Classification Of Diabetic Retinopathy by Using a Deep Learning Model

Milong Irene Niqaise
Université Libre de Bruxelles (ULB)
PROJ H419
Encadré par: Prof. Olivier Debeir
Email: *[Irene.milong@ulb.be]*

***Abstract*—La rétinopathie diabétique (RD) est une complication grave et parfois chronique induite par une hyperglycémie prolongée, caractéristique du diabète sucré. Elle affecte la microvascularisation rétinienne, entraînant une altération de la barrière hémato-rétinienne, une hypoperfusion locale et une ischémie tissulaire. Cette atteinte vasculaire se manifeste par des micro-anévrismes, des hémorragies, des exsudats lipidiques et dans les cas avancés, par la prolifération de néovaisseaux fragiles pouvant entraîner un décollement de la rétine. Sur le plan clinique et histopathologique, on distingue deux stades évolutifs : la rétinopathie diabétique non proliférante (RDNP)et la rétinopathie diabétique proliférante (RDP).Elle peut engendrer des complications sévères telles qu'un décollement de la rétine ou une hémorragie intravitréenne. La RD est l'une des principales causes de cécité évitable chez l'adulte.Dans la science médicale moderne, les images sont le moyen le plus utilisé pour un diagnostic précis de maladie.De nos jours, la vision par ordinateur avec des réseaux neuronaux profonds permet d'entrainer un modéle et avoir un niveau de précision élevée. Dans notre étude, Les images utilisées proviennent du dataset "Diabetic Retinopathy (resized)" de Kaggle, dérivé de la compétition EyePACS 2015. Elles ont été acquises par photographie du fond d'œil à l'aide de caméras rétiniennes couleur, puis redimensionnées pour être compatibles avec les modèles d'apprentissage automatique. Le but de ce rapport est de proposer un modéle de Deep Learning pour identifier des antécédants de RD. Nous avons choisis un XGboost et un modéle préentrainé basé sur les réseaux de neurones avancés ResNet50 pour identifier la classe appropriée de gravité des images de rétinopathie diabétique.**

I. Introduction

Le diabète constitue une véritable épidémie mondiale, avec une prévalence particulièrement élevée dans certains pays comme l'Inde, la Chine et les États-Unis. D'après les estimations les plus récentes, plus de 422 millions de personnes sont actuellement touchées dans le monde, un chiffre en forte augmentation ces dernières décennies. Parmi les nombreuses complications chroniques liées au diabète, la rétinopathie diabétique (RD) représente l'une des principales causes de perte de vision irréversible chez l'adulte.

La rétinopathie diabétique progresse à travers quatres stades : Le niveau 1 qui est la rétinopathie légére non proliférative, le niveau 2 qui est la rétinopathie non proliférative modérée, le niveau 3 qui est la rétinopathie non proliférative sévére et le niveau 4 qui est la rétinopathie diabétique proliférative et représente le niveau le plus avancé de la maladie.

Dans ce contexte, le dépistage systématique par analyse d'images du fond d'œil (imagerie du fondus) est fortement recommandé. Cependant, face à l'augmentation massive des cas de diabète, le recours à des systèmes automatisés d'aide au diagnostic devient indispensable pour assister les ophtalmologistes dans la détection précoce des lésions rétiniennes. L'apprentissage profond (deep learning), notamment les réseaux de neurones, se sont imposés comme une approche prometteuse pour l'analyse des images rétiniennes. Ces modèles sont capables d'apprendre automatiquement des caractéristiques discriminantes à partir d'images couleur du fond d'œil, facilitant ainsi l'identification des signes précoces de la RD, tels que les micro-anévrismes, les exsudats et les hémorragies.

Ce travail s'inscrit dans cette dynamique, avec pour objectif le développement d'un modèle automatisé performant basé sur des techniques d'apprentissage profond, pour améliorer le dépistage et le suivi de la rétinopathie diabétique à partir d'images issues de bases de données publiques.

II. TRAVAIL PROPOSE

Etant donné la taille du dataset que nous avons soit plus de 35000 images, nous avons effectué des prétraitements essentielles pour éviter tous bias et mauvaises performances des modéles utilisés. Nous l'avons fait en commençant pas reduire notre dataset à un nombre équivalent pour tous les niveaux de retinopathie, pour cela la reduction a été faite en attribuant aux autres classes le meme nombre d'images que celui de la classe moins représentée qui est la classe 4 avec 708 images. Nous avons donc travaillés avec 3540 images. Une autre approche que nous avons utilisés est l'extraction des features personnalisées afin de pouvoir offrir au modéle une meilleure exploration. Les images sont entraînées avec modéle de réseau neuronal et un modéle d'apprentissage supervisé basé sur les arbres de décision.

III. Travail Proposé

Etape 1 : SOUS-ECHANTILLONAGE

Etape 2 : Division du nouveau dataset en Train,Val et Test.

