RAPPORT DU
PROJET DE CASSIOPÉE
Année 2023 - 2024

Sujet : Neural network-based model for the identification of Bladder Cancers subtypes

Lien du GitHub : https://github.com/Ominican/Cassiopee

Institut Polytechnique de Paris
Télécom SudParis

Auteurs :

Dorard Valentin
Khoury Rémi

Encadrantes :

Mme. Zehraoui Farida
Mme. Creux Constance

Table des matières

Résumé

Notre projet porte sur l'identification des sous types de cancers de la vessie grâce aux réseaux de neurones en graphes (GNN). Il s'inscrit dans la continuité d'un autre travail effectué sous la direction de nos encadrants à l'université d'Evry. Ce travail a été mené par une stagiaire de l'université et cherchait à identifier les sous types de cancers de la vessie. Notre projet repose sur l'utilisation de données omiques, cliniques et d'images pour identifier le type de cancer de la vessie dont souffrent les patients. Les données omiques sont des données portant sur les protéines du corps ou sur les gènes du patient à l'échelle microscopique. Ces données sont utilisées pour diagnostiquer ou même prédire des maladies. Les données cliniques, quant à elles, sont issues d'examens avec le patient, sont donc qualifiées de données macroscopiques. Ces données comportent de très nombreux attributs, ce qui les rend difficiles à analyser, mais aussi difficiles à traiter par ordinateur : l'utilisation du machine learning est appropriée. Notre approche a consisté à utiliser un réseau de neurones en graphe, le GATv2 (Graph Attention Network v2)[1] à travers deux approches : une approche inductive et une approche transductive.

Notre projet explore une nouvelle manière d'identifier ce type de cancer, et ce, en utilisant des données réelles. Il ouvre de nouvelles perspectives quant à l'utilisation des réseaux de neurones en graphe dans le milieu médical, particulièrement pour des applications où les caractéristiques moléculaires de chaque patient sont à prendre en compte.

1 Introduction

Contexte applicatif

Le cancer de la vessie est le 7^e cancer le plus fréquent en France. Plus de 13000 cas ont été recensés en 2018, touchant les hommes dans 81% des cas, tandis que le taux de survie à 5 ans de ce cancer est estimé autour de 40%, avec un taux de mortalité plus important chez les femmes. Ce type de cancers est généralement détecté chez les personnes âgées (+70 ans), et les chances de survie diminuent avec l'âge. Dans certains rares cas (entre 10% et 20%), le cancer peut devenir invasif, les cellules tumorales s'infiltrent alors dans les muscles, et de cela, le cancer se transforme en MIBC (Muscle Invader Bladder Cancer).

Plusieurs classifications existent pour cette maladie. Nous utiliserons celle adaptée à notre base de données, c'est-à-dire une proposition de classification proposée par Kamoun et al[2], qui décrit en 6 types les cancers que les patients peuvent développer. Chaque patient est donc victime d'un des types de cancers suivant : Luminal Papillary (LumP), Luminal NonSpecified (LumNS), Luminal Unstable (LumU), Stroma-Rich, Basal/Squamous (Ba/Sq), Neuroendocrine-like (NE-like).

Un prétraitement des données effectué par nos encadrantes nous a permis d'avoir à disposition des données omiques et cliniques déjà utilisables dans notre modèle. Nous avons 404 patients à notre disposition dans notre base de données. Leurs données omiques ont été obtenues d'après un traitement sur les images et les données cliniques d'après les résultats d'examen de chaque patient. L'utilisation de données omiques et cliniques pour la mise en place d'un réseau de neurones en graphe représentent une nouvelle approche pour identifier les cancers de la vessie et un réel intérêt pour l'avancement des techniques d'Intelligence Artificielle dans le domaine de la santé. Les données omiques sont utilisés pour représenter les nœuds de notre graphe, et les données cliniques et pathologiques seront utilisées pour les arêtes de notre graphe.
Le modèle que nous avons construit se repose sur les toutes dernières avancées en matière de graphe neural network. Nous utiliserons donc un graphe d'attention, qui introduit la notion de coefficient d'attention décrivant l'importance d'un nœud au sein du graphe en plus des fonctionnalités d'un GNN classique.

Enfin, notre modèle comparera deux méthodes de construction de graphe différentes, l'induction et la transduction.

- La transduction consiste à faire rencontrer à notre modèle les données de tests et les données d'entrainement, pour pouvoir piocher dans toutes les données existantes pour attribuer une classe à une nouvelle donnée. Cependant, à chaque fois que nous ajoutons une donnée au graphe, nous devons réentrainer tout le modèle.

- L'induction consiste à faire ne faire rencontrer au graphe que les données d'entrainement, sous formes de sous graphes, et de prédire les étiquettes des données de tests dans le graphe de test.

Ces deux méthodes seront mises en concurrence pour déterminer celle qui présentera les meilleurs résultats. Nous ajusterons également les hyperparamètres pour optimiser l'apprentissage et l'efficacité du modèle

2 Etat de l'art

2.1 Une implémentation novatrice

Les modèles actuellement en vigueur intègrent le plus souvent les données omiques et les images, mais aucun modèle ne combine les deux ensemble. C'est toute l'originalité de notre travail. Nous voulons combiner les informations apportées par les données pathologiques et cliniques (sous forme d'images) du cancer de chacun de nos patients ainsi que leurs caractéristiques omiques dans le même graphe.

L'utilisation du deep learning avec les images est une méthode classique pour les soucis de classifications médicales. Elle est largement démocratisée et son efficacité n'est plus à prouver. En combinant cette pratique aux informations que l'on possède sur le génome d'une personne, et en calculant donc les similarités entre les patients du point de vue des images de leurs cancers et de leurs génomes, nous pouvons atteindre un degré de précision très satisfaisant. Le concept de graphe sied parfaitement à cette idée puisqu'il connecte tous les patients similaires entre eux.

2.2 Les GNNs et les GATs

Les réseaux de neurones en graphe (GNN) et les réseaux de neurones attentionnels en graphe (GAT) représentent une avancée significative dans le traitement des données structurées sous forme de graphes. Ces modèles ont trouvé des applications dans divers domaines, notamment les réseaux sociaux, la biologie, la chimie, et l'analyse des réseaux de transport.
Les réseaux de neurones traditionnels tels que les CNN peinent à traiter efficacement les données de graphe complexes. Les GNNs sont donc bien plus habilités à traiter les problèmes complexes imbriquant des relations entre les différentes données à classifier.

