REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre :04 /IA/M2/2022

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **L'intelligence Artificielle (IA)**

La segmentation des images médicales par la Symétries et CNN

Par :

BENBRAHIM SALIHA

Soutenu le 21/06/2023 devant le jury composé de :

Hiouani Rima	MAA	Président
Hamida Ammar	MAA	Rapporteur
Bettira Roufaida	MAA	Examinateur

Année universitaire 2022-2023

Remerciements

Avant tout, je sincère louange à ALLAH le tout puissant qui ma donné la fois, la volonté, la santé et la patience pour accomplir ce travail.

Je tiens à exprimer ma sincère gratitude à mes parents qui ont contriburéalisation de cette mémoire. Leur soutien, leurs conseils et leurs encouragements ont été d'une valeurinestimable tout au long de ce parcours.

Je tiens à remercier chaleureusement mon superviseur, [**Hamida Ammar**], pour sa précieuse orientation, son expertise et sa disponibilité. Sa guidance éclairée a été un moteur essentiel dans l'élaboration de ce travail de recherche.

Mes remerciements vont également à tous les **membres de mon jury** qui ont consacré leur temps et leur expertise à l'évaluation de cette mémoire.

Ensuite, je tien remercier tous les enseignants d'epartement informatique , j'ai beaucoup appris d'eux.

Ennfin , Je n'oublie pas de remercier ma famille et mes amis, qui m'ont soutenu de manière inconditionnelle tout au long de cette aventure. Leurs encouragements, leur amour et leur soutien moral m'ont permis de surmonter les obstacles et de persévérer dans mes efforts.

Abstract

Recently, artificial intelligence (AI) has dominated all areas of scientific research because of the solutions it offers. The health industry is no exception, The tumor is an abnormal mass of cells that can be benign or malignant, often requiring appropriate medical treatment. Image segmentation based on deep learning is now a reliable tool. It has been widely used as the first critical element in the diagnostic and treatment chain to separate homogeneous areas.

The objective of this work is to provide an epidemiological approach of deep learning and symmetry for the identification of brain tumors and also The problem is that only standard CNN models use translational invariance, ignoring the more realistic symmetries found in medical images, such as rotations and reflexes. To solve this problem, we have worked to implement an equally segmented training model, encoding the inherent symmetries to produce more accurate representations.

In this work, we built a Unet model and use the symmetric methods of rotation and reflection ; the created model were initiated and proved their effectiveness by achieving high accuracy and F1 score in their tests using the 2019 BraTS data.

Keywords : Image segmentation, convolutive neural network, artificial intelligence, brain tumors.

Résumé

Récemment, l'intelligence artificielle (IA) a dominé tous les domaines de la recherche scientifique en raison des solutions qu'elle offre. L'industrie de la santé n'est pas une exception, La tumeur est une masse anormale de cellules qui peut être bénigne ou maligne, nécessitant souvent un traitement médical approprié.La segmentation des images basée sur le deep learning est désormais un outil fiable. Il a été largement utilisé comme premier élément critique de la filière de diagnostic et de traitement pour séparer les zones homogènes.

L'objectif de ce travail est de fournir une approche épidémiologique d'apprentissage profond et symétrie pour l'identification des tumeurs cérébrales et aussi Le problème est que seuls les modèles standard de CNN utilisent l'invariance translationnelle, ignorant les symétries plus réalistes trouvées dans les images médicales, telles que les rotations et les réflexes. Pour résoudre ce problème, nous avons travaillé à mettre en œuvre un modèle d'entraînement à segmentation égale, encodant les symétries inhérentes pour produire des représentations plus précises.

Dans ce travail ,nous avons construit un modéle Unet et utilise les méthodes de symétries soit la rotation et réflexion ;le modéle créé ont été initiés et ont prouvé leur efficacité en atteignant une précision élevée et un score F1 dans leurs tests à l'aide de données BraTS 2019.

Mots-clés : Segmentation d'images, réseau de neurones convolutifs, intelligence artificielle ,tumeurs cérébrales.

Table des matières

Table des figures

Liste des code sources

Introduction générale

Segmentation d'image consiste à diviser l'image en régions simples d'une homogénéité particulière. En d'autre terme ;C'est la division d'une image en groupes uniformes de pixels (ou voxels dans le cas d'images 3D) selon un critère prédéterminé[10]. Les résultats de la segmentation varient en fonction des algorithmes utilisés, des critères d'homogénéité et de l'initialisation. Dans le domaine de la médecine, l'utilisation d'algorithmes de segmentation d'images médicales est cruciale pour obtenir des informations précises à partir d'images de diagnostic comme les radiographies, les scanners et l'IRM. La capacité de délimiter et d'isoler des régions anatomiques importantes grâce à la segmentation facilite la détection des maladies, l'évaluation des traitements et la planification chirurgicale.

L'une des méthodes les plus utilisées pour la segmentation de l'image médicale est l'application des réseaux de neurones convolutifs (CNN). Les CNN sont des modèles d'apprentissage automatique qui se sont avérés efficaces dans une variété de tâches de vision assistée par ordinateur, y compris la segmentation d'image.

La segmentation d'images médicales par CNN peut être améliorée en exploitant les symétries présentes dans les images anatomiques. Les symétries, telles que la symétrie bilatérale des organes, peuvent fournir des informations supplémentaires pour guider la segmentation et améliorer sa précision. Les CNN peuvent être adaptés pour prendre en compte ces symétries, en utilisant des architectures spécifiques ou en incorporant des techniques de traitement d'image qui exploitent les symétries.

Objectif du travail

L'objectif de ce projet est d'apprendre davantage sur certaines techniques d'apprentissage profond utilisées et d'appliquer la méthode de symétrie pour ségmentation l'image médicale, de choisir la structure d'apprentissage profond, la mise en œuvre et l'application à l'ensemble de données BraTS19.

Dans ce context, le mémoire est organisé comme suit :

— **Chapitre 1 : Diagnostique des tumeurs cérébrales et L'imagerie médicale.** Dans ce chapitre, nous avons traité les notions de base nécessaires et la compréhension des techniques de traitement d'images et suivi des notions d'anatomie du cerveau, qui permettront d'introduire les principes de l'imagerie par résonance magnétique cérébrale.

— **Chapitre 2 : Deep Learning et ségmentation d'image médicale.** Dans ce chapitre, Nous éxposons certaines approches de Deep Learning et les méthodes de segmentation des images. Parmis lesquelles nous avons décrit un certain nombre de techniques de segmentation appliqué sur les images IRM cérébrale.

— **Chapitre 3 : Conception** Dans ce chapitre , nous exposons de couvre la conception de notre système proposé. Nous commencerons par présenter les architectures générales et détaillées.

— **Chapitre 4 : Implémentation et résultat.** Dans ce chapitre, nous mettrons en œuvre les modèles que nous proposons et nous parlerons des résultats.

Enfin, nous conclurons nos travaux par une conclusion générale.

Chapitre 1

Diagnostique des tumeurs cérébrales et imagerie médicale

1.1 Introduction

Le cerveau est également sensible aux infections et autres maladies de gravité variable, telles que le cancer du cerveau, les tumeurs, qui sont au centre de notre projet. Chacun de ces troubles affecte négativement les processus cérébraux. Pour détecter ces infections et problèmes, il existe plusieurs méthodes cliniques telles que les tests médicaux, le diagnostic assisté par ordinateur, etc.

Dans ce chapitre, nous présenterons quelques concepts sur l'anatomie du cerveau et le type de tumeur , puis décrirons les images IRM qui sont principalement utilisées pour le diagnostic , et leur fonctionnement.

1.2 L'anatomie du cerveau

Environ 1,3 kg pèse le cerveau. Une barrière de protection est assurée par les méninges et la boîte crânienne en forme de crâne qui l'entoure. L'anatomie du cerveau fait référence à la structure et à l'organisation des différentes régions et parties qui composent cet organe complexe du système nerveux central.[52]

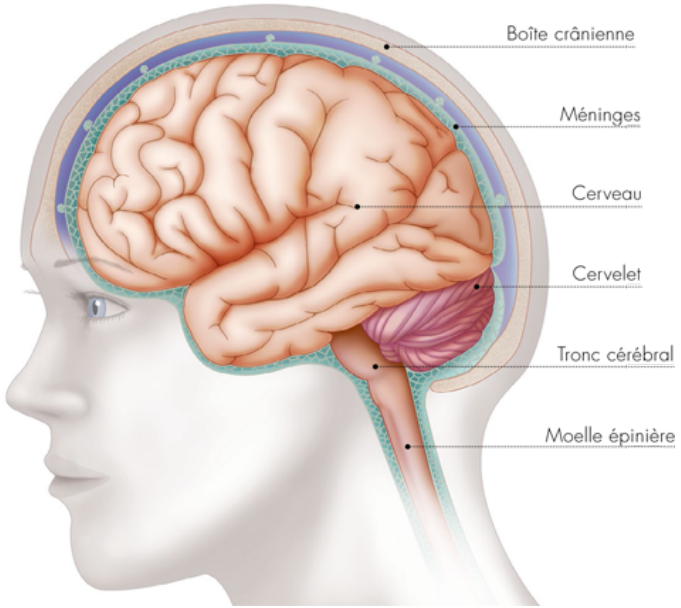


FIGURE 1.1 – Le Cerveau Humain [52]

1.3 Tumeurs cérébrales

Les tumeurs cérébrales sont des masses de cellules anormales appelées lésions néoplasiques qui se développent de manière incontrôlable à l'intérieur du cerveau et peuvent être décrites comme une augmentation du nombre de cellules sans en préciser la cause. Selon le degré de cancer, il existe deux grandes catégories de tumeurs cérébrales [37] :

— **Tumeurs bénignes (non cancéreuses)**

Le corps est capable de développer des tumeurs non cancéreuses. Ils se forment progressivement et il sont le plus souvent isolés localement du tissu cérébral . Il est possible de classer les kystes en tumulus bénigne.

— **Les tumeurs malignes (cancéreuses)**

Les cellules cancéreuses ont la capacité de se propager à d'autres tissus par les vaisseaux lymphatiques ou spléniques, entraînant des tumeurs métastatiques, qui sont de nouvelles tumeurs.

1.4 L'imagerie médicale

L'imagerie médicale est une branche de la science qui utilise diverses techniques d'imagerie pour visualiser l'intérieur du corps humain. Ces méthodes comprennent la radiographie, l'échographie, la tomodensitométrie (scanner), l'IRM (imagerie par résonance magnétique) et la scintigraphie. En fournissant des images détaillées des organes, tissus et structures internes du corps, l'imagerie médicale permet aux médecins de diagnostiquer et de surveiller les maladies, d'identifier les anomalies et d'évaluer l'efficacité des traitements [22].

1.5 Présentation des images medicale (IRMs)

L'IRM est basée sur le phénomène de résonance magnétique du proton. Ce phénomène de résonance est obtenu en imagerie clinique en plaçant le sujet dans un champ magnétique homogène intense d'une amplitude allant de 0,2T à 3T, ce qui équivaut à des fréquences d'excitation RF comprises entre 8 MHZ et 120 MHZ. Après avoir été excités, les protons qui composent

les tissus retrouvent leur état d'équilibre à des vitesses variables propres à leur composition physico-chimique en émettant des ondes RF [12].

La technique IRM pour l'imagerie médicale tomographique code les fréquences de résonance après qu'elles aient été dispersées par des gradients de champ. La principale source de contraste de l'image est déterminée par le temps de relaxation de l'image ou la vitesse de restauration de l'équilibre. Les distances et informations tridimensionnelles sont conservées par l'IRM, contrairement à l'imagerie par projection ou la radiographie, qui projettent une représentation 2D d'objets tridimensionnels.

1.5.1 Importance d'imagerie médicale

L'imagerie médicale a transformé la pratique de la médecine et entraîné des changements spectaculaires dans le secteur des soins de santé. Elle a choqué la communauté scientifique en lui donnant les faits dont il avait besoin pour faire valoir ses arguments concernant l'anatomie humaine. Grâce à une compréhension précise de l'anatomie du corps humain et de la localisation de la cible chirurgicale (possibilité de malformations congénitales facilement réparables en 3D), cette technologie permet de planifier les actes chirurgicaux tout en minimisant les risques potentiels et en rationalisant la procédure d'adaptation [26].

Un système d'imagerie médicale peut diagnostiquer des tumeurs, suivre leur progression et définir précisément leur nature grâce aux spécifications IRM.

1.5.2 Fonctionnement de l'IRM

Les rayons X utilisés dans le scanner, la radiographie et l'échographie sont des ultrasons. En revanche, IRM repose sur un principe physique différent qui utilise les propriétés magnétiques des atomes dans orbite pour éviter la haute fréquence qui pourrait nuire à la santé d'une personne. Les protons des atomes d'hydrogène dans le corps peuvent tous être stimulé simultanément en utilisant un ion très fort, provoquant les atomes d'hydrogène à tous point dans le même direction. Une fois cette étape initiale est terminée, les atomes seront excités par le champ électromagnétique (radiofréquence) et entrer dans un état de résonance. Après l'arrêt de la stimulation, les atomes libèrent leurs l'énergie, qui est ensuite mesurée et analysée. Étant

donné que tous les tissus n'ont pas le même nombre d'atomes d'hydrogène, la quantité d'énergie libérée variera en fonction de la composition du tissu. Des images en deux et trois dimensions ont été produites par analyse informatique des données [27]. Trois plans spatiaux peuvent être reconstruites sont :plan axiale, coronale et sagittale.

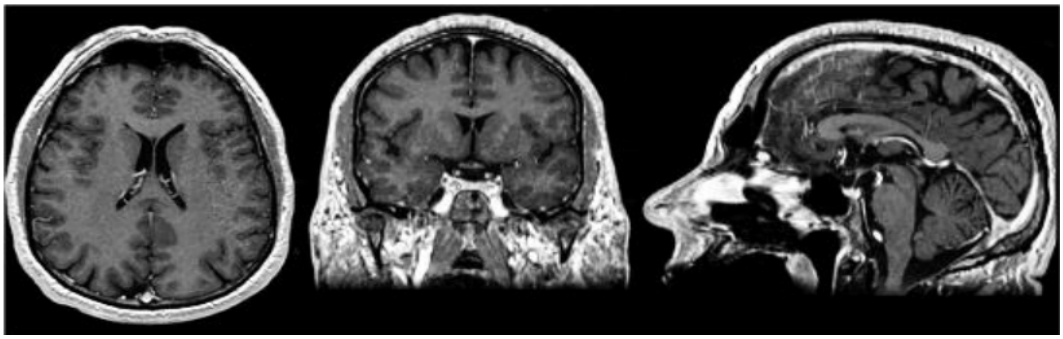


FIGURE 1.2 – plans axial, coronal et sagittal sur une acquisition en T1.

1.5.3 Qualité des images IRM

Contrairement à d'autres types d'images, il est évident qu'en imagerie médicale, il est nécessaire de minimiser la distorsion ou la dégradation visuelle afin d'éviter d'alarmer ou de fausser le diagnostic. Les facteurs qui influencent le plus la qualité des images IRM sont ceux liés aux erreurs d'acquisition. Nous distinguons principalement les problèmes de bruit, les artefacts de volume partiel, variations du champ magnétique et les artefacts de mouvement :

- Le bruit

Un bruit (parasite) dans une image est un phénomène de changement brusque d'intensité d'un pixel par rapport à ses voisins; elle résulte de l'éclairement des composants optiques et électroniques du capteur. Décrivez le (figure 1.3)Le bruit de l'image est causé par divers facteurs [35] :

— Imperfections du système RM :
  — Inhomogénéités de B0 ou B1.
  — Bruit thermique antennes.

  - Nonlinéarités amplificateurs.
- Lié au traitement de l'image.
- Bruit physiologique (mouvements respiratoires, cardiaques, ...).
- Bruit RF provenant de l'environnement.

FIGURE 1.3 – le Bruit.

Variations du champ magnétique

Les variances ou inhomogénéités d'intensité du champ magnétique (RF) sont des variations d'intensité observées pour un même tissu. Ce phénomène est dû au fait que le champ magnétique lors d'une acquisition n'est pas parfaitement homogène dans l'espace et dans le temps. Cette forme d'artefact pose des problèmes pour toute technique de traitement d'image basée sur l'intensité de l'image (segmentation, recalage...)[48].

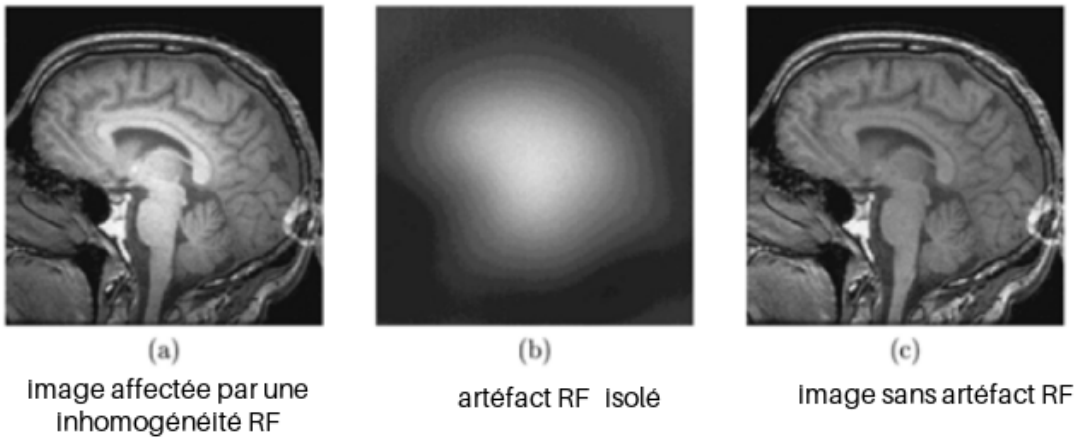


FIGURE 1.4 – Inhomogénéités RF.

Artéfact de mouvement

Artefact fréquent entraînant une baisse de la qualité de l'image. Elle se manifeste par une duplication de la région anatomique étudiée dans la Phase [48].

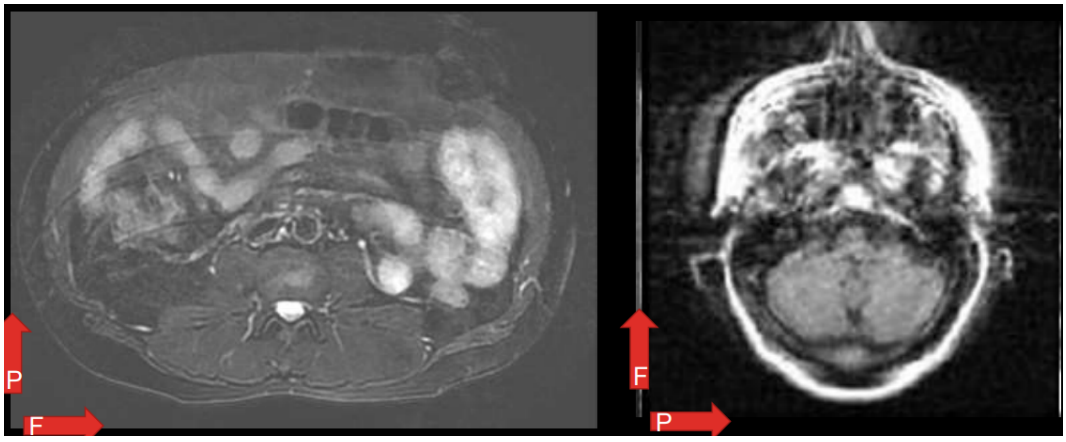


FIGURE 1.5 – Artéfact dû à un mouvement de la tête sur une IRM.

