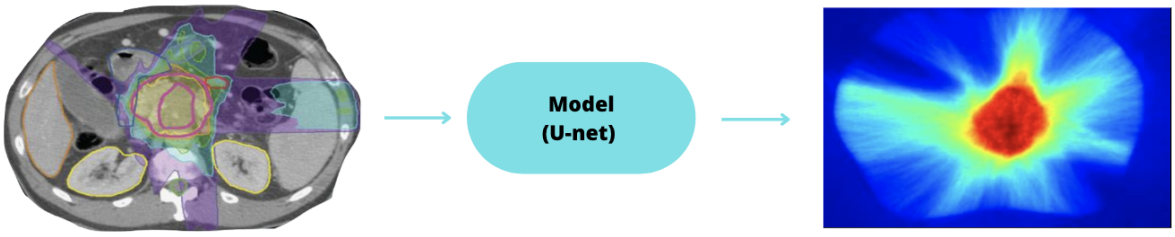
RADIATION DOSE PREDICTION

Leila Berrada & Antoine Gorceix

16 avril 2023



Input :
- **CT**
- **Segmentation masks of organs**
- **Maximum possible irradiation**

Output :
- **Radiation dose**
- **Size 128 x 128**

Contents

1 Introduction

La radiothérapie est un type de traitement largement utilisé pour les patients atteints de cancer. Elle consiste à délivrer une quantité spécifique de dose de rayonnement gamma sur la tumeur afin de la bruler tout en minimisant autant que possible l'exposition des tissus sains aux radiations. Cependant, plus la précision est de mise, plus la tâche est difficile et requiert des connaissances médicales poussées pour déterminer parfaitement quelle dose de radiation envoyer et où l'envoyer. Le processus de planification de la dose de radiation implique de nombreuses variables telles que la taille, la forme, la localisation et l'hétérogénéité de la tumeur, ainsi que les tissus environnants à risque. Cette complexité peut rendre le traitement imprécis et potentiellement dangereux pour les tissus sains environnants. Néanmoins, les avancées récentes en Deep Learning offrent la possibilité d'une segmentation précise des organes, ainsi que d'une approche personnalisée à la planification de la dose de radiation. Ainsi l'utilisation du Deep Learning pour prédire la dose de radiation offre une approche innovante et prometteuse pour améliorer les résultats des traitements de radiothérapie, ce qui est d'une importance capitale pour les patients atteints de cancer.

Problematique : Comment développer un réseau adéquat pour prédire la dose de radiation à administrer sur des scans en 2D et en 3D ?

Le code peut se trouver sur notre repository github : en cliquant ici

2 State of the art

La radiothérapie est couramment utilisé afin de traiter le cancer. Comme nous l'avons vu dans l'introduction elle nécessite d'une part une segmentation précise de la tumeur cible et des organes environnants pour éviter d'endommager les tissus sains ainsi qu'une connaissance précise pour savoir ensuite quelle dose de rayonnements envoyer. La pratique actuelle de la planification du traitement est en grande partie un processus manuel, qui prend beaucoup de temps et de travail. Par conséquent, la qualité d'un plan créé par différents planificateurs peut être incohérente et limitée par des considérations d'ordre pratique. Les méthodes basées sur l'apprentissage profond ont montré un grand potentiel dans la prédiction de la dose de radiation.

En 2016, Shiraishi et al. [4] ont mis au point une méthode de prédiction de la distribution de dose en trois dimensions (3D) basée sur un artificial neural network (ANN). L'erreur de prédiction pour tous les voxels était ¡8% pour les cas de prostate testés, ce qui était à l'époque une performance intéressante. Depuis, des efforts importants ont été fait par les chercheurs dans ce sens, et l'utilisation de l'apprentissage profond dans la prédiction de la dose a été largement explorée. Différentes architectures de CNN ont été utilisées pour la prédiction de la distribution de dose en 2D et en 3D.

Campbell et al. [5] ont eux développé un modèle ANN de prédiction de la distribution de la dose en 3D pour la chirurgie stéréotaxique du pancréas. Les erreurs de dose moyennes prédites étaient ¡5%. D'excellentes performances du modèle ont été démontrées pour le volume recevant une dose supérieure à 25 Gy, mais des erreurs de prédiction beaucoup plus importantes ont été observées dans la zone de dose inférieure.

Ou encore Kajikawa et al [6] ont prédit l'admissibilité dosimétrique de patients atteints de cancer de la prostate traités par IMRT en utilisant un système neuronal convolutif appelé Alex-Net. Le réseau Alex-Net a été pré-entraîné avec un grand jeu de données ouvert appelé Image-Net puis fine-tuné sur un nouvel ensemble de données de CT-scan. Cette méthode n'était pas la meilleure en termes de précision de la prédiction. Cependant, elle a ouvert une nouvelle direction pour l'utilisation de transfert learning dans ce domaine.

