RAPPORT D'ALTERNANCE RECHERCHE - S6, S7

Intelligence Artificielle pour l'aide au diagnostic médical à partir d'IRM cardiaques

Esteban CARLIN
Tuteur d'alternance : Mr BOURENNANE Salah
Période d'alternance : du 06/02/2023 au 31/01/2024

Contents

1 Remerciements

Merci à Salah Bourennane de m'avoir accompagné dans cette découverte de la recherche en IA et en particulier en Deep Learning. J'ai beaucoup appris et cela a décidément façonné mon avenir professionnel.

2 Résumé

J'ai eu l'opportunité d'être encadré et accompagné pour apprendre le métier de chercheur. Sur mes 2 semestres d'alternance, j'ai passé le premier à me documenter et à comprendre le fonctionnement et les enjeux du Deep Learning. J'ai ensuite passé le semestre suivant à implémenter ma propre IA en code python permettant de diagnostiquer des malformations / insuffisances cardiaques. L'objectif étant mon apprentissage et la maîtrise des techniques employées, plutôt que d'aller chercher les derniers modèles de réseaux de neurones à la pointe de l'innovation, j'ai pris le parti de travailler avec un outil connu et reconnu : le réseau U-Net 3D [2].

3 Abstract

I had the opportunity to be supervised and accompanied as I learned the job of a researcher. During my 2 semesters of work-study, I spent the first one researching and understanding how Deep Learning works and what's at stake. I then spent the following semester implementing my own AI in Python code to diagnose heart malformations/failures. As my objective was to learn and master the techniques used, rather than go looking for the latest neural network models at the cutting edge of innovation, I decided to work with a well-known and recognized tool: the U-Net 3D network [2].

4 Mots-clefs

- Alternance
- Alternance Recherche
- Centrale Méditerranée
- Centrale Marseille
- Institut Fresnel
- Intelligence Artificielle
- Deep Learning
- U-Net
- Machine Learning
- IRM Cardiaque

- Segmentation d'images
- Classification
- Réseau de neurones
- Python
- PyTorch

5 Glossaire

IA : Intelligence Artficielle
IRM : Imagerie par Résonance Magnétique
LV : Left Ventricule - Ventricule Gauche
RV : Right Ventricule - Ventricule Droit
Myo : Myocarde
CNN : Convolutional Neural Networks - Réseau Neuronal Convolutif
ACDC : Automated Cardiac Diagnosis Challenge
NOR : Patient normal
MINF : Patient ayant déjà subi un infarctus de myocarde
DCM : Patient atteint de cardiomyopathie dilatée
HCM : Patient atteint de cardiomyopathie hypertrophique
AR : Patient présentant une anomalie du ventricule droit

6 A propos de l'Institut Fresnel

Situé à Marseille, l'Institut Fresnel offre une expertise dans les domaines de l'optique, de l'électromagnétisme et de l'ingénierie optique. L'institut a été créé en 1979 et a été nommé en hommage à Augustin-Jean Fresnel, un physicien français du XIXe siècle qui a contribué de manière significative au domaine de l'optique. L'objectif principal de l'Institut Fresnel est de réaliser des études fondamentales et appliquées dans les secteurs de l'optique, de l'électromagnétisme et des techniques pertinentes. Les objectifs de ces études sont de dépasser les limites de la connaissance dans ces domaines et de créer des applications technologiques créatives.

7 Objectifs et organisation de l'alternance

- Semestre 6
  - Documentation sur la segmentation d'images en Machine Learning
  - Documentation sur la bibliothèque python "PyTorch"
  - Mise en place du squelette du code et du projet
- Semestre 7
  - Réaliser un algorithme fonctionnel
  - Analyse des performances, optimisation et validation du système mis en place

8 Partie théorique

8.1 Réseaux de neurones

Les réseaux de neurones sont des algorithmes informatiques conçus en s'inspirant du peu que l'on comprend du fonctionnement du cerveau humain, afin de résoudre des tâches complexes. Ces modèles sont une sous-catégorie de l'intelligence artificielle appelée apprentissage profond (Deep Learning).

8.1.1 Structure

- Entrées (Inputs) : Un neurone artificiel reçoit des signaux d'entrée provenant d'autres neurones ou d'une source externe. Chaque signal est multiplié par un poids spécifique, qui représente l'importance relative de cette entrée pour le neurone.

- Somme pondérée : Les signaux pondérés sont ensuite sommés. Cette somme pondérée est utilisée pour déterminer l'activité du neurone. Mathématiquement, la somme pondérée (S) est calculée comme la somme des produits des entrées (x) par leurs poids correspondants (w), avec un terme de biais (b) ajouté :

  $$S = \sum_{i=1}^{n} (x_i \cdot w_i) + b$$

  où n est le nombre d'entrées.

- Fonction d'Activation : On applique ensuite une fonction d'activation à la somme pondérée. Cette fonction introduit de la non-linéarité dans le modèle, permettant au réseau de neurones d'apprendre des relations complexes dans les données.

  $$Output = f(S)$$

  Où $f()$ est la fonction d'activation. Des fonctions couramment utilisées sont la fonction sigmoïde ou la fonction ReLU (Rectified Linear Unit).

  $$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

  où : e est la base du logarithme naturel (environ 2.71828). et

  $$\text{ReLU}(x) = \max(0, x)$$

- Sortie : La sortie du neurone, généralement désignée comme l'activation, est utilisée comme signal d'entrée pour les neurones de la couche suivante dans un réseau de neurones ou alors comme sortie du réseau s'il n'y a pas de couches supplémentaires.