1.6 Le diagnostic de la tumeur

Le diagnostic d'une tumeur fait partie d'un processus clinique précis et complexe qui conduit ultérieurement à une décision thérapeutique appropriée. Elle est composée de plusieurs étapes dont la suspicion, la détection, l'observation et la détermination de sa nature histologique[37] .

Pour diagnostiquer ces tumeurs, il faut d'abord obtenir des images à l'aide de techniques d'imagerie médicale (telles que la tomodensitométrie par rayons X, IRM et TEP), après quoi les images doivent être traitées par des radiologues ou des algorithmes de traitement d'images [37].

1.7 Conclusion

Dans ce chapitre ,les idées fondamentales autour du capture d'images médicales par l'IRM sont discutées en détail dans ce chapitre, qui offre également une variété d'informations sur les tumeurs cérébrales.

Nous intéressons particulièrement aux technologies d'apprentissage en profondeur car nous avons vu avec quelle facilité des idées complexes peuvent être générées par les ordinateurs. Les détails seront développés dans le chapitre suivant.

Chapitre 2

Classification automatique des tumeurs et segmentation

2.1 Introduction

Dans le domaine de l'analyse d'images cérébrales, la segmentation est une tâche chronophage qui permet la mesure et la visualisation des caractéristiques anatomiques du cerveau, l'analyse des changements cérébraux et la délimitation des régions pathologiques. Pour cette raison, de nombreuses approches de segmentation ont été créées et rapportées dans la littérature. Dans ce chapitre, les approches de segmentation précédent ainsi que les méthodes basées sur l'apprentissage en profondeur sont discutées.

2.2 Définition de la segmentation

Une technique de traitement de base appelée segmentation divise l'image A en petits groupes, ou régions, en s'assurant qu'aucune région n'est laissée vide. L'intersection des deux régions doit être vide et l'image entière doit être visible dans toutes les régions. Une région est un groupe de pixels connectés qui partagent des caractéristiques qui les distinguent des pixels des régions voisines. En d'autres termes, la segmentation est la division d'une image en un certain nombre de zones distinctes qui ne fusionnent pas pour former l'image complète. Les caractéristiques de l'image, telles que le bruit, les primitives étrangères, le contour et les textures, sont des considérations importantes pour choisir la meilleure technique de segmentation [8].

2.3 Méthodes de segmentations

Les concepts fondamentaux de différence et de similitude vus par le système visuel humain sont référencés par segmentation. C'est en effet une étape cruciale dans le traitement des photos puisqu'elle conditionne leur interprétation. Ainsi, de nombreux algorithmes ont été proposés au cours de la dernière décennie [51]. Ils reposent généralement sur de nombreux principes sous-jacents. Dans cette section, nous proposons une étude non exhaustive des nombreuses méthodologies qui ont été présentées. Dans ce travail, nous nous concentrerons sur les trois catégories énumérées ci-dessous :

— Approches basées contour.

— Approches basées région

— Approches basées classification.

2.3.1 Approches basées contour

Les changements d'intensité, de couleur ou de texture, etc ..des images correspondent à des ajustements apportés aux propriétés physiques ou géométriques de la scène ou d'un objet. Dans de nombreuses situations, les opérations de segmentation bénéficient de la connaissance de l'évolution de ces propriétés. En raison des discontinuités de profondeur et de réflectivité de la scène observée, les contours sont des régions de transition qui divisent deux régions homogènes. Ils définissent les limites des régions qui correspondent aux bords ou aux groupes des éléments de scène [6]. Ces techniques existent depuis un certain temps et elles ont été les premières à utiliser la géométrie. Ils ont été développés au milieu des années 1980. Les contours extraits ne sont typiquement pas fermés ni même continus, il est donc fréquemment nécessaire de les combiner avec une méthode de fermeture ou de suivi en fonction du résultat calculé.

Méthode dérivative

La technique la plus populaire pour détecter les changements d'intensité par différenciation numérique est appelée méthode dérivée, qui fonctionne en détectant les changements dans un signal en fonction de sa dérivation. Dans ce cas, on peut pointer vers les opérateurs les plus courants du premier ordre : Robert [46], de Prewit [43], de Sobel [49]. Les opérateurs de deuxième ordre, le laplacien-gaussien , dans le cas de filtrage , on peut citer le filtre de canny [11].

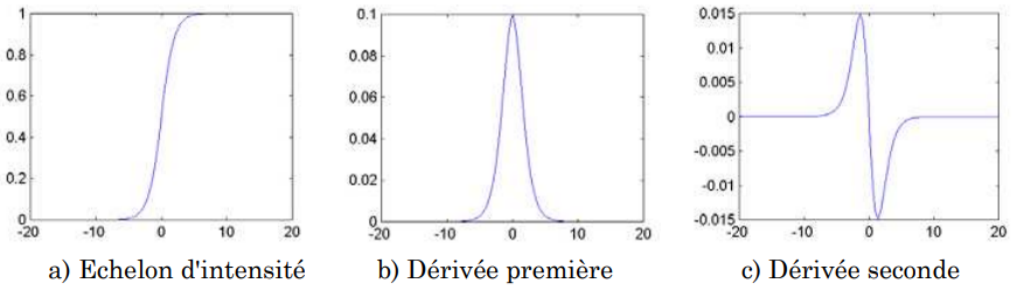


FIGURE 2.1 – Détection de contours .

Un opérateur de dérivation avec filtrage optimal est apparu afin d'améliorer la qualité des contours et pallier les problèmes de précision, de localisation et d'efficacité de détection. Le filtre idéal est un différenciateur qui détecte les contours tout en respectant les critères de Canny [11]. Celles-ci se résument comme suit :

1. Une bonne détection : l'opérateur donne une réponse au voisinage d'un contour ;
2. Une bonne localisation : optimisation de la précision avec laquelle le contour est détecté ;
3. Unicité de la réponse : le contour doit provoquer une réponse unique de l'opérateur.

2.3.2 Approche régions

La segmentation par région est une méthode particulière dans laquelle on essaie de construire une surface en regroupant des pixels adjacents selon un critère d'homogénéité.la segmentation par région crée un ensemble de régions qui ont les propriétés suivantes [1] :

— la réunion de toutes les régions donne l'image entière.
— les régions sont connexes c'est à dire que tous les pixels d'une même région sont jointifs.
— tous les pixels d'une même région sont homogènes entre eux.
— les pixels de deux régions adjacentes ne sont pas homogènes entre eux.

Ce type de méthodes de segmentation intègre automatiquement des informations spatiales dans le processus de segmentation en plus du niveau d'information de gris [32][15][40].

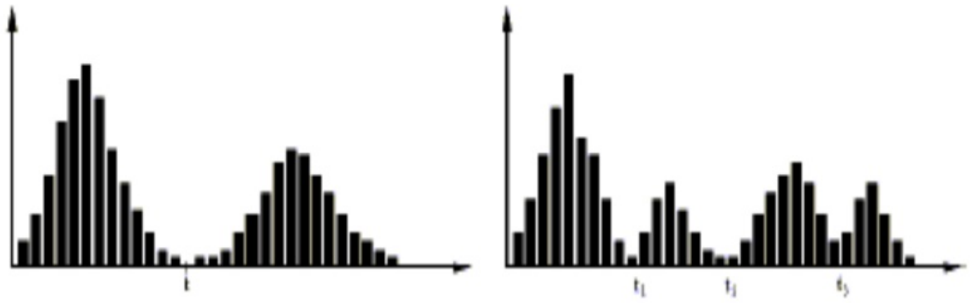
Cette approche se distingue par exemple des segmentations par contours ou seuillage dans lesquelles les régions créées ne possèdent pas toutes ces propriétés.

seuillage

La technique de segmentation la plus avancée pour extraire des objets du fond d'une image est le seuillage [30].C'est l'opérateur qui attribue une classe de luminosité spécifique à chaque point de l'image. Le seuillage est réalisé après comparaison de la luminosité à un ou plusieurs seuils.

La façon la plus courante de déterminer les seuils à l'aide de ces analyses est de les appliquer localement ou globalement à l'image ou au volume. Cette technique tente de déterminer les classes de pixels en utilisant l'analyse d'histogrammes monodimensionnels [30]. Lorsqu'une

image comporte une ou plusieurs régions de même nature et un fond visuellement distinct, l'histogramme est bimodal et il est possible de diviser les régions du fond à l'aide d'un simple seuil qui se positionne dans la vallée entre les deux modes (figure 2.2). L'histogramme de l'image est multimodal (figure 2.2(b)) s'il est constitué de régions aux caractéristiques diverses, chaque mode correspondant à une région différente. La difficulté à déterminer le minimum entre deux modes voisins, ou vallées, est ainsi qualifiée de problème de seuillage. La même région est alors affectée aux pixels avec des niveaux de gris au milieu de deux vallées.



(a) Histogramme bimodal (b) Histogramme multi-modal

FIGURE 2.2 – Exemples d'histogrammes

En général, il existe deux manières différentes de choisir les critères de sélection de l'histogramme. Contrairement au second, où le seuil est calculé localement sur la base d'une petite fenêtre de configuration, le premier détermine le seuil globalement pour chaque point de l'image.

Croissance de région

Une méthode simple de segmentation d'une image basée sur une région est appelée croissance de région. Parce qu'elle implique de choisir des points de départ initiaux, elle est également classée comme une méthode de segmentation d'image basée sur les pixels. Cette approche de segmentation examine les pixels entourant les points de départ initiaux et détermine si ces pixels doivent être ajoutés à la région. De manière similaire aux algorithmes généraux de regroupement de données, la procédure est itérée. Un aperçu général de l'algorithme de croissance régionale est fourni ci-dessous sous le titre région en croissance.

Méthode de fusion-diffusion

Le point commun entre ces méthodes est qu'elles commencent toutes par une division initiale non homogène de l'image (qui est généralement l'image elle-même). La division est effectuée l'obtention de partitions homogènes [53].

Suite au processus de partitionnement, de nombreuses petites pièces de surface sont fréquemment connectées. L'étape de fusion rassemble des régions voisines. La fusion des régions se poursuit selon le critère d'homogénéité jusqu'à l'obtention du plus grand segment connexe. Utilisant un algorithme de "fusion de division", cette technique est appliquée aux images en niveaux de gris. Cet algorithme est divisé en deux étapes [36] :

— Dans la première étape, chaque bloc (correspondant à un nœud) est examiné pour déterminer s'il doit être divisé en quatre blocs plus petits, jusqu'à ce que chaque bloc plus petit satisfasse à l'exigence d'homogénéité.

— Dans une étape ultérieure, une fusion de blocs adjacents ayant des propriétés colorimétriques similaires est réalisée.

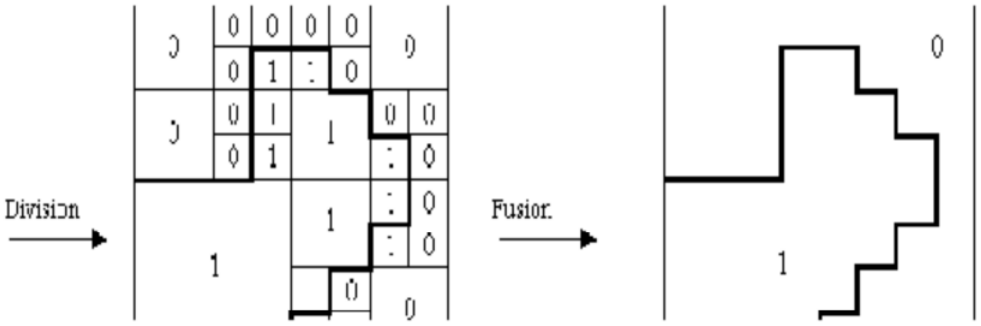


FIGURE 2.3 – La segmentation par division-Fusion

2.3.3 Approche par classification

Ces techniques relèvent des techniques d'apprentissage automatique.Ces méthodes de recherchent des modèles naturels dans les données qui fournissent des perspectives et nous aident à prendre des décisions et des prévisions plus judicieuses.

Ils sont employés quotidiennement pour prendre des décisions critiques liées au diagnostic médical. La division générale des techniques d'apprentissage automatique en algorithmes d'apprentissage supervisés et non supervisés [2].

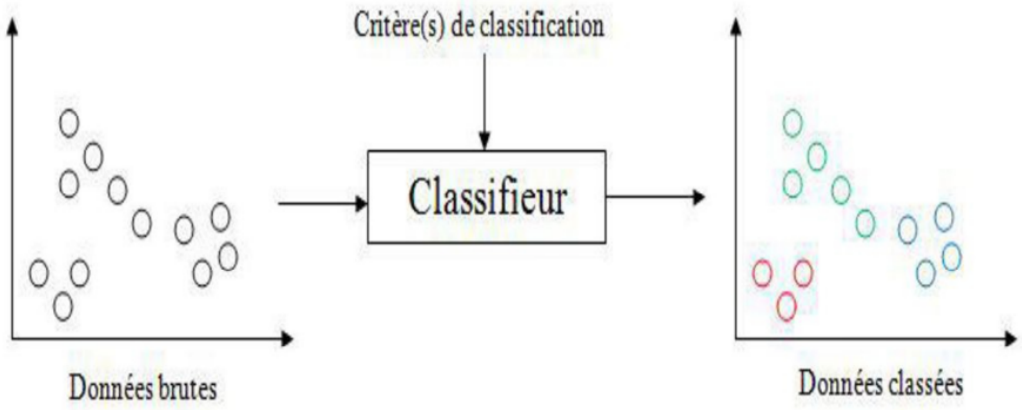


FIGURE 2.4 – Principe de classification

— **L'apprentissage supervisé**

Est l'apprentissage à l'ordinateur à faire quelque chose, C'est alors que le modèle commence à comprendre la corrélation entre les données et les étiquettes. Toutes les tâches de classification reposent sur des ensembles de données étiquetées, telles que celles utilisées pour la détection des visages, la reconnaissance des personnes sur les photos et la reconnaissance des expressions faciales (telles que la joie ou la surprise). La base de l'apprentissage en profondeur est l'apprentissage guidé.

— **L'apprentissage non supervisé**

C'est l'apprentissage sans étiquettes. Plus un algorithme peut générer de données, plus il sera précis. Par conséquent, l'apprentissage non supervisé peut aboutir à des modèles extrêmement précis. Autrement dit. Il permet aux ordinateurs d'acquérir de nouvelles compétences et utilise ces compétences pour rechercher dans les données des modèles

et des structures. L'exemple idéal d'un algorithme d'introduction non supervisée est K-mean.

Alors que le regroupement est un processus d'apprentissage non supervisé, la classification est une méthode d'apprentissage guidé. A titre d'exemple Afin de minimiser ou de maximiser une fonction objectif (fonction de perte), la méthode de classification SVM utilise une collection de données d'entraînement (images et étiquettes) ainsi qu'un nez de fonction de base radiale qui tente de maximiser la distance entre deux classes différentes . En conséquence, l'algorithme SVM affecte chaque pixel à l'une des classes prédéfinies [34]. Les algorithmes utilisés dans le clustering tentent de diviser les images d'entrée en un certain nombre de clusters séparés et disjoints. Un exemple d'une telle technique est K-means, qui divise une image en un certain nombre de centres gravitationnels, dont chacun sert de centre du cluster. En conséquence, l'algorithme K-means tente de classer chaque pixel en l'attribuant à l'un des nombres prédéfinis de centres gravitationnels, ce qui signifie que le nombre de classes est égal au nombre de centres gravitationnels [49].

Ces techniques traitent des informations plus complètes extraites d'une image ; ils tentent de segmenter un IRM sans relation claire entre les régions de pixels et les intensités. Dans notre projet, nous nous intéressons particulièrement à la classification des méthodes basées sur les réseaux de neurones que nous décrirons plus haut.

2.4 Apprentissage profond

Apprentissage profond, souvent appelé « deep learning » en anglais, est une méthode d'apprentissage automatique. Les systèmes d'apprentissage en profondeur utilisent des couches de neurones qui communiquent entre elles pour analyser des données non structurées et prédire les résultats. Ces systèmes ont la capacité de s'auto-améliorer et d'apprendre. En effet, lorsqu'une couche détecte une erreur, les données sont automatiquement éliminées et renvoyées à la couche précédente afin de corriger le modèle mathématique.Le modèle pourra alors combiner et séparer ces nouvelles données sans que personne n'ait à lui dire de le faire lorsqu'il lui sera présenté [47]. .

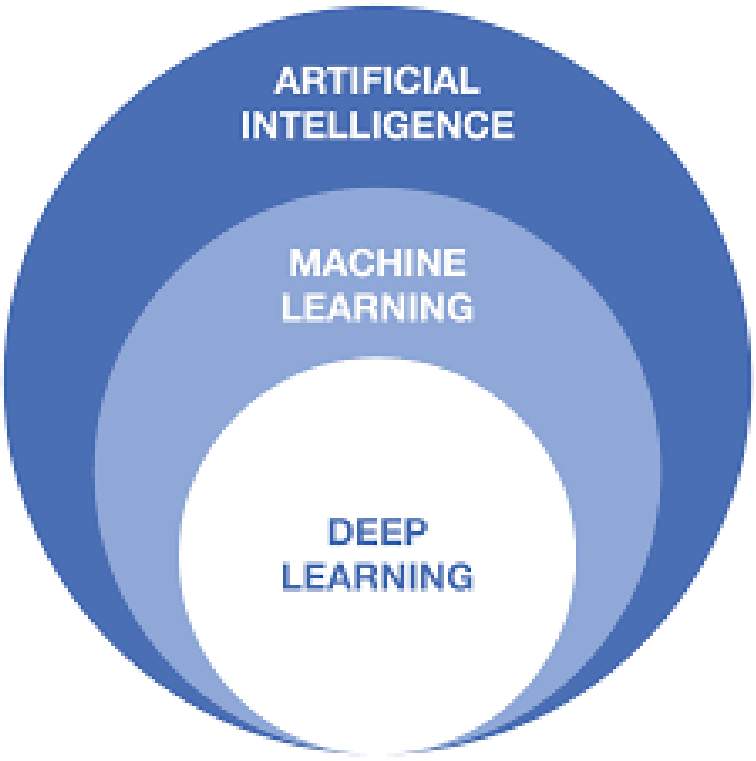


FIGURE 2.5 – Definition de Deep Learning.

2.5 Réseaux de neurones

Les réseaux de neurones sont Les réseaux parallèles d'éléments de traitement ou nœuds qui composent les réseaux neuronaux s'inspirent du fonctionnement des neurones biologiques. Chaque noeud est capable d'effectuer des calculs. L'apprentissage s'effectue en ajustant le poids attribué aux raccords noeud à noeud. Il est le plus souvent utilisé comme classificateur en imagerie médicale, où le poids est établi en utilisant des données d'apprentissage et contribue ensuite à la segmentation de nouvelles données. De plus, ils peuvent être utilisés dans des méthodes non contrôlées [42].

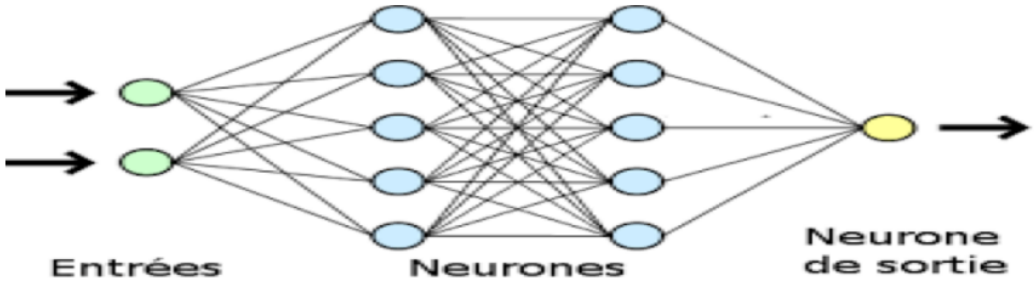


FIGURE 2.6 – Réseau de neurones.

