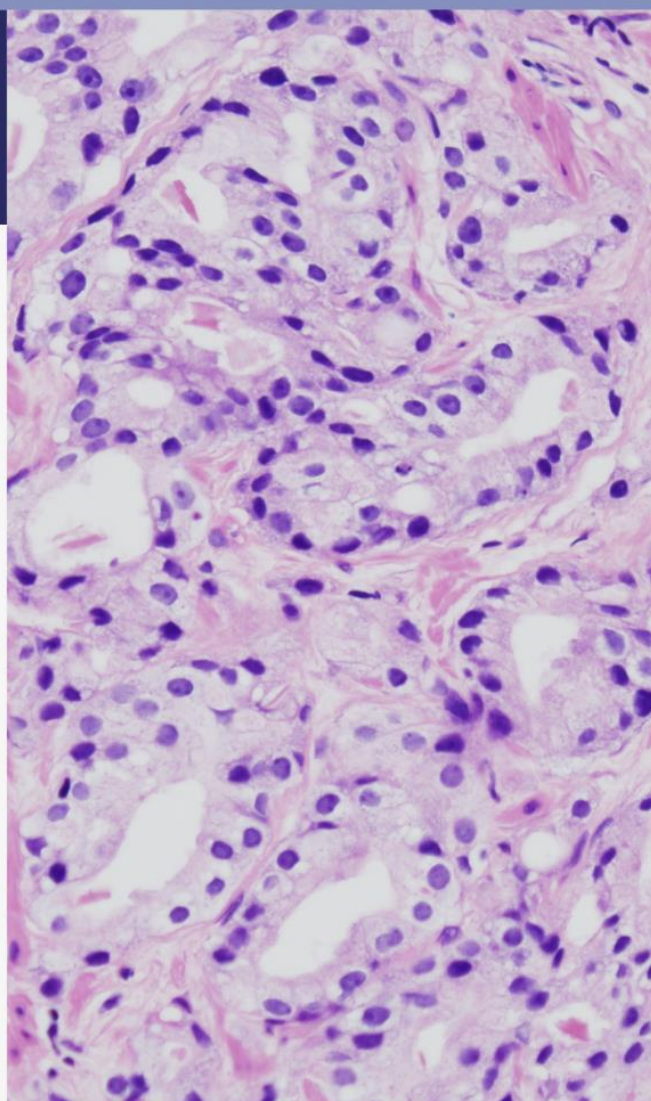
JUNIA ISEN

Rapport technique du second semestre :

Classification d'images microscopiques de tissus cancéreux à l'aide de l'intelligence artificielle

GROUPE 18 - MASTER 1



ETUDIANTS :

Marine GEORGES
Harith JADID
Carlos JIMÉNEZ GARCÍA
Michel SAUVAGE
Adrien Junior TCHUEM TCHUENTE

REFERENTS PROJET :

Bilel GUETARNI
Feryal WINDAL

2022-2023

Introduction - Rappel du contexte

Comme vu au premier semestre, le lymphome non hodgkinien (LNH) est un cancer agressif qui affecte les systèmes lymphatique et immunitaire. Le lymphome diffus à grandes cellules B (DLBCL), qui représente 30 à 40 % des cas de LNH dans le monde, est le sous-type le plus courant et peut être mortel s'il n'est pas traité. Deux versions principales peuvent être trouvées pour ce sous-type : ABC (Activated B-Cell) et GCB (Germinal Center B-cell), qui ont respectivement un taux de mortalité plus élevé et plus faible.

Dans le cadre de notre projet, nous avons eu pour objectif d'implémenter et d'entraîner un système de Machine Learning performant et capable de classifier, à partir des biopsies d'un patient donné, le sous-type de lymphome diffus à grandes cellules B (ABC ou GCB) dont il est atteint. Pour cela, il nous a été fourni des biopsies sous la forme de WSIs (Whole Slide Images) qui proviennent d'environ 160 patients et qui ont été annotées par des anatomopathologues.

Ce document développe des différentes étapes de la mise en œuvre technique du projet. La création du dataset est abordée dans une première partie, suivie de la mise en place des modèles de classification du tissu (cancéreux ou normal) et du sous-type (ABC-GCB). Une explication détaillée du prototype final est également proposée.