[5]

8.1.2 Optimisation

Correction et amélioration des modèles :

- Propagation avant (expliquée précédemment) :
  - Les données d'entrée sont introduites dans le réseau et propagées à travers les différentes couches jusqu'à la couche de sortie.
  - Les poids des connexions et les biais sont utilisés pour calculer la sortie du réseau.

$$\hat{y} = f(W \cdot x)$$

- Calcul de l'erreur :
  - La sortie prédite du réseau est comparée à la sortie réelle (le label) à l'aide d'une fonction de coût.
  - La fonction de perte mesure à quel point la prédiction du réseau diffère de la réponse attendue.

$$J = Loss(\hat{y}, y)$$

- Rétropropagation du gradient : C'est un algorithme clé utilisé pour entraîner les réseaux de neurones. C'est un processus itératif qui permet au réseau d'ajuster ses poids (w) en fonction des erreurs de prédiction.
  - L'objectif est donc de minimiser la fonction de coût en ajustant les poids et les biais.
  - Le gradient de la fonction de coût par rapport aux poids et aux biais est calculé à l'aide des dérivées partielles. Cela donne une indication de la "direction" dans laquelle les poids doivent être modifiés pour minimiser la perte.

$$\frac{\partial J}{\partial W} = \frac{\partial J}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial W}$$

  - Les gradients sont propagés en sens inverse à travers le réseau, d'où le terme "rétropropagation". Chaque couche contribue au calcul des gradients de la couche précédente.

- Mise à jour des poids (w) :
  - Les poids et les biais sont mis à jour en utilisant une méthode d'optimisation, généralement la descente de gradient.
  - La mise à jour des poids est proportionnelle au produit du gradient et d'un taux d'apprentissage (learning rate). Cela permet de contrôler la taille des ajustements effectués à chaque itération.

$$W_{\text{new}} = W_{\text{old}} - \alpha \cdot \frac{\partial J}{\partial W}$$

    Où α est le taux d'apprentissage, un (hyper)paramètre ajustant la taille des pas de mise à jour des poids.

- Répétition du processus : les étapes sont répétées sur plusieurs itérations (epochs) jusqu'à ce que le modèle atteigne une performance satisfaisante sur les données d'entraînement.

- Validation et Test : Pendant et/ou après l'entraînement, le modèle est évalué sur des données de validation et de test pour s'assurer de sa capacité à généraliser à de nouvelles données.

[9]

8.2 Réseau Neuronal Convolutif (CNN)

Les Convolutional Neural Networks (CNN) sont des architectures de réseaux de neurones qui ont été spécifiquement conçues pour résoudre des problèmes liés à la computer vision, en particulier la classification et la segmentation d'images. Ces réseaux ont démontré une efficacité significative dans la reconnaissance de motifs complexes au sein de données visuelles structurées. Leur conception est caractérisée par l'incorporation de couches de convolution, de pooling et des couches "fully connected", couches de neurones "classiques", créant ainsi une structure adaptée à la nature spatiale des données visuelles.

- Convolution :
  - La couche de convolution est fondamentale dans les CNN. Elle implique l'application de filtres, ou kernels, à des régions locales de l'image.
  - Ces filtres permettent l'extraction de caractéristiques spécifiques, telles que des textures, des contours, ou d'autres motifs visuels, en mettant en évidence des relations spatiales locales.
- Pooling :
  - Les couches de pooling succèdent souvent à la convolution, réduisant la dimension spatiale de la sortie.
  - Cette opération vise à conserver les caractéristiques significatives tout en diminuant la complexité computationnelle du réseau.
- Couches Fully Connected :
  - Les caractéristiques extraites par les couches précédentes sont propagées à des couches fully connected pour la classification finale.
  - Ces couches utilisent des poids pour combiner les caractéristiques, générant ainsi une sortie adaptée à la tâche spécifique, souvent la classification d'objets dans l'image.
- Activation : La sortie du réseau est souvent traitée avec une fonction softmax. C'est une fonction d'activation qui transforme un vecteur de nombres réels en une distribution de probabilité. La formule mathématique de la fonction softmax pour un vecteur $\mathbf{z}$ de dimension K est la suivante :

$$\text{softmax}(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

où : $\text{softmax}(\mathbf{z})_i$ est la i-ème composante du vecteur résultant après l'application de la fonction softmax, e est la base du logarithme naturel (environ 2.71828), z_i est la i-ème

composante du vecteur d'entrée $\mathbf{z}$, $\sum_{j=1}^{K} e^{z_j}$ est la somme exponentielle sur toutes les composantes du vecteur $\mathbf{z}$. Cette fonction attribue une probabilité à chaque élément du vecteur d'entrée, de sorte que la somme de toutes les probabilités soit égale à 1. Elle est couramment utilisée à la sortie d'un réseau de neurones pour convertir les scores bruts en probabilités, facilitant ainsi l'interprétation des résultats, notamment dans des tâches de classification.