2.5.1 Neurone biologique

Les synapses naturelles sur les dendrites des neurones transmettent des signaux aux neurones. Le neurone sera activé et enverra un signal dans l'axone après avoir reçu les signaux. Ce signal peut être envoyé à un autre site synaptique et peut activer plus de neurones [33].

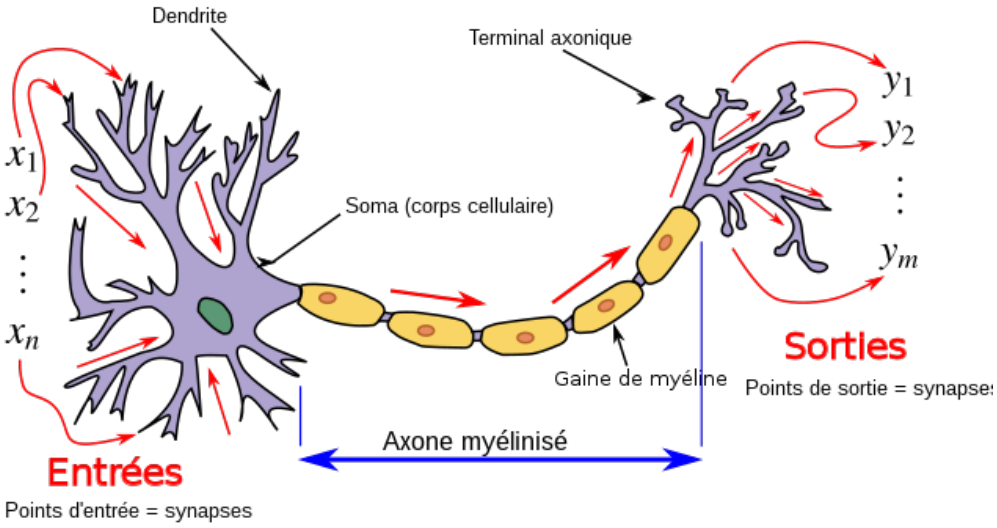


FIGURE 2.7 – Neurone biologique.

2.5.2 Neurone artificiel

Le neurone artificiel, également connu sous le nom de perceptron, est la pierre angulaire d'un réseau de neurones. C'est une opération mathématique basée sur un modèle neuronal biologique. Elle peut être vue comme une porte logique de sortie binaire (Figure 2.7). Cela consiste en [33] :

— **Couche d'entrée**

Une autre façon de le dire est une valeur d'entrée. Pour qu'un neurone utilisant cette couche fonctionne, nous transmettons des valeurs d'entrée qui représentent un nombre spécifique de variables, telles que X = x1, x2, x3,..., xn.

— **Poids et biais**

Chaque entrée neuronale artificielle a un poids appelé w qui représente la valeur de la connexion.. La somme pondérée est la somme des valeurs d'entrée multipliée par le poids $\sum_{i=1}^{n} w_i * x_i$. Une valeur de biais est ajoutée pour obtenir la valeur finale de la prédiction.

— **couche d'activation**

C'est un nœud qu'un réseau de neurones place au début ou au milieu. Avant d'envoyer l'entrée à la couche suivante de neurones ou de la convertir en sortie, nous appliquons la fonction d'activation, une transformation non linéaire. La liste suivante des fonctions de couche d'activation utilisées par CNN est fournie :

Sigmoid :

$$Sigmoid(x) = \frac{1}{(1 - e^{-x})}$$

ReLU :

$$ReLU(x) = \max(0, x)$$

Tanh :

$$Tanh(x) = \frac{2}{(1 - e^{-2x})} - 1$$

Softmax :

$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

— **Couche de sortie :** Elle fournit la sortie finale d'un neurone, qui peut ensuite être envoyée à d'autres neurones du réseau ou utilisée comme valeur finale de sortie après avoir été comparée à une valeur seuil.

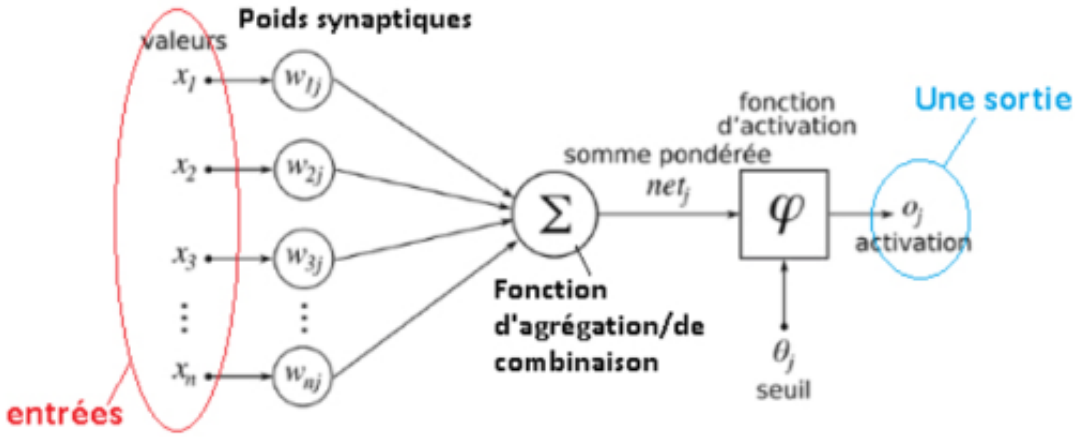


FIGURE 2.8 – Neurone artificiel.

On peut résumer les comparaisons entre neurones biologiques et artificiels dans le tableau 2.1 suivant après avoir expliqué les deux types de neurones :

Neurone artificiel	neurone biologique
entrée	Dendrites
Nœud	Soma
Sortir	Axone
Interconnexion	Synapse

TABLE 2.1: Neurone biologique vs neurone artificiel.

2.6 réseau neuronal convolutif

Un réseau de neurones artificiels d'apprentissage en profondeur connu sous le nom de réseau de neurones convolutés (CNN), parfois appelé CNN ou ConvNet, est couramment utilisé pour analyser l'entrée visuelle. Leur fonctionnement est basé sur l'opération de convolution mathématique. Il y a plusieurs couches dans une structure CNN. Tous ces couches sont assemblés d'une manière ou d'une autre pour créer un modèle CNN complet. L'avantage fondamental de CNN par rapport à ses prédécesseurs est qu'il reconnaît les fonctionnalités pertinentes automatiquement et sans intervention humaine [23]. Il existe plusieurs cachés dans un circuit neuronal de réseau, les plus importants de CNN étant :

✓ Couche convolutive.

✓ Couche d'activation (Activation layer).

✓ Couche de mutualisation (Pooling layer).

✓ Couche entièrement connectée (Fully-connected layer).

✓ Dropot.

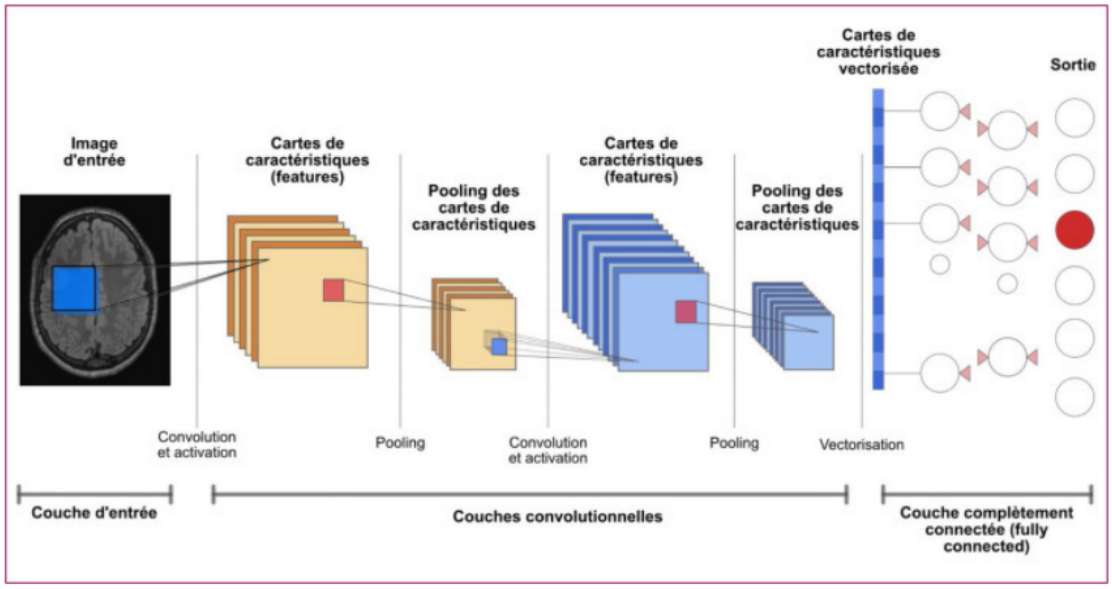


FIGURE 2.9 – Architecture CNN.

2.6.1 Couche de convolution

La majorité des calculs sont effectués dans le noyou et la première couche du composant de construction convolutif CNN. En réalité, l'image est tirée d'une carte caractéristique, également appelée carte d'activation, après avoir traversé une couche convolutive. Une couche convolutive est créée par plusieurs filtres effectuant l'opération convolutive. Chaque image est considérée comme une matrice de valeurs de pixel [55]. Nous utilisons une taille de noyau de m et C dans les couches convolutives, où m est la taille du noyau convolutif et C est la profondeur d'un filtre. L'opération convolutive peut être notée comme suit :

$$x_j{}^l = \sum_{l=1}^{n} x_i^{l-1} * k_{ij}^{l-1} * b_i^l$$

où x désigne la convolution, x_j^i est le j^{th} h carte de sortie dans la couche l, le noyau convolutif k_{ij}^{l-1} (également appelé poids) peut être mis à jour lors de la formation du réseau. Il relie le i^{th} carte de sortie dans la couche $l1$ et le j^{th} carte de sortie dans la couche l. b_i^l est le paramètre de biais entraînable de la j^{th} carte de sortie dans la couche l [55].

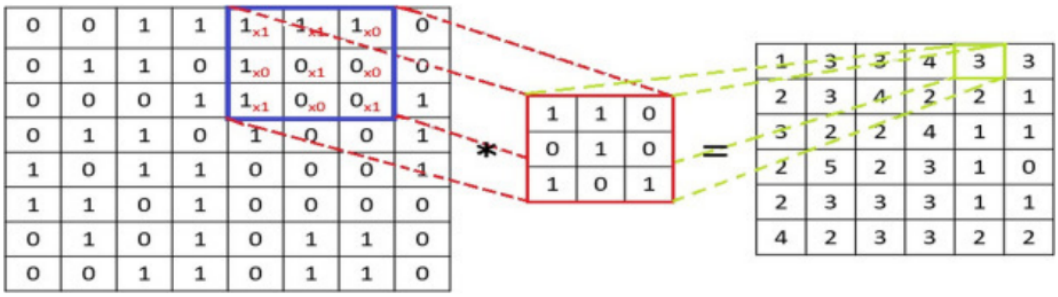


FIGURE 2.10 – Illustration d'un exemple simple de carte d'activation informatique.

2.6.2 Couches de correction

Acronyme de RectiedLinear Unit(ReLU), qui est une fonction d'activation appliquée après chaque opération de convolution et fréquemment utilisée. Elle remplace toutes les valeurs de pixel négatives en les mettant à zéro. La formule suivante interprète la fonction relu : f(x) =

max(0 ; x). Plusieurs fonctions d'activation sont disponibles, notamment la correction par tangente hyperbolique, la correction par fonction sigmoïde et la correction par tangente hyperbolique saturante. Cependant, la correction Relu est préférable car elle conduit au développement d'un réseau de neurones plusieurs fois plus rapide sans altérer significativement la généralisation de la précision [5].

2.6.3 Couche de regroupement

La couche de regroupement (pooling)est une autre couche des réseaux de neurones complexes. Le but de cette couche est de réduire les dimensions spatiales (L et H) tout en conservant la même profondeur que la couche précédente (C'). Les entités identifiées dans une zone spécifique de la carte d'entités produite par la couche convolutive sont agrégées par la couche de regroupement. Dans la plupart des cas, une couche de regroupement est située entre deux couches convolutionnelles. Différentes catégories d'opérations courantes existent, comme les pools maximum, minimum, moyen et maximum. Les méthodes d'échantillonnage les plus fréquemment utilisées sont le maximum et la moyenne [56].

— **Max pooling**

  Une couche max-cluster sélectionne le cluster des éléments les plus grands de la région de la carte d'entités couverte par le filtre. Par conséquent, la sortie de la couche max-cluster sera une carte d'entités contenant les entités les plus importantes de la carte d'entités précédente [56].

— **Moyen Pooling**

  Il calcule la moyenne des éléments présents dans la région de la carte d'entités couverte par le filtre. Comme nous l'avons déjà dit, la mise en commun maximale renvoie les fonctionnalités les plus importantes, tandis que la mise en commun moyenne renvoie la moyenne de toutes les fonctionnalités du bloc [56].

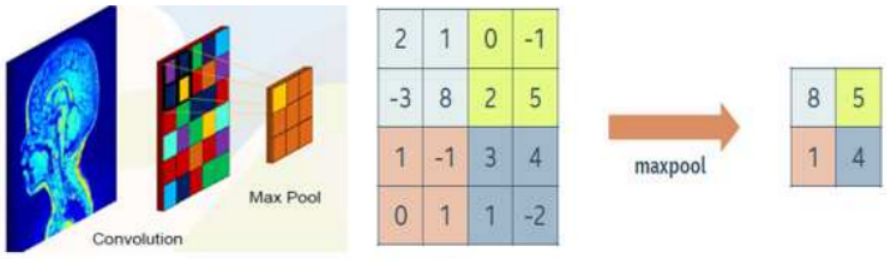


FIGURE 2.11 – Example simple de max pooling.

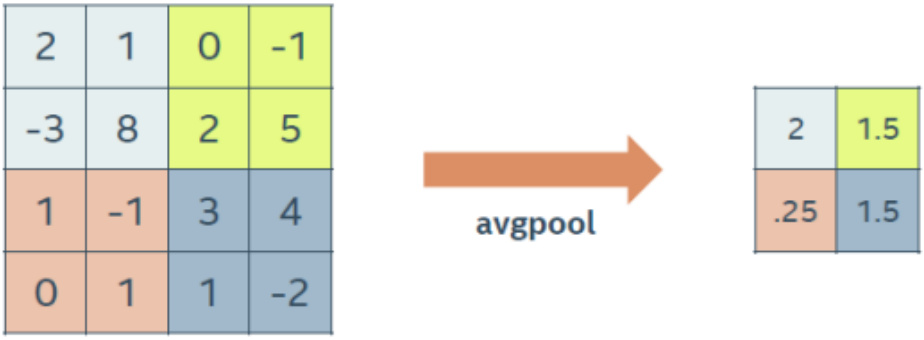


FIGURE 2.12 – Example simple de moyen pooling.

2.6.4 Couche entièrement connectée

Chaque neurone de la couche entièrement connectée est connecté à tous les autres neurones de la couche précédente, de la même manière que les neurones sont disposés dans un réseau neuronal standard. Cette couche effectue une classification basée sur les caractéristiques récupérées par les couches précédentes (couches convolutionnelles et de pooling), ainsi que leurs différents filtres. Ce sont les paramètres les plus utilisés avec le CNN dans ces couches et prennent beaucoup de temps en formation [9].

2.6.5 Couche Dropot

Dropot est un exemple de méthode de régularisation et de traitement des problèmes de surajustement et de calcul coûteux. Il empêche le surajustement et permet des combinaisons efficaces de nombreuses topologies de réseaux neuronaux exponentiellement différentes. L'idée d'un dropot est de placer des unités (neurones) dans un réseau de neurones où une unité de dropot est sélectionnée au hasard avec une probabilité. Lorsqu'une unité est abandonnée, toutes ses connexions entrantes et sortantes sont ignorées [41]. Comme illustré à (Figure 2.13) Dans le cas le plus simple, chaque unité est retenue avec une probabilité fixe p indépendante des autres unités, où p peut être choisi à l'aide d'un ensemble de validation, ou simplement fixé à 0,5, ce qui semble proche de l'optimal pour une large gamme de réseaux , et la tâche. Cependant, pour les cellules d'entrée, la meilleure probabilité de rétention est généralement plus proche de 1 que de 0,5 [51].

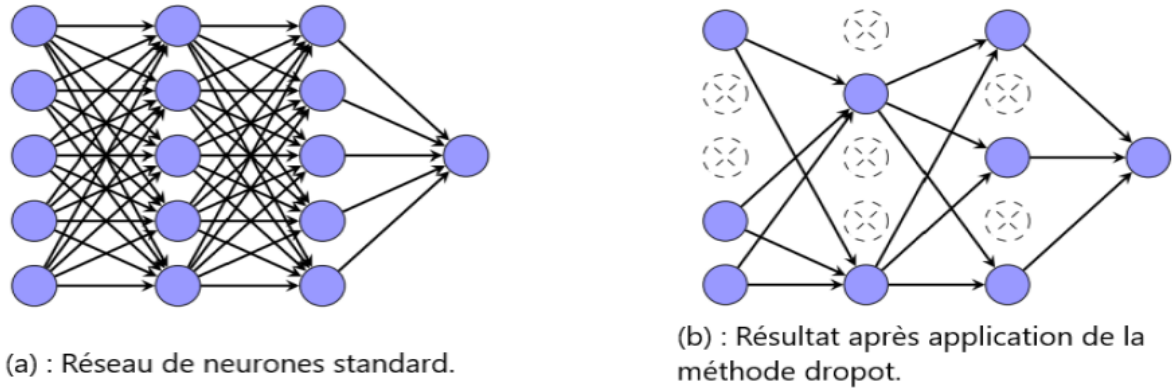


FIGURE 2.13 – Illustration d'un exemple de méthode de dropot.

2.6.6 Augmentation des données

La quantité et la variété des données d'apprentissage déterminent le succès de la plupart des modèles d'apprentissage automatique, en particulier les modèles d'apprentissage en profondeur. Cependant, l'un des problèmes les plus courants dans les applications d'apprentissage automatique est le manque de données disponibles [18].

L'augmentation des données est un ensemble de techniques utilisées pour générer des points de données supplémentaires à partir de données réelles afin d'augmenter artificiellement la quantité de données disponibles. Par exemple, apporter de petites modifications aux données ou appliquer des modèles d'apprentissage en profondeur pour créer des points de données supplémentaires. Ces modifications sont : remplissage, rotation aléatoire, mise à l'échelle, retournement et mise à l'échelle verticaux et horizontaux, etc.

2.6.7 Apprentissage par transfert et réglage

En fait, apprendre un modèle à partir de zéro est un processus difficile dans des situations pratiques. Il se peut que l'algorithme d'entraînement ne fonctionne pas correctement, qu'il y ait trop d'époques d'entraînement ou qu'il n'y ait pas assez de données d'entraînement. L'apprentissage par transfert est l'un des moyens de faciliter la formation. Les développeurs peuvent prendre un modèle pré-formé et l'appliquer à une tâche similaire avec quelques modifications. L'apprentissage par transfert peut être défini comme un processus d'ajustement [28]. Il présente de nombreux avantages, dont les plus importants sont un temps de formation réduit et des performances de réseau de neurones plus élevées (dans la plupart des cas).

— **réglage fin(Fine-Tuning) :**

Alors que les poids de toutes les couches sauf les avant-dernières d'un réseau neuronal pré-entraîné (sur l'ensemble de données A) sont gelés, le réseau neuronal est entraîné sur l'ensemble de données B. Cela nous permet d'améliorer le modèle de base.

