PRTSI1 : Rapport d'avancement du projet CLIPS[1]

1 Problématique et objectifs

Ce projet est un partenariat entre l'École Centrale de Nantes, le LS2N et le CHU de Nantes. Il intervient au cours de la thèse d'Oriane THIERY, une doctorante du LS2N, et vise à améliorer les performances d'un réseau de neurones convolutif (ou CNN en anglais) performant un type particulier de segmentation. En effet, dans le cadre du diagnostic d'un patient atteint par un cancer du type lymphome B-diffus à grandes cellules (ou DLBCL), il a été montré que le calcul du volume métabolique tumoral total à l'échelle du corps entier (TMTV : Total Metabolic Tumoral Volume) était une bonne évaluation de l'état de santé du patient. Cette donnée permet de classer les patients à risque faible et élevé dès la prise en charge par les médecins (c'est-à-dire avant la mise en place d'un traitement). Cependant, cette donnée est difficilement accessible, car les patients atteints de DLBCL possèdent de trop nombreuses lésions différentes pour que le calcul du volume total puisse être fait manuellement dans un délai raisonnable. C'est pour cette raison que des CNN ont été développés pour accomplir cette tâche de manière automatique à partir d'images TEP (Tomographie par Émission de Positons) et CT (Computed Tomography) des patients. L'un d'entre eux est le TMTV-Net : il a été développé par l'Université de Vancouver et réalise des segmentations d'assez bonne qualité, mais il présente à l'heure actuelle quelques faiblesses. Les images de certains patients passent en effet au travers du réseau sans que les lésions n'aient été segmentées ou bien de manière très éloignée de la réalité. Notre but est donc de résoudre ce problème en réalisant de nouveau l'étape d'apprentissage de ce réseau, mais cette fois-ci en appliquant une méthode de Curriculum Learning.

Cette méthode a pour but de guider l'entrainement d'un réseau de neurones en choisissant l'ordre dans lequel on va montrer les données d'entrainement au réseau. Dans la plupart des cas, on va privilégier les exemples "simples" en premiers avant de monter graduellement en difficulté jusqu'à montrer les exemples les plus "difficiles" au réseau. Ce raisonnement se base sur la manière d'apprendre des étudiants dans la réalité et il a été démontré que cette méthode d'apprentissage pouvait en effet améliorer la vitesse et les performances d'un CNN réalisant une segmentation d'images TEP [1]. Le problème de cette approche consiste à définir la signification de ce que l'on appelle "difficulté" d'un exemple et la manière exacte de "montrer" des exemples au réseau étudiant. C'est donc à nous de trouver des solutions à ces problèmes et de les implémenter.

Comme mentionné précédemment, ce projet repose sur un travail mené par une équipe de recherche basée à Vancouver, dont nous avons pu récupérer les codes, disponibles sur GitHub[2], mais également sur le travail d'un ancien stagiaire du CHU. Ce dernier a déjà exploité le réseau pré-entrainé par l'équipe de Vancouver pour identifier les patients problématiques ainsi que calculé des métriques relatives à la qualité des segmentations réalisées. Nous pourrons donc utiliser ces connaissances comme base pour notre projet et ainsi aider le réseau à mieux performer.

2 Travail réalisé

2.1 TMTV-Net

La première étape de ce projet fut de se familiariser avec le fonctionnement du réseau TMTV-Net. Nous avons donc lu l'article original rédigé par l'équipe de Vancouver [2]. Grâce à cet artcilen, nous avons ainsi pu comprendre le fonctionnement grossier du réseau TMTVT-Net (voir également Figure 1):

1. Entrée : Le réseau prend en entrée deux images en trois dimensions pour chaque patient, un scan TEP et un scan CT.

[1] Curriculum Learning for Improved PET Segmentation
[2] Lien vers le dépôt de l'équipe de Vancouver: `https://github.com/qurit-frizi/TMTV-Net/`

2. Pré-traitement : Les deux images d'entrée ne sont pas à la même résolution (un voxel d'une image TEP représente un espace plus large qu'un voxel d'une image CT), il est donc nécessaire de les ré-échantillonner pour que l'on puisse comparer les deux images voxel à voxel. À partir ces "nouvelles" images, cinq types d'image différents ont été créés, chacun permettant au réseau de se concentrer sur certains types de tissus ou de lésions.