Etape 3 : Preparation du Generateur avec Data Augmentation

Etape 4 : Extraction des features avec ResNet50 et Extraction des features classiques des images.

Etape 5 : Etudes des features extraites avec ResNet50 pour ne garder que les plus importantes et les envoyer au modéle.

Etape 6 : Application du modele Hybride avec Features extraites et Image(XGboost), Fine Tuning ResNet50 de CNN.

Etapes 7 : Evaluation des performances des modéles.

IV. Methodologie

Dans cette section, nous détaillons la méthodologie suivie pour traiter les données et construire le modèle de prédiction. La méthodologie se divise en trois grandes étapes : sous-échantillonnage, extraction des caractéristiques (features), et construction du modèle de prédiction.

A. Sous-Echantillonnage

Le sous-échantillonnage est une technique utilisée pour équilibrer la distribution des classes dans les données. Dans le contexte de notre projet, nous avons observé que certaines classes étaient surreprésentées par rapport à d'autres, ce qui pouvait entraîner un biais dans le modèle. Pour remédier à cela, nous avons appliqué une méthode de sous-échantillonnage, visant à réduire le nombre d'exemples dans les classes majoritaires tout en maintenant un ensemble de données représentatif. Cette approche nous permet de renforcer la capacité du modèle à apprendre de manière équilibrée les caractéristiques des différentes classes, sans être influencé de manière disproportionnée par celles qui sont dominantes.

Afin de corriger le déséquilibre des classes dans notre dataset, nous avons appliqué une méthode de sous-échantillonnage aléatoire (random undersampling). Cette technique consiste à réduire le nombre d'exemples dans les classes majoritaires en en sélectionnant un sous-ensemble aléatoire, de manière à obtenir un nombre équivalent d'éléments dans chaque classe.

Concrètement, nous avons procédé en regroupant les données par classe, puis en échantillonnant aléatoirement un nombre fixe d'exemples (708) pour chaque classe. Cela permet d'assurer une distribution équilibrée tout en limitant l'impact des classes surreprésentées sur l'apprentissage du modèle. Cette méthode simple mais efficace permet de prévenir les biais dans la prédiction, bien qu'elle présente l'inconvénient potentiel de perdre des informations en excluant une partie des données disponibles.

B. Extraction des Features

L'extraction des caractéristiques constitue une étape clé dans le pipeline de classification d'images. Elle permet de transformer les données visuelles en vecteurs numériques exploitables par les algorithmes de machine learning. Dans ce projet, nous avons adopté une approche hybride combinant des caractéristiques profondes issues d'un modèle de convolution pré-entraîné (ResNet-50), avec des descripteurs classiques capturant des propriétés statistiques, texturales et morphologiques des images.

1) Caractéristiques profondes issues de ResNet-50: Nous avons utilisé le modèle ResNet-50, un réseau de neurones convolutifs (CNN) à 50 couches introduit par He et al. (2015), reconnu pour sa capacité à extraire des représentations de haut niveau à partir d'images complexes. Ce modèle repose sur le concept de "résidus" avec des connexions de saut (skip connections) qui permettent de faciliter l'apprentissage en profondeur en évitant le problème de disparition du gradient.

Dans notre approche, ResNet-50 a été utilisé en tant que extracteur de features pré-entraîné sur ImageNet, en supprimant la couche de classification finale. Chaque image a été propagée à travers le réseau, et le vecteur de sortie de la dernière couche convolutionnelle globale (Global Average Pooling) a été extrait. Ce vecteur est de dimension 2048.Ces features capturent des informations de haut niveau telles que les formes, les textures, les couleurs dominantes et les motifs structurels présents dans les images, tout en restant invariants à certaines transformations comme la rotation ou le redimensionnement.

2) Descripteurs classiques: En complément des features profonds, nous avons extrait plusieurs caractéristiques classiques afin de fournir au modèle des informations additionnelles issues de l'analyse d'images traditionnelles. Ces features se répartissent en trois catégories principales :

Caractéristiques texturales (GLCM) :

Entropy, energy, contrast, homogeneity, variance locale $\rightarrow$ issues de la matrice de co-occurrence des niveaux de gris (GLCM), elles décrivent la distribution spatiale des pixels dans l'image.

Statistiques de couleur (par canal RGB) :

Moyenne, écart-type, kurtosis (aplatissement), couleur dominante $\rightarrow$ elles permettent de caractériser la couleur globale et les distributions statistiques des intensités dans chaque canal.

Caractéristiques morphologiques des lésions :

Surface, périmètre, et nombre d'éléments pour deux types de lésions : microaneurysmes et exsudats $\rightarrow$ elles fournissent une description quantitative des anomalies détectées dans les images du fond d'œil. Ces descripteurs sont essentiels pour capter des signaux cliniquement pertinents, notamment dans le cadre de la détection de pathologies telles que la rétinopathie diabétique.

