == 'Moderate']
df_class3 = df[df['type_5class'] == 'Severe']
df_class4 = df[df['type_5class'] == 'Proliferate_DR']
# Oversample les deux classes minoritaires
df5_oversampled = pd.concat([
    df,
    df_class1.sample(1435, replace=True),    #necessite 1435 → 1805
    df_class2.sample(806, replace=True),    # necessite 806 → 1805
    df_class3.sample(1612, replace=True),    #necessite 1612 → 1805
    df_class4.sample(1510, replace=True)    #necessite 1510 → 1805
])

# Mélanger le tout
df5_oversampled = df5_oversampled.sample(frac=1, random_state=42).reset_index(drop=True)
```

FIGURE 4.15 – Oversampling utilisé

• Répartition des données : le dataset a été réparti en trois parties, 15% des données ont été réservées pour la validation et les 85% restants ont été redivisés en 70% pour l'entraînement et 15% pour le test comme le montre la Figure 4.16.

```
from sklearn.model_selection import train_test_split
# division : 85% pour train+test, 15% pour validation
train_val_df, val_5 = train_test_split(
    df5_oversampled, test_size=0.15, stratify=df5_oversampled['type_5class'], random_state=42
)

# division de train+test en 70% train / 15% test
train_5, test_5 = train_test_split(
    train_val_df, test_size=0.15 / (1 - 0.15), stratify=train_val_df['type_5class'], random_state=42
)
print("Train:", train_5['type_5class'].value_counts(),'\n')
print("Val:", val_5['type_5class'].value_counts(),'\n')
print("Test:", test_5['type_5class'].value_counts(),'\n')
```

FIGURE 4.16 – Division des données en sous-ensembles train/val/test

• Normalisation des données : Les images sont normalisées en divisant les valeurs des pixels par 255 pour amener les valeurs de pixels dans la plage [0, 1]. Cette opération est effectuée dans ImageDataGenerator avec le paramètre rescale=1./255 (voir Figure 4.17).

```
# Val
val_batches = ImageDataGenerator(rescale=1./255)
```

FIGURE 4.17 – La normalisation utilisé pour les images

• One hot encoding : pour toute les données, les étiquettes de diagnostic ont été converties en format catégorique (one-hot encoding) (voir FIgure 4.18). Cela permet de transformer les étiquettes textuelles correspondant aux classes ('No_DR', 'Mild', etc.) en vecteurs one-hot encoded, ce qui est nécessaire pour l'entraînement d'un modèle de classification multi-classe.

id_code	No_DR	Mild	Moderate	Severe	Proliferative_DR
000c1434d8d7	0	0	1	0	0
001639a390f0	0	0	0	0	1
0024cdab0c1e	0	1	0	0	0
002c21358ce6	1	0	0	0	0
005b95c28852	1	0	0	0	0

FIGURE 4.18 – Application de one-hot encoding sur les labels du dataset

a.2 - Approche en cascade

Dans cette seconde approche, nous avons utilisé une stratégie en cascade, dans l'objectif d'affiner et simplifier la tâche de classification progressivement via la division de problème en nombreux étapes hiérarchiques, premièrement une classification binaire, ensuite une classification à 3 classes, et enfin une classification finale à 5 classes.

• Tout d'abord, nous avons construit un modèle pré-entraîné sur ImageNet pour distinguer entre deux groupes de classes, les images saines (NO_DR) et les images pathologiques (Mild, Moderate, Severe, Proliferate_DR). Pour cela, Le modèle DenseNet121 a été utilisé comme

extracteur de caractéristiques sans sa partie supérieure, ensuite nous avons ajouté une couche GlobalAveragePooling2D, une couche Dropout à 40% pour éviter le sur-apprentissage, et une couche de sortie contient 2 neurones avec une fonction d'activation Sigmoid adapté aux deux classes (voir Figure 4.19). Cette première étape permet d'enlever les cas non pathologiques du reste.

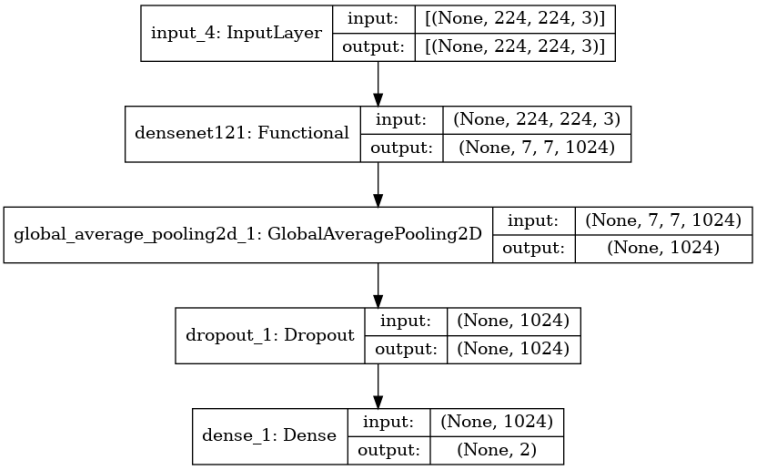


FIGURE 4.19 – architecture de l'approche en cascade (étape 1 : binaire) du modèle 1

• Ensuite, le modèle binaire entrainé lors de l'étape précédente a été utilisé comme base pour une classification plus fine pour un second modèle à 3 classes. Pour cela, toutes les couches ont été gelées, sauf les 50 dernières, afin de permettre un fine-tuning léger. Ensuite, nous avons ajouté une couche dense de 128 neurones avec une fonction d'activation ReLU, suivie d'un Dropout à 50%, puis d'une couche de sortie de 3 neurones avec une fonction d'activation Softmax adapté à la classification multiclasse (voir Figure 4.20). Cette étape vise à distinguer des niveaux de gravité intermédiaires.

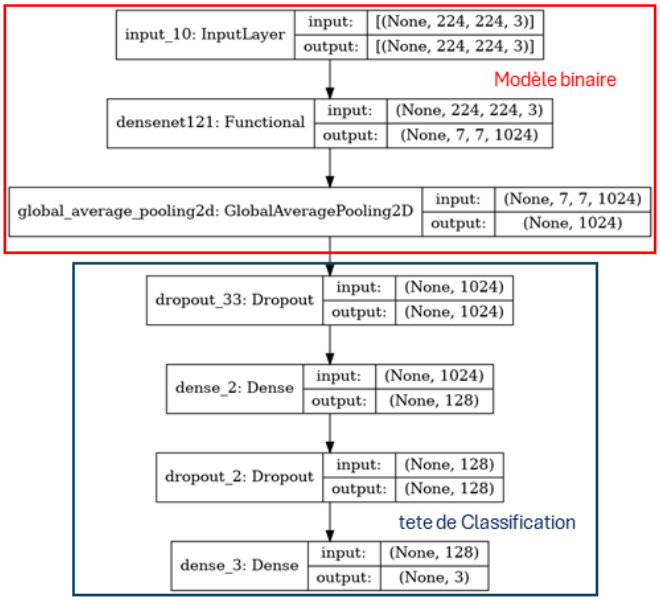


FIGURE 4.20 – architecture de l'approche en cascade (étape 2 : 3 classes) du modèle 1

• Enfin, le modèle précèdent a servi de base pour effectuer un modèle final afin de classer les 5 classes correspondant aux différents stades de la rétinopathie diabétique. Nous avons dégelé les 30 dernières couches pour un fine-tuning, et en a ajouté une couche dense de 256 neurones

avec une fonction d'activation ReLU, une couche de BatchNormalization pour de stabiliser l'apprentissage puis d'un Dropout de 50% et d'une couche de sortie softmax avec 5 neurones (voir Figure 4.21).

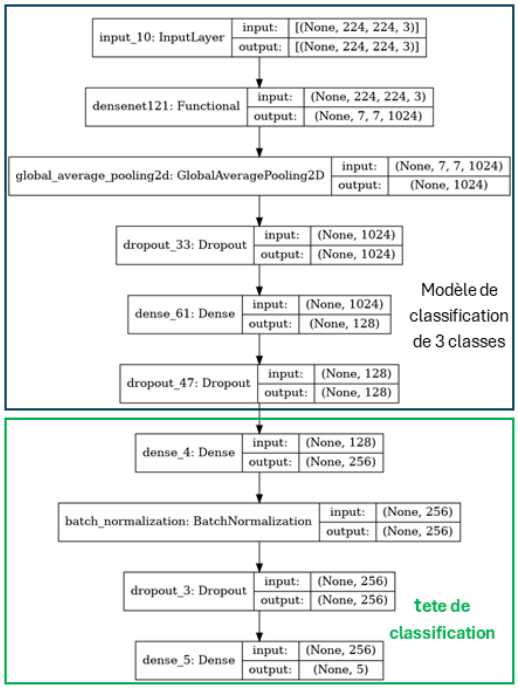


FIGURE 4.21 – architecture de l'approche en cascade (étape 3 : 5 classes) du modèle 1

• Chargement des données : Dans cette étape, nous avons chargé et préparé les images de la même manière de l'approche précédante en utilisant un fichier CSV contenant les informations sur les images, telles que les identifiants des images et les étiquettes de diagnostic, est chargé à l'aide de la bibliothèque pandas (voir Figure 4.13). Les étiquettes des classes ont été converties en deux formats différents selon le cas de classification (binaire, 3 classes, 5 classes) comme illustré dans la Figure 4.22.

```
#ajouter une colonne pour diagnostic binaire DR, No_DR
df = pd.read_csv(r'../input/diabetic-retinopathy-224x224-gaussian-filtered/train.csv')

diagnosis_dict_binary = {
    0: 'No_DR',
    1: 'DR',
    2: 'DR',
    3: 'DR',
    4: 'DR'
}

diagnosis_dict = {
    0: 'No_DR',
    1: 'Mild',
    2: 'Moderate',
    3: 'Severe',
    4: 'Proliferate_DR',
}


df['binary_type'] =  df['diagnosis'].map(diagnosis_dict_binary.get)
df['type'] = df['diagnosis'].map(diagnosis_dict.get)
df.head()
```

```
#ajouter une colonne pour diagnostic 3 classe DR, DR_No_Proliferative, DR_Proliferative
df = pd.read_csv(r'../input/diabetic-retinopathy-224x224-gaussian-filtered/train.csv')

diagnosis_3class_dict = {
    0: 'No_DR',
    1: 'DR_No_Proliferative',
    2: 'DR_No_Proliferative',
    3: 'DR_No_Proliferative',
    4: 'DR_Proliferative'
}

diagnosis_dict = {
    0: 'No_DR',
    1: 'Mild',
    2: 'Moderate',
    3: 'Severe',
    4: 'Proliferate_DR',
}
# Ajout de la colonne type_3class pour les 3 classes
df['type_3class'] = df['diagnosis'].map(diagnosis_3class_dict.get)

df['type'] = df['diagnosis'].map(diagnosis_dict.get)
print(df['type_3class'].value_counts())
df.head()
```

FIGURE 4.22 – chargement des données pour la classification de deux et trois et cinq classes

• Data augmentation : après le chargement du dataset, nous avons constaté un déséquilibre dans la distribution du nombre d'images par classe dans le cas de classification 3 classes et 5 classes, comme illustré dans la Figure 4.23, ce qui provoque des problèmes d'apprentissage tels que le sur-apprentissage, où ces classes seront mieux classées par rapport aux restes des classes.

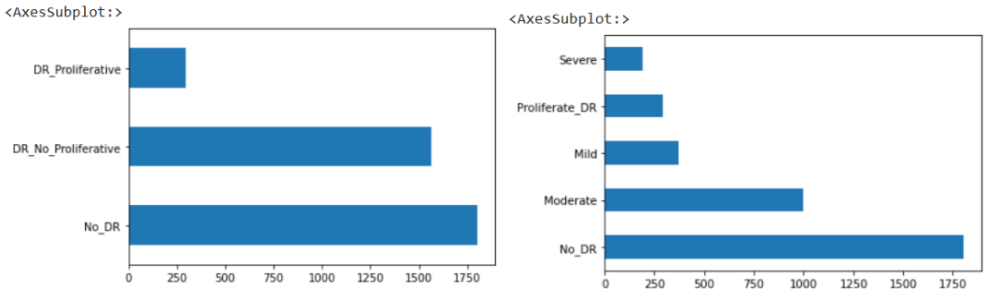


FIGURE 4.23 – Distribution des classe dans le cas de classification trois et cinq classes

Pour résoudre ce problème, un oversampling a été effectué qui consiste à dupliquer aléatoirement des exemples des classes minoritaires (voir Figure 4.24) pour garantir une répartition égale des classes dans les ensembles de formation sans créer de nouvelles images non vues par les modèles déja entraîné (binaire et 3 classe), ce qui pourrait compromettre la cohérence entre les étapes. Une augmentation pour eviter le sur-apprentissage.

```python
# Oversample (dupliquer) Afin d'obtenir equilibrage des 3 classe
# 1. séléctionner les classe minoritaire
df_class1 = df[df['type_3class'] == 'DR_No_Proliferative']
df_class2 = df[df['type_3class'] == 'DR_Proliferative']
# 2. dupliquer et sauvgardé dans Csv df_oversampled
df_oversampled = pd.concat([
    df,
    df_class1.sample(243, replace=True),    # DR_No_Proliferative necessite 243 → 1805
    df_class2.sample(1510, replace=True)    # DR_Proliferative necessite 1510 → 1805
])

# Mélanger le tout
df_oversampled = df_oversampled.sample(frac=1, random_state=42).reset_index(drop=True)
```

FIGURE 4.24 – Application de oversampling sur les classes minoritaires

- Répartition des données : Nous avons utilisé la même division d'approche précédante (voir Figure 4.16).
- Normalisation des données : Nous avons utilisé la même normalisation d'approche précédante (voir Figure 4.17).
- One hot encoding : Pour toute les données, les étiquettes de diagnostic ont été converties en format catégorique (one-hot encoding) (voir Figure 4.18).

b. Application sur Dataset-2 (DDR)

Nous avons appliqué une approche directe sur cette base de donnée, exploiter l'architecture DenseNet121 pré-entraîné (voir chapitre 3).
Dans cette approche, Comme structure de la partie classification du modèle (voir Figure 4.12), nous avons ajouté une couche de GlobalAveragePooling2D afin de diminuer les dimensions des cartes de caractéristiques générées et le nombre de paramètres avant la couche de classification. Ensuite, une couche de Dropout de 40% a été intégrée pour éviter le sur-apprentissage (overfitting), en désactivant aléatoirement 40% des neurones pendant l'entraînement, ce qui rendre le modèle généralise bien. A la fin une seule couche dense ayant 5 neurones avec fonction d'activation "Softmax" afin d'obtenir les probabilités d'appartenance pour chaque classe.

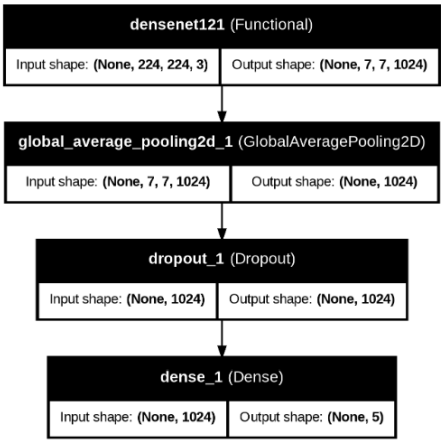


FIGURE 4.25 – Architecture de l'approche directe du modèle 1 (dataset 2)

• Les images du dataset DDR ont été soumises à plusieurs étapes de prétraitement (voir Figure 4.27) pour assurer une uniformité visuelle et dimensionnelle avant d'être utilisées pour l'entraînement du modèle. Tout d'abord, les bordures noires entourant certaines images ont été détectées et supprimées à l'aide d'une méthode de détection automatique basée sur le seuil de pixel afin de conserver que la région informative du fond d'œil. Puis, les images ont été recadrées pour recentrer ces zones inutiles. Afin de garantir une concordance avec les architectures de réseaux de neurones convolutionnels (CNN), toutes les images ont été mises au format carré en ajoutant un remplissage (padding) symétrique avec des pixels noirs, de sorte à obtenir des dimensions uniformes de 224 × 224 pixels (voir Figure 4.26). Ce même prétraitement a été également appliqué aux autres modèles (ViT-B/16 et YOLOv8).