Les GNN sont des modèles d'apprentissage profond conçus pour traiter les données structurées sous forme de graphes. Les graphes sont constitués de nœuds (ou sommets) et d'arêtes (ou liens) qui relient ces nœuds. Chaque nœud et chaque arête peut avoir des attributs ou des caractéristiques associées.
Les GNNs fonctionnent selon le principe de l'agrégation de messages, où chaque nœud dans le graphe agrège les informations de ses voisins ainsi qu'une matrice de poids variable, pour mettre à jour sa propre représentation. Ce processus se fait de manière itérative de la manière suivante :

- Une fois les nœuds initialisés avec les informations voulues, nous pouvons réaliser des combinaisons linéaires des informations voisines ainsi que d'une matrice de poids d'entrainements.
- on met le tout dans une fonction d'activation telle que, si l'on note h_i l'information de base, W le matrice de poids d'apprentissage, σ la fonction d'activation et h'_i la matrice résultante, on obtient $h'_i = \sigma(\sum_j W * h_j)$
- on répète cette étape sur plusieurs couches pour diffuser les informations partout dans le graphe.

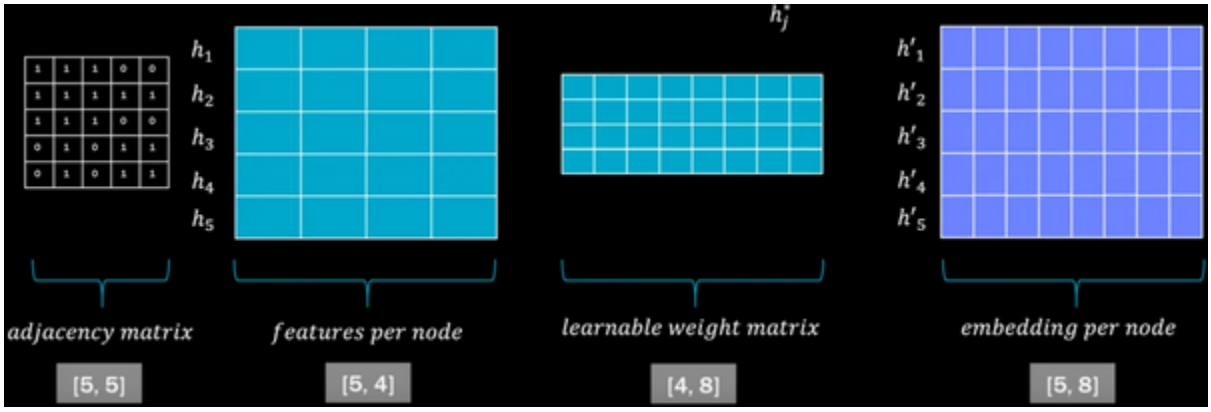

Figure 1: Modèle de calcul d'un GNN [3]

Introduits par Veličković et al. en 2017, les graphes d'attention (GAT) permettent une meilleure modélisation des structures complexes et hétérogènes des graphes. Les graphes d'attentions sont un héritage des graphes neuronaux (GNN), où l'on pondère l'information donnée par chacun des voisins d'un noeud donné grâce à un coefficient d'attention, e_{ij} calculé en fonction de la multiplication de la matrice de poids d'entrainement et des informations de chacune des deux parties: $e_{ij} = a(W * h_i, W * h_j)$, où a est le coefficient calculé, lors de l'itération précédente et initialisée uniformément. h'_i devient alors $h'_i = \sigma(\sum j e_{ij} * W * h_j)$.

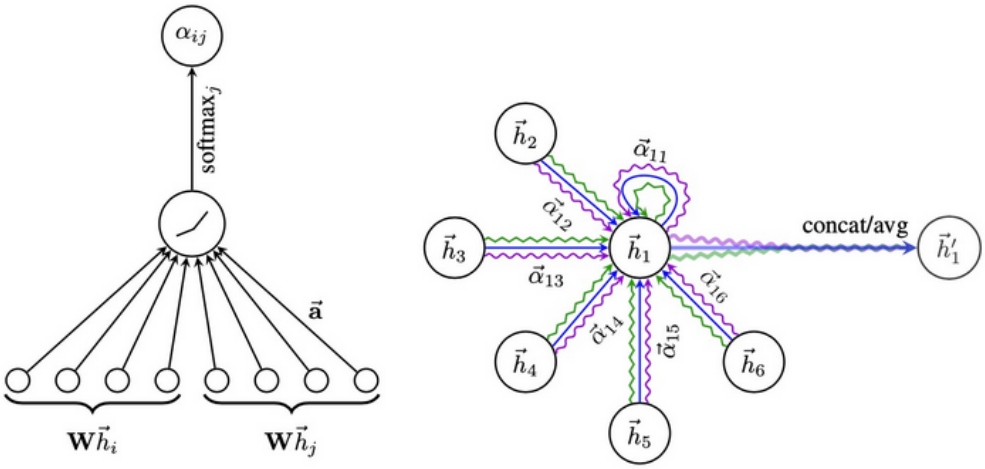

Figure 2: Exemple de calcul de coefficient d'attention

[2]

2.3 La transduction et l'induction

La transduction est une approche d'apprentissage où l'objectif est de prédire les labels des nœuds du graphe à partir d'un sous-ensemble de nœuds étiquetés. L'apprentissage, la validation et le test se font sur un seul graphe. Les GNN et GAT sont particulièrement bien adaptés pour la transduction, car ils exploitent les relations entre les nœuds pour propager l'information des nœuds étiquetés vers les nœuds non étiquetés.

L'induction, en revanche, concerne la généralisation du modèle à des graphes ou des parties de graphes non vus pendant l'entraînement. Les méthodes inductives doivent être capables de généraliser à de nouveaux nœuds ou graphes sans étiquette. Les GAT, avec leurs mécanismes d'attention, sont souvent mieux adaptés à cette tâche en raison de leur capacité à modéliser les interactions complexes et à généraliser au-delà des données d'entraînement.
Nous reviendrons dans ce rapport sur ces deux approches et sur leur implémentation dans ce projet.