Une amélioration importante des performances a eu lieu avec Nguyen et al. [3] qui ont proposé une architecture U-Net 2D modifiée pour la prédiction de la distribution de dose afin de traiter le cancer de la prostate. En plus des améliorations de performances, un autre élément intéressant : les tomodensitométrie (CT-scan) sont introduits directement dans le modèle, aucune extraction ou sélection manuelle de caractéristiques n'a été nécessaire. Nguyen et al. [2] ont également proposé un U-Net hiérarchiquement dense pour la prédiction de la distribution de dose 3D notamment pour traiter les patients atteints de cancer de la tête et du cou.

Par rapport aux ANNs, les CNNs et U-net ont amélioré la prédiction de la distribution de dose, grâce à leur capacité à extraire des caractéristiques locales et globales des CT-scans des patients dans des dimensions plus élevées. Toutes ces études démontrent donc le potentiel des méthodes Deep Learning dans l'amélioration de l'efficacité et de la précision pour la planification et le traitement par radiothérapie. Cependant, d'autres recherches et développements sont encore nécessaires pour relever les défis et les limites de ces méthodes et ainsi d'améliorer leur généralisation et leur applicabilité clinique.

3 Approche

Depuis son introduction en 2015 [1], le U-net s'est avéré très efficace dans les tâches de segmentation sémantique d'images biomédicales. Le U-Net est une architecture de réseau neuronal convolutif qui se compose d'une partie d'encodage (qui réduit la taille de l'image tout en préservant les caractéristiques importantes) et d'une partie de décodage (qui restaure la taille de l'image et prédit les masques de segmentation). Voici un exemple d'architecture dans la figure suivante :

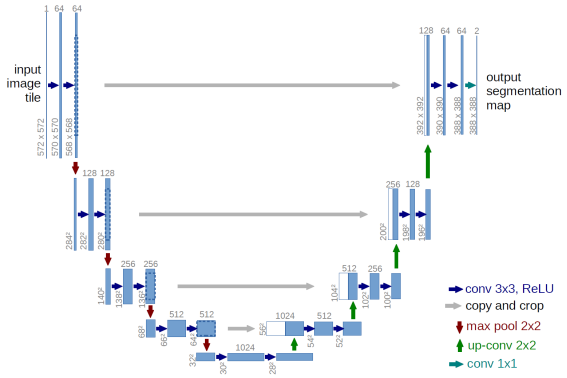


Figure 1: Architecture d'un U-net

L'U-Net possède deux caractéristiques principales :

- La taille de l'image d'entrée est sous-échantillonnée par 2 à chaque bloc par une couche appelée "Max-Pooling" dans la partie encodeur du modèle. Dans la partie décodeur, les caractéristiques extraites sont upsampled progressivement en utilisant une convolution transposée.
- Deuxièmement, afin de conserver les informations de haute résolution, des "skip-connections" sont utilisés pour passer les informations de la partie encodeur du réseau à la partie décodeur.

L'avantage de cette architecture : sa capacité à effectuer une segmentation précise en utilisant une quantité relativement faible de données d'entraînement. Cela est dû à la combinaison de ces deux stratégies : l'utilisation de l'architecture en U qui permet la segmentation à différentes échelles spatiales, et l'utilisation de "skip-connections" qui préservent les informations spatiales importantes à différentes échelles spatiales pendant la phase de décodage. Plus précisément les "skip-connections" permettent de connecter les couches d'encodage aux couches de décodage correspondantes, afin de préserver les informations spatiales importantes qui ont été perdues pendant la phase d'encodage. Cette technique améliore la précision de la segmentation en permettant au réseau de conserver des informations sur la position spatiale des structures anatomiques.

L'architecture U-Net a été largement utilisée pour la segmentation d'images biomédicales, notamment dans le domaine de la neuro-imagerie, de la segmentation d'organes et de la détection de tumeurs. Elle est également utilisée pour des taches de regression tel que prédire la dose de radiation pour le traitement du cancer. Ainsi dans ce projet, nous allons ainsi adapter l'architecture du U-net afin de répondre au mieux à nos doubles problèmes. Nous étudierons l'influence des hyperparamètres sur le score final notamment l'influence de la profondeur et des skip-connections.