— **Modèles pré-entraînés :**

AlexNet, GoogleNet, ResNet, Xception, VGG, Inception,InceptionResNet, MobileNet, MobileNet SSD ...

2.7 Les modèles de segmentation par Apprentissage profond

2.7.1 Unet

U-Net est un modèle créé pour la segmentation des images biomédicales qui est dérivé des réseaux conventionnels de convolutions de neurones. Il effectue la localisation et la segmentation en catégorisant chaque pixel de sorte que la taille de l'entrée et la sortie sont les mêmes. Ce modèle en U bien connu (figure 2.14) est symétrique et se compose de deux parties principales :

— Le côté gauche est appelé le chemin de la contraction et est composé de nombreux blocs de contraction dans le processus général alambiqué. Chaque bloc applique deux couches convolutionnelles 3x3, suivies d'une fonction Relu et d'un pool max 2x2. Après chaque bloc, il y a deux fois plus de filtres ou de cartes caractéristiques pour que l'architecture puisse apprendre efficacement des structures complexes [7].

— Le côté droit est appelé le chemin de l'expansion et est également composé de nombreux blocs d'expansion. Chaque bloc applique une couche de transposition convolutive (ou deconvolution) 2x2, suivie d'une concaténation avec les caractéristiques correspondantes provenant du chemin de la contraction. Ensuite, deux couches convolutionnelles 3x3 sont appliquées, suivies d'une fonction Relu. Cette structure en U permet de capturer à la fois les informations contextuelles et les informations de localisation précises.

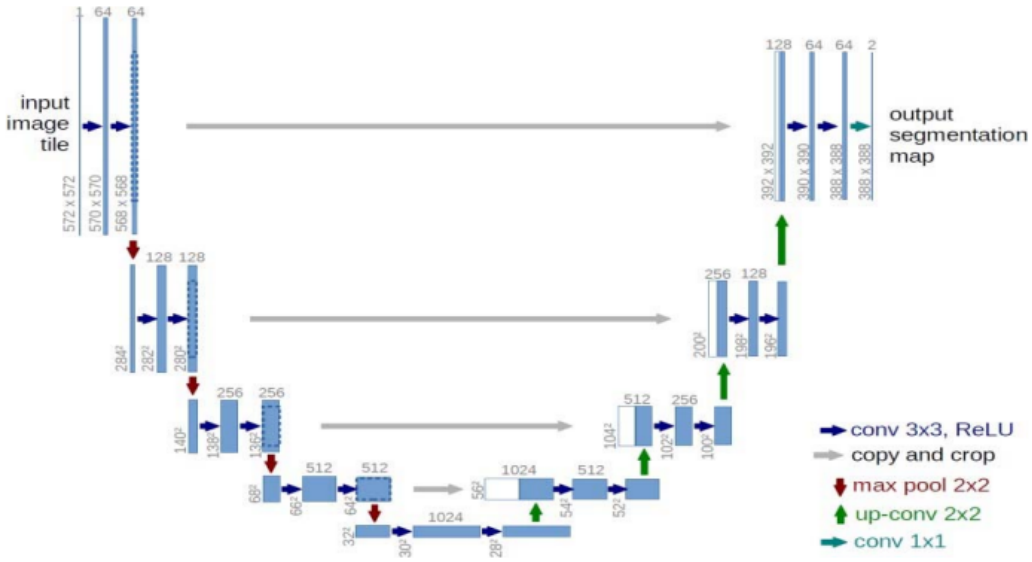


FIGURE 2.14 – L'architecture du modèle Unet.

Les avantages de U-NET :

— Pour former un modèle d'apprentissage profond, une grande collection de données est nécessaire. Il pourrait être difficile de compiler autant de données que nécessaire pour traiter la catégorisation du problème d'image en termes de temps, d'argent et de ressources matérielles.

— U-Net peut apprendre avec très peu d'images étiquetées qui sont bien adaptées à la segmentation d'image en biologie ou en médecine et ne nécessite pas beaucoup d'exécutions pour faire la segmentation d'image. Les auteurs 3 ont testé leur architecture sur quelques défis de segmentation d'image et ont constaté moins d'erreurs qu'avec les réseaux conventionnels de neurones tordus.

2.7.2 SegNet

Le modèle SegNet établit avec succès un équilibre entre le temps de calcul et l'exactitude de la classification. L'architecture symétrique de SegNet permet un remplacement précis des caractéristiques abrasives au bon emplacement spatial.

L'architecture d'encodeur-décodeur de SegNet est basée sur la notion de couche convolutive du

modèle VGG-16 (qui se compose de 16 couches convolutives et est très attrayante en raison de son architecture très unifiée) [19].

L'encodeur est composé d'un certain nombre de couches convolutionnelles suivies d'une normalisation par lots (BN) et d'une fonction de transfert linéaire (ReLU). Chaque bloc de deux ou trois convolutions est suivi d'une couche de pooling (sous-échantillonnage) qui n'est pas égale à deux; le résultat est la symétrie de l'encodeur, qui a le même nombre de convolutions et de blocs [19].

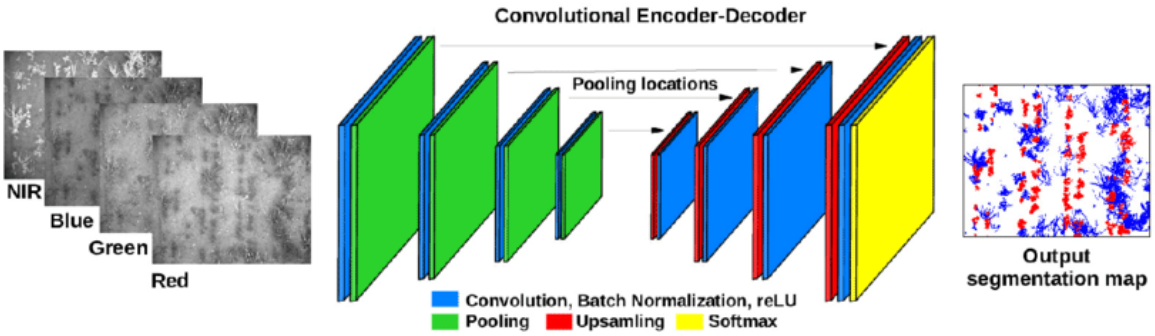


FIGURE 2.15 – L'architecture du modèle SegNet[24].

2.7.3 ResNet

Chaque module dans ResNet occupe un couche, constituant le réseau. Chaque couche a un groupe de fonctions qui peuvent être utilisées à l'entrée. La profondeur de ResNet peut changer considérablement.[24]

- Les ResNets sont assez flexibles. Pour créer un réseau, des centaines ou des milliers de couche individuels peuvent être ajoutés.
- Ils pourraient être faits pour déterminer la profondeur d'un réseau spécifique.
- L'utilisation des blocs de construction restants permet d'obtenir une architecture plus facile à optimiser et d'obtenir une précision de profondeur considérablement accrue.
- Ils augmentent les propriétés de convergence et facilitent le flux de rétropropagation.

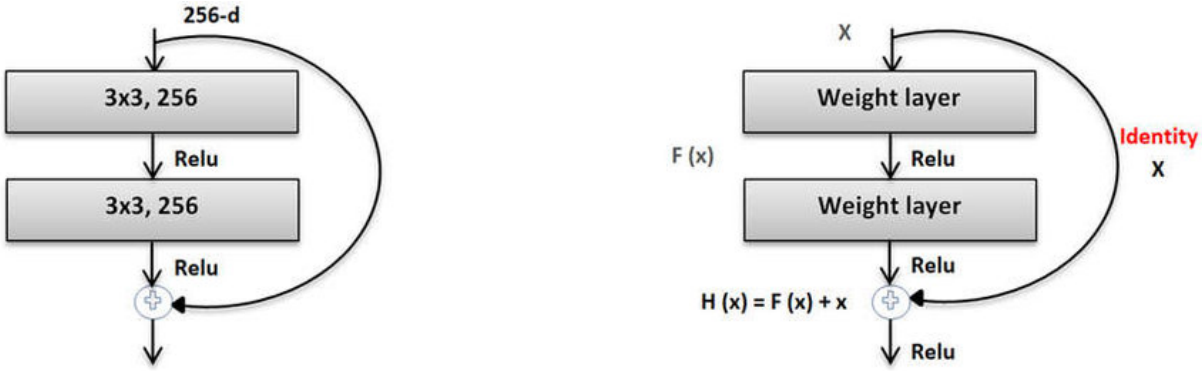


FIGURE 2.16 – Architecture ResNet.

Les advantages ResNet :

- La capacité de ces réseaux à réduire la défaillance du gradient dans les couches les plus basses du réseau et leur capacité à être réduits sont également des avantages.
- Ce dernier permet d'ajuster l'architecture à la tâche de classification à accomplir.
- Malgré l'utilisation de beaucoup de ressources de calcul, ils obtiennent de très bons résultats de classification sur tous les niveaux de détails d'image, y compris les plus minuscules comme les nodules pulmonaires ou les hémorragies intracrâniennes [54].

2.8 Symétries

Le terme "symétrie" désigne la similitude ou la correspondance entre les structures anatomiques des côtés opposés du corps ou de l'axe dans l'analyse des photographies médicales. Pour cette raison, la "segmentation de symétrie" consiste généralement à identifier et caractériser des structures symétriques dans des images médicales, telles que le cerveau, les poumons [14].

Il existe plusieurs applications pour la segmentation de la symétrie, y compris l'évaluation des asymétries structurelles ou des changements de symétrie provoqués par une maladie . Par exemple, la segmentation cérébrale symétrique en neuroimagerie peut être utilisée pour évaluer les différences de volume ou de forme entre les rotations gauche et droit, ainsi que la façon dont ces différences sont liées à diverses maladies neurologiques [44].

2.9 Les options d'apprentissage

2.9.1 Optimiseurs

— **Adam (adaptive moment estimation) :** L'optimiseur Adam inclut des informations sur le taux d'apprentissage, le facteur de régularisation L2 et la taille des tailles de mini-lot.

— **SGDM (stochastic gradient descent with momentum) :** Options d'entraînement pour la descente de gradient stochastique avec des moments, qui incluent des informations sur le taux d'apprentissage, le facteur de régularisation L2 et la taille du taille de mini-lot [3].

2.9.2 Epoque

— **Epoch :** (numéro d'époque) correspond à un passage complet des données.

— **Max Epochs :** Nombre maximal d'époques à utiliser pour l'apprentissage.

2.9.3 Taux d'apprentissage

— **Initial Learn Rate :**

Taux d'apprentissage initial pour l'entrainement, spécifié par la paire séparée par des virgules de "InitialLearnRate" et d'un scalaire positif. La valeur par défaut de l'optimiseur "Adam" est 0,001. Si le taux d'apprentissage est trop faible, la formation prend du temps.

— **LearnRateSchedule :**

Possibilité de réduire le taux d'apprentissage pendant l'entrainement, Spécifié sous la forme d'une paire séparée par des virgules composée de "LearnRateSchedule" et de l'un des éléments suivants :

— «None» : Le taux d'apprentissage reste constant tout au long de l'entrainement.

— «Piecewise » : Le logiciel met à jour le taux d'apprentissage après certain nombre d'époques en multipliant par un certain facteur.

— **LearnRateDropPeriod :**

Nombre d'époques pour supprimer le taux d'apprentissage, en spécifiant le nombre d'époques entre les multiplications. Cette option n'est valide que lorsque la valeur de LearnRateSchedule est "Piecewise".

— **LearnRateDropFactor :**

Facteur d'abandon du taux d'apprentissage, spécifiez la valeur multiplicateur. Cette option n'est valide que lorsque la valeur LearnRateSchedule est "Piecewise".

2.9.4 Fréquence de validation

Fréquence de validation du réseau en itérations, spécifiée sous la forme d'un entier positif. La valeur ValidationFrequency est le nombre d'itérations entre les évaluations des métriques de validation [3].

2.9.5 Taille du mini-lot

Est un sous-ensemble de l'ensemble d'apprentissage utilisé pour évaluer le gradient de la fonction de perte et mettre à jour les poids.

2.9.6 Ressource matérielle

L'environnement d'exécution (ExecutionEnvironment) : Ressource matérielle pour l'entrainement du réseau : 'cpu', 'gpu','multi-gpu' et 'multi-cpu'.

— **CPU :** Est le processeur central de traitement de l'ordinateur, son acronyme signifie l'unité principale de traitement. Le CPU est chargé d'effectuer divers calculs. Il est capable d'effectuer de nombreuses tâches différentes [17].

— **GPU :** Désigne un processeur équipé d'une carte graphique, et son acronyme est Graphics Processing Unit. Les GPU ne gèrent que les calculs graphiques.Il gère l'affichage des pixels, des textures et des formes à l'écran, ainsi que le traitement de la vidéo [17]. L'entrainement avec un gpu est beaucoup plus rapide qu'un cpu.

2.10 Synthèse des traveaux relatifs

Quelque traveaux relatifs :

— **Exploitation de la symétrie cyclique dans les réseaux de neurones convolutifs [45] :**

Dans cet article, les auteurs proposent une technique qui exploite la symétrie binaire in-

hérente du cerveau pour améliorer la précision de la division de la lésion. En appliquant cet enregistrement, les auteurs peuvent prouver la présence de correspondance entre l'hémisphère sain et la moitié infectée, permettant la comparaison directe et l'identification des ravageurs. Les résultats suggèrent qu'en exploitant la symétrie binaire et l'enregistrement réfléchissant, la méthode proposée peut améliorer considérablement la précision de la division de la lésion cérébrale. En alignant les hémisphères sains et affectés, la méthode permet la comparaison directe et l'identification des ravageurs, ce qui donne des résultats de fragmentation plus précis et plus fiables .Ce modèle a produit une précision 82 % [45].

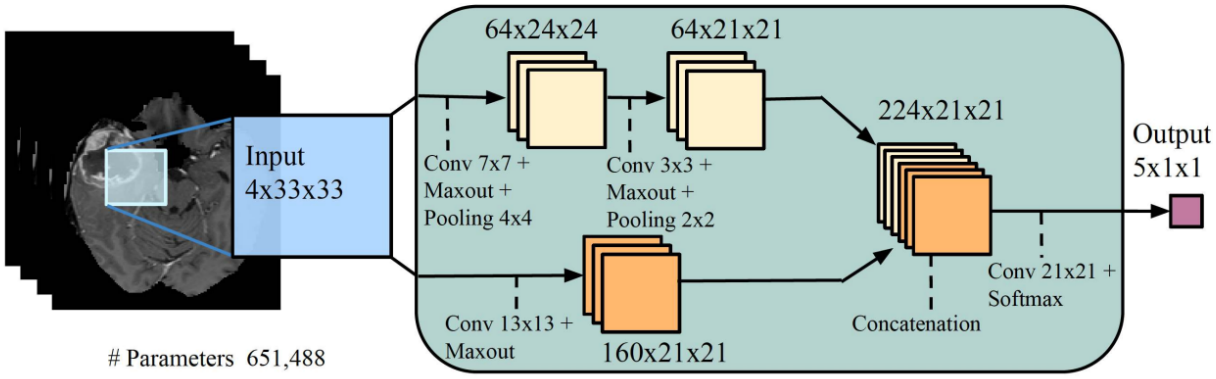


FIGURE 2.17 – Architecture du modèle.

— **Au-delà des CNN : exploiter d'autres symétries inhérentes à la segmentation des images médicales [39] :**

Dans l'article "Beyond CNNs : Exploiting More Seminars Inherent in the Division of Medical Images", les auteurs suggèrent de nouvelles façons de tirer parti des symétries inhérentes supplémentaires dans la division des images médicales. Ces méthodes visent à améliorer la précision et l'efficacité des modèles de fragmentation au-delà de l'utilisation traditionnelle des NC. Ces réseaux sont conçus pour capturer et exploiter les symétries de spin, permettant une fragmentation plus efficace des structures analogues dans les images médicales. Les auteurs suggèrent d'intégrer des réseaux neuronaux équivalents à la pensée qui peuvent capturer et utiliser efficacement les symétries de réflexion. Les

auteurs proposent un réseau neuronal équivalent hiérarchique qui peut capturer et exploiter ces symétries hiérarchiques. En considérant les informations de symétrie multi-bandes, le modèle de fragmentation peut mieux identifier les structures hiérarchiques complexes dans les images médicales. En général, les méthodes proposées dans l'article visent à outrepasser les limites des sociétés non nationales en exploitant des symétries sous-jacentes supplémentaires dans la fragmentation des images médicales. Ce modèle a produit une précision 87,23 %, score F1 avec 87,51 % et distance de Hausdorff la plus courte avec 24,79 pixels [39].

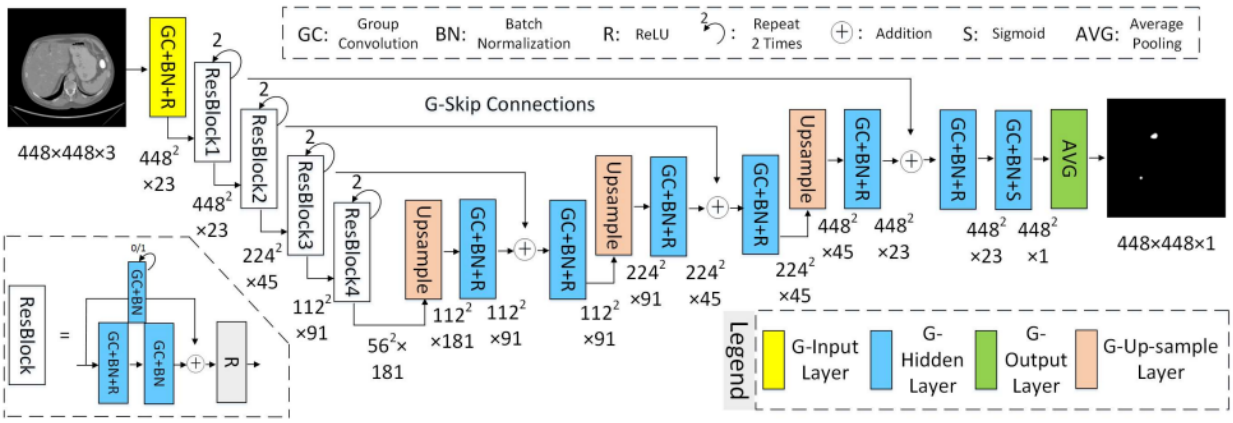


FIGURE 2.18 – Architecture du modèle.

— **Exploitation de la symétrie cyclique dans les réseaux de neurones convolutifs [20] :** proposed a rotation equivariant network by directly rotating each feature map at every layer, but this would substantially increase memory requirements. Recently, some rotated equivariant networks [8-9] rotate convolutional kernels to achieve local symmetries among all the orientations. As a result, the network architectures are usually very shallow and the noise generated from the intermediate layers makes it difficult to maintain global equivariance.[20]

2.11 Conclusion

Dans ce chapitre , Nous avons discuté des approches classiques de segmentation ainsi que des méthodes fondées sur l'apprentissage profond et de certains CNN bien connus et aussi la symétries . Enfin, travail connexe pour terminer le chapitre.

Chapitre 3

Conception

3.1 Introduction

Dans ce chapitre, nous décrivons notre architecture de réseau neuronal profond et les procédures requises pour la mise en place de la base de données. La performance des résultats obtenus, qui sera discutée au chapitre 4, est fortement influencée par ce dernier facteur.