```python
#  Fonctions de prétraitement
"""Supprime les bords noirs en détectant les pixels sombres en niveaux de gris"""
def crop_black_borders(im, threshold=15):
    im_array = np.array(im)
    gray = np.mean(im_array, axis=-1)
    coords = np.column_stack(np.where(gray > threshold))
    if coords.shape[0] > 0:
        x_min, y_min = coords.min(axis=0)
        x_max, y_max = coords.max(axis=0)
        im_array = im_array[x_min:x_max, y_min:y_max]
    return Image.fromarray(im_array)

 """Ajoute du padding noir pour centrer et obtenir une image carrée"""
def add_padding(im, desired_size=224):
    old_size = im.size
    ratio = float(desired_size) / max(old_size)
    new_size = tuple([int(x * ratio) for x in old_size])
    im = im.resize(new_size, Image.LANCZOS)

    new_im = Image.new("RGB", (desired_size, desired_size), (0, 0, 0))
    paste_position = ((desired_size - new_size[0]) // 2,
                      (desired_size - new_size[1]) // 2)
    new_im.paste(im, paste_position)
    return new_im
```

FIGURE 4.26 – Prétraitement utilisé sur dataset DDR

FIGURE 4.27 – Visualisation des images avant et après le prétraitement

Enfin, les chemins des images résultantes ont été stockés dans un fichier CSV, avec leurs étiquettes de de diagnostic correspondantes, prêtes à être préparées comme des données d'entrée pour l'entraînement.

• Chargement des données : Les données ont été chargées à partir du fichier CSV, contenant pour chaque échantillon le chemin de chaque image prétraitée et son étiquette de diagnostic. À l'aide de la bibliothèque Pandas, ces données ont été séparées entre les variables d'entrée (x), constituées des chemins vers les images et les étiquettes (y), correspondant aux niveaux de gravité de la rétinopathie diabétique (voir Figure 4.28).

```
# Charger les données
df = pd.read_csv("/kaggle/input/csv-data/data.csv")

# Séparer les features et labels
x = df[['id_code', 'image_path']]
y = df['diagnosis']
```

FIGURE 4.28 – chargement des données du dataset DDR

Nous avons utilisé les générateurs d'images pour ne charger pas manuellement en mémoire mais lues dynamiquement via la méthode flow_from_dataframe de Keras. Cette approche permet un chargement en mémoire efficace et progressif des images, ce qui est particulièrement adapté avec le grand nombre des images de nos dataset.

• Augmentation des données : Après le chargement du dataset, nous avons visualisé un grand déséquilibre dans la distribution du nombre d'images par classe, comme illustré dans la Figure 4.29. En effet, les classes No_DR , modérée contiennent un nombre elevé d'images par rapport aux autres classes, ce qui provoque des problèmes de sur-apprentissage.

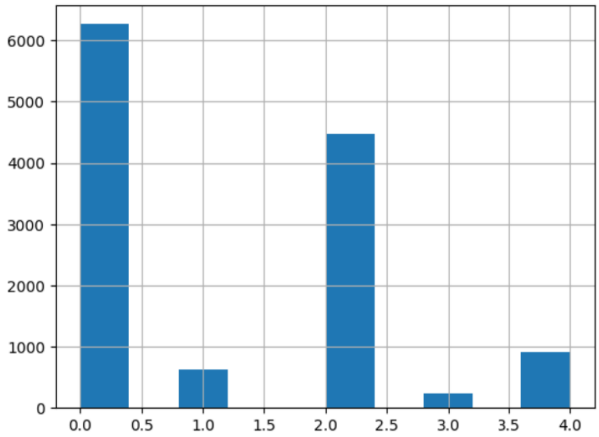


FIGURE 4.29 – visualisation de la distribution des images des classes origines

Afin de remédier au déséquilibre des classes dans dataset, nous avons aplliqué une stratégie d'augmentation à l'ensemble des classes. La classe 0, qui contenait 6266 images, a été légèrement augmentée pour d'atteindre un nombre total de 6629 images, et les autres classes minoritaires (0 à 4) ont été soumises à une augmentation plus importante pour égaler ce même nombre d'images. L'objectif de cette démarche est rendre le modèle plus robuste en évitant les biais d'apprentissage liés à la surreprésentation d'une classe (Figure 4.30).

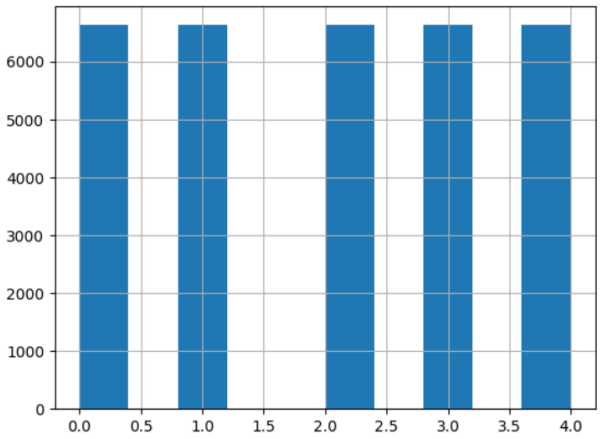


FIGURE 4.30 – visualisation de la distribution des images après l'augmentation pour chaque classe

Par ailleurs, une visualisation des images augmentées a été realisée pour s'assurer de la pertinence et de la diversité des transformations appliquées (Figure 4.31).

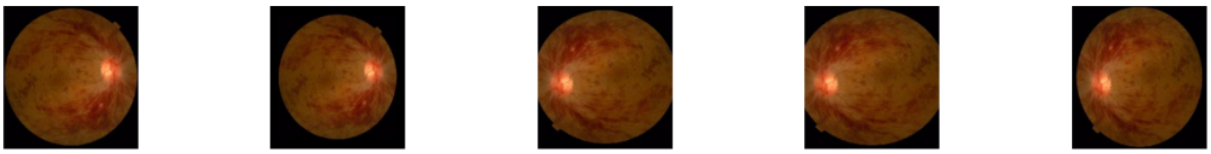


FIGURE 4.31 – visualisation des images après l'augmentation

• Répartition des données : La division du dataset en 3 sous-ensembles, train, validation et le test en utilisant la fonction train_test_split de la bibliothèque Scikit-learn. D'abord, 10% des données ont été gardées pour le test, et les 90% restants ont été utilisés pour le train/validation. Ensuite, les 90% ont été divisés en 80% pour le train et 20% pour la validation. En fin de compte, la répartition finale se situe à 72% des données pour le train, 18% pour la validation et 10% pour le test (voir Figure 4.32).

```
# Split 1 : Train + Test (90% / 10%)
x_train_full, x_test, y_train_full, y_test = train_test_split(
    x, y,
    test_size=0.1,
    stratify=y,
    random_state=2019
)
# Split 2 : Train partial + Validation (80% / 20% de train_full)
x_train_partial, x_val, y_train_partial, y_val = train_test_split(
    x_train_full, y_train_full,
    test_size=0.2,
    stratify=y_train_full,
    random_state=2019
)
```

FIGURE 4.32 – La division des données en sous-ensembles train/val/test

• Normalisation des images : Les images ont été normalisées à l'aide de la fonction preprocess_input donnée par Keras pour ce modèle (voir Figure 4.33). Cette fonction convertit les pixels RGB dans un format normalisé spécifique au modèle DenseNet121 pré-entraîné sur ImageNet, centré autour de zéro, ce qui facilite la convergence du modèle pendant l'entraînement.

```
val_datagen = ImageDataGenerator(preprocessing_function=preprocess_input)
test_datagen = ImageDataGenerator(preprocessing_function=preprocess_input)
```

FIGURE 4.33 – La normalisation utilisé pour les données

• One hot encoding : pour toute les données, les étiquettes de diagnostic ont été converties en format catégorique (one-hot encoding) (voir Figure 4.18)

4.4.2 Modèle 2 : Architecture Vision Transformer (ViT-B/16)

En supplément des modèles basées sur les réseaux convolutifs profonds, nous avons pareillement expérimenté une architecture récente et efficace basée sur les Transformers visuels (voir chapitre 3), en utilisant le modèle ViT-B/16, dans ce modèle on a utilisé une approche directe de classification parce qu'une approche en cascade sera très complexe a exécuté avec ce type de modèle.
Le ViT-B/16 prend en entrée des images de taille 224×224 divisées en patchs de 16×16, ce qui donne une séquence de 196 tokens en entrée du transformeur. Le modèle a été initialisé sans les poids préentraînés, puis ceux-ci ont été importés manuellement depuis un fichier .pth préentraîné sur ImageNet. Cette étape a été effectuée sur le CPU avant que le modèle ne soit transféré vers le GPU pour l'entraînement, afin d'éviter toute surcharge mémoire sur elle.
La tête de classification origine du modèle a été remplacée par une nouvelle structure adaptée à notre tâche à 5 classes composée d'une couche Dropout avec un taux de 40% permettant d'améliorer la régularisation, suivie d'une couche dense avec 5 neurones en sortie (voir Figure 4.34).

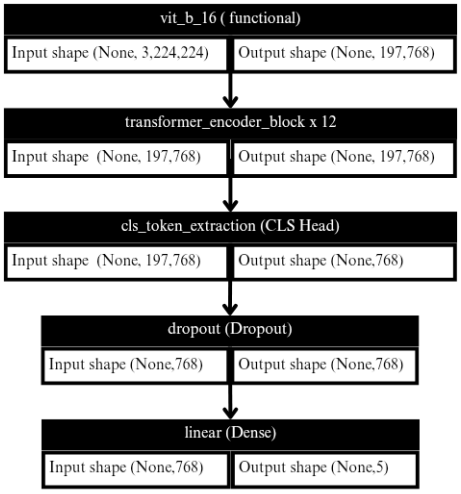


FIGURE 4.34 – Architecture de modèle 2

A. - Application sur Dataset-1 (Aptos 2019)

• Chargement des données : Dans cette étape, nous avons chargé et préparé les images du dataset APTOS 2019 prétraitées avec un filtre gaussien en utilisant un fichier CSV contenant les informations sur les images, telles que les identifiants des images et les étiquettes de diagnostic, est chargé à l'aide de la bibliothèque pandas. Nous avons chargé les données en utilisant une classe personnalisée CustomImageDataset (voir Figure 4.35), dérivée de torch.utils.data.Dataset. Cette classe permet de lire les chemins des images depuis le DataFrame issu du CSV, ouvre chaque image en format RGB, puis applique la conversion en tenseur et la normalisation. Cela

permet un chargement dynamique et efficace des données pendant l'entraînement, sans surcharge mémoire. Les images et leurs étiquettes sont renvoyées sous forme de paires (image, label).

```python
# Dataset personnalisé pour charger des images organisées par classe dans des sous-dossiers
class CustomImageDataset(Dataset):
    def __init__(self, dataframe, image_dir, transform=None):
        self.dataframe = dataframe.reset_index(drop=True)  # Réinitialise les index du DataFrame
        self.image_dir = image_dir  # Dossier contenant les images
        self.transform = transform  # Transformations à appliquer (ex: resize, normalisation,..)

        # Conversion des labels numériques en noms de dossier
        self.reverse_label_map = {
            0: 'No_DR', 1: 'Mild', 2: 'Moderate', 3: 'Severe', 4: 'Proliferate_DR'
        }

    def __len__(self):
        return len(self.dataframe)  # Nombre total d'éléments

    def __getitem__(self, idx):
        row = self.dataframe.iloc[idx]  # Récupère une ligne
        label = row['diagnosis']  # Label (entre 0 et 4)
        image_id = row['id_code'] + '.png'  # Nom du fichier image
        class_name = self.reverse_label_map[label]  # Nom du dossier de classe
        image_path = os.path.join(self.image_dir, class_name, image_id)  # Chemin complet de l'image
        image = Image.open(image_path).convert("RGB")  # Ouvre l'image en RGB
        if self.transform:
            image = self.transform(image)  # Applique les transformations si définies

        return image, label  # Retourne l'image et son label
```

FIGURE 4.35 – Chargement des images apartir de classe personnalisée

• Les images sont redimensionnées à 224×224 pixels, ce qui est compatible avec les exigences du modèle ViT-B/16.

• Le dataset d'origine présente un déséquilibre important entre les classes, avec une surreprésentation notable de la classe 0 (voir Figure 4.14). Afin de remédier à ce problème, nous avons utilisé la stratégie d'augmentation ciblant les classes minoritaires (1 à 4). dans l'objectif d'atteindre le nombre d'images dans chacune de ces classes au le nombre des images de la classe 0 comme illustré dans la figure 4.36. Une visualisation des images augmentées a été effectuée pour assurer la qualité et la diversité des transformations appliqués (voir Figure 4.37).

```python
# Augmentations douces adaptées à la rétinopathie
datagen = ImageDataGenerator(
    rotation_range=15,
    zoom_range=0.1,
    horizontal_flip=True,
    vertical_flip=True,
    brightness_range=(0.9, 1.1),
    fill_mode='nearest'
)
```

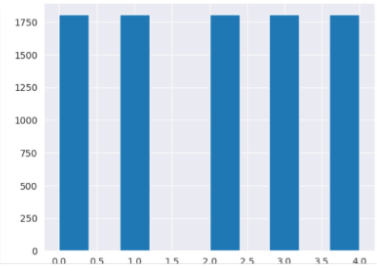


FIGURE 4.36 – Augmentation utilisé pour éséquilibré dataset aptos 2019

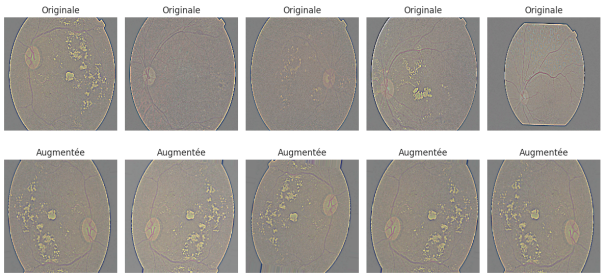


FIGURE 4.37 – Visualisation des images origine et augmenté

• Répartition des données : Le dataset équilibré a été divisé 10% des données ont été réservées pour le test et les 90% restants ont été redivisés en 75% pour l'entraînement et 15% pour la validation (voir Figure 4.38).

```
# division : 10% test, puis ce qui reste pour train/validation
x_train_full, x_test, y_train_full, y_test = train_test_split(
    df, df['diagnosis'], test_size=0.1, random_state=2019, stratify=df['diagnosis'])
# 90% restante : 15% validation et 75% train
x_train_partial, x_val, y_train_partial, y_val = train_test_split(
    x_train_full, y_train_full, test_size=0.15, random_state=2019, stratify=y_train_full)
```

FIGURE 4.38 – La répartition des données en sous-ensembles train/val/test

• Normalisation des données : Les images ont été transformées en tenseurs PyTorch et normalisées selon les statistiques du dataset ImageNet (voir Figure 4.39), elle a été appliquée à partir des transformations de torchvision.transforms. Cela permet de rendre les images compatibles avec le modèle ViT-B/16 pré-entraîné sur ImageNet, améliorant ainsi la convergence et la stabilité de l'apprentissage. Pour toute les données, les étiquettes ont été conservées au format entier (long) sans recours au one-hot encoding.

```
val_test_transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406],
                         std=[0.229, 0.224, 0.225])
])
```

FIGURE 4.39 – La normalisation sur les sous-ensembles train/val/test

B. - Application sur Dataset-2 (DDR)

• Les données sont organisées sous forme d'un fichier CSV contenant pour chaque échantillon l'identifiant de l'image prétraité et son étiquette de diagnostic. Ces images ont ensuite été stockées dans des sous-dossiers nommés selon leur classe (0, 1, ..., 4), accompagnées d'un fichier CSV listant les identifiants des images et leurs étiquettes (diagnosis). Nous avons chargé les données en utilisant une classe personnalisée CleanedImagesDataset(voir Figure 4.40), dérivée de torch.utils.data.Dataset. Cette classe permet de lire les chemins des images depuis le DataFrame issu du CSV, ouvre chaque image en format RGB, puis applique la conversion en tenseur et la normalisation. Cela permet un chargement dynamique et efficace des données pendant l'entraînement, sans surcharge mémoire. Les images et leurs étiquettes sont renvoyées sous forme de paires (image, label).