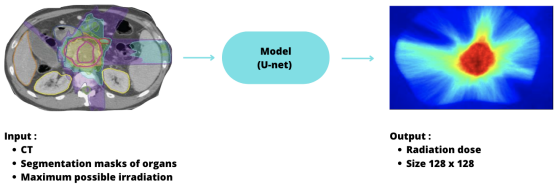


Figure 2: Etape de notre solution

Ainsi notre solution à l'architecture suivante :

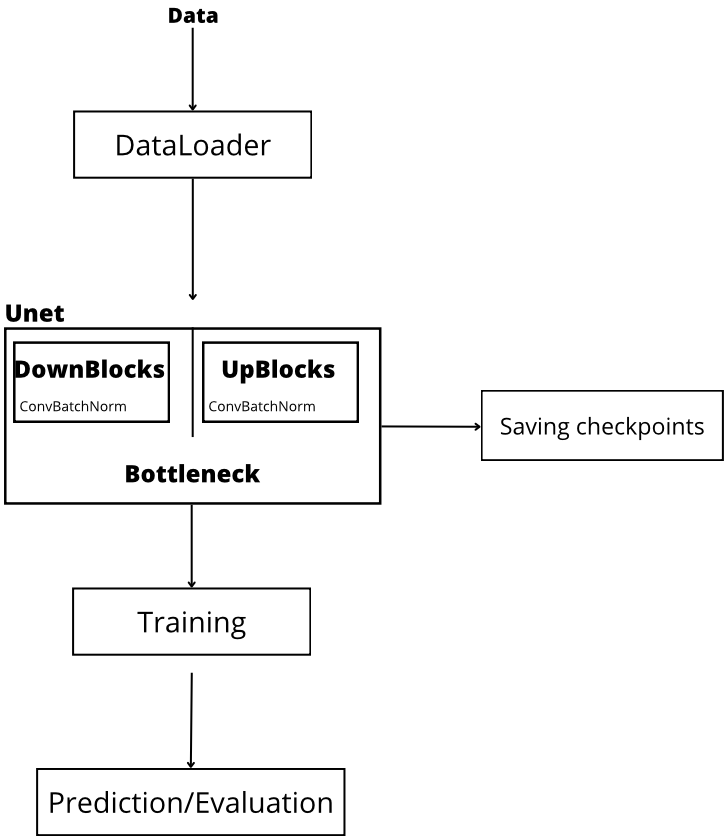


Figure 3: Schéma des fonctions implémentées

4 Cas en deux dimensions

4.1 Data

Le jeu de données est structuré en trois dossiers : un dossier d'entraînement contenant les données sur lesquelles notre modèle sera entraîné, un dossier de validation pour s'assurer que le modèle n'overfit/underfit pas, et un dossier de test qui servira à la prédiction de la dose de radiation.
Chacun de ces dossiers contient plusieurs sous-dossiers. Chaque sous-dossier correspond à un patient, et on y trouve : :

- Un CT-scan (128x128)
- Un fichier segmentant les différents organes (10x128x128)

- Un fichier caractérisant l'irradiation maximale possible (128x128)
- **Pour les dossier entraînement et la validation on trouve en plus** : La répartition de la dose de radiation (128x128)

Des exemples ci-suit :

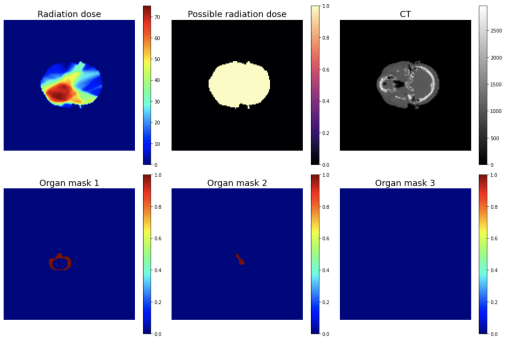


Figure 4: Données en entrée du modèle

4.2 Les métriques considérées pour évaluer les performances

Il s'agit d'un problème de régression donc on considère 3 métriques différentes pour évaluer notre modèle :

- La Mean Square Error (MSE) : Moyenne des écarts entre valeur prédite et valeur réelle au carré. L'avantage de la MSE : simple à calculer, elle est différentiable on peut donc aussi l'utiliser comme fonction de perte. Inconvénient : l'unité de sortie est au carré. Par ailleurs elle est sensible aux outliers : S'il y a des valeurs aberrantes dans l'ensemble de données, elles sont pénalisées plus fortement qu'avec par exemple la MAE. Dans le cadre d'une image la MSE correspond à la moyenne des écarts entre valeur prédite et valeur réelle au carré sur l'ensemble des pixels.
- La Root Mean Square Error (RMSE) : Il s'agit de la racine carrée de la MSE, Ses avantages : Son unité est la même que l'unité de notre target et elle est moins sensible aux outliers que la MSE.
- La Mean Absolute Error (MAE) : Il s'agit de la somme de la valeur absolue des écarts entre valeur prédite et valeur réelle. Son avantage : elle est plus robuste face aux outliers que la RMSE et la MSE. En revanche, elle n'est pas différentiable et ne peut donc pas être utilisée comme fonction de perte. Dans le cadre d'une image la MSE correspond à la moyenne des écarts absolus entre valeur prédite et valeur réelle sur l'ensemble des pixels.

