



**Intitulé : IA pour une imagerie médicale enrichie et non-invasive**

**Université de Poitiers**

**Année universitaire 2021 - 2022**

**SP2MI, UFR Sciences Fondamentales et Appliquées**

**Master Objets Connectés**

**Rapport de stage**

**Adressé pour la validation du Master 2 : Ingénierie des objets intelligents**

**Intelligence artificielle pour une imagerie médicale enrichie et non-invasive**

**Période de stage : 28 mars 2022 - 01 septembre 2022**

**Soumis par :**

**Marwan AL OMARI**

**Tuteur de stage**

**BOURDON Pascal**

**Laboratoire XLIM CNRS 7252**

**Bât. H1 - SP2MI**

**Enseignant référent :**

**Noël Richard**

**11 Bd Marie et Pierre Curie**

**86073 Poitiers Cedex 9**

## Table des matières

Liste des figures.....	III
Liste des tableaux .....	V
Abréviations et acronymes .....	VII
Résumé .....	X
I. Laboratoires et présentations de stage.....	XI
I.1 Laboratoire XLIM.....	XI
I.2 Laboratoire I3M.....	XII
I.3 Présentation de stage .....	XIII
I.3.1 Cahier des charges.....	XIII
I.3.2 Déroulement et Méthodologie.....	XIII
I.4 Gestion de projet .....	XV
I.4.1 Les systèmes d'exploitation.....	XV
I.4.2 Programmes de logiciels .....	XV
I.4.3 Plateformes de codage.....	XVII
1. Introduction .....	1
2. Travaux relatifs.....	1
3. Background .....	3
3.1 L'apprentissage profond dans le diagnostic médical .....	3
3.2 CNNs .....	5
3.2.1 Convolution (C1, C3, and C5).....	6
3.3.2 Couche de Pooling (S2 and S4) .....	7
3.3.3 FCL (F6).....	8
3.3.4 Fonctions d'activation.....	8
3.3 U-Net.....	10
3.4 GANs.....	12
3.5 U-Net et GAN .....	14
3.5 Jeu de données.....	15
4. Approche .....	16
4.1 Pré-traitement .....	17
4.1.1 Recadrage des images.....	17
4.1.2 Encodage one-hot .....	18
4.1.3 Normalisation .....	19
4.1.4 Filtration .....	19
4.1.5 Correction de l'intensité linéaire.....	19
4.1.6 Correction du biais .....	19

4.1.7 Augmentation des données.....	19
4.1.8 Répartition des données.....	20
4.2 Algorithme.....	21
4.2.1 SIMO.....	21
4.2.2 MIMO.....	22
4.2.3 MISO.....	22
4.3 Evaluation Metrics .....	22
5. Expériences et résultats .....	23
6. Conclusion.....	31
6.1 Professionnel .....	31
6.2 Personnel .....	31
6.3 Travaux futurs .....	31
6.4 Challenges .....	31
6.5 Analyse des risques .....	32
6.5.1 L'indisponibilité de l'ensemble des données.....	32
6.5.2 Utilisateur administratif non subventionné.....	32
6.5.3 L'ambiguïté du cahier des charges .....	33
6.5.4 Entraînement prolongé des architectures DL .....	33
6.5.5 Débogage d'architectures en Pytorch.....	34
6.5.6 Arrêt totale de serveur de calcul.....	35
7. Références .....	36
8. Annexe .....	I
8.1 Ensembles de données :.....	I
8.2 Paramètres de perte .....	I
8.2.1 Score de dice .....	I
8.2.2 Pénalité de gradient L1 .....	I
8.2.3 Segmentation .....	II
8.2.4 Perte de classification.....	II
8.2.5 Perte de reconstruction .....	II
8.3 Deep learning operations (DLOps) .....	III
8.3.1 Modèle de SIMO .....	III
8.4 Etat de l'art .....	IV

## Liste des figures

<b>Figure 1-</b> Représentation du laboratoire XLIM, qui se compose de 3 pôles et de 6 axes scientifiques XI	
<b>Figure 2-</b> Images FLAIR représentatives d'un sujet unique à trois intensités de champ : 1.5T, 3T et 7T, extraites de [23].....	XII
<b>Figure 3-</b> Diagramme de GANT - gestion du temps des tâches .....	XV
<b>Figure 4-</b> Graphique PERT classique - gestion du temps des tâches, représenté en tableau (2) .....	XVI
<b>Figure 5-</b> Établissement d'une connexion SSH avec le serveur en question pour la manipulation SFTP .....	XVII
<b>Figure 6-</b> Connexion SFTP à l'aide de FileZilla .....	XVII
<b>Figure 7-</b> Méthodes SISO, SIMO, MISO et MIMO dans le traitement des modalités IRM .....	5
<b>Figure 8-</b> LeNet-5, un exemple d'architecture CNNs, extrait de [21].....	5
<b>Figure 9-</b> Une déconstruction de l'opération convolutive dans les CNNs .....	6
<b>Figure 10-</b> Exemple de suréchantillonnage .....	7
<b>Figure 11-</b> Exemple de max-pooling avec fliter $2 \times 2$ et stride $2 \times 2$ .....	7
<b>Figure 12-</b> Exemple d'average pooling avec filtre $2 \times 2$ et stride $2 \times 2$ .....	7
<b>Figure 13-</b> Exemple de global max-pooling avec une taille de pool égale à la taille d'entrée.....	8
<b>Figure 14-</b> Exemple de global average pooling avec une taille de pool égale à la taille d'entrée.....	8
<b>Figure 15-</b> Fonction ReLU en représentation graphique .....	9
<b>Figure 16-</b> Fonction LeakyReLU en représentation graphique .....	9
<b>Figure 17-</b> Fonction sigmoïde en représentation graphique .....	10
<b>Figure 18-</b> L'architecture U-Net, qui est extraite de [19][20] .....	11
<b>Figure 19-</b> Architecture du réseau U-Net .....	12
<b>Figure 20-</b> Le concept des GANs .....	13
<b>Figure 21-</b> Démonstration de la cartographie générateur-discriminateur, extraite de [13] .....	14
<b>Figure 22-</b> L'architecture du GAN.....	14
<b>Figure 23-</b> L'architecture d'assemblage de U-Net et des GANs .....	15
<b>Figure 24 –</b> Echantillon de données illustrant les quatre modalités avec carte de segmentation (vérité terrain de tumeur) .....	15
<b>Figure 25-</b> Architecture du programme, spécifiant les différentes étapes de la saisie des données, du prétraitement, de l'augmentation, de l'algorithme, des résultats et des mesures d'évaluation. Note : le processus marqué en rouge a été suspendu sur les architectures finales .....	16
<b>Figure 26 –</b> Le résultat de l'étape de prétraitement, y compris la correction des biais, la normalisation, le filtrage et le recadrage. ....	17
<b>Figure 27-</b> Redimensionnement de l'image de $240 \times 240 \times 155$ à $128 \times 128 \times 155$ .....	18
<b>Figure 28-</b> rognage de l'image de $168 \times 168 \times 155$ à $128 \times 128 \times 155$ , mettant en évidence la zone de la tumeur. ....	18
<b>Figure 29-</b> Augmentation des données, y compris : des rotations à $90^\circ$ et $180^\circ$ et des retournements de gauche à droite.....	20
<b>Figure 30-</b> Données divisées en ensembles de test et de formation pour les prototypes SIMO, MISO T et MISO F rogné avec une taille d'entrée d'image de $128 \times 128$ . ....	20
<b>Figure 31-</b> Données divisées en ensembles de test et d'apprentissage pour le prototype MISO F avec un rognage de $128 \times 128$ à l'entrée de l'image.....	21
<b>Figure 32-</b> L'architecture SIMO prend T1 en entrée, et synthétise T2, FLAIR, T1c, et la carte de segmentation pour la discrimination .....	21
<b>Figure 33-</b> L'architecture MIMO prend T1 et T2 en entrée, et synthétise T1c et FLAIR pour la discrimination.....	22
<b>Figure 34-</b> MISO prend T1 et T2 en entrée, et synthétise FLAIR pour la discrimination .....	22
<b>Figure 35-</b> MISO prend T1 et T2, et FLAIR en entrée, et synthétise T1c pour la discrimination.....	22

<b>Figure 36-</b> Matrice de probabilité/gravité du risque de non-disponibilité de l'ensemble de données..	32
<b>Figure 37-</b> Matrice de probabilité/gravité du risque de non-subventionnement de l'utilisateur de l'administration .....	33
<b>Figure 38-</b> Matrice de probabilité/gravité du risque d'ambiguïté des spécifications .....	33
<b>Figure 39-</b> Matrice de probabilité/sévérité du risque de réalisation prolongée des modèles.....	34
<b>Figure 40-</b> Multiprocessus et erreur de mémoire.....	34
<b>Figure 41-</b> Matrice de probabilité/sévérité du risque de formation prolongée des Multiprocessus et erreur de mémoire .....	35
<b>Figure 42-</b> Matrice de probabilité/gravité du risque l'arrêt du serveur de calcul .....	35
<b>Figure 43-</b> Le modèle SIMO, en tant que microservice, accessible sur le service local pour le débogage 127.0.0.1: 5000.....	III
<b>Figure 44-</b> Les résultats du modèle déployé SIMO en prenant un exemple d'image d'internet .....	IV
<b>Figure 45-</b> Les résultats du modèle déployé SIMO en prenant un exemple réel de BRATS2018 .....	IV

## Liste des tableaux

<b>Tableau 1-</b> Spécifications de l'ordinateur personnelle.....	XIV
<b>Tableau 2-</b> Diagramme de PERT - gestion des tâches .....	XV
<b>Tableau 3-</b> BRATS2018 modality distribution over 285 patients .....	15
<b>Tableau 4-</b> Encodage one-hot des étiquettes de tumeur et de non-tumeur.....	19
<b>Tableau 5-</b> Spécifications de l'ordinateur de laboratoire .....	23
<b>Tableau 6-</b> Score de U-Net des 4 architectures, SIMO a atteint 87% ACC, MISO T a atteint 0.414%, MISO F redimensionné a atteint 0.414%, et MISO F rogné a atteint 0.548% .....	23
<b>Tableau 7-</b> Score de U-Net, une comparaison entre SIMO, MISO T, MISO F redimensionné, et MISO F rogné.....	25
<b>Tableau 8-</b> Score de SIMO, une comparaison entre les pertes de discrimination, génération, segmentation.....	26
<b>Tableau 9-</b> Score de SIMO, une comparaison entre les pertes de générateur .....	27
<b>Tableau 10-</b> Score de SIMO, une comparaison entre les pertes de discriminateur .....	27
<b>Tableau 11-</b> Score de SIMO, une comparaison entre les pertes de discriminateur et de générateur....	27
<b>Tableau 12-</b> Score de MISO T, une comparaison entre les pertes de discrimination, génération, segmentation.....	28
<b>Tableau 13-</b> Score de MISO T, une comparaison entre les pertes de générateur.....	28
<b>Tableau 14-</b> Score de MISO T, une comparaison entre les pertes de discriminateur.....	28
<b>Table 15-</b> Score de MISO T, une comparaison entre les pertes de discriminateur et de générateur ....	29
<b>Tableau 16-</b> Score de MISO F redimensionné, une comparaison entre les pertes de discrimination, génération, segmentation.....	29
<b>Tableau 17-</b> Score de MISO F redimensionné, une comparaison entre les pertes de générateur.....	30
<b>Tableau 18-</b> Score de MISO F redimensionné, une comparaison entre les pertes de discriminateur ..	30
<b>Tableau 19-</b> Score de MISO F redimensionné, une comparaison entre les pertes de discriminateur et de générateur .....	30
<b>Tableau 20-</b> Résumé des articles en ce qui concerne le numéro de l'étude, le jeu de données, l'approche utilisée, les résultats de l'expérience, les défis et les travaux futurs possibles. ....	VII

## **Remerciements**

J'adresse mes sincères remerciements tout d'abord à mon superviseur principal Pascal BOURDON, et à ma co-superviseuse Christine FERNANDEZ pour le suivi constant et les instructions directives tout au long du projet. J'aimerais également remercier mon tuteur Noël RICHARD pour son enseignement sur les différents aspects de la gestion de projet. Je dois également remercier le responsable du programme de master, Clency PERRINE, pour ses conseils et son organisation ferme. De plus, j'aimerais remercier le personnel d'XLIM et d'I3M, car j'ai eu l'occasion d'y travailler et d'établir de nouvelles connexions.

Enfin, j'aimerais adresser mes sincères remerciements à Campus France pour avoir financé mes études de master et une année entière de langue française en CFLE (Centre de Français Langue Etrangère) à Poitiers. J'apprécie vraiment l'opportunité de changement de vie qui m'a été donnée, à Poitiers, en France. En outre, je n'oublierais pas ma famille, qui est restée à mes côtés et m'a soutenu dans les moments difficiles pendant mes études de master ainsi que pendant mon stage. Je dois mentionner que je n'avais pas beaucoup de connaissances en langue française au début du programme de master. Je me souviens encore des jours que je passais à être bloquée, sans aucune avancée, j'essayais de comprendre quelques mots de la langue. Mais comme tout le reste dans ce monde, ça s'améliore au fur et à mesure que le temps passe.

## Abréviations et acronymes

Dans cette section, les abréviations et acronymes utilisés dans ce rapport sont clarifiés afin d'éviter la répétition des mêmes termes. Ces termes sont les suivants :

*Machine Learning* (ML)

*Deep Learning* (DL)

*Single-Input Single-Output* (SISO)

*Multi-Input Single-Output* (MISO)

*Multi-Input Multi-Output* (MIMO)

*Single-Input Multi-Output* (SIMO)

Centre Hospitalier Universitaire (CHU)

Imagerie Métabolique Multi-noyaux Multi-organes (I3M)

*BRAin Tumor Segmentation Challenge 2015* (BRATS2015)

*BRAin Tumor Segmentation Challenge 2018* (BRATS2018)

*Brain Tumor Segmentation Challenge* (BRATS)

*The CelebFaces Attributes* (CelebA)

*The Radboud Faces Database* (RaFD)

*T1-weighted* (T1)

*T1 and Contrast-enhance* (T1c)

*T2-weighted* (T2)

*Fluid-Attenuated Inversion Recovery* (FLAIR)

*High Grade Glioma* (HGG)

*Lower Grade Glioma* (LGG)

*Convolutional Neural Networks* (CNNs)

*Fully-Connected Layers* (FCL)

*Generative Adversarial Network* (GAN)

*Generative Adversarial Networks* (GANs)

Centre National de la Recherche Scientifique (CNRS)

*Convolutional PatchedGANs* (CGANs)

*Artificial Neural Networks* (ANNs)

*Deep Neural Networks* (DNNs)

*Recurrent Neural Networks* (RNNs)



*Rectified Linear Unit* (ReLU)

*CO*rona*VI*rus *D*isease **19** (Covid-19)

*Internet Protocol* (IP)

*Cross Validation* (CV)

*AD*aptive *M*oment estimation (ADAM)

Stochastic **G**radient **D**escent (SGD)

Imagerie par **R**ésonance **M**agnétique nucléaire (IRM)

Traitement **A**utomatique des **L**angues (TAL)

*ACC*uracy (ACC)

Intelligence Artificielle (IA)

*Black and White* (BW)

Taux d'apprentissage (TA)

Technique d'évaluation et d'examen de programmes/*Program Evaluation and Review Technique* (PERT)

*Machine Translation* (MT)

*Secure SHell* (SSH)

*SSH File Transfer Protocol* (SFTP)

*File Transfer Protocol* (FTP)

*Street View House Number* (SVHN)

*Modified National Institute of Standards and Technology* (MNIST)

*U.S. Postal Service* (USPS)

*Normalized Mean Absolute Error* (NMAE)

*Peak Signal-to-Noise Ratio* (PSNR)

*Signal-to-Noise Ratio* (SNR)

*Structural Similarity Index Measurement* (SSIM)

*Visual Information Fidelity* (VIF)

*Naturalness Image Quality Evaluator* (NIQE)

*Variational AutoEncoders* (VAEs)

*Cycle-Consistency* (CC)

*Amazon Mechanical Turk* (AMT)

*Intersection-Over-Union* (IOU)

*Mean-Squared Error* (MSE)

*Deep Learning OPerations* (DLOps)

*Central Processing Units* (CPUs)

*Graphics Processing Units* (GPUs)

*Random Access Memory* (RAM)

*Shared Wireless Access Protocol* (SWAP)

### Résumé

L'intelligence artificielle a progressé au cours des dernières années, montrant des capacités accrues dans le traitement des données pour diverses applications de la vision par ordinateur, notamment la classification, la segmentation, la détection, etc. Notamment, le réseau neuronal génératif a prouvé son efficacité dans la capture de caractéristiques plus profondes au cours du processus de synthèse d'images. Sur la base de l'état de l'art, les réseaux de générations sont compétitifs par rapport aux algorithmes traditionnels d'apprentissage automatique, dans l'apprentissage de structures similaires entre les images d'entrée. Par conséquent, ce projet de recherche développe des architectures améliorées à une entrée unique et multiple et à sortie unique dans le but de synthétiser l'imagerie médicale. Les architectures développées de U-Net et de *generative adversarial networks* sont importantes, car elles permettent de gagner beaucoup de temps sur le diagnostic des tumeurs et elles permettraient d'éviter l'injection d'une substance supplémentaire, par exemple du gadolinium, dans le cerveau lors de l'extraction de la modalité T1c. Pour prouver l'efficacité des architectures, les modèles de *deep learning* sont entraînés et expérimentés en utilisant des techniques d'hyper-optimisation pour tester sur le jeu de données BRATS2018. Ensuite, les architectures ont obtenu des résultats compétitifs en mettant en évidence les performances de l'état de l'art.

**Mots-clés :** Intelligence artificielle, réseaux neuronaux génératifs, apprentissage profond, synthèse médicale

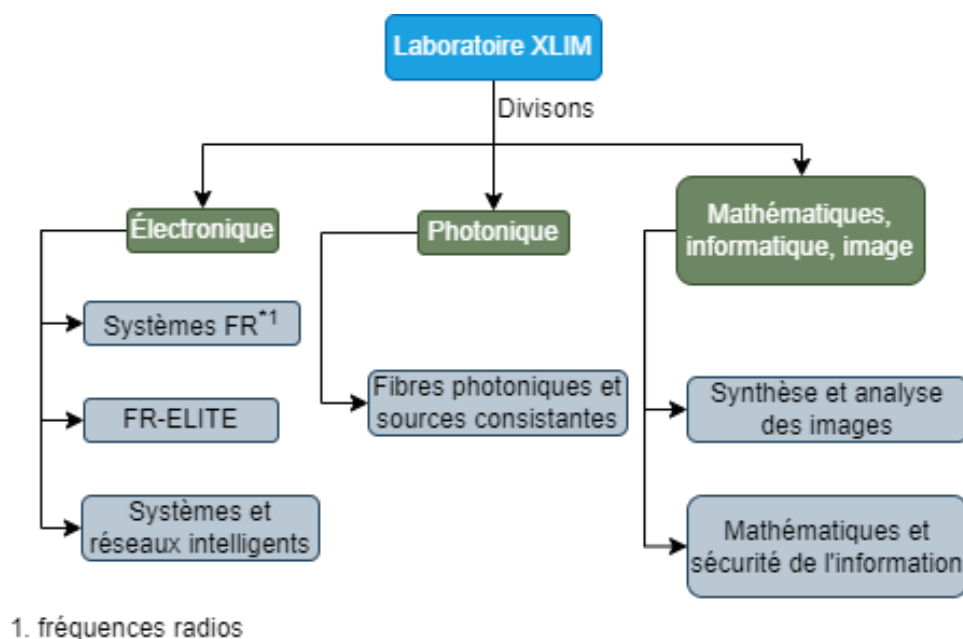
## I. Laboratoires et présentations de stage

Dans cette section, une brève introduction est représentée sur les laboratoires XLIM et imagerie métabolique multi-noyaux multi-organes (I3M) ainsi que leurs différentes spécialités de domaine. Ensuite, une présentation générale du poste de stage sera abordée avec ses objectifs souhaités et les missions requises, et surtout la gestion du projet.

### I.1 Laboratoire XLIM

XLIM, numéroté 7252, est un des centres de recherche français, connu sous le nom de centre national de la recherche scientifique (CNRS) [7]. Le laboratoire est largement centré sur l'électronique, l'optique, la photonique, les mathématiques, l'informatique, le traitement d'images, les télécommunications, la sécurité des réseaux, la bio-ingénierie et l'énergie. Une représentation des divisions de XLIM est présentée ci-dessous dans la figure (1).

**Figure 1-** Représentation du laboratoire XLIM, qui se compose de 3 pôles et de 6 axes scientifiques



XLIM est un institut de recherche pluridisciplinaire, implanté sur plusieurs sites géographiques, à Limoges sur les sites de la Faculté des Sciences et Techniques, de l'ENSIL, d'Ester-Technopole, sur le campus universitaire de Brive ; et sur le site de la Technopole du Futuroscope à Poitiers. Il rassemble plus de 440 enseignants-chercheurs, chercheurs CNRS, ingénieurs, techniciens administratifs, doctorants et post-doctorants.

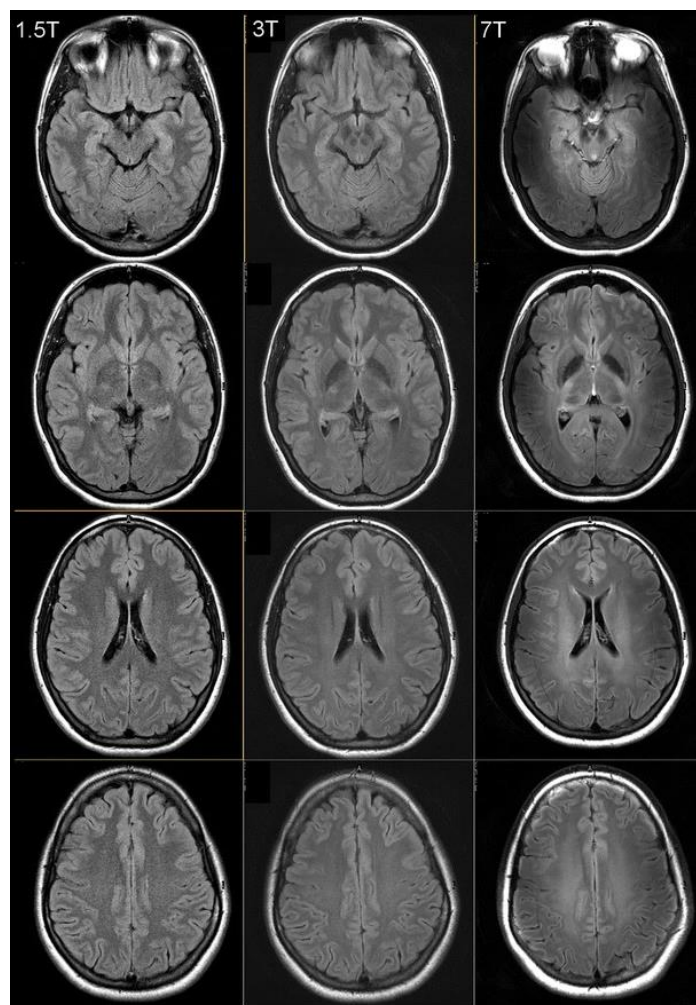
La recherche principale d'XLIM s'appuie principalement sur deux plateformes. D'une part, une plateforme permet aux chercheurs d'accéder à des équipements technologiques pour la création de structures optiques. Par exemple, un grand nombre de lasers et d'instruments sont disponibles pour la caractérisation expérimentale de dispositifs électroniques, optiques, électromagnétiques et radiants. D'autre part, le laboratoire s'engage dans des activités de modélisation et de simulation car il est équipé d'une forge logicielle, qui est à la disposition de ses membres et de leurs collaborateurs externes. Pour cela, il propose une simulation multi-échelle de systèmes complexes basée sur des modèles d'interface entre la physique, les mathématiques et l'informatique.

## I.2 Laboratoire I3M

I3M a été fondé en février 2019, une collaboration entre le CNRS, SIEMENS HEALTHINNERS, le centre hospitalier universitaire (CHU) de Poitiers et l'Université de Poitiers [24]. C'est un centre de recherche transdisciplinaire distinctif basé sur l'imagerie métabolique des organes en France et même en Europe. Le laboratoire utilise le traitement automatique et l'analyse multimodale avec l'aide de l'intelligence artificielle (IA) pour l'objectif d'images résonance magnétique nucléaire (IRM) multivariées ainsi que l'aide au suivi diagnostique et thérapeutique dans les maladies du cerveau, du cœur et des reins. Par ailleurs, l'I3M utilise une IRM à ultra-haut champ de 7Tesla, considérés comme la 23e du monde entier [25].

L'IRM 7Tesla permet d'accéder à une imagerie de très haute résolution des molécules et du métabolisme, en plus de l'anatomie et de la fonction des organes. Elle permet de mesurer la fonction et la structure des organes d'une manière inégalée. Par exemple, la figure (2) représente une comparaison d'images *fluid-attenuated inversion recovery* (FLAIR) entre 1.5T, 3T et 7T d'un même sujet. Par conséquent, les images FLAIR, capturées par le 7T, ont un *signal-to-noise ratio* (SNR) plus élevé et un contraste bien meilleur par rapport aux autres générations d'IRM Tesla.

**Figure 2-** Images FLAIR représentatives d'un sujet unique à trois intensités de champ : 1.5T, 3T et 7T, extraites de [23]



### I.3 Présentation de stage

Pendant mon stage, j'ai été un ingénieur de recherche, travaillant sur le développement et l'expérimentation d'architectures *deep learning* (DL) pour résoudre le problème de la génération de différentes séquences IRM, principalement sur la génération de fausses modalités sur *brain tumor segmentation challenge* 2018 (BRATS2018). J'ai acquis de nombreuses compétences, qui seront ma prochaine source de connaissances et de développement de stratégies à l'avenir. Certaines des compétences que j'ai acquises sont résumées dans ce qui suit :

1. Gestion du réseau et programmation d'un ordinateur à distance en utilisant *secure shell* (SSH) et *ssh file transfer protocol* (SFTP).
2. Prototypage d'un algorithme pour l'amélioration de la recherche.
3. Planification et autogestion de projets.

Le stage se déroule dans deux laboratoires différents et à des intervalles de temps différents :

- XLIM : présence du 28 mars au 22 juillet et du 18 août au 1er septembre.
- I3M : présence du 25 juillet au 17 août.