```
"""charger dynamiquement les images et leurs étiquettes depuis un DataFrame et un répertoire d'images"""

class CleanedImagesDataset(Dataset):
    def __init__(self, dataframe, image_dir, transform=None):
        self.dataframe = dataframe.reset_index(drop=True)
        self.image_dir = image_dir
        self.transform = transform

    def __len__(self):
        return len(self.dataframe)

    def __getitem__(self, idx):
        row = self.dataframe.iloc[idx]
        image_path = os.path.join(self.image_dir, str(row['diagnosis']), row['id_code'])
        label = row['diagnosis']
        image = Image.open(image_path).convert("RGB")

        if self.transform:
            image = self.transform(image)

        return image, label
```

FIGURE 4.40 – Chargement des images apartir de classe personnalisée

• Augmentation des données : Le jeu de donnée est désiquilibré (voir Figure 4.29), nous avons résoudre ce problème de la même stratégie d'augmentation appliquée dans le modèle

précédente(voir les figures 4.30, 4.31).

• Répartition des données : Nous avons appliqué la meme division de dataset 1 de ce modèle comme illustré dans la figure 4.38.

• Normalisation des images : Nous avons converti les images en tenseurs, ce qui a diminué les valeurs de pixels entre 0 et 1. Ensuite, une normalisation canal par canal a été appliquée avec les moyennes et écarts-types standards d'ImageNet (voir Figure 4.39).

4.4.3 Modèle 3 : Architecture YOLOv8

Nous avons expérimenté une architecture récente et efficace c'est YOLOv8n-cls, une version légère de YOLOv8 dédiée à la classification, Cette version "nano" a été choisie pour limiter le sur-apprentissage. L'approche de modèle 3 est directe sur les 5 classes de notre dataset, utilisation d'une approche en cascade cause des difficultés de clacules et mémoire. L' architecture YOLOv8n-cls comprend un backbone CSPDarknet pour l'extraction des caractéristiques, suivi d'une couche GlobalAveragePooling pour réduire les dimensions et limiter les paramètres. Une couche de Dropout est intégrée pour réduire le risque d'overfitting. Enfin, une couche Dense avec 5 neurones et une activation Softmax permet de prédire la probabilité d'appartenance à chaque classe (voir Figure 4.41).

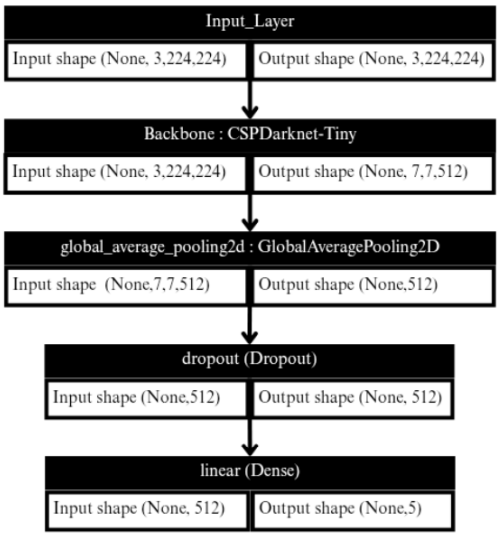


FIGURE 4.41 – Architecture de modèle 3

A. - Application sur Dataset-1 (Aptos 2019) et Dataset-2 (DDR)

• Chargement des données : Les données ont été organisées dans un dossier contenant cinq sous-dossiers, chacun correspondant à la gravité de la rétinopathie diabétique (allant de 0 à 4) (voir Figure 4.42).

```
# chemin de dataset contient 4 dossier (0 à 4)
SOURCE_DIR = '/kaggle/input/aptoscopmlet'

# mapper les labels de classe
class_map = {
    'No_DR': 0,
    'Mild': 1,
    'Moderate': 2,
    'Severe': 3,
    'Proliferate_DR': 4
}
```

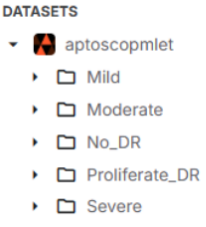


FIGURE 4.42 – Organisation et chargement de dataset

• Data augmentation : Nous avons applique les mêmes techniques d'augmentation de données que celles utilisées dans le modèle précédent pour chaque dataset.

• Répartition des données : Pour préparer les données, nous avons utilisé un script Python pour diviser automatiquement les images en trois sous-ensembles différents : 70% des images pour l'apprentissage (train), 15% pour la validation (val) et 15% pour le test (test) à l'aide de la fonction train_test_split de la bibliothèque Scikit-learn comme illustré dans la figure 4.43. Cette structure est directement compatible avec la méthode d'entraînement à la classification d'images proposée par le framework YOLOv8.

```python
# Répertoire de sortie
TARGET_DIR = 'Split_datasetDDR'
# Ratios de séparation entre l'entraînement, la validation et le test
train_ratio = 0.7
val_ratio = 0.15
test_ratio = 0.15
# creation des dossier train/val/test
for split in ['train', 'val', 'test']:
    for i in range(5):
        os.makedirs(os.path.join(TARGET_DIR, split, str(i)), exist_ok=True)
# Effectuer un traitement sur chaque classe de 0 à 4
for class_name, label in class_map.items():
    class_dir = os.path.join(SOURCE_DIR, class_name)
    images = [f for f in os.listdir(class_dir) if f.endswith('.png')]

    train_imgs, temp_imgs = train_test_split(images, train_size=train_ratio, random_state=42)
    val_imgs, test_imgs = train_test_split(temp_imgs, test_size=test_ratio / (val_ratio + test_ratio), random_state=42)

    for img_name in train_imgs:
        shutil.copy(os.path.join(class_dir, img_name), os.path.join(TARGET_DIR, 'train', str(label), img_name))

    for img_name in val_imgs:
        shutil.copy(os.path.join(class_dir, img_name), os.path.join(TARGET_DIR, 'val', str(label), img_name))