Le challenge hiérarchise les solutions selon la valeur de la MAE sur le test set, ainsi on se focalisera principalement sur cette métrique pour la suite.

4.3 Les fonctions coûts considérées pour entraîner le modèle

Initialement nous avions considérés 2 fonctions de coûts différentes pour entraîner notre modèle :

- La norme L^1
- La norme L^2

Néanmoins pour des entraînements sur de longues périodes nous nous sommes rendus compte que la norme L^1 était plus stable que la norme L^2. Nous avons donc entraîné pour la suite avec la L^1 Loss.

4.4 Choix de la profondeur

Nous avons d'aborder commencé par une première expérience simple : on part d'un U-net très proche de celui présenté pour la première fois en 2015 [1] conçu afin de réaliser de la segmentation de tumeurs sur des scans de cerveaux. Il nous faut adapter ce U-net à notre problème : en effet ce modèle classifie les pixels selon 4 catégories et répond ainsi à un problème de classification. Dans le cas de la prédiction de dose de radiation, il s'agit d'un problème de régression.
Pour cette première expérience, on garde une architecture quasiment identique : on modifie la dernière couche en remplaçant la fonction softmax par une fonction ReLu. On remarque qu'il manque une skip-connections dans l'architecture : la partie décodeur ne prend pas en entrée la première sortie du modèle (juste avant le premier Max-Pooling). On rajoute ainsi cette connexion. On change également le nombre de channels d'entrée et de sorties : on passe respectivement de 4 à 12 et de 4 à 1. En effet notre modèle doit prendre 12 channels en entrée puisqu'en stackant les 3 fichiers évoqués ci-dessus on arrive à un format 12x128x128. Après une étude sur le learning rate et le batch size on prend respectivement comme valeur 10^{-2} et 32. On prend un batch size relativement petit afin d'éviter le surapprentissage et ne pas surcharger la mémoire. On entraîne notre modèle grâce à Google Collab sur une Google Compute Engine Python 3. Par ailleurs on utilise l'optimizer Adam en raison de son caractère adaptatif : il ajuste automatiquement le taux d'apprentissage en fonction des caractéristiques du gradient et de la variation des poids du modèle. Cette première architecture est de profondeur 4, on la note "U-net depth 4". Pour cette première expérience on obtient une MAE de 0.47, une MSE de 3.57 et une RMSE de 1.90 ce qui n'est pas si mauvais pour un premier essai.

Il s'agit à présent de modifier l'architecture de notre modèle afin d'améliorer sa performance. Tout d'abord analysons quels sont les hyperparamètres que l'on peut modifier ? Le U-net est constituer d'une partie **encodeur** qui est caractérisé dans le code par une succession de couches DownBlock, suivi par une couche Bottleneck (utilisée pour réduire le nombre de paramètres du modèle et éviter le surapprentissage tout en préservant la capacité de représentation de l'information) et enfin une partie **décodeur** caractérisé dans le code par une succession de couches UpBlock. Il y a autant de couche DownBlock que de couche UpBlock, dans ce rapport on appelle cet hyperparamètre la **profondeur** du U-net. Un autre hyperparamètre que l'on peut faire varier est le nombre de "skip-connections" qui connectent les parties encodeur et décodeur du U-net. On adoptera la notation $(-i)$ lorsque l'on désigne une architecture complètement connectée à laquelle on a supprimée i skip-connections.

On réalise une série d'expériences dans lesquelles on fait varier la profondeur de notre U-net ainsi que le nombre de "skip-connections". Ainsi la profondeur varie entre 3 et 6 et le nombre de "skip-connections" supprimées entre 0 et 2. Les résultats obtenus sont présentés dans le tableau ci-dessous :

Modèle	**MAE**	**MSE**	**RMSE**
U-net depth 3	0.59	3.94	1.98
U-net depth 4	0.47	3.57	1.90
U-net depth 4 (-1)	0.55	3.86	1.95
U-net depth 4 (-2)	0.74	4.87	2.20
U-net depth 5	**0.35**	**2.77**	**1.66**
U-net depth 5 (-1)	0.42	3.17	1.78
U-net depth 5 (-2)	0.63	4.2	2.05
U-net depth 6	0.51	3.40	1.84
U-net depth 6 (-1)	0.49	3.36	1.83
U-net depth 6 (-2)	0.48	3.32	1.82

