Rapport sur mon job d'étudiant: Utilisation du deep reinforcement learning dans le cadre de la radiothérapie

Moreau Grégoire

Juin-Juillet 2019

1 Introduction

L'objet du travail que j'ai accompli lors de ce job étudiant était d'utiliser le reinforcement learning afin de trouver un meilleur planning de traitement (treatment schedule) pour la radiothérapie.
Quest ce que le cancer Radiothérapie Le reinforcement learning

Actuellement, pour planifier un traitement de radiothérapie, on détermine la dose totale en Gray à délivrer à la tumeur afin d'obtenir un haut pourcentage de Tumor Control Probability (probabilité que toutes les cellules cancéreuses composant la tumeur soient mortes après le traitement). La TCP augmente avec la dose totale délivrée, mais cette dernière ne peut pas être trop haute afin de limiter la normal tissue complication probability (NTCP). Une fois que la dose totale est déterminée, le traitement est généralement délivré sous forme de fractions journalières de 2 Gray. Par exemple, 35 doses de 2 Gray pour 80% TCP. Certain médecins pensent qu'ils pourrait y avoir des avantages à varier la dose des fractions individuelles, et/ou de varier le temps entre chaque fraction. Je vais donc implémenter un agent utilisant un algorithme de reinforcement learning et interragissant avec une simulation de prolifération tumorale. Après avoir été entrainé, l'agent devrait être capable de trouver un nouveau planning de traitement.

Le travail que j'ai accompli au cours des dernières semaines est en fait la première partie du travail de fin d'études que je vais accomplir l'année scolaire prochaine dans le cadre de ma dernière année de master d'ingénieur informaticien à l'Ecole Polytechnique de Louvain.
Dans ce rapport, je vais d'abord détailler les différentes étapes du travail que j'ai accompli et motiver les décisions que j'ai prises au cours de ces étapes. Ensuite, je vais présenterai les résultats que j'ai obtenus. Je terminerai par les pistes de futures améliorations du modèle actuel et les autres approches que je pourrai utiliser au cours de mon travail de fin d'étude.

2 Approche

L'approche utilisée dans ce travail est assez largement basée sur l'article "Agent-based modeling and reinforcement learning"[1]. Dans cet article, Jalalimanesh, Shahabi, Ahmadi et Soltani utilisent NetLogo, un programme permettant de simuler les actions concurrentes d'un grand nombres d'agents pour modéliser la prolifération d'une tumeur à partir d'une cellule cancéreuse dans du tissu sain. Cette simulation était basée sur le travail de Nicole O'Neil, "An agent based model of tumor growth and response to radiotherapy"[2].

O'Neil a inclu le code de la simulation dans son travail. Jalalimanesh et al. utilisent l'oxygène comme nutriment alors que O'Neil utilise le glucose. Dans O'Neil, la radiation est enclenchée directement par l'utilisateur et il peut l'envoyer où il le souhaite en cliquant sur la zone à irradier. Jalalimanesh et al. utilisent quant à eux un agent implémenté en R pour contrôler les doses de radiation envoyées à la simulation NetLogo.

Jalalimanesh et al. utilisent des simples observations basées sur le nombre de cellules cancéreuses restantes afin de limiter la dimensionnalité du problème étudié et d'ainsi obtenir un faible nombre d'états possibles dans son modèle. Cela leur permet d'utiliser le simple Q-learning basé sur une table. Comme je souhaitais utiliser des features plus complexes, potentiellement continues et que j'aimerais pouvoir passer à l'agent une image de la tumeur, je dois utiliser des algorithmes de deep learning comme les deep-Q-networks et les policy gradients. J'ai utilisé les implémentations de ces algorithmes du package deer créé par Vincent François-Lavet [3].

Il existe un package permettant de faire communiquer Python et NetLogo, PyNetLogo mais l'échange ne se fait pas de manière très rapide et n'était pas possible avec les dernières versions de tous les programmes impliqués donc j'ai préféré ne pas utiliser cette technique. J'ai implémenté une simulation de prolifération tumorale en Python très similaire à la simulation de O'Neil en NetLogo.