3 Approche proposée

3.1 Prétraitement

Nous avons à notre disposition plusieurs datasets pour mener à bien notre projet.

Dataset	Description	Nombre de patients
labels_str.csv	Type de cancer dont le patient est atteint	406
patient_norm.csv	Données cliniques de chaque patient	412
node_embedding.csv	Données omiques de chaque patient	404

Table 1: Descriptions des datasets utilisés dans le projet

Comme indiqué dans le tableau, un traitement des données a été d'abord nécessaire pour s'assurer que les données correspondaient bien toutes au même set de patients, soit donc 404 patients au total.
Ensuite, il nous a été possible de remarquer que la distribution des types de cancer n'était pas homogène.

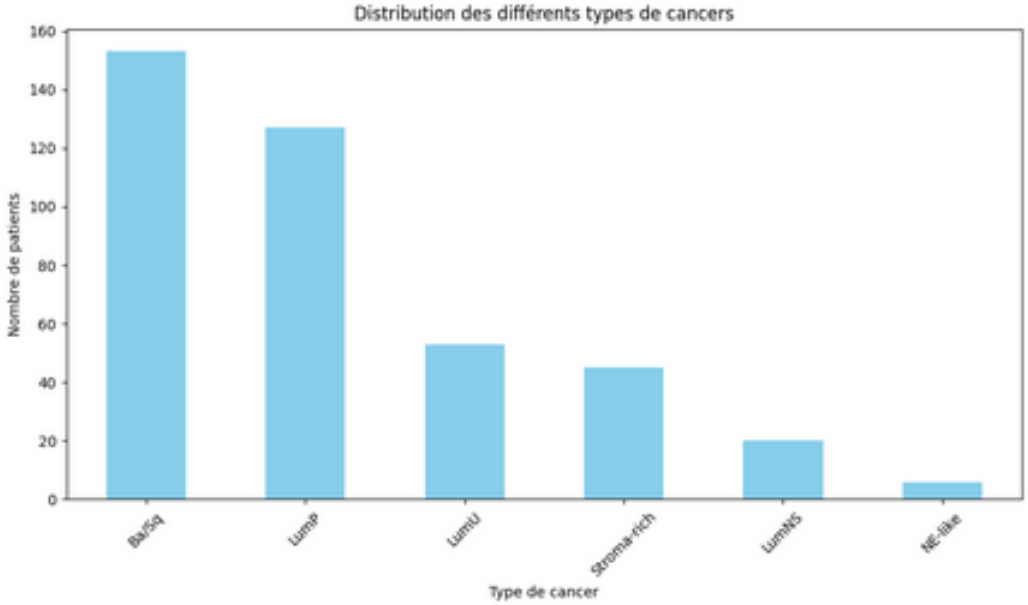

Figure 3: Répartition des types de cancers de la vessie

D'après la figure ci-dessus, on remarque que seulement très peu de patients sont atteints, par exemple, par le cancer NE-Like, et que la majorité des patients souffrent soit du cancer LumP, soit du cancer Ba/Sq. Cette observation nous a conduits à mettre en place un traitement des poids des données par classe pour s'assurer que notre modèle ne souffrirait pas d'un biais lié à la distribution des données. Ainsi, nous utiliserons la formule suivante pour initialiser les poids de chaque nœud :

$$\mu = \text{moyenne du nombre de patients}$$

n_i =nombre de représentants de chaque cancer

$$\boxed{w_i = \frac{\frac{\mu}{n_i}}{\Sigma\left(\frac{\mu}{n_j}\right)}(1)}$$

D'après cette formule, les poids alloués à chaque classe du modèle sont les suivants :

Types de cancers	**Poids alloué**
LumP	0,029
Ba/Sq	0.024
LumU	0.069
Stroma-rich	0.082
LumNS	0.184
NE-like	0.612

Table 2: Descriptions des datasets utilisés dans le projet

Après les premiers résultats obtenus, il est apparu que les deux dernières classes étaient difficilement identifiables. En effet, parmi les 404 patients, on compte à peine 20 patients labellisés LumNS (c'est-à-dire atteint du cancer LumU ou LumP, sans certitude) et 6 patients labellisés NE-Like. Cette faible représentation dans les données de ces deux classes pose des problèmes dans la stratification (pas assez de données pour effectuer la validation, l'entraînement et le test) mais aussi dans l'initialisation des poids du modèle (la classe NE-like étant très peu représentée, le modèle avait des difficultés à apprendre sur les autres données, car elles avaient toutes un poids trop faible par rapport à la classe NE-like). Nous avons donc décidé de ne traiter que les 4 premières classes, les deux dernières étant difficiles à inclure correctement dans le modèle. Nous nous retrouvons donc avec des nouveaux poids à attribuer à chaque classe (voir dans le tableau ci-dessous).

Types de cancers	**Poids alloué**
LumP	0,142
Ba/Sq	0.118
LumU	0.340
Stroma-rich	0.40

Table 3: Descriptions des datasets utilisés dans le projet

En plus, de cette mise à jour des poids, notons également que nous nous retrouvons désormais avec 378 patients dans notre base de données.

3.2 Construction de notre modèle

Le principe de notre approche consiste à créer un graphe à partir des données cliniques, pathologiques et omiques des patients. Ce graphe est constitué de nœuds représentant chaque patient (créés à partir de leurs données omiques respectives) ainsi que d'arêtes

(créées à partir des ressemblances sur les données cliniques de chaque patient). Cette approche nous permet donc d'évaluer la similitude sur l'aspect extérieur avec les images et les données cliniques, mais également sur les bilans génomiques avec les données omiques. La première étape de la mise en place des arêtes du graphe. Les arêtes sont construites à partir de la similarité des données cliniques des patients : on utilise la similarité cosine, ou le noyau cosine, qui calcule la similarité comme le produit scalaire normalisé de X et Y : $K(X, Y) =< X, Y > /(||X|| * ||Y||)$, sur les données normalisées L2.