Table 1: Ensemble des résultats obtenus pour différentes architectures

On obtient les meilleures performances pour l'architecture **U-net depth 5**, ce qui correspond à un U-net de profondeur 5 complètement connecté. On remarque que la performance s'améliore lorsque la profondeur passe de 3 à 5 puis diminue lorsque l'on passe à 6. On peut expliquer cela par le fait que les modèles 3 et 4 underfitt tandis que le modèle 6 overfitt. Par ailleurs à profondeur fixée on remarque que diminuer le nombre de skip-connections réduit la performance du modèle sauf pour le modèle 6. Une façon d'expliquer cela : lorsque l'on diminue le nom-

bre de skip-connections on réduit le nombre de paramètres et de connections entre la partie encodeur et décodeur. Dans le cas des modèles 3,4,5 les performances diminuent parce que l'on accentue l'underfitting. Tandis que pour le modèle 6 l'overfitting de la profondeur est compensée par l'underfitting provoquée par la suppression des skip-connections.
On remarque également que le modèle U-net depth 6 perfome moins bien que le modèle U-net depth 4 par rapport à la MAE mais mieux par rapport à la MSE. Cela peut s'expliquer par le fait que le modèle U-net depth 6 performe mieux sur certains outliers.
On peut visualiser la performance de notre modèle le plus performant **U-net depth 5** sur un exemple dans la figure ci-dessous :

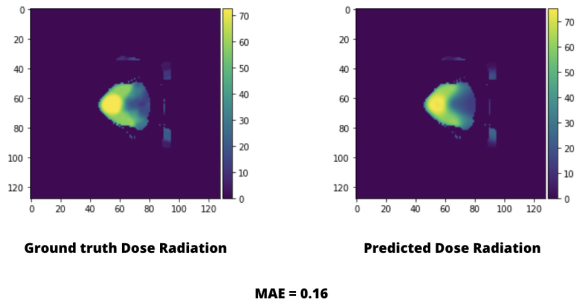


Figure 5: Comparaison visuelle prédiction Vs dose radiation réelle

Le modèle arrive bien à reproduire la répartition attendue, que ce soit au niveau de la forme globale ou des zones de fortes intensités.

4.5 Choix des blokcs

Nous avons ensuite cherché à améliorer les prédictions de l'U-net-5. Pour ce faire, nous avons mené plusieurs expériences pour ajouter des blocs potentiellement pertinents et évaluer leur impact sur les résultats.

4.5.1 Resblock

Une première idée nous est venue de l'article [7]. En effet, dans cet article, leurs blocs encodeurs étaient des Resblocks. Nous avons donc essayé de les ajouter à notre architecture, car nous supposions qu'ils pourraient apprendre des fonctions plus complexes. Nous avons d'abord testé la version de base du Resblock et cela a montré des résultats intéressants, nous avons donc continué à les explorer et avons finalement utilisé la version améliorée détaillée dans [8]. Les tests qui en ont résulté sont les suivants :
Tout d'abord, une rapide comparaison de 2 époques a montré que mettre un Resblock au début du modèle est très bénéfique :

Couche d'entrée	**Resblock**	ConvBatchNorm
MAE	**0.72**	1.08

Table 2: Résultats obtenus avec/sans un Resblocks à l'entrée du réseau

Ensuite, nous les avons placés dans les blocs encodeurs et avons obtenu les résultats suivants pour 50 époques :

Encodeur bloc	Resblock	ConvBatchNorm
MAE	0.43	0.41

Table 3: Résultats obtenus avec/sans un Resblocks dans l'encodeur

Les résultats sont extrêmement similaires, nous avons également essayé de voir si les Resblocks auraient un impact sur le Décodeur/Bottleneck, mais nous avons abandonné cette idée car cela n'a pas fonctionné. Nous avons donc décidé d'opter pour une architecture plus légère, en ayant un Resblock uniquement à l'entrée de notre U-net.

4.5.2 SE Blocks

Dans la littérature [9] [10], des blocs Squeeze-and-Excitation sont utilisés pour apprendre des caractéristiques pertinentes. Nous avons essayé d'effectuer une première implémention pour les ajouter à notre réseau et nous avons obtenu les résultats suivants après 5 époques :

SE location	**Nowhere**	Encoder	Bottleneck
MAE	**0.63**	0.65	0.68

Table 4: Résultats obtenus avec des SE blocks pour 5 epoques

Malheureusement, ils n'ont pas aidé à améliorer notre modèle, nous avons donc abandonné cette idée.

4.6 Expériences avec de la data augmentation

Afin d'améliorer les performances du modèle, il est également possible de réaliser de l'augmentation de données. En effet, la performance des U-net peut varier considérablement en fonction de la qualité et de la quantité des données d'entraînement disponibles. L'idée de l'augmentation des données est de créer de nouveaux échantillons artificiels et ainsi d'augmenter le nombre et la diversité des exemples sur lesquels notre réseau s'entraîne. L'objectif est d'élargir la variabilité des données d'entraînement afin de permettre au réseau de neurones d'apprendre des caractéristiques plus robustes et généralisables. C'est d'ailleurs ce qui est réalisé dans l'article [1].