Les CNN exploitent une architecture en couches profondes, permettant une abstraction hiérarchique des caractéristiques. Les couches initiales détectent des motifs simples, tandis que les couches plus profondes agrègent ces motifs pour représenter des structures plus complexes. Cette hiérarchie est particulièrement avantageuse pour des tâches comme la segmentation d'image, où la détection de caractéristiques à différentes échelles spatiales est cruciale. [3]

8.3 Segmentation d'images

La segmentation d'images est un processus fondamental de computer vision qui vise à diviser une image en zones homogènes où chaque zone représente un objet ou une caractéristique distincte. Afin de faciliter leur identification et leur analyse, cette tâche complexe nécessite une délimitation précise des contours des objets présents dans l'image. La segmentation d'image est largement utilisée dans divers domaines, notamment la reconnaissance d'objets, l'analyse d'images médicales et la cartographie. Les deux principales catégories de techniques de segmentation sont la segmentation sémantique et la segmentation d'instance. La première vise à classer chaque pixel de l'image en fonction de sa classe (par exemple, route, arbre, ciel), tandis que la seconde vise à identifier et isoler chaque instance individuelle d'objet.
La segmentation peut être effectuée à l'aide d'une variété de méthodes, allant des méthodes traditionnelles basées sur la détection de contours aux méthodes plus récentes qui utilisent des réseaux de neurones convolutionnels (CNN). Ces derniers ont montré une grande efficacité en raison de leur capacité à apprendre des représentations hiérarchiques complexes des données, ce qui permet une segmentation plus précise. [8]

8.4 Réseau U-Net

Afin de réaliser la segmentation, j'ai utilisé le modèle U-Net. Ce dernier, présenté par Ronneberger et al. [4] en 2015, représente une architecture de réseau de neurones convolutionnels (CNN) spécialement adaptée aux tâches de segmentation d'images biomédicales. La structure de cette architecture innovante ressemble à une lettre "U", avec des chemins de convolution en descente et des chemins de déconvolution en remontée. Pour la segmentation, où l'objectif est de classer chaque pixel d'une image en fonction de sa classe, le modèle U-Net s'est avéré efficace.
Voici son architecture :

- Encoder (Chemin de Convolution en Descente) : Le chemin de convolution en descente est responsable de la réduction de la résolution spatiale tout en extrayant des caractéristiques significatives de l'image.

Il est composé de blocs de convolution avec des couches de normalisation (comme la normalisation par lots) et des activations non linéaires (ReLU) pour apprendre des représentations hiérarchiques.

- Bridge (Pont) : Au bas de la structure en "U", un pont connecte le chemin de convolution en descente au chemin de déconvolution en remontée.
  Ce pont permet au modèle U-Net de capturer des caractéristiques à différentes échelles et de maintenir des informations contextuelles importantes.

- Decoder (Chemin de Déconvolution en Remontée) : Le chemin de déconvolution en remontée est responsable de l'augmentation de la résolution spatiale tout en effectuant une segmentation pixel par pixel.
  Il est composé de blocs de déconvolution (transposée) qui agrandissent progressivement la résolution spatiale de l'image.

- Connexions de Saut (Skip Connections) : L'élément clé du modèle U-Net réside dans les connexions de saut qui relient les caractéristiques extraites dans le chemin de convolution en descente aux étapes correspondantes dans le chemin de déconvolution en remontée.
  Ces connexions de saut permettent de conserver des informations détaillées tout au long du processus, facilitant ainsi la localisation précise des objets dans l'image.

- Fonction d'Activation et Sortie :** Les couches de sortie du chemin de déconvolution en remontée utilisent généralement une fonction d'activation softmax pour produire une carte de segmentation où chaque pixel est associé à une probabilité d'appartenir à une classe spécifique.

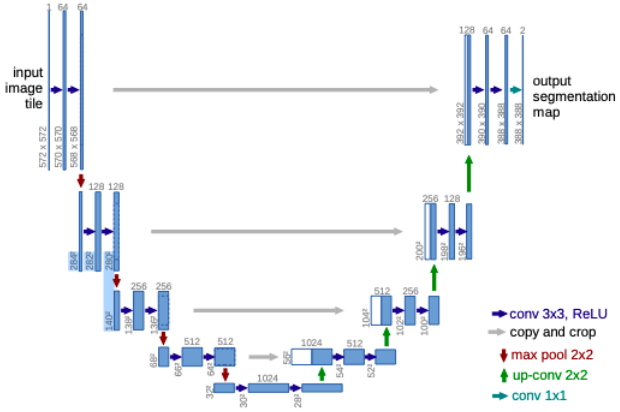

Figure 1: Un exemple des transformations dimensionnelles appliquées à une image pour de la segmentation à l'aide du réseau U-Net [4]

8.5 Classification

En intelligence artificielle (IA), la classification est un processus qui attribue des étiquettes ou des catégories préétablies à des données en fonction de leurs caractéristiques. Ce concept repose sur l'utilisation de modèles d'apprentissage automatique tels que les réseaux de

neurones, les machines à vecteurs de support (SVM) ou les arbres de décision pour analyser les caractéristiques extraites des données d'entrée.
Le processus de classification via des réseaux de neurones fonctionne de la même manière qu'un réseau de neurones typique, avec un vecteur qui contient autant de composants que de classes. Afin d'estimer la probabilité que cette image appartienne à cette classe, chaque composante a souvent une valeur volontairement comprise entre 0 et 1 ; par exemple, une photo de chien devrait renvoyer une forte probabilité pour la classe chien dans le vecteur. Il est courant d'utiliser la fonction softmax mentionnée précédemment.
La performance d'un modèle de classification est souvent mesurée à l'aide de métriques telles que la précision, le rappel, et la F1-score, qui évaluent la capacité du modèle à attribuer correctement des étiquettes aux données.