3 Simulation de prolifération des cellules tumorales

La simulation est basée sur une grille à deux dimensions sur laquelle est placée de manière aléatoire un certain nombre de cellules saines et de vaisseaux sanguins. Une cellule cancéreuse est ensuite placée au milieu de la grille. Au cours du temps, les cellules vont consommer des nutriments amenés par les vaisseaux sanguins et se diviser. La cellule cancéreuse va donc créer une masse tumorale composée d'un nombres toujours plus grand de cellules. Si on n'envoie aucune radiation, cette tumeur va petit à petit envahir l'entièreté de la grille en privant les cellules saines de nutriments et ne leur laissant pas la place de se propager.

En pratique, la simulation a une méthode go() qui permet de simuler une heure. Durant cette heure, les sources sont réapprovisionées en nutriments, les nutriments sont diffusés sur la grille, et les cellules avancent d'une heure dans leur cycle cellulaire.

3.1 Types de cellules

La simulation peut contenir trois différents types de cellules : des cellules saines, des cellules cancéreuses et des cellules d'organes à risque.

3.1.1 Cellules saines

Les cellules saines effectuent le cycle cellulaire en consommant des nutriments et ont la capacité de se diviser environ toutes les 24h. Elles peuvent seulement se diviser si leur environnement n'est pas trop dense et si il y a assez de nutriments sur le pixel où se trouve la cellule mère pour supporter

deux cellules. La nouvelle cellule sera envoyé sur le pixel adjacent de plus faible densité de telle manière que la densité du tissu sain reste relativement uniforme. Si une des conditions qui permettent la mitose n'est pas remplie, la cellule entre en quiescence et ne parcourt plus le cycle cellulaire. Elle consomme alors moins de nutriments. Elle pourra sortir de cet état si une des conditions qui empêchait la mitose est changée.

3.1.2 Cellules cancéreuses

Les cellules cancéreuses effectuent aussi le cycle cellulaire en consommant légèrement plus de nutriments. Elles ne peuvent pas entrer dans un état de quiescence et continuent donc à se diviser indéfiniment. La cellule fille est envoyée à un pixel adjacent de manière aléatoire. De cette manière, la tumeur se propage circulairement. Ces deux modifications font en sorte que la tumeur forme des zones très denses en cellules cancéreuses et qui sont assez rapidement vidées de leur nutriments ce qui cause donc la mort des cellules qu'elles contiennent.

3.1.3 Cellules d'organes à risque

Ces cellules représentent un organe à risque qui pourrait être proche de la tumeur et dont il faut tenir compte lors de la radiothérapie. Elles n'effectuent pas le cycle cellulaire et ne pourront donc pas être remplacées si elles sont tuées par la tumeur ou la radiation. Dans la simulation, elles ne consomment pas de nutriments.

3.2 Cycle cellulaire

3.3 Nutriments et réaction

Pour cette simulation, j'ai choisi de modéliser l'oxygène et le glucose comme nutriments disponibles pour les cellules. En effet, les deux sont indispensables au développement des cellules. La réaction de base de la respiration cellulaire est :

$$C_6H_{12}O_6 \ + \ 6\ O_2 \ \rightarrow \ 6\ CO_2 + ATP \ + \ chaleur$$

et elle nécessite ces deux nutriments. L'oxygène a également un rôle particulièrement important dans la radiothérapie puisque le niveau d'oxygènation

des cellules affecte leur radiosensitivité. Par exemple, les cellules hypoxiques sont plus résistantes à la radiation.

3.4 Angiogenèse et obstruction

L'angiogenèse est un phénomène où les cellules cancéreuses en manque d'oxygène envoie des signaux au système sanguin qui conduisent à la vascularisation de la tumeur et donc à l'oxygènation de ses cellules. Il m'a semblé important de modéliser ce phénomène puisque il affecte directement la concentration en nutriment des pixels de la tumeur et a une incidence directe sur leur survie. Ainsi, si on ne considère pas ce phénomène, la densité très forte au coeur de la tumeur va faire en sorte que ce dernier soit complètement mort après quelques jours.

Afin de garder un nombre de vaisseaux sanguins relativement constant, j'ai aussi intégrer un phénomène d'obstruction ayant un effet opposé à l'angiogenèse. Si une case comprend un vaisseau sanguin et une très forte densité de cellules, ce vaisseau sanguin meurt et est supprimé de la grille. Cela correspond à un phénomène réel où les cellules cancéreuses envahissent un vaisseau sanguin et le bouchent.