#### I.3.1 Cahier des charges

Les principales spécifications des stages sont les suivantes :

1. Développement d'une solution pour la construction de séquences IRM à l'aide d'algorithmes d'IA.
2. Réalisation d'un prototype en langage Python.

Dans la section suivante, je vais décrire et donner des détails supplémentaires sur les missions durant toute la période de stage.

#### I.3.2 Déroulement et Méthodologie

Dans cette section, une brève description des actions et des procédures est présentée.

##### 1.3.2.1 Présentation initial

Pour commencer, j'ai été chargé de découvrir le jeu de données BRATS2018 dans un contexte médical, en dehors de sa complexité structurelle et représentationnelle. Ensuite, j'ai l'analysé en effectuant différentes manipulations à l'aide du langage de programmation Python. Par exemple, j'ai découvert les données plus en détail avec des graphiques de représentation en utilisant la visualisation Matplotlib<sup>1</sup>.

##### 1.3.2.2 Algorithmes

En outre, j'ai développé des architectures de DL comme *single-input single-output* (SISO) et *multi-input single-output* (MISO), en utilisant des *generative adversarial networks* (GANs) et des algorithmes U-Net, pour prédire certaines modalités à partir d'une ou plusieurs entrées. Par conséquent, j'ai identifié les solutions et les algorithmes les plus récents pour améliorer les résultats, y compris les méthodes de prétraitement et les structures d'algorithme. J'ai expérimenté différentes méthodes pour affiner l'algorithmes GANs, y compris le pipeline et les tests d'échantillons.

##### 1.3.2.3 Réunion

Lors des réunions en face-à-face et sur le web, je devais présenter les résultats obtenus à chaque étape ; ainsi, à travers ces présentations, ma langue française s'améliorait régulièrement voilà

---

<sup>1</sup> Matplotlib est une bibliothèque de graphisme pour le langage de programmation Python et son extension de mathématiques numériques NumPy.

que je rédigeais des rapports hebdomadaires et mensuels et des courriels à mon superviseur ainsi qu'à mon superviseur pédagogique.

### 1.3.2.3 Compétence acquis

En plus des tâches techniques, mes qualifications, dans le cadre du programme de stage en imagerie médicale d'XLIM et d'I3M, se sont massivement développées en matière de communication, d'organisation, d'autonomie de recherche, de vision analytique, de curiosité et de créativité. J'ai appris à présenter des idées avec de bonnes capacités de communication, tant à l'oral qu'à l'écrit. Néanmoins, l'écriture n'est pas exempte de limites bien que j'aie fait de mon mieux pour garder les écrits sans erreurs.

Dans les bases quotidiennes, j'apprenais de nouvelles choses qui avaient façonné mon esprit dans la science des données et l'IA. Je pense avoir prouvé que j'étais un membre essentiel en apportant des idées révolutionnaires et des solutions innovantes pour améliorer la synthèse d'images.

En ce qui concerne l'importance, j'ai constaté que l'organisation est l'un des éléments essentiels de la gestion du temps et du traitement des différentes tâches à chaque étape de l'avancement. J'ai remarqué que la science des données et la DL requièrent un esprit dynamique et analytique pour visualiser des solutions adaptées tout en couvrant un large éventail de littérature dans les travaux connexes en imagerie médicale. En pratique, la revue de la littérature a élargi mes horizons pour relever certains défis et tirer des leçons de ce que d'autres chercheurs ont vécu.

En somme, j'ai vraiment obtenu de grandes connaissances sur les clés et les capacités potentielles de la synthèse d'images grâce à l'analyse des données et au développement algorithmique. Je crois qu'elles se manifestent rarement dans les laboratoires de recherche français, surtout après la période difficile de la pandémie de *coronavirus disease 19* (Covid-19).

### 1.3.2.4 Défis

Cependant, j'ai été confronté à de nombreux problèmes pour mettre en place l'environnement d'entraînement des algorithmes, car je me déplace d'un laboratoire à l'autre et j'ai à chaque fois un équipement différent à utiliser, sans droit d'administration. Je pourrais dire que tout dépend d'une hiérarchie donc si je voulais quelque chose, comme installer un programme, je dois l'attendre car je n'ai aucune autorité sur l'ordinateur. Ainsi, pour la première phase, j'ai découvert que l'utilisation de mon ordinateur personnel permettrait de gagner beaucoup de temps sur le développement des différents prototypes d'architectures GANs.

Néanmoins, j'ai été un peu limité dans mon ordinateur personnel, qui se plantait lors de l'entraînement avec une énorme quantité de données. La spécification de mon ordinateur, détaillé dans le tableau (1) suivant :

**Tableau 1-** Spécifications de l'ordinateur personnelle

<i>Central processing units</i> <b>(CPUs)</b>	Processeur Intel Core i7-10750H (Hexa-Core 2.6 GHz/5 GHz Turbo - 12 Threads - Cache 12M)
<i>Random access memory</i> <b>(RAM)</b>	16 Go de RAM DDR4 à 2666 MHz (2 × 8 Go)
<i>Graphics processing units</i> <b>(GPUs)</b>	Nvidia GeForce RTX 2060 avec 6 Go de mémoire dédiée



Plus tard, j'ai eu accès à un serveur à distance équipé de Quadro RTX, sur lequel j'ai effectué toutes les expérimentations. Enfin, j'ai effectué les dernières expérimentations sur un autre serveur à distance équipé de Nvidia Tesla dans XLIM et I3M.

## I.4 Gestion de projet

Dans cette section, je vais présenter les logiciels que j'ai utilisés pendant le stage, en commençant par les systèmes d'exploitation, les programmes et les plateformes de codage.

### I.4.1 Les systèmes d'exploitation

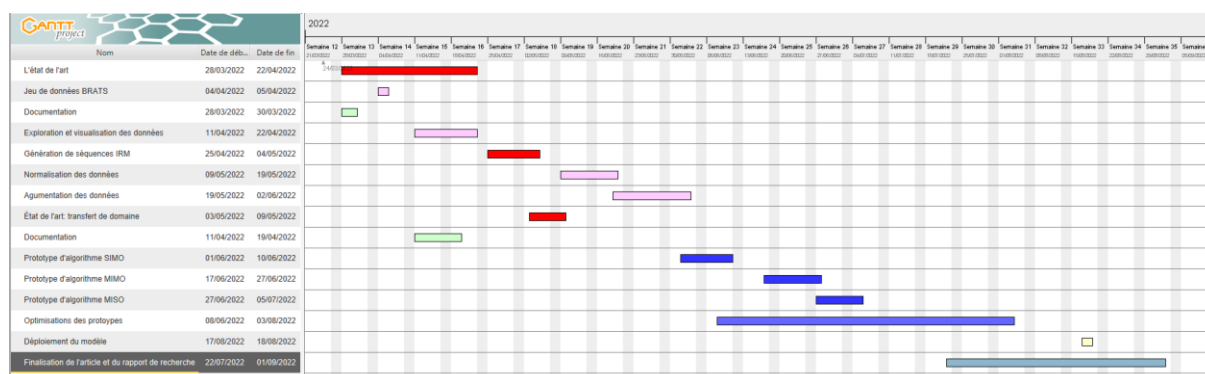
**Linux** et **Windows** systèmes d'exploitation sont des logiciels qui gèrent le matériel informatique, les ressources logicielles et fournissent des services communs aux programmes. Ainsi, il est considéré comme le logiciel fondamental qui pourrait gérer les autres logiciels que nous utilisons, par exemple, anaconda.

### I.4.2 Programmes de logiciels

**Service SourceSup**, exploité par Renater, est une plateforme web de gestion pour les organismes d'enseignement supérieur et de recherche français. Chaque membre du réseau peut créer un projet sur la plateforme en permettant la collaboration supplémentaire de personnes extérieures au projet créé.

**GanttProject** décompose le travail, construit un diagramme de Gantt, affecte les ressources et calcule les coûts du projet. Nous l'avons utilisé pour débriefer les tâches du projet pour les semaines en cours à travers les tâches réalisées et celles encore en cours.

**Figure 3-** Diagramme de GANT - gestion du temps des tâches



Comme dans la figure (3) ci-dessus, nous pouvons voir les tâches en cours, terminées et continues sur lesquelles j'ai travaillé. Par exemple, du 28/03/2022 au 22/04/2022, j'ai couvert l'état de l'art concernant les GANs et la synthèse médicale.

Technique d'évaluation et d'examen de programmes (**PERT**), un outil de planification, d'organisation et d'ordonnancement des tâches au sein d'un projet, représente le calendrier du projet et catégorise les tâches individuelles. Les diagrammes PERT sont semblables aux diagrammes de Gantt, mais avec une structure différente. Dans le tableau (2) et la figure (4) suivants, nous présentons les principales tâches et leurs représentations en fonction de leur priorité et de leur durée au plus tôt et au plus tard.

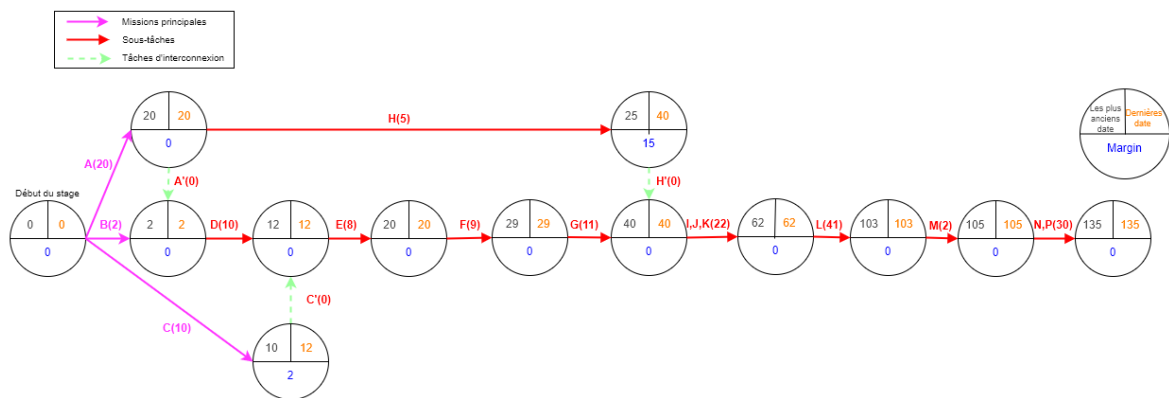
**Tableau 2-** Diagramme de PERT - gestion des tâches

Tâche	Nature	Durée	Antériorité
A	État de l'art	20	-
B	Jeu de données BRATS	2	-



C	Documentation	10	-
D	Visualisation et exploration des données	10	B
E	Génération de séquences IRM	8	D, C
F	Normalisation des données	9	A, B, C
G	Augmentation des données	11	F
H	État de l'art : transfert de domaine	5	A
I	Prototype SIMO	8	H, G
J	Prototype MIMO	7	H, G
K	Prototype MISO	7	H, G
L	Optimisations des prototypes	41	I, J, K
M	Déploiement du modèle	2	I, J, K, L
N	Rédaction d'un article de recherche	15	I, J, K
P	Rapport final	15	I, J, K
<b>Total</b>	15	24 semaines ≅ 1190 ≅ 170jours	

**Figure 4-** Graphique PERT classique - gestion du temps des tâches, représenté en tableau (2)



L'élément-clé que nous pouvons remarquer dans le graphique PERT, c'est que nous pouvons estimer la date la plus proche de l'achèvement d'une tâche et l'état de liberté, comme dans la tâche H. Par conséquent, la tâche pourrait être retardée de 15 jours de plus parce que les autres tâches, par exemple, D, E ou F, ne dépendent pas de H.

**draw.io** est un logiciel de diagramme en ligne gratuit. Il est utilisé pour écrire et simplifier des algorithmes, du code et des informations dans un schéma visuel facile à comprendre.

**Webex**, de Cisco, est un logiciel de vidéoconférence, de réunions en ligne, de partage d'écran et de webinaires. Il a été utilisé pour rapporter les résultats, les problèmes et les plaintes au superviseur ainsi que pour planifier les réunions hebdomadaires.

**ENT Zimbra** est un espace collaboratif pour écrire aux collaborateurs de stage, partager des messages, et créer des rendez-vous.

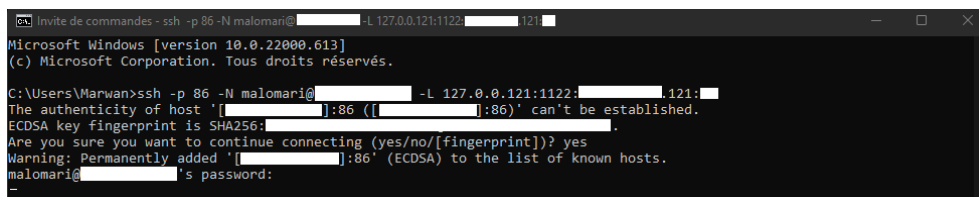
**Microsoft Office** est une application de traitement de texte qui permet de configurer les attributs d'un document, tels que la mise en page, les styles de contenu, et d'ajouter leur contenu de diverses manières et dans divers formats pour produire des documents. Il est utilisé pour rédiger les rapports hebdomadaires ainsi que la version finale du rapport.

**Microsoft PowerPoint** est un programme de présentation, qui utilise des diapositives pour transmettre des informations riches soutenues par différents multimédias.

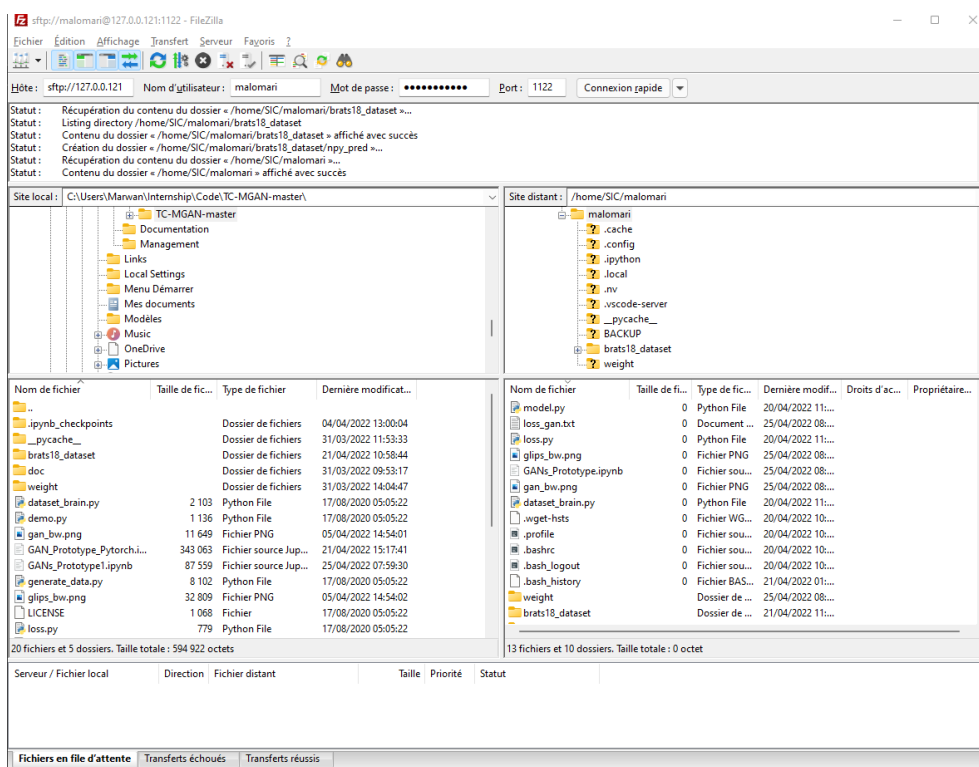
**Zotero<sup>2</sup>** aide à collecter, organiser, citer et partager différentes sources de recherche en ligne.

**FileZilla** est une application *file transfer protocol* (FTP) gratuite et open-source, multiplateforme, composée de FileZilla Client et FileZilla Serveur. Nous avons utilisé le client FileZilla, qui prend en charge les connexions aux serveurs FTP et SFTP. Un exemple de connexion réseau effectuée au serveur en question est représenté dans les deux figures suivantes (5) et (6). La première figure (5) établit la connexion avec la machine distante par son *internet protocol* (IP) à l'adresse 127.0.0.121, port 1122. Dans FileZilla, nous entrons la coordination qui est bien établie dans le terminal pour interroger le serveur en question, comme il est bien précisé dans la figure (6).

**Figure 5-** Établissement d'une connexion SSH avec le serveur en question pour la manipulation SFTP



**Figure 6-** Connexion SFTP à l'aide de FileZilla



### I.4.3 Plateformes de codage

**Jupyter Notebook** fournit des services de calcul interactifs en utilisant Python. Ainsi, en codant, Jupyter visualisera les résultats dans une même fenêtre de script. Ainsi, il facilite le débogage de l'algorithme ou du code de l'application.

**Anaconda** est une distribution du langage de programmation Python pour le calcul scientifique, notamment la science des données, les applications machine learning (ML), le traitement des

<sup>2</sup> [www.zotero.org](http://www.zotero.org)

données à grande échelle, l'analyse prédictive, etc. Elle est utilisée dans le but de simplifier la gestion et le déploiement des paquets.

**Microsoft Visual Studio Code**, appelé VS code, est un éditeur de code. Nous l'avons utilisé pour faciliter les connexions SSH et l'intégration du code dans le même espace de travail. En outre, il est plus facile d'utiliser et de manipuler les fichiers du carnet Jupyter sans qu'un serveur local ne s'ouvre en arrière-plan, comme le montre la figure (5) ci-dessus.

## 1. Introduction

Le traitement des images fait appel à des méthodologies différentes et multiples, depuis les caractéristiques forgées à la main jusqu'à la DL, pour les objectifs de détection, de classification et de segmentation. En tant que développement conséquent de l'IA, la DL utilise des unités de traitement plus profondes appelées neurones pour apprendre des caractéristiques et des représentations hiérarchiques des données [8]. Dans ce cas, les méthodes de DL surpassent les prédictions humaines, car elles connaissent des progrès et des évolutions impressionnants dans différents domaines scientifiques, par exemple la vision par ordinateur [9].

Dans le contexte de l'imagerie médicale, la classification et la régénération des images sont généralisées pour distinguer les différentes classes et catégories dans l'expérimentation du réglage des paramètres dans les tâches auto-génératives [13]. Les réseaux génératifs, notamment le transfert de style de collection, la transfiguration d'objets, le transfert de saison, l'amélioration de photos, etc. [15][16].

En ce qui concerne l'objectif de ce stage, nous avons donc adopté le modèle GANs pour générer différents attributs à partir d'un seul et de plusieurs attributs en entrée. Les architectures proposées permettent de gagner du temps dans la prédiction des attributs d'imagerie médicale et d'éviter l'injection de certaines substances aux patients. Par exemple, dans le cas du T1c, le patient est injecté avec du gadolinium pour extraire une modalité avec un meilleur contraste. Les données de BRATS, utilisées dans cette étude, sont largement utilisées pour améliorer le diagnostic et la segmentation des lésions cérébrales [6]. En pratique, BRATS2018 comprend des images de fichiers de patients qui contiennent toutes quatre modalités (attributs) et une carte de segmentation (vérité terrain) : T1-weighted (T1), T1 and contrast-enhance (T1c), T2-weighted (T2), et FLAIR.

Les travaux de ce projet sont spécifiés en 5 sections. La première section présente déjà le sujet à l'étude. Plus tard, dans la deuxième section, nous découvrirons les travaux connexes qui sont bien fournis sur la synthèse d'images. La troisième section détaille le contexte DL dans le cadre de la synthèse médicale ainsi que le jeu de données BRATS2018. La quatrième section précise l'approche de recherche en discutant les étapes de prétraitement, les algorithmes et les mesures d'évaluation. Dans la cinquième section, nous présentons les résultats des expériences avec une analyse spécifique. La sixième section conclut l'idée générale du projet de recherche en soulignant les défis à relever et les améliorations à apporter à l'avenir. Enfin, la liste des références est numérotée tout au long de l'article pour faciliter la navigation.

## 2. Travaux relatifs

Dans cette section, nous présentons les principaux articles de recherche qui ont été expérimentés sur des jeux de données open-source provenant de différents défis, par exemple CelebA<sup>3</sup>, RaFD<sup>4</sup> ou BRATS [4]. L'objectif de la couverture des travaux connexes sur l'utilisation de l'architecture des GANs est de mieux comprendre la littérature et les techniques employées dans les approches des chercheurs par rapport à notre étude de projet. À la fin de cette section, nous fournissons un résumé des études détaillé dans le tableau (20) présenté en **8. Annexe**.

---

<sup>3</sup> Voir Annexe pour plus de détaillé

<sup>4</sup> Voir Annexe pour plus de détaillé

Les chercheurs de [3] ont adopté la stratégie StarGAN [2] pour cartographier les modalités d'imagerie par IRM à travers 12 modèles single-input single-output (SISO) (une modalité à la fois), et un GAN unifié tous ensemble. Leur méthode a été évaluée sur BRATS2015 qui a prétraité en réduction la taille originale de patch à  $72 \times 72 \times 72$ . Sur une estimation des régions fixée à  $48 \times 48 \times 48$ , T1 comme modalité d'entrée, les *peak signal-to-noise ratio* (PSNRs) pour les T1c, T2, FLAIR générés sont  $32.353 \pm 2.525$  dB,  $30.016 \pm 2.577$  dB, et  $29.091 \pm 2.795$  dB, respectivement.

D'autre part, StarGAN [2], fonctionne bien sur les tâches de transfert de caractéristiques faciales et de synthèse d'expressions faciales, a été testée sur deux jeux de données : CelebA et RaFD. Sur CelebA, StarGAN a obtenu 66.2%, 39.1%, 70.6%, 47.4%, 61.5%, 49.8%, 52.2% pour la couleur des Cheveux, le Sexe, l'Âge, C+S, C+A, S+A, et C+S+A, respectivement. Le modèle a correctement appris le rôle prévu d'un vecteur de masque dans les traductions d'image à image lorsque toutes les étiquettes proviennent de plusieurs ensembles de données. En conclusion, StarGAN a généré des images de meilleure qualité visuelle que les méthodes existantes.

Pix2pix [13], un générateur d'U-Net et un discriminateur de CGANs, combine une perte contradictoire avec une perte L1 pour capturer les basses fréquences. Progressivement, leur modèle sous-échantillonne les images d'entrée sur différentes tâches et des ensembles de données, y compris des Cityscapes, des façades GMP, des photos Google Maps, des photos en couleur, etcétera. En conclusion, la perte de L1, le CGAN et L1+CGAN ont atteint sur les Cityscapes 86%, 76% et 83% d'*accuracy* (ACC) par pixel, et 42%, 28% et 36% d'ACC par classe, respectivement.

UNIT [14] combine deux générateurs partagent des poids pour apprendre la distribution des images dans un domaine croisé entre les domaines source et cible, de manière interchangeable. Ils ont utilisé un ensemble de données cartographiques dans deux domaines (images satellites et cartes). L'algorithme a obtenu des précisions de classification : *the street view house numbers* (SVHN)  $\rightarrow$ <sup>5</sup> *modified national institute of standards and technology* (MNIST) 0,9053, MNIST  $\rightarrow$  *U.S. postal service* (USPS) 0,9597, USPS  $\rightarrow$  MNIST 0,9358.

DiscoGAN [15] et CycleGAN [16] sauvegardent les attributs d'information entre les images d'entrée et les images traduites en utilisant une perte *cycle-consistency* (CC). Cependant, ils souffrent tous d'une généralisation limitée, car il s'agit de modèles SISO.

D'une part, DiscoGAN, *generative adversarial network* (GAN) couplés en forçant les images générées par une perte de reconstruction réduits les images en entrée à taille  $64 \times 64 \times 3$ . Le modèle proposé de rotations entre un GAN standard et un GAN à perte reconstructive, les images générées ne varient pas autant que les images d'entrée. Lors d'une expérience sur le jeu de données de voitures, les deux modèles ont souffert d'un effondrement soudain. En outre, ils ont expérimenté la conversion de visage par attributs de visage sur les jeux de données CELEBA, facescrub, où une seule caractéristique, comme le sexe ou la couleur des cheveux, varie entre deux domaines et des domaines partagés. Les résultats traduits ont non seulement des couleurs et des motifs similaires, mais ils ont également un niveau de formalité de mode similaire à celui de l'article de mode d'entrée.

---

<sup>5</sup> Synthèse d'un ensemble de données d'image à un autre

D'autre part, CycleGAN, utilise deux pertes CC qui capturent la translation d'image d'un domaine à l'autre et vice-versa. L'architecture échouée dans certaines tâches de traduction impliquant des changements de couleur, de géométrie et de texture, par exemple, un chat en chien ou à cheval, et un cavalier sur un cheval en zèbre. De plus, une faiblesse sémantique a été repérée dans les tâches d'étiquetage de photos, et l'ambiguïté de la traduction a donc formé des limites importantes. En conclusion, le modèle a obtenu des étiquettes réelles de  $26,8\% \pm 2,8\%$  pour la carte vers la photo, et de  $23,2\% \pm 3,4\%$  pour la photo vers la carte. En outre, il a obtenu 58% de précision par pixel, 22% de précision par classe et 16% de reconnaissance de classe moyenne pour les étiquettes de photo sur le jeu de données Cityscapes.

En conclusion, la littérature nous a permis de constater que la recherche se concentre sur la méthode SIMO, comme nous l'avons vu dans les articles suivants [2][3][15][16]. De là, nous avons consacré notre travail de recherche à l'introduction de l'architecture MISO, inspirée de [2][3]. Elle tire profit des performances de pointe, discutées dans la littérature, pour prédire un seul attribut à partir de plusieurs entrées. L'objectif principal de la recherche est de prédire le T1c à partir de plusieurs entrées afin de réduire le temps d'examen IRM et d'éviter l'injection de gadolinium.

### 3. Background

Dans cette section, nous discuterons du DL dans le contexte du diagnostic médical, en mettant en évidence les architectures GANs ainsi qu'une description détaillée du jeu de données BRATS2018.

#### 3.1 L'apprentissage profond dans le diagnostic médical

L'IA est prometteuse dans le domaine du diagnostic médical, car elle permet d'éviter les erreurs potentielles et les erreurs médicales préjudiciables. La DL, basée sur les *artificial neural networks* (ANNs) avec apprentissage de représentation par renforcement, méthodes supervisées, semi-supervisées et non supervisées [10], améliore le diagnostic des symptômes, qui sont difficiles à repérer même par les meilleurs experts. Faisant partie de la grande famille des méthodes ML, les architectures DL telles que les *Deep Neural Networks* (DNNs), *Recurrent Neural Networks* (RNNs) et *convolutional neural networks* (CNNs) ont été appliquées à divers domaines, notamment la vision par ordinateur, le *traitement automatique des langues* (TAL), la *machine translation* (MT) et l'analyse d'images médicales, où elles ont produit des résultats compétitifs qui surpassent les performances humaines [8].