3.2 Architecture général

La figure 3.1 ci-dessous représente la structure générale de notre système :

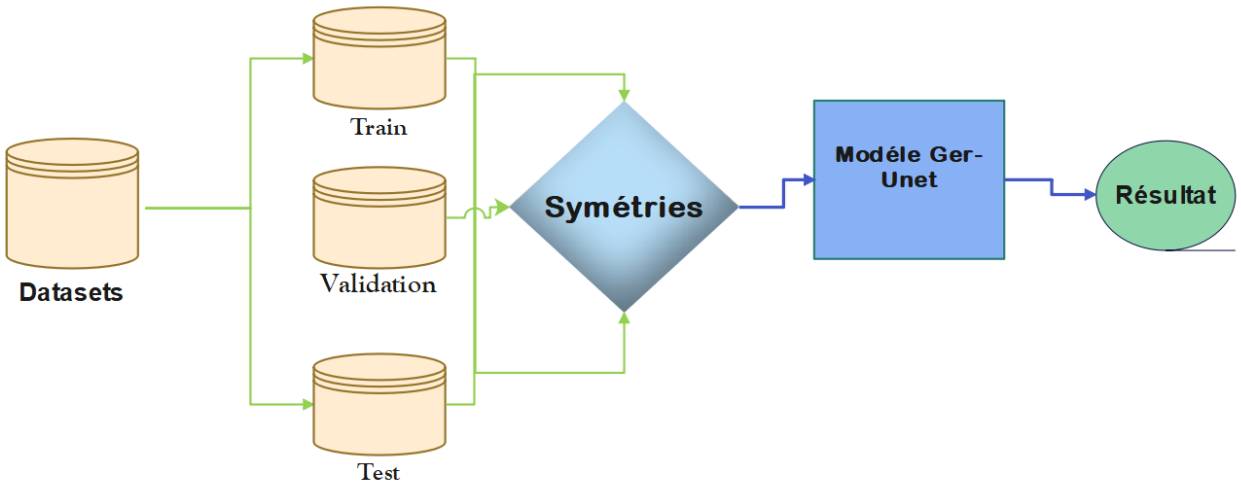


FIGURE 3.1 – L'architecture général du système

Afin de construire un modèle d'apprentissage en profondeur et de détecter et catégoriser les tumeurs,Le schéma global est le suivant :

1. Le système commence par recueillir toutes les données,y appliquer un prétraitement, puis les diviser.
2. Nous entrons ensuite les données divisées pour appliqué dans symétries et le modèle (G-unet), qui produit un resultat précis.

L'objectif

Ce travail est que seuls les CNN normaux exploitent l'invariance translationnelle, ignorant les symétries plus réalistes trouvées dans les images médicales telles que les rotations et les réflexions. Pour résoudre ce problème, nous avons travaillé pour introduire un modèle de trame de segmentation égale en codant ces symétries inhérentes pour trouver des représentations plus précises.

3.3 Architecture détaillée

Cette section aborde efficacement le cœur de notre travail avec sa conception détaillée étape par étape, en commençant par le chargement et la collecte des données et en terminant par la mise en œuvre (voir la figure 3.2).

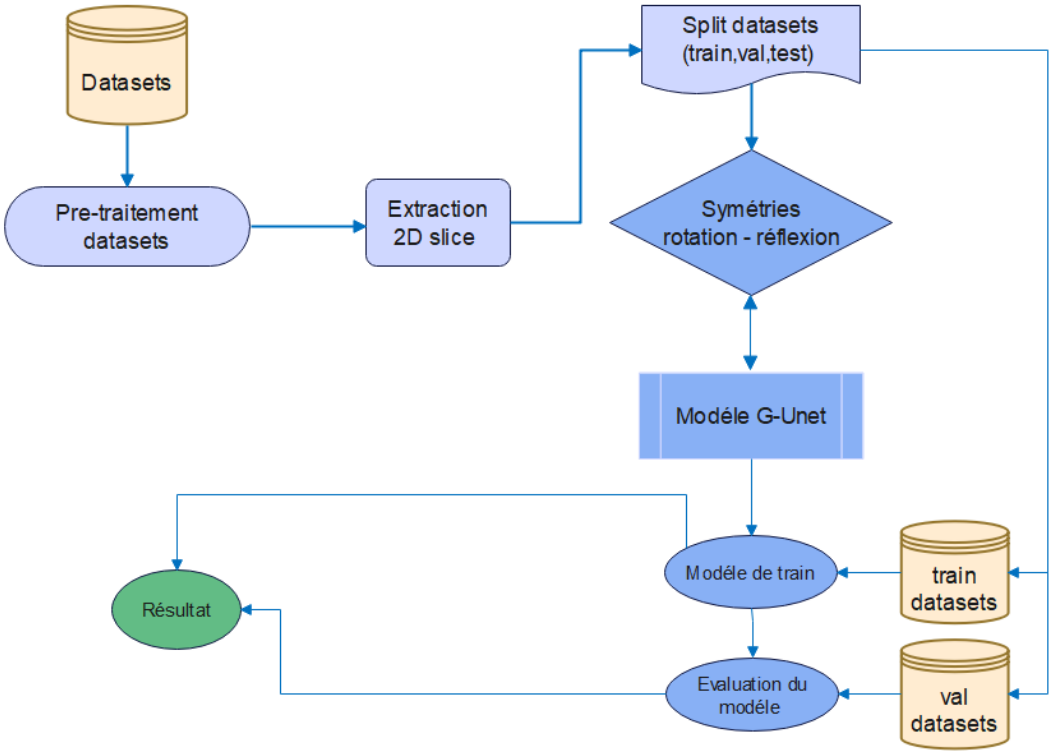


FIGURE 3.2 – L'architecture détaillée du système

3.3.1 La collecte de données

La phase de chargement, y compris la collecte de données pour la formation et les tests, est la première étape de notre système d'apprentissage. L'ensemble de données BRATS (Brain Tumor Segmentation) est un ensemble de données bien connu dans l'analyse d'images médicales qui consiste en des IRM de tumeurs cérébrales avec des étiquettes de segmentation pour les régions sous-tumorales. L'ensemble de données BRATS2019 est la version 2019 de l'ensemble de données BRATS et contient 335 examens IRM, chacun avec quatre modalités (T1, T1-ce, T2 et FLAIR) lle est disponible sous forme de fichiers NIfTI(.nii.gz) et les étiquettes de segmentation correspondantes pour la tumeur entière, le noyau de la tumeur et les régions tumorales rehaussées.

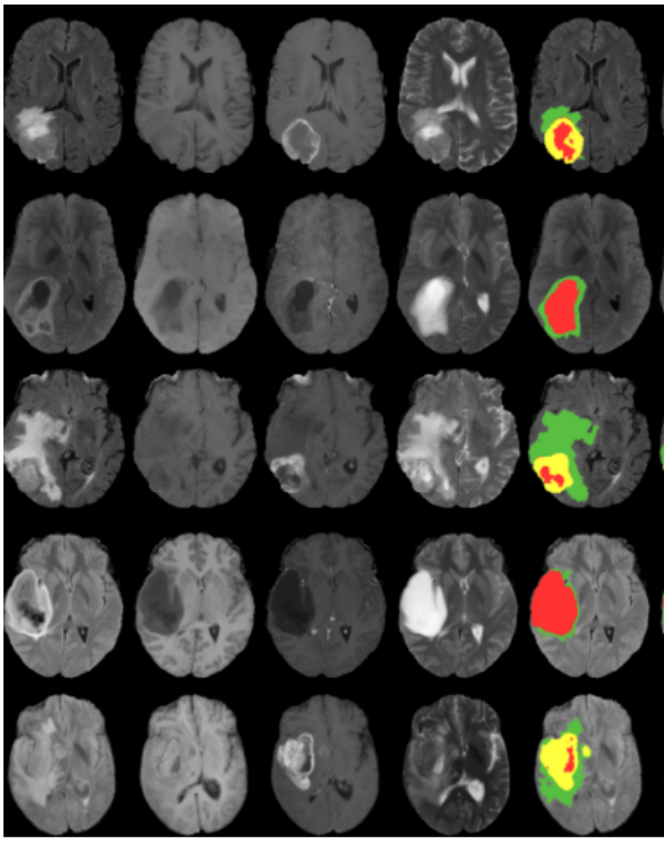


FIGURE 3.3 – Exemple sur l'ensemble de données BraTS 2019.

3.3.2 Pré traitement

Certaines méthodes de traitement d'image peuvent être appliquées pour préparer les données avant de les alimenter à notre modèle. Dans notre travail, les étapes de prétraitement comprennent la transformation affine , le peeling du crâne , et le débruitage, le redimensionnement et la normalisation .

— **Division Base de données :**À ce partie , nous utilisons l'ensemble de données BraTS 2019 , qui comprend des images de 335 patients (259 HGG et 76 LGG) et est disponible sous forme de fichiers NIfTI (.ni.gz). Les dimensions de la base de données sont 240 * 240 * 155 (flèche, couronne, axe), chaque élément contient 155 pièces, qui sont en fait des images IRM. chaque élément a 155 tranches sont en fait des images IRM. Quatre séquences d'acquisition ont été réalisées sur chaque patient :

— Flair (faild Attenuated Inversion Recovery)

— T1 séquence pondérée

— T1Ce séquence post-contraste

— T2 séquence pondérée

— segmentation mask

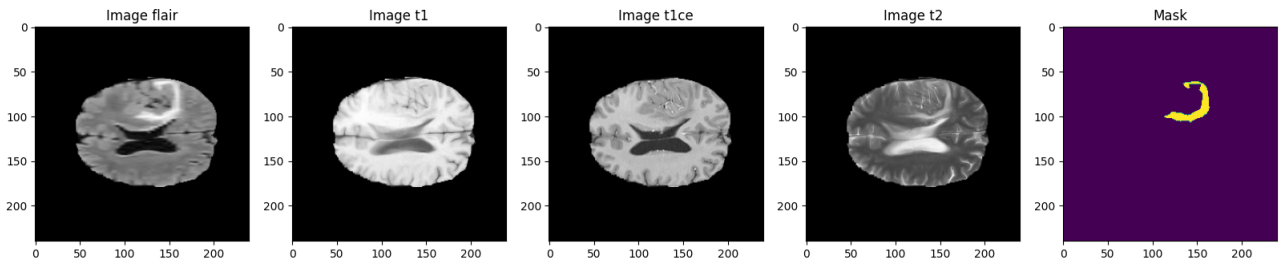


FIGURE 3.4 – Quatre modalités (T1, T1-Gd, T2 et FLAIR) pour la segmentation .

— **le dépouillement du crâne (skull stripping) :**Le décapage crânien est une procédure de prétraitement typique dans l'analyse de neuroimagerie parce qu'il aide à isoler et à se concentrer sur les zones cérébrales pertinentes. Le décapage crânien peut être utilisé pour améliorer la précision et la fiabilité des résultats des tâches d'analyse ultérieures effectuées sur les volumes IRM de l'ensemble de données BraTS 2019. cette phase a

été utilisée dans lequel Toutes les valeurs inférieures au seuil (tissus de fond et non de pluie) sont réglées à zéro dans l'opération de seuillage, qui est essentiellement appliquée à chaque volume d'IRM. Seule la région du cerveau est visible dans les photos produites par cette technique, qui dépouille le crâne .

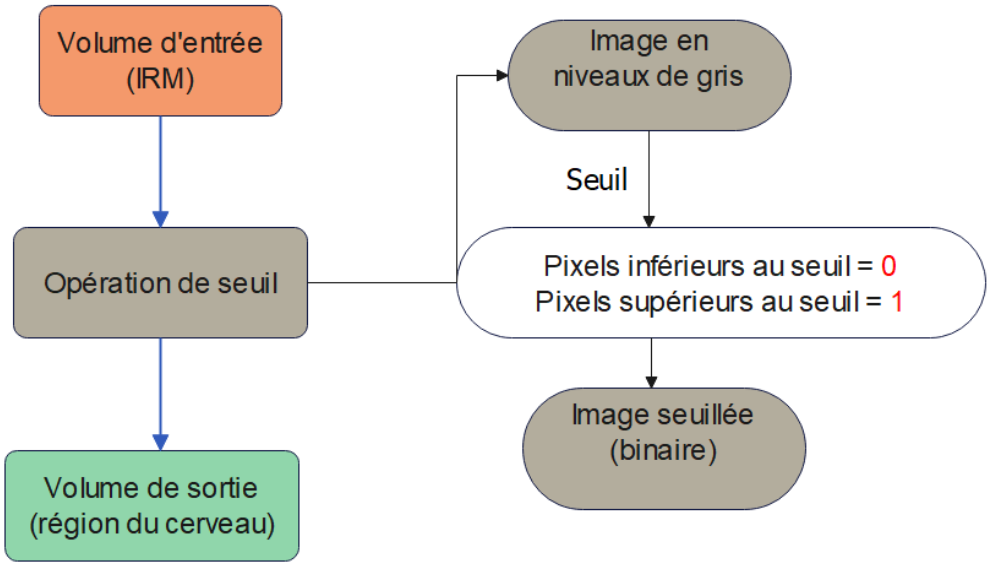


FIGURE 3.5 – schéma de skull stripping

— **Redimensionner l'image :**l'ensemble de Les données contiennent des images de différentes tailles, ce qui pourrait rendre l'architecture inexacte. En conséquence, nous redimensionnons chaque image de notre modèle à (128 x 128).

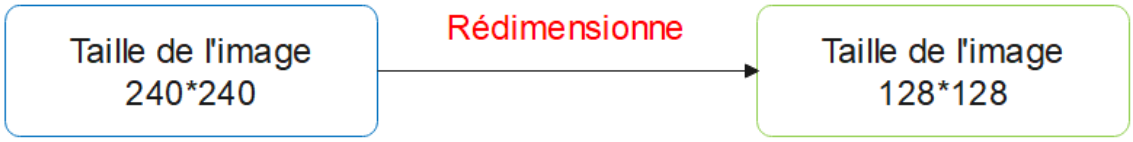


FIGURE 3.6 – Redimensionne de l'image

— **Normalisation d'image :** Il s'agit d'une procédure de normalisation de la plage des variables indépendantes ou des caractéristiques des données, ou du changement de la plage des valeurs d'intensité des pixels. Il est nécessaire d'ajuster la plage des valeurs d'intensité de l'image sur une échelle de [0, 1]selon l'équation :

$$F(x) = \frac{x - \min(x)}{\max(x) - \min(x)}$$

— "x" représente la valeur d'intensité d'un pixel spécifique dans l'image d'origine.

— "min(x)" correspond à la valeur minimale d'intensité des pixels dans l'image.

— "max(x)" correspond à la valeur maximale d'intensité des pixels dans l'image.

La nouvelle valeur pour un pixel donné est calculée à l'aide de la formule F(x) fournie. Elle déplace la plage de valeurs vers zéro en convertissant la valeur d'intensité minimale des pixels de l'image à l'intensité du pixel lui-même. Ensuite, cette valeur est divisée par la différence entre les valeurs d'intensité de pixel maximum et minimum, permettant d'étendre la plage de valeurs jusqu'à un maximum de 1.

3.4 Conception détaillée du Symétries et Modèle proposer

3.4.1 Symétrie

Pour compléter les données de formation, le modèle comprend trois types de symétries : rotation vers la gauche, rotation vers la droite et réflexion. Ces opérations sont effectuées sur la sortie de la troisième couche convolutive du chemin de passation de marché . Nous allons expliquer quelque methodes de symetries :

— **rotation**

Lorsqu'il est utilisé dans ce contexte,Le terme « opération de rotation» fait référence à la rotation d'une image d'entrée de 90 degrés vers la gauche ou la droite. La diversité des données d'entrée fournies par ce processus peut faciliter l'augmentation des données en améliorant la capacité du modèle à généraliser et à gérer diverses orientations.

Son principe

Modifier l'orientation de l'image de 90 degrés, soit dans une direction horaire ou antihoraire. Cela signifie que l'image initiale est tournée de sorte que ce qui était initialement en haut devient le côté droit de l'image tournée.

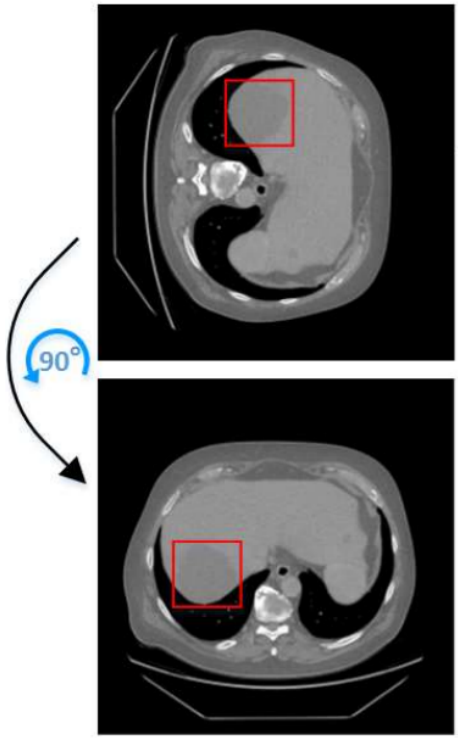


FIGURE 3.7 – Exemple de rotation 90.

— **La réflexion**

La réflexion, également appelée symétrie miroir, est l'idée de faire tourner l'axe de symétrie d'un objet ou d'une image. Cela indique que certains aspects de l'élément ou de l'image sont reflétés de manière à produire une image miroir.

L'opération de réflexion est fréquemment utilisée dans le contexte de l'apprentissage automatique pour élargir la diversité des données d'apprentissage et améliorer la capacité du modèle à détecter des modèles dans diverses orientations spatiales.

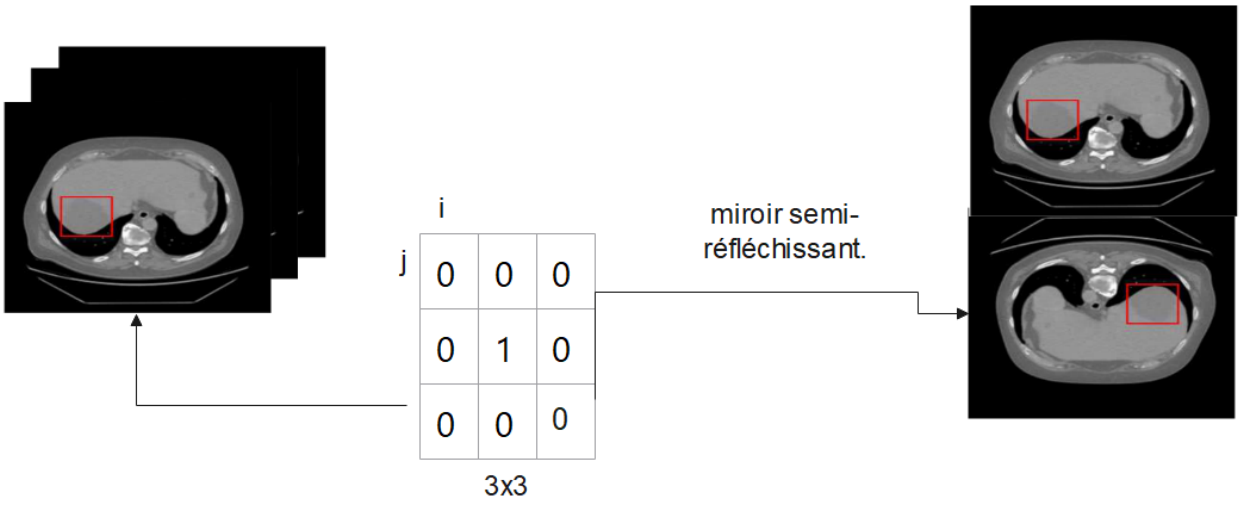


FIGURE 3.8 – Exemple de reflexion

Opérations sur la réflexion de la fenêtre :

1. Initialiser une nouvelle matrice pour la réflexion de la fenêtre.
2. Parcourir les lignes et les colonnes de l'image cérébrale : Pour chaque pixel (i, j) dans l'image cérébrale :
   — Vérifier si le pixel (i, j) correspond à la position centrale de la fenêtre 3x3.
   — Si oui, copier les valeurs des pixels voisins de la fenêtre dans la nouvelle matrice de réflexion.
   — Sinon, continuer le parcours sans rien faire.