I. Première partie : création du dataset

Dans un premier temps, il a fallu créer le dataset pour le projet. Cette étape a été essentielle afin de garantir le déroulement de la suite. En effet, la manipulation des données est plutôt complexe car pour chaque patient, plusieurs WSIs peuvent être associées. De plus, des annotations pour chacune de ces WSIs ont été fournies et représentent soit les zones cancéreuses sur l'image, soit les zones non cancéreuses. Il a donc fallu faire preuve de rigueur pour organiser l'ensemble de ces données d'une manière optimale et dans des dossiers adéquats.

Le dataset a été créé selon les étapes qui suivent. Tout d'abord, les WSIs étant de très lourdes images, nous avons eu besoin d'utiliser une bibliothèque capable de les ouvrir et de les utiliser. Pour cela, nous avons choisi d'utiliser Openslide[1]. Une autre option testée était d'utiliser cv2[2]. Cependant, cette bibliothèque nous donnait accès à des parties de l'images non voulues (étiquettes, bord de la lame...), et qui n'étaient pas présentes avec Openslide. De plus, Openslide permet de sélectionner le niveau de détail/zoom que l'on souhaite, ce qui est avantageux pour réduire la durée de traitement des images par l'ordinateur.

Pour chaque image, on sépare le tissu du fond avec un masque créé à l'aide de la bibliothèque cv2. Le masque est réalisé grâce à un seuil de coloration de l'image encadré entre deux valeurs (couleurs trop sombre ou claire) déterminées manuellement grâce à des tests. Cette méthode permet de sélectionner uniquement les zones de tissus (80% de tissu) et sans artefacts ou zones trop sombres.

Ensuite, on utilise un autre masque pour identifier, cette fois-ci, les zones d'annotations. On récupère depuis un fichier .geojson les coordonnées des annotations d'anatomopathologistes afin d'identifier les zones dans notre image. Pour cela, on dessine les formes selon les coordonnées disponibles et on les remplit de blanc. Tout ce qui est en dehors est donc noir. Pour cette étape, la bibliothèque cv2 a été également utilisée.

Enfin, on extrait les patch selon la méthode suivante. On regarde l'image en se déplaçant patch par patch. Pour chaque patch, on procède par étapes. On utilise le masque de tissu. Si le patch est situé dans

[1] https://openslide.org/api/python/
[2] https://pypi.org/project/opencv-python/

une zone blanche (zone de tissu), on passe à l'étape suivante du masque des zones d'annotations. Il est important de noter qu'il y a deux types d'annotations : celles qui entourent une zone de cancer (extérieur = non cancéreux) et celles qui entoure une zone non cancéreuse (extérieur = cancéreux). Ceci est bien pris en compte dans l'algorithme. On sauvegarde ensuite le patch dans le dossier cancéreux ou normal selon ses caractéristiques. Les patch sont sauvegardés à un niveau de zoom 1 selon Openslide, ce qui permet d'avoir une bonne résolution d'image.

Chemin d'accès = ProjetM1/data_saved/n° patient/nom slide WSI/cancéreux ou normal/patch.png

II. Deuxième partie : classification

1. Grouper les données en sets de train, de validation et de test

Avant de commencer la classification, il est nécessaire de séparer les données en différents sets. Pour cela, nous nous sommes basés sur les patients et leur nombre de patchs. Bien que cela ait légèrement changé par la suite, nous avions tout d'abord choisi de séparer les patients en trois set : train (70%), validation (15%) et test (15%). Pour réaliser la répartition, nous nous sommes basés sur le nombre de patchs en créant des seuils. Cependant, un patient peut avoir une partie de ses patchs en-dessous et une autre au-dessus du seuil fixé. Dans ce cas de figure, le patient est assigné à la catégorie suivante pour éviter les doublons. Trois dossiers sont créés dans le répertoire de travail afin de ranger les patients en fonction de leur catégorie. Ainsi, dans chaque dossier train, validation et set, on a donc des liens symboliques vers les dossiers des patients pour éviter de déplacer les patchs préalablement enregistrés.

2. Modèle pour la classification des patchs cancéreux/normal

Plan A - Développement du CNN :

Après les recherches effectuées dans la première partie du projet en décembre, nous avons choisi de développer un CNN qui était initialement basé sur le modèle SEF utilisé dans l'article « A Deep Learning Approach for Breast Invasive Ductal Carcinoma Detection and Lymphoma Multi-Classification in Histological Images » (Brancati et al. 2019)[3], car il fournissait la "accuracy" la plus élevée (0.97). Cependant, nous n'avons pas pu trouver la structure de ce modèle et avons décidé d'essayer la deuxième option, qui consistait à utiliser un modèle ResNet dont la métrique "accuracy" n'était inférieure que de 2% à celle du modèle précédent.