En général, les ANNs ont été inspirés par le traitement de l'information et les nœuds de communication distribués dans les systèmes biologiques. Les ANNs se distinguent des cerveaux biologiques par leur tendance à être statiques et symboliques, alors que le cerveau biologique tend à être dynamique et analogique [26]. Cependant, le noyau principal des ANNs fait référence à l'utilisation de couches multiples pour extraire progressivement des caractéristiques de plus haut niveau d'une entrée brute. Par exemple, dans le traitement des images, les couches inférieures peuvent identifier les bords, tandis que les couches supérieures peuvent identifier les concepts pertinents pour les humains, tels que les chiffres, les lettres ou les visages.

De nos jours, la DL reste la méthodologie ML la plus prometteuse et la plus utilisée pour la radiologie et la détection des maladies en général. Ce n'est pas une surprise, car l'imagerie diagnostique prévaut dans le diagnostic clinique et la reconnaissance d'images. En 2016,



Geoffrey Hinton, un informaticien et chercheur notable, a prédit que les radiologues et les spécialistes qui diagnostiquent les maladies à partir de l'imagerie médicale comme les rayons X, les tomodensitogrammes et les IRM ; ils allaient bientôt perdre leur emploi. "Les gens devraient arrêter de former des radiologues dès maintenant", va-t-il annoncé, "Il est évident que d'ici cinq ans, l'apprentissage profond va faire mieux que les humains" [8].

Les progrès de la DL la rendent très utile, mais pas assez fiable pour que les machines puissent remplacer les experts humains, comme l'a indiqué Hinton. Cependant, la DL est toujours utilisée pour aider les médecins à présélectionner et à classer les cas par ordre de priorité, mais pas en tant qu'outil principal pour diagnostiquer les patients. A partir de là, il existe diverses utilisations de DL en radiologie et dans d'autres pratiques de diagnostic :

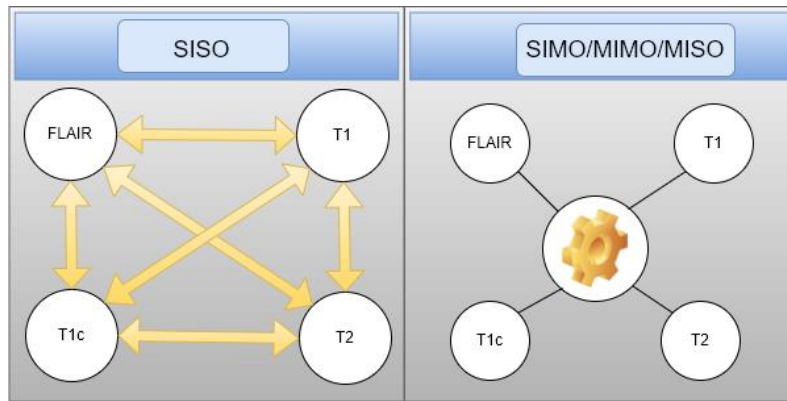
- |   |   |
|---|---|
| • Détection des anomalies neurologiques.                      | la performance dans les divers pratiques de diagnostic.                 |
| • Dépistage des cancers courants.                             | • Détection des tumeurs cérébrales à haut débit.                        |
| • Identification des infections des reins et du foie.         | • Analyse de l'imagerie dentaire.                                       |
| • Segmentation des tumeurs cérébrales.                        | • Détection des fractures osseuses et des lésions musculosquelettiques. |
| • Génération et synthèses des nouvelles images pour améliorer |   |

Particulièrement, la synthèse d'images médicales, qui fait partie des pratiques DL, est une alternative aux séquences d'impulsions multiples pour l'acquisition d'IRM à contrats multiples [3]. L'IRM est un processus non standardisé dans différentes institutions, la synthèse d'image cross-modale a été proposée pour relever ce défi en fournissant les modalités manquantes. Elle est utilisée dans la pratique clinique en raison de sa capacité à fournir des informations utiles, par exemple les quatre contrats :

1. T1 distingue les matières blanches et grises.
2. T1c évalue le changement de forme de la tumeur avec une démarcation améliorée autour de la tumeur.
3. T2 montre la présence de liquide dans le tissu cortical.
4. FLAIR montre les contours de la lésion.

Dans la recherche et les pratiques médicales, différentes architectures peuvent être envisagées lors de l'application de la DL. Elles sont résumées dans la liste suivante et la figure (7) :

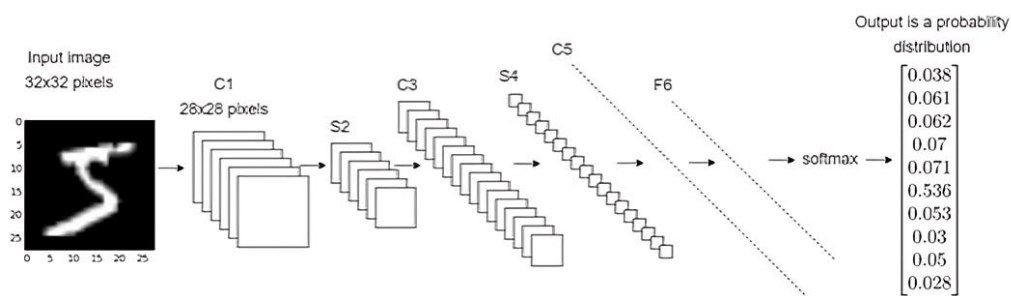
1. *Single-Input Multi-Output (SIMO)* : une image cible générée à partir d'une image source donnée.
2. *Multi-Input Single-Output (MISO)* : surmonter les limites de SISO lorsque les images source et cible sont faiblement corrélées grâce à l'apprentissage de représentations latentes partagées.
3. *Multi-Input Multi-Output (MIMO)* : synthétiser une ou plusieurs modalités à partir d'une entrée de modalités d'IRM.
4. *Single-Input Single-Output (SISO)* : où une seule modalité est disponible en entrée, mais où plusieurs contrastes sont nécessaires en sortie.

**Figure 7-** Méthodes SISO, SIMO, MISO et MIMO dans le traitement des modalités IRM

En observation, l'IRM de différentes modalités pourrait fournir des informations complémentaires pour le diagnostic médical, mais il est également difficile et très coûteux d'accéder à toutes les modalités. De nombreuses méthodes se concentrent sur une seule modalité pour en faire une synthèse, ce qui limite considérablement la généralisation des résultats à d'autres modalités. Par conséquent, pour chacune des deux modalités, nous devons développer un modèle distinct afin de pouvoir les mettre en correspondance. Pour résoudre ce problème difficile, nous proposons un GAN à modalités multiples pour synthétiser trois modalités IRM (FLAIR, T2 et T1c) à partir d'une modalité IRM T1. Un autre défi consiste à prédire FLAIR ou T1c, en évitant d'injecter une substance (par exemple du gadolinium) dans le cerveau du patient, à partir de différentes entrées : T1 et T2 ou T1, T2, FLAIR. Les architectures proposées seront détaillées ultérieurement dans **Algorithm**.

### 3.2 CNNs

Comme les architectures U-Net et GAN sont toutes deux basées sur des CNNs, l'architecture est détaillée dans la présente section en prenant comme exemple l'architecture LeNet-5, publiée par Yann LeCun en 1998 [22]. Elle est bien présentée dans la figure (8) suivante :

**Figure 8-** LeNet-5, un exemple d'architecture CNNs, extrait de [21]

Les CNNs ou ConvNets sont couramment utilisés dans les tâches de classification, de détection et de régression de la vision par ordinateur. En général, les CNNs prennent une entrée sous la forme d'un tableau ou d'une matrice, par exemple une image. Dans ce cas, il s'agirait d'un tableau de pixels et cela dépendrait de sa résolution. L'entrée proposée se présente sous la forme d'une hauteur ( $h$ )  $\times$  largeur ( $l$ )  $\times$  dimension ( $d$ ). La dimension pourrait se référer à rouge, vert, et bleue (RVB) par l'indicateur 3 et 1 pour les images en niveaux de gris. Pour procéder à la formation et au test du modèle CNNs, chaque image d'entrée passe par une série de couches de convolution avec des filtres (noyaux) spécifiques, des fonctions de mise en commun, de *fully-*



*connected layers* (FCL) et de perte pour obtenir finalement une classification probabiliste. Pour une explication claire des CNNs, LeNet-5 sera utilisé pour démontrer les différentes couches de CNNs ainsi que les architectures U-Net et GAN.

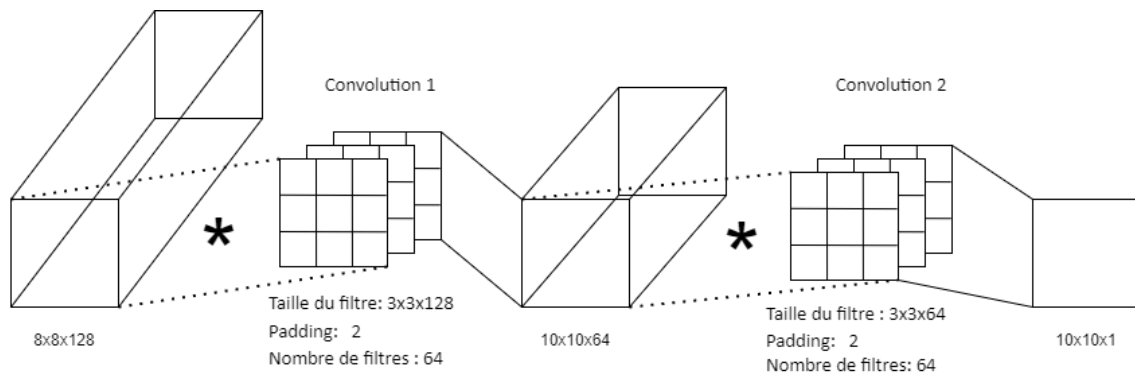
- Convolution (C1, C3, and C5)
  - a. Up-Convolution
  - b. Convolution transposée
- Couche de Pooling (S2 and S4)
- FCL (F6)
- Fonctions d'activation

### 3.2.1 Convolution (C1, C3, and C5)

La première couche qui extrait les caractéristiques de l'image d'entrée en préservant la relation entre les pixels en apprenant les caractéristiques de l'image à l'aide de petits carrés de données. Elle utilise une opération mathématique de deux entrées comme une matrice d'image et un noyau. De tels filtres, par exemple, la détection des bords, le flou et la netteté. Le filtre utilise le stride, qui est le nombre de pixels déplacés sur les données d'entrée. Ainsi, si le stride est fixé à 4, le filtre déplacera 4 pixels à la fois et ainsi de suite.

- Par exemple, la convolution pour augmenter la largeur et la hauteur et réduire la profondeur de l'entrée, par exemple l'image, appliquerait deux couches convolutionnelles consécutives, comme le montre bien la figure (9) :
  - a. Première convolution :  $3 \times 3 \times 128$  filtre, 2 padding, 64 nombre de filtres. Résultats :  $10 \times 10 \times 64$ .
  - b. Deuxième convolution :  $3 \times 3 \times 64$  filtre, 2 padding, 64 nombre de filtres. Résultats :  $10 \times 10 \times 1$ .

**Figure 9-** Une déconstruction de l'opération convolutive dans les CNNs

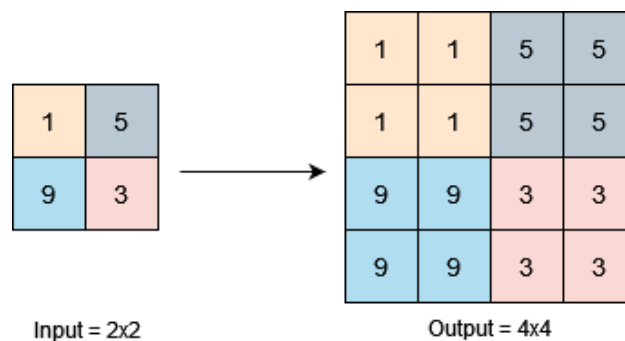


#### 3.2.1.1 Convolution transposée

D'autre part, la convolution transposée, connue sous le nom de déconvolution, est l'opération inverse de la convolution. Elle permet de compresser la dimension des caractéristiques et d'agrandir leur taille.

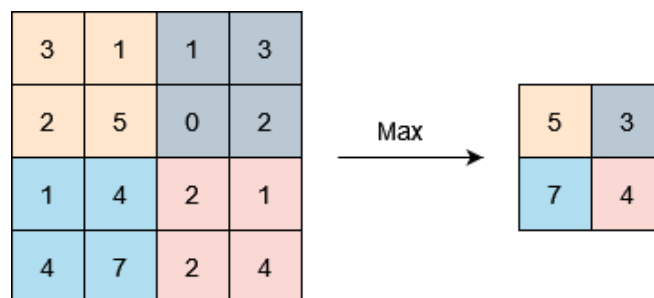
#### 3.2.1.2 Up-Convolution

L'up-convolution ou sur-échantillonnage augmente la largeur et la hauteur de l'image d'entrée en dupliquant les valeurs des pixels pour chaque couche sans poids supplémentaires ni autres opérations complexes. Un exemple d'up-convolution est représenté dans la figure suivante (10), où il y a une entrée plus petite, et la sortie est sur-échantillonnée en effectuant des duplications.

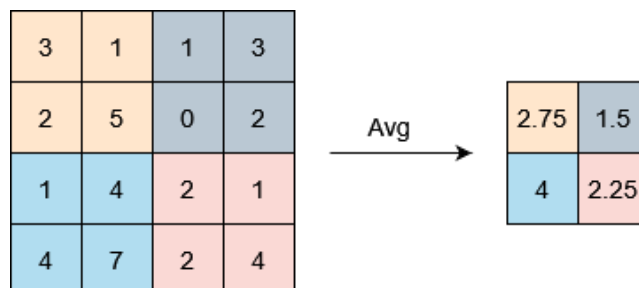
**Figure 10-** Exemple de suréchantillonnage**3.3.2 Couche de Pooling (S2 and S4)**

Lorsque l'entrée est importante, le sous-échantillonnage ou le sous-échantillonnage permet de réduire le nombre de paramètres. Il existe différents types de regroupement, comme indiqué ci-après :

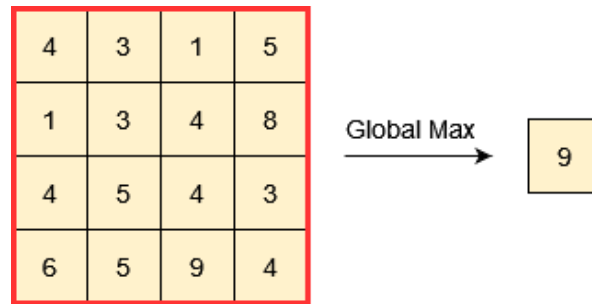
1. **Max-pooling** prend le plus grand élément de la carte des caractéristiques, comme présente dans la figure (11) :

**Figure 11-** Exemple de max-pooling avec fliter  $2 \times 2$  et stride  $2 \times 2$ 

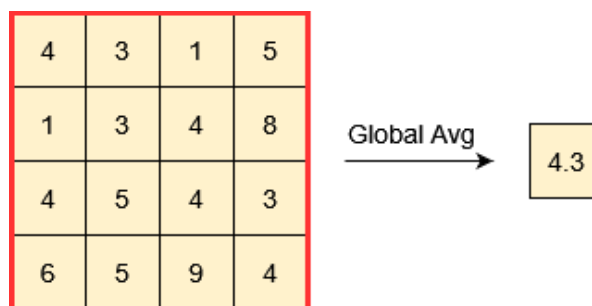
2. **Average pooling** prend l'élément moyen de la carte des caractéristiques, comme présente dans la figure (12) :

**Figure 12-** Exemple d'average pooling avec filtre  $2 \times 2$  et stride  $2 \times 2$ 

3. **Pooling global** a deux types, qui sont détaillés dans les sous-sections suivantes :
  - a. **Global Max-pooling** prend le plus grand nombre dans une fenêtre d'entrée globale, comme présente dans la figure (13) :

**Figure 13-** Exemple de global max-pooling avec une taille de pool égale à la taille d'entrée

- b. **Global Average Pooling** prend la moyenne totale d'une fenêtre d'entrée globale, comme présente dans la figure (14) :

**Figure 14-** Exemple de global average pooling avec une taille de pool égale à la taille d'entrée

### 3.3.3 FCL (F6)

La couche FCL flatte l'entrée de la couche précédente en un vecteur. La couche FCL alimente le vecteur par les ANN, où plus tard une fonction d'activation, comme la fonction SoftMax, classera les sorties de l'entrée alimentée.

### 3.3.4 Fonctions d'activation

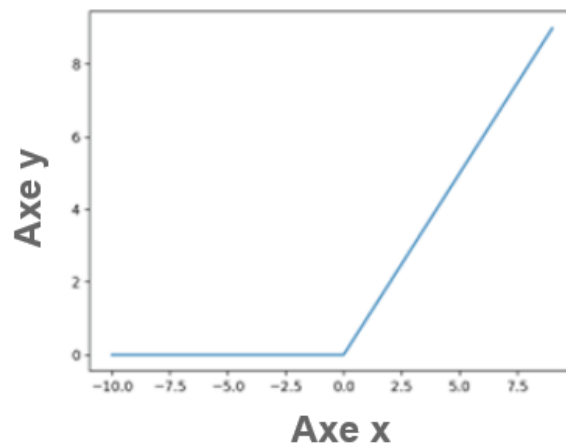
Elle est appelée fonction de transfert si la plage de sortie est infinie, tandis qu'elle est appelée fonction d'écrasement dans le cas où la plage de sortie de la fonction d'activation est limitée, par exemple, la fonction sigmoïde qui transpose les nombres réels entiers en  $[0,1]$ . Désormais, la fonction d'activation calcule la somme pondérée de la couche en la cartographiant dans un vecteur spatial à l'aide de l'une des fonctions : ReLU, LeakyReLU, Sigmoïde, etc.

#### 3.3.4.1 ReLU

La fonction ReLU est calculée comme suit :

$$R(X) = \max(0, X), \quad (1)$$

, où  $X$  est l'entrée, et l'opération  $\max$  prend le maximum de  $(0, \text{entrée } X)$ . Par conséquent, si la valeur d'entrée  $X$  est négative, la fonction renvoie 0. Dans le cas contraire, la fonction renvoie une réponse très positive. Dans la figure suivante (15), la fonction ReLU est bien représentée.

**Figure 15-** Fonction ReLU en représentation graphique

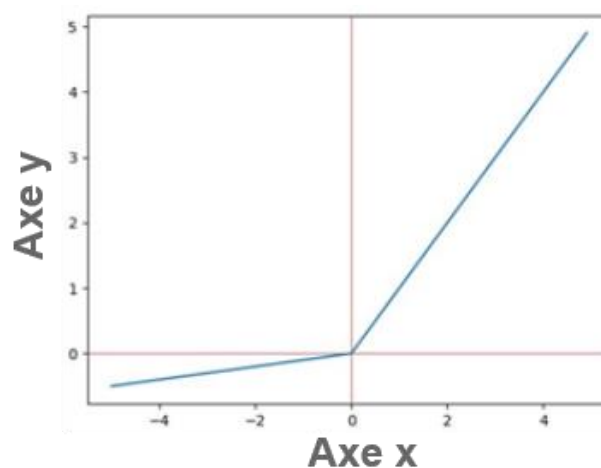
### 3.3.4.2 LeakyReLU

D'autre part, LeakyReLU effectue une valeur seuil chaque fois que l'entrée  $X$  est inférieure à 0. Par conséquent, l'entrée est multipliée par  $\alpha$ . L'opération est résumée comme suit :

$$R(X) = \text{Max}(0.1 * X, X), \quad (2)$$

$$F(X) = \begin{cases} X, & X > 0 \\ \alpha * X, & X < 0 \end{cases} \quad (3)$$

Le LeakyReLU est notamment utilisé pour résoudre le problème des gradients évanescents, car dans le ReLU, l'entrée négative  $X$  est toujours fixée à 0. Dans ce cas, un phénomène de ReLU mort est signalé car aucun apprentissage n'a lieu lorsque le nouveau poids reste comme l'ancien. Par conséquent, la dérivée est 0 dans la rétropropagation. Dans la figure suivante (16), la fonction LeakyReLU est bien représentée.

**Figure 16-** Fonction LeakyReLU en représentation graphique

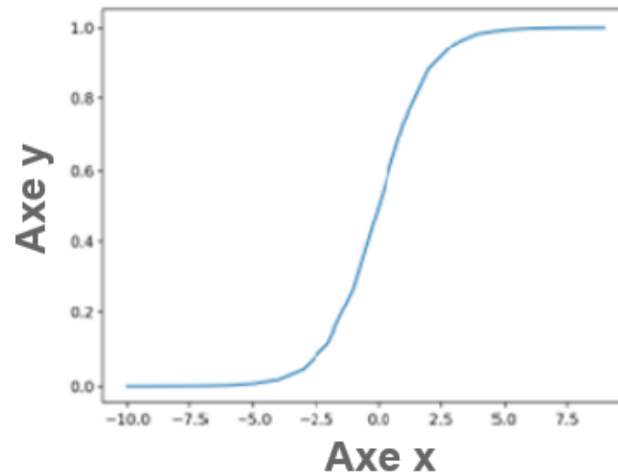
### 3.3.4.3 Sigmoide

L'activation sigmoïde, aussi appelée fonction logistique ou fonction en forme de S, se calcule comme suit :

$$F(X) = \frac{1.0}{1.0 + e^{-X}}, \quad (4)$$

Par conséquent, la fonction prend une entrée de valeur réelle  $X \in \mathbb{R}$  et sort une valeur dans la gamme de 0 à 1. Avec une entrée plus positive, plus la sortie sera proche de 1, tandis qu'avec une valeur plus négative, plus la sortie sera proche de 0. Dans la figure suivante (17), la fonction sigmoïde est bien représentée.

**Figure 17-** Fonction sigmoïde en représentation graphique

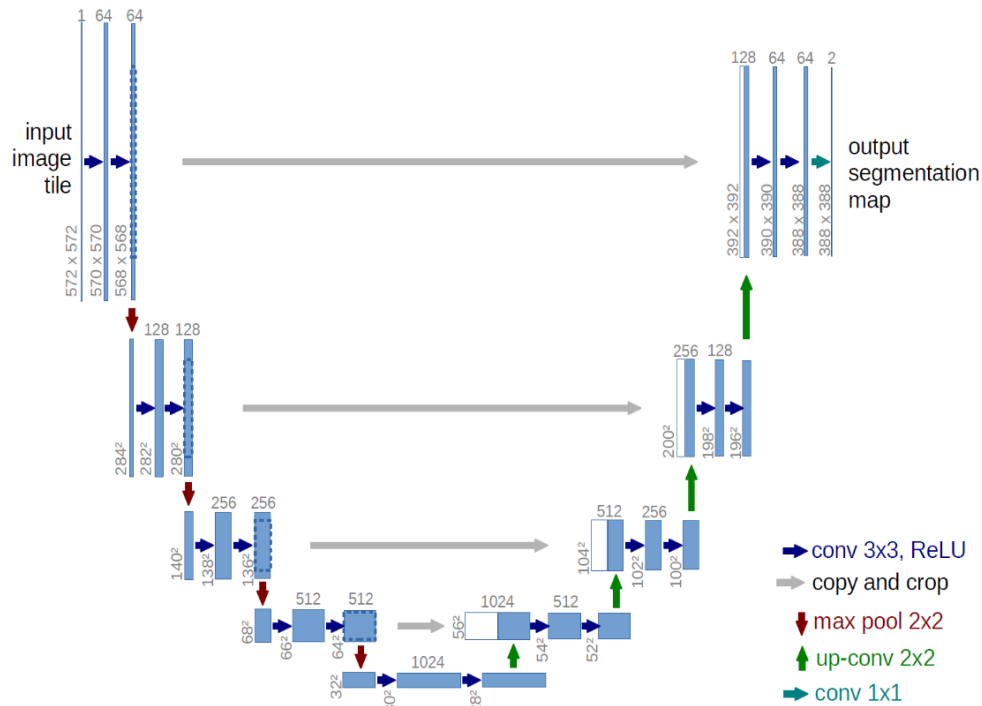


### 3.3 U-Net

U-Net, une segmentation d'image, permet de réduire le volume de données. L'U-Net est composé de deux canaux. Le premier canal ressemble à l'encodeur car il capture le contexte de l'image. D'autre part, le décodeur est une convolution transposée qui travaille pour construire l'image originale sur la base des segments d'image extraits au cours du processus. Fondamentalement, le réseau assemble des couches de convolution suivies d'une mise en commun maximale qui réduit la densité de l'image (réduit les paramètres d'apprentissage du réseau).

Il a été présenté pour la première fois par [19] en 2015 et, comme son nom l'indique, c'est un réseau en forme de U, comme cela est présenté en figure (18), composé de chemins contractés et étendus. Le réseau a une approche de segmentation d'image qui va au-delà de ses concurrents. L'intuition derrière le U-Net, sur une descente (chemin contracté), il apprend la classification des objets. En montée (chemin étendu), il apprend la position de l'objet. En d'autres termes, la couche correspondante sur le chemin raccourci transmet certaines informations au chemin étendu. De cette façon, le contexte de classification est transféré au module de localisation, ce qui rend l'ensemble du réseau beaucoup plus performant.

Figure 18- L'architecture U-Net, qui est extraite de [19][20]



En profondeur, l'architecture se compose de quatre blocs de cryptage et d'un autre de décryptage relié par un pont de connexion. Le réseau de codeurs (chemin contracté) divise par deux la taille de l'espace et double le nombre de filtres (canaux fonctionnels) à chaque bloc de codage. De même, le réseau de décodeurs double la taille de l'espace et divise par deux le nombre de canaux fonctionnels.

Tout d'abord, le réseau d'encodage agit comme un extracteur de caractéristiques et apprend la représentation abstraite de l'image d'entrée par une séquence de blocs d'encodage, qui consistent tous en deux convolutions  $3 \times 3$ . Ensuite, la couche de convolution est suivie d'une fonction ReLU, qui aide à mieux généraliser les données d'apprentissage et introduit une non-linéarité dans le réseau. La sortie de la ReLU sert de connexion de pontage pour le bloc décodeur correspondant.