Une fois cette similarité calculée entre les patients, une matrice de taille NxN (N=le nombre patients dans le graphe) contenant les liens entre les patients et leur taux de similarité est obtenue.
Ce calcul de la similarité est l'occasion pour nous de faire une précision importante sur le processus de création du graphe. En effet, il semble clair que lier tous les patients entre eux, quel que soit leur degré de similarité, n'est pas optimal. En plus d'augmenter la complexité du modèle, lier deux patients avec un très faible taux de similarité pourrait mener à une baisse de précision du modèle, les graphes d'attentions fonctionnant par propagation d'information entre les nœuds. Nous avons choisi de mettre en place un seuil de similarité, fixé lors de la formation de la matrice de similarité. Ce seuil est compris entre 0 et 1 et permet le cas échéant de choisir le niveau nécessaire de similarité entre les patients pour qu'ils soient liés dans le graphe. Dans la suite, ce seuil sera fixé à 0.5.

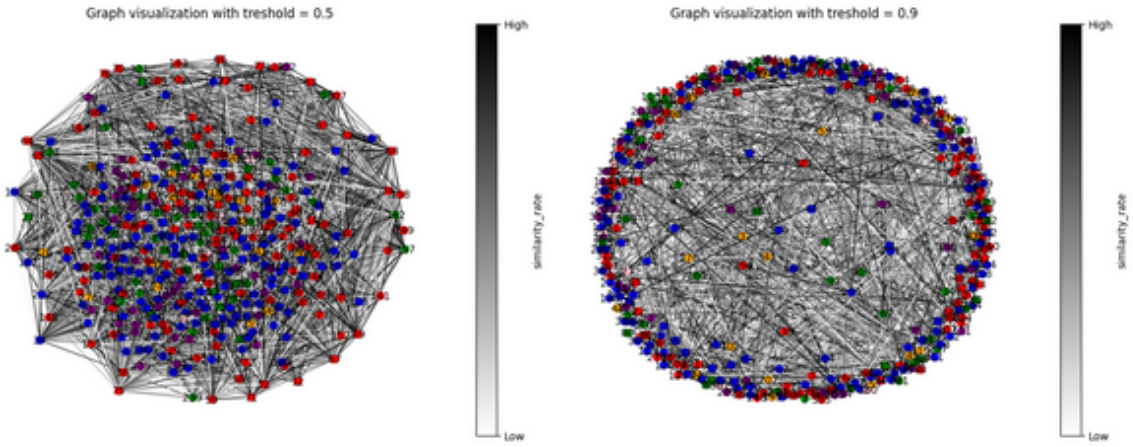

Figure 4: Visualisation du graphe

La figure 4 nous permet d'observer les différentes connexions entre les nœuds, en fonction du seuil de similarité choisi, on voit bien que pour un seuil normal, énormément de connexions se sont faites, ce qui permet d'avoir un graphe viable pour une bonne implémentation de notre méthode de classification. Cependant, lorsque le treshold devient trop grand, certains noeuds ne sont pas inclus dans le modele, puisqu'aucun de leur lien n'est suffisament grand pour leur permettre d'exister. Nous pouvons observer sur la figure si dessus que les noeuds se "repoussent" les uns les autres, ce qui nous donne un indice sur le fait que la variance entre les noeuds de classe différente augmente.
Une fois que le calcul de similarité a été effectué, nous effectuons la mise en place de la matrice de poids permettant d'attribuer à chaque nœud un poids spécifique à son type de cancer. Il s'agit ensuite de regrouper les données dans un objet "data" qui va centraliser

les informations contenues par les nœuds, les labels de chaque nœud, les arêtes ainsi que le nombre de classes à identifier.

3.3 Modèle par induction / transduction

Dans ce projet, nous avons choisi de mettre en place deux approches quant au mode d'entraînement et de test de notre réseau de neurones. Traditionnellement, un modèle dispose de données d'entraînement et de validation, ainsi qu'un set de données permettant de tester l'efficacité de ce modèle d'après les hyperparamètres choisis. Cette approche sera présenté dans la partie *Resultats*, mais n'est pas optimale *àpriori*. D'une part, dans notre cas, il s'agit de construire un graphe, c'est-à-dire de construire un ensemble de nœuds et d'arêtes qui comportent des relations entre eux, et des approches moins conventionnelles existent pour obtenir les meilleurs résultats. D'autre part, nous ne disposons que de très peu de données en réalité. Obtenir un modèle robuste avec seulement 404 données sur les patients, qui comportent peu de représentants de certains types de cancer, s'annonce difficile. Ainsi, il est nécessaire de formuler une approche différente.

Nous allons réaliser dans notre première méthode une division du graphe en plusieurs sous graphes, que cela soit pour l'entraînement, la validation ou le test. Cette méthode est l'approche inductive, qui va nous permettre d'évaluer efficacement la robustesse du modèle dans la généralisation à des nouveaux nœuds différents de ceux étudiés par le modèle.

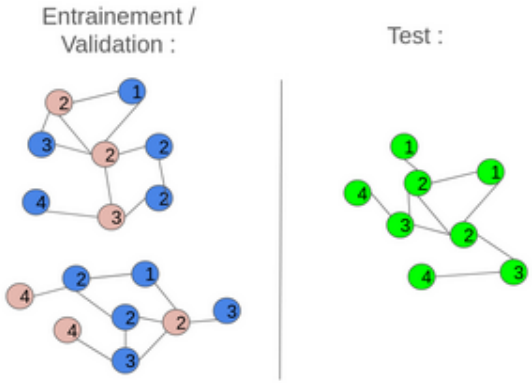

Figure 5: Exemple de modèle implémentant l'induction

Notre seconde méthode consistera à utiliser entièrement le graphe pour l'entraînement et la validation du modèle. Le modèle sera testé en cachant certains nœuds, c'est-à-dire en réintroduisant certains nœuds dans le modèle en supposant leurs labels inconnus. Cette approche est l'approche transductive et permet d'évaluer efficacement la performance d'un modèle sur des données proches des données d'entraînement.

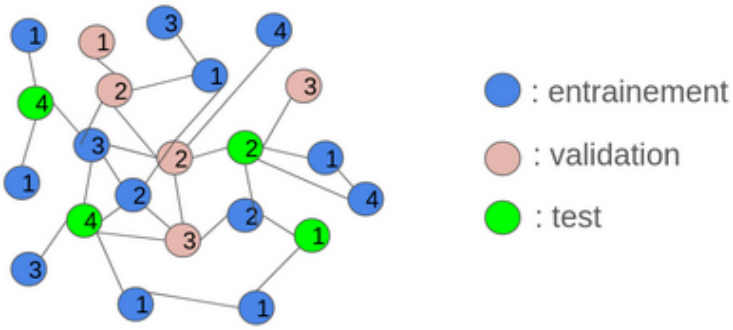

Figure 6: Exemple de modèle implémentant la transduction

L'utilisation et la comparaison entre ces deux modèles nous permettra d'obtenir des résultats plus complets et d'analyser en profondeur le comportement des graphes que nous avons mis en place.