ResNet (Residual Networks) est une architecture CNN qui se caractérise par sa capacité à former des réseaux neuronaux très profonds avec un petit nombre de paramètres. Elle est largement utilisée dans le traitement des images. Ces réseaux neuronaux sont constitués de blocs résiduels (que l'on pourrait qualifier d'unité de base du ResNet), qui sont regroupés et donnent lieu à des 'stacks'. Une `stack' peut contenir différentes quantités de blocs résiduels en fonction de la profondeur souhaitée du réseau. De cette manière, nous pouvons trouver différents types de ResNet, le plus grand nombre ayantune plus grande profondeur et la plus grande capacité à apprendre des représentations de caractéristiques plus complexes et à s'adapter aux paramètres d'apprentissage.

[3] Brancati, Nadia, Giuseppe De Pietro, Maria Frucci, et Daniel Riccio. 2019. « A Deep Learning Approach for Breast Invasive Ductal Carcinoma Detection and Lymphoma Multi-Classification in Histological Images ». *IEEE Access* 7: 44709-20. https://doi.org/10.1109/ACCESS.2019.2908724.

Comme nous disposions d'un très grand ensemble de données pour le training, nous avons décidé, dans un premier temps, de tester ResNet50. Cependant, comme le nombre de paramètres à ajuster était trop élevé pour le temps dont nous disposions, nous avons décidé de réduire sa taille et de développer un ResNet14, en privilégiant le temps par rapport à la précision et à l'exactitude (accuracy). Les résultats obtenus pour le meilleur modèle étaient une accuracy de 86,7% et une précision de 71%.

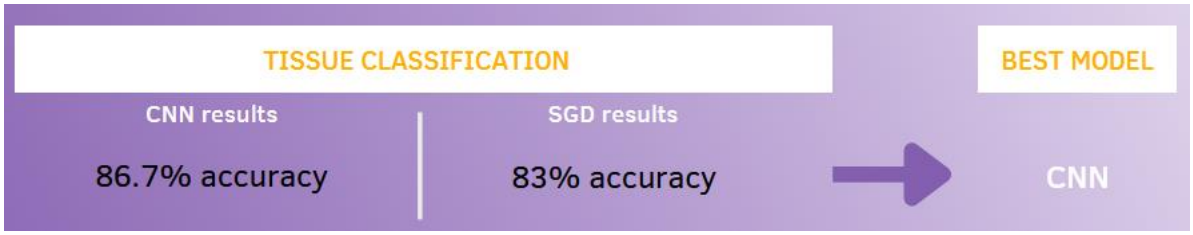
Il convient de noter que nous aurions aimé avoir le temps de continuer à tester le ResNet et d'ajuster les paramètres, voire d'utiliser PyTorch, un cadre qui nous aurait permis de déboguer le code et d'expérimenter différents modèles plus facilement, ainsi que de personnaliser les différentes strates du ResNet, créant ainsi un modèle beaucoup plus spécifique pour la tâche à accomplir.

En parallèle, nous avons également décidé d'entraîner un Stochastic Gradient Descent (SGD), utilisé pour la classification du type de cancer, mais adapté, dans ce cas, pour distinguer les tissus cancéreux des tissus non cancéreux, afin d'avoir une alternative au cas où ResNet ne produirait pas les résultats escomptés.

Plan B - Développement du SGD :

La deuxième option convenue pour classifier les patch cancéreux et normaux était donc d'utiliser un SGD, qui est une approche de modèle de classification linéaire. Le SGD appliqué ici est le même que celui initialement prévu pour la partie suivante ; seule l'entrée diffère entre les deux versions. Ici, on aura donc en entrée pour l'entraînement des patchs labellisés cancéreux et normal, en fonction du dossier duquel ils proviennent. En sortie du modèle, seuls les patchs prédits comme cancéreux sont conservés pour l'étape suivante. Son fonctionnement complet est détaillé dans la section suivante.

Les meilleurs résultats pour cette étape de classification sont les suivants :



1 - Meilleurs résultats

3. Modèle pour la classification du sous-type ABC ou GCB

Pour cette étape (ainsi que la précédente), nous avons utilisé un SGD, qui s'apparente à un SVM. La particularité de ce modèle SGD est qu'il est capable de fonctionner en réalisant des mini-batchs (*partial_fit)*. Cela est essentiel pour ce projet étant donné la quantité des données à traiter. De plus, nous avions souligné dans la première partie de ce projet en décembre l'utilisation intéressante du modèle SVM pour la classification d'images, selon le document « Multi-radial LBP Features as a Tool for Rapid Glomerular Detection and Assessment in Whole Slide Histopathology Images » (Simon et al. 2018)[4].