Après l'encodeur, un max-pooling jusqu'à  $2 \times 2$  réduit les dimensions spatiales (hauteur et largeur) de la carte de caractéristiques. De plus, la normalisation par lots réduit les changements de covariance interne et rend le réseau plus stable pendant l'apprentissage.

Les connexions de saut fournissent des informations supplémentaires au décodeur pour générer de meilleures caractéristiques sémantiques. Elles agissent également comme une connexion de dérivation, permettant un flux de gradation indirect vers les couches précédentes sans dégradation. En outre, elles rationalisent le flux de gradients pendant la rétropropagation, ce qui permet d'améliorer le réseau dans le processus d'apprentissage et de représentation.

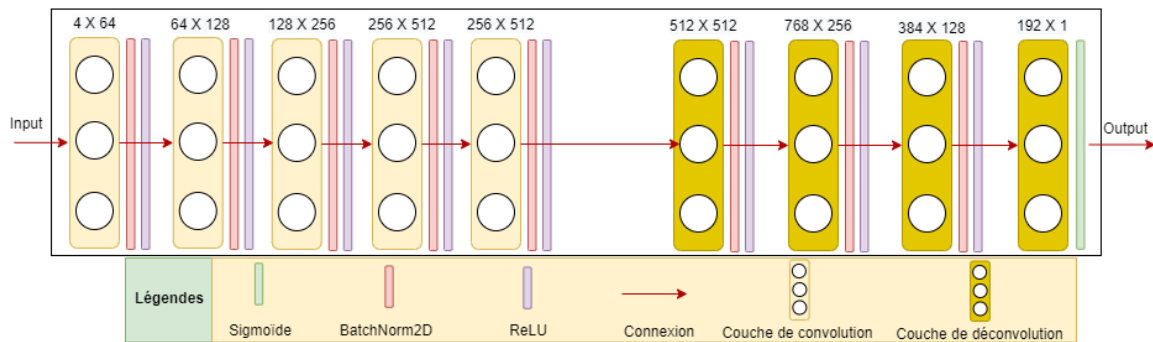
Le réseau d'encodeurs et de décodeurs complète le flux d'informations. Il consiste en deux convolutions  $3 \times 3$ , où chaque convolution est suivie d'une fonction d'activation ReLU. Enfin, le réseau de décodeurs est utilisé pour dériver la représentation abstraite et générer le masque de segmentation sémantique. Le bloc décodeur commence par une convolution de transposition  $2 \times 2$ . Ensuite, il est concaténé avec la carte caractéristique de connexion correspondante du

bloc de données. Ces connexions sautantes fournissent des fonctionnalités des couches précédentes qui sont parfois perdues en raison de la profondeur du réseau. Ensuite, deux convolutions  $3 \times 3$  sont utilisées, où chaque convolution est suivie d'une fonction d'activation ReLU.

La sortie finale du décodeur subit une convolution  $1 \times 1$  avec activation sigmoïde. La fonction d'activation sigmoïde fournit un masque de segmentation qui représente la classification pixel par pixel.

Dans notre cas, le réseau U-Net est bien simplifié dans la figure (19) suivante :

**Figure 19-** Architecture du réseau U-Net



Pour les besoins de la synthèse médicale, il est important de garantir l'apprentissage de la modalité croisée à travers l'architecture. En d'autres termes, le réseau multi-modalité assure l'apprentissage de la segmentation de la tumeur pour guider le processus d'apprentissage. Par conséquent, l'image synthétisée de sortie est concaténée avec l'étiquette de modalité correspondante afin que le réseau de segmentation  $S$  soit capable de générer les cartes de segmentation correspondantes.  $S$  est pré-entraîné sur le jeu de données BRATS2018 en prenant 600 modalités et les paramètres appris sont fixés lors de l'entraînement de l'architecture complète avec les GANs. La perte cohérente multi-modèle est définie comme suit :

$$L_{\text{seg}} = \mathbb{E}_{x,c}[S(G(x,c),l)], \quad (5)$$

Où  $l$  est la vérité de la segmentation de la tumeur.  $S$  garantit la génération  $G$  de la segmentation des tumeurs à partir des différentes modalités  $x$ , car elles contiennent des caractéristiques distinctes, mais il n'est pas nécessaire qu'elles soient identiques à  $c$ .

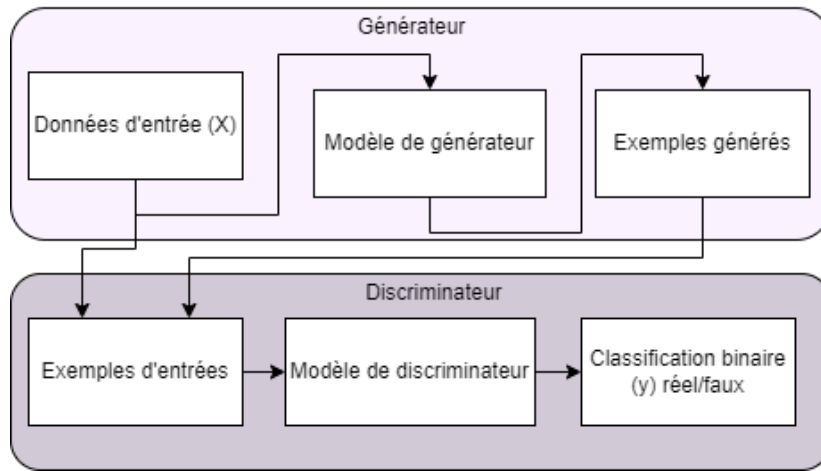
### 3.4 GANs

En spécifiant un objectif de haut niveau, en produisant des résultats indiscernables de la réalité, puis en apprenant automatiquement une fonction de perte appropriée pour satisfaire cet objectif, c'est exactement ce que font les GANs. Les GANs à apprentissage par perte tentent de classer si l'image de sortie est réelle ou fausse en formant un modèle généralisé pour minimiser cette perte. Par conséquent, les images floues ne seront pas acceptées car elles auront manifestement l'air fausses. Comme les GAN apprennent une perte adaptative aux données, ils peuvent être appliqués à une multitude de tâches qui nécessitent traditionnellement différents types de fonctions de perte. La perte contradictoire fait le succès clé des GANs, qui forcent essentiellement les images générées à être indiscernables des vraies.

Par définition, les GANs sont une approche de modélisation générative utilisant des méthodes DL comme dans les CNNs. En pratique, les GAN forment un modèle génératif à partir de deux sous-modèles : le modèle générateur, d'une part, apprend à générer de nouveaux exemples en mettant en correspondance les modalités des entrées des réseaux et, d'autre part, alimente le modèle discriminatoire pour classer les exemples comme réels ou faux.

Comme nous pouvons le voir dans la figure suivante (20), le modèle de générateur de GAN est considéré comme un problème d'apprentissage non supervisé car il génère un échantillon par lot. En retour, ces entités avec les exemples réels alimentent le modèle GANs. Ensuite, le modèle discriminatoire, considéré comme un problème d'apprentissage supervisé, est mis à jour en fonction de sa performance de classification de l'échantillon comme réel ou faux.

**Figure 20-** Le concept des GANs



D'autre part, le GAN unifié, composé d'un générateur et d'un discriminateur uniques, tente de mettre en correspondance des images provenant de quatre modalités. En bref, le générateur prend une image avec sa modalité en entrée pour la synthétiser vers une modalité cible. Le discriminateur est capable de différencier les images réelles et synthétisées selon leurs modalités correspondantes.

En termes mathématiques,  $G: X \rightarrow Y$  tel que la distribution des images de  $G(X)$  est indiscernable de la distribution  $Y$  en utilisant une cartographie de perte adversariale. Le GAN apprend une cartographie à partir de l'image observée  $X$  et du bruit aléatoire  $Z$ , vers l'étiquette  $y$  à travers  $G$  :  $\{X, Z\} \rightarrow Y$  [13]. Le générateur  $G$  est entraîné à produire des sorties qui sont indiscernables des "vraies" images par un discriminateur  $D$  hautement entraîné.  $G$  est entraîné à générer au mieux des images générées par des faux et  $D$  fait de son mieux pour distinguer les vraies des fausses jusqu'à ce qu'elles deviennent semblables.

Le processus d'apprentissage est fortement sous-contraint ; nous le couplons avec un mapping inverse  $F: Y \rightarrow X$  et introduisons une perte CC pour imposer  $F(G(X)) \approx X$ , et vice versa. Pour ne pas perdre de vue qu'il existe deux calculs de pertes CC différents :

1. Perte de CC à avance :

$$X \rightarrow G(X) \rightarrow F(G(X)) \approx X, \quad (6)$$

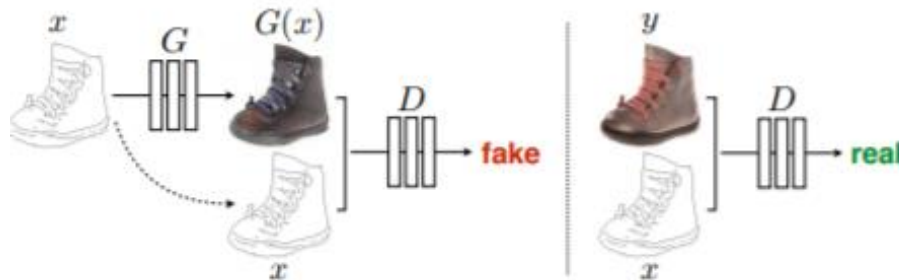
2. Perte de CC à reculons :

$$Y \rightarrow F(Y) \rightarrow G(F(Y)) \approx Y, \quad (7)$$



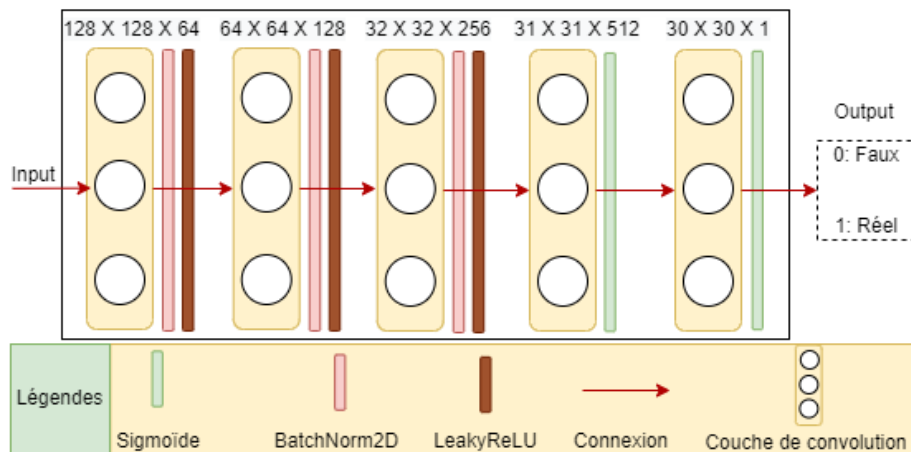
Ainsi, cette procédure de formation est mathématisée dans la figure (21).

**Figure 21-** Démonstration de la cartographie générateur-discriminateur, extraite de [13]



La figure suivante (22) détaille l'architecture GAN en suivant la sortie de l'architecture U-Net, représentée dans la figure (19), sera l'entrée du présent réseau.

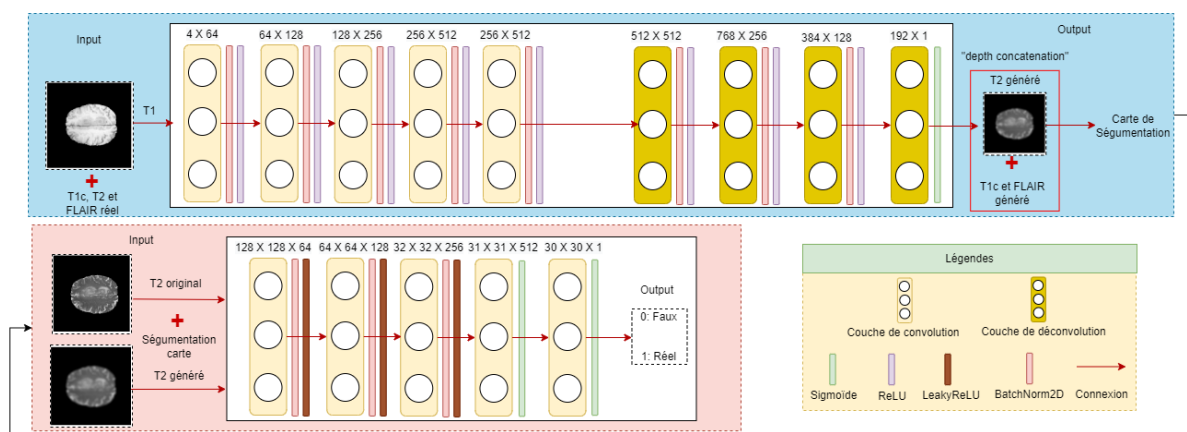
**Figure 22-** L'architecture du GAN



### 3.5 U-Net et GAN

Après la représentation séparée des deux architectures du générateur U-Net et du discriminateur GAN, la figure suivante (23) représente le réseau DL complet en fournissant un exemple fictif. Le générateur prend en entrée T1 pour générer un faux T2, qui sera l'entrée du GAN avec le vrai T2. À la fin, le GAN s'efforcera de différencier le vrai du faux.

Une importance de l'architecture U-Net comme on peut le voir sur la figure (23), une concaténation en profondeur *depth concatenation* des quatre modules générés a été calculée en additionnant leur carte de segmentation afin de générer la segmentation de la tumeur, détaillée dans le réseau S dans la fonction numéro (5).

**Figure 23-** L'architecture d'assemblage de U-Net et des GANs

### 3.5 Jeu de données

BRATS2018 contient des scans IRM 3T multimodaux acquis en clinique, disponibles sous forme de fichiers NIfTI «.nii.gz», sont des volumes T1, T1c, T2 et FLAIR natifs. Ils ont été obtenus à l'aide de différents protocoles cliniques et de différents scanners provenant de plusieurs institutions [5][6].

Toutes les images ont été segmentées manuellement par 1 à 4 évaluateurs utilisant le même protocole d'annotation. Ces annotations ont été finalement approuvées par un neuroradiologue expérimenté. Les données ont été distribuées après enregistrement simultané en utilisant le même modèle anatomique, interpolation à la même résolution  $1mm^3$ , et prétraitement du décapage du crâne.

La figure suivante (24) représente quatre modalités différentes avec leur carte de segmentation.

**Figure 24** – Echantillon de données illustrant les quatre modalités avec carte de segmentation (vérité terrain de tumeur)

L'ensemble de données contient 1425 échantillons de modalités, répartis sur 285 patients chacun. Les patients ont été divisés en sous-catégories : *high grade glioma* (HGG) et *lower grade glioma* (LGG).

$$\text{BRATS2018} \begin{cases} \text{HGG (contient 210 patients)} \\ \text{LGG (contient 75 patients)} \end{cases}$$

Le tableau (4) suivant résume la distribution de l'ensemble des données.

**Tableau 3-** BRATS2018 modality distribution over 285 patients

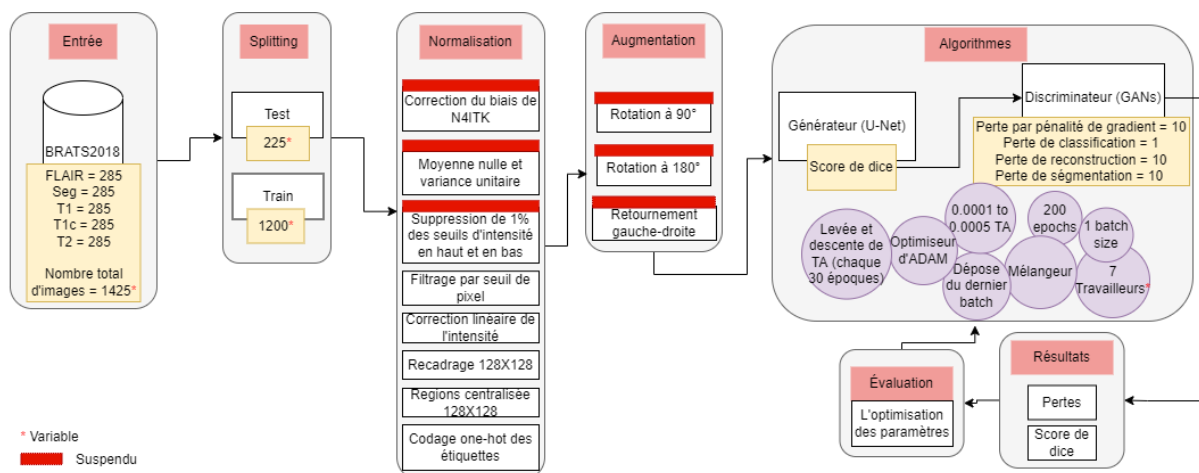
Modalité	Distribution
----------	--------------

T1	285
T2	285
FLAIR	285
T1c	285
Carte de segmentation (seg)	285
<b>Total</b>	<b>1425</b>

#### 4. Approche

Cette section précise les différentes étapes de l'architecture en mettant en lumière l'entrée des données, la division, la normalisation ainsi que les techniques d'augmentation, l'algorithme générateur-discriminateur avec les résultats générés, et enfin l'évaluation et l'optimisation des paramètres. L'ensemble du schéma est présenté dans la figure (25).

**Figure 25-** Architecture du programme, spécifiant les différentes étapes de la saisie des données, du prétraitement, de l'augmentation, de l'algorithme, des résultats et des mesures d'évaluation. Note : le processus marqué en rouge a été suspendu sur les architectures finales



Pour la démonstration, les données sont divisées en 80% de formation et 20% de test. Ensuite, les images sont normalisées en appliquant 5 techniques différentes : correction de biais, normalisation, suppression des intensités, recadrage et enfin codage à un coup. Plus tard dans le processus, les images normalisées sont augmentées par l'application de 3 techniques supplémentaires : retournement de gauche à droite, rotations à 90° et 180°.

L'algorithme consiste donc en un générateur qui prend l'entrée pour générer des modalités fausses que le discriminateur doit classer en deux classes : fausse ou réelle. Plus l'algorithme est incapable de distinguer la différence entre la réalité et l'illusion, plus l'algorithme tend à atteindre la simple perfection.

Le générateur et le discriminateur ont certains paramètres en commun, comme détaillé dans ce qui suit :

1. Optimiseur ADAM avec décroissance de 0,0001 – 0,0005 taux d'apprentissage (TA) et relèvement à chaque 25 époques.
2. 7 travailleurs de calcul (modifié à cause d'erreur en mémoire et processus multiples de Pytorch).

3. L'option "Batch drop last" est activée pour ignorer le dernier lot lorsque le nombre d'exemples n'est pas divisible par le batch size.
4. Mélange des données.
5. 200 époques.

D'autre part, le générateur et le discriminateur ont des paramètres différents. Tout d'abord, le générateur possède les paramètres uniques suivants :

- 1 batch size.
- Score de dice

Le discriminateur distingue certains paramètres, différents du générateur U-Net, détaillés dans ce qui suit :

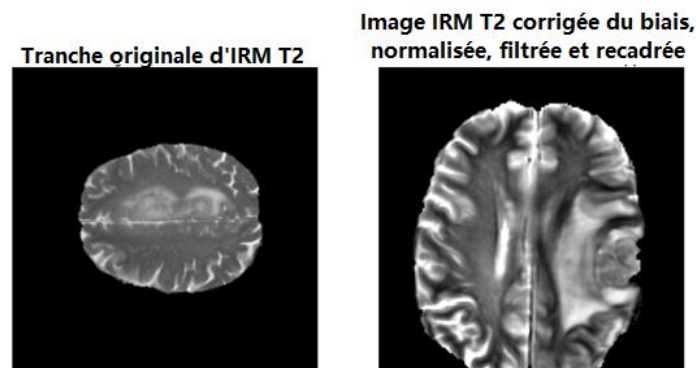
- 1 batch size.
- 10 Perte de reconstruction.
- 10 Pénalité de perte de gradient.
- 10 Perte de segmentation.
- 1 Perte de classification.

Après le processus de formation, les résultats sont évalués en considérant la perte ainsi que le score de dés. A la fin, nous avons adopté une optimisation des paramètres basée sur la performance de chaque processus de formation. Ainsi, une amélioration globale est manifestement dégagée dans le processus d'optimisation.

#### 4.1 Pré-traitement

Cette section comprend différentes étapes, notamment l'encodage des étiquettes en un coup, la suppression de 1% des intensités supérieure et inférieure, la moyenne nulle et la variance unitaire, la correction du biais N4ITK et le recadrage  $128 \times 128 \times 155$  ainsi que  $128 \times 128 \times 155$  centralisé. Un résultat du prétraitement des données est représenté dans la figure suivante (26).

**Figure 26** – Le résultat de l'étape de prétraitement, y compris la correction des biais, la normalisation, le filtrage et le recadrage.



Malheureusement, la normalisation et la correction des biais sont suspendues dans les prototypes finaux car elles ont généré des échecs dans le processus d'apprentissage. Cette partie est suspecte pour une investigation future. En outre, l'augmentation des données est également suspendue dans le prototype final car le modèle de base a pris la plupart du temps dans l'expérimentation, et il serait intéressant de comparer les résultats de base aux résultats du modèle augmenté à l'avenir.

##### 4.1.1 Recadrage des images

Dans cette section, redimensionnement et rognage des images sont détaillés.

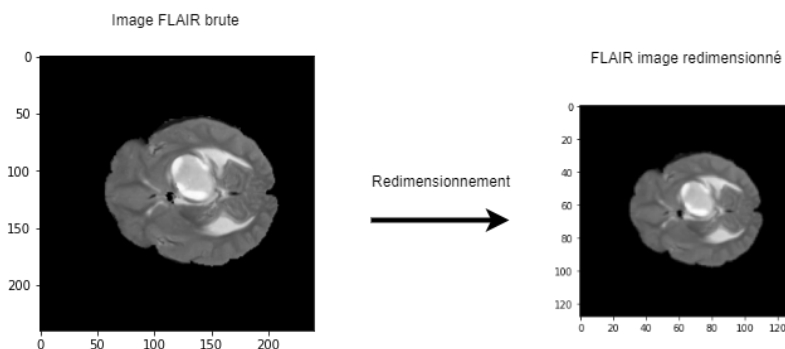
#### 4.1.1.1 Redimensionnement

L'image d'entrée est de  $240 \times 240 \times 155$  redimensionné à  $128 \times 128 \times 155$  pour les 3 prototypes d'architecture DL, comme indiqué ci-dessous :

1. SIMO ( $T1 \rightarrow {}^6 T2$ , FLAIR et  $T1c$ ).
2. MISO T ( $T1$  et  $T2 \rightarrow$  FLAIR).
3. MISO F ( $T1$ ,  $T2$ , et FLAIR  $\rightarrow T1c$ ).

Le résultat du redimensionnement est bien représenté dans la figure suivante (27).

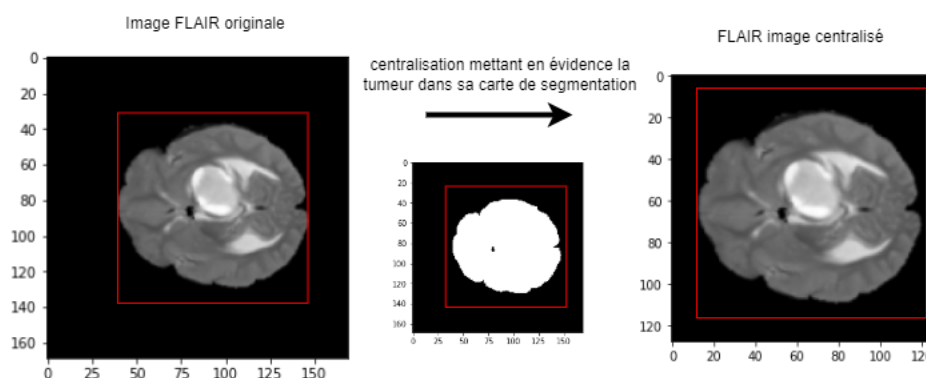
**Figure 27-** Redimensionnement de l'image de  $240 \times 240 \times 155$  à  $128 \times 128 \times 155$



#### 4.1.1.2 Rognage

En outre, une prototype MISO F ( $T1$ ,  $T2$ , et FLAIR  $\rightarrow T1c$ ) rogné de  $128 \times 128 \times 155$  est adopté pour mettre en évidence la zone tumorale dans les modalités à l'aide de la carte de segmentation. Il est bien appliqué au troisième prototype de MISO F, car le but du projet de recherche est d'éviter l'insertion de gadolinium. Donc comme nous voyons dans la figure (28), le cerveau a bien rogné pour centraliser sur la tumeur à l'aide de la carte de segmentation.

**Figure 28-** rognage de l'image de  $168 \times 168 \times 155$  à  $128 \times 128 \times 155$ , mettant en évidence la zone de la tumeur.



#### 4.1.2 Encodage one-hot

Les étiquettes des images sont transférées à l'encodage à un coup, qui consiste essentiellement à mettre 1 chaque fois qu'il y a une tumeur, sinon, elle est notée avec la valeur zéro. La représentation de l'encodage est représentée dans le tableau (4) suivant :

<sup>6</sup>  $T1$  en entrée pour synthétiser  $T2$ , FLAIR et  $T1c$

**Tableau 4-** Encodage one-hot des étiquettes de tumeur et de non-tumeur.

Patients	Catégorie		Patients	Étiquette
Patient #1	Tumeur	→	Patient #1	0
Patient #2	Non-tumeur		Patient #2	1
...	...		...	...

#### 4.1.3 Normalisation

Dans l'étape de normalisation, l'algorithme supprime les 1% d'intensités supérieures et inférieures, puis normalise l'image avec une moyenne nulle et une variance unitaire. Par conséquent, la normalisation des caractéristiques garantit que la moyenne des valeurs de chaque caractéristique dans les données est égale à zéro (si la moyenne est soustraite du numérateur) et a une variance unitaire [18]. Cette méthode est souvent utilisée pour déterminer la moyenne de distribution et l'écart-type de chaque caractéristique. Ensuite, soustrayez la moyenne de chaque caractéristique. Divisez ensuite la valeur de chaque caractéristique (la moyenne a déjà été soustraite) par son écart-type, comme suit :

$$x' = \frac{x - \bar{x}}{\sigma}, \quad (8)$$

où  $x$  est le vecteur de caractéristiques original,  $\bar{x}$  = moyenne( $x$ ) est la moyenne du vecteur de caractéristiques, et  $\sigma$  est son écart-type.