3. Ré-échantillonnage : Les cinq types d'image ont ensuite été ré-échantillonnés à différente taille de voxels ($2mm^3$, $4mm^3$, $6mm^2$, $8mm^3$ et une taille aléatoire entre 2 et 10 mm^3). Cela crée ainsi cinq versions pour chacun des cinq types d'images précédentes, chaque version étant traitée par la suite par un réseau différent.

4. Augmentation des données : Le nombre de patients étant relativement limité dans la base de données, les images précédentes sont utilisées pour la création de "nouvelles" données, qui sont en réalité des versions déformées, transformées avec des opérations simples des images obtenues à l'étape précédente.

5. Première segmentation : Les images augmentées sont alors utilisées pour entrainer cinq 3D U-Nets différents, un par résolution. Cette différence de résolution facilite ainsi l'analyse grossière des motifs globaux et des dépendances étendues entre les lésions. Chaque 3D U-Net est entrainé avec une fonction de coût semi-supervisée (non détaillée ici) et réalise une prédiction pour chaque voxel en leur associant une valeur comprise entre 0 et 1 correspondant à la probabilité que ce voxel corresponde à du tissu cancéreux. C'est cela que l'on appelle une segmentation.

6. Agrégation des segmentations : À partir de ces cinq segmentations (que l'on ré-échantillonne à une taille de voxel moyenne), on applique un algorithme de vote. Ce vote prend en compte, pour chaque voxel, la probabilité que chaque modèle lui a associée et prend une décision en fonction de ces "votes". Ce processus permet d'obtenir alors une unique segmentation, reflétant les cinq précédentes.

7. Deuxième segmentation : La segmentation résultante de processus est ensuite concaténée avec les données originales (échantillonées à la bonne taille de voxel) afin de servir comme donnée d'entrée à un dernier 3D U-Net qui réalise la segmentation finale du patient.

Ce modèle est assez complexe, notamment en raison de sa structure en cascade, mais également en raison du peu de détail que nous fournit l'article original. Encore aujourd'hui, certaines de nos questions sur la manière exacte dont se déroule le processus n'ont pas encore trouvé de réponses détaillées, seulement des suppositions. Néanmoins, la lecture de cet article nous a permis d'avoir accès à une compréhension globale du fonctionnement du réseau. Nous avons ainsi pu entreprendre un travail de recherche afin de décider de la méthode de Curriculum Learning la mieux adaptée à notre problème.

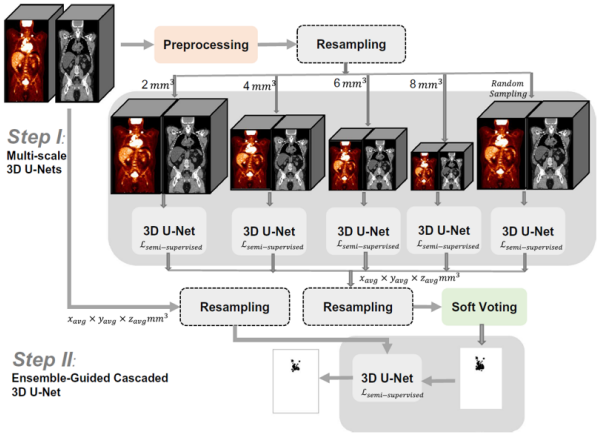

Figure 1: Schéma du fonctionnement de TMTV-Net

2.2 Curriculum Learning

Il existe de nombreuses méthodes d'utilisation du Curriculum Learning (CL), chacune ayant ses forces et ses faiblesses. Notre but était de déterminer celle qui serait la plus adaptée à notre cas. Nous avons donc poursuivi notre lecture bibliographique [3] [4] afin d'avoir un éventail de possibilités parmi lesquelles choisir la solution que nous implémenterons.