- Précision (Precision) : La précision mesure la proportion d'instances positives correctement prédites par le modèle parmi toutes les instances prédites comme positives. Elle est définie comme suit :

$$\text{Précision} = \frac{\text{Vrais Positifs}}{\text{Vrais Positifs} + \text{Faux Positifs}}$$

- Rappel (Recall) : Le rappel, aussi appelé sensibilité, mesure la proportion d'instances positives correctement prédites par le modèle parmi toutes les instances réellement positives. Il est défini comme suit :

$$\text{Rappel} = \frac{\text{Vrais Positifs}}{\text{Vrais Positifs} + \text{Faux Négatifs}}$$

- F1-score : La F1-score est une métrique qui combine à la fois la précision et le rappel en une seule valeur. Elle est définie comme la moyenne harmonique de la précision et du rappel, et permet d'équilibrer ces deux aspects de la performance du modèle. La formule est la suivante :

$$\text{F1-score} = \frac{2 \cdot \text{Précision} \cdot \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

[7]

8.6 Implémentation Python

Voici la liste des principales bibliothèques Python utilisées

- os : La bibliothèque d'exploitation Python offre des fonctionnalités pour interagir avec le système d'exploitation. Elle offre des outils pour accéder à des informations spécifiques au système d'exploitation et pour effectuer des opérations telles que la manipulation de fichiers, la navigation dans la structure des répertoires ou la vérification d'informations système.

- NumPy :
  La bibliothèque NumPy de Python est essentielle pour la manipulation efficace de tableaux multidimensionnels, car elle offre un large éventail de fonctions pour effectuer des opérations mathématiques complexes. NumPy est largement utilisé en

science des données, en apprentissage automatique et dans d'autres domaines nécessitant des calculs numériques intensifs en raison de ses performances optimisées. Elle offre un ensemble d'outils puissants pour travailler avec des tableaux, des matrices et des opérations mathématiques, ce qui facilite la résolution efficace et concise de problèmes liés à la manipulation et à l'analyse de données numériques.

- Scikit-Learn :
  La bibliothèque scikit-learn (sklearn) de Python est un outil essentiel pour les tâches d'exploration de données et d'apprentissage automatique. Elle propose une variété d'algorithmes pour la classification, la régression, le clustering et la sélection de modèles, ainsi que des utilitaires pour la préparation des données et l'évaluation des modèles. Le développement et la mise en œuvre d'algorithmes d'apprentissage automatique dans diverses applications est facilité par l'interface simple et cohérente de scikit-learn.

- Matplotlib.pyplot :
  La bibliothèque matplotlib.pyplot de Python est un module de visualisation de données qui fournit de nombreux outils pour créer divers graphiques, diagrammes et visualisations. Elle permet de générer des représentations visuelles de données pour analyser les tendances, communiquer les résultats et fournir un aperçu graphique de l'information. Matplotlib.pyplot est largement utilisé dans le domaine de l'analyse de données, de la recherche scientifique et de la création de graphiques pour la publication en raison de sa syntaxe simple et facile à comprendre.

- NiBabel :
  La bibliothèque NiBabel en Python est spécialement conçue pour manipuler et lire des données neuroimagerie stockées dans divers formats de fichiers, comme NIfTI et ANALYZE. Elle fournit les capacités nécessaires pour effectuer des opérations de prétraitement sur des données médicales et accéder et les interpréter. Dans le domaine de la neuroimagerie, NiBabel est largement utilisé pour faciliter l'analyse et la visualisation des images cérébrales, permettant aux chercheurs et aux professionnels de la santé de travailler efficacement avec des données complexes d'imagerie médicale.

- PyTorch :
  PyTorch est une bibliothèque open-source qui se concentre sur l'apprentissage et le calcul numérique avancés. Elle est particulièrement prisée dans le domaine de l'apprentissage automatique et de l'intelligence artificielle. Les graphes de calcul peuvent être construits dynamiquement tout au long du programme grâce à l'approche dynamique de PyTorch. Les deux principaux éléments de PyTorch sont les tenseurs et l'autograd, qui sont responsables de son fonctionnement global. Les tenseurs sont des structures de données similaires aux tableaux multidimensionnels, et l'autograd (calcul automatique du gradient) permet le suivi automatique des opérations effectuées sur les tenseurs, facilitant ainsi l'implémentation d'algorithmes d'optimisation tels que la rétropropagation. De plus, PyTorch propose un ensemble de classes et de modules pour la création, l'entraînement et le déploiement de réseaux de neurones, y compris les classes "nn.Module" pour définir l'architecture du modèle et les classes "nn.Linear" et "nn.Conv2d", etc. Avec des outils comme "DataLoader" pour le chargement par lots et des fonctionnalités pour le transfert simple de modèles entre CPU et GPU,

la bibliothèque facilite la gestion des données. PyTorch a gagné en popularité grâce à sa flexibilité, à sa facilité d'utilisation et au fait qu'il est largement utilisé dans le domaine de l'apprentissage automatique.

9 Base de données utilisée

Le ACDC Challenge (Automated Cardiac Diagnosis Challenge) est une initiative qui visait à promouvoir la recherche et le développement de méthodes automatisées pour l'analyse des images par résonance magnétique (IRM) cardiaques. L'ACDC Challenge s'est concentré spécifiquement sur l'évaluation automatisée des pathologies cardiaques à partir d'images IRM, en mettant l'accent sur la segmentation précise des différentes structures cardiaques. C'est avec la Base de Données de cette compétition que j'ai travaillé.

- Types d'images : La base de données ACDC Challenge comprend 300 images d'IRM cardiaques, 2 images par patient et 150 patients. Nous avons 2 images par patient car chacune correspond à une phase du cycle cardiaque: la phase de systole, la contraction et la phase de diastole, la détente.
- Pathologies cardiaques : Les images de la base de données incluent des cas équiproportionnels présentant diverses pathologies cardiaques: la cardiomyopathie dilatée (DCM), la cardiomyopathie hypertrophique (HCM), la myocardite (MINF), et la cardiomyopathie ischémique (AR). Ainsi, en considérant qu'il y a également les patients sains (NOR), nous avons 5 catégories avec 30 patients chacune.
- Annotations (labels) : Chaque image est annotée avec des contours précis des différentes structures cardiaques, notamment les ventricules gauche et droit, le myocarde et le péricarde. Les annotations (segmentations correctes attendues de l'IA) ont été réalisées à la main par 2 médecins de l'Hôpital Universitaire de Dijon.

