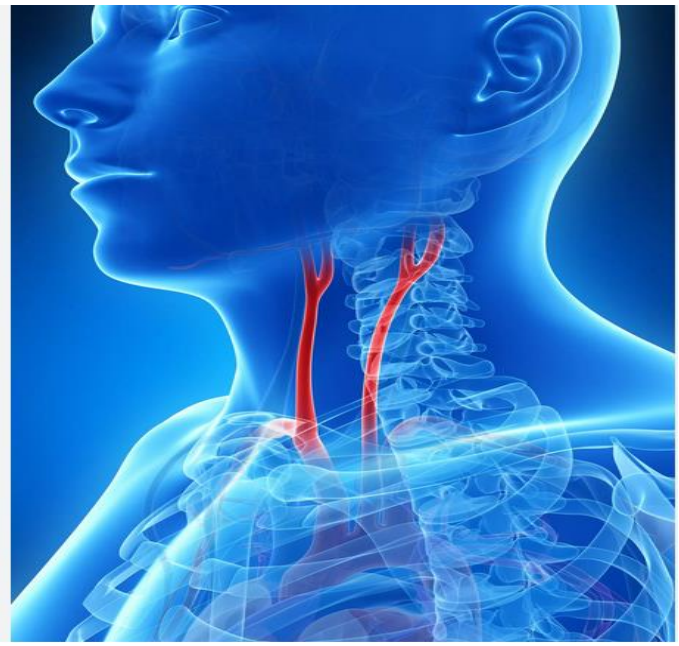
Deep Bridge 22/23

Plateforme Deep Learning d'analyse de données médicales



Étudiants

Judicaël Nshimiye

Adams Pierre David

Encadrant

Gregory Galli

Summary

The Deep Bridge project is a multi-year innovation project derived from the Master 2 MBDS program - Mobiquity, Big Data, and System Integration. Since 2016, MBDS students have worked in partnership with organizations such as CHU, IMREDD, Air PACA, and AtmoSud to prototype the project. In 2019, the Deep Bridge project took on a European scale, aiming to create a digital environment dedicated to medical imaging analysis, allowing for the prediction of risks of cardiovascular disease complications through the use of Machine Learning techniques.

For the year 2023, the project aims primarily to provide a functional prototype of image analysis (reconstructable 3D slices) from a medical scanner to detect potential complications risks due to carotid stenosis. We had the opportunity to work on this project and be part of the innovation process during the year 2023, from January to March.

The Deep Bridge project was designed as a Proof of Concept (PoC) to support the submission of a project aimed at conducting more in-depth clinical trials in collaboration with INRIA, at least two European countries (hospitals and laboratories), and PhD researchers.

Résumé

Le projet Deep Bridge est un projet d'innovation pluriannuel issu du programme Master 2 MBDS - Mobiquité, Big Data et intégration de Systèmes. Depuis 2016, les étudiants du Master MBDS ont travaillé en partenariat avec des organisations telles que le CHU, l'IMREDD, Air PACA et AtmoSud pour prototyper le projet. En 2019, le projet Deep Bridge a pris une ampleur européenne, visant à créer un environnement numérique dédié à l'analyse de l'imagerie médicale, permettant de prédire les risques de complications de maladies cardiovasculaires grâce à l'utilisation de techniques de Machine Learning.

Pour l'année 2023, le projet vise avant tout à fournir un prototype fonctionnel d'analyse des images (tranches reconstituables en 3D) issues d'un scanner médical pour détecter d'éventuels risques de complications dues aux sténoses carotidiennes. Nous avons eu la chance de travailler sur ce projet et de faire partie de ce processus d'innovation durant l'année 2023, de janvier à mars.

Le projet Deep Bridge a été conçu comme un Proof of Concept (PoC) pour appuyer le dépôt d'un projet visant à mener des essais cliniques plus approfondis en collaboration avec l'INRIA, au moins deux pays européens (hôpitaux et laboratoires) et des chercheurs en thèse de doctorat.

Table des matières

Introduction

Deep Bridge a pour but de fournir un modèle d'intelligence artificielle capable de déterminer l'emplacement de la sténose et estimer sa gravité.

Ce modèle ne sera pas utilisé sur des patients réels d'une manière décisionnaire, il servira d'appui au dépôt d'un projet visant à aboutir à des essais cliniques européens plus poussés dans un second temps. L'objectif est donc de prouver que l'approche fonctionne et est viable.

Notre objectif est de construire, évaluer et améliorer un modèle en Deep Learning qui permet de prédire Complications à la suite d'une intervention chirurgicale sur une sténose carotidienne.

Pour l'année 2023, nous sommes donc chargés de :

- Concevoir et implémenter une plateforme Big Data d'imageries médicales
- Catégoriser sténose des images scanners (pré-op et post-op)
- Pouvoir détecter les complications à la suite d'une sténose ou d'une intervention chirurgicale de la sténose
- Analyser/identifier des corrélations entre complications et critères morphologiques
- Construire une IHM mobile permettant aux chirurgiens de consulter les prédictions

Nous nous sommes basés sur l'existant, à savoir les différents projets des étudiants des années précédentes. Ces étudiants ont testé plusieurs technologies de visualisation et de traitement d'images, et certaines d'entre elles ont retenu notre attention.

Le CHU de Nice nous a également fourni 150 dossiers patients avec un bon nombre d'images ainsi que des métadonnées sur les patients. Chaque dossier patient étant un ensemble de fichiers DICOM (Digital Imaging and Communications in Medicine) formant un angioscanner (Examen médical visant à explorer les veines et artères).

Contexte et historique du projet Deep Bridge

La sténose carotidienne est un rétrécissement de l'artère carotidienne qui peut réduire le flux sanguin vers le cerveau, entraînant ainsi une privation d'oxygène et potentiellement la mort de cellules cérébrales si le flux sanguin est interrompu pendant plus de trois minutes. En outre, la sténose peut entraîner la formation d'un anévrisme, qui est une accumulation de sang, ainsi que le détachement de fragments de la plaque d'athérome, qui est un bouchon de cholestérol et de calcium, vers le cerveau ou l'œil, pouvant provoquer une embolie cérébrale.

L'artère carotide commune ou primitive est une artère importante qui se situe dans le thorax et le cou et qui fournit de l'oxygène à la tête et à une partie du cou. Elle se divise en deux parties principales :

- L'artère carotide interne, qui irrigue le cerveau.
- L'artère carotide externe, qui irrigue le cou et le visage et assure ainsi leur bon fonctionnement en fournissant l'oxygène nécessaire.