3.4 Outils d'optimisation

Nous disposons de la stratégie que nous allons mettre en place pour produire nos modèles de réseaux de neurones en graphe, il s'agit maintenant de chercher les meilleurs hyperparamètres correspondant à notre modèle et à nos données. Pour ce faire, nous avons utilisé le framework *optuna* pour obtenir des hyperparamètres probants pour les deux approches.

Parmi les hyperparamètres que nous pouvons optimiser se trouvent : le learning rate, le weight_decay, le nombre de couches à appliquer dans notre modèle, le nombre de hidden_layers et le nombre de heads. Nous avons choisi d'optimiser les hidden_layers ainsi que les heads, le learning rate et le weight decay intervenant peu dans les performances des deux modèles. Quant au nombre de couches, il serait intéressant de chercher à modifier leur nombre, mais le temps de calcul nécessaire en plus des optimisations citées précédemment étant trop élevé, nous avons choisi de laisser au lecteur la chance d'explorer cette piste d'amélioration.

4 Résultats expérimentaux

4.1 Mise en place de l'optimisation des hyperparamètres

Avant de présenter les résultats des modèles, et dans la continuité de ce qui a été fait au paragraphe précédent, présentons l'étape d'optimisation des modèles. Que cela soit pour la transduction ou l'induction, les intervalles de valeur pour les hidden_channels et les heads ont été fixés comme suit :

- le nombre heads peut varier de 1 à 16
- le nombre hidden_channels peut varier de 16 à 30

4.1.1 Transduction

La transduction est l'approche qui demande le plus grand temps de calcul à l'ordinateur en raison du fait qu'elle n'utilise qu'un seul et même graphe avec beaucoup de liens entre les nœuds. Pour cette raison, nous avons choisi d'effectuer 10 essais pour évaluer les performances de chaque hyperparamètre. Il est apparu que 16 heads et 30 hidden channels donnaient les meilleurs résultats.

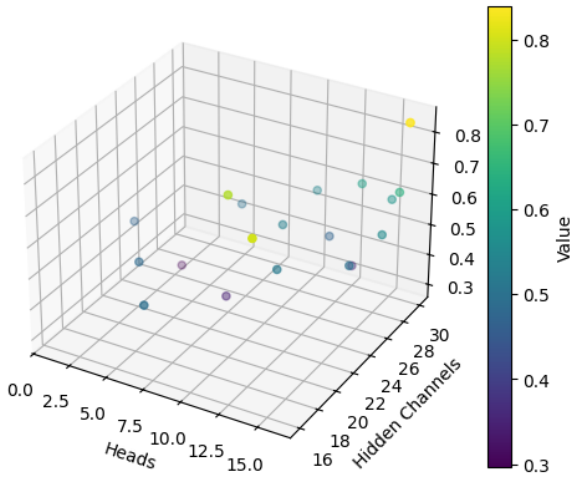

Figure 7: Précision en fonction du nombre de heads et de hidden channels pour la transduction

4.1.2 Induction

Contrairement à la transduction, l'approche inductive est moins couteuse en temps de calcul puisqu'elle fonctionne avec moins de liens entre les nœuds et avec plusieurs graphes de plus petites taille. Ainsi, nous avons choisi d'effectuer 25 essais, soit 15 de plus que pour la transduction, pour choisir des bons hyperparamètres. Il est apparu que 8 heads et 20 hidden channels donnaient les meilleurs résultats.

4.2 Résultats des modèles

Maintenant que nous disposons du bon couple d'hyper paramètres pour l'induction et la transduction, il nous est possible de présenter les résultats des modèles. Dans cette section, nous allons présenter 3 modèles :

- un modèle étalon constitué d'un graphe d'entraînement et un graphe de test
- un modèle utilisant la transduction
- un modèle utilisant l'induction

4.2.1 One graph only model

Le premier modèle présenté est un modèle qui fonctionne comme la plupart des algorithmes d'apprentissage profond : il apprend sur des données d'entraînement avant d'être

testé sur un set différent. Dans un sens, il combine la transduction et l'induction. Même s'il s'entraîne sur un seul et même grand graphe, il est testé par la methode inductive (soit donc sur un autre graphe). Ce modèle peut nous servir de base pour estimer les performances des approches inductives et transductives.

Ce modèle est initialisé avec 20 hidden channels et 8 heads, où l'on applique une cross validation avec 5 folds où chaque fold voit le modèle s'entraîner sur 1000 époques. Ci-contre les performances de ce modèle sur les données de validation :