Pour augmenter les données, il existe plusieurs transformations possibles : des rotations, des translations, des zooms, des réflexions, des déformations, des modifications de la luminosité, du contraste, etc. Dans notre cas on va s'intéresser principalement aux rotations de 90 degrés, aux transformations affines, aux zooms et aux Axialflip. On applique ainsi ces transformations afin d'obtenir de nouveaux datasets d'entraînement et de validation. On adapte également nos hyperparamètres pour optimiser l'entraînement : on les laisse à l'identique excepté le learning rate que l'on fixe à 10^{-4}.

On essaye avec toutes les transformations puis on sélectionne uniquement certaines d'entre elles. Le modèle semble le plus performant en utilisant uniquement les transformations rotation de 90 degrés et Axialflip. Mais malgré toutes ces optimisations on ne parvient pas à obtenir une MAE inférieure à 0.42.
Première hypothèse : les méthodes de data augmentation mises en place ne sont pas pertinentes ou ne sont pas implémentées correctement. Un autre explication possible : notre dataset test est en réalité assez proche de nos datasets de validation et d'entraînement, ce qui fait que pour bien performer sur le challenge il n'est pas nécessaire d'être capable de bien généraliser, cela pénalise même le score. Il faudrait pour s'assurer de la validité de cette hypothèse faire des tests sur un autre jeu de données plus diversifiées.

4.7 Etude d'ablation

Des études d'ablation ont été menées pour déterminer la longueur optimale de notre modèle, ainsi que la pertinence du Resblock dans la section précédente. Ces résultats soulignent l'importance de choisir la bonne profondeur pour l'U-net afin d'obtenir des scores plus élevés. Nous avons également effectué un test pour supprimer le Bottleneck de l'U-net, ce qui a donné un score de 1,27 après 2 époques, comparé à 0,7 avec lui. Ce test a prouvé que le Bottleneck était extrêmement important, car pendant l'entraînement, le réseau avait tendance à overfitter après 2 époques, avec une loss d'environ 0,8 mais une MAE de validation plus élevée de 1,27.

4.8 Modèle final et paramètres

4.8.1 Modèle

Ainsi après nos différentes expériences, nous avons choisi un modèle : U-net de profondeur 5, avec un Resblock à l'entrée du réseau et des dropout après chaque bloc pour augmenter la généralisation.

4.8.2 Hyperparamètres

Les paramétres finaux sont :

- Learning rate : 10e-4
- Weight decay : 10e-5
- Nombre d'epoques : 100
- Taille du Batch : 32

5 Cas en trois dimensions

Après avoir traité le problème en 2 dimensions sur des données mise à disposition dans le cadre d'un cours d'imagerie médicale nous nous attaquons au cas 3D originellement proposé dans le cadre du défi OpenKBP lancé par l'Association américaine de médecine. Il s'agit ainsi de prédire la dose de radiation pour chaque voxel. Il y a plusieurs différences avec le cas 2D. Tout d'abord il faut modifier le preprocessing des données ainsi que le DataLoader. Ensuite il faut adapter l'architecture de notre réseau au cas 3D.

5.1 Data, preprocessing et DataLoader

Dans le cas 2D, l'ensemble de données fourni se composait d'images 2D de 10 200 patients, divisées en un training set de 7800 patients, et un ensemble de validation et de test comprenant chacun 1200 patients. Nous prenions en entrée un CT-scan (128x128), un fichier caractérisant la délimitation des différents organes (10x128x128) et un fichier caractérisant l'irradiation maximale possible (128x128) afin de prédire la répartition de la dose de radiation (128x128). Dans le cas 3D, l'Association américaine de médecine fournit des données pour 340 patients qui ont été traités pour un cancer de la tête et du cou avec une radiothérapie à modulation d'intensité. Les données sont divisées en un training set (n=200), de validation (n=40) et de test (n=100). Les données d'entrée sont la répartition de la dose de radiation (128x128x128), un CT-scan (128x128x128), un fichier caractérisant l'irradiation maximale possible (c'est-à-dire la zone dans laquelle la dose peut être non nulle) et un fichier caractérisant la délimitation des différents organes (10x128x128x128). Notre set de données en 2 dimensions est ainsi très similaire à celui que nous avions en 2 dimensions.

Pour le pre-processing on réalise principalement deux transformations :

- On aligne correctement les données qui sont de la forme $(h, w, -z, c)$ ou $(y, x, -z, c)$ en les mettant sous la forme (c, z, x, y).
- De façon similaire au cas 2D, on réalise une concaténation de toutes les données en un seul tenseur.