Les différentes familles de caractéristiques ont été concaténées pour former un vecteur unique représentant chaque image. Cette fusion permet au modèle de tirer parti à la fois de la puissance des représentations apprises automatiquement par le deep learning et de l'interprétabilité des descripteurs classiques. Elle renforce ainsi la capacité discriminante de l'ensemble des données utilisées pour l'apprentissage supervisé.

C. Modele de Prediction

1) Étude exploratoire avec un classificateur aléatoire: Avant de recourir à des modèles supervisés complexes, nous avons utilisé un classificateur aléatoire pour évaluer l'informativité intrinsèque des caractéristiques extraites, indépendamment d'un algorithme prédictif précis. Ce modèle attribue une classe aléatoire à chaque observation, sans apprentissage réel, mais il peut être utilisé dans une analyse permutationnelle de l'importance des variables.

Dans ce cadre, chaque feature est perturbée de manière aléatoire, et l'impact de cette perturbation sur la performance

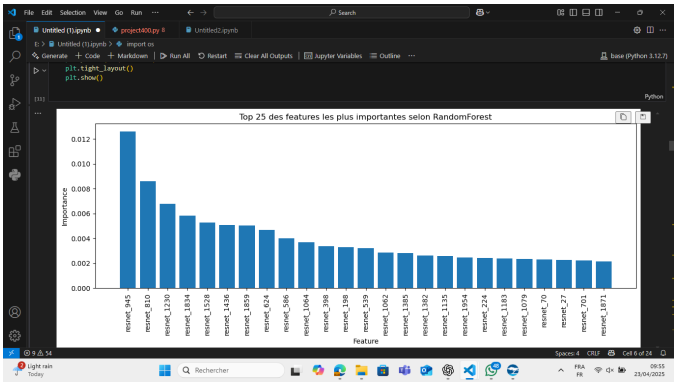


Fig. 1: 25 features ResNet les plus importantes

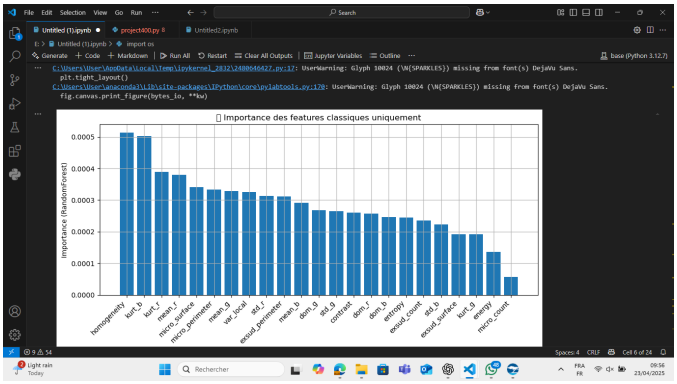


Fig. 2: Importance des features classiques extraites

du classificateur est mesuré. Une dégradation notable des performances suite à la permutation d'une variable suggère que celle-ci joue un rôle important dans la structure des données, même dans un cadre non supervisé strict.

Cette approche permet :d'identifier les features les plus discriminantes,de réduire éventuellement la dimensionnalité du problème,et de guider le choix des variables à inclure dans les modèles prédictifs ultérieurs.

2) Modéle supervisé : XGBoost: Suite à cette analyse, pour la tâche de classification nous avons utilisé l'algorithme XGBoost (Extreme Gradient Boosting), une méthode d'ensemble reposant sur la technique du gradient boosting appliquée aux arbres de décision. Cet algorithme est particulièrement réputé pour sa robustesse, sa capacité à gérer les données hétérogènes, et son efficacité en termes de performance, tant en précision qu'en vitesse d'exécution.

3) Principe du Gradient Boosting: Le gradient boosting est une méthode d'ensemble séquentielle qui construit un modèle fort en combinant de nombreux modèles faibles, typiquement des arbres de décision peu profonds. À chaque itération, le nouvel arbre est entraîné pour corriger les erreurs du modèle précédent, en se basant sur le gradient de la fonction de perte. Autrement dit, le modèle apprend à prédire les résidus (ou erreurs) des prédictions précédentes, en optimisant la fonction de coût.

4) Fonctionnement de XGBoost: XGBoost améliore le gradient boosting traditionnel par plusieurs avancées clés :

Optimisation par second ordre : contrairement au gradient boosting classique qui n'utilise que le premier ordre (le gradient), XGBoost utilise également la hessienne (dérivée seconde) de la fonction de perte pour une meilleure direction de descente.

Régularisation explicite : XGBoost intègre une pénalisation directe de la complexité des arbres dans la fonction de perte (par les termes λ et α)), ce qui réduit le surapprentissage (overfitting). Cela se traduit par une fonction objectif régulière :

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k)$$

$$\text{où } \Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_j w_j^2$$

Pruning (élagage) : au lieu de construire un arbre en profondeur puis de l'élaguer, XGBoost stoppe la croissance prématurément selon un gain de perte minimum.