La sténose carotidienne peut être asymptomatique, ce qui signifie qu'un patient n'a pas présenté d'accident cérébral hémisphérique homolatéral dans les six derniers mois, ou symptomatique, ce qui indique que le patient est à haut risque d'AVC. Dans le cas d'une sténose carotidienne asymptomatique, une prescription médicamenteuse peut être suffisante. Cependant, si la sténose est symptomatique, une intervention chirurgicale est généralement recommandée, sauf dans des cas particuliers où l'espérance de vie du patient est inférieure à trois ans.

Le défi majeur pour les chirurgiens et les médecins est de déterminer si la sténose carotidienne d'un patient nécessite une intervention chirurgicale ou non. C'est là que le projet Deep Bridge intervient. Grâce aux avancées remarquables de l'informatique dans le domaine de l'intelligence artificielle, notamment le Deep learning, une solution est envisagée.

Il fallait donc tenter d'entraîner un modèle pour détecter l'emplacement exact de la sténose et prédire le risque cardiovasculaire lié à cette dernière.

Par exemple, ce modèle pourrait analyser les images d'imagerie médicale telles que les IRM et les scanners pour identifier les caractéristiques de la sténose carotidienne. Ces caractéristiques incluent la localisation, la taille et la gravité de la sténose. Le modèle pourrait également prédire le risque cardiovasculaire lié à la sténose en utilisant des données démographiques et cliniques telles que l'âge, le sexe, la pression artérielle et le taux de cholestérol.

En fin de compte, le projet Deep Bridge pourrait aider les médecins à prendre des décisions plus éclairées quant à la nécessité d'une intervention chirurgicale chez les patients atteints de sténose carotidienne. Cela permettrait d'optimiser le traitement pour les patients tout en réduisant les coûts et les risques associés aux interventions chirurgicales inutiles.

Étude de l'existant

Analyse

Dans le cadre de ce projet, Monsieur Galli nous a fourni, en plus des 150 dossiers patients, les réalisations des étudiants des années 2020 à 2022 sur Deep Bridge pour nous aider à avancer. Ces étudiants ont réalisé d'importants progrès, en particulier en ce qui concerne la visualisation des fichiers DICOM. Ils ont testé plusieurs outils de visualisation, tels que :

• Vedo, qui est un outil open-source pour la visualisation et l'analyse de données en 3D et 4D. Il peut être utilisé pour visualiser les images DICOM en 3D, ce qui peut aider à comprendre la structure de l'artère carotide et à déterminer la position et l'étendue de la sténose.

• OpenCV, qui est une bibliothèque open-source pour le traitement d'images et de vidéos. Elle propose des fonctions pour la normalisation d'échelle, la correction du bruit, la détection de bords, etc. qui peuvent être utilisées pour pré-traiter les images DICOM.

• PyDICOM, qui est une bibliothèque pour la manipulation et l'analyse des images DICOM. Elle permet d'importer et de lire des images DICOM et d'extraire les informations métadonnées.

En plus de ces outils, Monsieur Galli nous a présenté le logiciel RadiAntViewer, qui permet également de visualiser les fichiers DICOM et de zoomer sur la partie qui nous intéresse, à savoir la carotide.

Tous ces outils nous ont permis d'obtenir une première vue des images et de l'emplacement de la carotide, bien que cet emplacement ne soit pas toujours exactement au même endroit d'un patient à l'autre. Cependant, nous avons rapidement constaté que nous étions confrontés à la même problématique que les étudiants des années précédentes. En effet, bien que nous puissions visualiser les images, nous ne pouvons pas découper l'image pour ne garder que la partie qui nous intéresse (voir la partie « segmentation des images ») à l'aide de ces outils.

Après une discussion avec Monsieur Galli, nous avons convenu de partir d'un logiciel réalisé par l'un des groupes de l'année scolaire 2021-2022. Ce logiciel est basé sur un script Python, et notre objectif est de l'améliorer pour le rendre plus fluide et pour extraire la partie de l'image où se trouve la carotide. Les étudiants qui ont réalisé ce logiciel ont pris soin de noter les limitations de leur outil, afin d'aider les générations futures à avancer plus rapidement.

En somme, nous avons étudié les réalisations précédentes des étudiants et avons utilisé des outils tels que Vedo, OpenCV, PyDICOM et RadiAntViewer pour visualiser les fichiers DICOM

« Lorsqu'on change de paramètre de rotation ou de profondeur, la génération de la nouvelle image n'est pas instantanée. Ceci est dû au temps de calcul pour régénérer l'image 2D. Pour optimiser cela, l'idée serait de stocker en mémoire les images de toutes les combinaisons possibles des paramètres de rotation et de profondeur au démarrage de l'application. Cette solution augmenterait bien sûr le temps de démarrage, mais l'application deviendrait beaucoup plus fluide »

Critique de l'existant

Dans un premier temps, tout comme les étudiants de l'année scolaire 2021-2022, nous avions conclu que la solution proposée par ceux de 2020-2021 n'est pas viable.

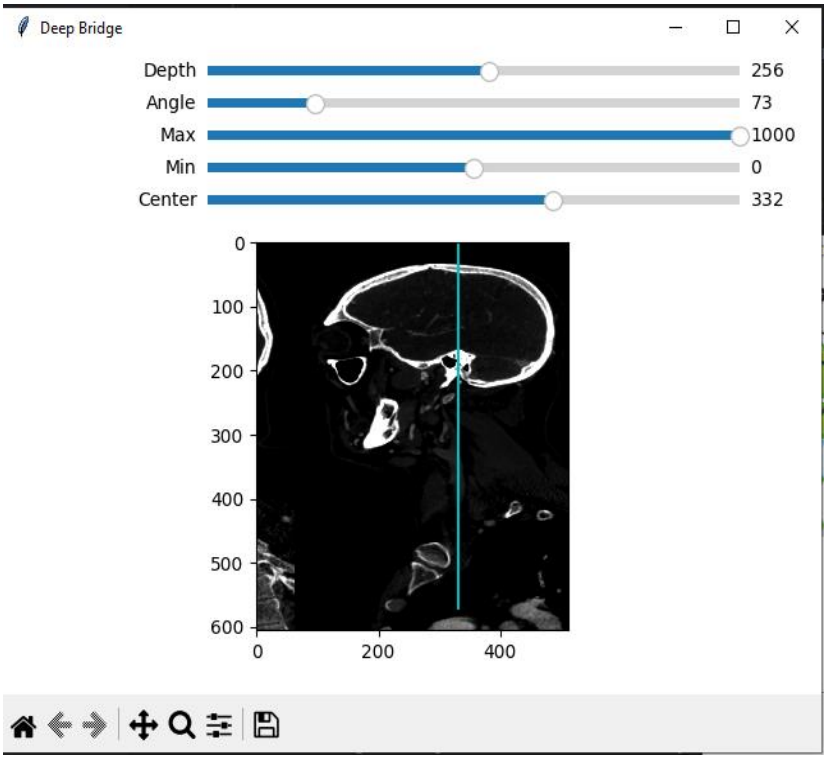
« La solution n'est tout simplement pas viable. Elle met en jeu la puissance mémoire de l'ordinateur trop importante et empêcherait le praticien de santé d'effectuer des opérations sur d'autres logiciels de façon asynchrone lors du long chargement de ces images.