Tout d'abord, on récupère en entrée les données nécessaires pour le modèle. Dans l'étape précédente, on utilisait des patchs labellisés cancéreux ou normal. Ici, on utilise les patchs cancéreux uniquement. Ces derniers sont labellisés avec le sous-type ABC ou GCB selon les informations fournies

[4] Simon, Olivier, Rabi Yacoub, Sanjay Jain, John E. Tomaszewski, et Pinaki Sarder. 2018. « Multi-Radial LBP Features as a Tool for Rapid Glomerular Detection and Assessment in Whole Slide Histopathology Images ». Scientific Reports 8 (1): 2032. https://doi.org/10.1038/s41598-018-20453-7.

au préalable pour le projet dans un fichier « .csv ». Il est important de noter que les données ont finalement été réparties en deux sets train (85% = dossiers de patients train + test rassemblées) et validation (15%). Il n'était pas significatif de garder un set test car il n'était pas possible de comparer les résultats de ce projet avec un autre projet déjà réalisé auparavant avec une méthode et un objectif similaire. Les informations relatives à chaque patch sont stockées dans un tuple, ce qui permet de les manipuler aisément.

Ensuite, afin d'entraîner le modèle, un système de batch a été créé. Le nombre de batch et la taille de ceux-ci ont été déduits dans chaque cas du nombre de patch que l'on a au total afin d'optimiser leur utilisation. Pour effectuer le *partial fit*, il est nécessaire d'utiliser une boucle afin d'aller de batch en batch. A l'intérieur de celle-ci, on réalise tout d'abord un Local Binary Pattern (LBP) sur chaque patch puis on fait un histogramme. On récupère ainsi les features de l'histogrammes (concaténé) ainsi que les labels. Pour information, dans le cas où un patch est mal enregistré ou n'a pas les bonnes dimensions, celui n'est pas utilisé pour la suite du programme. On a accès au chemin du patch et au label correspondant grâce au tuple fait précédemment. On effectue ensuite l'entraînement grâce au *partial fit* avec les features selon les classes de classification (cancer ou sous-type selon le cas). On supprime ensuite le contenu du batch pour libérer de l'espace de la RAM.

Grâce à la bibliothèque joblib, on sauvegarde le modèle entrainé. Bien qu'il n'ait pas été possible de toutes les créer par manque de temps, nous avions tout de même choisi idéalement d'utiliser les métriques suivantes pour évaluer le modèle :

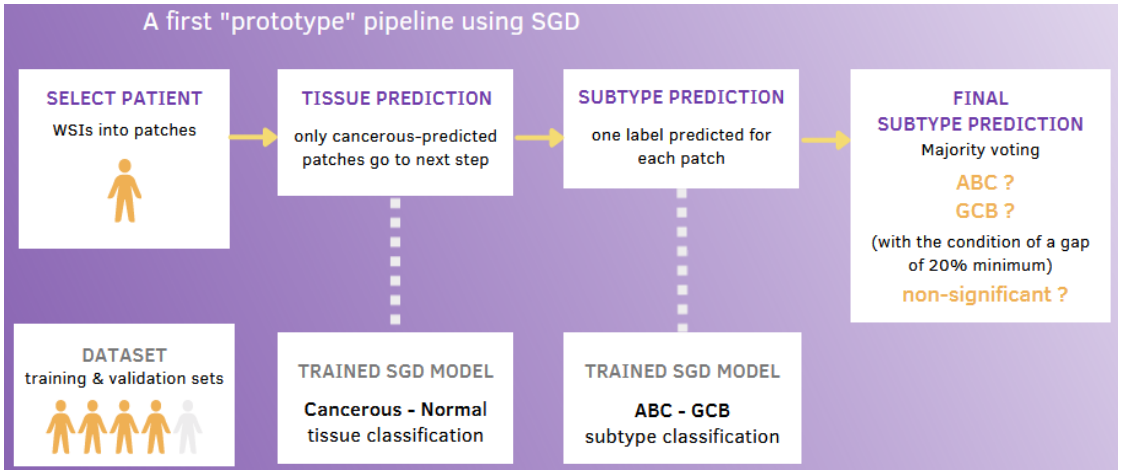
- Accuracy : le pourcentage de bonnes réponses.
- F1-Score : calcule la moyenne de la précision et du recall.
- Confusion matrix : affiche dans une matrice le nombre de True Positives, True Negatives, False Positives et False Negatives.
- ROC AUC : permet de savoir s'il y a une différence marquée entre les deux classes.