Traitement efficace des valeurs manquantes : XGBoost apprend la meilleure direction à prendre en présence de valeurs absentes pendant l'entraînement, et l'applique à la prédiction.

Parallélisation et cache mémoire optimisé : contrairement à d'autres implémentations de boosting, XGBoost permet une construction parallèle des arbres, réduisant le temps d'exécution.

5) Justification de l'utilisation: Nous avons choisi XGBoost en raison de sa capacité à : gérer un grand nombre de caractéristiques, y compris les features issues du deep learning (ResNet-50),modéliser des relations complexes non linéaires entre les variables,offrir un bon compromis biais-variance,et fournir une importance relative des variables en sortie, utile pour l'interprétation.

6) Implémentation du modéle: Afin d'implémenter le modèle XGBoost, nous avons construit une matrice de caractéristiques,rassemblant différentes sources d'information visuelle, dérivées des images d'entrée. Cette représentation vise à capturer à la fois des informations bas niveau interprétables et des représentations profondes non linéaires issues du réseau neuronal convolutionnel. Avant l'entraînement du modèle XGBoost, un pipeline de prétraitement a été appliqué aux données afin d'assurer leur qualité et leur cohérence. Ce pipeline comprenait les étapes suivantes :

Séparation des variables explicatives et de la cible : Le jeu de données a été divisé en :

X-tab: matrice des features tabulaires (concaténation des features ResNet, des descripteurs classiques, et des caractéristiques lésionnelles), Y-tab : variable cible correspondant au niveau de rétinopathie (0 à 4).

Traitement des valeurs manquantes :

Les variables contenant des valeurs manquantes ont été identifiées,

Une imputation par la médiane a été appliquée pour les colonnes concernées afin de limiter l'influence des outliers tout en conservant la distribution des données.

Standardisation des features : Les variables de X-tab présentent des échelles hétérogènes (e.g., intensité lumineuse, entropie, dimensions morphologiques, valeurs de ResNet). Afin d'assurer une comparabilité des variables et d'optimiser la convergence du modèle, toutes les colonnes ont été standardisées selon la transformation : $x' = \frac{x-\mu}{\sigma}$ où μ et σsont respectivement la moyenne et l'écart-type calculés sur l'ensemble du jeu d'entraînement.

Découpage du jeu de données : Une stratification a été appliquée lors du découpage en jeux d'entraînement et de test, afin de préserver la distribution des classes dans les deux sous-ensembles.

Les hyperparamètres principaux utilisés dans l'entraînement sont :

n-estimators = 100 : nombre d'arbres,
max-depth = 6 : profondeur maximale des arbres,
learning-rate = 0.1 : taux d'apprentissage,

D. Fine-Tuning de ResNet50 avec PyTorch

Dans le cadre de ce projet, en complément au modéle XGBoost le modèle `ResNet50` a été utilisé pour la classification des niveaux de rétinopathie. Ce modèle pré-entraîné sur le dataset ImageNet a été fine-tuné pour s'adapter à notre problème spécifique, en réajustant ses poids pour mieux prédire les classes de rétinopathie à partir des images. Le `ResNet50` est un réseau de neurones convolutifs très profond, appartenant à la famille des *Residual Networks* (ResNet), qui se distingue par l'utilisation de connexions résiduelles permettant de faciliter l'apprentissage de réseaux très profonds.

a) Fonctionnement de ResNet50: Le modèle `ResNet50` se compose de 50 couches profondes et utilise des blocs résiduels pour permettre une transmission plus fluide de l'information à travers ces couches. Voici une explication des principales couches et de leur rôle dans le réseau :

- **Blocs résiduels** : Les blocs résiduels ajoutent des connexions directes entre l'entrée et la sortie de chaque bloc, ce qui permet de lutter contre le problème du *vanishing gradient* dans les réseaux très profonds. Cela permet d'améliorer l'efficacité du réseau et d'empêcher la perte d'information au fur et à mesure que le réseau devient plus profond.
- **Couches de convolution** : Ces couches appliquent des filtres pour détecter des motifs locaux dans les images (bords, textures). Elles jouent un rôle clé dans l'extraction des caractéristiques visuelles.
- **Batch Normalization** : Chaque couche de convolution est suivie d'une couche de normalisation pour réduire la variance interne du réseau, ce qui améliore la stabilité et la vitesse de convergence pendant l'entraînement.
- **Activation ReLU** : Après chaque opération de convolution, la fonction d'activation ReLU (Rectified Linear Unit) est appliquée pour introduire une non-linéarité dans le modèle. Cela permet au réseau de mieux modéliser des relations complexes.
- **Pooling (MaxPooling)** : Le pooling est utilisé pour réduire la dimensionnalité des cartes de caractéristiques tout en conservant les informations essentielles. Cela permet également de réduire le nombre de paramètres du modèle, améliorant ainsi son efficacité.
- **Couches denses** : Après la phase d'extraction de caractéristiques, les couches denses sont utilisées pour effectuer la classification. La sortie est obtenue par une couche `Softmax` qui donne les probabilités d'appartenance aux différentes classes.

b) Fine-Tuning de ResNet50: Le fine-tuning consiste à réajuster un modèle pré-entraîné (ici, `ResNet50`) sur un jeu de données spécifique. Le modèle `ResNet50` préalablement entraîné sur ImageNet est modifié pour mieux correspondre à notre tâche de classification des images de rétinopathie. Le processus de fine-tuning permet de tirer parti des connaissances préalablement acquises par le modèle sur un grand nombre d'images, tout en l'adaptant spécifiquement à notre problème.