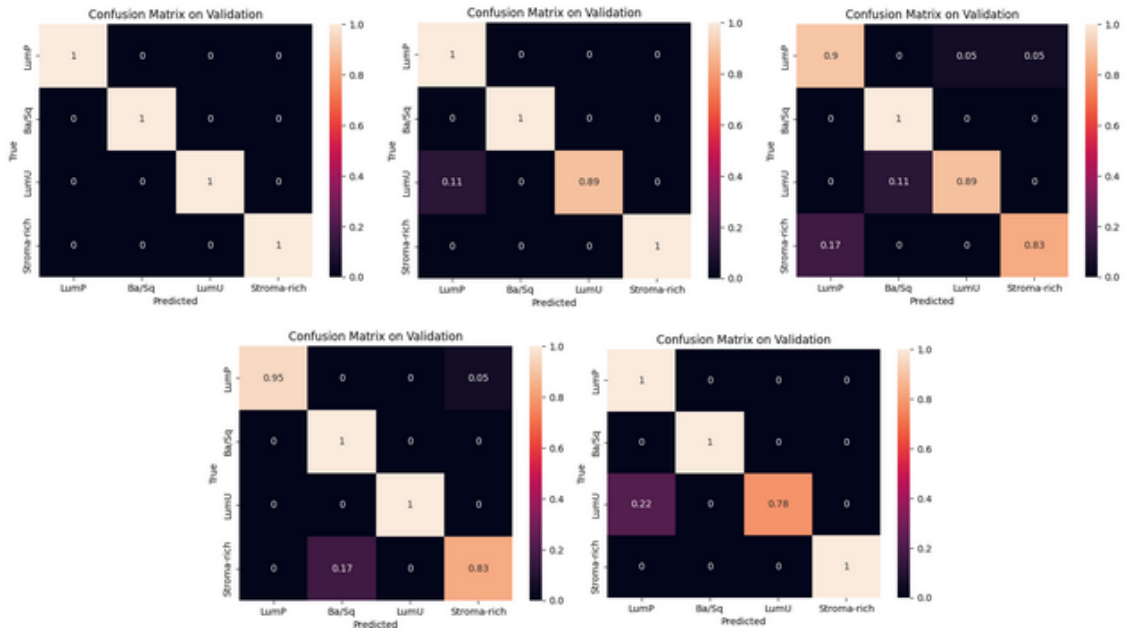

Figure 8: Matrice de confusion sur le set de validation du modèle one graph

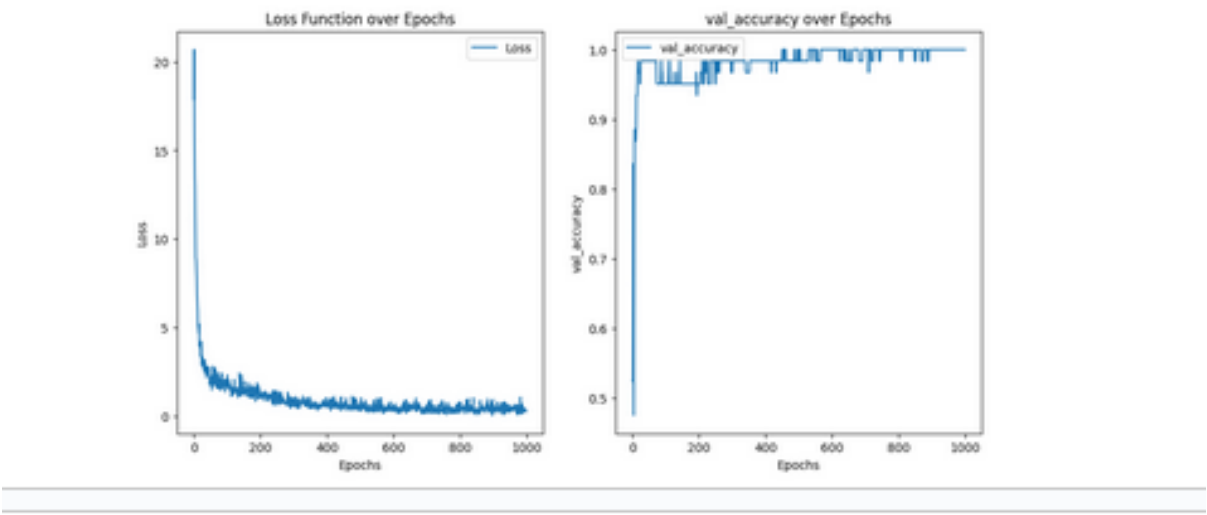

Figure 9: Evolution de la fonction de coût et de la précision du modèle One Graph au fil des époques pour la fold 1

Avec 97 % de précision sur les données de validation, le modèle one graph se a de très bons résultats. Voici ses performances sur les données de test :

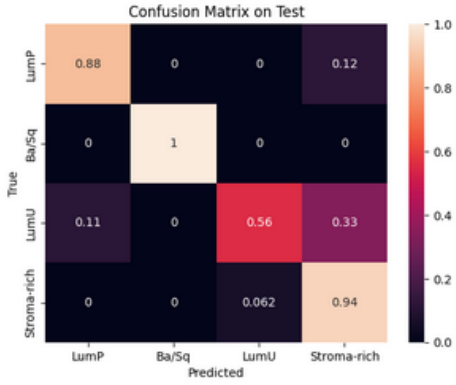

Figure 10: Matrice de confusion sur les données de test du modèle one graph

Sur la figure ci-dessus, nous pouvons observer que le modèle a de bonnes performances. Il obtient même 87% de précision sur les données de test.

4.2.2 Induction model

Le second modèle présenté est un modèle utilisant l'induction. Il se décompose en 5 sous graphes d'apprentissage et un graphe de test. Ce modèle est initialisé avec 20 hidden channels et 8 heads, où l'on applique une cross validation avec 5 folds où chaque fold voit le modèle s'entraîner sur 1000 époques. Ci-contre les performances de ce modèle sur les données de validation.