    for img_name in test_imgs:
        shutil.copy(os.path.join(class_dir, img_name), os.path.join(TARGET_DIR, 'test', str(label), img_name))
```

FIGURE 4.43 – La répartition des images du dataset en des sous-ensembles train/val/test

• Normalisation d'image : YOLOv8 applique automatiquement la normalisation d'image en interne en fonction de la norme utilisée lorsque son modèle a été pré-entraîné sur ImageNet pour optimiser l'apprentissage et accélérer la convergence. Par conséquent, nous avons pas besoin de normaliser les pixels manuellement.

4.5 Apprentissage et hyperparamètres des modèles

4.5.1 Modèle 1 DenseNet121

Pour entraîner notre modèle basé sur l'architecture DenseNet121, nous avons accordé une attention particulière au choix des hyperparamètres, car ces derniers influencent fortement les performances du modèle. Dans notre expérimentation, les valeurs suivantes ont été définies empiriquement :

- **Optimizer** : nous avons utilisé l'optimiseur Adam avec un taux d'apprentissage initial fixé à 10^{-4} ou 10^{-5}, car ils permetent une convergence plus rapide et efficace.
- **Loss Function** : la fonction de perte utilisée est categorical_crossentropy, adaptée à notre problème de classification multi-classes.
- **Epochs / Batch size** : nous avons fixé le nombre d'époques de 30 à 50 et la taille des lots (batches) à 32 images par itération.

Afin d'améliorer la performance du modèle tout en évitant le surapprentissage, plusieurs techniques de régularisation et d'optimisation ont été mises en œuvre :

1. **EarlyStopping** : interrompt automatiquement l'entraînement si la performance sur les données de validation ne s'améliore plus (après 3 époques), tout en rétablissant les meilleurs poids obtenus.

2. **ReduceLROnPlateau** : réduit automatiquement le taux d'apprentissage de moitié si l'accuracy de validation reste inchangée pendant 2 époques consécutives.

4.5.2 Modèle 2 ViT-B/16

Pour entraîner notre modèle Vision Transformer (ViT), nous avons adopté plusieurs techniques afin d'optimiser l'apprentissage tout en limitant le surapprentissage :

- **Initialisation** : Le modèle vit_b_16 de torchvision a été utilisé sans la tête de classification d'origine. Celle-ci a été remplacée par une couche linéaire adaptée à notre problème à 5 classes, précédée d'un Dropout de 30%.
- **Chargement des poids** : Les poids pré-entraînés ont été importés depuis un fichier .pth en utilisant map_location="cpu" afin d'éviter une surcharge GPU.
- **Optimiseur** : Adam avec un taux d'apprentissage initial de 10^{-4}.
- **weight_decay** : un décroissance du poids de 10^{-4}, est une technique qui empêche les poids de devenir trop grands pendant l'entraînement, en les réduisant un peu à chaque mise à jour afin d'éviter le surapprentissage et améliorer la généralisation du modèle.
- **Loss Function** : Nous avons utilisé la fonction CrossEntropyLoss avec label smoothing (0.1) qui est une méthode utilisée pour empêcher que le modèle soit trop sûr de lui pendant l'entraînement, remplaçe une étiquette vrai (comme [0, 0, 1, 0, 0]) par une version floue (comme [0.05, 0.05, 0.8, 0.05, 0.05]).
- **Epochs** : nous avons fixé le nombre d'époques à 30.

Afin de renforcer l'efficacité du modèle, en empêchant plusieurs problème, Différents méthodes de régularisation et d'optimisation ont été mises en œuvre :

1. **Scheduler** : Un StepLR qui divise le taux d'apprentissage par deux toutes les 10 époques.
2. **EarlyStopping** : Nous avons implémenté un mécanisme d'arrêt anticipé avec une patience de 5 époques, ce qui permet d'interrompre l'apprentissage si la perte de validation ne s'améliore plus.
3. **Accumulation des gradients** : pour contourner les limites mémoire du GPU, les gradients ont été accumulés sur 8 mini-batchs avant chaque mise à jour des poids.
4. **Précision mixte (AMP)** :C'est une astuce d'optimisation utilisée lors d'entrainement des réseaux de neurones complexes sur un grand jeu de données, elle combine entre les calculs en 16 bits (moins précis mais plus rapides) et en 32 bits (plus précis mais prennent du temps et beaucoup de mémoire). Et donc la précision mixte utilise des nombres en 16 bits pour faire la plupart des calculs et des nombres en 32 bits pour les calculs très importants dans l'objectif de d'accélérer l'entrainement en diminuant la consommation de la mémoire sans perdre la qualité du modèle.

4.5.3 Modèle 3 YOLOv8

Pour la classification d'images à l'aide de YOLOv8, nous avons utilisé l'architecture légère yolov8n-cls.pt afin de limiter le surapprentissage. L'entraînement a été réalisé avec les paramètres suivants :

- **Architecture** : YOLOv8-nano , adaptée aux environnements à ressources limitées.
- **Taille d'image** : les images ont été redimensionnées à 224×224 pixels pour correspondre à l'entrée du modèle.

- **Optimiseur** : nous avons utilisé l'optimiseur AdamW, est plus robuste pour les modèles profonds avec un taux d'apprentissage de 5×10^{-4} pour une convergence plus stable.
- **Epochs / Batch size** : 120, avec un mécanisme d'early stopping activé si aucune amélioration n'est observée pendant 10 époques et avec taille des batchesà 64 images par itération, ce qui accélère l'entraînement tout en maintenant la stabilité.

Pendqnt l'entrainement nous avons utilisé l'Accuracy comme une métrique d'évaluation afin de surveiller le taux d'apprentissage, les Figures 4.44, 4.45 et 4.46 montrent quelques performances à certains epochs d'entrainement.

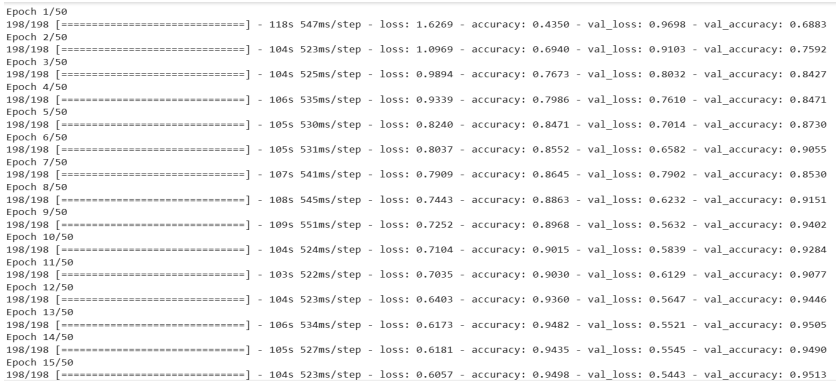


FIGURE 4.44 – Apprentissage du Modèle 1

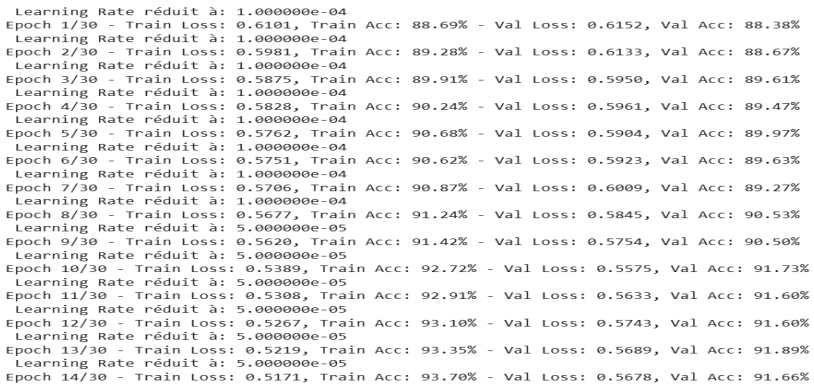


FIGURE 4.45 – Apprentissage du Modèle 2

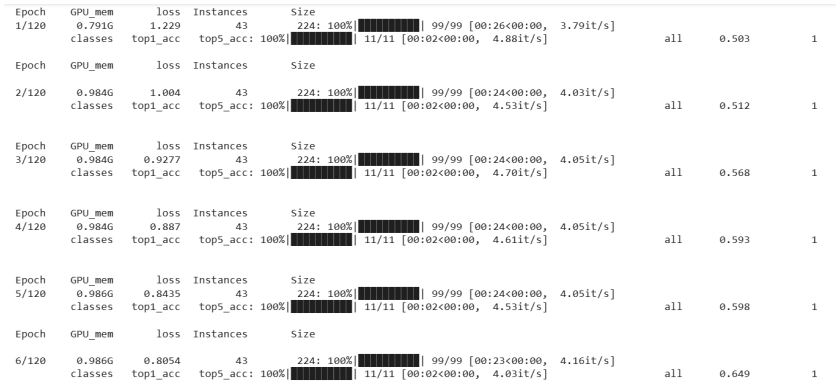


FIGURE 4.46 – Apprentissage du Modèle 3

4.6 Résultats et Discussion

L'évaluation des perfermances des modèles a été effectuée sur deux jeux de données différents discutés dans la section 4.3. Des expérimentations ont été menées en utilisant des sous-ensembles séparés pour l'apprentissage, la validation et le test.

Concernant le dataset Aptos 2019, Le modèle 1 (DenseNet121) a été testé selon deux approches :

- Approche directe : il comporte 7 302 213 paramètres.
- Approche en cascade : il comprend trois configurations correspondant aux niveaux de classification 7 039 554, 7 169 091 et 7 204 037 paramètres respectivement pour la classification binaire, à trois classes et à cinq classes.

Le modèle 2 (ViT-B/16) intègre 85 802 501 paramètres tandis que le modèle 3 (YOLOv8) utilise 1 444 693 paramètres.

Pour dataset DDR, les mêmes ont été appliqués, en conservant une configuration similaire de nombre de paramètres : 7 042 629 paramètres, pour le modèle 1 (approche directe), 85 802 501 paramètres pour le modèle 2 (ViT-B/16), et 1 444 693 paramètres pour le modèle 3 (YOLOv8).

A. Dataset-1 (Aptos 2019)

Les Figures 4.47, 4.48, 4.49 et 4.46 illustrent les performances des modèles lors de la phase d'apprentissage où l'on trouve la fonction perte (Loss) et l'accuracy sur l'ensemble d'entrainement et de validation. La faible variance dans les graphes indique que nos modèles ne sont pas sur_ajustés (overffiting).

Au cours de la phase d'apprentissage selon l'approche directe appliquée au modèle 1, nous avons configuré le paramètre EarlyStopping avec une patience de 5 itérations. Ainsi, l'apprentissage à été s'arrêté à la 40ème époque sur 50, avec un taux d'accuracy de 86.53% pour l'ensemble d'entraînement et 83.46% pour l'ensemble de validation, ce qui correspond à une valeur de perte respectivement de 0.6988 et 0.7424 comme illustré dans la Figure 4.47. L'apprentissage a duré 1 heure, 1 minute et 18 secondes.

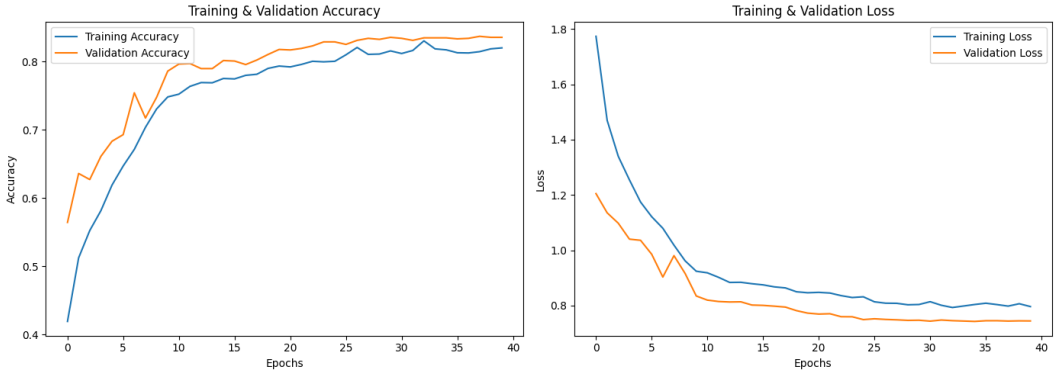


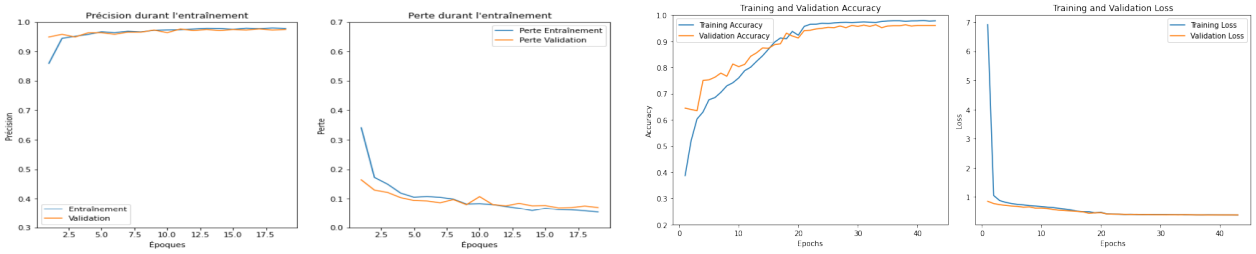
FIGURE 4.47 – Performances de l'apprentissage du Modèle 1 (approche directe)

Par la suite une approche en cascade adoptée au modèle 1, trois sous-modèle ont été entrainés successivement didiés à la classification selon les trois niveaux binaire, trois classes, puis cinq classes.

- Pour la classification binaire, L'apprentissage a été s'arrêté à 19ème époque sur 50, suite à l'activation de EarlyStopping avec une patience de 2 itérations. Il attient un taux d'accuracy de 98.05% pour l'ensemble d'entraînement et 97.27% pour l'ensemble de validation, ce qui correspond à une valeur de perte respectivement de 0.0520 et 0.0683 comme illustré dans la Figure 4.48(a). L'apprentissage a duré 11 minutes et 4 secondes.
- Pour la classification en trois classes, nous avons configuré EarlyStopping à 5. Ainsi, l'arrêt de l'pprentissage s'est produit à la 43ème époque sur 50, avec un taux d'accuracy de 98.15% pour l'ensemble d'entraînement et 96.31% pour l'ensemble de validation, ce qui correspond à

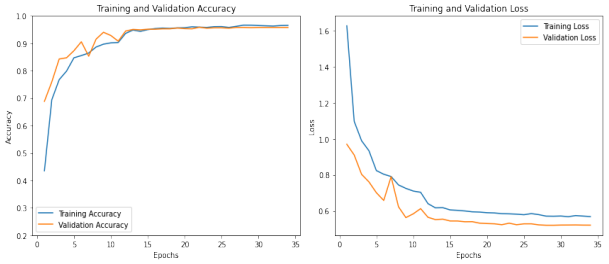
une valeur de perte respectivement de 0.3360 et 0.3828 comme illustré dans la Figure 4.48(b) et pour une duré de 14 minutes et 57 secondes.

• Enfin, pour la classification en cinq classes, l'apprentissage à duré 34 époque sur 50, suite à l'activation de EarlyStopping également 5 itérations. Il a obtenu un taux d'accuracy de 98.21% pour l'ensemble d'entraînement et 95.72% pour l'ensemble de validation, avec une valeur de perte respectivement de 0.4528 et 0.5205 comme illustré dans la Figure 4.48(c) et une durée de 59 minutes et 38 secondes.



(a) Classification binaire

(b) Classification à 3 classes



(c) Classification à 5 classes

FIGURE 4.48 – Performances de l'apprentissage du Modèle 1 (approche en cascade)

Concernant le modèle 2, l'EarlyStopping est défini également 5 itérations. L'apprentissage est interrompu en 27ème époque sur un maximum de 30. Le modèle a attient un un taux d'accuracy de 98.10% pour l'ensemble d'entraînement et 92.45% pour l'ensemble de validation, ce qui correspond à une valeur de perte respectivement de 0.4271 et 0.5346 (voir Figure 4.49) et une durée de 2 heures, 40 minutes et 2 secondes.

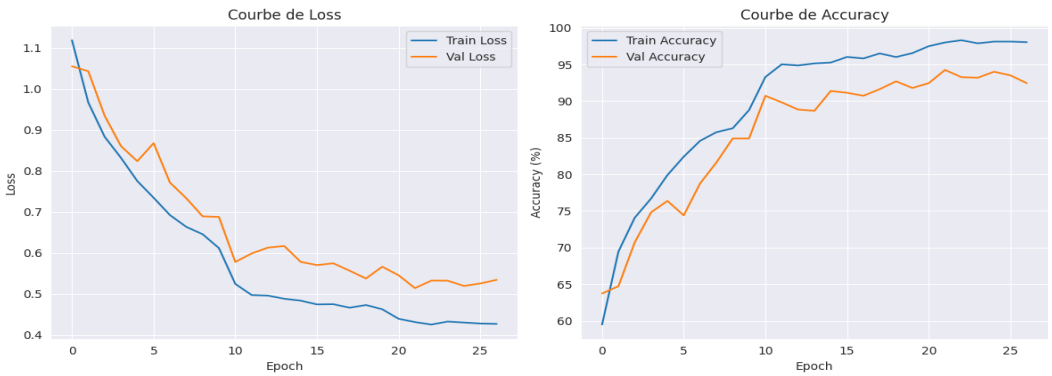


FIGURE 4.49 – Performances de l'apprentissage du Modèle 2

Enfin le modèle 3, nous avons configuré le paramètre EarlyStopping à 10 itérations. Ainsi, l'apprentissage à été cassé au bout de la 100ème époque sur 120, avec un taux d'accuracy de 97.42% pour l'ensemble d'entraînement et 91.66% pour l'ensemble de validation, ce qui correspond à un une valeur de perte respectivement de 0.15 et 0.17 Le modèle a également

obtenu une top-5 accuracy de 100%, indiquant que la bonne classe était toujours parmi les cinq premières prédictions (voir Figure 4.50). L'apprentissage a duré 45 minutes.

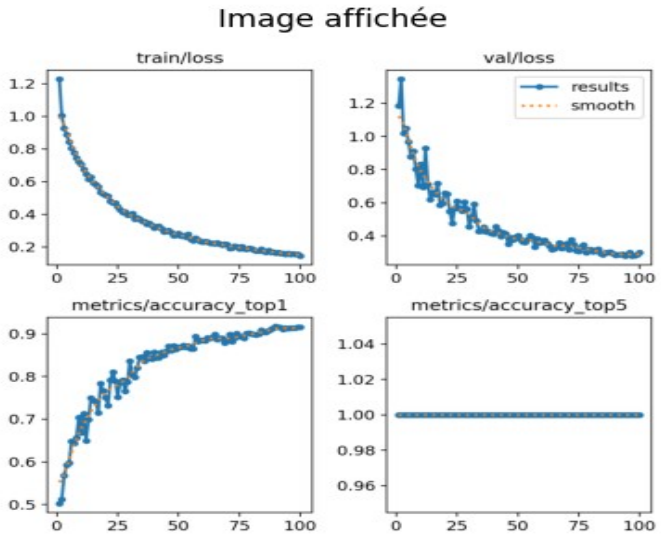


FIGURE 4.50 – Performances de l'apprentissage du Modèle 3

De plus, nous montrons la matrice de confusion pour une analyse et une visualisation faciles des mauvaise classification. La matrice de confusion nous permet de comparer les prédictions des modèles relatif à la vraie catégorie (valeur réelle). Chaque ligne de la matrice représente la classe prédite, tandis que chaque colonne représente une instance de la classe réelle (ou vice versa). Le tableau 4.1 représente la matrice de confusion pour un problème de classication binaire.

	Classe réelle positive	**Classe réelle négative**
Classe prédite positive	True Positif (TP)	False Positif (FP)
Classe prédite négative	False Négatif (FN)	True Négatif (TN)

TABLE 4.1 – Matrice de confusion binaire

Telle que :

• Vrai Positif (True Positif "TP") : lorsque la classe prédite est positive et la classe réelle est également positive ;

• Faux Positif (False Positif "FP") : lorsque la classe prédite est positive mais la classe réelle est négative ;

• Vrai Négatif (True Negatif "TN") : lorsque la classe prédite est négative et la classe réelle est également négative ;

• Faux Négatif (False Negatif "FN") : lorsque la classe prédite est négative mais la classe réelle est positive.

Dans le but d'évaluer la qualité de prédiction de chaque modèle de manière rigoureuse, nous utilisons les métriques classiques de classification que sont :

• Précision : Il s'agit du nombre total de vrais positifs divisé par le nombre total d'entrées marquées comme appartenant à la classe positive ;

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

• Rappel : Il s'agit du nombre total de vrais positifs divisé par le nombre total d'éléments qui appartiennent réellement à la classe positive ;

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

• F1_score : Calculé en combinant la précision et le rappel du modèle. C'est particulièrement utile lorsque les classes sont déséquilibrées.

$$F_1 = 2\left(\frac{\text{PrécisionRappel}}{\text{Précision} + \text{Rappel}}\right)$$