Les étapes du fine-tuning sont les suivantes :

1) **Geler certaines couches** : Les premières couches du réseau, qui sont responsables de l'extraction des caractéristiques générales des images (par exemple, les bords, les textures), sont gelées, c'est-à-dire qu'elles ne seront pas mises à jour pendant l'entraînement. Cela permet de conserver les connaissances acquises lors de l'entraînement sur ImageNet.
2) **Adapter la dernière couche** : La dernière couche du réseau, qui correspond à la classification des classes d'ImageNet, est remplacée par une nouvelle couche correspondant au nombre de classes de notre propre dataset (c'est-à-dire, les différents niveaux de rétinopathie).
3) **Entraîner les couches restantes** : Seules les couches profondes du réseau, responsables de la classification des caractéristiques spécifiques à notre tâche, seront ajustées pendant l'entraînement.
4) **Ajustement du taux d'apprentissage** : Un taux d'apprentissage plus faible est utilisé pour fine-tuner les couches pré-entraînées afin de ne pas perturber trop brusquement les poids déjà appris sur ImageNet.

c) Implémentation avec PyTorch: L'implémentation du fine-tuning de `ResNet50` a été réalisée avec la bibliothèque PyTorch. Le modèle a été chargé avec les poids pré-entraînés sur ImageNet et la dernière couche a été modifiée pour s'adapter à notre problème de classification des niveaux de rétinopathie.Nous avons utilisé les `DataLoader` de PyTorch pour charger efficacement nos jeux de données d'entraînement et de validation. Le `DataLoader` est une classe de PyTorch qui permet de charger et de traiter des batches de données à partir de notre dataset de manière optimisée. Cela est particulièrement important lorsque l'on travaille avec des ensembles de données volumineux ou des images.

d) Fonctionnement du DataLoader: Le `DataLoader` dans PyTorch est utilisé pour itérer sur un dataset, diviser ce dataset en batches, et effectuer des transformations sur les données comme des augmentations d'images(important pour les dataset de petites taille),une normalisation pendant l'entraînement. Il fournit une manière flexible et efficace de traiter les données pendant l'entraînement du modèle.

Voici les principales étapes pour configurer un `DataLoader` :

- **Chargement des données** : Nous utilisons un objet `Dataset` qui représente notre jeu de données. Le `Dataset` contient les images ainsi que leurs étiquettes associées. Ce dernier peut être facilement créé à partir de la classe `ImageFolder` ou en définissant un objet personnalisé hérité de `torch.utils.data.Dataset`.
- **Création du DataLoader** : Une fois le dataset défini, nous utilisons la classe `DataLoader` pour itérer sur le dataset en batchs. Le DataLoader permet également d'effectuer un *shuffling* (mélange des données) pour éviter que le modèle n'apprenne des ordres spécifiques de données et surcharger certaines régions des données.
- **Batching** : Le DataLoader divise automatiquement les données en petits lots ou *batches*, ce qui permet de traiter les données plus efficacement en mémoire. Chaque fois que le DataLoader est utilisé, il renvoie un batch d'images et les étiquettes correspondantes.
- **Transformations sur les données** : Le DataLoader peut également appliquer des transformations sur les images à la volée, comme la mise à l'échelle, la normalisation, ou l'augmentation des données (par exemple, en effectuant une rotation ou une mise à l'échelle aléatoire pour améliorer la robustesse du modèle).

e) Avantages du Fine-Tuning: Le fine-tuning de `ResNet50` présente plusieurs avantages pour notre tâche de classification des images de rétinopathie :

- **Amélioration des performances** : Le fine-tuning permet d'exploiter la capacité de `ResNet50` à extraire des caractéristiques pertinentes à partir des images, ce qui améliore les performances de classification, en particulier avec des jeux de données plus petits.
- **Réduction du besoin de données étiquetées** : En utilisant un modèle pré-entraîné, le fine-tuning permet de réduire la nécessité d'une grande quantité de données étiquetées, ce qui est particulièrement utile dans des domaines comme la médecine, où les jeux de données annotées peuvent être rares.
- **Convergence plus rapide** : Le fine-tuning accélère le processus d'entraînement en réutilisant les poids déjà appris par le modèle sur des données génériques (ImageNet).

