RAPPORT MINI-PROJET

Détection de tumeur cérébrale avec un modèle CNN

Brahim ARIANI

Sommaire

Liste des figures

Chapitre 1 : La Tumeur Cérébrale et IRM

1. Introduction

Les tumeurs cérébrales sont un problème de santé important pour tous les groupes d'âge, et les données suggèrent que leur incidence est en augmentation. Chez les nourrissons et les jeunes enfants, les tumeurs du cerveau sont la deuxième forme de cancer la plus courante. Chez les adolescents et les jeunes adultes, les tumeurs cérébrales vont du cinquième au huitième cancer le plus fréquent. Dans la population âgée, l'incidence des tumeurs cérébrales, primitives et méta-statiques, est en augmentation. Les tumeurs cérébrales primaires ont une prévalence de 14,7 pour 100 000 aux États-Unis, et il y a environ 80 000 à 100 000 nouvelles tumeurs cérébrales Méta-statiques chaque année. Ainsi, pratiquement tous les médecins, quelle que soit leur spécialité, rencontreront des patients atteints de tumeurs cérébrales et la plupart des familles auront un parent ou un ami atteint d'une tumeur cérébrale.

2. Qu'est-ce qu'une tumeur ?

Une tumeur est une masse plus ou moins volumineuse due à une multiplication anormale de cellules. La cellule est l'unité de base dont sont constitués tous les tissus des organismes vivants. Il en existe dans le corps plus de deux cents types différents : les cellules musculaires, nerveuses, osseuses, etc. Chaque cellule a un rôle précis et une durée de vie limitée. En permanence, les cellules vieillissent, meurent et sont remplacées par d'autres. Pour des raisons encore inconnues, il arrive que certaines cellules se modifient et continuent à se multiplier au lieu de disparaître naturellement. Elles se multiplient alors jusqu'à former une tumeur.

D'une manière générale, il existe trois types de tumeurs :

- **Les tumeurs non cancéreuses :** appelées tumeurs bénignes. Elles se développent lentement et restent localisées. Une fois traitées, elles ne récidivent généralement pas.
- **Les tumeurs cancéreuses :** appelées tumeurs malignes. Elles se développent plus rapidement et ont tendance à envahir d'autres zones que celles où elles sont apparues au départ. Elles peuvent développer alors de nouvelles tumeurs appelées métastases.
- **Les tumeurs intermédiaires :** dites « atypiques » ou « évolutives ». Bénignes au début, elles peuvent se transformer en cancer dans un laps de temps variable.

3. Qu'est-ce qu'une tumeur du cerveau ?

On appelle tumeur du cerveau toutes les tumeurs qui se développent à l'intérieur du crâne. Elles peuvent se développer dans n'importe quelle zone du cerveau : les hémisphères, le cervelet, le tronc cérébral, l'hypophyse, etc.

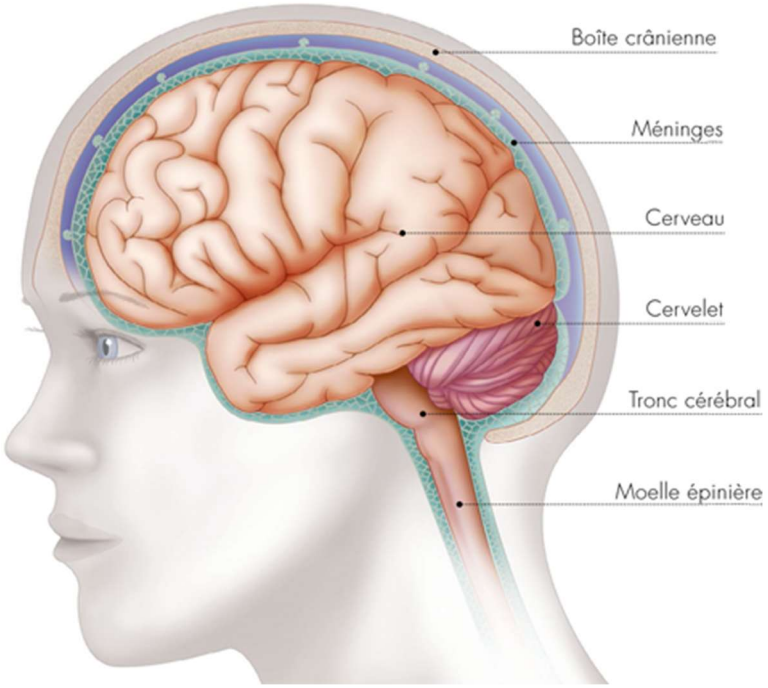


Figure 1:Présentation du cerveau

a. Leur localisation dans le cerveau

Le cerveau est organisé en plusieurs zones, qui gèrent chacune des activités spécifiques : le langage, l'équilibre du corps, les battements du cœur, la circulation du sang, la mémoire, etc. Une tumeur peut entraîner des troubles très différents selon la zone dans laquelle elle se développe. La localisation de la tumeur est également un élément essentiel pour le choix des traitements. Une tumeur située à la surface du cerveau par exemple, est généralement plus facile à extraire qu'une tumeur située au centre du cerveau.

b. Leur type

Le cerveau est composé de différents types de cellules, qui peuvent chacune être à l'origine de tumeurs différentes. Les tumeurs du cerveau portent généralement le nom des cellules à partir desquelles elles se développent : les gliomes se développent à partir des cellules gliales, qui nourrissent et soutiennent les neurones ; les méningiomes se développent à partir des cellules composant les méninges* (enveloppes du cerveau), etc. Selon leur type (les médecins parlent de type histologique), les tumeurs ne se comportent pas de la même manière. Certaines se développent plus vite que d'autres ou ont un risque accru de récidive par exemple.