Ajoutant aux mises en garde relevées dans le rapport, nous avons également analysé le code. Nous en avons déterminé que ce code n'est tout simplement pas maintenable, car le programme fait appel à des librairies non mises à jour et non optimisées. De plus, le programme et les librairies rendent l'application d'une intelligence artificielle difficile et, avec un programme qui a déjà du mal à tourner, le rendrait inutilisable. »

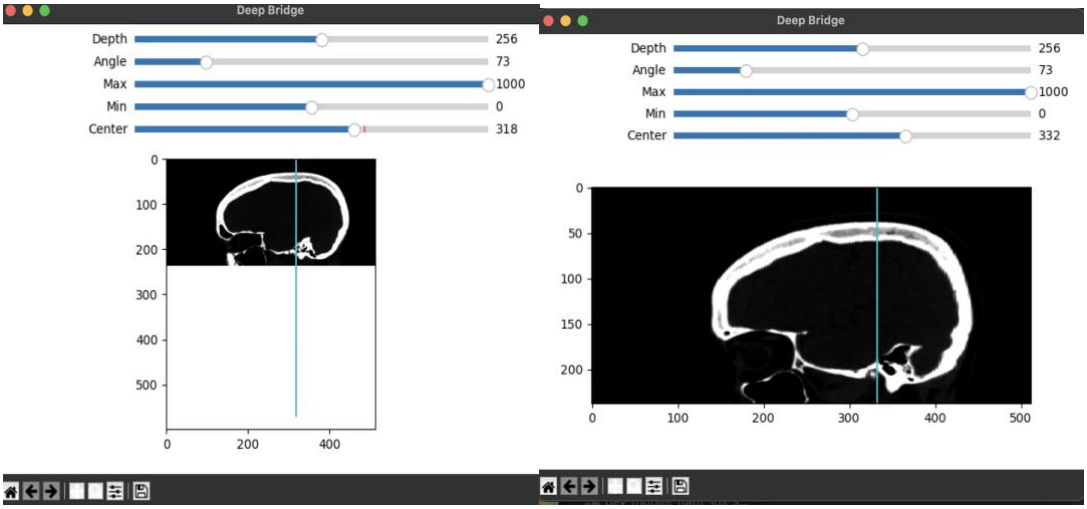
Sur le plan logique, il est risqué de stocker en mémoire la combinaison de toutes les images, qui peuvent être plus de 500 fichiers DICOM, avec les différents angles possibles. Cela nous a conduits à penser que cette solution pourrait ne pas être viable. Cependant, cette critique a été examinée de manière plus approfondie dans la section "Reconsidération de l'existant".

Après avoir analysé ces problèmes rencontrés par les étudiants pendant l'année scolaire 2021-2022, nous avons identifié, en plus de cela, plusieurs autres problèmes, notamment :

- La difficulté, voire l'impossibilité, de visualiser la carotide sur la plupart des dossiers patients, même en modifiant les valeurs d'affichage.



- Un dysfonctionnement dans le script utilisé avec certains dossiers de patients. Plus spécifiquement, nous avons constaté que les fichiers n'étaient pas correctement classés dans les dossiers des patients et que certains scans ne contenaient qu'un seul fichier DICOM. Ce problème a entraîné des erreurs lors de l'exécution du script « scriptrotation.py » des étudiants de l'année scolaire 2020-2021. *Voir les 2 figures ci-dessous.*



- La latence quand on essaie de faire une rotation.
- L’axe central ne se trouve toujours pas au même endroit selon le dossier patient et se déplace quand on fait une rotation.
- L'utilisation de ‘Tkinter’ pour l’interface graphique peut ne pas être idéale pour tous cas d’utilisations.
- L'utilisation de valeurs constantes dans le code (ex: 570) qui peuvent être spécifique à une image et non toutes les images

État de l’art

Dicom

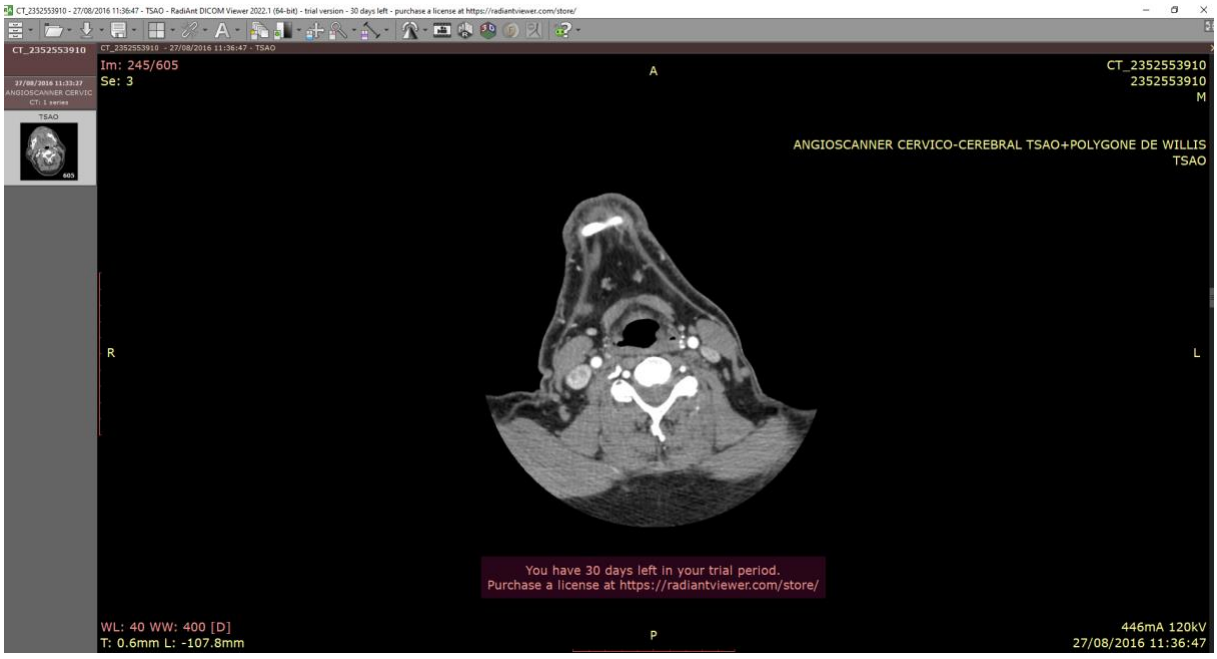
Le format DICOM (Digital Imaging and Communications in Medicine) est un format de fichier standard utilisé pour stocker, transférer et afficher des images médicales et des informations associées. Ce format est largement utilisé dans le domaine de l'imagerie médicale, notamment pour les scanners, les IRM, les échographies, les radiographies et les mammographies. Les fichiers DICOM contiennent des informations sur les images, telles que les données sur les patients, les informations sur l'appareil qui a été utilisé pour acquérir l'image, les dimensions de l'image, les données de pixel, ainsi que d'autres informations utiles pour l'analyse et l'affichage de l'image. Le format DICOM est conçu pour être utilisé avec une variété d'appareils et de systèmes différents, ce qui permet aux professionnels de la santé de partager facilement des images médicales et de collaborer sur des diagnostics et des traitements.