V. Resultats et Discussions

A. Resultats XGBoost

Le modèle XGBoost a été évalué sur les données d'images de rétinopathie à partir des features d'images extraites

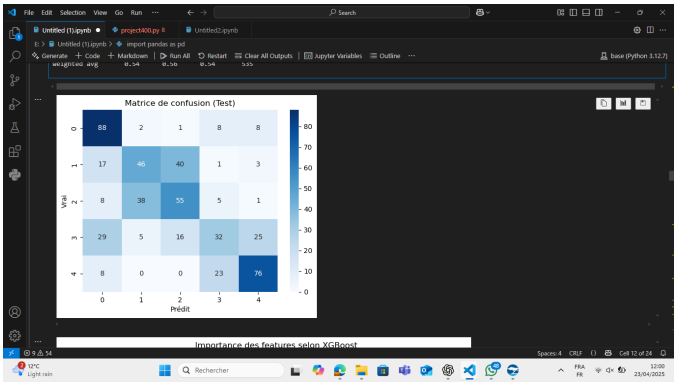


Fig. 3: Matrice de confusion du modèle XGBoost sur le jeu de test

(ResNet50, descripteurs classiques et lésionnels). Les performances globales obtenues sont les suivantes :

- **Accuracy** : 56%
- **F1-score macro** : 0.54
- **Précision moyenne** : 0.56
- **Recall moyen** : 0.56

Le modèle obtient de très bonnes performances sur les classes 0 (aucune rétinopathie) et 4 (forme sévère), avec des F1-scores respectifs de 0.66 et 0.73. En revanche, la classe 3 est mal détectée (F1 = 0.40), ce qui indique une difficulté à bien séparer les formes modérées.

a) Test Set ::

- **Accuracy** : 56%
- **F1-score macro** : 0.54
- **F1-score pondéré** : 0.54

Les performances sur le jeu de test confirment les observations faites sur le jeu de validation. La classe 0 (absence de maladie) est bien prédite (F1 = 0.68), tout comme la classe 4 (F1 = 0.69). Cependant, la classe 3 (formes intermédiaires de rétinopathie) est prédite avec plus de difficulté (F1 = 0.36), traduisant potentiellement une ambiguïté dans ses caractéristiques visuelles.

La **matrice de confusion** (figure 3) met en évidence les classes les mieux prédictibles ainsi que celles qui sont fréquemment confondues. Les niveaux avancés de la maladie (par exemple 3 et 4) sont globalement mieux identifiés, tandis que les cas précoces (niveau 1) sont parfois confondus avec l'absence de pathologie (niveau 0), ce qui reflète la difficulté clinique du diagnostic précoce.

Un arbre de décision issu du modèle (figure 4) a été visualisé afin d'illustrer le processus de prise de décision basé sur les features extraits.

B. Discussions XGBoost

Malgré des performances satisfaisantes, certaines limitations doivent être considérées :

- **Taille de l'échantillon** : le sous-échantillonnage à taille fixe par classe limite la représentativité de la population réelle.

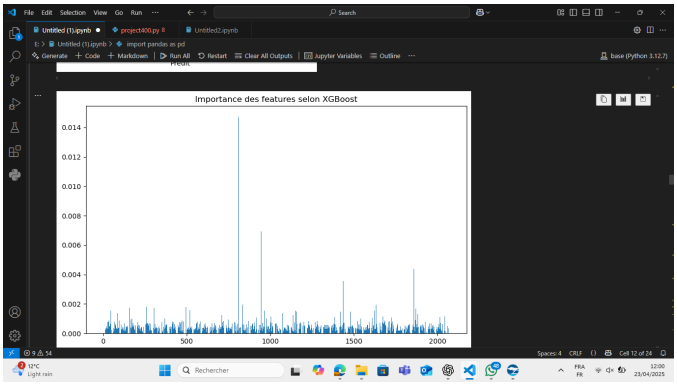


Fig. 4: Exemple d'arbre extrait du modèle XGBoost

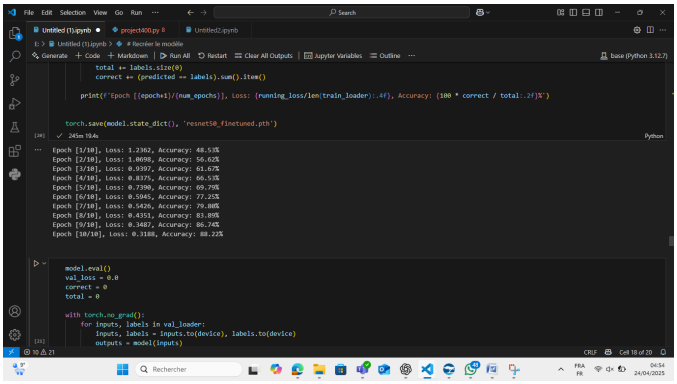


Fig. 5: Evoultion de la Loss et L'accuracy

- **Déséquilibre entre les classes** : certaines classes cliniques sont naturellement moins fréquentes, ce qui peut affecter la précision de leur détection.
- **Qualité des annotations** : les labels d'origine peuvent contenir des incertitudes cliniques, pouvant biaiser l'apprentissage supervisé.
- **Complexité du modèle** : XGBoost, bien que performant, reste difficile à interpréter pour un clinicien sans visualisation dédiée.