Les résultats illustrés par les Figures 4.51, 4.52 et 4.53, représentent les matrices de confusion des modèles, où l'on trouve les cinq (5) classes sur les axes X et Y. En revanche, dans l'approche en cascade, les matrices sont affichées séparément pour chaque niveau de classification, binaire (2 classes), en trois classes, puis en cinq classes (voir Figures 4.52).

Selon l'approche directe appliquée au architecture du modèle 1, un total de 237 images sur 1355 ont été mal classées (17.5%), comme le montre la Figure 4.50. Il s'agit notamment de huit (8) images de la classe Mild qui ont été classées à tort comme No_DR, treize (13) comme Modérate . Du côté de la classe Modérate, cinquante-trois (53) images ont été prédites à tort comme Mild, cinq (5) comme No_DR, neuf (9) comme Proliferate_DR et huit (8) comme Sévère. Pour la classe No_DR, douze (12) images ont été mal classées comme Mild, quatre (4) comme Modérate, et une (1) comme Proliferate_DR , et une (1) comme Sévère ,. Concernant la classe Proliferate_DR, dix-huit (18) images ont été confondues avec Mild, quante-six (46) avec Modérate et douze (12) avec Sévère. Enfin, pour la classe Severe, quatre (4) images ont été classées comme Mild, trente-quatre(34) images ont été mal classées comme Modérate, et dix (10) comme Proliférate_DR.

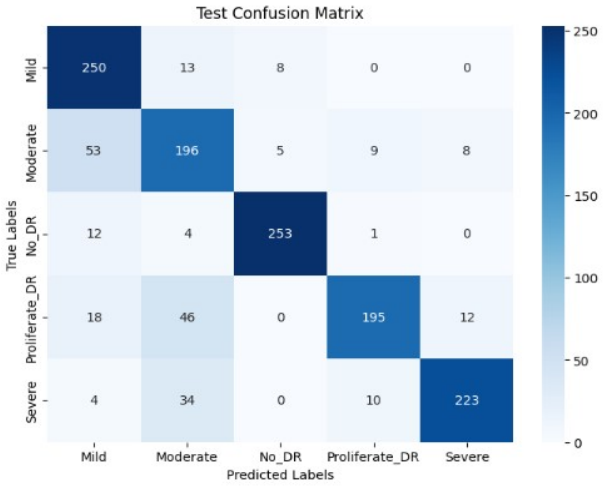


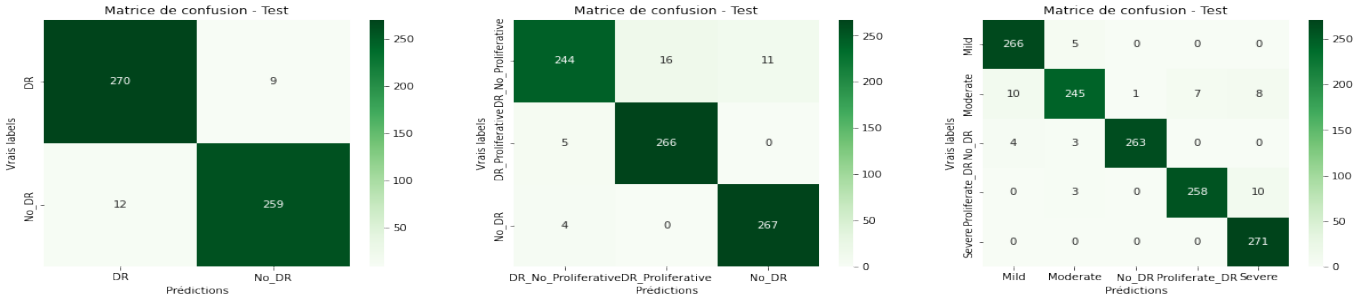
FIGURE 4.51 – Matrice de confusion du Modèle 1 (approche directe)

Pour l'approche en cascade du modèle 1, trois étapes successives ont été réalisés. Lors de la première étape (classification binaire), un total de 21 images sur 529 ont été mal classées (3.96%), comme le montre la Figure 4.52(a). Il s'agit notamment de neuf (9) images de la classe DR qui ont été classées à tort comme No_DR, et douze (12) images de classes No_DR ont été classées comme DR.

À la deuxième étape (classification en trois classes), un total de 36 images sur 912 ont été mal classées (3.94%), comme le montre la Figure 4.52(b) . Il s'agit notamment de seize (16) images

de la classe DR_No_Proliferative qui ont été classées à tort comme DR_Prolifération, et onze (11) images de classe DR_No_Proliferative ont été classées comme DR. Concernant la classe DR_Proliferative cinq (5) ont été détectées comme classe DR_No_Proliferative. A la fin pour la classe No_DR quatre (4) images ont été prédites comme classe DR_No_Proliferative.

À la fin, au niveau de la classification fine (en cinq classes), un total de 51 images sur 1355 ont été mal classées (3.76%), comme le montre la Figure 4.52(c). Il s'agit notamment de cinq (5) images de la classe Mild qui ont été classées à tort comme Modérate. Du côté de la classe Modérate, dix (10) images ont été prédites à tort comme Mild, une (1) comme No_DR, sept (7) comme Proliferate_DR et huit (8) comme Sévère. Pour la classe No_DR, quatre (4) images ont été mal classées comme Mild, trois (3) comme Modérate. Concernant la classe Proliferate_DR, trois (3) avec Modérate et dix (10) avec Sévère. Enfin, pour la classe Sévère, tout est bien classées.



(a) Matrice binaire (b) Matrice de 3 classes (c) Matrice de 5 classes

FIGURE 4.52 – Les matrices de confusion du Modèle 1 (approche en cascade)

Pour l'architecture du Modèle 2, un total de 58 images sur 903 ont été mal classées (6.4%), comme le montre la Figure 4.53 . Il s'agit notamment de qautre (4) images de classe 0 (No_DR) comme étant de classe 1 (Mild). Pour la classe 1, cinq (5) images ont été mal classées comme classe 0 (No_DR). Aussi cinq (5) images de la classe 1 ont été classées comme de classe 2 (Moderate). Pour la classe 2, trois (3) images ont été prédites à tort comme classe 0 (No_DR), huit (8) images de classe 2 ont été predites comme classe 1 (Mild),et trois (3) images de classe 2 ont été classées comme classe 3 (sévère) et aussi trois (3) images de classe 2 ont été prédites comme classe 4 (Proliferate_DR). Pour la classe 3 une (1) image a été prédite comme classe 1 et six (6) images comme classes 2 et à la fin trois (3) images comme classes 4 (Proliferate_DR). Concernant la derniere classe 4 , huit(8) images ont ete predite comme classe 2 et six (6) comme classe 3 .

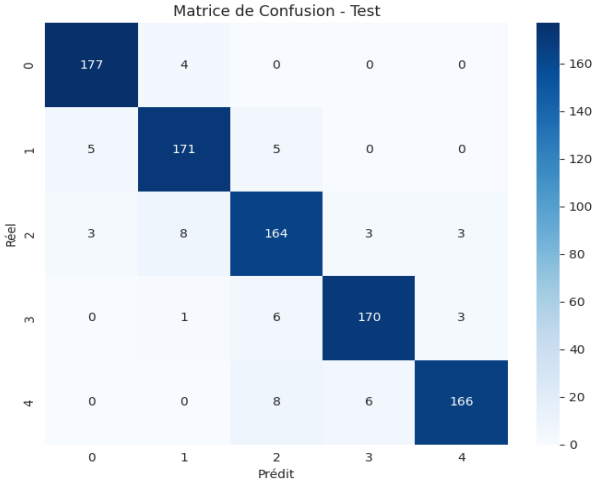


FIGURE 4.53 – Matrice de confusion du Modèle 2

Le Modèle 3, un total de 112 images sur 1355 ont été mal classées (8.2%)(voir Figure 4.54). Il s'agit notamment de sept (7) image de classe 0 (No_DR) comme étant de classe 1 (Mild), et trois (3) images de classe 0 qui ont été classées comme classe 2 (Modérate). Pour la classe 1 (Mild), deux (2) images ont été mal classées comme classe 0. En plus, onze (11) images de classe 1 (Mild) ont été mal classées comme classe 2 (Modérate) et une (1) image comme classe 3 , trois (3) images de classe 1 ont été classées comme classe 4 (Proliferate_DR). Pour la classe 2, vingt (20) images ont été mal prédites comme classe 1, quatorze (14) images classe 2 (Modérate) ont été prédites tort comme classe 3, et vingt-deux (22) images de classe 2 ont été classées comme classe 4 (Proliferate_DR). Pour la classe 3 douze (12) image ont été prédite comme classe 2 aussi a la fin huit (8) images comme classes 4. Enfin, pour la classe 4 une (1) image été prédite comme classe 2 et cinq (5) classe comme classe 3.

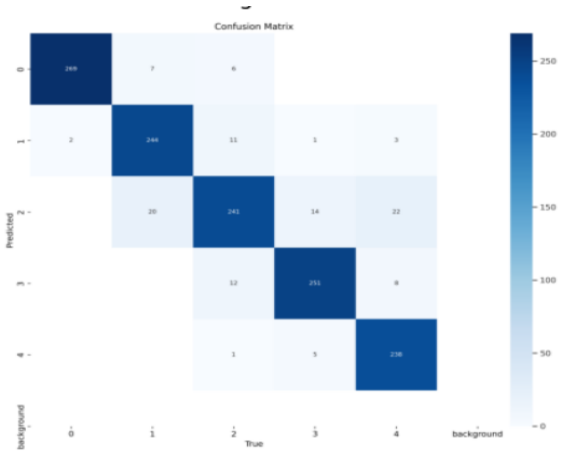


FIGURE 4.54 – Matrice de confusion du Modèle 3

B. Dataset-2 (DDR)

Les Figures 4.55, 4.56 et 4.57 illustrent les performances des modèles lors de la phase d'apprentissage où l'on trouve la fonction perte (Loss) et l'accuracy sur l'ensemble d'entrainement et de validation. La faible variance dans les graphes indique que nos modèles ne sont pas sur_ajustés (overffiting).

Au cours de la phase d'apprentissage du modèle 1, nous avons configuré le paramètre EarlyStopping avec une patience de 5 itérations. Ainsi, l'apprentissage à été s'arrêté à la 25ème époque sur 30, avec un taux d'accuracy de 94,4% pour l'ensemble d'entraînement et 91.1% pour l'ensemble de validation, ce qui correspond à un taux d'erreur respectivement de 0.1592 et 0.2485 comme illustré dans la figure 4.55. La durée totale était de 106 minutes (= 1h 46 min).

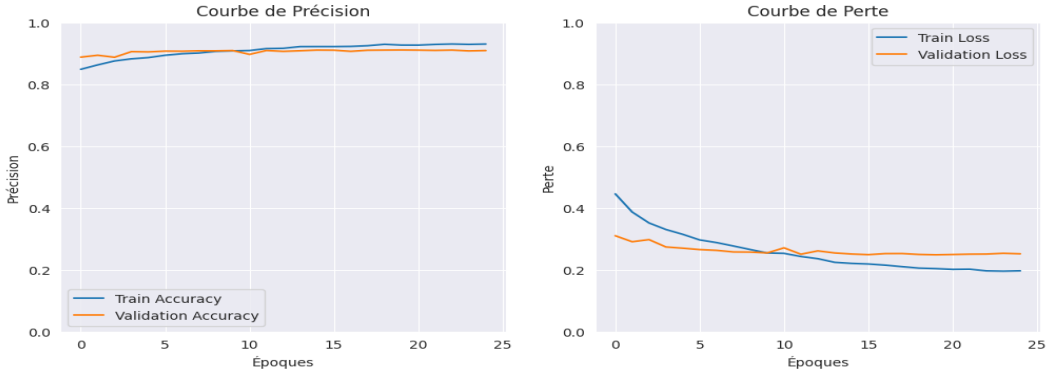


FIGURE 4.55 – Performances de l'apprentissage du Modèle 1

Ensuite concernant le modèle 2, le paramètre EarlyStopping est défini à 5 itérations. L'apprentissage est interrompu en 15ème époque sur un maximum de 30. Le modèle a attient un un taux d'accuracy de 94,08% pour l'ensemble d'entraînement et 91.06% pour l'ensemble de validation, ce qui correspond à un taux d'erreur respectivement de 0.5132 et 0.5761 comme illustré dans la figure 4.56 et de durée 1 heure, 57 minutes et 31 secondes.

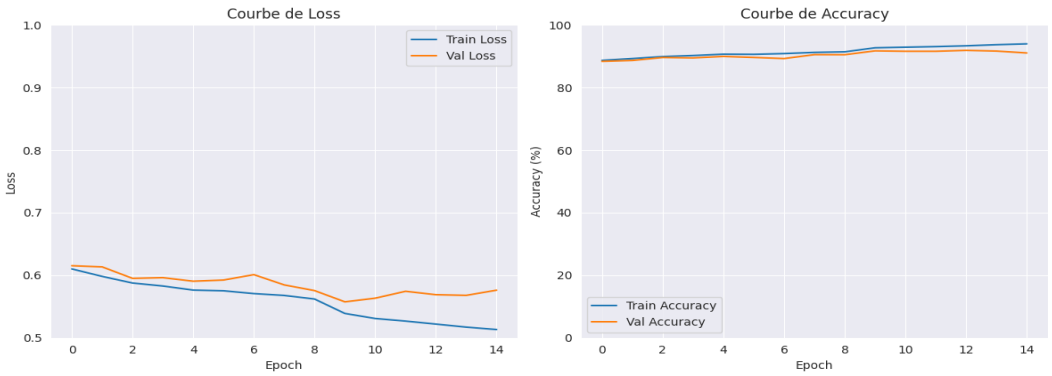


FIGURE 4.56 – Performances de l'apprentissage du Modèle 2

Enfin pour le Modèle 3, nous avons configuré le paramètre EarlyStopping à 10. Ainsi, l'apprentissage s'est arrêté au bout de la 48ème époque sur 50, avec un taux d'accuracy de 96.94% pour l'ensemble d'entraînement et 92.94% pour l'ensemble de validation, ce qui correspond à un taux d'erreur respectivement de 0.124 et 0.22. Le modèle a également obtenu une top-5 accuracy de 100%, indiquant que la bonne classe était toujours parmi les cinq premières prédictions comme illustré dans la figure 4.57. L'apprentissage a duré 72 minutes.

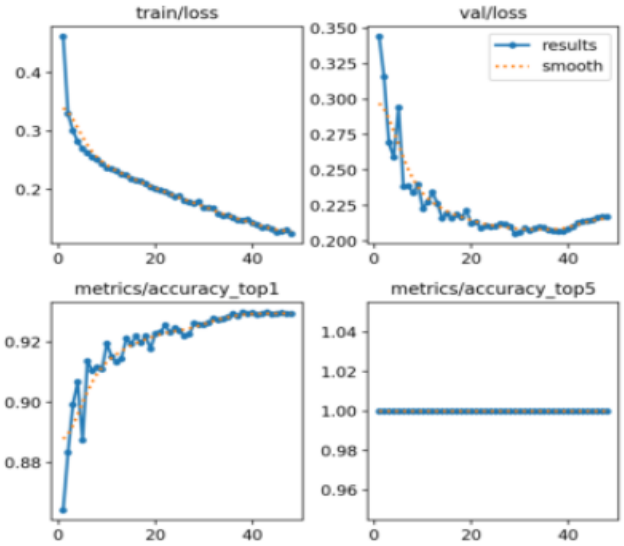


FIGURE 4.57 – Performances de l'apprentissage du Modèle 3

De plus, nous montrons les matrices de confusion pour une analyse et une visualisation faciles des mauvaise classification. Les résultats illustrés par les Figures 4.58, 4.59 et 4.60, représentent la matrice de confusion pour chaque modèle, où l'on trouve les cinq (5) classes sur les axes X et Y.

Avec l'architecture du Modèle 1, un total de 289 images sur 3315 ont été mal classées (8.7%) comme le montre la Figure 4.58. Il s'agit notamment de cinq (5) images de classe 0 (No_DR) comme étant de classe 1 (Mild), et soixante-neuf (69) images de classe 0 (No_DR) qui ont été

classées comme classe 2 (Moderate), trois (3) images de classe 0 (No_DR) ont été mal classées comme classe 4 (Proliferative_DR). Pour la classe 1, Trente-cinq (35) images de la classe 1 (Mild) ont été mal classées comme classe 0 (No_DR). En plus, vingt-deux (22) images de classe 1 (Mild) ont été mal classées comme classe 2 (Moderate), une (1) image de classe 1 (Mild) a été classe comme classe 4 (Proliferative_DR). Pour la classe 2 (Moderate), qautre-vingt-dix (90) images ont été prédites a tort comme classe 0 (No_DR), huit (8) images classe 2 (Modérate) ont été prédites comme classe 1 (Mild), et dix-sept(17) images de classe 2 ont été classées comme classe 4 (Proliferative_DR). Pour la classe 3 treize (13) images ont été prédites comme classe 2 et deux (2) images comme classes 4. Enfin, pour la classe 4 vingt-trois (23) images ont été mal classées comme classe 2 et une image comme classe 1.

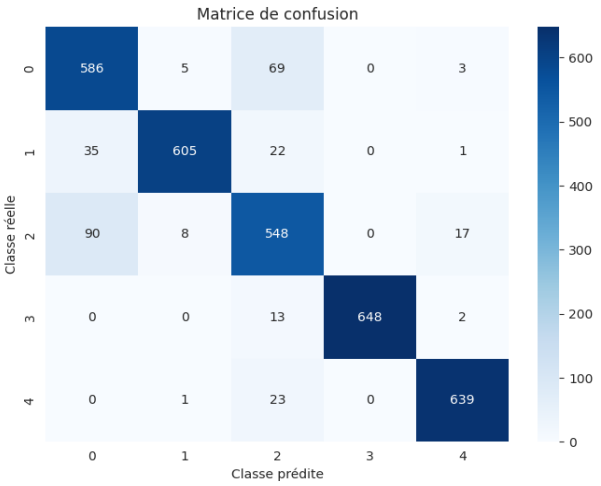


FIGURE 4.58 – Matrice de confusion du Modèle 1

Pour l'architecture du Modèle 2, un total de 282 images sur 3315 ont été mal classées (8.5%), comme le montre la Figure 4.59. Il s'agit notamment de une (1) image de classe 0 (No_DR) comme étant de classe 1 (Mild), et cinquante (50) images de classe 0 (No_DR) qui ont été classées comme classe 2 (Modérate), deux (2) images de classe 0 (No_DR) ont été mal classées comme classe 4 (Proliferative_DR). Pour la classe 1, Trente-deux (32) images de la classe 1 (Mild) ont été mal classées comme classe 0 (No_DR). En plus, vingt-deux (22) images de classe 1 (Mild) ont été mal classées comme classe 2 (Moderate). Pour la classe 2 (Modérate) , qautre-vingt-dix (90) images ont été prédites a tort comme classe 0 (No_DR), sept (7) images classe 2 (Modérate) ont été prédites comme classe 1 (Mild), et quatorze (14) images de classe 2 ont été classées comme classe 3 (severe) et vingt-huit (28) images classe 2 (Moderate) ont été prédites comme classe 4. Pour la classe 3 une (1) image a été prédite comme classe 0 et treize (13) images comme classes 2 et à la fin deux (2) images comme classes 4. Enfin, pour la classe 4 deux (2) images ont été mal classees comme classe 0 et quatorze(14) images ont été classées comme classes 2, et quatre (4) images comme classe 3.

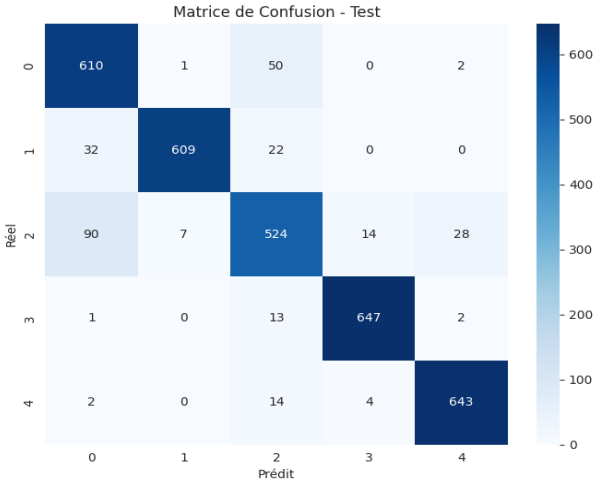


FIGURE 4.59 – Matrice de confusion du Modèle 2

Avec le Modèle 3, un total de 342 images sur 5250 ont été mal classées (6.5%), comme le montre la Figure 4.60. Il s'agit notamment de quarante-six (46) image de classe 0 (No_DR) comme étant de classe 1 (Mild),et quatre-vingt-neuf (89) images de classe 0 (No_DR) qui ont été classées comme classe 2 (Moderate), une (1) image de classe 0 (No_DR) a été mal classée comme classe 3 (severe). Pour la classe 1, onze (11) images de la classe 1 (Mild) ont été mal classées comme classe 0 (No_DR). En plus, dix-sept (17) images de classe 1 (Mild) ont été mal classées comme classe 2 (Moderate) et trois (3) images de classe 1 ont été classees comme classe 4. Pour la classe 2 (Moderate), soixante-neuf (69) images ont été predites a tort comme classe 0 (No_DR), quarante-quatre (44) images classe 2 (Moderate) ont été predites comme classe 1 (Mild), et vingt-deux (22) images de classe 2 ont été classees comme classe 3 (severe) et trente-trois (33) images classe 2 (Moderate) ont été predites comme classe 4 (Proliferative_DR). Pour la classe 3 deux (2) images ont été predite comme classe 2 et aussi a la fin deux (2) images comme classes 4 (Proliferative_DR). Enfin, pour la classe 4 une (1) image a été mal classee comme classe 0 et aussi une image a été classe comme classe 1 et dix (10) images ont été classees comme classes 2, et une (1) image comme classe 3.

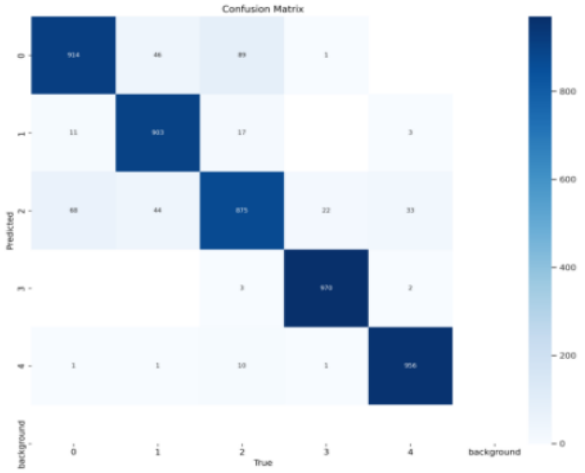


FIGURE 4.60 – Matrice de confusion du Modèle 3

La classification erronée des images de rétinopathie diabétique peut être attribuée à la forte similitude visuelle entre certains stades de la maladie, ainsi qu'à une variabilité intra-classe importante. En fait, même les images appartenant au même stade (par exemple, modéré ou

proliférative_DR) peuvent présenter des différences significatives de texture, de contraste ou de présence de lésions en raison de facteurs biologiques individuels ou de la qualité des images acquises. Ce changement visuel rend la tâche de classification plus difficile pour le modèle, en particulier lorsque les limites entre les étapes sont subtiles ou mal définies.

C. Récapitulatif des résultats

- Aptos 2019 dataset