Python

Python est un langage de programmation interprété de haut niveau, multiplateforme et polyvalent, utilisé pour le développement de nombreux types d'applications, notamment des applications de bureau, des applications web, des applications mobiles, des jeux vidéo, de l'analyse de données, du Machine Learning, de l'automatisation et bien plus encore. Il est conçu pour être facile à lire, à écrire et à comprendre, ce qui en fait un langage de programmation populaire pour les débutants et les experts en informatique. Python est également connu pour sa grande bibliothèque standard, qui fournit de nombreuses fonctionnalités utiles pour les tâches courantes de programmation.

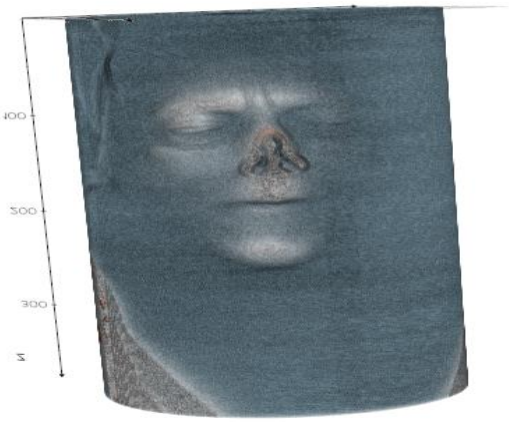
RadiAntViewer

RadiAnt DICOM Viewer est un logiciel de visualisation d'images médicales qui prend en charge les fichiers DICOM. Il permet de visualiser des images provenant de différents types d'examens d'imagerie médicale tels que les rayons X, l'échographie, l'IRM, le scanner, etc. Le logiciel offre des fonctionnalités avancées pour la manipulation et l'analyse des images, telles que la mesure des distances, la sélection de régions d'intérêt, la création de points de référence et de lignes de mesure, la superposition de plusieurs images pour les comparer, et bien d'autres encore. RadiAnt DICOM Viewer est largement utilisé dans les établissements de santé et les cliniques pour visualiser et analyser les images médicales.



Vedo

Vedo est une bibliothèque Python pour la visualisation de données volumétriques et 3D. Elle est basée sur la bibliothèque VTK (Visualisation Toolkit) et fournit des outils pour créer des rendus interactifs de données scientifiques complexes. Vedo permet de charger des données volumétriques à partir de divers formats, tels que DICOM, NRRD, MHA, ou même des données structurées en fichier CSV, et de les visualiser sous forme de coupe transversale, de surface, ou de volume. Il est également possible de manipuler et de segmenter les données volumétriques, ainsi que de créer des animations et des rendus 3D pour les exporter sous forme de fichiers vidéo ou d'images.



Pydicom

Pydicom est une bibliothèque Python pour lire et travailler avec des fichiers DICOM. Il fournit des fonctionnalités pour la manipulation des métadonnées et des images DICOM. Pydicom permet également de convertir des images DICOM en tableaux NumPy pour un traitement plus facile avec des bibliothèques Python telles que NumPy et Matplotlib.

NumPy

Numpy est une bibliothèque Python pour le calcul numérique avec Python. Elle fournit des tableaux multidimensionnels de haut niveau, des fonctions mathématiques pour l'opération sur ces tableaux, ainsi que des outils pour l'intégration de code C/C++ et Fortran. NumPy est souvent utilisé pour le traitement d'images et la manipulation de données dans les projets de science des données.

Matplotlib

Matplotlib est une bibliothèque Python pour la visualisation de données. Elle permet de créer des graphiques 2D et 3D, des diagrammes de dispersion, des histogrammes, des graphes en barres, des graphes en secteurs, etc. Matplotlib peut être utilisé avec des tableaux NumPy pour visualiser des données.

Multiprocessing

Multiprocessing est un module Python qui permet de créer et de gérer des processus en parallèle sur plusieurs cœurs ou processeurs. Cela permet d'augmenter les performances des applications en tirant parti des capacités de traitement parallèle des machines modernes. Le module multiprocessing fournit des fonctions et des classes pour créer et gérer des processus, des tâches en arrière-plan et des communications entre processus. Il permet également de partager des données entre les processus, en utilisant des objets tels que des queues, des pipes ou des tableaux partagés. Multiprocessing est particulièrement utile pour les applications qui nécessitent un traitement intensif de données, telles que l'analyse de grandes quantités de données, le traitement d'images ou de vidéos, ou la simulation de systèmes complexes.

Reconsidération de l’existant

Après avoir examiné attentivement le code du logiciel et effectué plusieurs tests, il est apparu que la latence ne résultait pas nécessairement de la mise en mémoire des images DICOM. Cela peut s’expliquer par le fait que les fichiers DICOM sont téléchargés en mémoire dans leur code

```
ROTATION_X = 332
files = []
dossier = "./SF103E8_10.241.3.232_20210127172347611/1.2.840.113619.2.55.3.2148147470.668.1396266453.354/*"


"""
    Chargement des fichiers
"""
print('glob: {}'.format(dossier))
for fname in glob.glob(dossier, recursive=False):
    files.append(pydicom.dcmread(fname, force=True))
print("file count: {}".format(len(files)))
```

Lorsque le script est exécuté, il recherche les fichiers DICOM dans le dossier spécifié sur le disque. Cependant, la rotation n'affecte pas les sources de fichiers DICOM, et probablement pas celles en mémoire. La rotation est uniquement une transformation visuelle appliquée à l'affichage des images DICOM à l'écran. Cette constatation nous a amenés à reconsidérer notre évaluation de la viabilité du logiciel.