- CL par lissage : Cette méthode consiste à introduire un filtre gaussien dans les couches de convolution. Ce filtre agit comme un lissage sur les données en atténuant les détails et le bruit. On diminue ensuite progressivement la variance du filtre afin que le réseau apprenne au fur et à mesure à gérer les détails. Nous n'avons pas retenu cette solution, car elle ne nous permet pas d'utiliser l'information a priori que nous possédons déjà sur les patients (ie. les patients dont les segmentations ont été mal réalisées) mais également en raison de la difficulté que nous avons à exploiter le code jusqu'à présent. Ce dernier est en effet aussi complexe que la structure du réseau qu'il implémente, le manque de temps sur ce projet ne nous oriente donc pas dans cette direction.

- CL prédéfini : Cette méthode ne se base que sur de l'information a priori, c'est-à-dire que c'est à l'utilisateur de définir lui-même une fonction évaluant la difficulté de chaque image et de choisir à partir de quelle epoch le réseau peut commencer à apprendre sur des données jugées plus "difficiles". Nous n'avons pas non plus retenu cette méthode car elle ne s'adapte pas au réseau lui-même et s'appuie uniquement sur la réflexion de l'utilisateur, ce qui nous a semblé limité par rapport aux possibilités apportées par les solutions suivantes.

- Self-Paced Learning : Cette méthode consiste à associer un poids à chaque exemple puis à modifier ce poids en fonction d'un paramètre d'âge λ : plus λ est grand, plus le poids des exemples dits "difficiles" (ie. une valeur élevée pour cet exemple dans la fonction de perte) sera élevé, jusqu'à atteindre ceux des exemples dits "faciles" (faible valeur de perte). Le paramètre λ évolue alors à chaque époque en suivant une suite arithmétique ou géométrique (dans la version la plus simple de l'implémentation). Cette méthode peut être améliorée en incluant des informations a priori sur les valeurs aberrantes, sur la régularité temporelle/spatiale des données, sur l'importance de certains exemples ou sur la diversité des données. Bien que le Self-Paced Learning soit une méthode semi-supervisée (qui s'adapte en partie au réseau), nous n'avons pas retenu cette méthode car pour avoir une méthode permeformante, il est nécessaire d'optimiser des hyperparamètres, une étape que nous avons jugé trop chronophage pour un premier essai d'utilisation de CL. De plus, si le réseau choisit lui-même l'ordre des exemples à apprendre alors qu'il n'a encore rien appris, il est possible qu'il puisse se tromper et ainsi commencer son apprentissage sur des exemples que l'on sait en réalité "difficiles".

- Transfer Teacher : Dans cette approche, on ne demande plus au réseau de savoir lui-même comment classer les exemples "difficiles" des exemples "faciles" mais on confie cette tâche à un réseau expérimenté (identique ou différent du réseau étudiant), ie. déjà entrainé sur les mêmes données. Ce réseau professeur donne alors de l'information sur chaque exemple à apprendre, que ce soit grâce à la valeur de perte de ce dernier ou bien, dans notre cas, grâce à la qualité de la segmentation qu'il a réalisée. Cette approche nous a semblé pertinente pour notre problème puisque nous avons déjà accès à un réseau pré-entraîné ainsi qu'à des métriques de la qualité des segmentations réalisées par ce dernier. De plus, cette méthode est très simple à implémenter car elle ne suppose pas que nous devions modifier la structure de TMTV-Net comme dans le cas du CL par lissage. Cela nous permet ainsi de rapidement programmer cette solution et savoir si elle fonctionne ou non et ensuite revoir notre stratégie en fonctions des résultats. Néanmoins, cette méthode ne permet pas d'implémenter directement de l'information à priori comme pour le Self-Paced Learning. On pourra tout de même tenter, après un premier essai sans information à priori, de combiner le Transfer Teacher avec des méthodes de Self-Paced Learning, en exploitant la fonction de perte du réseau professeur plutôt que celle du réseau étudiant.