3.4.2 G-Unet

L'architecture proposée est composée :

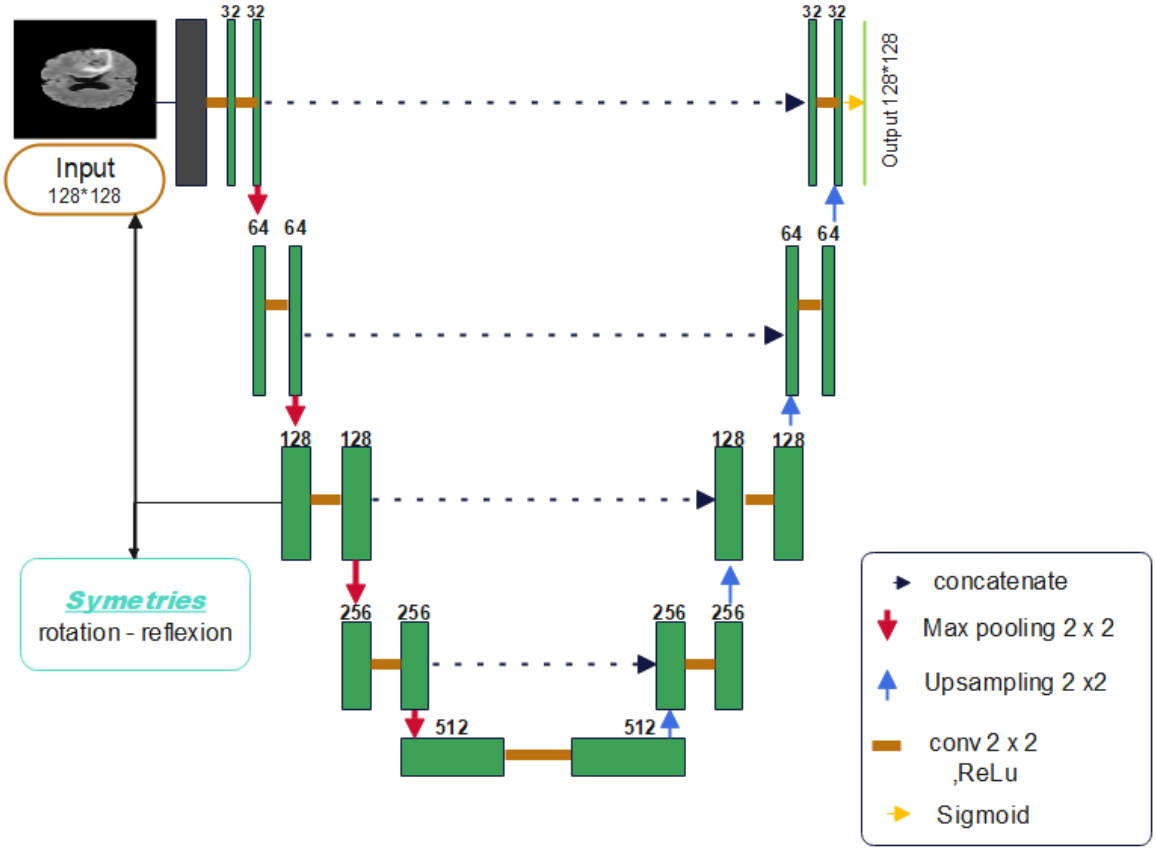


FIGURE 3.9 – L'architecture du G-net

— Le modèle utilise une image 2D en entrée et produit un masque de même taille mais avec des valeurs de pixel comprises entre 0 et 1.

— L'architecture se compose d'une partie codeur et d'une partie décodeur.

— L'encodeur réduit l'image d'entrée à une représentation basse résolution, tandis que le décodeur agrandit la représentation basse résolution à sa taille d'origine. Ensemble, ces deux composants constituent le modèle. L'encodeur est composé de quatre blocs de couche convolutive, chacun étant suivi d'une couche de regroupement maximal. Quatre blocs de couches convolutionnelles constituent le décodeur, et chaque bloc est suivi d'une couche de suréchantillonnage qui double la taille de la carte des caractéristiques. Le décodeur peut retrouver l'information spatiale perdue lors de la phase de contraction car chaque bloc du décodeur est concaténé avec le bloc correspondant de l'encodeur.

— le chemin d'expansion est défini comme suit :

1. Upsampling est effectué à l'aide d'une interpolation bilinéaire suivie d'une couche convolutive 2x2 avec activation ReLU. Les cartes d'entités suréchantillonnées sont redimensionnées à des dimensions spécifiques .
2. Les cartes d'entités Upsampling sont concaténées avec les cartes d'entités correspondantes du chemin de contraction, les versions pivotées des cartes d'entités et leurs réflexions.
3. Deux couches convolutionnelles avec 256 filtres de taille 3x3 et activation ReLU sont appliquées aux cartes de caractéristiques concaténées.
4. Ce processus est répété pour les couches suivantes, en réduisant progressivement le nombre de filtres (128 et 64).
5. La couche finale consiste en un Upsampling 2x2 suivi d'une convolution 1x1 avec activation sigmoïde pour générer le masque de segmentation.

— Le modèle applique ensuite une couche convolutive 1x1 avec une fonction d'activation sigmoïde à la sortie du décodeur (conv9) pour créer un masque de probabilité. L'entropie croisée binaire est la fonction de perte utilisée pour comparer le masque anticipé au masque réel.

Voici les principales opérations qui contrôlent les différentes couches de notre réseau GerNet nouvellement créé :

1. **Des couches groupe convolutives**

   En particulier, l'opération de convolution de groupe divise les cartes de caractéristiques d'entrée en plusieurs groupes, effectue une opération de convolution sur chaque groupe en utilisant un ensemble séparé de filtres, puis concatène les cartes de caractéristiques de sortie pour chaque groupe. Le paramètre "groupes" de la couche Conv2D contrôle le nombre de groupes.

2. **Des couches groupe up-sampling**La couche "UpSampling2D" dans le modèle UNet est un type particulier de couche qui répète les lignes et les colonnes du tenseur d'entrée pour suréchantillonner les cartes d'entités. Dans les travaux nécessitant la préservation de petites caractéristiques, comme la segmentation d'image, cela permet une amélioration de la résolution spatiale des cartes de caractéristiques. La couche "UpSampling2D"

duplique les valeurs des pixels voisins dans les cartes d'entités en utilisant une technique du plus proche voisin. Vous pouvez spécifier la taille du suréchantillonnage en donnant un tuple au paramètre "taille" de la couche. L'objectif du UpSampling d'une image est de créer une version plus grande et plus dense en pixels de l'image d'origine. Pour ce faire, des pixels supplémentaires sont souvent ajoutés à l'image pour améliorer sa taille.

3. **Les Fonctions D'Activation**La fonction d'activation la plus courante dans pratiquement tous les NCN est la fonction RELU (unité linéaire rectifiée) (figure 3.3). Elle n'ouvre la porte que lorsque l'entrée est plus élevée, donc un montant précis.

FIGURE 3.10 – Un graphique d'application pour la fonction d'activation Relu.

4. **Le MaxPooling :**Cette couche, qui est la deuxième dans le bloc convolutif, est utilisée pour fournir une forme de représentation abstraite pour aider dans le processus de Sur-apprentissage. En outre, en réduisant le nombre de paramètres à apprendre et en garantissant une invariance cruciale de la traduction à la représentation interne, elle a diminué le coût du calcul. Dans notre cas, nous profitons de ses avantages pour réduire la taille des images en réduisant le nombre de pixels dans la sortie de la couche précédente révolutionnaire afin d'extraire des caractéristiques de bas niveau comme les bordures et les points.

3.5 Conclusion

Dans ce chapitre,Nous avons discuté du concept global et des modules de CNN pour classer les traumatismes chez les personnes malades . en commençant par la conception globale de notre modèle.

Chapitre 4

Implementation et résultats

4.1 Introduction

Dans ce chapitre, on présente le développement détaillé de notre architecture de réseau de neurones profond pour la prédiction du maladie tumeur proposé au chapitre 3 de ce mémoire. Cette partie comprend aussi :

— Les Environnements et outils utilisées.

— L'implémentation du modèle.

— La comparaison des résultats.

4.2 Environnements et outils de développement

Pour mettre cette application en action, les matériaux suivants ont été utilisés :

— **Processeur :**Intel(R) Core(TM) i5-5300U CPU @ 2.30GHz 2.29 GHz.

— **RAM :**8.00 Go.

— **Disque dur :**256Go.

— **Graphique :**HD Graphics .

4.2.1 Bibliothèques et outils de développement

Cette section fournira des environnements de programmation, des outils et des bibliothèques pour la programmation qui peut être exécutée sur divers IDE (périphériques d'entrée/sortie).

— **Python :**Python est un langage de programmation interprétable, multiparamètre et multiplateforme. Il préconise une programmation impérative structurée, fonctionnelle et orientée objet. Grâce à des bibliothèques spécialisées, c'est un langage utilisable dans toutes les situations et adapté à tout type d'utilisation. Cependant, il est surtout utilisé comme langage de script pour automatiser des tâches simples mais laborieuses[31].

FIGURE 4.1 – Python logo

— **TensorFlow :**eIl s'agit d'un ensemble de logiciels open source d'intelligence artificielle et d'apprentissage automatique. Il peut être utilisé à de nombreuses fins différentes, bien qu'il se concentre principalement sur le développement et la pénétration de réseaux de neurones profonds[4].

FIGURE 4.2 – Tenserflow logo

— **Keras :**La bibliothèque Keras permet une interaction avec Tensorflow et d'autres réseaux de neurones profonds et algorithmes d'apprentissage automatique. Il est conçu pour permettre une expérimentation rapide des réseaux de neurones profonds, en mettant l'accent sur son ergonomie, sa modularité et son évolutivité. Il a été créé dans le cadre du projet Open NeuroElectronic Intelligent Robot Operating System (ONEIROS). Son auteur

est François Cholet[16].

FIGURE 4.3 – Keras logo

— **NiBabel :**Le successeur de PyNIIfTI est Nibabel. Ce package est une bibliothèque créée pour le langage de programmation Python qui permet aux utilisateurs d'accéder à certains formats de fichiers médicaux et de neuroimagerie largement utilisés, tels que les fichiers Nifti. Il fournit différentes classes de formats d'image, permet un accès complet ou sélectif aux méta-informations et rend les données d'image accessibles via les tables NumPy[21].

FIGURE 4.4 – NiBabel logo

— **MatplotLib :**est un package graphique pour Python avec NumPy, l'extension de mathématiques numériques Python. Il fournit une API orientée objet pour intégrer des tracés dans des applications utilisant des kits d'outils GUI tels que Tkinter, wxPython, Qt ou GTK[25].

FIGURE 4.5 – Matplotlib logo

— **Numpy :**La bibliothèque NumPy de Python est utilisée pour travailler avec les tables. En outre, elle a des outils pour travailler avec les matrices, les transformations de Fourier et l'algèbre linéaire. Travis Oliphant a inventé NumPy en 2005. Vous êtes libre d'utiliser ce projet car il est open source[38].

FIGURE 4.6 – NumPy

— **OpenCv :**(Open Source Computer Vision Library) : est un ensemble de logiciels libres pour la vision assistée par ordinateur et l'apprentissage automatique. OpenCV a été développé pour offrir une infrastructure partagée pour les applications de vision par ordinateur et pour permettre aux produits commerciaux d'intégrer plus rapidement la perception de la machine[29].

FIGURE 4.7 – OpenCv logo

— **Google colaboratory :**Nous utilisons la plate-forme cloud Google Colab pour cette tâche parce que le développement d'un modèle d'apprentissage en profondeur peut placer un

fardeau important sur le CPU et le GPU. Colaboratory est une initiative de recherche lancée par Google pour aider à diffuser les connaissances sur l'éducation et la recherche dans l'apprentissage automatique en fournissant à ses utilisateurs des ressources à utiliser. C'est un environnement de notebook Jupyter qui ne nécessite aucune configuration à utiliser. Et il fonctionne entièrement dans le cloud[13].

FIGURE 4.8 – Google colab Logo

4.3 Préparation des données

Après avoir recherché et téléchargé toutes les données, nous avons divisé la base de données en trois groupes distincts : formation, validation et test, en utilisant une répartition de 70 % de formation, 20 % de validation et 10 % de test. Le rapport des échantillons distribués dans chaque ensemble est déterminé par des expériences. Nous avons utilisé la fonction Python prédéfinie "split folders.ratio".

4.3.1 Base des données

```
import nibabel as nib
import numpy as np
from skimage import exposure
from nilearn.image import math_img

# Load the scans
```

```
t1 = nib.load('/content/MICCAI_BraTS_2019_Data_Training/HGG/
    BraTS19_2013_10_1/BraTS19_2013_10_1_t1.nii').get_fdata()
t2 = nib.load('/content/MICCAI_BraTS_2019_Data_Training/HGG/
    BraTS19_2013_10_1/BraTS19_2013_10_1_t2.nii').get_fdata()
t1ce = nib.load('/content/MICCAI_BraTS_2019_Data_Training/HGG/
    BraTS19_2013_10_1/BraTS19_2013_10_1_t1ce.nii').get_fdata()
flair = nib.load('/content/MICCAI_BraTS_2019_Data_Training/HGG/
    BraTS19_2013_10_1/BraTS19_2013_10_1_flair.nii').get_fdata()

# Load the segmentation map
seg = nib.load('/content/MICCAI_BraTS_2019_Data_Training/HGG/
    BraTS19_2013_10_1/BraTS19_2013_10_1_seg.nii').get_fdata()
```

Listing 4.1 – Chargement base de données

4.3.2 Décapage du crâne (Skull-stripping)

Après l'enregistrement, une étape de dépouillement du crâne a été effectuée pour supprimer les informations non pertinentes des images. La clé ici était de trouver la valeur appropriée pour l'hyper-paramètre principal : le seuil d'intensité fractionnaire.

```
# Skull stripping
t1_data = math_img("img1 * (img1 > 0)", img1=nib.load('/content/
    MICCAI_BraTS_2019_Data_Training/HGG/BraTS19_2013_10_1/
    BraTS19_2013_10_1_t1.nii')).get_fdata().astype(np.float64)
t1ce_data = math_img("img1 * (img1 > 0)", img1=nib.load('/content/
    MICCAI_BraTS_2019_Data_Training/HGG/BraTS19_2013_10_1/
    BraTS19_2013_10_1_t1ce.nii')).get_fdata().astype(np.float64)
t2_data = math_img("img1 * (img1 > 0)", img1=nib.load('/content/
    MICCAI_BraTS_2019_Data_Training/HGG/BraTS19_2013_10_1/
    BraTS19_2013_10_1_t2.nii')).get_fdata().astype(np.float64)
flair_data = math_img("img1 * (img1 > 0)", img1=nib.load('/content/
    MICCAI_BraTS_2019_Data_Training/HGG/BraTS19_2013_10_1/
    BraTS19_2013_10_1_flair.nii')).get_fdata().astype(np.float64)
```

Listing 4.2 – Décapage du crâne (Skull-stripping)

4.3.3 Normalisation de l'intensité

Le code effectue une normalisation de l'intensité des images médicales T1, T1 avec contraste (T1CE), T2 et FLAIR en utilisant la fonction exposure.equalizehist de la bibliothèque skimage de Python.

```

# Intensity normalization
t1_data = exposure.equalize_hist(t1_data)
t1ce_data = exposure.equalize_hist(t1ce_data)
t2_data = exposure.equalize_hist(t2_data)
flair_data = exposure.equalize_hist(flair_data)

###
seg_data = nib.load('/content/MICCAI_BraTS_2019_Data_Training/HGG/
    BraTS19_2013_10_1/BraTS19_2013_10_1_flair.nii').get_fdata().astype(np.
    int16)

seg_data[seg_data == 4] = 3  # Remove the background label

# Convert to binary format
seg_processed = np.zeros_like(seg_data)
seg_processed[seg_data > 0] = 1
```

Listing 4.3 – Normalisation de l'intensité

4.3.4 Extraction de 2D slice

Ce code extrait des coupes 2D de chaque modalité d'imagerie et de la segmentation à partir des données d'imagerie volumétriques, en utilisant la fonction extract 2d slices.

La fonction extract 2d slices prend en entrée un tableau tridimensionnel data, représentant une image volumétrique. Elle parcourt ensuite les tranches de données le long de la dimension z (profondeur) de l'image volumétrique en utilisant une boucle (for) et pour chaque tranche elle extrait les données des deux dimensions x et y pour produire une coupe 2D, qui est ensuite ajoutée à une liste de coupes 2D appelée slices. La fonction retourne la liste de coupes 2D.

```

def extract_2d_slices(data):
    slices = []
    for i in range(data.shape[2]):
        slice_data = data[:, :, i]
        slices.append(slice_data)
    return slices

# Extract 2D slices from each modality and the segmentation label
t1_slices = extract_2d_slices(t1_data)
t1ce_slices = extract_2d_slices(t1ce_data)
```

```
t2_slices = extract_2d_slices(t2_data)
flair_slices = extract_2d_slices(flair_data)
seg_slices = extract_2d_slices(seg_processed)
```

Listing 4.4 – Extraction de 2D slice

4.3.5 Entraînement et validation de données

```

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(combined_slices,
    seg_combined_slices, random_state=42,  test_size=0.15)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train,
    random_state=42, test_size=0.15)
X_train = np.nan_to_num(X_train, nan=0.0)


X_train = np.array(X_train)
X_val = np.array(X_val)
X_test = np.array(X_test)


X_train = np.nan_to_num(X_train, nan=0.0)

X_train = X_train.astype('float32') / 255.
X_val = X_val.astype('float32') / 255.
X_test = X_test.astype('float32') / 255.
```

Listing 4.5 – Entraînement et validation de données

4.3.6 Paramètres d'augmentation de données

Le code spécifie plusieurs paramètres d'augmentation pour l'augmentation des données dans un dictionnaire appelé augment-params. Ces paramètres comprennent une fonction de prétraitement, une plage de luminosité, un mode de remplissage, une plage de décalage de canal, une plage de cisaillement, une plage de zoom, un retournement horizontal et un retournement vertical. Les types et le degré d'augmentation qui seront appliqués aux images d'entraînement sont déterminés par ces paramètres.

```

from keras.preprocessing.image import ImageDataGenerator
augment_params = dict(
    rotation_range=90,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    vertical_flip=True,
    brightness_range=[0.5, 1.5],
    fill_mode='reflect',
    channel_shift_range=0.1,
    preprocessing_function=lambda x: x + np.random.normal(loc=0.0, scale
   =0.01, size=x.shape)
)



image_datagen = ImageDataGenerator(**augment_params)
mask_datagen = ImageDataGenerator(**augment_params)
```

Listing 4.6 – Paramètres d'augmentation de données:

4.4 Construire notre modèle CNN

Le modèle fourni est l'architecture U-Net pour la segmentation d'image. Il se compose d'un passe encodeur et d'un passe décodeur. L'encodeur capture les informations contextuelles à travers des couches convolutionnelles et de regroupement, et le décodeur reconstruit la sortie segmentée. Les sauts de connexion préservent les informations spatiales. Ce modèle inclut des méthodes pour symétries qui applique une transformation d'image et améliore la généralisation en ajoutant un terme de régularisation à la fonction de perte.

```

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.layers import Input, UpSampling2D