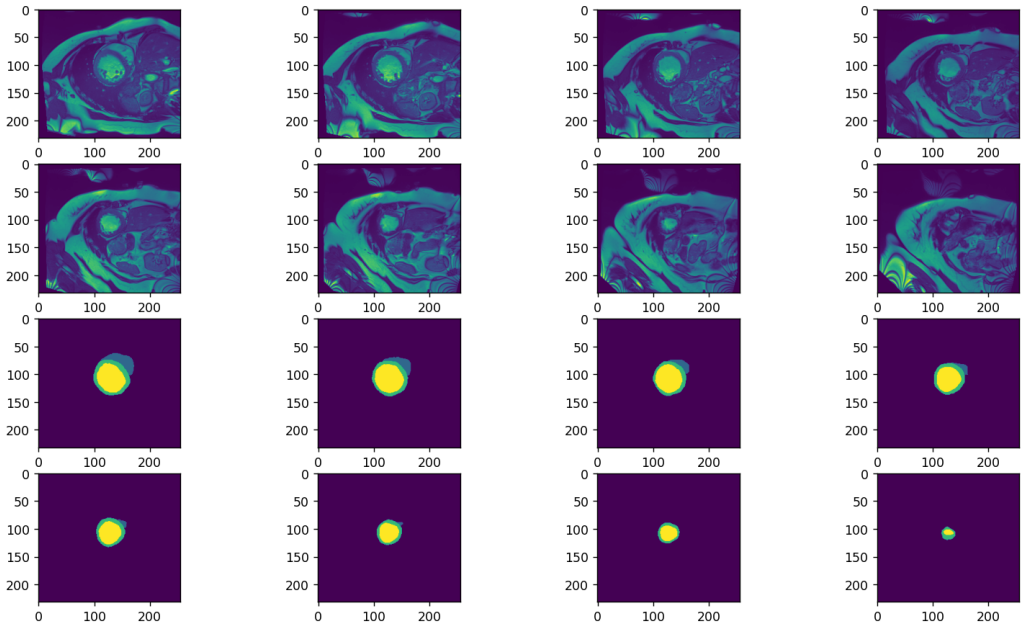

Figure 2: Plusieurs tranches d'un des IRM ainsi que la segmentation attendue par l'IA à mettre en place

10 Approche choisie pour atteindre les objectifs

Ainsi, le système final visé ressemble à cela :

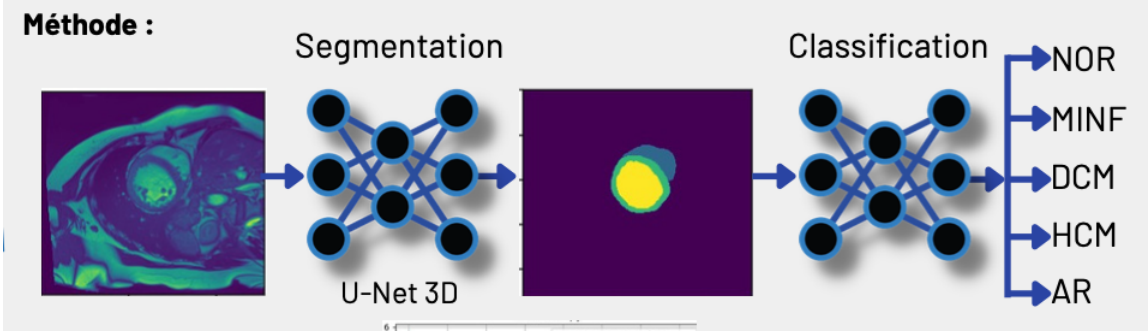

Figure 3: Schéma du système final afin de prédire les malformations cardiaques

Une fois nos IRM récupérés, il suffirait de les transmettre à notre système pour avoir une prédiction d'une quelconque malformation cardiaque :

1. Premièrement, il faut segmenter les IRM. Pour cela nous entraînons les paramètres de notre modèle U-Net 3D. Nous pouvons désormais savoir quel pixel correspond à quelle partie du coeur, toujours avec une approximation prise en compte et optimisée.

2. A partir de cela, nous pouvons extraire des informations qui permettent de prédire les pathologies cardiaques telles que :

   - Le volume de chaque ventricule
   - L'épaisseur du myocarde
   - La variation de volume de chaque ventricule entre les phases de contraction de détente
   - Le ratio volumique entre les 2 ventricules

3. Une fois toutes ces données mises sous forme de vecteur, nous avons alors autant de vecteurs que d'IRM et leurs composantes correspondes aux estimation physique. Il suffit ensuite de réaliser de la classification d'images, nous entraînons un modèle tel que VGG-Net qui est un modèle fiable pour des classifications à peu de classes (5 ici) [6].