En outre, les régions hors arrière-plan doivent être normalisées puisque les intensités vont de 0 à 5000. Le minimum dans la tranche normalisée correspond à l'intensité 0 dans la tranche non normalisée, qui a été remplacée par  $-9$  pour garder la trace des intensités 0.

#### 4.1.4 Filtration

Les tranches d'images 2D sont filtrées en fixant un seuil où le nombre de pixels dans la zone du cerveau est inférieur à 2000.

#### 4.1.5 Correction de l'intensité linéaire

Les tranches d'images ont été mises à l'échelle de façon linéaire de leur intensité originale à  $[-1, +1]$ .

#### 4.1.6 Correction du biais

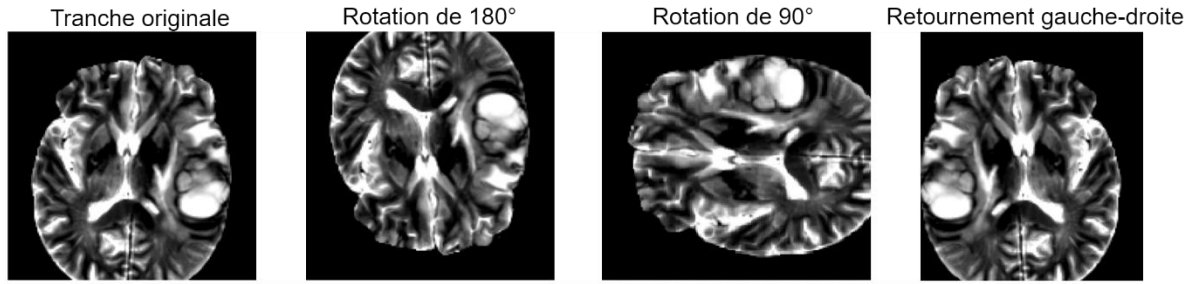
L'algorithme de correction de champ biaisé N4 corrige les inhomogénéités d'intensité à basse fréquence présentes dans les données d'image IRM, connues sous le nom de champs biaisés ou renforcés. Cette méthode a été rapportée comme ayant été appliquée avec succès comme correction de champ plat pour les données de microscopie. En outre, elle part d'un modèle paramétrique simple et ne nécessite pas de classification des tissus [17].

#### 4.1.7 Augmentation des données

L'augmentation des données est adoptée car l'ensemble de données est considéré comme petit pour la mise en œuvre de DL et en particulier de réseaux génératifs. Par conséquent, nous avons adopté différentes techniques d'augmentation en utilisant des fonctions définies et l'aide de TensorFlow en appliquant des rotations de  $90^\circ$  et  $180^\circ$  et un retournement de gauche à droite de l'image d'entrée originale. Par exemple, la figure (29) suivante représente un exemple d'augmentation de données appliquée à la tranche « Brats18\_TCIA01\_390\_1\_78 ».



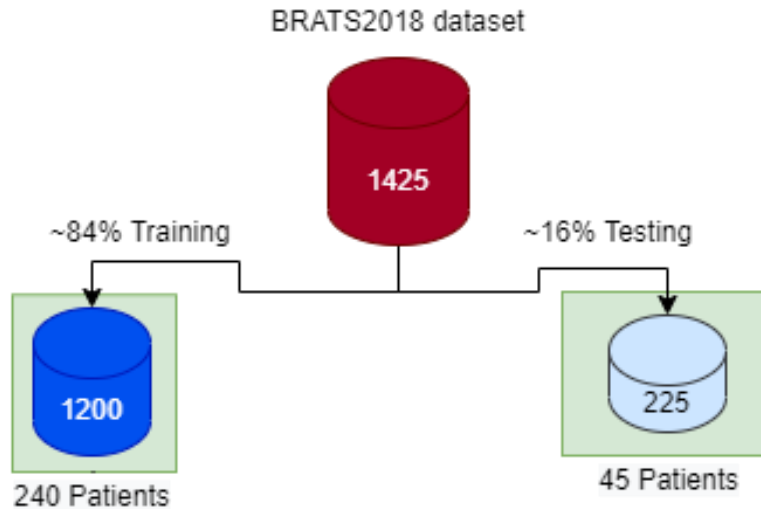
**Figure 29-** Augmentation des données, y compris : des rotations à  $90^\circ$  et  $180^\circ$  et des retournements de gauche à droite.



#### 4.1.8 Répartition des données

Les données d'entraînement sont divisées en deux ensembles, l'un d'entraînement et l'autre de test. Pour les 3 prototypes indiqués précédemment dans la section **4.1.1.1 Redimensionnement**, Tout d'abord, l'ensemble d'entraînement représente  $\sim 84\%$  de l'ensemble de données d'entraînement et l'ensemble de test représente  $\sim 16\%$ . Par conséquent, l'ensemble d'apprentissage contient 1200 images pour chaque générateur et discriminateur et l'ensemble de test comprend le reste des images, soit 225 images au total. Le processus de fractionnement est détaillé et bien documenté dans la figure suivante (30) :

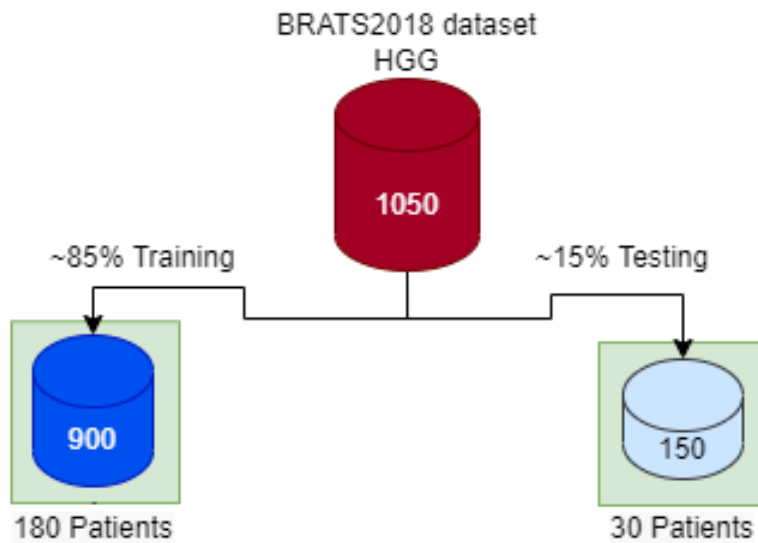
**Figure 30-** Données divisées en ensembles de test et de formation pour les prototypes SIMO, MISO T et MISO F rogné avec une taille d'entrée d'image de  $128 \times 128$ .



Comme nous avons pu le voir sur la Figure (30), l'ensemble de test contient 225 échantillons divisés par 4 modalités : T1, T2, T1c, FLAIR et sa carte de segmentation. D'autre part, 1200 échantillons d'entraînement sont dédiés à l'entraînement de U-Net et de GANs.

D'autre part, pour le quatrième prototype de MISO F rogné des régions  $128 \times 128$  détaillé dans la section **4.1.1.2 Rognage**, le jeu de données est divisé en 150 ensembles de test et 900 ensembles d'entraînement sur la seule sous-catégorie HGG BRATS2018. La catégorie HGG a été sélectionnée car elle ne contient que des tumeurs de type très malin. Le processus de fractionnement est représenté dans la figure suivante (31) :

**Figure 31-** Données divisées en ensembles de test et d'apprentissage pour le prototype MISO F avec un rognage de  $128 \times 128$  à l'entrée de l'image.



## 4.2 Algorithme

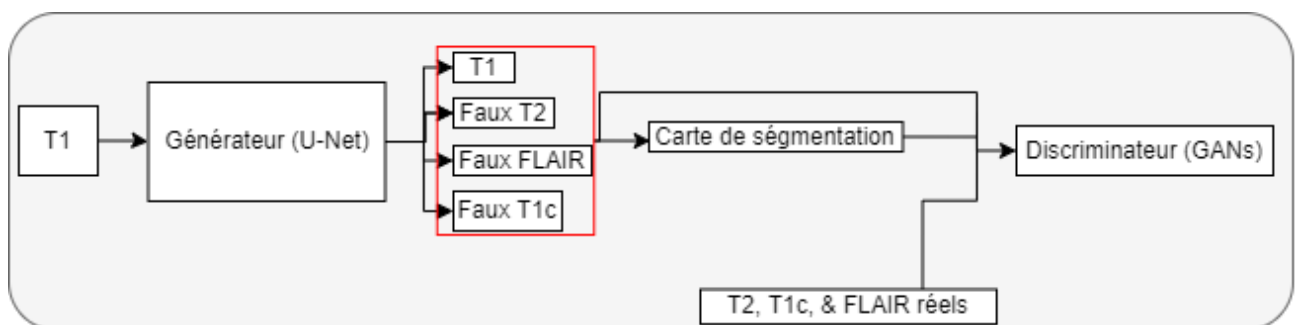
Avec l'existence de nombreux modèles CNN profonds qui obtiennent des résultats compétitifs en termes d'état de l'art, j'ai choisi l'U-Net et le GANs pour affiner le jeu de données que nous avons entre les mains pour son efficacité et sa simplicité. L'U-Net, comme décrit précédemment, gère la cartographie de la segmentation tandis que l'algorithme GAN joue le rôle de discriminer les fausses images générées plus tard dans le processus de formation.

Dans la section suivante, nous présenterons les différentes méthodes algorithmiques que nous avons développées pour combler les lacunes de la recherche dans la littérature par 3 méthodes différentes, mais nous limiterons l'expérimentation à la fin aux méthodes SIMO et MISO car nous visons à prédire FLAIR et T1c.

### 4.2.1 SIMO

L'architecture SIMO prend une seule entrée, par exemple, dans notre cas T2 pour prédire le reste des modalités en passant par le générateur U-Net et le discriminateur GAN. L'architecture est bien présentée dans la figure suivante (32) :

**Figure 32-** L'architecture SIMO prend T1 en entrée, et synthétise T2, FLAIR, T1c, et la carte de segmentation pour la discrimination

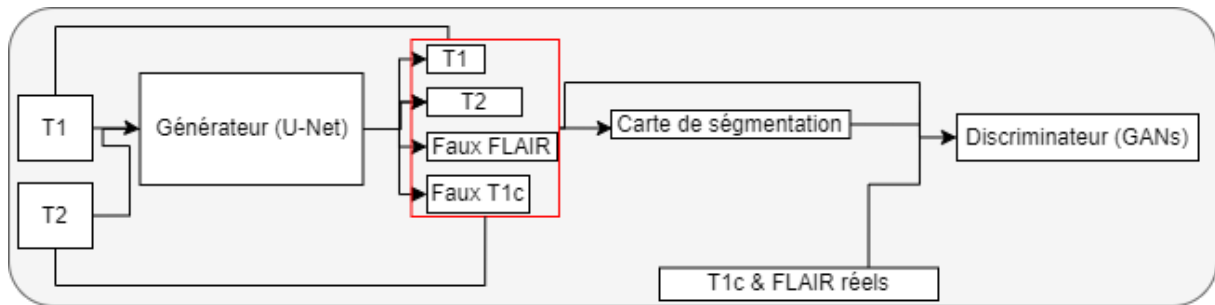




#### 4.2.2 MIMO

L'architecture MIMO prend plusieurs entrées, par exemple, dans notre cas, T1 et T2 pour prédire le reste des modalités en passant par le générateur U-Net et le discriminateur GAN. L'architecture est bien présentée dans la figure suivante (33) :

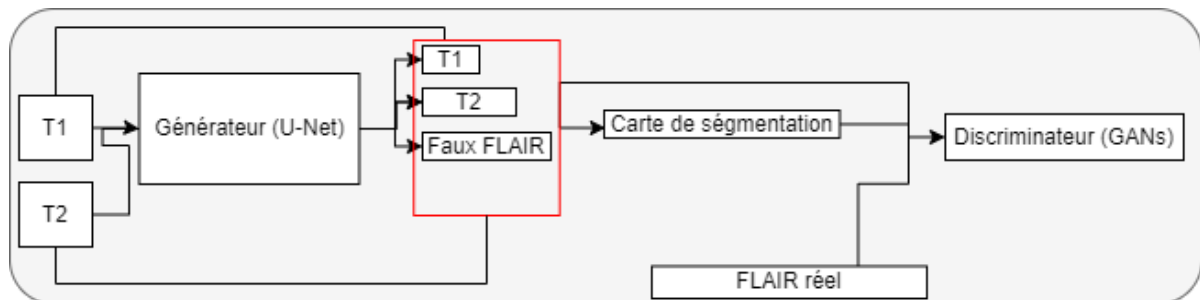
**Figure 33-** L'architecture MIMO prend T1 et T2 en entrée, et synthétise T1c et FLAIR pour la discrimination



#### 4.2.3 MISO

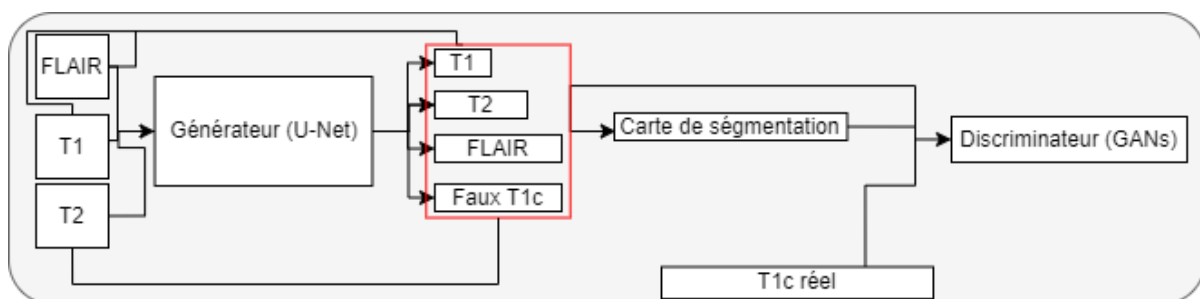
L'architecture MISO prend plusieurs entrées, par exemple, dans notre cas T1 et T2 pour prédire FLAIR ou T1c en passant par le générateur U-Net et le discriminateur GAN. La première architecture de T1 et T2 → FLAIR est bien présentée dans la figure suivante (34) :

**Figure 34-** MISO prend T1 et T2 en entrée, et synthétise FLAIR pour la discrimination



D'autre part, la seconde est l'architecture de T1, T2, et FLAIR → T1c est bien présentée dans la figure suivante (35) :

**Figure 35-** MISO prend T1 et T2, et FLAIR en entrée, et synthétise T1c pour la discrimination



#### 4.3 Evaluation Metrics

Le générateur U-Net, prédisant la carte de ségmentation et le sous-échantillonnage, évalué par le score du dé. Le générateur U-Net et le discriminateur GAN sont évalué par différents poids de perte comme mentionner dans la liste suivante :

- Pénalité de gradient
- Perte de classification
- Perte de reconstruction
- Perte de segmentation

Pour plus de détails, les paramètres de perte sont détaillés en **8. Annexe**.

## 5. Expériences et résultats

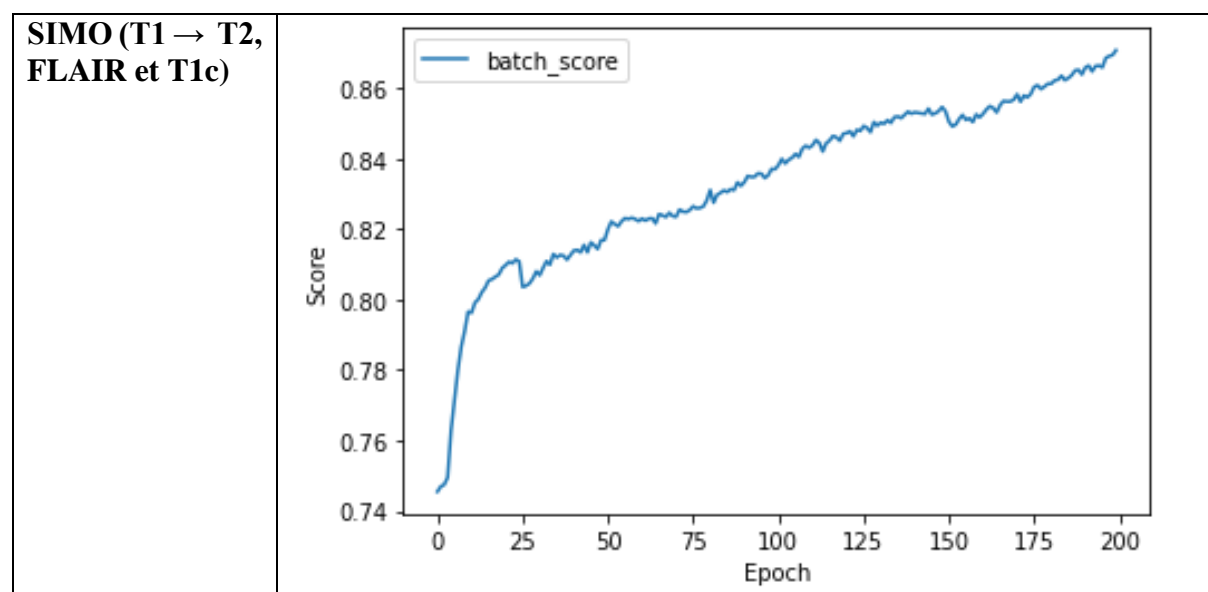
L'entraînement de 3 modèles a été effectué sur le laboratoire XLIM, tandis que l'apprentissage de l'ensemble de U-Net et GAN dans l'architecture de MISO F rogné a été suspendu puisque les ressources sur I3M ne pouvaient pas le gérer. Par conséquent, il est laissé pour une expérimentation future sur Nvidia TESLA XLIM. L'expérience finale a été lancée sur les spécifications informatiques suivantes dans le tableau (5) du 23 juin 2022 au 31 août 2022 :

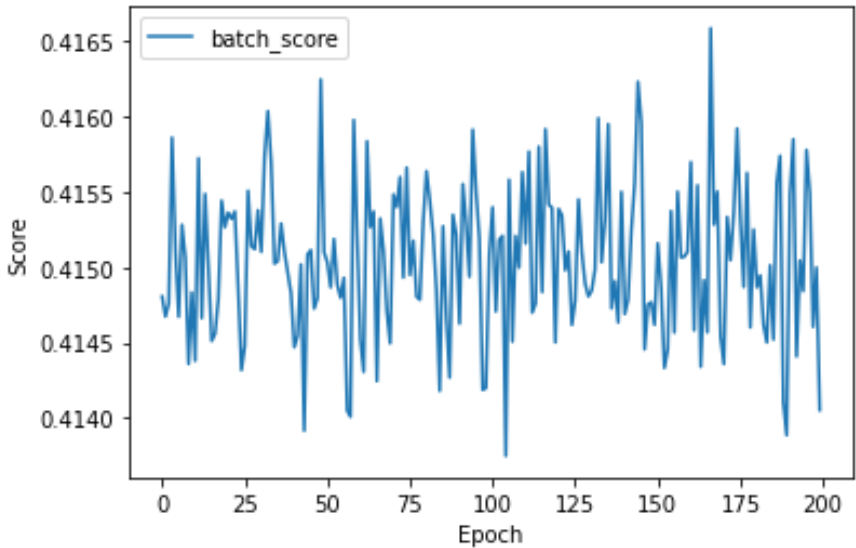
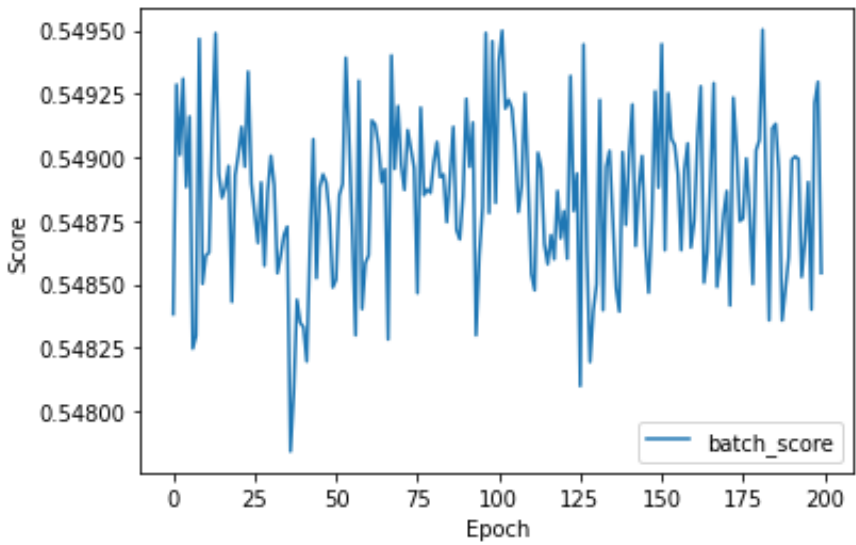
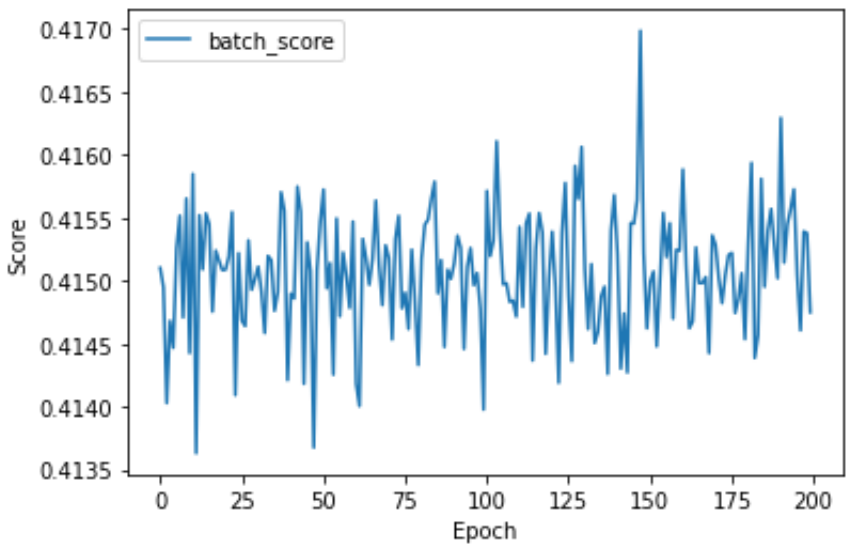
**Tableau 5-** Spécifications de l'ordinateur de laboratoire

<b>Marque</b>	DELL R740
<b>CPUs</b>	2 Intel Xeon Silver 4208 2.1GHz, 11M Cache
<b>GPUs</b>	3 Nvidia Tesla V100S 32G
<b>RAM</b>	384 GB
<b>Shared wireless access protocol (SWAP)</b>	128 GB
<b>Stockage</b>	22 TB
<b>Carte de réseau</b>	10 GB

Dans la première phase, le générateur assure de meilleurs résultats car il génère à la fois l'image faussée et sa carte de segmentation. Plus tard, au cours de l'entraînement de l'architecture de l'ensemble U-Net et GAN, les résultats remarqueront une amélioration significative grâce à la perte contradictoire.

**Tableau 6-** Score de U-Net des 4 architectures, SIMO a atteint 87% ACC, MISO T a atteint 0.414%, MISO F redimensionné a atteint 0.414%, et MISO F rogné a atteint 0.548%



<b>MISO T (T1 et T2 → FLAIR)</b>	
<b>MISO F rogné (T1, T2, et FLAIR → T1c)</b>	
<b>MISO F redimensionné (T1, T2, et FLAIR → T1c)</b>	

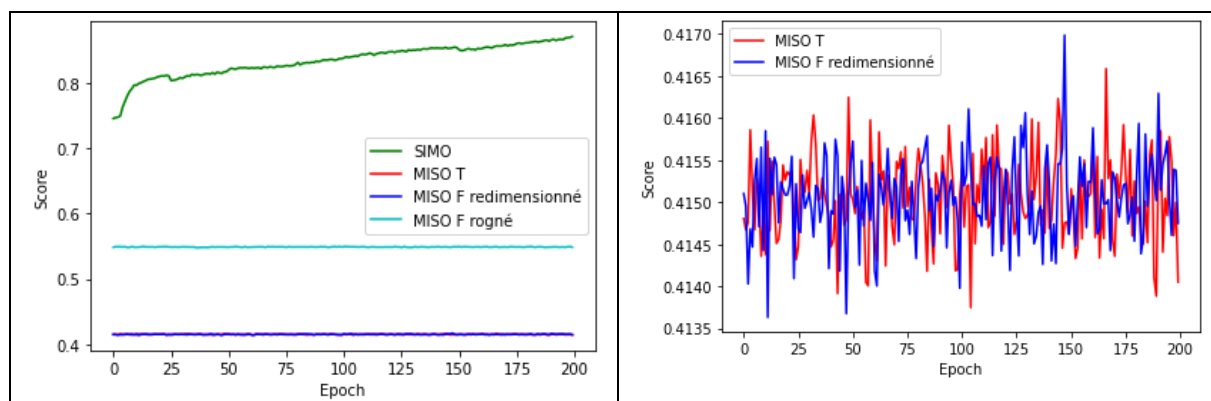
A partir du tableau (6), nous avons pu constater une bonne performance de l'architecture U-Net qui atteint une précision de 87% sur l'architecture SIMO. Cependant, nous avons pu observer

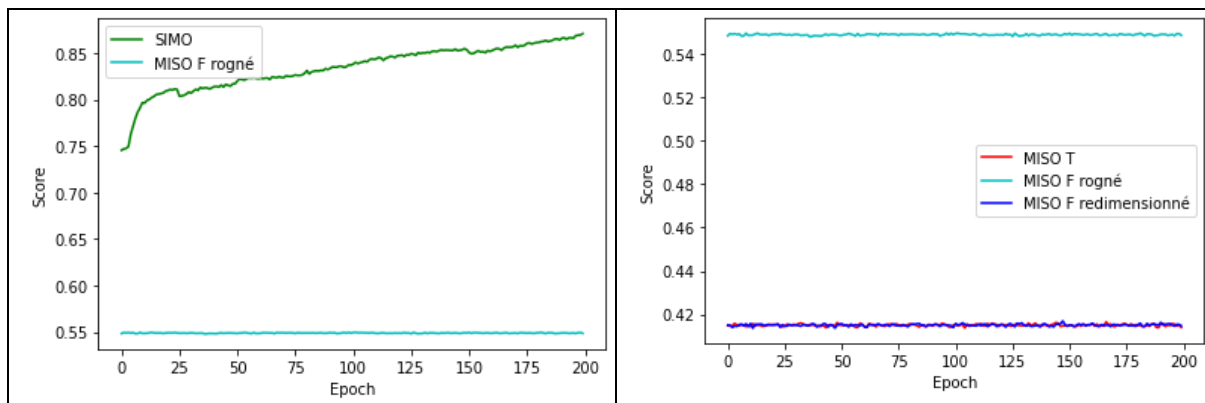
une forte oscillation sur les trois autres architectures. En fait, l'algorithme U-Net, bien qu'il utilise la même architecture que SIMO, mais avec des sorties multiples, souffre dans l'apprentissage du processus de formation. Il est bien observé une énorme tendance fluctuante sur les époques, la fonction de l'algorithme est soufferte pour se mettre à jour et trouver le minimum local ou le minimum global des données d'entrée. Par conséquent, la fonction n'a pas convergé. De nombreuses explications peuvent être tirées des résultats obtenus :

- Le TA est plus élevé ou plus bas qu'il ne devrait l'être, malgré l'adoption de la décroissance du TA et de l'augmentation de la mise à jour.
- La taille du lot pourrait être si petite que l'algorithme ne parvient pas à générer de fausses images à partir de 3 entrées différentes, comme c'est le cas dans l'architecture SIMO. Par conséquent, l'inclusion de deux branches supplémentaires de traitement ultérieur dans le modèle est encouragée pour traiter les trois entrées en une seule, suivie d'une couche FCL pour aplanir la sortie.
- Indépendamment de la nouveauté et de l'efficacité de la deuxième proposition, elle est susceptible de provoquer un manque de mémoire et l'échec général de l'apprentissage.
- L'échantillon de données pourrait être moins représentatif puisqu'il comprend des HGG et des LGG dans la MISO T et MISO F redimensionné. Cependant, l'évidence d'une amélioration est représentée dans MISO F rogné car il est bien entraîné sur HGG BRATS2018, où il inclut un type de tumeur maligne.
- Le score de dice n'est généralement pas bien adopté pour évaluer les performances de U-Net.