- RL Teacher : Cette dernière méthode exploite l'idée d'un professeur et d'un étudiant se faisant mutuellement des retours sur l'évolution de l'apprentissage afin d'améliorer la vitesse et la qualité de ce dernier. Ce principe repose notamment sur des méthodes d'apprentissage par renforcement pour le professeur, qui n'est donc pas lui-même un réseau comme c'était le cas dans la méthode du Transfer Teacher. Le

professeur est par conséquent un mesureur de la difficulté des exemples ainsi qu'un planificateur d'étude (ie. choix de l'ordre dans lequel montrer les exemples) pour le réseau étudiant. Cette méthode est de loin la plus complexe, et nous ne maitrisons malheureusement que très peu les méthodes d'apprentissage par renforcement. Nous avons donc choisi d'écarter cette possibilité de notre projet afin de ne pas perdre de temps sur une méthode qui demanderait beaucoup de temps de programmation en plus de beaucoup de temps de calcul d'apprentissage puisqu'il faudrait entrainer le professeur et l'étudiant en même temps.

3 Étapes suivantes

Grâce à notre étude bibliographique, nous avons déterminé la méthode de Curriculum Learning qui nous semblait la plus adaptée à notre problème. Il nous faut donc désormais l'implémenter dans le code de l'équipe de recherche de Vancouver pour entamer un nouvel entrainement du réseau et comparer ses performances à celles obtenues jusqu'à présent.

Pour ce faire, nous avons besoin de nous rendre régulièrement sur le site du CHU d'Hôtel Dieu pour développer et faire tourner notre futur code. En effet, la taille des images TEP et CT est très conséquente, il nous est ainsi impossible de réaliser un entrainement complet sur nos propres machines personnelles qui ne disposent pas de suffisamment d'espace de stockage ni de mémoire vive pour faire fonctionner l'algorithme à une vitesse raisonnable. Nous nous sommes déjà rendus une première fois sur place pour mettre en place un environnement de travail (configuration d'un ordinateur Linux à notre disposition) mais nous devons désormais nous y rendre plus fréquemment pour avancer réellement sur ce projet.

Bien que le site du CHU soit le meilleur endroit pour travailler sur ce projet, nous ne pouvons nous y rendre tous les jours. Nous avons donc créé un dépôt GitHub[3] sur lequel nous avons pu déposer le code provenant de l'Université de Vancouver, et ainsi travailler sur le code depuis chez nous sans avoir besoin de nous rendre au CHU. Nous avons également pu récupérer les données de trois patients dont un a été mal segmenté par le réseau original. Cela nous permettra de tester notre implémentation du Curriculum Learning sur un jeu de données réduit avant de lancer l'entrainement sur l'ensemble de la base de données que nous utiliserons, nommée GAINED.

L'étape de l'entraînement risque cependant de poser problème : notre tuteur, M. Thomas Carlier, nous a informés que l'équipe de Vancouver était pour le moment légèrement réticente à nous aider dans notre démarche de réapprentissage des poids de TMTV-Net. Sans leur aide, nous risquons donc de prendre beaucoup plus de temps pour comprendre le code. Malgré cette difficulté, nous commencerons par le lire dans sa globalité (une étape que nous avons déjà entamée) afin de tenter de comprendre par nous-mêmes son fonctionnement. Nous espérons ensuite recevoir des nouvelles de l'équipe de Vancouver pour qu'elle nous épaule et nous permette d'obtenir un code fonctionnel ainsi que des résultats dans le temps qui nous est imparti. Si les chercheurs de Vancouver ne nous contactent pas, nous ferons néanmoins tout notre possible pour exploiter au mieux ce que nous aurons pu comprendre du code et obtenir les résultats souhaités.

References

[1] F. Yousefirizi et al, *Curriculum learning for improved tumor segmentation in PET imaging*, Proc IEEE Nuclear Science Symp Medical Imaging Conf, 2022

[2] F. Yousefirizi et al, *TMTV-Net: fully automated total metabolic tumor volume segmentation in lymphoma PET/CT images — a multi-center generalizability analysis*, Eur J Nucl Med Mol Imaging 2024

[3] S. Sinha et al, *Curriculum by smoothing*, Adv Neural Inf Process Syst, 2020

[4] Xin Wan et al, *A Survey on Curriculum Learning*, IEEE Transactions on pattern analysis and machine intelligence, Vol. 44, No. 9, 2022

[3] Lien vers notre dépôt pour le projet : `https://github.com/S3vy/CLIPS/`