La Figure 4.61 montre les performances de nos trois modèles sur les données de test sur dataset Aptos 2019, où l'on trouve les statistiques pour chaque type de classes (5 cas : classe 0 jusqu'à 4) ainsi que la moyenne des performances pour chaque modèle.

Classification Report (Test):

	precision	recall	f1-score	support
Mild	0.74	0.92	0.82	271
Moderate	0.67	0.72	0.70	271
No_DR	0.95	0.94	0.94	270
Proliferate_DR	0.91	0.72	0.80	271
Severe	0.92	0.82	0.87	271
accuracy			0.82	1354
macro avg	0.84	0.83	0.83	1354
weighted avg	0.84	0.82	0.83	1354

(a) Modèle 1 (approche directe)

Rapport de classification (Test) :

	precision	recall	f1-score	support
Mild	0.95	0.98	0.97	271
Moderate	0.96	0.90	0.93	271
No_DR	1.00	0.97	0.99	270
Proliferate_DR	0.97	0.95	0.96	271
Severe	0.94	1.00	0.97	271
accuracy			0.96	1354
macro avg	0.96	0.96	0.96	1354
weighted avg	0.96	0.96	0.96	1354

(b) Modèle 1 (approche en cascade)

Rapport de classification :

	precision	recall	f1-score	support
0	0.957	0.978	0.967	181
1	0.929	0.945	0.937	181
2	0.896	0.906	0.901	181
3	0.950	0.944	0.947	180
4	0.965	0.922	0.943	180
accuracy			0.939	903
macro avg	0.939	0.939	0.939	903
weighted avg	0.939	0.939	0.939	903

(c) Modèle 2

	precision	recall	f1-score	support
0	0.930	0.978	0.953	271
1	0.881	0.849	0.865	271
2	0.774	0.845	0.808	271
3	0.928	0.908	0.918	271
4	0.919	0.841	0.879	271
accuracy			0.884	1355
macro avg	0.886	0.884	0.884	1355
weighted avg	0.886	0.884	0.884	1355

(d) Modèle 3

FIGURE 4.61 – Les Rapports de classification des modeles 1, 2 et 3 sur APTOS dataset

L'approche en cascade appliquée au Modèle 1 a atteint des performances supérieures, très bon équilibre entre precision et rappel sur toutes les classes, surpasse l'approche directe, meme les autres modèles 2 et 3. Elle a permis d'atteindre une précision de 96% et un F1-score équilibré, tandis que l'approche directe a atteint des performances très moindres de 82.50%. Le Modèle 2 est légèrement en retrait, mais toujours performant avec une précision de 93.9%. Pour le Modèle 3 est plus léger et rapide, mais a des performances moindres de 88.4%.

Pour une évaluation plus approfondie, le tableau 4.3 montre la comparaison de nos modèle avec D'autres modèles apparentés proposés dans la littérature utilisant la même dataset APTOS2019.

Auteurs	**Dataset**	**Modèles**	**Accuracy**
[Vinuja et al., 2021 [118]]	APTOS 2019	InceptionV3 Xception	93.10% et 91.90%
[Aruna Kumari et al., 2024 [119]]	APTOS 2019	Modèle CNN personnalisé combinant l'extraction de texture et d'intensité en niveaux de gris	93.2%
[Jian et al., 2024 [120]]	APTOS 2019	Triple-DRNet en cascade	92.08%
Notre travail 1	APTOS 2019	DenseNet121 (directe)	82.5%
Notre travail 1	APTOS 2019	DenseNet121 (cascade)	**96.30**%
Notre travail 2	APTOS 2019	ViT-B/16 (directe)	**93.91**%
Notre travail 3	APTOS 2019	YOLOv8 (directe)	88.4%

Table 4.3 – Tableau de comparaison des travaux sur le dataset Aptos 2019

[Jian et al., 2024 [120]] ont adopté une cascade en 3 étapes pour décomposer une classification complexe en sous-problèmes plus simples, en utilisant un modèle personnalisé composé de trois CNN en cascade appelé Triple-DRNet. Notre travail atteint +4% de performance avec une stratégie similaire mais avec un réseau plus efficace (DenseNet121), suggérant que la qualité de DenseNet121 et le prétraitement utilisé jouent un rôle clé. La même architecture DenseNet121 avec le même prétraitement, mais le taux de classification directe tombe à 82.5%, indiquant que le problème multi-classes est trop complexe pour être résolu en une seule étape, en traite les images de manière progressive et structurée, en particulier dans le cadre déséquilibré.

[Vinuja et al., 2021 [118]], ont utilisé InceptionV3 et Xception pour le prétraitement basé sur le flou gaussien et l'amélioration de l'image (rotation, retournement, éclaircissement), les performances du modèle ont atteint successivement 93,1% et 91.90 %. [Aruna Kumari et al, 2024 [119]] utilisent modèle CNN personnalisé combinant l'extraction de texture et d'intensité en niveaux de gris pour les stratégies de sélection de fonctionnalités afin d'obtenir un apprentissage plus ciblé, avec des performances atteignant 93,2%. La précision des deux méthodes était similaire.

Notre travail 2 atteint des performances compétitives, superieurs à celles de ces deux modèles [Vinuja et al., 2021 [118]] et [Aruna Kumari et al., 2024 [119]] +0.5 %. Notre travail 3 a obtenu un score de 88% est moins performant que les autres en classification pure, ce qui s'explique par son orientation initiale vers la détection d'objets plutôt que la classification.

- DDR (Dataset for Diabetic Retinopathy) dataset

La Figure 4.62 montre les performances de nos trois modèles sur les données de test sur dataset DDR, où l'on trouve les statistiques pour chaque type de classes (5 cas : classe 0 jusqu'à 4) et la moyenne des performances pour chaque modèle.

	precision	recall	f1-score	support
0	0.82	0.88	0.85	663
1	0.98	0.91	0.94	663
2	0.81	0.83	0.82	663
3	1.00	0.98	0.99	663
4	0.97	0.96	0.96	663
accuracy			0.91	3315
macro avg	0.92	0.91	0.91	3315
weighted avg	0.92	0.91	0.91	3315

(a) Modèle 1

	precision	recall	f1-score	support
0	0.830	0.920	0.873	663
1	0.987	0.919	0.952	663
2	0.841	0.790	0.815	663
3	0.973	0.976	0.974	663
4	0.953	0.970	0.961	663
accuracy			0.915	3315
macro avg	0.917	0.915	0.915	3315
weighted avg	0.917	0.915	0.915	3315

(b) Modèle 2

	precision	recall	f1-score	support
0	0.863	0.916	0.888	995
1	0.973	0.922	0.947	995
2	0.853	0.866	0.860	995
3	0.983	0.978	0.980	995
4	0.980	0.962	0.971	995
accuracy			0.929	4975
macro avg	0.930	0.929	0.929	4975
weighted avg	0.930	0.929	0.929	4975

(c) Modèle 3

FIGURE 4.62 – Les Rapports de classification des modeles 1, 2 et 3 sur DDR dataset

Le Modèle 3 a atteint des performances élevées légèrement à celles des deux autres modèles lors de l'apprentissage avec une précision de 93% et un F1-score équilibré. Mais les Modèles 1 et 2 sont un peu en retrait, mais toujours performant avec une précision presque égalité à 91.28% et 91.49%.

Pour une évaluation plus approfondie, le tableau 4.5 montre la comparaison de nos modèle avec D'autres modèles apparentés proposés dans la littérature utilisant la même dataset DDR.

Auteurs	**Dataset**	**Modèles**	**Accuracy**
[Alwakid et al., 2023 [96]]	DDR	DenseNet121	79,67%
[Nanda et al., 2022 [121]]	DDR	inceptionV3, ResNet18	83,2% 82,2%
[Alyoubi et al., 2021 [99]]	DDR	CNN512 fusionné avec YOLOv3	89%
Notre travail 1	DDR	DenseNet121 (directe)	**91.28%**
Notre travail 2	DDR	ViT-B/16 (directe)	**91.49%**
Notre travail 3	DDR	YOLOv8 (directe)	**93%**

TABLE 4.5 – Tableau de comparaison des travaux sur le dataset DDR

Comparé aux travaux, notre travail atteint une plus grande précision. Bien que [Alwakid et al., 2023 [96]] et nous utilisions le même modèle 1 (DenseNet121), nous obtenons de meilleurs résultats qu'eux (+11,9%). Bien qu'ils aient utilisé une méthode d'amélioration complexe (CLAHE + ESRGAN), la précision était encore significativement inférieure. Cela suggère que l'utilisation d'un prétraitement adapté à la tâche (recadrage, remplissage, redimensionnement) et d'une entrée cohérente dans le modèle DenseNet peut produire des résultats plus significatifs que l'augmentation visuelle brute.

Notre travail 2 ViT-B/16 avec un bon prétraitement (recadrage, remplissage, redimensionnement) surpasse considérablement l'approche de [Nanda et al., 2022 [121]], qui utilisait un CNN traditionnel avec un optimiseur de haut niveau (QHMO). Malgré les optimisations avancées de [Nanda et al., 2022], notre méthode atteint une meilleure généralisation, obtenant une amélioration de +8,4% par rapport à leur meilleur résultat. Cela démontre la puissance des modèles de transformation même sans optimiseurs complexe et que le prétraitement ciblé est crucial pour la préparation des images médicales.

En raison de la nouvelle architecture et du prétraitement standardisé, notre travail 3 YOLOv8 surpasse la version YOLOv3 combiné avec CNN512 de [Alyoubi et al., 2021 [99]] (qui subit divers traitements, notamment l'augmentation, le nettoyage et la normalisation) après le prétraitement (recadrage, remplissage et redimensionnement). Le gain de 4% suggère que l'évolution de l'architecture YOLO est bénéfique.

4.6.1 Explication Modele Grad-CAM (Gradient-weighted Class Activation Mapping)

Dans le cadre de cette tâche, nous avons mis en œuvre la méthode Grad-CAM proposée par [Selvaraju et al., 2017 [122]] afin d'expliquer les décisions prises par notre modèle de classification basé sur les architectures (DenseNet121, ViT-B/16., YOLOv8). Grad-CAM est une technique de visualisation repose sur l'utilisation des gradients de la sortie associé à tout classe cible, circulant dans la dernière couche convolutive du réseau, qui permet de générer des cartes de localisation visuelle grossière mettant en évidence les régions importantes prédictées par le modèle dans l'image d'entrée (heatmaps). Dans l'objectif de rendre les modèles de deep learning plus interprétables et transparents, et aidant de comprendre les décisions de classification.

Dans notre cas, nous avons appliqué Grad-CAM sur les images de fond d'œil issues du dataset de la rétinopathie diabétique DDR afin de vérifier le modèle ViT-B/16 se focalise sur les régions pathologiques notamment les microanévrismes, les hémorragies ou les exsudats ... ect. Et afin d'illustré l'interprétation du modèle, nous avons visualisé quelques images rétiniennes appartenant à cinq classes de gravité (voir Figure 4.63). Dans l'objectif de montrer comment chaque modèle distingue visuellement entre les niveaux de la pathologie, effectivement, le modèle s'intéresse au zones importantes et qui permettent de différencier les différentes pathologies.

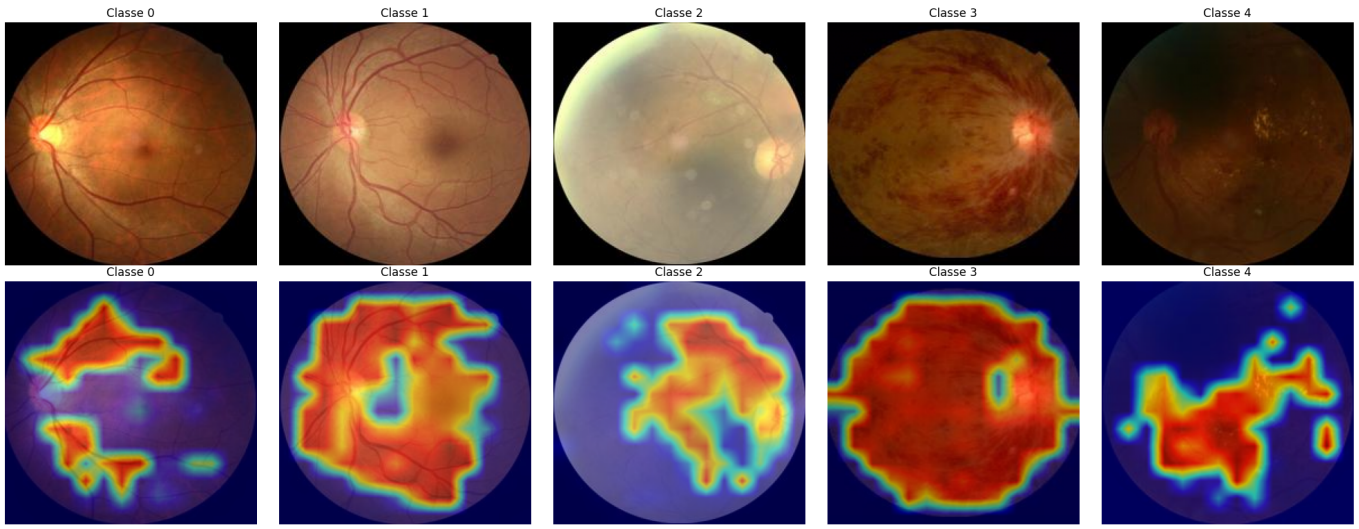


FIGURE 4.63 – Visualisation des images du dataset DDR appartenant a cinq classes de gravité utilisant Grad-CAM (ligne 1 : images originale, ligne 2 : images de sortie)

4.6.2 Interface Graphique Utilisateur (GUI)

Dans l'objectif de rendre notre système de classification de la rétinopathie diabétique (RD) adressé aux professionnels de santé et de faciliter la phase de test. Dans cette section, nous présentons une interface graphique simple, interactive et personnalisable qui a été conçue pour utiliser nos trois modèles présentés dans la section 4.4. Son développement a été réalisé en Python en utilisant la bibliothèque Tkinter.

Après le lancement de l'application, elle s'ouvre sur une interface d'accueil simple, comportant un seul bouton principal nommé "Entrer" comme illustré dans la figure 4.64.

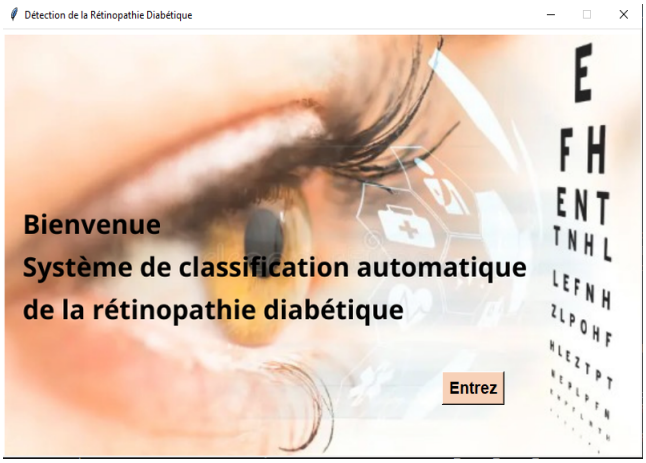


FIGURE 4.64 – Interface d'accueil

En cliquant sur ce bouton, l'utilisateur est redirigé vers une interface est considérée comme interface principale de l'application (voir Figure 4.65), où il pourra de :

- Saisir les informations personnelles du patient (Nom, Prénom et l'âge).
- Un bouton "Sélectionner" pour charger une image du fond d'œil.
- Un 2ème bouton "Analyser" pour lancer l'analyse automatique de l'image à l'aide des trois modèles intégrés (DenseNet121, ViT-B/16, YOLOv8), en affichant le pourcentage de classification dans les labels.
- Un autre bouton "Générer rapport" pour générer un bilan personnalisé médical sur état de patient.

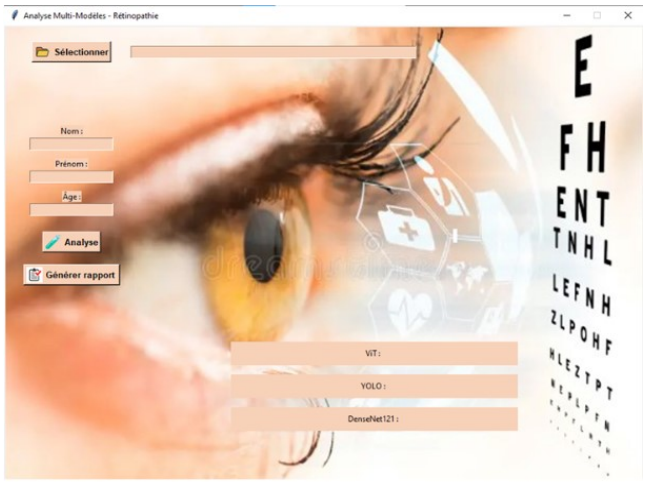


FIGURE 4.65 – Interface principale

Dans le but d'illustrer plus en détail le fonctionnement de notre application, nous exposons ci-dessous deux exemples concrets -voir Figure 4.66 et 4.67).
La première image est considérée comme saine (classe 0) tandis que la seconde est associée à un cas de rétinopathie diabétique proliférative (classe 4). Pour chaque exemple, nous montrons l'image analysée et les résultats du notre modèles, ainsi que le rapport généré du patient.

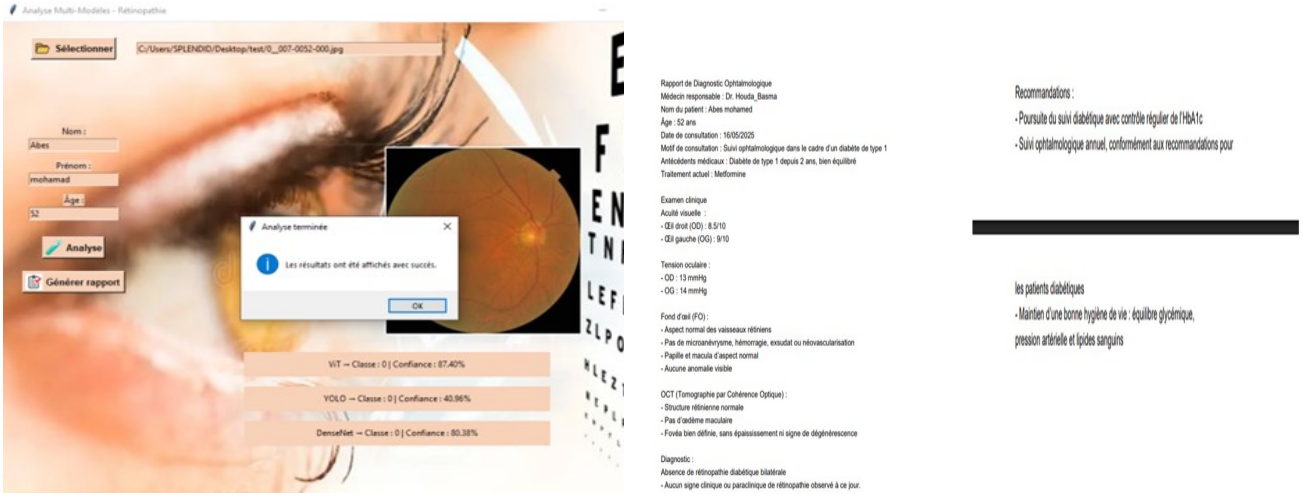


FIGURE 4.66 – Exemple d'analyse d'une image de rétine saine (classe 0)

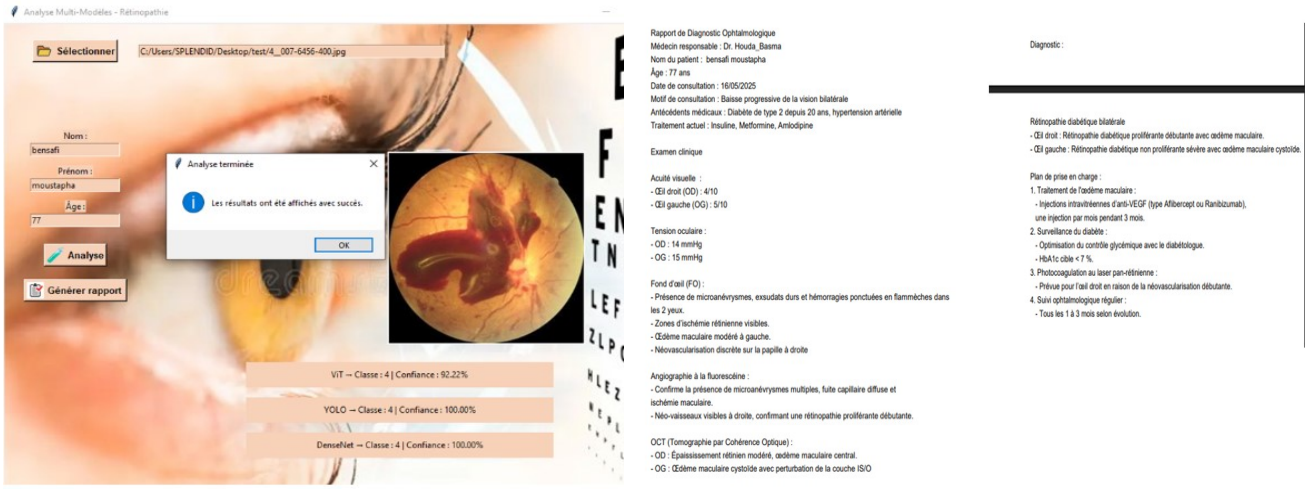


FIGURE 4.67 – Exemple d'analyse d'une image atteinte de rétinopathie proliférative (classe 4)

4.6.3 Difficultés rencontrées

Pendant le développement de notre système de détection de la rétinopathie diabétique, nous avons rencontré diverses complications liées majoritairement aux limitations matérielles des plateformes en ligne utilisées, y compris la plateforme Google Colab, qui fournit une RAM insuffisante et une limitation des ressources GPU, ce qui provoque des coupures fréquentes, surtout avec la capacité de nos données volumineuses de taille 224x224. Face à ces entraves, nous avons alors décidé de migrer vers la plateforme kaggle grâce à les ressources fournies (RAM plus élevée et un accès de 30 h GPU/semaine). Cependant, malgré ces améliorations, nous avons continué à rencontrer des saturations de mémoire GPU en raison de l'utilisation des architectures telles que DenseNet121 et ViT-B/16 avec un nombre important de paramètres, associés à un jeu de données volumineux.