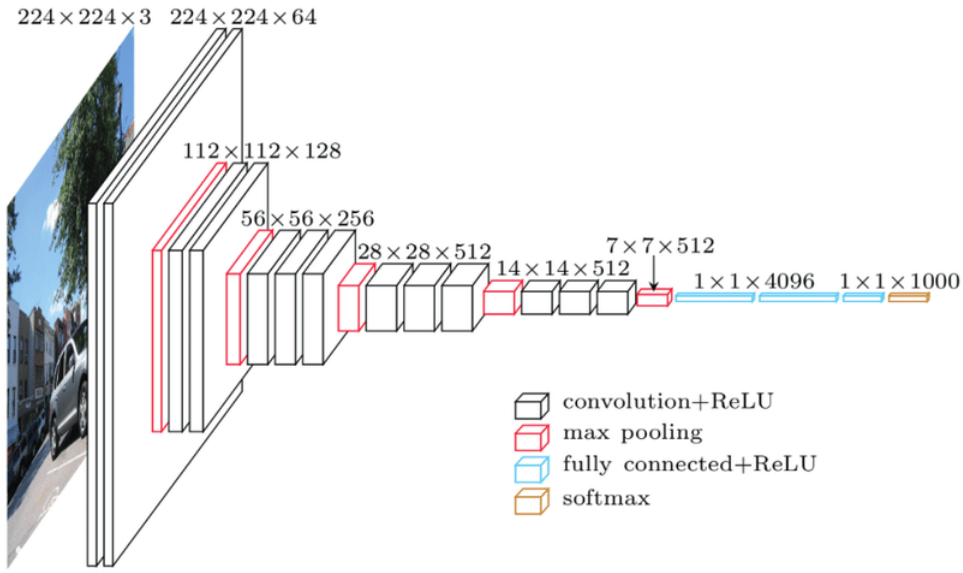

Figure 4: Modèle VGG-Net pour la classification de 1000 classes d'images ici [1]

4. Notre modèle nous renvoie alors un vecteur avec 5 coordonnées, 1 par classe, avec des valeurs entre 0 et 1. Il suffit de sélectionner le barème de décision optimale : prendre la plus grosse probabilismes pour choisir la classe ? Ou bien simplement réduire les classes possibles en éliminant celles ayant une probabilité plus faible que 0.25 par exemple. Tout dépend de l'usage spécifique qu'un médecin voudrait faire ce système d'ifs : un outil de diagnostic ou bien un simple outil qui prépare et aide celui du médecin.

Bien entendu, la Santé est une cause bien trop importante et impactante pour que de l'IA soit utilisée pour prendre des décisions ou assister des experts sans que les performances de cette dernière ne soient rigoureusement et plusieurs fois évalués.
Un outil pareil n'est pas considéré « assez bon » ou d'un niveau « acceptable ». Il se doit d'être d'une performance irréprochable, c'est avec cet objectif en tête que cette alternance s'est déroulée.

11 Résultats

Par manque de temps, la partie classification n'a pas encore été réalisée, en revanche la segmentation est finalisée et les résultats sont très encourageants. Les résultats présentés ci-dessous correspondent donc au modèle U-Net et à l'obtention des images divisées en 4 régions.
Les images à suivre comportent l'IRM de départ, la segmentation réalisée par le modèle et le résultat attendu. L'abréviation LV (Left-Ventricule) correspond au ventricule gauche, RV au ventricule droit et Myo au myocarde.

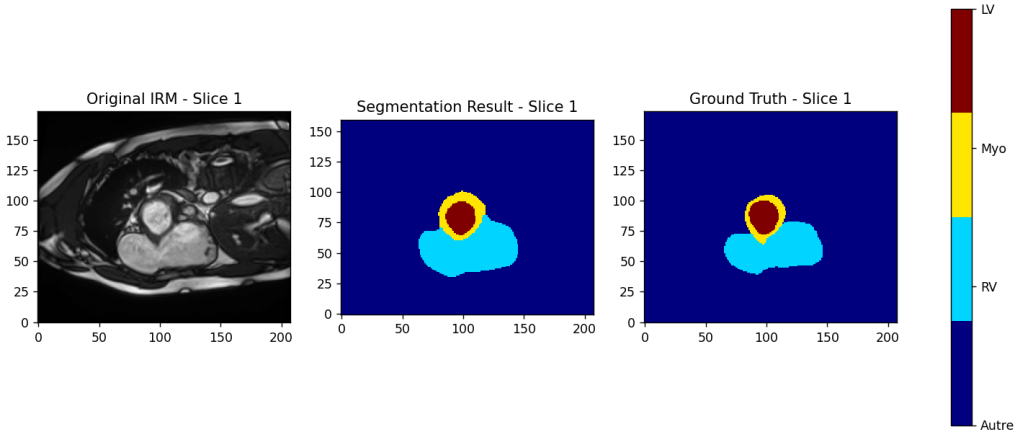

Figure 5: Résultats visuels obtenus (1)

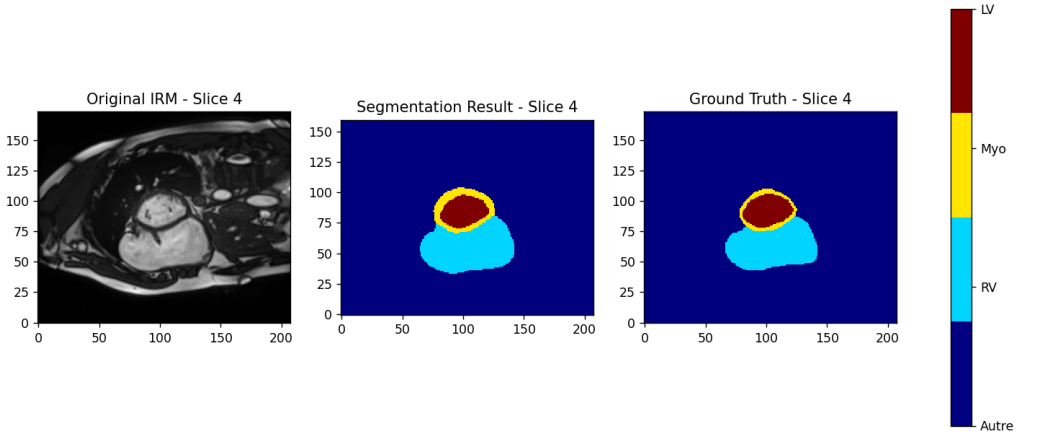

Figure 6: Résultats visuels obtenus (2)