```

```
# Rest of the code...

def rotate_left_tf(x):
    return tf.image.rot90(x, k=1)

def rotate_right_tf(x):
    return tf.image.rot90(x, k=3)

def reflect_tf(x):
    return tf.image.flip_left_right(x)

def get_Unet(input_shape=(128, 128, 2), dropout_rate=0.5):
    # Input layer
    inputs = Input(shape=input_shape)

    # Preprocess the input
    preprocessed_inputs = layers.Conv2D(32, 3, activation='relu', padding=
    'same')(inputs)

    # Operations for symmetry
    rot_left = layers.Lambda(rotate_left_tf)(preprocessed_inputs)
    rot_right = layers.Lambda(rotate_right_tf)(preprocessed_inputs)
    ref = layers.Lambda(reflect_tf)(preprocessed_inputs)

    # Contracting path (encoder)
    conv1 = layers.Conv2D(32, 3, activation='relu', padding='same')(
    preprocessed_inputs)
    conv1 = layers.Conv2D(32, 3, activation='relu', padding='same')(conv1)
    pool1 = layers.MaxPooling2D(pool_size=(2, 2))(conv1)

    conv2 = layers.Conv2D(64, 3, activation='relu', padding='same')(pool1)
    conv2 = layers.Conv2D(64, 3, activation='relu', padding='same')(conv2)
    pool2 = layers.MaxPooling2D(pool_size=(2, 2))(conv2)

    conv3 = layers.Conv2D(128, 3, activation='relu', padding='same')(pool2
    )
    conv3 = layers.Conv2D(128, 3, activation='relu', padding='same')(conv3
    )
    conv3 = layers.Conv2D(64, 1, activation='relu')(conv3)
    pool3 = layers.MaxPooling2D(pool_size=(2, 2))(conv3)

    conv4 = layers.Conv2D(256, 3, activation='relu', padding='same')(pool3
    )
    conv4 = layers.Conv2D(256, 3, activation='relu', padding='same')(conv4
    )
```

```
    conv4 = layers.Dropout(dropout_rate)(conv4)
    conv4 = layers.Conv2D(128, 1, activation='relu')(conv4)
    pool4 = layers.MaxPooling2D(pool_size=(2, 2))(conv4)


# Expanding path (decoder)
    up5 = layers.Conv2D(512, 2, activation='relu', padding='same')(UpSampling2D(size=(2, 2))(pool4))
    up5_resized = layers.experimental.preprocessing.Resizing(16, 16)(up5)
    merge5 = layers.concatenate([conv4, up5_resized, rotate_right_tf(conv4), reflect_tf(conv4)], axis=3)
    conv5 = layers.Conv2D(256, 3, activation='relu', padding='same')(merge5)
    conv5 = layers.Conv2D(256, 3, activation='relu', padding='same')(conv5)

    up6 = layers.Conv2D(256, 2, activation='relu', padding='same')(UpSampling2D(size=(2, 2))(conv5))
    up6_resized = layers.experimental.preprocessing.Resizing(32, 32)(up6)
    merge6 = layers.concatenate([conv3, up6_resized, rotate_left_tf(conv3)], axis=3)
    conv6 = layers.Conv2D(128, 3, activation='relu', padding='same')(merge6)
    conv6 = layers.Conv2D(128, 3, activation='relu', padding='same')(conv6)

    up7 = layers.Conv2D(128, 2, activation='relu', padding='same')(UpSampling2D(size=(2, 2))(conv6))
    up7_resized = layers.experimental.preprocessing.Resizing(64, 64)(up7)
    merge7 = layers.concatenate([conv2, up7_resized, rotate_right_tf(conv2)], axis=3)
    conv7 = layers.Conv2D(64, 3, activation='relu', padding='same')(merge7)
    conv7 = layers.Conv2D(64, 3, activation='relu', padding='same')(conv7)

    up8 = layers.Conv2D(64, 2, activation='relu', padding='same')(UpSampling2D(size=(2, 2))(conv7))
    up8_resized = layers.experimental.preprocessing.Resizing(128, 128)(up8)
    merge8 = layers.concatenate([conv1, up8_resized, rotate_left_tf(conv1), reflect_tf(conv1)], axis=3)
    conv8 = layers.Conv2D(32, 3, activation='relu', padding='same')(merge8)
    conv8 = layers.Conv2D(32, 3, activation='relu', padding='same')(conv8)

```

```
76
77      # Output layer
78      outputs = layers.Conv2D(4, 1, activation='sigmoid')(conv8)
79
80      # Create the model
81      model = keras.Model(inputs=inputs, outputs=outputs)
82
83      return model
```

Listing 4.7 – Modèle G-Unet et symétries:

4.5 L'implementation du symétrie et modèle G-Unet

Nous avons expérimenté un certain nombre de configurations lors du développement et de la configuration de notre modèle d'apprentissage profond (CNN), en utilisant des expériences empiriques. Nous présenterons les paramètres de notre architecture dans la Figure 4.9.

```
Model: "model_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_5 (InputLayer)        [(None, 128, 128, 2       0
                             )]

 conv2d_97 (Conv2D)          (None, 128, 128, 32       608
                             )

 conv2d_98 (Conv2D)          (None, 128, 128, 32       9248
                             )

 conv2d_99 (Conv2D)          (None, 128, 128, 32       9248
                             )

 max_pooling2d_16 (MaxPooling2D  (None, 64, 64, 32)    0
 )

 conv2d_100 (Conv2D)         (None, 64, 64, 64)        18496

 conv2d_101 (Conv2D)         (None, 64, 64, 64)        36928

 max_pooling2d_17 (MaxPooling2D  (None, 32, 32, 64)    0
 )

 conv2d_102 (Conv2D)         (None, 32, 32, 128)       73856

 conv2d_103 (Conv2D)         (None, 32, 32, 128)       147584

 conv2d_104 (Conv2D)         (None, 32, 32, 64)        8256

 max_pooling2d_18 (MaxPooling2D  (None, 16, 16, 64)    0
 )

 conv2d_105 (Conv2D)         (None, 16, 16, 256)       147712

 conv2d_106 (Conv2D)         (None, 16, 16, 256)       590080

 dropout_4 (Dropout)         (None, 16, 16, 256)       0

 conv2d_107 (Conv2D)         (None, 16, 16, 128)       32896

 max_pooling2d_19 (MaxPooling2D  (None, 8, 8, 128)     0
 )

 up_sampling2d_16 (UpSampling2D  (None, 16, 16, 128)   0
 )

 conv2d_108 (Conv2D)         (None, 16, 16, 512)       262656

 resizing_12 (Resizing)      (None, 16, 16, 512)       0

 tf.image.rot90_12 (TFOpLambda)  (None, 16, 16, 128)   0

 tf.image.flip_left_right_6 (TF  (None, 16, 16, 128)   0
 OpLambda)
```

```
max_pooling2d_18 (MaxPooling2D  (None, 16, 16, 64)  0
)

conv2d_105 (Conv2D)             (None, 16, 16, 256)  147712

conv2d_106 (Conv2D)             (None, 16, 16, 256)  590080

dropout_4 (Dropout)             (None, 16, 16, 256)  0

conv2d_107 (Conv2D)             (None, 16, 16, 128)  32896

max_pooling2d_19 (MaxPooling2D  (None, 8, 8, 128)   0
)

up_sampling2d_16 (UpSampling2D  (None, 16, 16, 128)  0
)

conv2d_108 (Conv2D)             (None, 16, 16, 512)  262656

resizing_12 (Resizing)          (None, 16, 16, 512)  0

tf.image.rot90_12 (TFOpLambda)  (None, 16, 16, 128)  0

tf.image.flip_left_right_6 (TF  (None, 16, 16, 128)  0
OpLambda)

concatenate_16 (Concatenate)    (None, 16, 16, 896)  0



conv2d_109 (Conv2D)             (None, 16, 16, 256)  2064640

conv2d_110 (Conv2D)             (None, 16, 16, 256)  590080

up_sampling2d_17 (UpSampling2D  (None, 32, 32, 256)  0
)

conv2d_111 (Conv2D)             (None, 32, 32, 256)  262400

resizing_13 (Resizing)          (None, 32, 32, 256)  0

tf.image.rot90_13 (TFOpLambda)  (None, 32, 32, 64)  0

concatenate_17 (Concatenate)    (None, 32, 32, 384)  0


conv2d_112 (Conv2D)             (None, 32, 32, 128)  442496

conv2d_113 (Conv2D)             (None, 32, 32, 128)  147584

up_sampling2d_18 (UpSampling2D  (None, 64, 64, 128)  0
)

conv2d_114 (Conv2D)             (None, 64, 64, 128)  65664
```

```
conv2d_112 (Conv2D)            (None, 32, 32, 128)  442496

conv2d_113 (Conv2D)            (None, 32, 32, 128)  147584

up_sampling2d_18 (UpSampling2D  (None, 64, 64, 128)  0
)

conv2d_114 (Conv2D)            (None, 64, 64, 128)  65664

resizing_14 (Resizing)         (None, 64, 64, 128)  0

tf.image.rot90_14 (TFOpLambda)  (None, 64, 64, 64)  0

concatenate_18 (Concatenate)   (None, 64, 64, 256)  0


conv2d_115 (Conv2D)            (None, 64, 64, 64)   147520

conv2d_116 (Conv2D)            (None, 64, 64, 64)   36928

up_sampling2d_19 (UpSampling2D  (None, 128, 128, 64  0
)                               )

conv2d_117 (Conv2D)            (None, 128, 128, 64  16448
                               )

resizing_15 (Resizing)         (None, 128, 128, 64  0
                               )

tf.image.rot90_15 (TFOpLambda)  (None, 128, 128, 32  0
                               )

tf.image.flip_left_right_7 (TF  (None, 128, 128, 32  0
OpLambda)                       )

concatenate_19 (Concatenate)   (None, 128, 128, 16  0
                               0)



conv2d_118 (Conv2D)            (None, 128, 128, 32  46112
                               )

conv2d_119 (Conv2D)            (None, 128, 128, 32  9248
                               )

conv2d_120 (Conv2D)            (None, 128, 128, 4)  132

=================================================================
Total params: 5,166,820
Trainable params: 5,166,820
Non-trainable params: 0
_________________________________________________________________
```

FIGURE 4.9 – Architecture Détaillée du CNN G-Unet et symétries.

Nous avons utilisé fit, Modelcheckpoint et EarlyStopping avec le modèle pour l'entraîner. Cette section l'expliquera.

```

# compling model and callbacks functions
adam = tf.keras.optimizers.Adam(lr =1e-4, epsilon = 0.1)
model.compile(optimizer = adam,
                  loss = 'binary_crossentropy',
                  metrics = ['accuracy']
                 )
#callbacks
earlystopping = EarlyStopping(monitor='val_loss',
                              mode='min',
                              verbose=1,
                              patience=30
                             )
# save the best model with lower validation loss
checkpointer = ModelCheckpoint(filepath="seg_model.h5",
                               verbose=1,
                               save_best_only=True
                              )
reduce_lr = ReduceLROnPlateau(monitor='val_loss',
                              mode='min',
                              verbose=1,
                              patience=10,
                              min_delta=0.0001,
                              factor=0.2
                             )
```

Listing 4.8 – model de Fitting:

— **Modelcheckpoint :** Est utilisé en conjonction avec la formation à l'aide de model.fit() pour enregistrer un modèle ou des poids (dans un fichier de point de contrôle) à un certain intervalle, de sorte que le modèle ou les poids peuvent être chargés plus tard pour continuer la formation à partir de l'état enregistré.

— **EarlyStopping :** Sa fonction de rappel Keras est utilisée pour mettre en pause l'entraînement du modèle au milieu. Lorsque vos modèles sont suréquipés. Il est utilisé pour empêcher le modèle de devenir suréquipé. Lors de la sauvegarde des points de contrôle du modèle, nous avons défini ce qu'il faut surveiller.

— **Modèle Fit :** Nous alimentons d'abord les données d'entraînement (X TRAIN) et les éti-

quettes d'entraînement (Y TRAIN) et la validation par (X VAL, Y VAL). Nous utilisons ensuite Keras pour permettre à notre modèle de s'entraîner pendant 100 époques sur une taille de lot de 32.

4.6 Entrainement et résultats du modèle

Dans cette section, nous discutons des phases d'entraînement et de test du modèle. Il est divisé en deux sous-sections, la première sous-section sera dédiée à la formation de notre modèle à l'aide d'un ensemble d'images d'entraînement, notez que notre modèle est formé pour plusieurs époques (35).

1. Entrainement :

Le modèles sont entraînés sur une base de données brats2019 du (4,57 GB),Nous utilisons des données (HGG) nombre des patients est 259 avec 1295 images. le tableau suivant résume :

Data	Nombre d'image
Training set	3358
Test set	618
Validation set	1002
Total	4978

Table 6.7-Tableau des paramètres de la structure proposée.

2. Spécification de l'initialisation des paramètres pour cette structure :

Dans cette structure, nous proposons le paramètres suivant dans le tableau 6.6.

	Informations
input shape	(128,128,2)
Nombre d'epochs	35
Step per epoch	50
dropOutRate	15%
Patience	25
dropOutRate	15%
Activation Model	sigmoid
dropOutRate	15%
ModelCheckpoint Monitor	Val_acc
EarlyStopping Monitor	val_loss

Table 6.6-Tableau des paramètres pour la structure proposée avec modifications.

3. Expérimentation :

Les résultats de la segmentation de l'image médicale à l'aide d'un système de reconnaissance CNN seront présentés ci-après.

graphique montrant la précision du modèle

Ce graphique montre l'exactitude du modèle «"Accuracy"» sur l'ensemble des données d'entraînement les données , ainsi que l'exactitude «"ValAccuracy"» de la validation «les "ValData"» sur l'ensemble des données d'entraînement.

FIGURE 4.10 – Présentation graphique du modèle de précision.

graphique montrant l'erreur du modèle

Ce graphique montre l'exactitude du modèle «"Loss"» sur l'ensemble des données d'entraînement les données , ainsi que l'exactitude «"ValLoss"» de la validation «les "ValData"» sur l'ensemble des données d'entraînement.

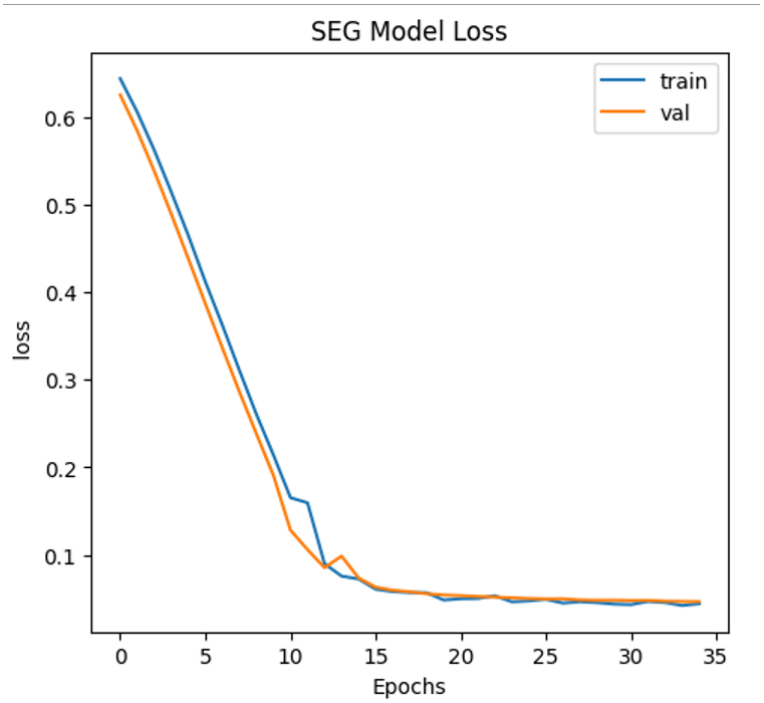


FIGURE 4.11 – Présentation graphique du modèle de l'erreur.

Après analyse des résultats, nous avons relevé les observations suivantes :

— La précision d'apprentissage et des tests augmente avec le nombre d'époque, ce qui illustre comment le modèle apprend plus de connaissances au du temps.

— De même, les erreurs d'apprentissage et de validation diminuent avec le nombre d'époques.

— Numero d'époque est 35.

— Précision à 98,34%.

— validation de la précision à 0,9824

— L'erreur à 0,08.

— La validation de l'erreur à 0,069

	Accuracy	Validation accuracy	Loss	Validation Loss
CNN Model	98.30%	96.12%	35.95%	32.42%

Table 6.7 :Tableau des résultats structure proposée pour l'ensemble de données

4.7 L'évaluation du modèle

Lors de la partie d'évaluation, le modèle a été évalué à l'aide des figures de patients IRM correspondant à la même classification dans notre base de données. Les résultats sont les suivants :

```
Evaluate on test set
32/32 [==============================] - 10s 278ms/step - loss: 0.0850 - accuracy: 0.9832 - iou: 0.8961 - f1_score: 0.9445 - dice_coefficient: 0.9445
test loss, test acc: [0.08500105142593384, 0.9831646680831909, 0.8961018919944763, 0.9444926977157593, 0.9444929361343384]
|**********__________Test Model__________*******|
32/32 [==============================] - 9s 270ms/step - loss: 0.0850 - accuracy: 0.9832 - iou: 0.8961 - f1_score: 0.9445 - dice_coefficient: 0.9445
Total images : 1295
Test Loss: 0.08500105887651443
Test Accuracy: 98.31648468971252
Test IOU: 89.61018919944763
Test F1: 94.44928765296936
Test dice_coefficient: 94.44931149482727
```

FIGURE 4.12 – Les résultats du modèle.

1- La figure 4.13 montre des images aprés du prédict modèle :
Ces images génèrent des prédictions pour un ensemble d'images de test et affichent les masques de segmentation prédits et réels pour les 10 premières images. Les prédictions sont obtenues sur le générateur de test.

Il s'agit d'évaluer visuellement la correspondance entre les masques de segmentation prédits par le modèle et les masques . La comparaison des deux masques permet de vérifier la justesse du modèle dans la segmentation de l'objet d'intérêt dans l'image test.

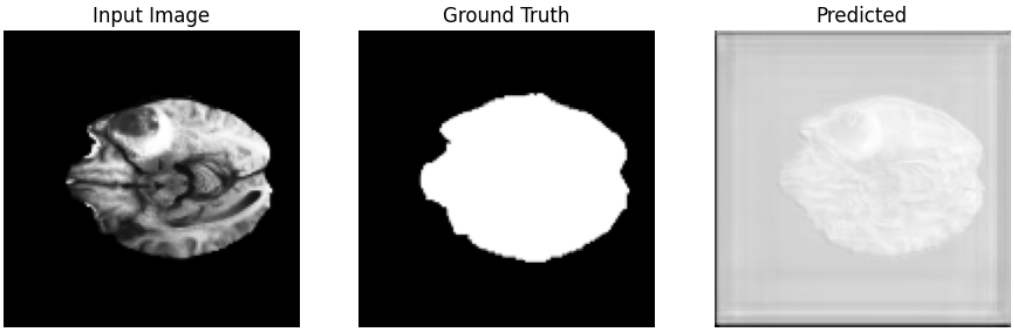


FIGURE 4.13 – Des images aprés du prédict modèle.