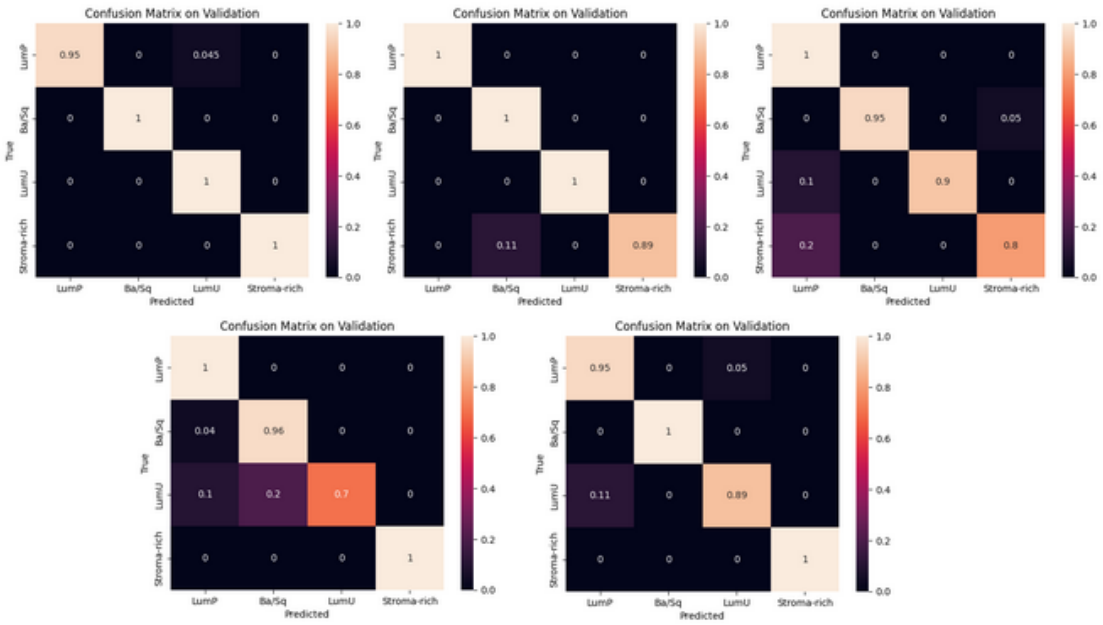

Figure 11: Matrices de confusion du modèle induction

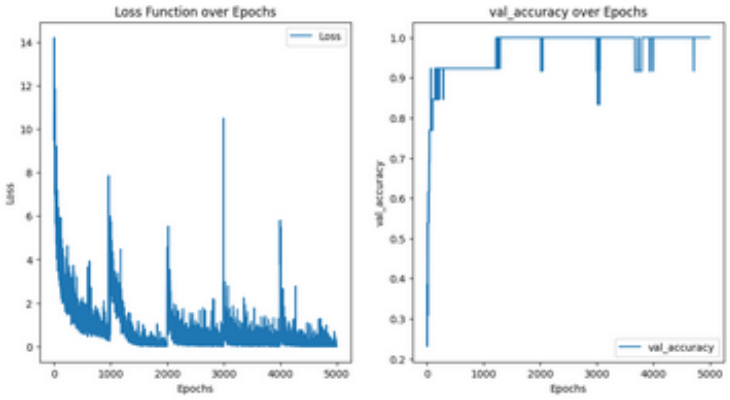

Figure 12: Evolution de la fonction de coût et de la précision du modèle Induction au fil des époques pour la fold 4

Le modèle Induction parvient à obtenir 100% de précision sur les données de validation. Voici ses performances sur les données de test :

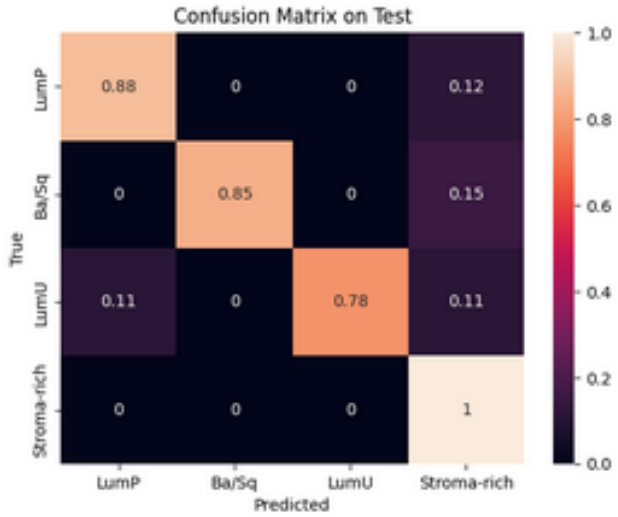

Figure 13: Matrice de confusion sur le set de test du modèle induction

Le modèle Induction performe très bien sur les données de test et de validation, il obtient 88% de précision sur les données de test tout en ayant un temps de calcul très faible.

4.2.3 Transduction model

Le dernier modèle présenté est un modèle utilisant la transduction. Contrairement aux deux précédents modèles, il est entraîné sur 1250 époques pour chaque fold.
Ce modèle est initialisé avec 24 hidden channels et 11 heads, où l'on applique une cross validation avec 5 folds. Il est important de noter que son temps de calcul est beaucoup plus élevé que les deux modèles précédents. Ci-contre les performances de ce modèle sur les données de validation :