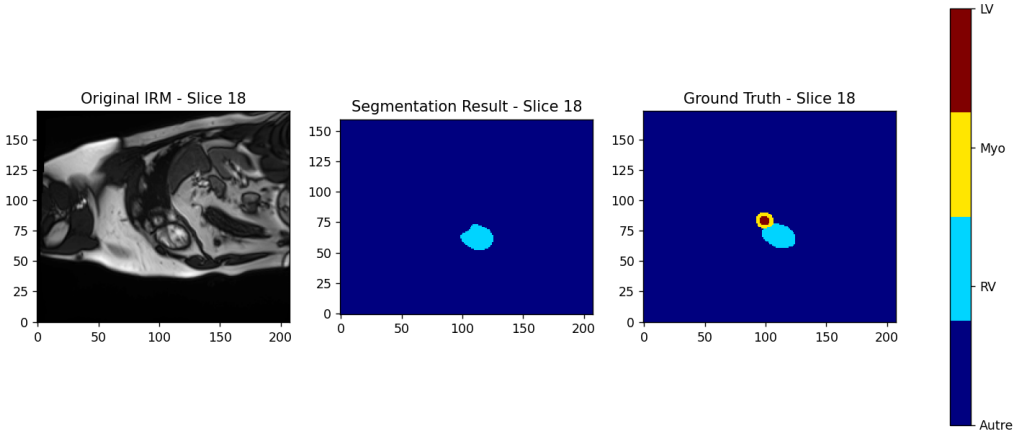

Figure 7: Résultats visuels obtenus (3)

Les résultats semblent visuellement assez bon malgré quelques rares ratés, mais ce n'est bien entendu pas suffisant pour déterminer la qualité du modèle.
Afin d'évaluer les performances d'un algorithme d'IA, il est usuel de surveiller l'évolution de la fonction de coût durant l'entraînement avec les données d'entraînement mais également avec des données qu'elle n'a jamais vu : la validation et le test.

- La validation a lieu tout au long de l'entraînement, à chaque epoch, et on n'effectue pas de correction des paramètres à partir de cette dernière. L'intérêt est de surveiller l'apprentissage de notre IA avec des données qu'elle n'a jamais vu / dont elle n'a aucun souvenir. Si la performance est beaucoup moins bonne en validation que pour l'entraînement alors il est possible qu'on soit face a de « l'overfitting », notre modèle « apprend par coeur ». Ainsi on peut essayer de changer les hyper-paramètres du modèle, des paramètres invariables pendant l'entraînement, tels que le nombre de couche de neurones, le nombre de neurones, le nombre d'epochs (d'itérations d'entraînement), le pas lors de la descente du gradient (taux d'apprentissage ou learning rate)

- Le test se réalise une fois que l'on a optimisé notre modèle en surveillant plusieurs entraînements avec la validation et en ayant choisi la meilleure performance. On calcule pour cela des métriques, des fonctions, qui donnent un aperçu des performances globales de notre IA.

11.1 Métriques

Les métriques utilisées sont : l'accuracy, l'IoU (Intersect over Union) sous 3 variantes micro-average (IoU sur le graphique), macro-average (IoU-avg sur le graphique) et weighted et la cross-entropie "standard" et weighted.

- Accuracy (Précision) : L'accuracy mesure la proportion de prédictions correctes parmi toutes les prédictions.

Formule :

$$\text{Accuracy} = \frac{\text{Nombre de prédictions correctes}}{\text{Nombre total de prédictions}}$$

- Intersection over Union (IoU) : L'IoU mesure le chevauchement entre les prédictions d'un modèle et les vraies valeurs. C'est particulièrement utilisé dans les tâches de segmentation d'image.
  Formule :

$$\text{IoU} = \frac{\text{Intersection}}{\text{Union}}$$

  - Intersection : Nombre de pixels correctement prédits.
  - Union : Somme des pixels prédits et des pixels réels moins l'intersection.

  – Micro-average : Calcule l'IoU globale en agrégeant les vrais-positifs, faux-positifs et faux-négatifs de toutes les classes.

  – Macro-average : Calcule l'IoU pour chaque classe séparément, puis fait la moyenne des résultats.

  – Weighted : Calcule l'IoU pour chaque classe séparément en prenant en compte une distribution de répartition du poids des classes.

- Cross-entropie : Elle mesure la divergence entre la distribution de probabilité prédite par le modèle et la distribution de probabilité réelle.

  – "Standard" :

$$\text{Cross-Entropy} = -\sum_i y_i \cdot \log(\hat{y}_i)$$

  y_i : Probabilité réelle de la classe i.
  $\hat{y}_i$: Probabilité prédite de la classe i.

  – Weighted : Permet de donner plus de poids à certaines classes par rapport à d'autres.

$$\text{Weighted Cross-Entropy} = -\sum_i w_i \cdot y_i \cdot \log(\hat{y}_i)$$

  w_i : Poids associé à la classe i.