2- La figure 4.14 Montre des images aprés du test modèle :

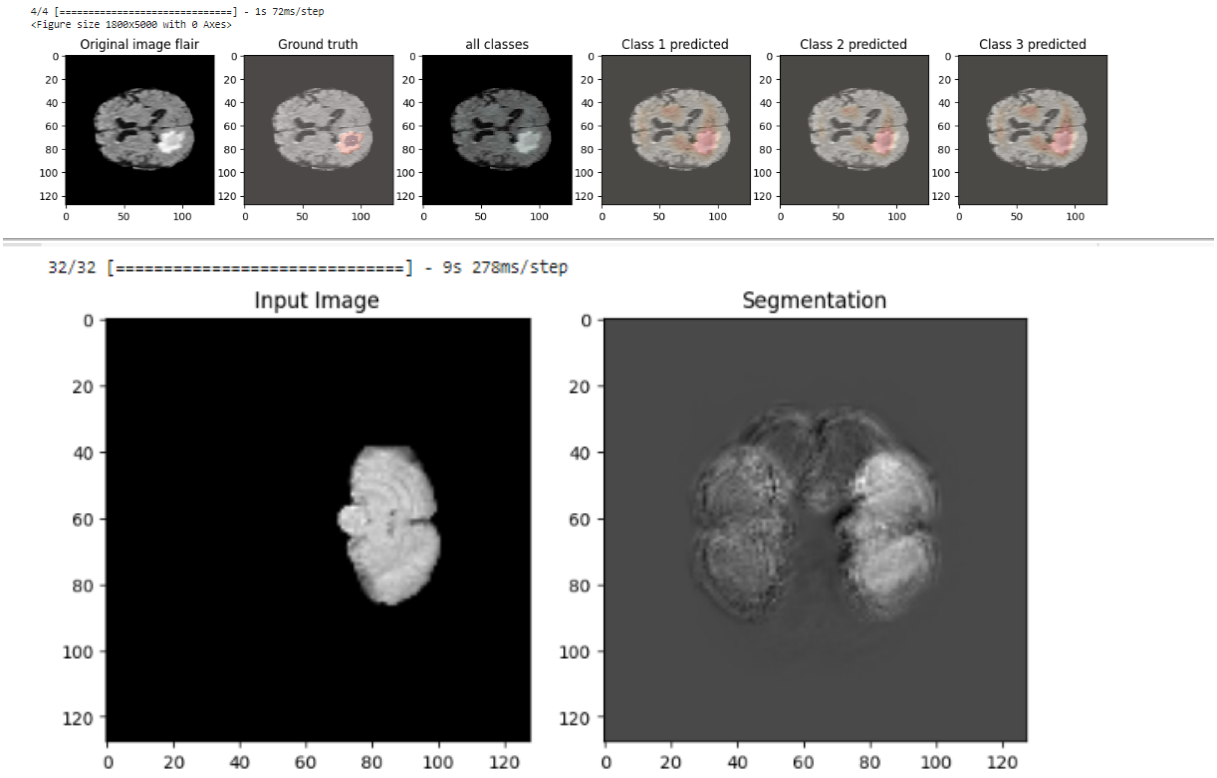


FIGURE 4.14 – Des images aprés du test modèle.

Le figure 4.14 fourni d'évaluer visuellement les résultats du modèle sur un échantillon aléatoire de l'ensemble de test. La première figure est illustrée :

1. **Flair d'image d'origine :** la première sous-parcelle montre l'image d'entrée d'origine (modalité flair) pour le cas spécifié. Il est affiché en niveaux de gris à l'aide de la palette "gris". Cette image sert de référence de comparaison.

2. **Ground Truth :** le deuxième sous-parcelle montre la segmentation de la vérité au sol obtenue à partir du fichier d'étiquettes de segmentation. Il représente la répartition réelle des différentes catégories. Utilisez la palette "rouge" pour afficher la segmentation de la vérité terrain dans les tons rouges.

3. **Toutes les classes :** la troisième sous-parcelle montre la segmentation attendue pour toutes les classes. Il visualise toutes les catégories prédites par le modèle. Utilisez la palette Rouge pour afficher les segments dans des tons rouges.

4. **Classe 1 prédite :** Le quatrième sous-état montre la segmentation prédite pour la classe 1 (œdème). Il visualise la répartition des plans pour cette classe particulière. Utilisez la palette

"OrRd" pour afficher les segments dans les tons rouges.

5. Catégorie 2 prévue : le cinquième sous-état indique la répartition prévue pour la catégorie 2 (noyau). Il visualise la répartition des plans pour cette classe particulière. Utilisez la palette "OrRd" pour afficher les segments dans les tons rouges.

Les résultats obtenus par notre modèle dans le domaine de l'imagerie sont très prometteurs. En utilisant des méthodes basées sur l'apprentissage en profondeur et en exploitant des méthodes de symétrie, notre modèle démontre une grande précision dans la classification des images médicales. Les résultats de la validation ont atteint une précision impressionnante de 98 %, démontrant l'efficacité de notre modèle dans l'identification et la classification des patients atteints de la maladie

4.8 Comparaison des résultats

Pour évaluer les performances de notre modèle, nous comparons nos résultats obtenus à certaines approches basées sur l'apprentissage profond existantes dans la littérature. En général, notre approche est performante et donne de bons résultats par rapport aux autres travaux proposés dans le même contexte. Nous obtenons une bonne précision d'environ 97,78 % contre 78 %, 82,09 % et 96 % et aussi f1 nous obtenons 94% par rapport 86%,96%et 87% dans [45],[39] et [50] respectivement.

Article	Architecture	symetries	Datasets	Précision	F1-score	Dice
[45]	2PathCNN	Oui	ISLES2015	78%	86%	83,24%
[39]	Ger-Net	Oui	Brats	82,09%	96%	73,93%
[50]	Unet	Non	Brats 2015	98,30%	87%	97%
Ce travail	**G-Unet**	**Oui**	**Brats 2019**	**97 %**	93%	**94,49%**

Table 6.9- Tableau comparatif des travaux connexes

En comparant nos résultats avec des travaux antérieurs, nous constatons que notre modèle surpasse la plupart des méthodes existantes en termes de précision. Cela montre que notre mé-

thode est capable de mieux capturer les caractéristiques et les modèles des données d'entrée, conduisant à de meilleures performances prédictives.
Outre la précision, nous avons également évalué d'autres mesures de performance telles que le rappel, la spécificité et le score F1. Ces mesures supplémentaires nous ont permis de quantifier la capacité de notre modèle à identifier correctement les vrais positifs, les vrais négatifs et les faux positifs. Dans l'ensemble, notre modèle fonctionne bien dans ces domaines, démontrant sa robustesse et ses capacités de généralisation efficaces.
Il convient de noter que bien que nous ayons obtenu de bons résultats, chaque approche a ses propres avantages et inconvénients en fonction du contexte spécifique de l'application et des données utilisées. Par conséquent, il est important de tenir compte de ces aspects lors de la comparaison des performances.

4.9 Conclusion

Dans ce chapitre, Nous avons détaillé comment notre modèle d'architecture CNN. Ensuite, nous avons discuté des résultats qui ont été comparés aux mesures de rendement avec les tâches suggérées après la présentation des constatations.

Conclusion générale

Ces dernières années, l'intelligence artificielle (IA) a suscité l'intérêt de diverses industries, dont le secteur de la santé. L'émergence de matériel informatique et d'applications logicielles dans les domaines médicaux a facilité le développement et la mise en œuvre de l'IA dans ce domaine. La prise de décision et le diagnostic sont deux applications de l'apprentissage profond dans le domaine médical.L'un des problèmes les plus importants auxquels font face les clients dans le domaine médical est la tumeur. L'utilisation de l'apprentissage profond pour diagnostiquer la maladie tumorale dans ses premiers stades et simplifier le traitement pour les cliniciens a produit des résultats positifs. où l'apprentissage profond est utilisé pour accélérer et augmenter la précision des diagnostics.
Pour conclure, rappelons que le but de ce travail est de développer et de mettre en œuvre un modèle de segmentation des tumeurs cérébrales basé sur le Deep Learning et les symétries.
Cette mémoire commence par une section qui présente quelques idées de base sur l'anatomie du cerveau. vient ensuite les tumeurs cérébrales. Ensuite, après avoir discuté des images IRM, de leur fonctionnalité et de leur qualité (avantages et inconvénients), nous avons conclu cette section par une discussion de la segmentation de l'image médicale tout en fournissant des exemples de méthodes de classification convolutionnelles, d'approches d'apprentissage profond et de symétries. La conception de notre modèle représentent une troisième partie de ce mémoire.

Par conséquent, au chapitre 3, nous avons discuté de notre contribution à la nouvelle architecture G-UNet de CNN, qui est un cadre complet pour la catégorisation équilibrée qui propose et utilise des techniques de brisage de symétrie. Les étapes nécessaires à la construction et à la

préparation de la base de données ont été décrites en détail. Ensuite, la méthode suggérée pour extraire les algorithmes et les coupes informationnelles. Puis, dans le chapitre 4, nous avons présenté le développement détaillé de notre architecture de réseau neuronal profond ainsi que les outils et environnements utilisés dans le développement de projets. Nous avons également présenté les résultats, montrant l'amélioration du rendement après une comparaison avec les approches de pointe existantes. En fait, en utilisant notre modèle, nous avons réussi à obtenir une cote de précision de 93% pour la validation de nos données de formation de trains.

Travail futur :

Passons aux tâches et aux perspectives. Nous sommes conscients qu'aucun travail n'est parfait. Nous attendons avec impatience de peaufiner nos modèles et de les rendre applicables au monde réel dans un proche avenir en plus de travailler comme un groupe de programmeurs qui ont travaillé sur un article et un modèle intitulé : Beyond CNNs : Exploiting Further Inherent Symmetries in Medical Image Segmentation ,en utilisant d'une autre base de données pour évaluer et améliorer le modèle et la participation à la compétition lancé par Brats en 2021.

Bibliographie

[1] Région. Consulté le 2 juin 2023. URL http://zonkooo.free.fr/image/segmentation.html.

[2] Introduction à l'apprentissage automatique : Monographie de CPA Nouveau-Brunswick. Consulté le 2 juin 2023, . URL https://www.cpapei.ca/fr/new-brunswick/evenements-et-medias/cpa-source/2019/intro-to-machine-learning-a-cpa-new-brunswick-monograph.

[3] Training options for adam optimizer - MATLAB - MathWorks switzerland. Consulté le 2 juin 2023, . URL https://ch.mathworks.com/help/deeplearning/ref/nnet.cnn.trainingoptionsadam.html.

[4] Martín Abadi, Michael Isard, and Derek G Murray. A computational model for tensorflow : an introduction. In *Proceedings of the 1st acm sigplan international workshop on machine learning and programming languages*, pages 1–7, 2017.

[5] Nawaf Mohammad H Alamri, Michael Packianather, and Samuel Bigot. Deep learning : Parameter optimization using proposed novel hybrid bees bayesian convolutional neural network. *Applied Artificial Intelligence*, 36(1) :2031815, 2022.

[6] Naouel Azouza. *Segmentation par classification floue : application à l'imagerie par résonance magnétique*. PhD thesis, université Akli Mouhend-Oulhadj de Bouira, 2018.

[7] Salima BELHADEF and Khadidja BENABDALLAH. *Segmentation des images IRM cérébrales par apprentissage profond*. PhD thesis.

[8] Philippe Bolon, Jean-Marc Chassery, Jean-Pierre Cocquerez, Didier Demigny, Christine

Graffigne, Annick Montanvert, Sylvie Philipp, Rachid Zéboudj, Josiane Zerubia, and Henri Maître. Analyse d'images : filtrage et segmentation, 1995.

[9] J Botros, F Mourad-Chehade, and D Laplanche. L'évaluation du risque d'insuffisance cardiaque à l'aide des signaux d'intervalles rr à court terme. *Revue d'Épidémiologie et de Santé Publique*, 71 :101462, 2023.

[10] Patrice Brault. Estimation de mouvement et segmentation d'image. *Partie I : Estimation de mouvement par ondelettes spatio-temporelles adaptées au mouvement, PhD Université Paris-Sud XI, Ecole Doctorale Sciences et Technologies de l'Information des Télécommunications et des Systèmes*, 2005.

[11] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6) :679–698, 1986.

[12] Anne-Sophie Capelle-Laizé. *Segmentation des images IRM multi-échos tridimensionnelles pour la détection des tumeurs cérébrales par la théorie de l'évidence*. PhD thesis, Université de Poitiers, 2003.

[13] Tiago Carneiro, Raul Victor Medeiros Da Nóbrega, Thiago Nepomuceno, Gui-Bin Bian, Victor Hugo C De Albuquerque, and Pedro Pedrosa Reboucas Filho. Performance analysis of google colaboratory as a tool for accelerating deep learning applications. *IEEE Access*, 6 : 61677–61685, 2018.

[14] Léa Castanon. Jean paul payet, anaïk purenne (dir), tous égaux! les institutions à l'ère de la symétrie. *Lectures*, 2016.

[15] Michael M Chang, M Ibrahim Sezan, and A Murat Tekalp. Adaptive bayesian segmentation of color images. *Journal of Electronic Imaging*, 3(4) :404–414, 1994.

[16] François Chollet and Joseph J Allaire. *Deep Learning mit R und Keras : Das Praxis-Handbuch von den Entwicklern von Keras und RStudio*. MITP-Verlags GmbH & Co. KG, 2018.

[17] Thibaut Citron, Maxime Denoyer, José Martins, and Cyprien Ruelle. Programmation hétérogène cpu-gpu.

[18] Andrea Daou, Jean-baptiste Pothin, Paul Honeine, and Abdelaziz Bensrhair. Amélioration des performances des réseaux de neurones convolutifs en localisation indoor par augmentation des données. In *ORASIS 2021*, 2021.

[19] Siddharth Das. CNN Architectures : LeNet, AlexNet, VGG, GoogLeNet, ResNet and more. . .. 6 2021. URL https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5.

[20] Sander Dieleman, Jeffrey De Fauw, and Koray Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. In *International conference on machine learning*, pages 1889–1898. PMLR, 2016.

[21] Silvia Fanton and William H Thompson. Netplotbrain : A python package for visualising networks and brains. *Network Neuroscience*, pages 1–21, 2023.

[22] Pr Guy Frija and Bernard Mazoyer. L'imagerie médicale. *Fondation recherche médicale*, 2002.

[23] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern recognition*, 77 :354–377, 2018.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[25] John D Hunter. Matplotlib : A 2d graphics environment. *Computing in science & engineering*, 9(03) :90–95, 2007.

[26] IndexSante. Importance d'imagerie medicale. Consulté le 2 mai 2023. URL https://www.indexsante.ca/chroniques/284/imagerie-medicale.php.

[27] Bruno Kastler, P Anstett, Bruno Kastler, and Daniel Vetter. *Comprendre l'IRM*. Elsevier, 2011.

[28] Thayyaba Khatoon Mohammed, M Shanmuga Sundari, and UL Sivani. Brain tumor image classification with cnn perception model. In *Soft Computing and Signal Processing : Proceedings of 3rd ICSCSP 2020, Volume 2*, pages 351–361. Springer, 2022.

[29] Robert Laganiere. *OpenCV computer vision application programming cookbook second edition*. Packt Publishing Ltd, 2014.

[30] Chaabane LAMICHE. Fusion et fouille de donnees guidees par les connaissances : Application a l'analyse d'image. 2013.

[31] Bin Liu, Fule Liu, Longyun Fang, Xiaolong Wang, and Kuo-Chen Chou. repdna : a python package to generate various modes of feature vectors for dna sequences by incorporating user-defined physicochemical properties and sequence-order effects. *Bioinformatics*, 31 (8) :1307–1309, 2015.

[32] A Lopès, R Fjørtoft, D Ducrot, P Marthon, and C Lemaréchal. Edge detection and segmentation of sar images in homogeneous regions. In *Information Processing for Remote Sensing*, pages 139–166. World Scientific, 1999.

[33] Samaya Madhavan and M Tim Jones. Deep learning architectures. *IBM Developer*, 2017.

[34] André Marion. *Acquisition & visualisation des images*. Eyrolles, 1997.

[35] Memoire Online. Bruit. Consulté le 2 mai 2023. URL https://www.memoireonline.com/12/09/3040/m_La-liaison-automatique-des-plusieurs-images-percues-sur-un-scanner4.html.

[36] Cyril Meurie. *Segmentation d'images couleur par classification pixellaire et hiérarchie de partitions*. PhD thesis, Caen, 2005.

[37] MSDManuals. Tumeurs cérébrales. Consulté le 2 juin 2023. URL https://www.msdmanuals.com/fr/accueil/troubles-du-cerveau,-de-la-moelle/.

[38] Travis E Oliphant et al. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[39] Shuchao Pang, Anan Du, Mehmet A Orgun, Yan Wang, Quan Z Sheng, Shoujin Wang, Xiaoshui Huang, and Zhenmei Yu. Beyond cnns : Exploiting further inherent symmetries in medical image segmentation. *IEEE transactions on cybernetics*, 2022.

[40] Theodosios Pavlidis and Y-T Liow. Integrating region growing and edge detection. *IEEE transactions on pattern analysis and machine intelligence*, 12(3) :225–233, 1990.

[41] Ekachai Phaisangittisagul. An analysis of the regularization between l2 and dropout in single hidden layer neural network. In *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, pages 174–179. IEEE, 2016.

[42] PraedictIA. Comment fonctionnent les réseaux de neurones - PraedictIA, 3 2022. URL https://praedictia.com/page/reseaux-de-neurones/comment-fonctionnent-les-reseaux-de-neurones.html.

[43] Judith MS Prewitt et al. Object enhancement and extraction. *Picture processing and Psychopictorics*, 10(1) :15–19, 1970.

[44] Sylvain Prima. *Étude de la symétrie bilatérale en imagerie cérébrale volumique*. PhD thesis, Université Paris Sud-Paris XI, 2001.

[45] Kevin Raina, Uladzimir Yahorau, and Tanya Schmah. Exploiting bilateral symmetry in brain lesion segmentation with reflective registration. In *BIOIMAGING*, pages 116–122, 2020.

[46] LG Roberts. Machine perception of three-dimensional solids, chapter optical and electrooptical information processing, 1965.

[47] HOCINE Saadi and Yasser Saidi. *DEEP-LEARNING APPLIQUE A LA SEGMENTATION DES IMAGES MEDICALES*. PhD thesis, université akli mohand oulhadj-bouira, 2021.

[48] Pr Douraïed BEN Salem and Dr Julien Ognard. LES PRINCIPAUX ARTEFACTS EN IRM.

[49] Irvin Sobel. Neighborhood coding of binary images for fast contour following and general binary array processing. *Computer graphics and image processing*, 8(1) :127–135, 1978.

[50] Meera Srikrishna, Rolf A Heckemann, Joana B Pereira, Giovanni Volpe, Anna Zettergren, Silke Kern, Eric Westman, Ingmar Skoog, and Michael Schöll. Comparison of two-dimensional-and three-dimensional-based u-net architectures for brain tissue classification in one-dimensional brain ct. *Frontiers in Computational Neuroscience*, 15 :123, 2022.

[51] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout : a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1) :1929–1958, 2014.

[52] Friedrich Tiedemann. *Anatomie du cerveau...* J.-B. Baillière, 1823.

[53] Alain Trémeau and Philippe Colantoni. Regions adjacency graph applied to color image segmentation. *IEEE Transactions on image processing*, 9(4) :735–744, 2000.

[54] Qin Wang, Fengyi Shen, Linyao Shen, Jia Huang, and Weiguang Sheng. Lung nodule detection in ct images using a raw patch-based convolutional neural network. *Journal of digital imaging*, 32 :971–979, 2019.

[55] Bin Yang, Honglei Guo, and Enguo Cao. Design of cyber-physical-social systems with forensic-awareness based on deep learning. In *Advances in Computers*, volume 120, pages 39–79. Elsevier, 2021.

[56] Xiaohong R Yang, Jenny Chang-Claude, Ellen L Goode, Fergus J Couch, Heli Nevanlinna, Roger L Milne, Mia Gaudet, Marjanka K Schmidt, Annegien Broeks, Angela Cox, et al. Associations of breast cancer risk factors with tumor subtypes : a pooled analysis from the breast cancer association consortium studies. *Journal of the National Cancer Institute*, 103 (3) :250–263, 2011.