5.2 Adaptation de l'architecture à la 3D

Nous sommes tout d'abord parti du modèle de base donné par le challenge qui était un U-net de profondeur 6. Nous l'avons entraîne avec plusieurs learning rate mais les résultats n'étaient pas très satisfaisant : une mae de 1.6 et une dose error de 21. Comme les données du cas 2D était une adaptation du cas en 3D, elles étaient très similaires, nous avons donc décidé de reprendre l'architecture que l'on avait implémenté en 2D et essayer de l'adapter à de la 3D. Pour cela, nous avons :

- Repris notre fonction get_activation
- Transformé au sein des différents blocs de notre réseau (DownBlock,Bottleneck,UpBlock) les con2D en conv3D avec les bon paramètres, ainsi que les BatchNorm en BatchNorm3D
- Utiliser la ConvTranspose3D dans les Upblock.

Cette transformation des blocks nous a permis d'améliorer les performances en obtenant une mae de 1.1 et une dose error de 13. Nous avons ensuite rajouté des dropout dans le réseau tout comme en 2D ce qui nous a permis de passer à une mae de 0.9 et une dose error de 10.

5.3 Présentation des métriques d'évaluation

Le challenge de l'Association américaine de médecine évalue le modèle selon deux métriques :

- l'erreur de dose qui mesure la différence absolue moyenne entre une soumission et son plan synthétique correspondant (c'est-à-dire la différence absolue moyenne de dose voxel par voxel)

- l'erreur DVH qui mesure la différence absolue des critères DVH entre une soumission et son plan synthétique correspondant. Plus précisément, DVH signifie "Dose-Volume Histogramme". Il est utilisé pour quantifier l'efficacité d'un traitement de radiothérapie en évaluant la couverture de dose du volume cible et en évaluant la préservation des organes à risque adjacents.

L'erreur DVH n'est pas évidente à calculé dans le cadre de notre étude nous nous sommes concentré sur l'erreur de dose et la MAE.

5.4 Choix du nombre de couches

Comme dans le cas en 2 dimensions, nous avons exploré différentes profondeurs d'architectures. Nous introduisons la notation U-net3D qui correspond à l'architecture U-net adaptée en 3D. Nous entrainons chaque architectures sur 15 epochs avec un learning rate de 0.01 afin d'obtenir ce tableau de comparaison.

Modèle	**MAE**	**Dose**
U-net3D depth 3	1.28	25
U-net3D depth 4	**1.18**	**20.3**
U-net3D depth 4 (-1)	1.21	21.6
U-net3D depth 4 (-2)	1.23	22.7
U-netD depth 5	1.30	25.9
U-net3D depth 5 (-1)	1.27	24.7
U-net3D depth 5 (-2)	1.25	23.3

Table 5: Ensemble des résultats obtenus pour différentes architectures

L'architecture U-net3D depth 4 est la plus performante. On remarque que l'architecture U-net3D depth 3 performe mieux que l'architecture U-net3D depth 5. En revanche, les architectures U-net3D depth 5 (-1) et U-net3D depth 5 (-2) performent mieux que U-net3D depth 3. Cela peut s'expliquer par le fait que lorsqu'en retirant des skip-connections à l'architecture U-net3D depth 5, on diminue l'overfitting. En revanche U-net3D depth 4 (-1) et U-net3D depth 4 (-2) performent moins bien que U-net3D depth 4, cela peut s'expliquer par le fait que ces deux architectures underfit.

5.5 Resblock

Après nos différentes expériences de la profondeur des couches, et au vu de l'intérêt qu'apportait l'ajout d'un Resblock à l'entrée du réseau en 2D, nous avons donc testé si un Resblock à l'entrée du réseau pour les images 3D aurait le même effet. Les résultats sur 15 époques sont dans le tableau suivant :

Entrée	**Resblock3D**	ConvBatchNorm3D
MAE	**0.50**	0.82
Dose score	**8.03**	12.1

Table 6: Score obtenus avec/sans Resblocks sur 15 époques

Nous avons également remarqué que pendant les 15 époques, le score de validation atteignait le meilleur score de 0.35 à la 14e époque (il a donc légèrement remonté à la 15eme) pour le Unet avec le Resblock3D, alors que sans, le meilleur score atteint est de 0.79 à la 9e époque.
Nous en concluons donc l'intérêt du Resblock également pour le cas en 3D.

5.6 Modèle final et paramètres

5.6.1 Modèle

Ainsi après nos différentes expériences, nous avons choisi un modèle en 3D : U-net3D de profondeur 4, avec un Resblock à l'entrée du réseau et des dropout après chaque bloc pour augmenter la généralisation.