C. Performances du modèle ResNet50

ResNet50 est particulièrement performant grâce à sa profondeur (50 couches) et à l'utilisation de connexions résiduelles qui permettent d'éviter la dégradation des performances lors de l'ajout de couches supplémentaires.

Les données (figure 7) indiquent une convergence stable du modèle, démontrant la capacité du modèle à extraire efficacement des caractéristiques discriminantes à partir des images.

En fin d'entrainement, le modéle présente une Val Loss: 0.5437, val Accuracy: 85.95% ce qui montre que le modéle a su généraliser sur des données qu'il ne connaissait pas.

a) Évaluation sur le jeu de test: Le score F1 équilibré entre les classes, démontrant une meilleure capacité de généralisation.

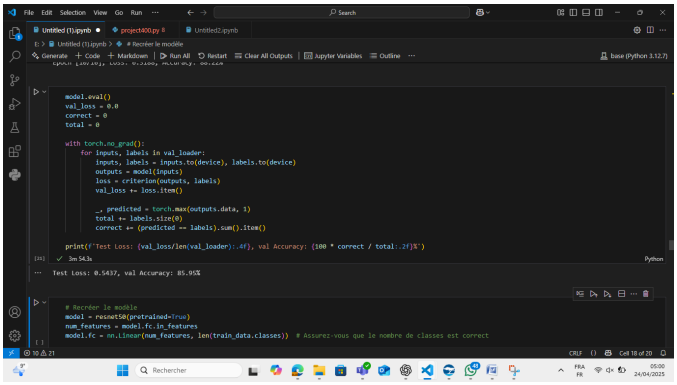


Fig. 6: Val loss et Val Accuracy

Classe	Précision	Rappel	F1-score
0	0.83	0.97	0.90
1	0.91	0.70	0.79
2	0.78	0.87	0.82
3	0.83	0.88	0.85
4	0.94	0.83	0.89
Macro Moyenne	0.71	0.70	0.70
Accuracy		0.73	

TABLE I: Résultats de ResNet50 sur l'ensemble de test

b) Comparaison entre modèles: ResNet50 démontre les meilleures performances globales, avec une bonne généralisation sur le test. Toutefois, cela se fait au prix d'un temps d'entraînement plus long et d'une consommation mémoire accrue.

D. Discussions : ResNet50

- ResNet50 nécessite des ressources computationnelles importantes (GPU fortement recommandé) pour un entraînement rapide et efficace.

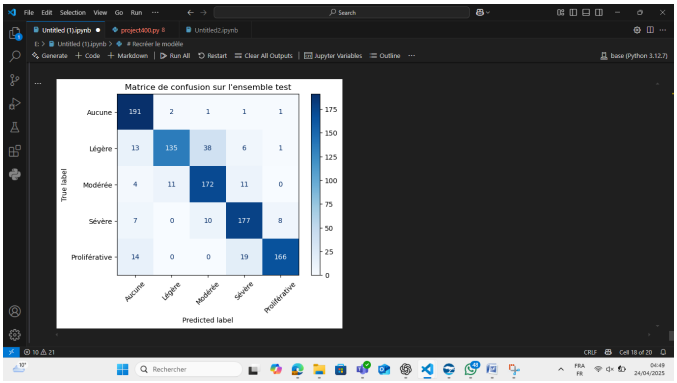
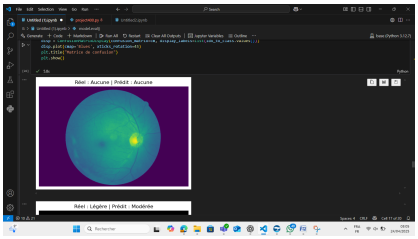


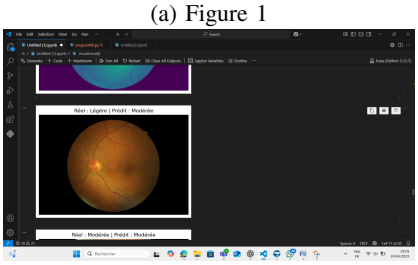
Fig. 7: Matrice de Confusion pour les données de test

Modèle	Accuracy (Test)	Macro F1	Training Time
XGBoost	0.56	0.54	Faible
ResNet50	0.85	0.85	Élevé