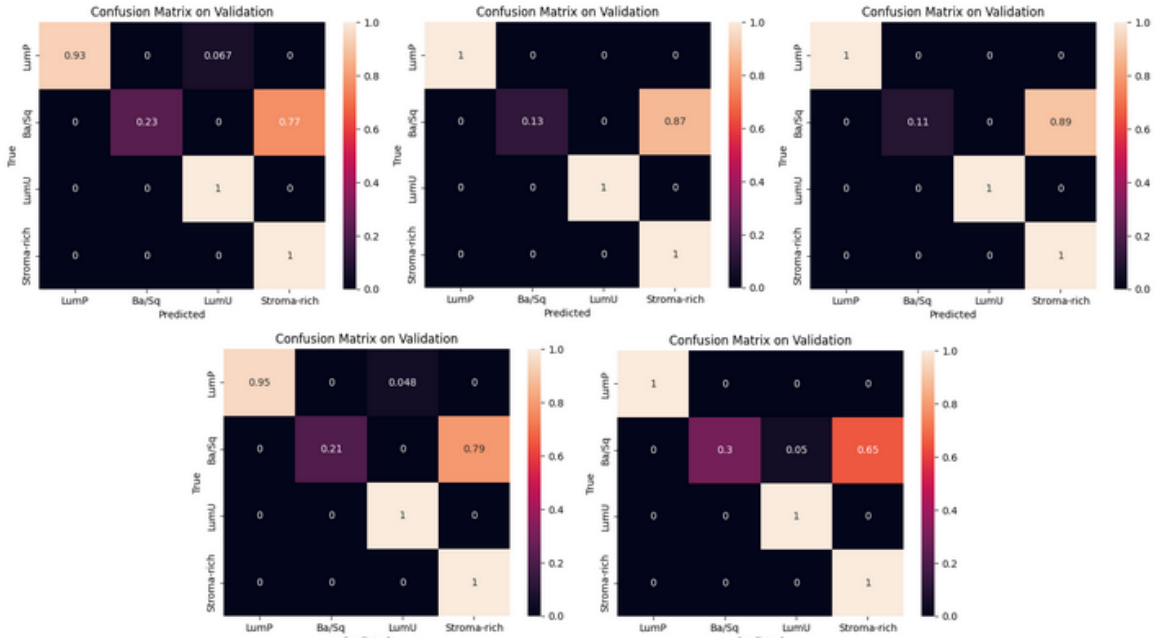

Figure 14: Matrice de confusion du set de validation du modèle transduction

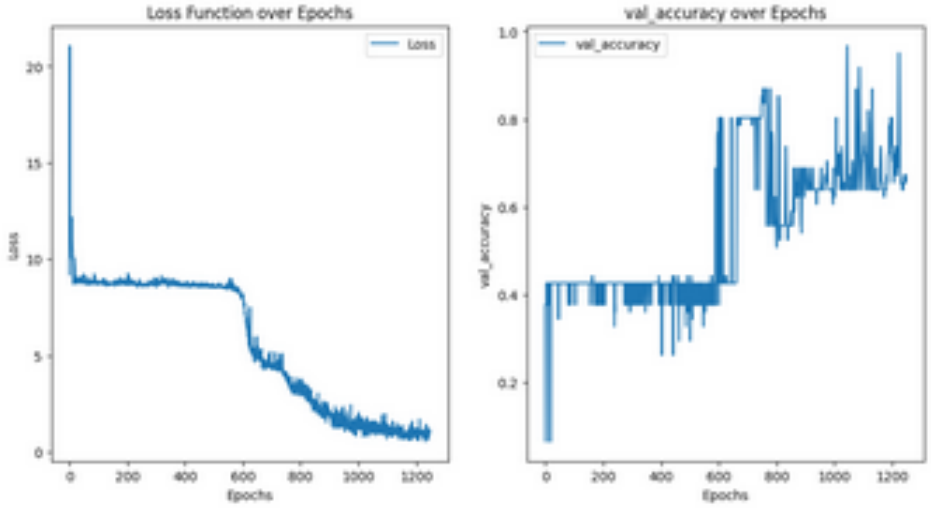

Figure 15: Evolution de la fonction de coût et de la précision du modèle Transduction au fil des époques pour la fold 2

Contrairement aux deux modèles précédents, le modèle transductif performe moins bien sur les données de validation (seulement 69%). Il a de moins bons résultats sur les données de test, mais de manière très légère (cf figure ci-dessous):

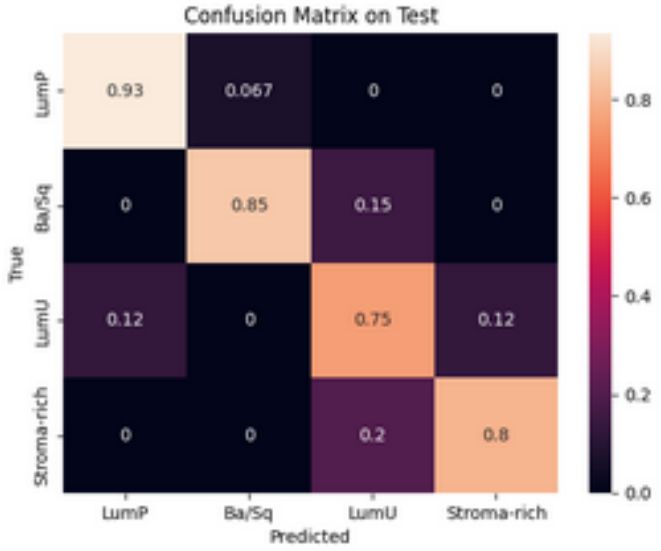

Figure 16: Matrice de transduction du set de test modèle transduction

Malgré une matrice de confusion qui semble indiquer des performances en deçà des autres modèles, le modèle Transduction obtient 87% de précision sur les données de test.

4.3 Analyse des résultats

Les résultats obtenus permettent de conclure que c'est le modèle par induction qui semble obtenir les meilleurs résultats. En effet, il possède un temps de calcul plus faible que les deux autres méthodes en ayant une précision plus élevée.

Le second modèle le plus efficace est le modèle One graph qui est deuxième en temps de calcul et ex aequo en précision. Le modèle le moins efficace est le modèle par transduction, car il a le temps de calcul le plus élevé. Cependant, nous pouvons nous interroger sur les raisons qui font que le modèle Transduction performe peu sur les données de validation, mais parvient à obtenir de bons résultats avec les données de test. Nous pouvons expliquer cela par le fait que le modèle transductif a déjà intégré les données de tests lors de son entraînement. En effet, même s'il n'apprend pas directement sur ces dernières, les liens qu'elles forment avec les données d'entraînement permettent au modèle d'avoir une bonne précision quand il s'agit de les classifier.

4.4 Limites et améliorations

Dans ce projet, nous avons pu explorer les deux approches principales pour la classification de données avec un graphe. Certains points ont été abordés avec plus ou moins de précision dans ce projet, et il est possible d'imaginer des améliorations pour parvenir à un modèle plus complet.

Durant tout le projet, notre approche a consisté à lier les patients entre eux grâce à leur similarité, lorsque cette dernière dépassait un certain seuil. Il aurait été possible de jouer avec ce seuil afin de former des graphes plus ou moins complexes (plus le seuil est élevé, moins il y a de liens, donc moins le modèle est complexe) et d'étudier l'impact de

ce changement sur le temps de calcul ou la précision.
De la même manière, certains paramètres n'ont pas pu être étudiés dans ce projet, notamment, comme dit plus haut, l'optimisation de certains hyperparamètres. Il aurait été par exemple possible de changer le nombre de sous graphes composants la méthode inductive. De même, optimiser le nombre de couches, le learning rate et le weight decay pourrait être intéressant pour obtenir le modèle le plus parfait possible.
Enfin, utiliser une cross validation avec plus de fold aurait également permis d'améliorer la robustesse des modèles présentés, même si le nombre de données mise à notre disposition rendait difficile leurs transformations.

5 Conclusion

Ce projet a permis de mettre en lumière de nouvelles techniques pour identifier des cancers de la vessie chez des patients. Nous avons pu utiliser les données cliniques, omiques et pathologiques pour créer plusieurs réseaux de neurones en graphe, et plus précisément des graphes d'attention (GATv2). Deux approches ont été privilégiées : une approche inductive et une approche transductive. Ces approches ont présenté de bons résultats, tant sur les données d'entraînement que sur les données de test, ce qui porte à croire que les techniques utilisées dans ce projet pourront être utiles pour des diagnostiques dans le futur.

6 Bibliographie

[1] Gatv2 : https://pytorch-geometric.readthedocs.io/en/latest/generated/
torch_geometric.nn.conv.GATv2Conv.html
[2] A Consensus Molecular Classification of Muscle-invasive Bladder Cancer :
https://www.sciencedirect.com/science/article/pii/S0302283819306955
[3] Comprendre les graphes d'attention : https://www.youtube.com/watch?v=A-yKQamf2Fct=5s