Nous avons donc tenté d'identifier les différents facteurs qui pourraient causer cette latence. Voici quelques-uns des problèmes que nous avons identifiés :

- Les performances de l'ordinateur : si l'ordinateur n'a pas une bonne puissance de traitement ou une carte graphique dédiée, cela peut ralentir la rotation, car il n'est pas capable de traiter les données rapidement. La rotation peut également être affectée par d'autres facteurs tels que la mémoire disponible sur l'ordinateur et les autres programmes en cours d'exécution. Si l'ordinateur est surchargé ou si la mémoire est saturée, cela peut de plus ralentir la rotation.
- Le nombre et la taille de fichiers DICOM : s'il est très élevé, cela peut ralentir la rotation.
- Le programme lui-même : le programme utilisé pour la rotation peut ne pas être optimisé pour les performances, ce qui peut entraîner des ralentissements :
  - Utilisation des algorithmes de rotations faites manuellement, au lieu des bibliothèques de traitement d’images optimisées comme OpenCV ou NumPy
  - Possibilités non saisies : utilisation de la rotation en parallèle en utilisant plusieurs cœurs de processeur pour réduire le temps d'exécution ou utilisation de bibliothèques qui le fait pour nous (ex : multiprocessing).

- Utilisation d'un tableau pour stocker les fichiers DICOM, il existe des structures de données plus adaptées à l'accès aux fichiers

Tout cela nous a amené à reconsidérer la viabilité du logiciel et à nous mettre sur l'amélioration de ce dernier.

Solutions et Réalisations

Solutions proposées

À cette étape du projet, nous avons exploré différentes solutions pour résoudre les problématiques identifiées. Certaines solutions ne nécessitent aucune intervention, telles que l'augmentation de la puissance de l'ordinateur. Cependant, d'autres nécessitent des solutions techniques plus ou moins complexes :

1) Ainsi, pour améliorer le script "scriptrotation.py", il est nécessaire de prétraiter les dossiers des patients avant de les soumettre au script « scriptrotation.py ». Cela implique d'organiser les fichiers DICOM en dossiers distincts pour chaque patient, en s'assurant que chaque dossier contient tous les fichiers requis pour le traitement.

   Par exemple, supposons qu'un patient ait subi une IRM de la tête et que les fichiers DICOM correspondants soient stockés dans un dossier nommé "Patient A".

   Dans ce cas, tous les fichiers DICOM de l'IRM de la tête doivent être stockés dans un sous-dossier de “Patient A” plutôt que d'être mélangés avec d'autres fichiers DICOM du même patient.

2) Utilisation de bibliothèques de traitement d’images telles que OpenCV et NumPy pour remplacer les fonctions faites à la main.
3) Utilisation de multiprocessing pour créer et gérer des processus en parallèles afin d’avoir des sous processus séparés du processus principal et de les exécuter de manière asynchrone.
4) Utilisation de vues ou références sur les données existantes pour éviter les opérations de copies de données inutiles et coûteuses.
5) Remplacer les données mises statiquement pour le calcul d’affichage de l’image par des données standards à éventuellement toutes les données à notre disposition
6) Trouver un angle adéquat à la visualisation de la carotide et faire des tranches afin d’avoir un affichage en 2D plutôt qu’en 3D, ce qui alourdit considérablement le logiciel.
7) Enfin, nous pouvons entraîner un modèle avec les différentes tranches et faire des prédictions.

Mise en place des solutions

Prétraitement des images

Le prétraitement des images est une étape cruciale dans l'analyse des artères carotides à partir de fichiers DICOM. Nous avons identifié 4 étapes clés pour cette étape :

- Importer les images DICOM à partir de leur source en utilisant Pydicom. Cette bibliothèque permet d'importer et de lire les images DICOM et d'extraire les informations métadonnées associées.

- Normaliser les intensités de gris de l'image pour améliorer la qualité de l'analyse. Cela peut être réalisé en utilisant des techniques telles que la normalisation z-score ou la normalisation min-max. La normalisation permet d'ajuster les intensités des pixels de l'image de manière qu'elles aient une distribution plus homogène, ce qui facilite l'analyse.

- Éliminer le bruit présent dans l'image en utilisant des algorithmes de filtrage tels que le filtre médian, le filtre gaussien ou le filtre de Wiener. Le bruit peut provenir de diverses sources telles que la qualité de la source d'imagerie ou le processus d'acquisition des données. L'élimination du bruit améliore la qualité de l'image et permet une analyse plus précise.

- Définir une région d'intérêt dans l'image qui contient l'artère carotide pour minimiser les effets du bruit en dehors de cette région. En utilisant la bibliothèque OpenCV conjointement avec Pydicom, il est possible d'importer les images DICOM avec Pydicom, puis d'utiliser les fonctions de prétraitement d'OpenCV pour préparer les images pour l'analyse. La définition d'une région d'intérêt permet de se concentrer sur la partie de l'image qui contient l'information la plus pertinente pour l'analyse des artères carotides.

En combinant les fonctionnalités de Pydicom et OpenCV, nous pouvons obtenir des résultats optimaux pour l'analyse des artères carotides à partir de fichiers DICOM

Segmentation des images

La segmentation de l'artère carotide se déroule en 4 étapes clés : détection des bords, extraction de la région d'intérêt, segmentation de l'artère et nettoyage de la segmentation.
Des algorithmes de détection de bords sont utilisés pour trouver les bords de l'artère, puis des algorithmes de sélection de régions pour extraire la région d'intérêt contenant l'artère.
Des algorithmes de segmentation sont ensuite utilisés pour séparer l'artère des autres structures de l'image, et enfin des algorithmes de filtrage sont utilisés pour nettoyer la segmentation et enlever les artefacts et le bruit.

Implémentation

Après avoir examiné les problèmes évoqués dans la section "critique de l'existant", nous avons élaboré une solution pour améliorer l'organisation des fichiers DICOM et faciliter l'analyse des images pour détecter les sténoses carotidiennes.

Nous avons créé un nouveau script, appelé "organize_dicom_files.py", qui permet de lire tous les fichiers dans les dossiers et de les classer par patient et par scan. Ce script garantit que les fichiers sont correctement organisés, ce qui évite les erreurs liées à la mauvaise classification des fichiers et facilite l'analyse des images.

Par exemple, si un patient a subi plusieurs scans, le script va regrouper tous les fichiers correspondants dans un seul dossier, ce qui permet une meilleure organisation et une analyse plus précise des données. Cette approche permet également de faciliter la recherche de données pour les professionnels de la santé, en leur donnant un accès rapide aux images des patients.

Voici un aperçu des principales fonctionnalités du script :

1. Le script lit les métadonnées des fichiers DICOM en utilisant la bibliothèque Pydicom.
2. Il vérifie si le fichier est un scan CT (tomodensitométrie) et extrait l'ID du patient et l'UID de la série.
3. Le script crée un nouveau dossier pour chaque patient et série de scans, s'ils n'existent pas déjà.
4. Il déplace les fichiers DICOM valides vers les dossiers correspondants aux patients et aux séries de scans.
5. Les fichiers DICOM invalides sont déplacés vers un dossier spécifique invalid_dicom_files.
6. Les fichiers non-DICOM sont également identifiés et les fichiers avec l'extension .cab sont supprimés.
7. Enfin, le script affiche des messages d'information et d'avertissement pour faciliter le suivi de son exécution.