Les vaisseaux sanguins créés par angiogenèse sont placés sur la grille à des endroits où les nutriments commencent à manquer de manière aléatoire. En pratique, si après un tour, après que les cellules sur un pixel aient chacun prises une portion de nutriment, la quantité de nutriments sur le pixel est passée sous un seuil critique qui impliquera la mort d'une cellule au tour suivant, la probabilité de création de vaisseau sanguin sur ce pixel sera :

$$\frac{Nbre\ de\ vaisseaux\ sanguins\ initial\ -\ Nbre\ de\ vaisseaux\ sanguins\ actuel}{2\ *\ Nbre\ de\ vaisseaux\ sanguins\ initial}$$

Cette formule permet d'éviter que tous les vaisseaux sanguins soient créés dans la même partie de la grille.

4 Simulation de la radiation

La deuxième partie de la simulation est la radiation que l'on veut envoyer sur les cellules pour simuler la radiothérapie. L'irradiation de la tumeur

se fait entre deux appels à la méthode go() et est donc considérée comme instantanée. Le rayon de la tumeur est déterminé par la simulation pour qu'elle soit complètement irradiée.

4.1 Equation LQ

L'équation LQ est utilisée pour déterminer l'effet de la radiation sur les cellules irradiées. La formule de cette équation est :

$$S(D) \ = \ exp(-\alpha D \ - \ \beta D^2)$$

où S(D) est la fraction des cellules ayant survécu à la radiation, D est la dose en Gray délivrée aux cellules et α et β sont des facteurs décrivant la radiosensitivité des cellules irradiées.

J'utilise cette équation pour déterminer la probabilité de survie de chaque cellule irradiée. Les paramètres α et β pour la tumeur ont été réglés de telle manière qu'il y ait environ 80% de chance que la tumeur soit totalement éliminées après 35 doses journalières de 2 Gray.

4.2 Isodose

Dans la simulation, la dose n'est pas déposée uniformément sur la dose pour représenter le fait que les cellules avoisinant la tumeur reçoivent aussi une partie de la radiation lors d'une séance de radiothérapie. La fonction qui décrit le dépot de la radiation sur la tumeur est représentée à la figure **??** et est le produit de convolution d'une fenêtre rectangulaire et d'une gaussienne. Les paramètes sont sélectionnés de manière à ce que les cellules au bord de la tumeur reçoivent environ 95 % de la dose sélectionnée.

5 Deep Reinforcement Learning (DeeR)

Comme spécifié dans les deux premières parties, j'utilise le framework du reinforcement learning pour trouver un bon "treatment schedule" dans le cadre de la radiothérapie. Dans ce framework, un agent choisit à chaque temps t une action en fonction de l'observation de l'environnement autour de lui. Après l'action, il reçoit une récompense ou "reward" sous forme d'un scalaire qui indique la qualité de l'action prise. Le but de l'agent est

de maximiser la somme des récompenses reçues au cours du temps. Pour utiliser le reinforcement learning avec ma simulation, j'ai donc dû définir les différentes parties composant le framework en fonction de l'état de la simulation.

5.1 Actions

Pour trouver un bon treatment schedule, il faut que l'agent détermine la dose à envoyer à chaque timestep et le temps à laisser passer entre chaque dose. Dans un premier temps, j'ai seulement considéré que l'agent choisissait les doses à envoyer en laissant passer 24h entre chaque dose. Les doses envoyée sont choisies entre 1 et 5 gray, de manière continue pour un algorithme et discrète pour un autre. Dans le cas discret, les doses possibles sont de 1, 1.5, 2 2.5, 3, 3.5, 4, 4.5 et 5 gray.

5.2 Observations

Les observations doivent permettre à l'agent de se faire une idée de l'état de la simulation. Elle est composée du nombre de cellules saines et cancéreuses restant ainsi que d'une image en deux dimensions de 20 pixels sur 20 représentant les types de cellules sur la grille.

5.3 Rewards

L'objectif de la radiothérapie est de détruire la tumeur en ménageant les tissus sains aux alentours. La fonction de reward donnée après qu'une dose ait été envoyée devrait donc prendre en compte le nombre de cellules cancéreuses tuées ainsi que le nombre de cellules saines tuées. On peut donner une importance différentes aux deux types de cellules en leur donnant un facteur multiplicatif différent.