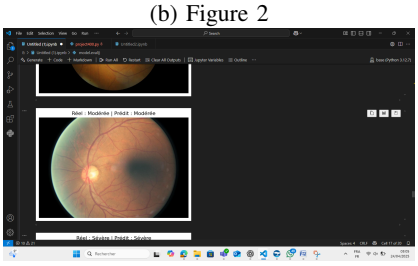
TABLE II: Comparaison des modèles sur les données de rétinopathie



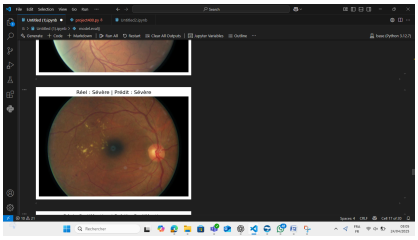
(a) Figure 1



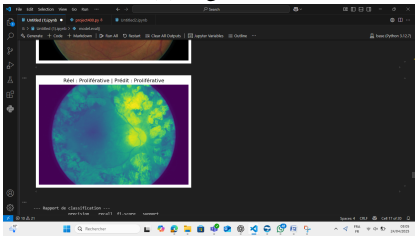
(b) Figure 2



(c) Figure 3



(d) Figure 4



(e) Figure 5

Fig. 8: Prédictions faites par le modéle sur quelques images de chaque classe.

- Le modèle est également plus sensible à un surapprentissage si le dataset n'est pas suffisamment riche ou bien prétraité.
- L'interprétabilité des décisions prises par le modèle est limitée, ce qui peut poser problème dans un contexte médical.

Les résultats obtenus confirment l'efficacité de la combinaison de features visuelles profondes (ResNet) et classiques (texture, couleur, lésions) dans la détection automatique de la rétinopathie. XGBoost, en tant que modèle robuste et régularisé, s'est révélé adapté à ce type de données hétérogènes.

Toutefois, la confusion observée entre les niveaux 0 et 1 souligne l'importance du prétraitement des images, et peut suggérer l'intérêt de méthodes de classification hiérarchique ou d'intégration de données cliniques supplémentaires (âge, durée du diabète, etc.).

L'utilisation de ResNet50 dans notre étude met en lumière l'intérêt des architectures profondes pour la classification d'images médicales. Sa capacité à extraire des caractéristiques visuelles de haut niveau permet une classification plus fine. Toutefois, en pratique, la sélection du modèle doit prendre en compte :

- Le compromis entre performance et temps de calcul.
- Les contraintes matérielles.
- L'importance de l'interprétabilité selon le domaine d'application.

L'amélioration future peut inclure l'ajout de techniques de data augmentation avancée, le fine-tuning de couches profondes, ou l'intégration de techniques de visualisation comme Grad-CAM pour justifier les prédictions du modèle.

VI. Conclusion

Ce rapport propose un modèle optimal pour la détection de la rétinopathie diabétique à partir d'images médicales. Le traitement des images de rétinopathie est crucial pour extraire des caractéristiques pertinentes et permettre une classification fiable. L'utilisation de XGBoost et du fine-tuning de ResNet50 a permis de démontrer l'importance de l'extraction de bonnes caractéristiques et de l'adaptation des modèles à la tâche spécifique de classification.

Le modèle XGBoost, basé sur des caractéristiques extraites à la fois par ResNet50 et des méthodes classiques, a montré des performances intéressantes mais a été limité par la qualité des données d'entrée, en particulier pour les images bruitées. En revanche, l'utilisation de ResNet50 avec fine-tuning a permis de tirer parti de l'apprentissage par transfert pour adapter le modèle aux spécificités des images médicales, offrant ainsi des résultats plus robustes et une meilleure capacité à capturer des relations complexes dans les données.

Cependant, le fine-tuning de ResNet50 a nécessité une grande puissance de calcul, ce qui pourrait limiter son utilisation dans des environnements à ressources limitées. Les performances du modèle pourraient être encore améliorées avec l'utilisation de GPU, permettant ainsi de traiter plus de

données et d'accélérer l'entraînement. De plus, l'intégration d'approches hybrides, telles que la combinaison des forces de plusieurs modèles, pourrait améliorer la précision et la robustesse du système.

Pour les travaux futurs, l'application de modèles plus complexes, comme les réseaux U-Net pour la segmentation des images, et l'augmentation de la quantité de données traitées pourraient contribuer à améliorer davantage la précision de la détection de la rétinopathie diabétique. Une application autonome pour l'identification des images de rétinopathie serait également bénéfique, en facilitant l'intégration du modèle dans les systèmes de dépistage existants et en améliorant l'efficacité des soins oculaires.

VII. Reference

file:///C:/Users/PATRICE/Downloads/2.ClassificationofDiabeticRetinopathyImagesbyUsingDeepLe....pdf

https://www.kaggle.com/datasets/tanlikesmath/diabetic-retinopathy-resized

APTOS 2019 Blindness Detection Competition

https://www.kaggle.com/competitions/aptos2019-blindness-detection

Lien vers mon code sur Github

Lien vers la base de données

Lien vers la dataset sous echantillonnée et prétraitée