Pour utiliser le script, il suffit d'exécuter la commande suivante :

```
python organize_dicom_files.py <source folder> <output folder>
```

Analyse des résultats

Pour l'analyse des résultats, nous avons développé un nouveau script appelé "analyse_results.py". Ce script est conçu pour donner une vision globale des résultats obtenus après l'exécution du script "organize_dicom_files.py".

En effet, le script "analyse_results.py" affiche le nombre total de dossiers patients avec des fichiers DICOM correctement classés, ainsi que le nombre de scans par patient et le nombre de fichiers DICOM par scan. Ces informations sont essentielles pour évaluer la qualité des résultats obtenus et identifier les zones qui nécessitent une amélioration.

Par exemple, si le nombre de fichiers DICOM par scan est faible, cela peut indiquer un problème lors de l'acquisition des images ou une erreur dans le processus de classification des fichiers. Si le nombre de scans par patient est incohérent, cela peut révéler une erreur dans le processus de consolidation des scans.

Initialement, on a les données suivantes :

Nombre de dossiers patients	Nombre de scans	Nombre de fichiers DICOM
150	204	148 551

Après avoir exécuté le script “organize_dicom_files.py”, on obtient les données suivantes :

Nombre de dossiers patients	Nombre de scans	Nombre de fichiers DICOM
148	192	145 660

ID Patient	Nombre de Scans
1350553313	1
1350593310	1
1350643714	1
1350653514	1
1350663716	1
1352563011	1
1352583213	1
1352633119	1
1352633614	1
1352663316	1
1352703418	1
1352763916	1
1353623118	1
1353673610	1
1353673812	1
1353703618	1
1353713517	1
1354633215	1
1354683417	1
1355603213	1
1355623012	1
1355663817	1
1355683212	1
1355713217	1
1355723813	1
1355773817	1
1356513110	1
1356653615	1
1356663812	1
1356713715	1
1357633010	1
1357653710	1
1357703218	1
1357713617	1
1357733515	1
1357753710	1

ID Patient	Nombre de Scans
0350563315	1
0351603315	1
0351643812	1
0352763915	1
0356763316	1
04571438144	1
345064341295	1
0356663913	1
0352643411	1
0350653512	1
9358743817	1
0355703518	1
0356633312	1
0457933131	1
0350593215	1
0351543316	1
0354463112	1
0355573511	1
0358703915	1
0353683019	1
0358673714	1
0359623013	1
0353533113	1
9453232101	1
0354593812	1
0355653219	1
0356683819	1
1351653614	3
1359643015	5
0350623919	6
0351503614	6
0356693416	11
1359723314	19

À la suite de l'exécution du script « organize_dicom_files.py », il a été observé que deux dossiers patients et douze séries d'images ont été supprimés de notre jeu de données. De plus, il a été retiré un total de 2 891 fichiers DICOM.

Cependant, il est important de noter que des incohérences ont été identifiées concernant le nombre de fichiers DICOM par scan. En effet, certaines séries d'images ne contiennent qu'un seul fichier DICOM, tandis que d'autres peuvent contenir jusqu'à 2 710 fichiers.

Pour illustrer les incohérences identifiées, voici quelques exemples de séries d'images et du nombre de fichiers DICOM qu'elles contiennent :

UID Scan	# DICOM files
2.16.840.1.113669.632.21.1518758148.813018979.186459887021773101	1
2.16.840.1.113669.632.21.1518758148.813018979.260396370230192065	1
2.16.840.1.113669.632.21.1518758148.813018979.261340083830393265	1
2.16.840.1.113669.632.21.1250388482.1355635.47065696727882891414	2
2.16.840.1.113669.632.21.1250388482.1355635.10141705352858799141	2
2.16.840.1.113669.632.21.1250388482.1355635.22938670072508411506	2
2.16.840.1.113669.632.21.1401581570.2685747106.60276598219461533	68
2.16.840.1.113669.632.21.1434811657.1889886034.30826410172026833	85
2.16.840.1.113669.632.21.1250388482.1355635.16255855032668067092	102
2.16.840.1.113669.632.21.1619032067.537267970.655795002026273972	112
2.16.840.1.113669.632.21.1434811657.1889886034.30826409052016823	138
2.16.840.1.113669.632.21.1434811657.1889886034.30826410012036843	168
2.16.840.1.113669.632.21.1250388482.1355635.17094715832698127092	212
2.16.840.1.113669.632.21.1250388482.1355635.16926943672708147092	212
2.16.840.1.113669.632.21.1250388482.1355635.16591399352688107092	352
2.16.840.1.113669.632.21.113634824.1085291611.243782229824483514	440
2.16.840.1.113669.632.21.1518366470.1347792643.40886923742648151	504
2.16.840.1.113669.632.21.1400994821.1617248018.34996056762517651	508
2.16.840.1.113669.632.21.1468296456.1890942787.41799205042668411	510
2.16.840.1.113669.632.21.1149464069.1080315699.30120896063471066	517
2.16.840.1.113669.632.21.774139651.1342353987.125558190823571714	632
2.16.840.1.113669.632.21.1535143431.1078312707.29677650982447939	1086
2.16.840.1.113669.632.21.2760325203.621432694.176091396724285113	1095
2.16.840.1.113669.632.21.1267105285.1080344418.26857360392438571	1098
2.16.840.1.113669.632.21.1535798787.1074118563.13947702122473613	1242
2.16.840.1.113669.632.21.1149987584.271863731.270356208124084413	1245
2.16.840.1.113669.632.21.1736473605.1617329922.45928224322275413	1251
2.16.840.1.113669.632.21.1368092676.2692030386.10750538342458344	2430
2.16.840.1.113669.632.21.1401581570.2685747106.18791947372579301	2460
2.16.840.1.113669.632.21.1653047040.1345728370.29055982323883213	2710

Cela nous a permis de voir que certains fichiers DICOM n'avaient pas d'attribut “Slice Location”, donc qu’ils sont inutilisables pour la suite du script.

Pour être sûr que l'on ne peut pas utiliser ces scans, on a testé tous les scans de tous les patients avec la fonction “slice_location_search” et tous les scans ne contenant qu'un seul fichier Dicom n'ont pas d'attribut Slice Location. On a en tout 30 dossiers de scans qui contenaient des fichiers Dicom sans attribut Slice Location.