5.4 Conditions d'arrêt

La radiothérapie peut être considérée comme terminée dans trois conditions :

- La tumeur est éliminée complètement, il n'y a plus de cellules cancéreuses.
- La tumeur s'est trop étendue, il n'y a plus de cellules saines

- Le traitement est trop long. En pratique, cette condition est activée lorsque le nombre d'heures de traitement dépasse 2000. Cette condition n'indique pas que la tumeur a trop grandi et elle pourrait être relativement maitrisé mais un traitement trop long implique de trop nombreuses séances de radiation qui causent des effets secondaires que la simulation ne prend pas en compte.

5.5 Algorithmes

Comme spécifié dans la section "Approche", j'ai utilisé le package deer [3] pour l'implémentation des algorithmes de deep reinforcement learning.

5.5.1 Deep Q-Learning

Cet algorithme est value-based. Cela signifie qu'avant de choisir une action, il estime la "valeur" de chaque action possible à ce moment là. La valeur Q d'une action a est définie récursivement comme la reward reçue en prenant cette action additionée à la valeur de la meilleure action qu'on peut prendre dans l'état atteint en prenant l'action a. Avec une certaine probabilité, il "exploite" alors l'action correspondant à la meilleure valeur, sinon il "explore" aléatoirement une autre action possible. Ce principe de balance exploitation-exploration permet à l'agent de continuer à explorer les possibilités d'action de son environnement en trouvant des stratégies qui lui donnent des sommes de rewards toujours plus grandes.

En deep Q-learning, le neural network reçoit les observations et calcule une valeur pour chaque action possible. Ensuite, après que la valeur réelle de l'action a été trouvée, on détermine l'erreur comme la différence entre la valeur calculée par le neural network et la valeur réelle observée avec la somme des rewards. L'objectif de l'entrainement du neural network sera donc de minimser cette erreur.

Cet algorithme considère des actions dans un domaine discret.

5.5.2 Actor-critic learning with Deep Deterministic Policy Gradient

Cet algorithme est policy-based. Cela signifie qu'au lieu de calculer les valeurs des actions, le neural network estime directement la stratégie de l'agent en calculant la probabilité de prendre chaque action dans un certain état. Dans ce cas, l'objectif de l'entrainement du neural network est de maximiser la probabilité de prendre des actions avec de hautes récompenses dans chaque état. Cet algorithme peut considérer des domaines d'action continus.

6 Entrainement des neural networks et résultats

6.1 Deep Q-learning

6.1.1 Hyperparamètres

Pour faire converger le neural network, il a fallu jouer avec les hyperparamètres, et notamment le learning rate utiliser lors de la backpropagation de l'erreur. J'ai finalement utilisé une valeur de $5 * 10^{-4}$ comme learning rate initial. Cette valeur était divisée par deux à la fin de chaque epoch d'apprentissage.

J'ai également remarqué que des batch sizes plus petites rendait l'apprentissage plus stable et l'entrainement a donc était fait avec une batch size de 0.8.

Enfin, j'ai utilisé une policy epsilon-greedy qui commençait à un taux d'exploration de 0.4 pour tomber à 0 à la fin de l'apprentissage. Ce taux initial de 40 % est relativement bas mais j'ai remarqué que sans cela le neural network surestimait la valeur de prendre l'action la plus faible dans chaque cas.

6.1.2 Entrainement

Le graphe **??** représente l'évolution de l'erreur pendant 5 epochs d'entrainement de 1000 steps chacune.

6.1.3 Rewards

6.1.4 Résultats

Après avoir été entrainé pendant 5 epochs de 1000 steps, l'agent choisit d'envoyer des doses relativement grandes. Il choisit toujours de commencer par envoyer 4.5 Gray, puis envoie 2 ou trois doses de 5 Gray. Après cela, il reste souvent une petite dizaine de cellules cancéreuses que l'agent élimine avec des doses de 2 Gray jusqu'à ce qu'il ne reste aucune cellule cancéreuse. La figure **??** représente l'évolution de la tumeur lorsqu'elle est traitée par l'agent.