4.7 Conclusion

Lors de la réalisation de notre système de classification des images rétiniennes, nous avons mis en évidence l'importance du choix de l'architecture de base pour l'extraction efficace des caractéristiques. L'utilisation des étapes de prétraitement des images (recadrage, remplissage,... etc.) afin d'améliorer la qualité visuelle des données. De plus, l'intégration des techniques d'ptimisation avancées, notamment l'accumulation de gradients, la précision mixte automatique (AMP), permet d'améliorer la stabilisation et la performance de l'apprentissage. En dernier, un ajustement précis des hyperparamètres tels que le nombre d'époques, taille du batch, optimiseur, fonction de perte…etc. a permis de garantir des performances fiables. Les résultats conclus ont permis d'évaluer l'efficacité et la précision de nos modèles. Nous avons présenté l'interface de notre application destinée aux utilisateurs afin de tester notre travail, avec des exemples de classification de la rétinopathie diabétique par notre application basée sur les models utiliser

CONCLUSION GÉNÉRALE ET PERSPECTIVES

La classification d'image médicale est l'un des sujets les plus importants dans le domaine de la vision par ordinateur. La Rétinopathie diabétique (RD) est une complication du diabète non traité à long terme. Les personnes atteintes de cette maladie souffrent souvent d'une perte de vision grave, car elle n'est détectée que tardivement.
L'analyse et la classification de la rétinopathie sont cruciales pour le diagnostic et le suivi de la rétinopathie diabétique, l'une des principales complications du diabète. Ce domaine suscite un intérêt croissant de la part des chercheurs. L'algorithme le plus efficace dans le domaine du traitement et de la classification d'images ces dernières années est le réseau neuronal convolutif, Avec le développement de l'intelligence artificielle et de l'apprentissage automatique. Dans le but de développer des méthodes automatisées capables de détecter et de classer avec précision les différents stades de la maladie. Traditionnellement, cette évaluation repose sur l'interprétation manuelle des images du fond d'œil par les ophtalmologistes, une tâche complexe, longue et sujette aux erreurs humaines.
L'objectif de ce projet est de développer un système de classification automatique et efficace avec une grande capacité de généralisation pour différentes images du fond d'œil en utilisant l'apprentissage profond. À cette fin, nous avons proposé trois méthodes d'apprentissage par transfert (modèles) basées sur des modèles convolutifs : DenseNet121, ViT-B/16 et YOLOv8. Ce manuscrit présente les différentes étapes de notre démarche et contributions. La première étape consiste à charger les données, suivie d'une étape de prétraitement. L'étape la plus importante consiste à choisir le bon réseau de base et les bons hyperparamètres pour obtenir un modèle robuste et précis. Enfin, la dernière étape consiste à classer les cinq (5) catégories RD. Les expériences sont menées à l'aide de deux bases de données, la première est APTOS 2019, où nous appliquons deux méthodes, en cascade sur le modèle DenseNet121 et des méthodes directes sur deux modèles ViT-B/16 et YOLOv8. Les résultats obtenus mettent en évidence l'efficacité des méthodes en cascade, notamment avec DenseNet121, qui a atteint une précision remarquable de 96,30%, dépassant les approches proposées dans la littérature. Concernant le deuxième dataset DDR les résultats obtenus sont très satisfaisants, avec une précision globale maximale de 93% sur l'ensemble des données de test. À notre connaissance, aucun travail basé sur ce dataset n'a obtenu un pourcentage aussi élevé.

Afin d'améliorer les performances obtenues, plusieurs aspects peuvent être considérés. Exploitation et intégration d'autres types de données (rapport, analyse, ...) multi-modale. L'utilisation de bases de données plus larges ou de modèles pré-entraînés sur des images médicales permettra une meilleure généralisation. Utilisation de méthodes avancées comme les modèles hybrides. Enfin, améliorer l'interface de l'application médicale interactive et ajouter des données cliniques (âge, antécédents médicaux, etc.), aussi développer une version mobile afin d'offrir un outil complet d'aide au diagnostic qui sera mis à disposition des professionnels de la santé.

Bibliographie

[5] Nicolas Vandenbroucke. « Segmentation d'images couleur par classification de pixels dans des espaces d'attributs colorimétriques adaptés : application à l'analyse d'images de football ». Thèse de doct. Lille 1, 2000.

[6] M.L Benomar. « Combinaison adaptative des informations texture et couleur pour la segmentation d'images médicales. » Thèse de doct. Université de Tlemcen-Abou Bekr Belkaid, 2018.

[10] D.LE Le Bihan. « Qu'est-ce qu'une image médicale ? Considérations médico-économiques ». In : *Bulletin de l'Académie Nationale de Médecine* 202.7 (2018), p. 1665-1678.

[16] I. Mersaoui et M Fouiri. « Etude sur l'imagerie médicale : prétraitement, segmentation et amélioration des exploitations ». In : *Mémoire de Master, Université SAAD DAHLAB de BLIDA* (2018).

[18] Semmame. *La microscopie*. Rapp. tech. Frère Mentouri Constantine1, 2020.

[20] A Dubois. « Tomographie par cohérence optique (OCT)-Technologie et applications biomédicales ». In : *Optique Photonique* (2023).

[48] Khadidja Boudjemai, Kawther Boukraa et Hakim Bendiabdallah Mohammed. « L'utilisation de services web et des réseaux de neurones pour le diagnostic médical à distance ». Mém. de mast. Université BelHadj Bouchaib, 2020.

[49] S. D. Saidj. « Techniques de NLP pour la détection des fausses nouvelles ». Mém. de mast. Université Ibn Khaldoun-Tiaret, 2022.

[51] Ludovic DE MATTEIS, S Janny, S Nathan et W Shu-Quartier. « Introduction à l'apprentissage automatique ». In : *Culture Sciences de l'Ingénieur* (2022).

[52] Roumaissa ANBERI, Fouzi BAKHTI et Reda YAGOUB. « Développement d'un système de localisation basé sur les signaux GSM ». Mém. de mast. 2024.

[55] S. Errouche et B. Merrouche. « L'apprentissage profond pour la reconnaissance des macro-expressions ». Mém. de mast. Université de Bordj Bou Arreridj, Faculty of Mathematics et Computer Science, 2024.

[60] Ahlem Ferchichi. « Propagation et réduction des incertitudes dans les modèles de changement d'occupation des sols ». Thèse de doct. Université de la Manouba, École Nationale des Sciences de l'Informatique, 2017.

[66] Ali Bechouche. « Utilisation des techniques avancées pour l'observation et la commande d'une machine asynchrone : application à une éolienne ». Thèse de doct. Université Mouloud Mammeri de Tizi-Ouzou, 2013.

[68] Yann LeCun, Léon Bottou, Yoshua Bengio et Patrick Haffner. « Gradient-based learning applied to document recognition ». In : *Proceedings of the IEEE* 86.11 (1998), p. 2278-2324.

[70] Alex Krizhevsky, Ilya Sutskever et Geoffrey E Hinton. « Imagenet classification with deep convolutional neural networks ». In : *Advances in neural information processing systems* 25 (2012), p. 1097-1105.

[71] Alexander Khvostikov, Karim Aderghal, Jenny Benois-Pineau, Andrey Krylov et Gwenaelle Catheline. « 3D CNN-based classification using sMRI and MD-DTI images for Alzheimer disease studies ». In : *arXiv preprint arXiv :1801.05968* (2018).

[72] Karen Simonyan et Andrew Zisserman. « Very deep convolutional networks for large-scale image recognition ». In : *arXiv preprint arXiv :1409.1556* (2014).

[74] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens et Zbigniew Wojna. « Rethinking the Inception Architecture for Computer Vision ». In : *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, p. 2818-2826.

[75] Tej Singh et Dinesh Kumar Vishwakarma. « A deeply coupled ConvNet for human activity recognition using dynamic and RGB images ». In : *Neural Computing and Applications* 33.1 (2021), p. 469-485.

[76] Kaiming He, Xiangyu Zhang, Shaoqing Ren et Jian Sun. « Deep residual learning for image recognition ». In : *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, p. 770-778.

[78] Gao Huang, Zhuang Liu, Laurens Van Der Maaten et Kilian Q Weinberger. « Densely Connected Convolutional Networks ». In : *arXiv preprint arXiv :1608.06993* (2016).

[79] Noha Radwan. « Leveraging sparse and dense features for reliable state estimation in urban environments ». Thèse de doct. University of Freiburg, Freiburg im Breisgau, Germany, 2019.

[80] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser et Illia Polosukhin. « Attention Is All You Need ». In : *Advances in Neural Information Processing Systems*. T. 30. Curran Associates, Inc., 2017. url : https://arxiv.org/abs/1706.03762.

[81] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly et al. « An image is worth 16x16 words : Transformers for image recognition at scale ». In : *arXiv preprint arXiv :2010.11929* (2020).

[82] Jianfang Wu, Ruo Hu, Zhenghong Xiao, Jiaxu Chen et Jingwei Liu. « Vision Transformer-based recognition of diabetic retinopathy grade ». In : *Medical Physics* 48.12 (2021), p. 7850-7863.

[83] Joseph Redmon, Santosh Divvala, Ross Girshick et Ali Farhadi. « You only look once : Unified, real-time object detection ». In : *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, p. 779-788.

[84] Michael Shenoda. « Real-time Object Detection : YOLOv1 Re-Implementation in PyTorch ». In : *arXiv preprint arXiv :2305.17786* (2023).

[92] Shu-I Pao, Hong-Zin Lin, Ke-Hung Chien, Ming-Cheng Tai, Jiann-Torng Chen et Gen-Min Lin. « Detection of diabetic retinopathy using bichannel convolutional neural network ». In : *Journal of Ophthalmology* 2020.1–8 (2020), p. 9139713.

[93] Mohamed R Shoaib, Heba M Emara, Jun Zhao, Walid El-Shafai, Naglaa F Soliman, Ahmed S Mubarak, Osama A Omer, Fathi E Abd El-Samie et Hamada Esmaiel. « Deep learning innovations in diagnosing diabetic retinopathy : The potential of transfer learning and the DiaCNN model ». In : *Computers in Biology and Medicine* 169 (2024), p. 107834.

[94] Nour Eldeen M Khalifa, Mohamed Loey, Mohamed Hamed N Taha et Hamed Nasr Eldin T Mohamed. « Deep transfer learning models for medical diabetic retinopathy detection ». In : *Acta Informatica Medica* 27.5 (2019), p. 327-332.

[95] T Aswathi, TR Swapna et S Padmavathi. « Transfer learning approach for grading of diabetic retinopathy ». In : *Journal of Physics : Conference Series*. T. 1767. 1. IOP Publishing. 2021, p. 012033.

[96] Ghadah Alwakid, Walaa Gouda, Mamoona Humayun et Noor Zaman Jhanjhi. « Deep learning-enhanced diabetic retinopathy image classification ». In : *Digital health* 9 (2023), p. 20552076231194942.

[97] Cheena Mohanty, Sakuntala Mahapatra, Biswaranjan Acharya, Fotis Kokkoras, Vassilis C Gerogiannis, Ioannis Karamitsos et Andreas Kanavos. « Using deep learning architectures for detection and classification of diabetic retinopathy ». In : *Sensors* 23.12 (2023), p. 5726.

[98] Nada Benabdessalam, Sabra Mabrouk et Faouzi Ghorbel. « Architectures CNN pour la classification de la rétinopathie diabétique : étude comparative ». In : *ResearchGate* (2023).

[99] Wejdan L Alyoubi, Maysoon F Abulkhair et Wafaa M Shalash. « Diabetic retinopathy fundus image classification and lesions localization system using deep learning ». In : *Sensors* 21.11 (2021), p. 3704.

[117] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li et Li Fei-Fei. « ImageNet : A large-scale hierarchical image database ». In : *2009 IEEE conference on computer vision and pattern recognition*. IEEE. 2009, p. 248-255.

[118] S Vinuja, Kaushek Kumar TR, R Karthika et al. « Performance analysis of diabetic retinopathy classification using cnn ». In : *2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)*. IEEE. 2021, p. 823-828.

[119] A Aruna Kumari, Avinash Bhagat et Santosh Kumar Henge. « Classification of Diabetic Retinopathy Severity Using Deep Learning Techniques on Retinal Images ». In : *Cybernetics and Systems* (2024), p. 1-25.

[120] Muwei Jian, Hongyu Chen, Chen Tao, Xiaoguang Li et Gaige Wang. « Triple-DRNet : A triple-cascade convolution neural network for diabetic retinopathy grading using fundus images ». In : *Computers in Biology and Medicine* 155 (2023), p. 106631.