Mise en Garde

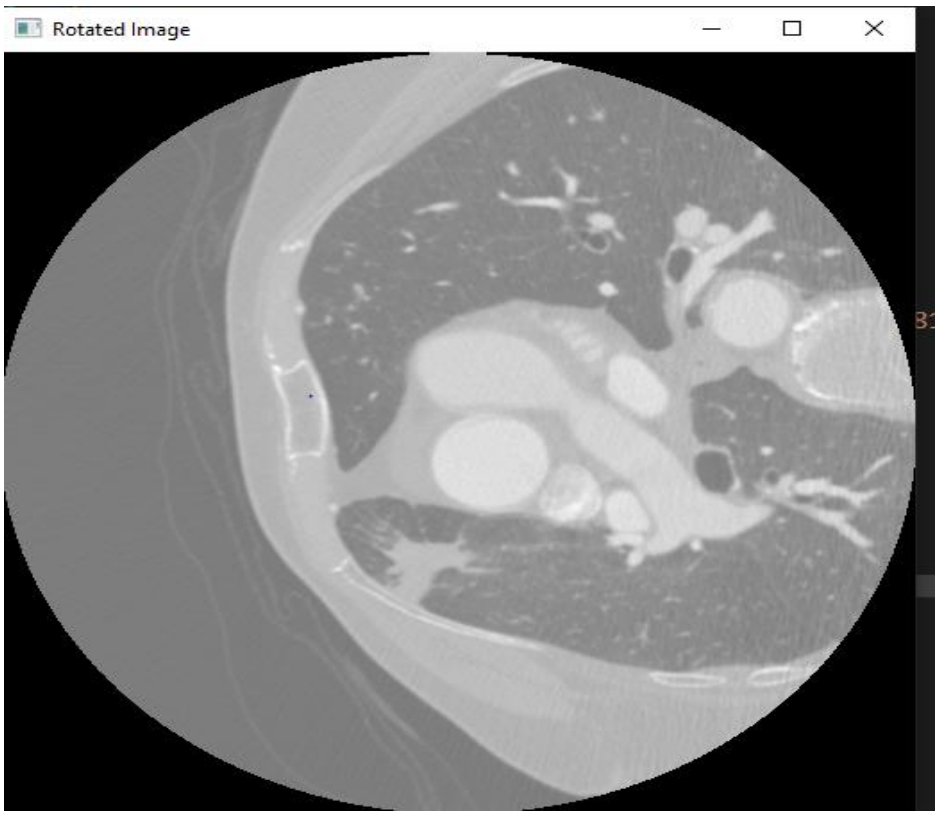
À noter qu'avant d'exécuter le script « organize_dicom_files.py », Il est recommandé de faire une copie des dossiers patients avant d'exécuter le script « organize_dicom_files.py ». En effet, ce script ne copie pas les fichiers DICOM pour les organiser, mais les déplace plutôt vers de nouveaux dossiers. Si vous souhaitez conserver les dossiers patients originaux, il est donc essentiel de faire une copie avant de lancer le script.

De plus, il est important de noter que pour la partie « analyse des résultats », il est nécessaire d'avoir une copie du dossier original. Ceci permettra de s'assurer que toutes les données sont présentes et que l'analyse peut être effectuée de manière fiable.

Améliorations du Logiciel

Après avoir validé la viabilité du logiciel, nous nous sommes concentrés sur l'identification des opportunités d'amélioration. Nous avons examiné diverses approches et cherché à adapter les solutions proposées de manière optimale (voir la section "Solutions proposées").

Dans un premier temps, nous avons pris soin d'analyser et d'exécuter le programme pour en comprendre le fonctionnement. Par la suite, nous avons progressivement intégré différentes bibliothèques, en commençant par l'analyse d'un fichier DICOM unique plutôt qu'un dossier entier.



Voir le fichier "image_visualisation.py"

Nous avons procédé à 3 versions du script, la première ne contenant que des modifications mineures par rapport à celle qu'ont fait les étudiants de l'année dernière. On a renommé certaines variables pour mieux refléter leur fonction. Et on a améliorer certaines fonctions en évitant les calculs répétitifs.

Nous avons procédé à la réorganisation du script, en ajoutant des commentaires explicites et en définissant clairement les sections de code pour faciliter la compréhension et la maintenance.

Nous avons amélioré le filtrage des fichiers du dossier patient en incluant uniquement les fichiers portant l'extension .dcm et en excluant les fichiers cachés.

La gestion des fichiers DICOM sans attribut SliceLocation a été optimisée en les comptabilisant et en les excluant du traitement ultérieur.

Le code a été structuré de manière plus modulaire, ce qui facilite la réutilisation et la maintenance. Les fonctions sont clairement définies et séparées les unes des autres.

Dans les scripts suivants on a utilisé la bibliothèque OpenCV pour faire réaliser la rotation des images Dicom au lieu de la fonction rotate_x ce qui améliore la fluidité de la rotation.

Le choix d'utiliser cv2 (OpenCV) plutôt que scipy pour effectuer les rotations d'image dans le troisième script peut s'expliquer par plusieurs raisons :

- Performance : OpenCV est une bibliothèque optimisée pour le traitement d'image et vidéo en temps réel. Elle repose sur le langage C/C++ et est conçue spécifiquement pour offrir des performances élevées. Comparativement, scipy est une bibliothèque plus générique dédiée aux opérations scientifiques et mathématiques. Bien qu'elle soit performante, elle peut ne pas être aussi rapide qu'OpenCV pour certaines opérations de traitement d'image.
- Fonctionnalités étendues : OpenCV propose une vaste gamme de fonctionnalités pour le traitement d'image, la vision par ordinateur et l'apprentissage automatique. Si les développeurs envisagent d'ajouter des fonctionnalités supplémentaires liées au traitement d'image à l'avenir, il serait plus judicieux d'utiliser OpenCV dès le départ.
- Interopérabilité : OpenCV fonctionne bien avec d'autres bibliothèques de traitement d'image telles que NumPy et matplotlib, facilitant ainsi l'intégration de différentes parties du code. Dans le troisième script, OpenCV est utilisé conjointement avec NumPy pour effectuer des opérations sur les images.

Dans la dernière version du script, nous avons décidé de remplacer la bibliothèque Tkinter par OpenCV pour l'affichage des images et la gestion des interactions utilisateur ce qui donne une meilleure performance en comparaison avec Tkinter.

Pour permettre aux utilisateurs d'interagir avec les images et de localiser la carotide, nous avons implémenté la gestion des événements clavier dans le script. Cette approche permet d'associer différentes actions sur l'image à des touches spécifiques du clavier, facilitant ainsi la navigation et l'interaction avec les images médicales.

Voici quelques exemples de commandes clavier que nous avons intégré :

- Les touches 'W' et 'S' permettent d'augmenter ou de diminuer l'angle de rotation de l'image.
- Les touches '+' et '-' permettent de naviguer à travers les différentes coupes de l'image.
- Les touches 'A' et 'D' permettent d'ajuster la position du centre de rotation de l'image.