Une comparaison détaillée entre les différentes architectures est bien représentée dans le tableau suivant (7).

**Tableau 7-** Score de U-Net, une comparaison entre SIMO, MISO T, MISO F redimensionné, et MISO F rogné

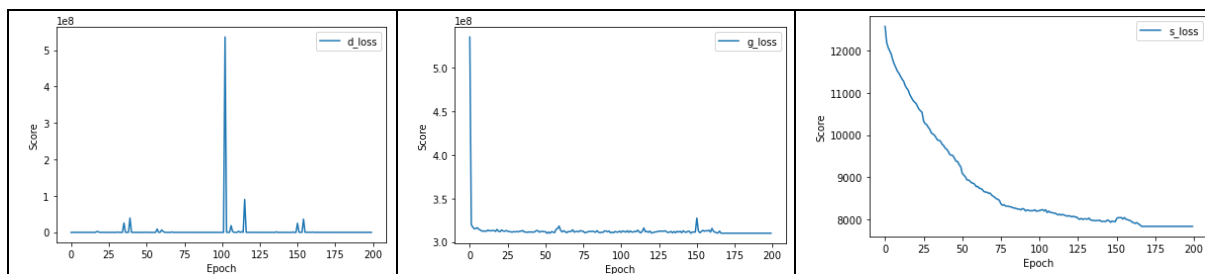




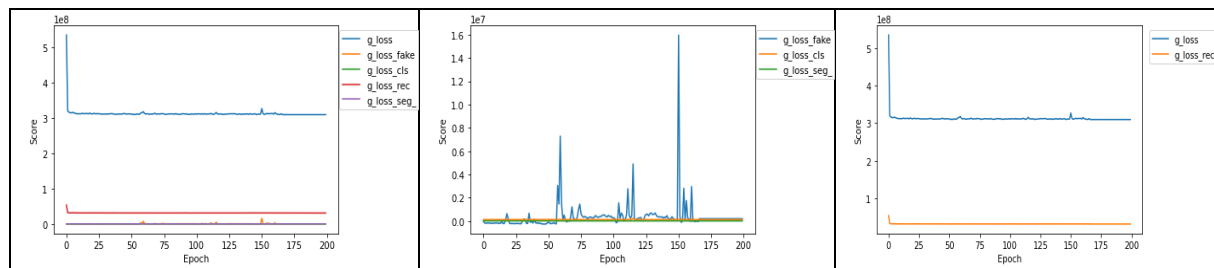
En contraire, l'ensemble d'algorithme est bien détaillé et discuter dans la sous-section suivante :

Premièrement, l'architecture SIMO a obtenu des résultats performants comme on peut le constater dans le tableau suivant (8) représentant la somme des pertes du discriminateur GAN, du générateur U-Net et du processus de segmentation. On remarque que la perte se rapproche de 0 sur l'indicateur de score tout au long du processus de croissance sur l'époché dans l'axe des abscisses. Apparemment, pour le générateur et le discriminateur, environ après la 10ème époque, la perte continue de s'approcher fermement avec une certaine fluctuation mais sans descendre plus bas. Cependant, la perte de segmentation a continué à s'approcher de 0 au fil de l'apprentissage.

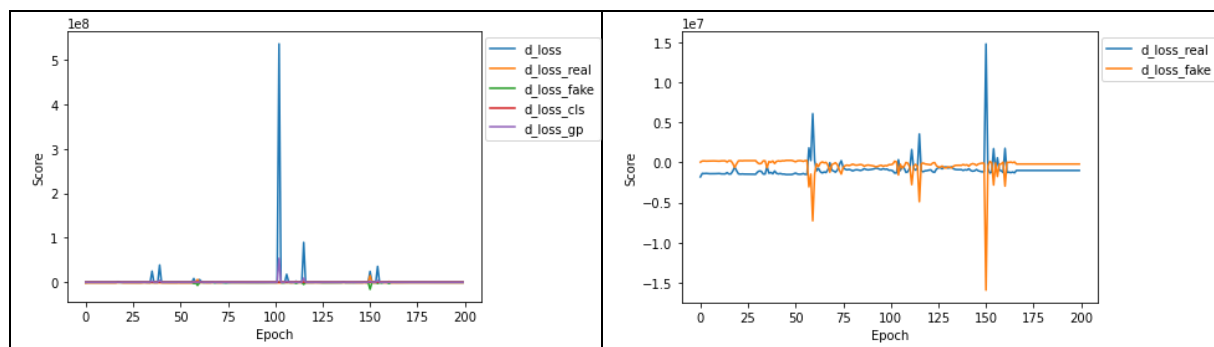
**Tableau 8-** Score de SIMO, une comparaison entre les pertes de discrimination, génération, segmentation



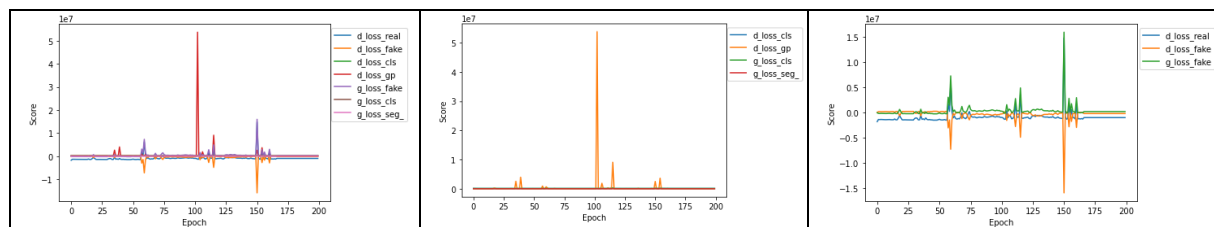
Dans le tableau (9), pour plus de détails sur les différentes pertes calculées du générateur, on peut remarquer que le générateur s'approche de 0, sauf pour la perte additionnée. Cependant, l'indicateur d'importance du modèle, la perte du générateur de faux et de segmentation, sont tous deux légèrement proches de zéro, ce qui signifie que la perte du modèle montre une bonne performance à partir de l'entraînement sur l'ensemble de données et du test sur l'ensemble de hold-out.

**Tableau 9-** Score de SIMO, une comparaison entre les pertes de générateur

En outre, le discriminateur montre les mêmes résultats que le générateur U-Net, les pertes sont proches de 0 comme il est représenté dans le tableau (10). La discrimination des modalités fausses et réelles s'approche de 0 dans une relation entrelacée, ce qui valide la performance du modèle et la perfection de l'entraînement et du test.

**Tableau 10-** Score de SIMO, une comparaison entre les pertes de discriminateur

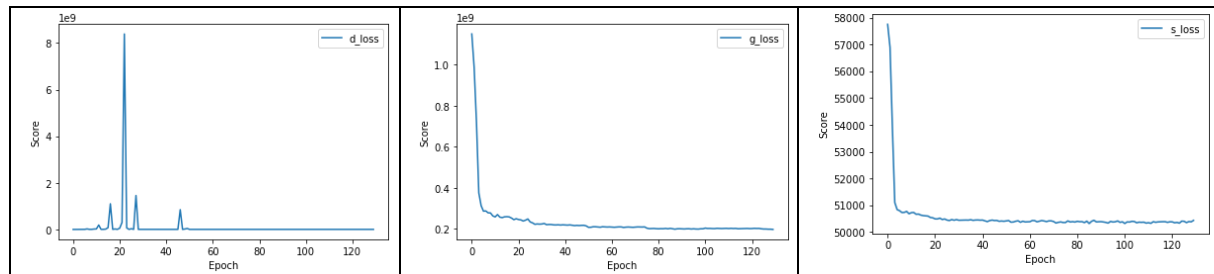
Par conséquent, le modèle SIMO a prouvé ses performances et son efficacité, comme en témoignent les pertes, qui sont proches de 0. Le modèle a été formé sur le jeu de données et testé sur le jeu d'essai avec de faibles pertes. Ceci conclut que le modèle n'est pas surajusté à l'ensemble d'entraînement comme constaté dans le tableau (11). Il peut donc être généralisé à d'autres jeux de données.

**Tableau 11-** Score de SIMO, une comparaison entre les pertes de discriminateur et de générateur

Ensuite, l'architecture MISO T aussi obtenu des résultats robustes comme on peut le voir dans le tableau (12) suivant représentant la somme des pertes du discriminateur GAN, du générateur U-Net et du processus de segmentation. On remarque que la perte tend vers zéro sur l'indicateur de score lors de la croissance dans le temps en abscisse. Évidemment, pour les générateurs et les discriminateurs, environ après la 20ème époque, la perte continue de s'approcher avec

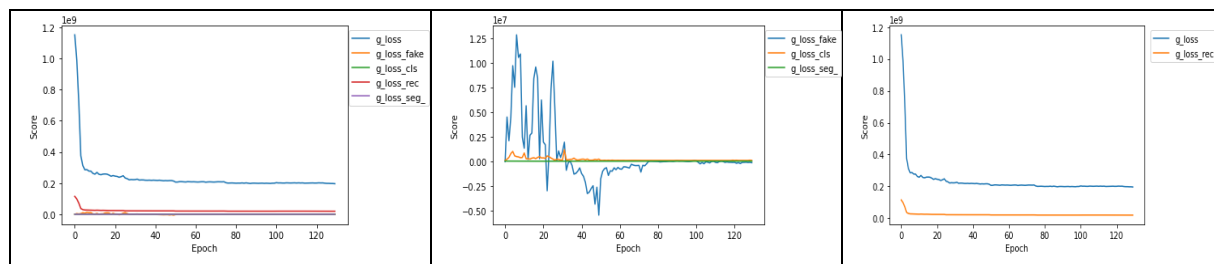
certitude avec une certaine volatilité mais ne baisse pas plus bas. Cependant, la perte de segmentation continue à zéro au fur et à mesure de l'apprentissage.

**Tableau 12-** Score de MISO T, une comparaison entre les pertes de discrimination, génération, segmentation



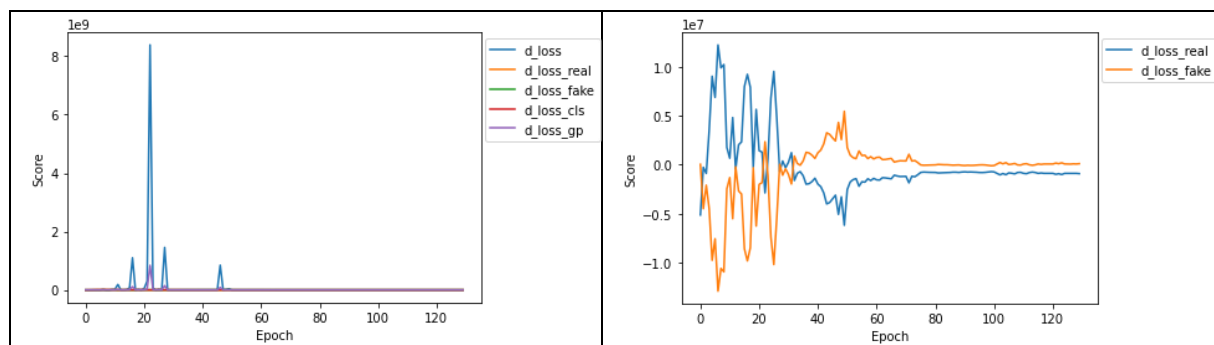
Dans le tableau (13), pour plus de détails sur les différentes pertes calculées du générateur, on peut remarquer que le générateur s'approche de 0, sauf pour la perte additionnée. Cependant, l'indicateur d'importance du modèle, la perte du générateur de faux et de segmentation, sont tous deux légèrement proches de zéro, ce qui signifie que la perte du modèle montre une bonne performance à partir de l'entraînement sur l'ensemble de données et du test sur l'ensemble de hold-out.

**Tableau 13-** Score de MISO T, une comparaison entre les pertes de générateur



En outre, le discriminateur montre les mêmes résultats que le générateur U-Net, les pertes sont proches de 0 comme il est représenté dans le tableau (14). La discrimination des modalités fausses et réelles s'approche de 0 dans une relation entrelacée, ce qui valide la performance du modèle et la perfection de l'entraînement et du test.

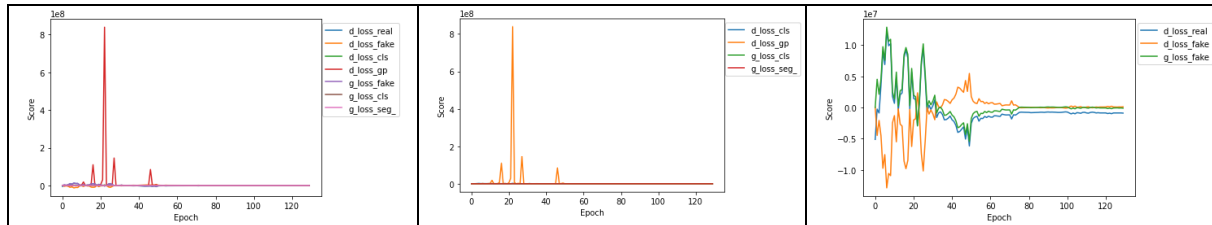
**Tableau 14-** Score de MISO T, une comparaison entre les pertes de discriminateur





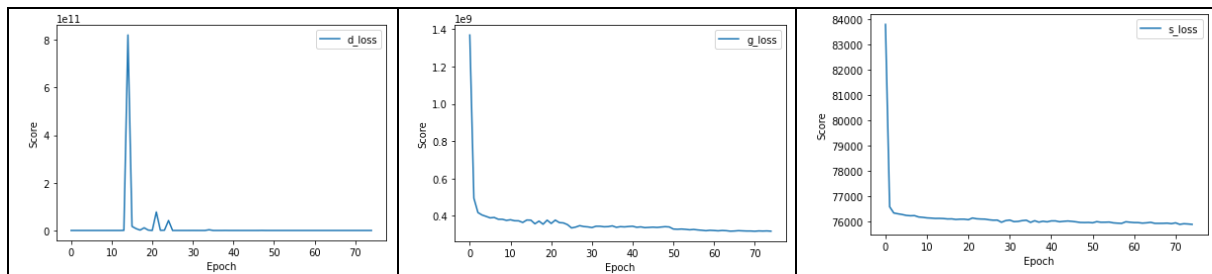
Par conséquent, le modèle MISO T a prouvé ses performances et son efficacité, comme en témoignent les pertes en tableau (15), qui sont proches de 0. Le modèle a été formé sur le jeu de données et testé sur le jeu d'essai avec de faibles pertes. Ceci conclut que le modèle n'est pas surajusté à l'ensemble d'entraînement. Il peut donc être généralisé à d'autres jeux de données.

**Table 15-** Score de MISO T, une comparaison entre les pertes de discriminateur et de générateur

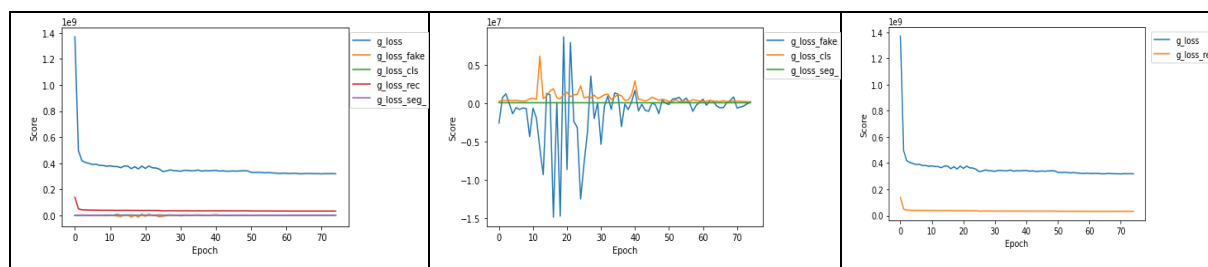


Aussi, l'architecture MISO F redimensionné aussi obtenu des résultats robustes comme on peut le voir dans le tableau (16) suivant représentant la somme des pertes du discriminateur GAN, du générateur U-Net et du processus de segmentation. On remarque que la perte tend vers zéro sur l'indicateur de score lors de la croissance dans le temps en abscisse. Évidemment, pour les générateurs et les discriminateurs, environ après la 10ème époque, la perte continue de s'approcher avec certitude avec une certaine volatilité mais ne baisse pas plus bas. Cependant, la perte de segmentation continue à zéro au fur et à mesure de l'apprentissage.

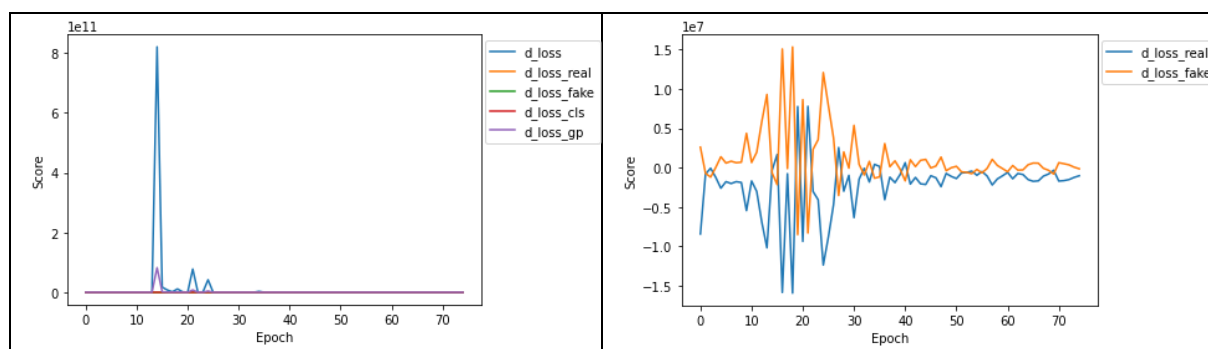
**Tableau 16-** Score de MISO F redimensionné, une comparaison entre les pertes de discrimination, génération, segmentation



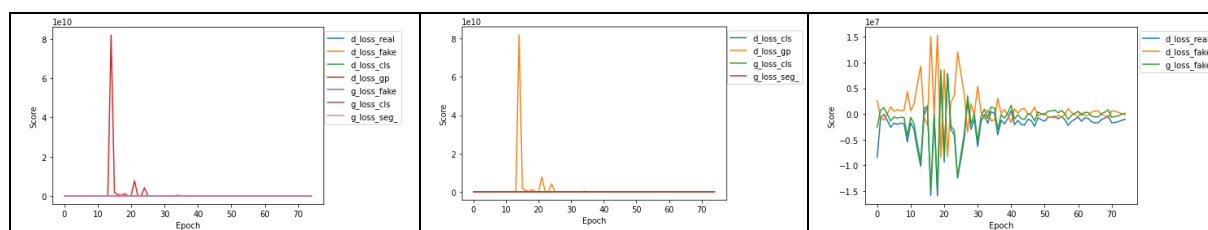
Dans le tableau (17), pour plus de détails sur les différentes pertes calculées du générateur, on peut remarquer que le générateur s'approche de 0, sauf pour la perte additionnée. Cependant, l'indicateur d'importance du modèle, la perte du générateur de faux et de segmentation, sont tous deux légèrement proches de zéro, ce qui signifie que la perte du modèle montre une bonne performance à partir de l'entraînement sur l'ensemble de données et du test sur l'ensemble de hold-out.

**Tableau 17-** Score de MISO F redimensionné, une comparaison entre les pertes de générateur

En outre, le discriminateur montre les mêmes résultats que le générateur U-Net, les pertes sont proches de 0 comme il est représenté dans le tableau (18). La discrimination des modalités fausses et réelles s'approche de 0 dans une relation entrelacée, ce qui valide la performance du modèle et la perfection de l'entraînement et du test.

**Tableau 18-** Score de MISO F redimensionné, une comparaison entre les pertes de discriminateur

Par conséquent, le modèle MISO F redimensionné a prouvé ses performances et son efficacité, comme en témoignent les pertes en tableau (19), qui sont proches de 0. Le modèle a été formé sur le jeu de données et testé sur le jeu d'essai avec de faibles pertes. Ceci conclut que le modèle n'est pas surajusté à l'ensemble d'entraînement. Il peut donc être généralisé à d'autres jeux de données.

**Tableau 19-** Score de MISO F redimensionné, une comparaison entre les pertes de discriminateur et de générateur

A partir des résultats obtenus, nous avons pu conclure à la preuve du concept de génération de T1c, et d'autres modalités dans différents contextes, ce qui valide l'objectif de notre recherche.

## **6. Conclusion**

Cette section est consacrée à la conclusion de la recherche. Elle comprend les deux sous-sections des conclusions professionnelles et personnelles.

### **6.1 Professionnel**

Grâce à l'art DL, j'ai pu expérimenter différents paramètres d'architecture sur le jeu de données. Grâce à ces expériences, j'ai également constaté que pour ce jeu de données, l'ajustement fin de l'ensemble du modèle donne non seulement de meilleurs résultats mais aide également le modèle à converger beaucoup plus rapidement que l'ajustement fin du reste des couches.

Les architectures développées, SIMO, MISO T, MISO F redimensionné ont prouvé l'objectif final dans la synthèse d'image en générant des modalités fausses qui ressemblent aux vraies. La recherche a prouvé sa nouveauté en proposant diverses architectures, l'une d'entre elles MISO F pouvant générer T1c avec injection de gadolinium.

### **6.2 Personnel**

L'opportunité de stage que j'ai eue avec le laboratoire XLIM a été une grande chance d'approfondir mon apprentissage et ma carrière professionnelle dans le traitement de l'image (précisément, sur l'imagerie médicale et la bio-ingénierie), ML, et DL. Ainsi, je me considère comme une personne chanceuse d'avoir le bon poste dans l'ingénierie des données et du ML. Je suis très reconnaissant d'avoir la chance de mettre en pratique de nombreuses ressources de connaissances et de compétences d'indépendance, de recherche et de développement personnel. C'est avec une grande confiance que je peux dire que j'ai la chance de rencontrer différents spécialistes dans le secteur de la santé ainsi qu'en informatique, notamment en DL et ML. En général, ils m'ont conduit bien sûr dans la bonne direction et dans la direction souhaitée pendant la période de stage du 28 mars au 1er septembre de l'année 2022.

### **6.3 Travaux futurs**

Nous pourrions également envisager d'expérimenter différentes fonctions de perte, l'optimisation des hyper-paramètres, des réseaux génératifs plus profonds et l'augmentation des données.

### **6.4 Challenges**

La recherche a été confrontée à plusieurs obstacles concernant le manque de contrôle de l'administration, l'indisponibilité des données pour le stage, un débogage important et des risques inattendus. Avec la limitation d'accès sur les ordinateurs des laboratoires, j'ai été contraint de prototyper l'algorithme dans mon ordinateur personnel puis de lancer le test final sur la machine du laboratoire. Cela n'empêche pas de fait les problèmes émergents concernant les différentes versions des bibliothèques ainsi que les réglages de l'environnement de programmation. Cependant, la création d'un environnement virtuel permet d'assurer la compatibilité. D'autre part, le processus de débogage a pris plus de temps car les architectures sont longues et les problèmes viennent de plusieurs côtés. Par conséquent, j'ai dû prendre l'initiative dans le processus de débogage, en cherchant et en adoptant des solutions. En outre, les données n'étaient pas disponibles pour le stage jusqu'à la preuve de concept des architectures développées. La preuve de concept a été réalisée comme discuté dans les résultats, donc, ils seraient disponibles pour la perspective de recherche future. Enfin, des risques inattendus ont été soulevés comme l'arrêt du serveur de calcul TESLA qui a ruiné l'apprentissage de tout un mois. Cependant, grâce à l'enregistrement automatique du modèle et des poids dans des fichiers extérieurs, les résultats ont pu être représentés, analysés et utilisés pour de futures perspectives.

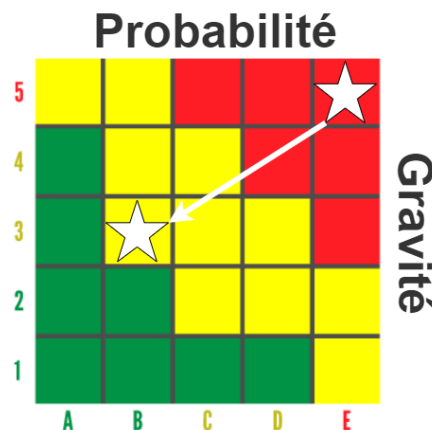
## 6.5 Analyse des risques

Dans cette section, je présenterai les risques rencontrés pendant toute la période du stage.