Voici donc les courbes obtenus pour les paramètres les plus performants :

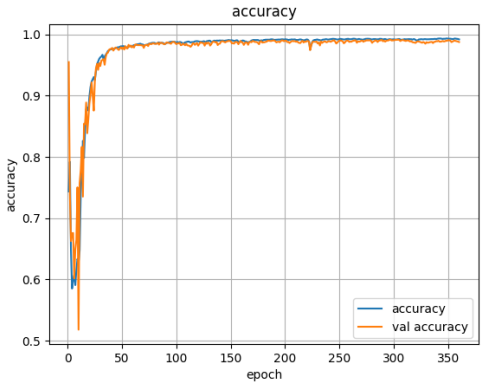

Figure 8: Courbes obtenues pour l'accuracy

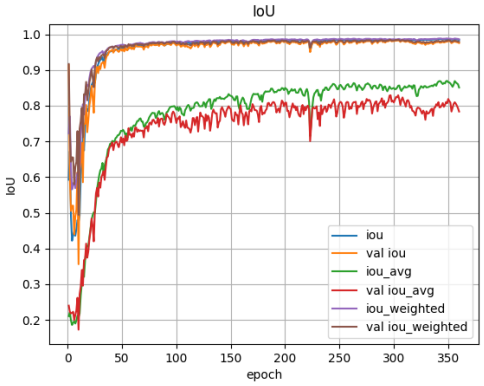

Figure 9: Courbes obtenues pour les IoUs

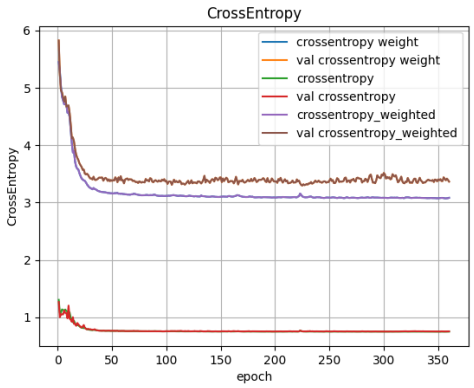

Figure 10: Courbes obtenues pour les cross-entropies

Comme attendu, toutes les fonctions convergent vers un plateau. L'accuracy atteint très rapidement des valeurs proches de 100%.
IoU converge également bien qu'il y ait une disparité entre les "variantes". Ceci s'explique

car l'IoU Macro-average, qui calcule pour chaque classe puis réalise la moyenne, a tendance à grossir l'erreur au moindre pixel mal placé et cela fait baisser l'IoU de plusieurs classes à la fois. Par ailleurs, l'IoU Weighted qui permet d'accorder plus d'importance aux 3 classes (LV, RV et Myo) et moins au "fond" visuel de l'IRM a de très bons résultats et est plus fidèle à ce que l'on cherche à accomplir ici.
La cross-entropie quant à elle converge bien vers un minimum. La question qui s'est posée a été de savoir si ce minimum est local ou global. Il est difficile de le savoir car la fonction correspond à 4 classes et donc est très complexe. Comme première démarche, il a été possible de réaliser une centaine d'entraînements dans les mêmes conditions mais avec des paramètres différents à chaque fois, initialisés aléatoirement.
Mathématiquement, les résultats semblent au premier abord bons. Je considère tout de même nécessaire de montrer et de discuter des performances avec un ou des médecins qualifiés afin qu'un spécialiste puisse trancher quant à la qualité des résultats.

12 Travail restant

Afin d'arriver au bout de ce projet, en considérant les performances obtenues en segmentation comme très bonnes, il faut maintenant mettre en place le modèle de classification et l'optimiser. La classification d'images est un domaine bien maîtrisé et parcouru depuis bientôt 10 ans en Deep Learning [6] et c'est sans aucun doute la partie la plus courte qu'il reste à réaliser.

13 Conclusion

Mon alternance touche à sa fin avec ce rapport. Je suis déçu de ne pas avoir pu aller au bout dans le temps qui m'était imparti, je suis néanmoins très fier du chemin parcouru, des leçons apprises et de la connaissance absorbée.
Merci encore à mon tuteur Salah Bourennane, au plaisir de travailler de nouveau avec vous.

References

[1] Timea Bezdan and Nebojsa Bacanin. "Convolutional Neural Network Layers and Architectures". In: Jan. 2019, pp. 445–451. DOI: 10.15308/Sinteza-2019-445-451.

[2] Özgün Çiçek et al. *3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation*. 2016. arXiv: 1606.06650 [cs.CV].

[3] Keiron O'Shea and Ryan Nash. *An Introduction to Convolutional Neural Networks*. 2015. arXiv: 1511.08458 [cs.NE].

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV].

[5] Jürgen Schmidhuber. "Deep learning in neural networks: An overview". In: *Neural Networks* 61 (Jan. 2015), pp. 85–117. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2014.09.003. URL: http://dx.doi.org/10.1016/j.neunet.2014.09.003.

[6] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV].

[7] Farhana Sultana, Abu Sufian, and Paramartha Dutta. "Advancements in Image Classification using Convolutional Neural Network". In: *2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*. IEEE, Nov. 2018. DOI: 10.1109/icrcicn.2018.8718718. URL: http://dx.doi.org/10.1109/ICRCICN.2018.8718718.

[8] Farhana Sultana, Abu Sufian, and Paramartha Dutta. "Evolution of Image Segmentation using Deep Convolutional Neural Network: A Survey". In: *Knowledge-Based Systems* 201–202 (Aug. 2020), p. 106062. ISSN: 0950-7051. DOI: 10.1016/j.knosys.2020.106062. URL: http://dx.doi.org/10.1016/j.knosys.2020.106062.

[9] Jiawei Zhang. *Gradient Descent based Optimization Algorithms for Deep Learning Models Training*. 2019. arXiv: 1903.03614 [cs.LG].