En utilisant OpenCV pour l'affichage des images et la gestion des interactions, nous avons pu créer une expérience utilisateur plus fluide et efficace, facilitant ainsi la localisation de la carotide dans les images médicales. De plus, cette approche a également permis une meilleure intégration avec les autres fonctionnalités liées au traitement d'image déjà présentes dans le script.

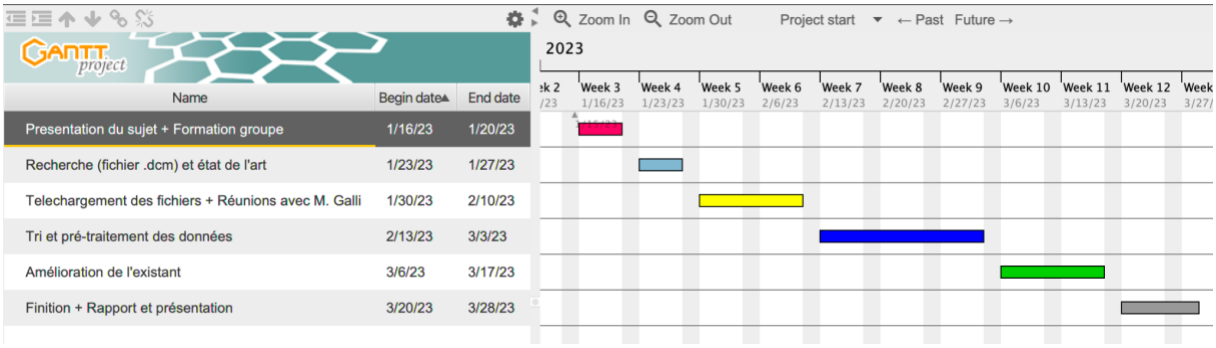
Gestion de projet

Répartition des tâches et organisation

Pour mener à bien le projet, nous avons travaillé en étroite collaboration, en nous répartissant les tâches de manière équilibrée pour optimiser notre temps et nos compétences respectives. Voici un résumé de la répartition des tâches et de l'organisation de notre travail :

1. Analyse et optimisation du script rotation_script.py
    - Nous avons étudié ensemble le script pour en comprendre le fonctionnement et déterminer les axes d'amélioration possibles.
2. Résolution des problèmes liés au téléchargement et à l'organisation des dossiers patients
    - Pour cette partie, Adams, s'est chargé de télécharger tous les fichiers .cab et les organiser correctement sur notre espace de travail.
3. Création du script organize_dicom.py
    - Nous avons travaillé conjointement sur le développement du script en nous répartissant les différentes fonctions à implémenter pour organiser et classifier les fichiers DICOM.
4. Analyse des résultats et pré-traitement des données
    - Une fois le script organize_dicom_files.py finalisé, nous avons utilisé ensemble le script analyze_result.py pour étudier les résultats et identifier les données exploitables.
    - Nous avons décidé conjointement de la méthode à adopter pour traiter les fichiers DICOM sans attribut "SliceLocation" et avons partagé le travail pour l'utilisabilité des scans concernés.
5. Amélioration du script rotation_script.py
    - Nous avons partagé les tâches pour exploiter les différentes possibilités d'amélioration par rapport aux librairies et outils existants, ce qui nous a amené à avoir deux autres scripts avec les améliorations que l'on a jugées utiles.
6. Rédaction du rapport
    - Enfin, nous avons réparti la rédaction des différentes sections du rapport, en nous relisant mutuellement pour garantir la qualité et la cohérence de notre travail.

Planning



Communication et suivi

Durant tout le projet, nous avons organisé des réunions régulières afin de faire le point sur l'avancement de nos tâches respectives, de discuter des problèmes rencontrés et de prendre des décisions. En complément de ces réunions, nous avons utilisé des outils de collaboration tels que GitHub et Google Docs pour suivre l'évolution du travail et centraliser les informations et les documents relatifs au projet.

Grâce à cette communication régulière, nous avons pu mener à bien toutes les étapes du projet. Pour gagner du temps, nous avons effectué certaines tâches indirectement liées au projet, comme le téléchargement des dossiers patients et la rédaction du rapport, en dehors des heures de travail dédiées au projet.

Nous avons également veillé à diviser le travail autant que possible afin d'éviter de travailler sur les mêmes tâches. Cette approche nous a permis d'avoir de nouvelles idées distinctes sur le projet, ce qui a enrichi nos discussions lors des réunions régulières.

Conclusion

Le Projet Deep Bridge a été un défi ambitieux mais motivant, qui nous a demandé une implication totale. Bien que nous n'étions que deux, nous avons beaucoup appris et acquis des compétences tant sur le plan médical que sur le plan informatique. Ce projet nous a poussés à effectuer des recherches approfondies pour trouver des solutions innovantes et efficaces pour améliorer la détection de la sténose carotidienne.

Nous avons rencontré des difficultés lors de la sélection du logiciel à améliorer, ce qui a entraîné une perte de temps. Cependant, nous sommes convaincus que notre travail et nos recherches aideront les générations futures à avancer plus rapidement et efficacement dans ce domaine. Nous espérons que nos conclusions seront utiles pour développer des modèles d'apprentissage automatique plus précis et plus fiables dans la détection de la sténose carotidienne.

Nous souhaitons remercier M. Galli pour sa confiance en nous en nous confiant ce projet ambitieux, ainsi que le CHU de Nice pour nous avoir fourni les dossiers médicaux nécessaires. Nous espérons que les résultats de nos analyses pourront contribuer à identifier les dossiers incomplets afin de les enlever des dossiers à traiter dans le cadre de ce projet.

En conclusion, le Projet Deep Bridge a été une expérience enrichissante pour nous deux et nous sommes convaincus que ce projet ouvre la voie à de nouvelles avancées dans l'utilisation de l'intelligence artificielle pour améliorer la santé et la qualité de vie des patients.

Sources

CNN using DICOM files of CT Medical Images

https://www.kaggle.com/code/rinichristy/cnn-using-dicom-files-of-ct-medical-images

Image filtering tutorial - https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html

What is a dicom file – Leslie Wubbel https://www.youtube.com/watch?v=eLS9nDVJx5Y

Dataset Reading (pydicom.filereader)
https://pydicom.github.io/pydicom/stable/reference/fileio.read.html

U-Net: Convolutional Networks for Biomedical Image Segmentation - Olaf Ronneberger, Philipp Fischer, and Thomas Brox https://arxiv.org/pdf/1505.04597.pdf

Blood vessel segmentation for neck and head computed tomography angiography - Anders Hedblom
http://www.diva-portal.org/smash/get/diva2:667230/FULLTEXT01.pdf

Multimodality carotid plaque tissue characterization and classification in the artificial intelligence paradigm: a narrative review for stroke application

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8350643/