[121] Pranamita Nanda et N Duraipandian. « A Novel Optimizer in Deep Neural Network for Diabetic Retinopathy Classification. » In : *Computer Systems Science & Engineering* 43.3 (2022).

[122] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh et Dhruv Batra. « Grad-CAM : Visual Explanations from Deep Networks via Gradient-Based Localization ». In : *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017, p. 618-626.

Webographie

[1] . *Imagerie Médicale*. URL : https://www.groupe-revelis.com/revelis/radiologie-medicale/imagerie-medicale/ (visité le 02/02/2025).

[2] . *Image numérique*. URL : https://fr.wikipedia.org/wiki/Image_num%C3%A9rique (visité le 02/02/2025).

[3] *Introduction au traitement d'image*. URL : https://zestedesavoir.com/tutoriels/1557/introduction-au-traitement-dimage/?page=2 (visité le 02/02/2025).

[4] . *Le traitement de l'image : de la couleur aux niveaux de gris*. URL : https://cedric-lelievre.canoprof.fr/eleve/2SNT/05%20Module_La_Photographie_Num%C3%A9rique/co/Le_traitement_de_l_image_de_la_couleur_aux_niveaux_de_gris.html (visité le 02/02/2025).

[7] . *L*a*b* CIE 1976*. URL : https://fr.wikipedia.org/wiki/L*a*b*_CIE_1976 (visité le 02/02/2025).

[8] . *Qu'est-ce que les gamma et Gamut cine (espace de couleur) ?* URL : https://neelnajaproduction.com/quest-ce-que-les-gamma-et-gamut-cine-espace-de-couleur/ (visité le 02/02/2025).

[9] . *Le codage HSL*. URL : https://web.maths.unsw.edu.au/~lafaye/CCM/video/hsl-tsl.htm (visité le 02/02/2025).

[11] . *Radiographie : principe, indications et déroulement*. URL : https://www.doctissimo.fr/html/sante/imagerie/radiographie_standard.htm (visité le 02/02/2025).

[12] . *Radiographies X*. URL : https://www.chuv.ch/fr/chuv-home/patients-et-familles/specialites-medicales/atlas-medical-thematique/examens-complementaires/radiographies-x#:~:text=La%20technologie%20de%20radiographie%20utilise,conducteurs%2C%20appel%C3%A9es%20cathode%20et%20anode. (visité le 02/02/2025).

[13] *Imagerie Médicale*. URL : https://www.cea.fr/comprendre/Pages/sante-sciences-du-vivant/essentiel-sur-imagerie-medicale.aspx (visité le 03/02/2025).

[14] . *Tout savoir sur le scanner*. URL : https://www.medimagesa.ch/scanner-explications/#:~:text=%C3%89galement%20appel%C3%A9%20tomodensitom%C3%A9trie%20(TDM)%2C,organes%2C%20les%20tissus%2C%20etc. (visité le 03/02/2025).

[15] . *Imagerie par résonance magnétique (IRM)*. URL : https://www.msdmanuals.com/fr/accueil/sujets-particuliers/examens-d-imagerie-courants/imagerie-par-r%C3%A9sonance-magn%C3%A9tique-irm (visité le 03/02/2025).

[17] . *Echographie : principe, indication, déroulé, coût*. URL : https://www.doctissimo.fr/html/sante/imagerie/echographie.htm (visité le 03/02/2025).

[19] *Fiche explicative de la leçon : Microscopie*. URL : https://www.nagwa.com/fr/explainers/356107286405/ (visité le 04/02/2025).

[21] . *La tomographie par cohérence optique (OCT)*. URL : https://www.edmundoptics.fr/knowledge-center/application-notes/optics/optical-coherence-tomography/ (visité le 04/02/2025).

[22] . *Tomographie en Cohérence Optique (OCT)*. URL : https://hugobourdon.com/examens/oct/ (visité le 04/02/2025).

[23] . *Rétinopathie diabétique*. URL : https://www.eyepedia.ch/en/post/diabetische-retinopathie? (visité le 04/02/2025).

[24] . *Anatomie de l'œil*. URL : https://dr-leininger.fr/loeil-et-la-vision/anatomie-de-loeil (visité le 04/02/2025).

[25] . *Anatomie du globe oculaire*. URL : https://ophtalmologie-provence.fr/activite/anatomie/ (visité le 04/02/2025).

[26] . *Comment se déroule un fond d'œil ?* URL : https://www.ameli.fr/assure/sante/examen/exploration/deroulement-fond-oeil (visité le 04/02/2025).

[27] . *Bien choisir son ophtalmoscope*. URL : https://www.pharma-gdd.com/fr/bien-choisir-son-ophtalmoscope (visité le 04/02/2025).

[28] . *Biomicroscopie. Rédigé par le webmestre*. URL : https://www.inflamoeil.org/l-oeil/examens/article/biomicroscopies (visité le 04/02/2025).

[29] . *Cataracte*. URL : https://www.15-20.fr/offre-de-soins/maladies-de-loeil/cataracte/ (visité le 07/02/2025).

[30] . *Cataracte : Symptômes, causes et traitements*. URL : https://cliniquebellevue.com/pathologies/cataractes/ (visité le 07/02/2025).

[31] . *Glaucome*. URL : https://www.15-20.fr/offre-de-soins/maladies-de-loeil/glaucome/ (visité le 07/02/2025).

[32] . *Le glaucome à angle fermé*. URL : https://www.allaboutvision.com/fr-fr/maladies/glaucome-a-angle-ferme/ (visité le 07/02/2025).

[33] . *Maladies infectieuses de l'œil*. URL : https://www.15-20.fr/offre-de-soins/maladies-de-loeil/maladies-infectieuses-de-l%c2%92%c2%9cil/ (visité le 07/02/2025).

[34] . *Rétinopathie diabétique*. URL : https://www.15-20.fr/offre-de-soins/maladies-de-loeil/retinopathie-diabetique/ (visité le 07/02/2025).

[35] . *Diabetic Retinopathy 224x224 (2019)*. URL : https://www.kaggle.com/datasets/sovitrath/diabetic-retinopathy-224x224-2019-data (visité le 07/02/2025).

[36] . *Types de diabète, qu'est-ce que le diabète ?* URL : https://www.federationdesdiabetiques.org/information/diabete (visité le 07/02/2025).

[37] . *Rétinopathie diabétique*. URL : https://www.barraquer.com/fr/pathologie/retinopathie-diabetique (visité le 08/02/2025).

[38] . *La rétinopathie diabétique : definition, symptomes, traitement*. URL : https://www.sciencesetavenir.fr/sante/ophtalmo/retinopathie-diabetique-definition-symptomes-traitement_102338 (visité le 08/02/2025).

[39] . *Rétinopathie diabétique - Troubles oculaires*. URL : https://www.msdmanuals.com/fr/professional/troubles-oculaires/pathologies-de-la-r%C3%A9tine/r%C3%A9tinopathie-diab%C3%A9tique#Diagnostic_v957376_fr (visité le 08/02/2025).

[40] . *Rétinopathie diabétique-Les symptômes*. URL : https://www.braille.be/fr/retinopathie-diabetique (visité le 08/02/2025).

[41] . *Sémiologie oculaire : le fond d'œil*. URL : https://couf.fr/wp-content/uploads/2021/06/Chapitre-1_2021.pdf (visité le 08/02/2025).

[42] . *Rétinopathie diabétique-Le traitement*. URL : https://www.braille.be/fr/retinopathie-diabetique (visité le 08/02/2025).

[43] . *La rétinopathie diabétique*. URL : https://www.ophtacenter.fr/pathologies/retinopathie-diabetique/ (visité le 08/02/2025).

[44] . *Intelligence artificielle*. URL : https://fr.wikipedia.org/wiki/Intelligence_artificielle (visité le 01/03/2025).

[45] . *Notions de base sur l'apprentissage profond*. URL : https://rpubs.com/alexvezeau/SCI1035-CH1 (visité le 01/03/2025).

[46] . *Système expert*. URL : https://fr.wikipedia.org/wiki/Syst%C3%A8me_expert (visité le 01/03/2025).

[47] . *Langage de programmation - Intelligence artificielle (IA) - Système expert*. URL : https://www.gladir.com/CODER/IA/systeme-expert.htm (visité le 01/03/2025).

[50] . *Supervised Learning*. URL : https://sciences24.com/%D8%A7%D9%84%D8%AA%D8%B9%D9%84%D9%85-%D8%A7%D9%84%D8%AE%D8%A7%D8%B6%D8%B9-%D9%84%D9%84%D8%A5%D8%B4%D8%B1%D8%A7%D9%81/ (visité le 01/03/2025).

[53] . *Linear regression*. URL : https://sciences24.com/%d8%a7%d9%84%d8%a5%d9%86%d8%ad%d8%af%d8%a7%d8%b1-%d8%a7%d9%84%d8%ae%d8%b7%d9%8a-linear-regression/ (visité le 01/03/2025).

[54] . *Logistic Regression Classifier*. URL : https://sciences24.com/%d8%a7%d9%84%d8%a7%d9%86%d8%ad%d8%af%d8%a7%d8%b1-%d8%a7%d9%84%d9%84%d9%88%d8%ac%d8%b3%d8%aa%d9%8a/ (visité le 01/03/2025).

[56] . *Unspervised Learning*. URL : https://sciences24.com/%D8%A7%D9%84%D8%AA%D8%B9%D9%84%D9%85-%D8%A7%D9%84%D8%A2%D9%84%D9%8A-%D8%BA%D9%8A%D8%B1-%D8%A7%D9%84%D8%AE%D8%A7%D8%B6%D8%B9-%D9%84%D9%84%D8%A5%D8%B4%D8%B1%D8%A7%D9%81-unsupervised-ml/ (visité le 01/03/2025).

[57] . *Qu'est-ce que l'apprentissage non supervisé ?* URL : https://fr.linedata.com/quest-ce-que-lapprentissage-non-supervise (visité le 02/03/2025).

[58] . *Semi-supervised ML*. URL : https://sciences24.com/1-%D8%A3%D9%86%D9%88%D8%A7%D8%B9-%D8%AA%D8%B9%D9%84%D9%85-%D8%A7%D9%84%D8%A2%D9%84%D8%A9/ (visité le 02/03/2025).

[59] . *Deep Learning Spreads*. URL : https://semiengineering.com/deep-learning-spreads/ (visité le 02/03/2025).

[61] . *Réseau neuronal convolutif*. URL : https://fr.wikipedia.org/wiki/R%C3%A9seau_neuronal_convolutif (visité le 02/03/2025).

[62] . *Les réseaux de neurones convolutifs*. URL : https://www.datasciencetoday.net/index.php/fr/deep-learning/173-les-reseaux-de-neurones-convolutifs (visité le 02/03/2025).

[63] . *Mieux comprendre le Deep Learning appliqué à la reconnaissance d'images*. URL : https://www.devoteam.com/fr/expert-view/mieux-comprendre-le-deep-learning-applique-a-la-reconnaissance-dimages/ (visité le 02/03/2025).

[64] . *CNN : Couche de Convolution*. URL : https://inside-machinelearning.com/cnn-couche-de-convolution/ (visité le 02/03/2025).

[65] . *Fonction d'activation*. URL : https://fr.wikipedia.org/wiki/Fonction_d%27activation (visité le 02/03/2025).

[67] . *Mish vs ReLU : Quelle est la meilleure fonction d'activation ?* URL : https://penseeartificielle.fr/mish-vs-relu-meilleure-fonction-activation/ (visité le 02/03/2025).

[69] . *LeNet-5*. URL : https://datahacker.rs/deep-learning-lenet-5-architecture/ (visité le 02/03/2025).

[73] . *An Overview of VGG16 and NIN Models*. URL : https://lekhuyen.medium.com/an-overview-of-vgg16-and-nin-models-96e4bf398484 (visité le 03/03/2025).

[77] . *Understanding ResNet : A Milestone in Deep Learning and Image Recognition*. URL : https://www.ikomia.ai/blog/mastering-resnet-deep-learning-image-recognition (visité le 03/03/2025).

[85] .*Transfer Learning : What is it ?* URL : https://datascientest.com/en/transfer-learning-what-is-it (visité le 03/03/2025).

[86] . *Qu'est-ce que le surajustement ?* URL : https://www.ibm.com/fr-fr/think/topics/overfitting (visité le 03/03/2025).

[87] . *Overfitting & Underfitting Concepts - Interview Questions*. URL : https://vitalflux.com/overfitting-underfitting-concepts-interview-questions/ (visité le 03/03/2025).

[88] .*Qu'est-ce que la Classification Binaire ?– Meilleur Guide*. URL : https://inside-machinelearning.com/classification-binaire/ (visité le 03/03/2025).

[89] . *Predicting Movie Genres using NLP - A Multi-Label Classification Model*. URL : https://www.analyticsvidhya.com/blog/2019/04/predicting-movie-genres-nlp-multi-label-classification/ (visité le 03/03/2025).

[90] . *Qu'est-ce que la Classification Multi-Classes ? – Meilleur Guide*. URL : https://inside-machinelearning.com/classification-multi-classes/ (visité le 03/03/2025).

[91] . *Multi-label classification*. URL : https://en.wikipedia.org/wiki/Multi-label_classification (visité le 03/03/2025).

[100] . *Définition Google Colaboratory*. URL : https://www.wildcodeschool.com/lexique/google-colaboratory (visité le 25/04/2025).

[101] . *Kaggle : Tout ce qu'il faut savoir sur cette plateforme*. URL : https://datascientest.com/kaggle-tout-ce-quil-a-savoir-sur-cette-plateforme (visité le 25/04/2025).

[102] . *Python (langage)*. URL : https://fr.wikipedia.org/wiki/Python_(langage)#Utilisation (visité le 25/04/2025).

[103] . *TensorFlow*. URL : https://www.databricks.com/fr/glossary/tensorflow-guide (visité le 25/04/2025).

[104] . *TensorFlow logo.svg*. URL : https://en.m.wikipedia.org/wiki/File:TensorFlow_logo.svg (visité le 25/04/2025).

[105] . *keras*. URL : https://keras.io/ (visité le 25/04/2025).

[106] . *logo keras*. URL : https://penseeartificielle.fr/tp-reconnaissance-faciale/logo-keras/ (visité le 25/04/2025).

[107] . *PyTorch*. URL : https://fr.wikipedia.org/wiki/PyTorch#Historique (visité le 25/04/2025).

[108] . *PyTorch : tout savoir sur le framework de Deep Learning de Facebook*. URL : https://datascientest.com/pytorch-tout-savoir (visité le 25/04/2025).

[109] . *NumPy documentation*. URL : https://numpy.org/doc/stable/ (visité le 25/04/2025).

[110] . *NumPy logo 2020.svg*. URL : https://en.m.wikipedia.org/wiki/File:NumPy_logo_2020.svg (visité le 25/04/2025).

[111] . *Scikit-learn*. URL : https://fr.wikipedia.org/wiki/Scikit-learn (visité le 25/04/2025).

[112] . *Matplotlib*. URL : https://fr.wikipedia.org/wiki/Matplotlib (visité le 25/04/2025).

[113] . *Matplotlib – Python Library*. URL : https://studyopedia.com/python-libraries/matplotlib-library/#google_vignette (visité le 25/04/2025).

[114] *.kaggle*. URL : https://www.kaggle.com/ (visité le 05/05/2025).

[115] . *Diabetic Retinopathy 224x224 Gaussian Filtered*. URL : https://www.kaggle.com/datasets/sovitrath/diabetic-retinopathy-224x224-gaussian-filtered (visité le 05/05/2025).

[116] . *DDR dataset*. URL : https://www.kaggle.com/datasets/mariaherrerot/ddrdataset (visité le 05/05/2025).