5.6.2 Hyperparamètres

Les paramétres finaux sont :

- Learning rate : 1e-2
- Weight decay : 1e-4
- Nombre d'epoques : 15
- Taille du Batch : 1

Ci-dessous un exemple d'entraînement. On observe que la loss diminue bien, ainsi l'entraînement est assez stable :

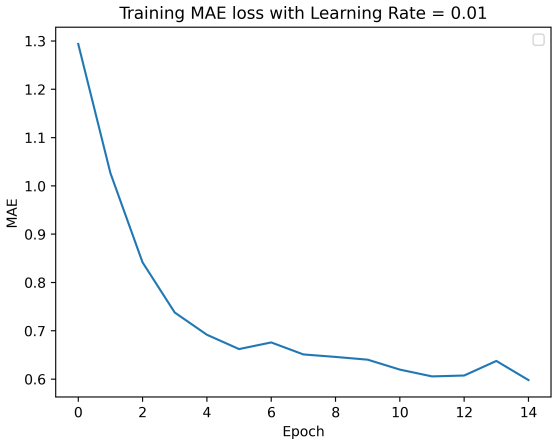


Figure 6: Évolution de la training loss

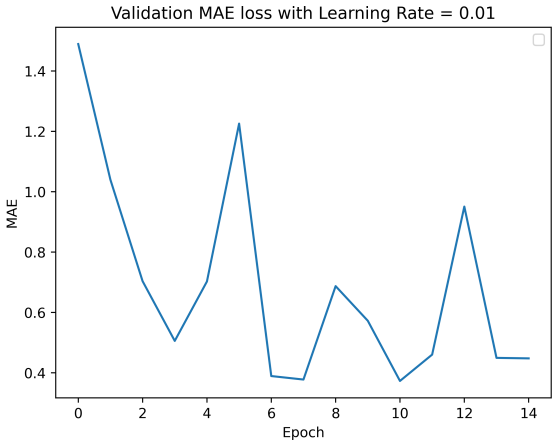


Figure 7: Évolution de la MAE du validation set

6 Conclusion

En conclusion, pour résoudre le problème de prédiction de dose de radiation, nous avons utilisé un modèle U-net. Sur ce réseau avons pu explorer l'influence de différentes profondeurs, des hyperparamètres, et des types de blocs utilisé, ainsi que nous nous sommes confronté à la tache du 3D. Nous avons ainsi pu voir avec ce projet l'intérêt et le potentiel que peut avoir le Deep Learning pour la tache qui s'avère parfois difficile qu'est la radiothérapie.
Les pistes d'améliorations du projets seraient d'essayer d'optimiser au mieux le Unet en 3D afin d'avoir des meilleurs résultats notamment en testant la data augmentation ou des mécanismes d'attentions dans le réseau. Une autre approche serait de tester d'effectuer la tache avec un GAN.

References

[1] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18. Springer International Publishing, 2015.

[2] Nguyen, D., Jia, X., Sher, D., Lin, M. H., Iqbal, Z., Liu, H., & Jiang, S. (2019). 3D radiotherapy dose prediction on head and neck cancer patients with a hierarchically densely connected U-net deep learning architecture. Physics in medicine Biology, 64(6), 065020.

[3] Nguyen, D., Long, T., Jia, X., Lu, W., Gu, X., Iqbal, Z., & Jiang, S. (2019). A feasibility study for predicting optimal radiation therapy dose distributions of prostate cancer patients from patient anatomy using deep learning. Scientific reports, 9(1), 1076.

[4] Shiraishi S, Moore KL. Knowledge-based prediction of three-dimensional dose distributions for external beam radiotherapy. Med Phys. (2016) 43:378. doi: 10.1118/1.4938583

[5] Campbell W, Miften M, Olsen L, Stumpf PK, Schefter TE, Goodman KA, et al. Neural network dose models for knowledge-based planning in pancreatic SBRT. Med Phys. (2017) 44:6148–58. doi: 10.1002/mp.12621

[6] Kajikawa T, Kadoya N, Ito K, Takayama Y, Chiba T, Tomori S, et al. A convolutional neural network approach for IMRT dose distribution prediction in prostate cancer patients. J Radiat Res. (2019) 60:685– 93. doi: 10.1093/jrr/rrz051

[7] D.Zhang,R.Confidence,andU.Anazodo,"Strokelesionsegmentationfromlow-qualityandfew-shotmrisviasimilarity- weighted self-ensembling framework," 09 2022, pp. 87–96.

[8] K.He,X.Zhang,S.Ren,andJ.Sun,"Identitymappingsindeepresidualnetworks,"2016

[9] J.Hu,L.Shen,S.Albanie,G.Sun,andE.Wu,"Squeeze-and-excitationnetworks,"2019.

[10] W.Jung,S.Park,K.-H.Jung,andS.Hwang,"Prostatecancersegmentationusingmanifoldmixupu-net,"052019.