6.2 Deep deterministic policy gradient

6.2.1 Hyperparamètres

Cet algorithme était moins stable. Le neural network a été entrainé avec un learning rate de $5 * 10^{-6}$. La batch size était également de 8. Je n'ai pas utilisé de stratégie d'exploration.

6.2.2 Entrainement

Le graphe **??** représente l'évolution de l'erreur pendant 5 epochs d'entrainement de 1000 steps chacune.

6.2.3 Rewards

6.2.4 Résultats

Après avoir été entrainé pendant 5 epochs de 1000 steps, l'agent ne semble toujours pas bien "comprendre" le problème auquel il fait face. En effet, il semble préférer d'envoyer des doses faibles entre 1.3 et 1.8 grays qui stabilisent relativement la tumeur, et de temps en temps envoyer une dose légèrement plus grosse de 3.5 qui permet d'éviter que la tumeur se propage trop, après quoi il recommence à envoyer de faible dose. il atteint à chaque essai la limite de temps. La figure **??** représente l'évolution de la tumeur lorsqu'elle est traitée par l'agent. Je crois que l'agent a besoin de plus d'entrainement pour mieux comprendre le problème, ou d'une meilleure fonction de reward.

7 Améliorations futures

Tout d'abord, il serait intéressant d'intégrer à la simulation le bystander effect. Cet effet consiste en la mort de cellule non-irradiées proches des cellules irradiées par l'envoi de signaux de ce derniers dans le milieu intercellulaire. L'intensité de cet effet ne serait pas fortement lié à la dose d'irradiation et il serait donc intégrer puisqu'il permettrait sans doute de faire comprendre à l'agent que le traitement ne peut pas être trop long.

Ensuite, les fonctions de rewards ont été ajustées pour améliorer la stabilité de l'entrainement des neural networks et ne sont donc pas forcément parfaites pour le problème concerné. Il me semble donc important de trouver une meilleure fonction de reward.

Il est aussi nécessaire d'intégrer l'effet de la concentration d'oxygène sur la radiosensitivité des cellules. Il faudra alors ajouter aux observations données à l'agent une image représentant les concentrations en oxygène des pixels de la grille.

Après ces ajustements, l'entrainement devra se faire sur un plus grand nombre d'epochs puisque les résultats obtenus ne sont pour le moment pas forcément optimaux sachant que les erreurs des neural networks peuvent manifestement encore diminuer en ajustant mieux les hyperparamètres.

Les prochaines simulations pourront également comprendre des zones d'organes à risque puisque il est important que l'agent comprenne qu'il faut absolument éviter d'endommager certains organes qui ne peuvent pas forcément se régénerer.

Il pourrait aussi être intéressant d'intégrer au modèle des images radiomics de tumeurs réelles permettant à la simulation d'atteindre des états plus réalistes et à l'agent de recevoir des features plus complexes ou plus représentatives de l'état de la tumeur.

Enfin, pour trouver un meilleur treatment schedule, il faudrait sans doute laisser à l'agent la possibilité de choisir le temps se déroulant entre deux irradiations de la tumeur. Il semble assez complexe d'utiliser un des algorithmes de deep reinforcement learning pour choisir des actions à deux dimensions,

surtout que ces deux actions ne seraient pas forcément toutes les deux continues ou discrètes. En effet, idéalement la dose serait choisie dans un domaine continu alors que le temps entre deux dose serait choisi dans un domaine discret. Toutefois, une idée utilisant deux agents en coopération utilisant le deep reinforcement learning et interragissant avec le même environnement est exposée à la dernière page de ce rapport. Cette idée vaut probablement la peine d'être explorée puisqu'elle offre une solution élégante au problème rencontré.

Références

[1] Ammar Jalalimanesh, Hamidreza Shahabi Haghighi, Abbas Ahmadi, and Madjid Soltani. Simulation-based optimization of radiotherapy: Agent-based modeling and reinforcement learning. *Mathematics and Computers in Simulation*, 06 2016.

[2] Nicole O'Neil. An agent based model of tumor growth and response to radiotherapy. Master's thesis, Virginia Commonwealth University, Richmond, Virginia, 5 2012.

[3] Vincent François-Lavet et al. Deer. `https://deer.readthedocs.io/`, 2016.