### 6.5.1 L'indisponibilité de l'ensemble des données

En raison de l'indisponibilité du jeu de données du CHU de Poitiers, nous avons décidé d'expérimenter les architectures de réseaux génératifs sur un jeu de données open-source, appelé BRATS2018. Après avoir pris en compte le plan proposé, nous avons réussi à réduire la gravité du risque d'une gravité et d'une probabilité élevées à une probabilité plus faible avec une gravité moyenne faible.

**Figure 36-** Matrice de probabilité/gravité du risque de non-disponibilité de l'ensemble de données



### 6.5.2 Utilisateur administratif non subventionné

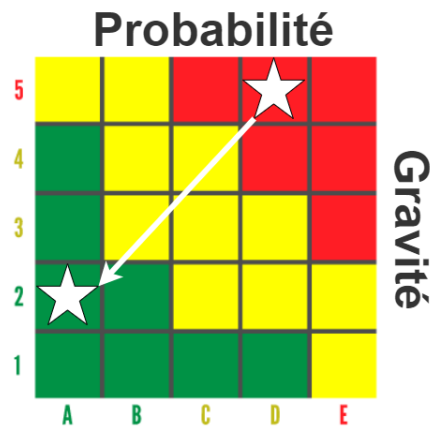
Dans le laboratoire XLIM, le droit d'administration n'est pas donné à un stagiaire, pour une période temporaire. Le problème est que je n'ai pas le droit d'installer les applications par moi-même et que la gestion de l'environnement de programmation me semble impossible. A partir de là, j'ai pris les solutions suivantes pour limiter le risque :

1. Utiliser mon ordinateur personnel pour effectuer le test du prototype.
2. Installer des applications avec des extensions \*.tar pour éviter les droits d'administration.
3. Acquérir deux machines différentes à distance pour finaliser les prototypes finaux.
4. Travailler avec l'environnement virtuel Python car anaconda demande beaucoup de droits d'administration.

Après avoir pris en compte les solutions ci-dessus, le risque a donc considérablement diminué, comme nous pouvons le visualiser dans la figure suivante (37) :

<sup>7</sup> Le nom de l'application souhaitée.

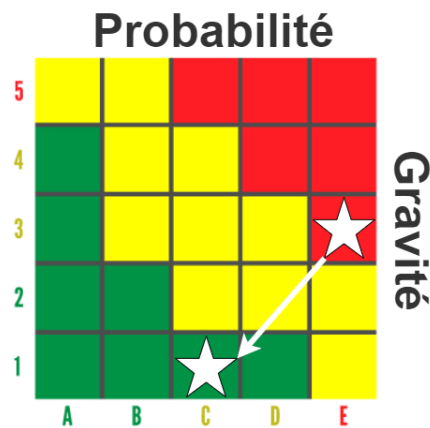
**Figure 37-** Matrice de probabilité/gravité du risque de non-subventionnement de l'utilisateur de l'administration



### 6.5.3 L'ambiguïté du cahier des charges

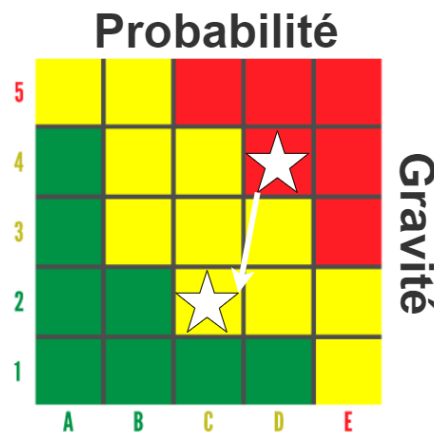
Au début du stage, le cahier des charges demandé pour l'ensemble de la période de stage était un peu trop large et ne se limitait pas à un domaine de recherche précis. Plus tard, après une réunion avec le tuteur pédagogique, il m'a demandé d'organiser un planning complet pour l'ensemble du stage. Pour cela, j'ai fait du GanttProject ainsi que du PERT pour occulter les tâches à venir et les tâches en cours. Dès lors, la gravité du problème s'est réduite avec une probabilité moindre.

**Figure 38-** Matrice de probabilité/gravité du risque d'ambiguïté des spécifications



### 6.5.4 Entraînement prolongé des architectures DL

En raison du délai prolongé et du délai d'apprentissage des différentes architectures DL, nous avons proposé de recadrer les tranches d'IRM. Un recadrage de  $128 \times 128$  est mis en œuvre pour réduire la gravité du problème sur l'architecture MISO, qui prend en entrée : T1, T2, FLAIR, et des sorties : T1c. Il n'est implémenté que sur cette architecture car l'objectif de la recherche est d'éviter l'injection de gadolinium. Le risque est bien diminué comme le montre la figure suivante (39).

**Figure 39-** Matrice de probabilité/sévérité du risque de réalisation prolongée des modèles

### 6.5.5 Débogage d'architectures en Pytorch

Le débogage des architectures est un défi constant et difficile avec l'implémentation de Pytorch dans le projet. En plus de la complexité des algorithmes et du développement, les erreurs et la nouveauté du sujet font qu'il est difficile de retrouver l'erreur exacte comme dans le débogage d'un problème apporté par un autre. Par conséquent, le débogueur python dans le code VS a été utilisé pour suivre la zone des erreurs déclenchées en plus du simple « print » pour suivre les modifications.

Cependant, en déplacement sur I3M, les prototypes développés dans XLIM sur TESLA ne fonctionnent plus sur l'équipement là-bas. Tout d'abord, le problème « client\_loop : send disconnect » a été constant au début de la réalisation de l'architecte de GAN de MISO F centralisée. En outre, un autre problème est le multiprocessing déclenché lorsque l'algorithme commence à tourner en arrière-plan. La mémoire a été largement utilisée comme le montre bien la figure suivante (40) :

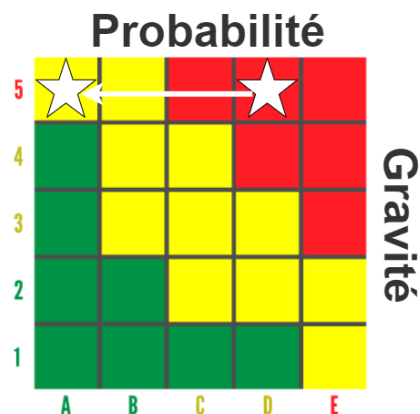
**Figure 40-** Multiprocessus et erreur de mémoire

```
PS C:\Users\CH8178> & C:/Users/CH8178/Miniconda3/envs/scpenv/python.exe e:/Marwan/MISOT128cropped/Algo.py
initialize network with normal
launch algo
Traceback (most recent call last):
  File "<string>", line 1, in <module>
  File "C:\Users\CH8178\Miniconda3\envs\scpenv\lib\multiprocessing\spawn.py", line 116, in spawn_main
    exitcode = _main(fd, parent_sentinel)
  File "C:\Users\CH8178\Miniconda3\envs\scpenv\lib\multiprocessing\spawn.py", line 126, in _main
    self = reduction.pickle.load(from_parent)
MemoryError
```

Après avoir adopté quelques solutions sur le réseau, le problème est resté entier, avec une perte de temps considérable dans le traitement. À la fin, « from subprocess import call » a résolu tout le problème du multiprocessing.

En ce qui concerne l'expansion de la mémoire, la réduction du nombre de travailleurs diminuera l'utilisation de la mémoire, et l'algorithme sera donc entraîné en douceur. Comme il est présenté dans la figure (41), la gravité du risque est élevée mais, comme dans l'expérimentation, elle est considérée comme faible, sauf dans différents contextes, y compris l'environnement de développement et les spécifications d'ordinateur.

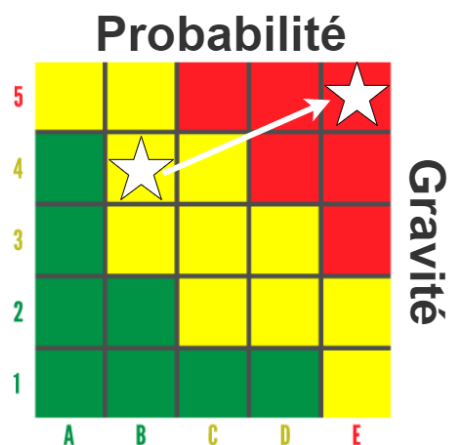
**Figure 41-** Matrice de probabilité/sévérité du risque de formation prolongée des Multiprocessus et erreur de mémoire



### 6.5.6 Arrêt totale de serveur de calcul

En raison de la maintenance du serveur après les vacances d'été, le serveur TESLA a été mis en veille afin de corriger le problème déclenché par une inadéquation de la *Nvidia management library (NVML)* entre les drivers installés et des library utilisées par le serveur. L'intervention n'est prévue et le maintenant a fait sans consultation de réservation ni les researchers. Au milieu d'apprentissage, ce genre d'intervention peut également être programmé pour un moment ultérieur car il n'est pas important pour le moment où les 2 modèles s'entraînent en parallèle. Il est donc important d'avoir une connexion SSH et un serveur stable pour assurer la stabilité de l'apprentissage. Les modèles sont forcés de s'arrêter, donc nous avons eu une gravité d'impact élevée avec une faible probabilité mais le problème s'est aggravé à une probabilité/gravité maximum car les modèles doivent être réentraînés à partir de zéro.

**Figure 42-** Matrice de probabilité/gravité du risque l'arrêt du serveur de calcul





## 7. Références

- [1] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net : Réseaux convolutifs pour la segmentation d'images biomédicales. Informatique des images médicales et intervention assistée par ordinateur. Notes de lecture en sciences informatiques, vol 9351. Springer, Cham. DOI : [https://doi.org/10.1007/978-3-319-24574-4\\_28](https://doi.org/10.1007/978-3-319-24574-4_28)
- [2] Choi, Y., Choi, M., Kim, M.S., Ha, J., Kim, S. & Choo, J. (2018). StarGAN : réseaux adversariaux génératifs unifiés pour la traduction image à image multi-domaine. IEEE Conférence sur la vision par ordinateur et la reconnaissance des formes, 8789-8797.
- [3] Dai, X., Lei, Y., Fu, Y., Curran, W.J., Liu, T., Mao, H. & Yang, X. (2020). Synthèse multimodale d'IRM à l'aide de réseaux génératifs adversariens unifiés. La physique médicale. DOI : <https://doi.org/10.1002/mp.14539>
- [4] Menze et al. (2015). Le banc d'essai multimodal de segmentation d'images de tumeurs cérébrales (BRATS). IEEE transactions sur l'imagerie médicale, 34(10), 1993-2024. DOI : <https://doi.org/10.1109/TMI.2014.2377694>
- [5] Bakas, S., Akbari, H., Sotiras, A., Bilello, M., Rozycki, M., Kirby, J.S., Freymann, J.B., Farahani, K. & Davatzikos, C. (2017). Faire progresser les collections d'IRM de gliome du Cancer Genome Atlas avec des étiquettes de segmentation expertes et des caractéristiques radiomiques. Données scientifiques, 4.
- [6] Bakas et al. (2018). Identification des meilleurs algorithmes d'apprentissage automatique pour la segmentation des tumeurs cérébrales, l'évaluation de la progression et la prédiction de la survie globale dans le cadre du défi BRATS. ArXiv, abs/1811.02629.
- [7] Xlim Présentation. (n.é.<sup>8</sup>). Consulté le 10 mai 2022, à l'adresse <https://www.xlim.fr/laboratoire/presentation>
- [8] Université de Standford. (2017). RSNA 2017 : Les rads qui utilisent l'IA vont remplacer les rads qui ne le font pas. Consulté le 3 mai 2022 sur le site <https://aimi.stanford.edu/news/rsna-2017-rads-who-use-ai-will-replace-rads-who-don-t>
- [9] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). L'apprentissage par gradient appliqué à la reconnaissance de documents. Proc. IEEE, 86, 2278-2324.
- [10] Meskó, B., & Görög, M. (2020). Un petit guide pour les professionnels médicaux à l'ère de l'intelligence artificielle. npj Digit. Med. 3, 126. DOI : <https://doi.org/10.1038/s41746-020-00333-z>
- [11] Krizhevsky, A., Sutskever, I., & G. E. Hinton. (2012). Classification d'images avec des réseaux neuronaux convolutifs profonds. Avancées dans les systèmes de traitement de l'information neuronale, 1097-1105.
- [12] K. He, X. Zhang, S. Ren & J. Sun. (2015). Apprentissage résiduel profond pour la reconnaissance d'images. arXiv préprint arXiv:1512.03385.
- [13] Isola, P., Zhu, J., Zhou, T. & Efros, A. (2017). Traduction d'image à image avec des réseaux adversariaux conditionnels. IEEE Conférence sur la vision par ordinateur et la reconnaissance des formes, 5967-5976.
- [14] Liu, M.-Y., Breuel, T. & Kautz, J. (2017). Réseaux de traduction image à image non supervisés. arXiv préprint arXiv:1703.00848.
- [15] Kim, T., Cha, M., Kim, H., Lee, J.K. & Kim, J. (2017). Apprendre à découvrir des relations inter-domaines avec des réseaux adversariaux génératifs. DOI : <https://doi.org/10.48550/arXiv.1703.05192>

---

<sup>8</sup> Non mentionné

- [16] Zhu, J., Park, T., Isola, P. & Efros, A.A. (2017). Traduction image à image non appariée à l'aide de réseaux adversariaux cohérents avec le cycle. IEEE Conférence internationale sur la vision par ordinateur, 2242-2251.
- [17] Tustison, N. J., Avants, B. B., Cook, P. A., Zheng, Y., Egan, A., Yushkevich, P. A., & Gee, J. C. (2010). N4ITK : correction améliorée du biais N3. IEEE transactions sur l'imagerie médicale, 29(6), 1310-1320. DOI : <https://doi.org/10.1109/TMI.2010.2046908>
- [18] Grus, Joel (2015). La science des données à partir de zéro. Sebastopol, CA : O'Reilly. pp. 99, 100. ISBN 978-1-491-90142-7.
- [19] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net : Réseaux convolutifs pour la segmentation d'images biomédicales. ArXiv, abs/1505.04597.
- [20] Ronneberger, O. (2015). U-Net : Réseaux convolutifs pour la segmentation d'images biomédicales. Université de Freiburg. Consulté le 20 juillet 2022, à l'adresse <https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net>
- [21] Wood, T. (2019). Réseau neuronal convolutif. Consulté le 22 juillet 2022, à l'adresse <https://deeptai.org/machine-learning-glossary-and-terms/convolutional-neural-network>
- [22] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). L'apprentissage par gradient appliqué à la reconnaissance de documents.
- [23] Zwanenburg, J.J.M., Hendrikse, J., Visser, F. et al. (2010). Fluid attenuated inversion recovery (FLAIR) IRM à 7.0 Tesla : comparaison avec 1.5 et 3.0 Tesla. Eur Radiol 20, 915-922. DOI : <https://doi.org/10.1007/s00330-009-1620-2>
- [24] Laboratoire I3M. I3M. (n.é.). Consulté le 29 juillet 2022, à l'adresse <https://i3m.labo.univ-poitiers.fr>
- [25] Plateforme IRM 7 TESLA. Plateforme IRM 7 tesla. (n.é.). Consulté le 29 juillet 2022 sur le site <https://www.chu-poitiers.fr/specialites/irm7tesla>
- [26] Al Omari, M. & Al-Hajj, M. (2019). Classificateurs pour le traitement automatique des langues arabes : enquête. Journal international de la complexité informatique et des algorithmes intelligents. DOI : <https://doi.org/10.1504/IJCCIA.2018.10019805>
- [27] Gulrajani, I, Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Amélioration de l'apprentissage des GANs du wasserstein. Avancées dans les systèmes de traitement de l'information neuronale, 5767–5777.

## 8. Annexe

Dans cette section, nous fournirons quelques détails sur les ensembles de données qui ne sont pas bien développés dans la section principale. De plus, un détail sera fourni concernant les fonctions de perte des architectures U-Net et GAN, et enfin un test d'un modèle déployé de SIMO en tant que microservice. En outre, un résumé détaillé de l'état de l'art est présenté dans la dernière partie. Le code source est disponible : <https://github.com/marwanalomari/Intelligence-artificielle-imagerie-mdicale-enrichie-et-non-invasive>

### 8.1 Ensembles de données :

Les ensembles de données CelebA et RaFD sont détaillés dans ce chapitre :

1. Le jeu de données CelebA contient 202,599 images de visages de célébrités, chacune annotée de 40 attributs binaires, par exemple, couleur des cheveux (noir, blond, brun), sexe (homme/femme) et âge (jeune/vieux).
2. Le jeu de données RaFD se compose de 4,824 images recueillies auprès de 67 participants, qui ont exprimé huit expressions faciales dans trois directions et angles de regard différents

### 8.2 Paramètres de perte

Dans cette partie, les fonctions de perte du GAN et de l'U-Net sont détaillées.

#### 8.2.1 Score de dice

Le coefficient de similarité (ou score des dés) va de 0 à 1 ou de 0 à 100%. Il existe situations pour le score dice :

1. S'il n'y a pas de correspondance entre l'image de référence (telle que fournie par un humain) et l'image prédite, le score dice est de 0 ou 0%.
2. Dans les cas où il y a un petit chevauchement, le score dice sera un petit nombre, environ 0,1 ou 10% dans ce cas.
3. Dans le cas où elles correspondent exactement, le score dice est de 1,0 ou 100%.

Le coefficient dice est défini comme suivant :

$$\text{Score de dice} = \frac{2 * \text{la zone de correspondance}}{\text{nombre total de pixels dans les deux images}} \quad (9)$$

Par conséquent, la perte de dice compare le résultat de la segmentation avec la carte des étiquettes de tumeurs, qui est ensuite utilisée pour optimiser le générateur par rétropropagation.

#### 8.2.2 Pénalité de gradient L1

Pour stabiliser le processus d'apprentissage et améliorer la généralisation du modèle GAN, la fonction objective et la pénalité de gradient proposées dans les expériences, elle est définie comme dans [27] :

$$\min_G \max_D \mathcal{L}(G, D_{\text{src}}) = E_{x,y}[\log(D_{\text{src}}(x,y))] + E_x[\log(1 - D_{\text{src}}(x, G(x)))] + \lambda_{L1} \mathcal{L}_{L1}(G(x), y), \quad (10)$$

$$\mathcal{L}_{\text{gp}} = E_{x,\hat{x}}[(\|\nabla \hat{D}_{\text{src}}(x, \hat{x})\|_2 - 1)^2], \quad (11)$$

où les deux premiers éléments de l'équation (10) sont des pertes adverses. De plus,  $L_{l1}$  est L1 perte pour assurer la similarité en pixels entre les images synthétisées et les images réelles.  $\lambda_{l1}$  est un hyperparamètre pour équilibrer son poids. Le pix2pix est optimisé par un jeu à somme

nulle [13], ce qui permet finalement au G de générer des échantillons réalistes. Dans le processus d'apprentissage, le générateur G essaie de générer des échantillons  $G(x)$  ressemblant à des échantillons réels  $y$  pour tromper le discriminateur  $D_{src}$ . Ensuite,  $D_{src}$  essaie à distinguer les échantillons synthétisés des échantillons réels.  $\hat{x}$  est échantillonné uniformément le long d'une ligne droite entre une paire d'une image réelle et une image synthétisée.

La fonction objective finale du discriminateur est définie comme suit :

$$\mathcal{L}_D = -\mathbb{E}_{x,y,c}[D_{src}(x, y_c)] + \mathbb{E}_{x,c}[D_{src}(x, G(x, c))] + \lambda_{cls}\mathcal{L}_{cls}^r + \lambda_{gp}\mathcal{L}_{gp}, \quad (12)$$

La fonction objective finale du générateur est définie comme suit :

$$\mathcal{L}_G = -\mathbb{E}_{x,c}[D_{src}(x, G(x, c))] + \lambda_{cls}\mathcal{L}_{cls}^r + \lambda_{L1}\mathcal{L}_{L1} + \lambda_{seg}\mathcal{L}_{seg}, \quad (13)$$

où les hyper-paramètres  $\lambda_{cls}$ ,  $\lambda_{gp}$ ,  $\lambda_{L1}$ ,  $\lambda_{seg}$  sont utilisés pour équilibrer les différents termes. Pendant chaque itération d'apprentissage, l'étiquette de modalité cible  $c$  générée aléatoirement pour le générateur G afin de synthétiser la modalité cible correspondante et pour le réseau de segmentation S afin de segmenter la zone tumorale pour la modalité synthétisée.

### 8.2.3 Segmentation

Le CC tumoral multi-modalité est défini comme suit :

$$\mathcal{L}_{seg} = \mathbb{E}_{x,c}[S(G(x, c), l)], \quad (14)$$

où  $l$  est la vérité de base de la carte de segmentation des tumeurs. Il convient de noter que chaque modalité reflète naturellement des caractéristiques distinctes de l'anatomie humaine. S n'est pas proposé pour forcer la modalité cible  $x$  à avoir la même apparence de tumeur que la modalité source  $c$ , mais pour contraindre le générateur G à se concentrer sur l'apprentissage de la cartographie multimodale dans la zone tumorale.

### 8.2.4 Perte de classification

Pour permettre au générateur G de synthétiser n'importe quelle modalité cible, nous introduisons des étiquettes de modalité  $c$  à l'entrée du générateur, ce qui nous permet de spécifier la modalité des images synthétisées en ajustant les valeurs de  $c$ . Nous concaténons l'étiquette de modalité  $c$  à l'image source  $x$  comme entrée du générateur, de sorte que l'image synthétisée est  $G(x, c)$ . Pour que l'étiquette de modalité  $c$  puisse prendre effet dans le générateur G, nous avons besoin du discriminateur D non seulement pour distinguer les vraies images des fausses images, mais aussi pour classer la modalité de l'image d'entrée. À cette fin, la perte de classification de modalité est définie comme suit :

$$\mathcal{L}_{cls}^r = \mathbb{E}_{x,y,c}[-\log D_{cls}(c|(x, y_c))], \quad (15)$$

$$\mathcal{L}_{cls}^f = \mathbb{E}_{x,c}[-\log D_{cls}(c|(x, G(x, c)))], \quad (16)$$

où l'exposant  $r$  représente l'entrée de l'image réelle et  $f$  désigne une image fausse en entrée.

### 8.2.5 Perte de reconstruction

La perte de reconstruction aide le réseau à produire l'image réaliste proche de l'image  $G$ , définie comme elle est présentée dans [13]:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,c}[||y - G(x, c)||_1] \quad (17)$$

L1 La perte agit comme un terme de régularisation, pénalisant le générateur si la qualité de reconstruction de l'image traduite n'est pas similaire à celle de l'image cible.

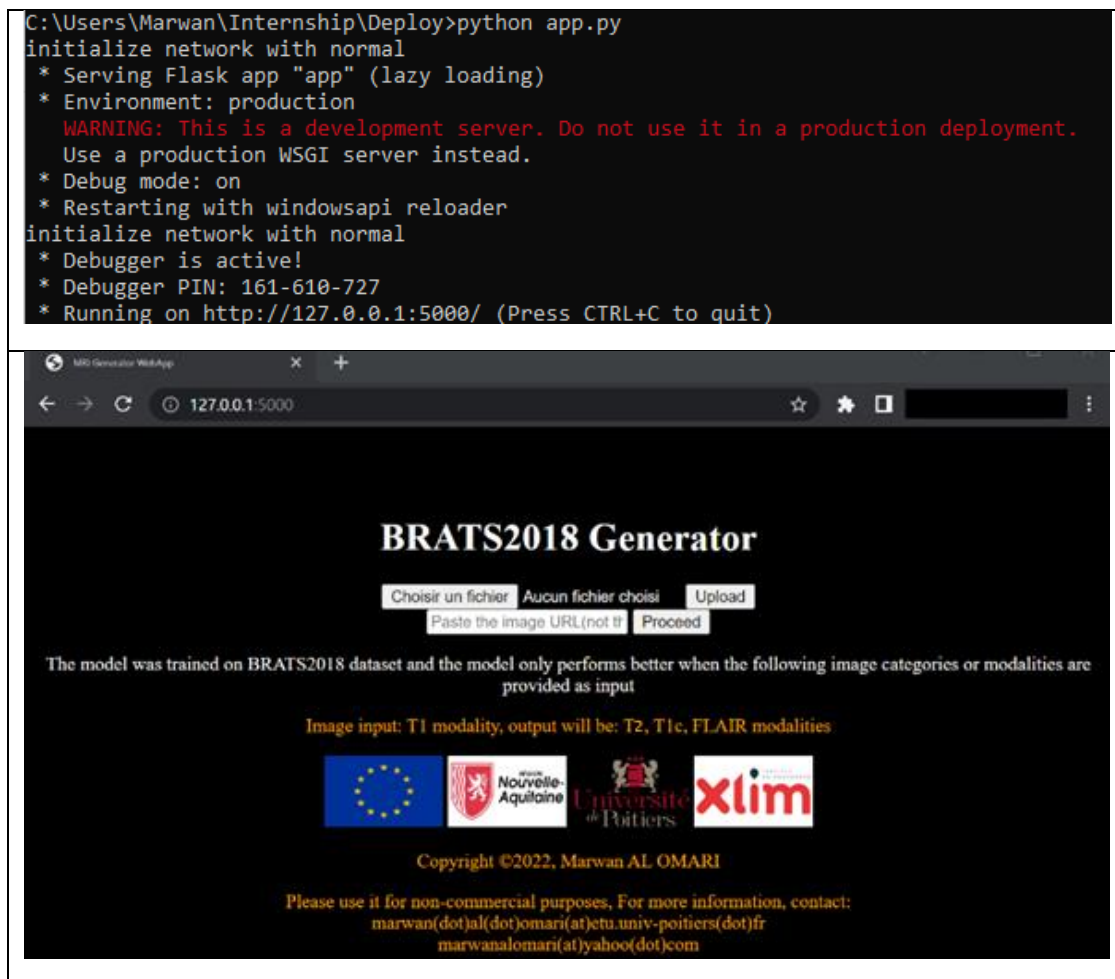
### 8.3 Deep learning operations (DLOps)

Dans cette partie, une présentation du modèle SIMO est présentée dans ce qui suit. Le modèle était un petit prototype entraîné sur 225 images au total, donc les résultats sont flous et ne sont pas de haute qualité. Par conséquent, il ne reflète pas les résultats finaux du projet, mais il est sujet à des recherches expérimentales dans le futur.