Avec l'accuracy, on peut déterminer si le modèle se trompe beaucoup ou non. Le F1-score permet de déterminer pour chaque classe quelles sont les performances du modèle. La matrice de confusion permet de visualiser, par classe, combien de fois le model s'est trompé ou a déterminé le bon résultat. Pour finir, le ROC AUC permet de préciser si le modèle arrive bien à différencier les deux classes. On peut noter qu'il aurait été intéressant d'accéder aux métriques par patient pour améliorer la qualité de l'information.

Par ailleurs, deux loss ont été testées pour le modèle : hinge (SVM linéaire) modified_huber (apporte une tolérance aux valeurs aberrantes ainsi que des estimations de probabilité). Ces dernières donnaient des résultats assez similaires. Dans un deuxième temps, il a aussi été tenté d'équilibrer le jeu de donné ABC-GCB afin d'avoir de meilleures performances.

Après différents tests, les résultats finaux de l'entraînement et de la validation du programme ont montré que l'on obtient une accuracy de 70% pour le meilleur modèle, avec un dataset équilibré mais qui ne donnais pas de résultat constant. Pour améliorer cela, nous avons tenté deux différentes approches. La première consistait à utiliser des batchs plus nombreux mais en entraînant une plus petite quantité de patchs à la fois car la supposition ici était que le partiel fit était tellement grand qu'il rendait les anciens entrainement négligeable. Cela n'a pas donné de résultats améliorés. Une autre méthode testée est celle d'utiliser un Support Vector Classification (SVC), en entraînant en une seule fois 100 000 patchs. En effet, l'hypothèse ici était que l'on ne peut pas entraîner tout le dataset à cause de sa taille très grande (1700000 patchs). Ainsi, peut-être qu'avec 100 000 pris aléatoirement dans le dataset, on peut obtenir de meilleures résultats. Les résultats n'étant pas encore disponibles, il n'est pas possible de conclure sur cet essai.

III. Programme final et explications techniques



2 - Schéma d'ensemble du prototype proposé

Enfin, une mise en commun de chacune des étapes a été réalisée pour réaliser un premier prototype de l'ensemble du programme. Ainsi, en entrée du programme, on choisit un patient dont on veut connaître le sous-type de cancer. Pour que le programme fonctionne comme prévu, les WSIs relatives au patient doivent être enregistrées dans un dossier nommé par le numéro d'identification de ce dernier. Pour chacune de ses slides, le programme va réaliser l'ensemble des étapes discutées au préalable dans une boucle et la décision finale du modèle se fera en tout dernier.

Pour rappel, la première étape consiste à effectuer les patchs de 512x512 pixels de la WSI. Ces derniers sont sauvegardés pour garantir le bon fonctionnement du code dans le cas où un patient aurait beaucoup de WSIs. Les patchs sont tous enregistrés ensemble dans un même dossier. Le numéro du patient et le chemin de chacun des patchs est conservé dans un tuple.

Ensuite, le premier modèle de classification SGD se lance. Pour chaque batch, le LBP est réalisé pour créer un histogramme de features puis on effectue la prédiction. En sortie de ce premier modèle, on crée un tuple conservant uniquement les informations des patch prédits comme cancéreux.

Pour le deuxième modèle de classification, on procède donc de la même manière, mais cette fois en utilisant le tuple créé en sortie du premier modèle. Pour chaque patch, on refait un LBP, puis le modèle prédit par batch le sous-type pour chaque patch. Ces prédictions sont enregistrées dans une liste, qui se complète au fur et à mesure des WSIs du patient.

Enfin, une fois que toutes les WSIs du patient sont passées dans le programme, on utilise la liste des prédictions ABC ou GCB. En fonction du nombre de patch ABC et GCD prédits, on pourra déterminer quel est le type de cancer du patient grâce à la méthode de vote par majorité. Dans le cas où on obtient une différence de moins de 20% entre le total de ABC et le total de GCB prédits, aucun résultat n'est donné car il est considéré comme non significatif.

D'autre part, par manque de temps, il n'a cependant pas été possible de tester davantage le programme avec le CNN ou d'autres paramètres et métriques. Néanmoins, cela apporte une première confirmation de la mise en commun possible des modèles créés et pourrait être réutilisée pour améliorer l'ensemble de la réalisation.