#### 8.3.1 Modèle de SIMO

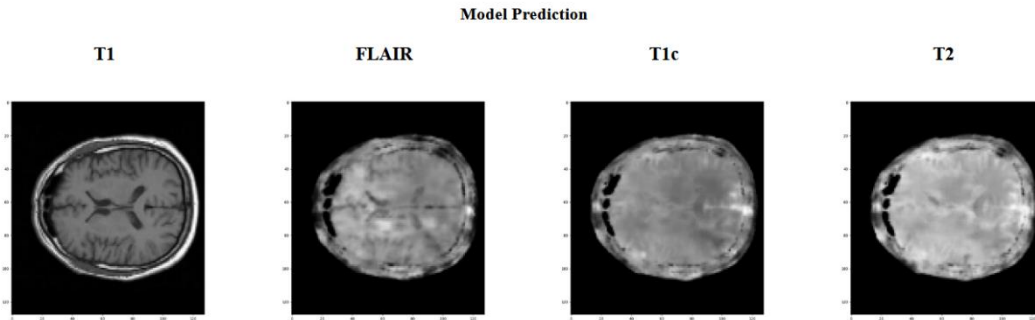
Comme la plupart des recherches s'arrêtent au développement de l'algorithme, un prototype du modèle SIMO est déployé en tant que microservice, en utilisant la framework de flask en Python. Le modèle déployé est accessible sur le serveur local de la machine pour le débogage sur 127.0.0.1: 5000, comme le montre la figure (43).

**Figure 43-** Le modèle SIMO, en tant que microservice, accessible sur le service local pour le débogage 127.0.0.1: 5000



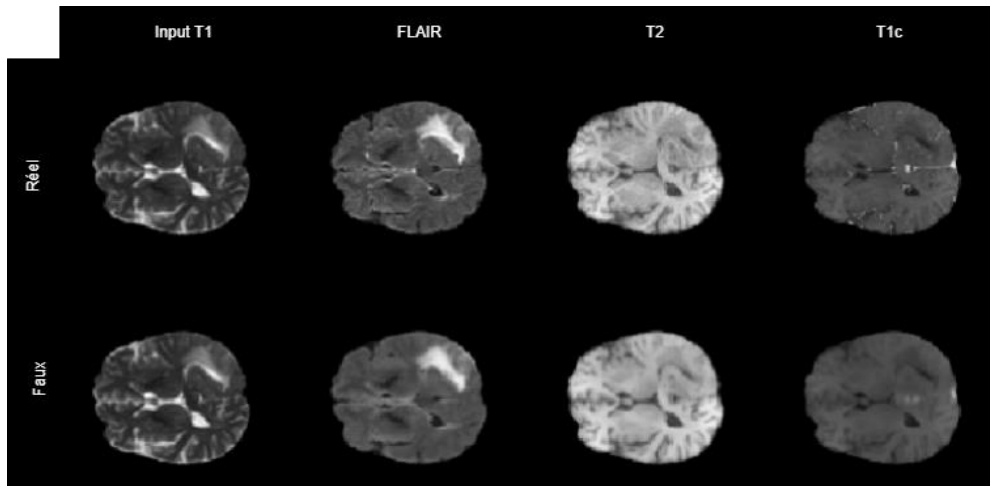
Comme nous pouvons le voir dans la fenêtre qui s'ouvre dans le navigateur Internet, l'application est chargée et prête à être utilisée. En prenant une image T1 comme entrée pour le modèle, la sortie est respectivement présentée dans la figure suivante (44).

**Figure 44-** Les résultats du modèle déployé SIMO en prenant un exemple d'image d'internet



En prenant plusieurs images en format numpy, on peut avoir des meilleurs résultats comme à déjà présenter dans la figure (45) :

**Figure 45-** Les résultats du modèle déployé SIMO en prenant un exemple réel de BRATS2018



## 8.4 Etat de l'art

Dans cette section, les travaux de recherches sont expliqués en détail par donner tous les paramétrages de modèles appliqué pour des recherches qui veulent appliquer les idées de chaque algorithme. Un regroupement des informations représentées dans l'état de l'art représenté dans le tableau (20) à la fin ci-dessous ;

Leur réseau multimodal a été évalué sur quatre contrastes de T1, T1c, T2 et FLAIR en calculant *normalized mean absolute error* (NMAE), PSNR, *structural similarity index measurement* (SSIM), *visual information fidelity* (VIF), *naturalness image quality evaluator* (NIQE) [3]. Ils ont adopté la stratégie StarGAN [2] pour cartographier les quatre modalités d'imagerie par IRM à travers 12 modèles SISO, et un GAN unifié pour les cartographier ensemble. Leur méthode a été évaluée sur BRATS2015, qui comprend 274 sujets, dont 54 patients atteints de LGG et 220 patients atteints de HGG. La taille originale de chaque image de l'ensemble de données est de  $240 \times 240 \times 155$ . Pour le prétraitement, chaque image est réduite à une taille de patch de  $72 \times 72 \times 72$ . Ensuite, l'estimation finale des régions superposées a été fixée à  $48 \times 48 \times 48$  avec une validation croisée 5 fois. La plupart des paramètres importants de leur modèle sont



fixés à  $\alpha = 2$ ,  $\beta = 5$ ,  $\gamma = 10$ ,  $\delta = 2$  et  $\mu = 0,1$  pour équilibrer les poids de la classification, de la cohérence synthétique, de la CC et des pertes adverses, respectivement. En conclusion, par exemple, avec T1 comme modalité d'entrée, les NMAEs pour les T1c, T2, FLAIR générés sont  $0.034 \pm 0.005$ ,  $0.041 \pm 0.006$ , et  $0.041 \pm 0.006$ , les PSNRs sont  $32.353 \pm 2.525$  dB,  $30.016 \pm 2.577$  dB, et  $29.091 \pm 2.795$  dB, les SSIM sont de  $0,974 \pm 0,059$ ,  $0,969 \pm 0,059$ , et  $0,959 \pm 0,059$ , les VIF sont de  $0,750 \pm 0,087$ ,  $0,706 \pm 0,097$ , et  $0,654 \pm 0,062$ , et les NIQE sont de  $1,396 \pm 0,401$ ,  $1,511 \pm 0,460$ , et  $1,259 \pm 0,358$ , respectivement. Enfin, leur étude est limitée aux images multimodales spatialement co-enregistrées avant même qu'elles ne soient utilisées pour la formation et le test sur un petit ensemble de données. Pour les travaux futurs, ils essaieront d'augmenter la quantité d'images d'entraînement et de test par des techniques d'augmentation.

StarGAN [2], un GAN unifié, gère les traductions multi-domaines d'image à image en utilisant un modèle unique de discriminateur et de générateur un-à-un. Un modèle unique prend en entrée une image et des informations sur le domaine par codage à un coup. Il apprend les correspondances entre tous les domaines disponibles, en utilisant un seul générateur. Le processus d'apprentissage génère une étiquette de domaine cible aléatoire afin d'assurer une traduction parfaite de l'image. En raison des limites des modèles uniques qui ne pouvaient pas être généralisés à plus d'une tâche individuelle, le chercheur a proposé une nouvelle approche qui pourrait gérer plusieurs domaines en utilisant un seul générateur et un discriminateur. Leur approche fonctionne bien sur les tâches de transfert de caractéristiques faciales et de synthèse d'expressions faciales. Leur architecture a été testée sur deux jeux de données. Tout d'abord, CelebA contient 40 étiquettes liées à des caractéristiques faciales telles que la couleur des cheveux, le sexe et l'âge. Deuxièmement, le jeu de données RaFD contient 8 étiquettes pour les expressions faciales telles que la joie, la colère et la tristesse. En outre, pour éviter les valeurs manquantes dans le jeu de données, ils ont appliqué un vecteur de masque de l'étiquette du domaine qui permet d'ignorer les étiquettes inconnues et de se concentrer sur ce qui est déjà disponible. Sur CelebA, StarGAN a obtenu 66.2%, 39.1%, 70.6%, 47.4%, 61.5%, 49.8%, 52.2% pour la couleur des Cheveux, le Sexe, l'Âge, C+S, C+A, S+A, et C+S+A, respectivement. Sur RaFD : StarGAN a atteint 2,12 pertes avec  $53,2M \times 1$  paramètres. Sur CelebA et RaFD, le modèle a correctement appris le rôle prévu d'un vecteur de masque dans les traductions d'image à image lorsque toutes les étiquettes proviennent de plusieurs ensembles de données. En conclusion, StarGAN a généré des images de meilleure qualité visuelle que les méthodes existantes.

Pix2pix [13], un générateur de U-Net et un discriminateur de CGANs, combine une perte contradictoire avec une perte L1 pour capturer les basses fréquences. Progressivement, leur modèle sous-échantillonne les images d'entrée à partir d'une grille haute résolution vers une couche de connecteurs de saut (concaténation des canaux des couches précédentes) suivie de U-Net. Leur algorithme s'est avéré efficace, tout comme PatchGAN (qui exécute des patches pouvant capturer les hautes fréquences de la structure de l'image). Pour l'optimisation de l'algorithme, ils ont utilisé des minibatches stochastic gradient descent (SGD) et adaptive moment estimation (ADAM), avec un TA de 0,0002, et des paramètres de momentum  $\beta_1 = 0,5$ ,  $\beta_2 = 0,999$ . Ils ont également utilisé diverses normalisations de lots, allant de 0 à 10. Ils ont entraîné leurs CGAN sur différentes tâches et ensembles de données, y compris des Cityscapes, des façades GMP, des photos Google Maps, des photos en couleur et *black and white* (BW), des bords de photos, des esquisses dessinées par l'homme, des images de jour et



de nuit, des images thermiques et en couleur, et enfin des photos à pixels manquants et des photos non peintes. Ils ont déclaré que même avec une petite taille d'entraînement, ils ont été en mesure d'obtenir un résultat de descente sur un seul GPU Pascal Titan X. Comme une limitation pour mesurer les pertes de structure en appliquant des mesures traditionnelles comme le mean-squared error (MSE) par pixel, ils ont appliqué une méthode conjointe. Premièrement, un test de génération de cartes, de photos aériennes et de colorisation d'images a été effectué pour résoudre des problèmes de graphes sur l'Amazon mechanical turk (AMT). Ensuite, un système métrique a été utilisé pour reconnaître la correspondance d'objets réalistes dans les images. En conclusion, la perte de L1, le CGAN et L1+CGAN ont atteint sur les Cityscapes 86%, 76% et 83% d'ACC par pixel, et 42%, 28% et 36% d'ACC par classe, respectivement.

UNIT [14] combine les *variational autoencoders* (VAEs) avec CoCAN, où deux générateurs partagent des poids pour apprendre la distribution des images dans un domaine croisé. Le partage de l'espace latent implique le mappage CC entre les domaines source et cible, de manière interchangeable. Ils ont utilisé la contrainte de partage des poids pour relier les poids des dernières couches des VAE, extraites des hautes fréquences codées, pour le décodage. Ils ont utilisé ADAM avec 0,0001 TA et 0,5 à 0,999 momentum et un batch size d'une image de différents domaines. Leur cadre a différents paramètres de  $\lambda_0 = 10$ ,  $\lambda_3 = \lambda_1 = 0.1$  et  $\lambda_4 = \lambda_2 = 100$ . Les codeurs étaient constitués de 3 couches convolutionnelles et de 4 blocs résiduels de base. D'autre part, les générateurs étaient constitués de 4 blocs résiduels de base comme frontal et de 3 couches convolutives transposées en back-end. D'autre part, les discriminateurs étaient constitués de piles de couches convolutionnelles avec LeakyReLU non linéaire. Ils ont utilisé un ensemble de données cartographiques, qui contenait des paires d'images correspondantes dans deux domaines (images satellites et cartes). Pour 100K itérations, l'expérience a fonctionné dans un cadre non supervisé en utilisant 1096 images satellites de l'ensemble de formation comme premier domaine et 1098 cartes de l'ensemble de validation comme second domaine. Les cartes de vérité du sol correspondantes ont été traitées pixel par pixel, chaque translation de pixel étant considérée comme correcte uniquement si la différence de couleur était inférieure à 16 des valeurs de couleur de la vérité du sol. Comme mesure de performance, ils ont utilisé la moyenne des pixels ACC. L'algorithme a obtenu des précisions de classification : SVHN→MNIST 0,9053, MNIST→USPS 0,9597, USPS→MNIST 0,9358.

DiscoGAN [15] et CycleGAN [16] sauvegardent les attributs d'information entre les images d'entrée et les images traduites en utilisant une perte CC. Cependant, ils souffrent tous d'une généralisation limitée car il s'agit de modèles SISO.

D'une part, DiscoGAN, un GAN multi-domaine sans aucun ensemble de pré-entraînement explicite, prend une image d'entrée dans un domaine et génère celle qui lui correspond. Ils ont proposé des GAN couplés en forçant les images générées par une perte de reconstruction à être un double de l'image originale, par exemple, une image de sac à main. Dans les expériences, toutes les images d'entrée et traduites sont de taille  $64 \times 64 \times 3$ . Certains des paramètres adoptés sont 0.0002 TA, 200 minibatches, et l'optimiseur ADAM avec  $\beta_1 = 0.5$  et  $\beta_2 = 0.999$ . En outre, la normalisation par lots a été appliquée à toutes les couches de convolution et de déconvolution, à l'exception de la première et de la dernière couche. De plus, le coefficient de décroissance du poids varie entre 10 – 4. Les images de données varient en rotation azimutale de  $-90^\circ$  à  $+90^\circ$ . Leurs expériences se résument à des ensembles de données de voiture à voiture et de visage à visage. L'ensemble de données sur les voitures est constitué

d'images rendues de modèles de voitures en 3D avec des angles d'azimut variables à intervalles de  $15^\circ$ . Pour l'évaluation, ils ont traduit les images dans l'ensemble de test et leurs angles d'azimut ont été prédits en utilisant un régresseur. Le modèle proposé montre une forte corrélation entre les angles prédits des images d'entrée et des images traduites, ce qui permet de découvrir la relation d'azimut entre deux domaines. En termes de rotations entre un GAN standard et un GAN à perte reconstructive, les images générées ne varient pas autant que les images d'entrée. Lors d'une expérience sur le jeu de données de voitures, les deux modèles ont souffert d'un effondrement soudain. En outre, ils ont expérimenté la conversion de visage par attributs de visage sur les jeux de données CELEBA, facescrub, où une seule caractéristique, comme le sexe ou la couleur des cheveux, varie entre deux domaines et des domaines partagés. Les résultats traduits ont non seulement des couleurs et des motifs similaires, mais ils ont également un niveau de formalité de mode similaire à celui de l'article de mode d'entrée. Les travaux futurs porteront sur la modification de l'algorithme pour le traitement des modalités mixtes (par exemple, texte et image).

D'autre part, CycleGAN, utilise deux pertes CC qui capturent la translation d'image d'un domaine à l'autre et vice versa. L'architecture contient trois convolutions, des blocs résiduels, deux convolutions à stride fractionné avec  $\frac{1}{2}$  stride, et une convolution qui fait correspondre la caractéristique à l'espace RVB. Leur réseau contient 6 blocs pour les images  $128 \times 128$  et 9 blocs pour les images  $256 \times 256$ . Pour le discriminateur, ils ont utilisé des PatchGAN  $70 \times 70$  avec normalisation des instances. Ils ont utilisé différentes techniques de LGAN et d'oscillation pour stabiliser leur modèle. Premièrement, LGAN remplace l'objectif de log like-hood négatif par la perte des moindres carrés. Deuxièmement, une mise à jour de l'oscillation du discriminateur utilise un historique des images générées plutôt que celles produites par les derniers générateurs. Certains paramètres importants sont  $\lambda = 10$ , 0.0002 ADAM TA, un batch size de 1, et une décroissance linéaire vers zéro sur 100 époques. Les expériences ont donné des résultats positifs, mais ont échoué dans certaines tâches de traduction impliquant des changements de couleur, de géométrie et de texture, par exemple, un chat en chien ou en cheval, et un cavalier sur un cheval en zèbre. De plus, une faiblesse sémantique a été repérée dans les tâches d'étiquetage de photos, et l'ambiguïté de la traduction a donc formé des limites importantes. En conclusion, le modèle a obtenu des étiquettes réelles de  $26,8\% \pm 2,8\%$  pour la carte vers la photo, et de  $23,2\% \pm 3,4\%$  pour la photo vers la carte. En outre, il a obtenu 58% de précision par pixel, 22% de précision par classe et 16% de reconnaissance de classe moyenne *intersection-over-union* (IOU) pour les étiquettes de photo sur le jeu de données Cityscapes.

**Tableau 20-** Résumé des articles en ce qui concerne le numéro de l'étude, le jeu de données, l'approche utilisée, les résultats de l'expérience, les défis et les travaux futurs possibles.

Étude	Ensembles de données	Approche	Résultats	Problèmes et défis	Travaux futurs
[3]	<b>Description :</b> BRATS2015 a 274 sujets de 4 modalités : T1, T1c, T2, et FLAIR. La taille de chaque image : $240 \times 240 \times 155$ <b>Pré-traitement :</b>	<b>Modèles :</b> - 12 modèles SISO et un réseau GAN unifié  <b>Configurations :</b> - Paramètres de perte : ■ $\alpha = 2$	T1 en entrée : - NMAEs : ■ $0.034 \pm T1c$ ■ $0.041 \pm T2$ ■ $0.041 \pm FLAIR$ - PSNRs : ■ $32.35 \pm T1c$ ■ $30.016 \pm T2$ ■ $29.09 \pm FLAIR$	- Images multimodales spatialement co-enregistrées  - Petit jeu de données	- Techniques d'augmentation des données

Étude	Ensembles de données	Approche	Résultats	Problèmes et défis	Travaux futurs
	<ul style="list-style-type: none"> <li>72 × 72 × 72 images</li> <li>48 × 48 × 48 Régions</li> </ul> <p><b>Ensemble de validation :</b></p> <ul style="list-style-type: none"> <li>5-fold CV</li> </ul>	<ul style="list-style-type: none"> <li><math>\beta = 5</math></li> <li><math>\gamma = 10</math></li> <li><math>\delta = 2</math></li> <li><math>\mu = 0.1</math></li> </ul> <p>- Mesures d'évaluation :</p> <ul style="list-style-type: none"> <li>NMAE</li> <li>PSNR</li> <li>SSIM</li> <li>VIF</li> <li>NIQE</li> </ul>	<p>- SSIMs :</p> <ul style="list-style-type: none"> <li><math>0.974 \pm T1c</math></li> <li><math>0.969 \pm T2</math></li> <li><math>0.959 \pm FLAIR</math></li> </ul> <p>- VIF :</p> <ul style="list-style-type: none"> <li><math>0.750 \pm T1c</math></li> <li><math>0.706 \pm T2</math></li> <li><math>0.65 \pm FLAIR</math></li> </ul> <p>- NIQE :</p> <ul style="list-style-type: none"> <li><math>1.396 \pm T1c</math></li> <li><math>1.511 \pm T2</math></li> <li><math>1.259 \pm FLAIR</math></li> </ul>		
[2]	<p><b>Description :</b></p> <ul style="list-style-type: none"> <li>CelebA contient 40 étiquettes de caractéristiques faciales, par exemple la couleur des cheveux, etc.</li> <li>RaFD contient 8 étiquettes d'expressions faciales, par exemple, heureux, etc.</li> </ul> <p><b>Pré-traitement :</b></p> <ul style="list-style-type: none"> <li>Vectorisation des étiquettes</li> </ul>	<p><b>Modèles :</b></p> <ul style="list-style-type: none"> <li>Un GAN unifié composé d'un discriminateur et d'un générateur multiples un-à-un</li> </ul> <p><b>Configurations :</b></p> <ul style="list-style-type: none"> <li>Non spécifié</li> </ul>	<p>- ACC CelebA :</p> <ul style="list-style-type: none"> <li>66.2% Cheveux</li> <li>39.1% Sexe</li> <li>70.6% Âgés</li> <li>47.4% C+S</li> <li>61.5% C+A</li> <li>49.8% S+A</li> <li>52.2% C+S+A</li> </ul> <p>- Loss RaFD :</p> <ul style="list-style-type: none"> <li>2.12</li> </ul>	Non spécifié	Non spécifié
[13]	<p><b>Description :</b></p> <ul style="list-style-type: none"> <li>Cityscapes</li> <li>Façades GMP</li> <li>Google Maps</li> <li>BW à la couleur</li> <li>Croquis d'Humandrawn</li> <li>Jour à nuit</li> <li>Thermique vers couleur</li> <li>Pixels manquants</li> <li>Top inpainted</li> </ul> <p><b>Pré-traitement :</b></p> <ul style="list-style-type: none"> <li>Génération de mappage</li> <li>Colorisation d'images</li> <li>Système métrique</li> </ul>	<p><b>Modèles :</b></p> <ul style="list-style-type: none"> <li>U-Net</li> <li>CGANs</li> </ul> <p><b>Configurations :</b></p> <ul style="list-style-type: none"> <li>Pertes adverses &amp; L1</li> <li>Déséchantillonnage</li> <li>Minibatch</li> <li>0.0002 TA</li> <li>SGD &amp; ADAM</li> <li>0.5 <math>\beta_1</math> &amp; 0.9 <math>\beta_2</math></li> <li>Momentum</li> <li>0 – 10 batch size</li> </ul>	<p>Perte de L1, CGAN, &amp; L1+CGAN ACC sur Cityscapes :</p> <p>- Par-pixel :</p> <ul style="list-style-type: none"> <li>86%</li> <li>76%</li> <li>83%</li> </ul> <p>- Par-class :</p> <ul style="list-style-type: none"> <li>42%</li> <li>28%</li> <li>36%</li> </ul>	<p>- Petit jeu de données</p> <p>- Pertes structurelles des mesures traditionnelles en tant que MSE par pixel</p>	Non spécifié
[14]	<p><b>Description :</b></p> <p>Le jeu de données cartographiques contient des paires d'images dans deux domaines : les images satellites et les cartes.</p> <p><b>Ensemble d'entraînement :</b></p>	<p><b>Modèles :</b></p> <ul style="list-style-type: none"> <li>VAEs</li> <li>CoCAN</li> </ul> <p><b>Configurations :</b></p> <ul style="list-style-type: none"> <li>Partage du poids</li> <li>0.0001 TA</li> <li>0.5 – 0.999 momentums</li> <li>ADAM</li> </ul>	<p>Classification ACC :</p> <ul style="list-style-type: none"> <li>SVHN→MNIST: 0.9053</li> <li>MNIST→USPS: 0.9597</li> <li>USPS→MNIST: 0.9358</li> </ul>	Non spécifié	Non spécifié

Étude	Ensembles de données	Approche	Résultats	Problèmes et défis	Travaux futurs
	1096 images satellites comme premier domaine <b>Ensemble de validation :</b> 1098 cartes en tant que second domaine	<ul style="list-style-type: none"> <li>- LeakyReLU</li> <li>- 1 batch size</li> <li>- Traduction des pixels</li> <li>- 100K itérations</li> <li>- Moyenne de pixels ACC</li> <li>- Perte de CC</li> <li>- Paramètres de perte : <ul style="list-style-type: none"> <li>▪ <math>\lambda_0 = 10</math></li> <li>▪ <math>\lambda_3 = \lambda_1 = 0.1</math></li> <li>▪ <math>\lambda_4 = \lambda = 100</math></li> </ul> </li> </ul>			
[15]	<b>Description :</b> <ul style="list-style-type: none"> <li>- Données sur les voitures</li> <li>- CELEBA</li> <li>- Facescrub</li> </ul> <b>Pré-traitement :</b> <ul style="list-style-type: none"> <li>- <math>64 \times 64 \times 3</math> images</li> <li>- Rotation en azimut de <math>-90^\circ</math> à <math>+90^\circ</math></li> </ul>	<b>Modèles :</b> <ul style="list-style-type: none"> <li>▪ GANs couplés</li> </ul> <b>Configurations :</b> <ul style="list-style-type: none"> <li>- 0.0002 TA</li> <li>- Perte de reconstruction</li> <li>- 200 minibatch</li> <li>- Normalisation de batch</li> <li>- 10 – 4 dégradation du poids</li> <li>- ADAM optimiseur : <ul style="list-style-type: none"> <li>▪ <math>\beta_1 = 0.5</math></li> <li>▪ <math>\beta_2 = 0.999</math></li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>- Découverte de la relation azimutale entre 2 domaines</li> </ul>	<ul style="list-style-type: none"> <li>- Limitation de généralisation du modèle SISO</li> <li>- Effondrement soudain du modèle</li> </ul>	<ul style="list-style-type: none"> <li>- Algorithme modification dans le traitement des modalités mixtes (par exemple, texte et image)</li> </ul>
[16]	<b>Description :</b> <ul style="list-style-type: none"> <li>- Cityscapes</li> <li>- Carte vers Photo</li> </ul> <b>Pré-traitement :</b> <ul style="list-style-type: none"> <li>- <math>70 \times 70</math> images</li> <li>- <math>128 \times 128</math> images</li> <li>- <math>256 \times 256</math> images</li> </ul>	<b>Modèles :</b> <ul style="list-style-type: none"> <li>- PatchGANs</li> <li>- 3 convolutions</li> <li>- Blocs résiduels <ul style="list-style-type: none"> <li>▪ <math>6 \ 128 \times 128</math> blocs</li> <li>▪ <math>9 \ 256 \times 256</math> blocs</li> </ul> </li> <li>- 2 convolutions fractionnelles</li> <li>- <math>\frac{1}{2}</math> stride</li> <li>- Convolution RVB</li> </ul> <b>Configurations :</b> <ul style="list-style-type: none"> <li>- Instance normalisation</li> <li>- Pert de CC</li> <li>- Perte de least-squares</li> <li>- 0.0002 TA</li> <li>- <math>\lambda = 10</math></li> <li>- 1 batch size</li> <li>- Dégradation linéaire sur 100 époques</li> </ul>	<ul style="list-style-type: none"> <li>- Carte vers Photo : <ul style="list-style-type: none"> <li>▪ <math>26.8\% \pm 2.8\%</math></li> </ul> </li> <li>- Photo vers Carte : <ul style="list-style-type: none"> <li>▪ <math>23.2\% \pm 3.4\%</math></li> </ul> </li> <li>- Cityscapes : <ul style="list-style-type: none"> <li>▪ 0.58 par-pixel ACC</li> <li>▪ 0.22 par-class ACC</li> <li>▪ 0.16 classe moyenne IOU</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>- Limitation de généralisation du modèle SISO</li> <li>- Faiblesse sémantique entraînant une ambiguïté de traduction</li> <li>- Échec des traductions sur les changements de couleur, de géométrie et de texture</li> </ul>	Non spécifié