c. Leur degré

D'agressivité Plus la tumeur se développe rapidement, plus elle est jugée agressive. Les tumeurs du cerveau sont classées en différents grades selon leur agressivité. On parle de bas grade pour les tumeurs

les moins agressives et de haut grade pour les tumeurs qui le sont davantage. Le grade peut aussi être exprimé par un chiffre romain allant de I à IV. Le grade I correspond aux tumeurs non cancéreuses, ce sont les moins agressives. Le grade IV correspond aux tumeurs les plus agressives. Connaître le grade d'une tumeur est déterminant dans le choix des traitements et dans le pronostic*. Plus le grade est bas, plus le pronostic est favorable. Lorsqu'une tumeur du cerveau est découverte, il est indispensable de déterminer ses caractéristiques : sa localisation, son type et son degré d'agressivité. Pour cela, plusieurs examens doivent être réalisés. Ce n'est qu'après ces différents examens que l'on sait de quel type de tumeur il s'agit et que l'on peut définir les traitements appropriés.

4. Quelles sont les causes des tumeurs du cerveau ?

Les causes des tumeurs du cerveau sont mal connues. On ne sait pas pourquoi, à un moment donné, les cellules se multiplient de manière incontrôlée jusqu'à former une tumeur. De nombreuses études scientifiques ont été menées ou sont en cours, pour tenter de déterminer les facteurs qui favorisent l'apparition d'une tumeur au niveau du cerveau. On parle de facteurs de risques. Trois types de facteurs de risques sont étudiés : les risques liés à l'environnement, à l'hérédité et aux virus. Un facteur de risque n'explique pas à lui seul pourquoi une tumeur est apparue. En effet, pour deux personnes confrontées au même risque, l'une peut développer une tumeur et l'autre non. À l'inverse, une personne peut être atteinte d'une tumeur alors qu'elle n'est concernée par aucun facteur de risque.

5. Quels sont les symptômes d'une tumeur du cerveau ?

Les symptômes provoqués par une tumeur du cerveau sont très variables et n'apparaissent pas systématiquement. Ils dépendent du volume de la tumeur, de la vitesse à laquelle elle se développe et surtout de son emplacement. L'apparition de symptômes peut être soudaine ou très progressive, selon la rapidité à laquelle la tumeur se développe. Certaines tumeurs ne provoquent aucun symptôme, ce qui signifie généralement qu'elles se développent lentement. Trois types de symptômes sont possibles :

- **Des maux de tête (céphalées),** liés à une augmentation de la pression à l'intérieur du crâne.
- **Des crises d'épilepsie**, liées à un dérèglement de l'activité des neurones.
- **Des troubles fonctionnels**, directement liés à la localisation de la tumeur et aux fonctions gérées dans cette zone du cerveau. Ce peut être des troubles de la vision, des modifications de la personnalité, des difficultés à coordonner ses mouvements ou à trouver ses mots. . .

6. Le diagnostic d'une tumeur du cerveau

Le médecin traitant joue un rôle important dans le diagnostic, car c'est souvent lui qui est consulté en premier. En cas de suspicion de tumeur, l'intervention de plusieurs médecins spécialistes est indispensable : neurologie, neurochirurgien, neuro-oncologue, radiologue, anatomopathologiste (appelé aussi pathologiste) . . .

Le diagnostic d'une tumeur du cerveau se fait en plusieurs étapes :

- Un examen clinique et neurologique complet ;
- Des examens d'imagerie ;
- Un examen anatomopathologique, qui consiste à analyser un échantillon de tumeur ;

a. Les examens d'imagerie

Les examens d'imagerie consistent à réaliser des images précises du cerveau. Ils permettent de détecter une tumeur, de la localiser de façon précise, de mesurer sa taille et d'évaluer les conséquences qu'elle a ou peut avoir sur le cerveau. Pour le diagnostic d'une tumeur du cerveau, une IRM est indispensable. Pour des raisons pratiques, l'IRM est souvent précédée d'un scanner, pour lequel il est plus facile d'obtenir un rendez-vous rapidement.

b. Un scanner

Un scanner est un examen qui permet d'obtenir des images du cerveau en coupe, grâce à un appareil qui projette des rayons X au niveau du crâne. L'appareil est constitué d'un lit d'examen et d'un gros anneau. Vous êtes allongé sur la table d'examen et votre tête est placée à l'intérieur de l'anneau. Un faisceau de rayons X dirigé sur le crâne permet d'obtenir plusieurs centaines de radiographies Les radiographies sont transmises à un ordinateur, qui reconstitue des images du cerveau dans les trois dimensions. Généralement, un produit de contraste à base d'iode est injecté avant ou pendant l'examen. Ce produit permet de mettre en évidence certains aspects du cerveau, notamment les vaisseaux sanguins et facilite l'interprétation des images. L'examen dure environ 15 minutes. Il n'est pas douloureux...

Figure 2: Scanner cérébrale

c. Une IRM

Une IRM (imagerie par résonance magnétique) est réalisée grâce à un grand appareil en forme de cylindre. Cet appareil est composé d'un aimant très puissant (d'où le terme de magnétique). Il produit des ondes radio qui sont projetées sur le cerveau et permettent d'obtenir des images « en coupe ». Les images sont ensuite assemblées par un ordinateur pour obtenir une reproduction très précise du cerveau. Pendant l'examen, un produit de contraste est injecté dans une veine du bras. Il permet de mettre en évidence certains aspects du cerveau, comme les vaisseaux sanguins et facilite l'interprétation des images. L'examen dure généralement entre 15 et 30 minutes. Il n'est pas douloureux, mais bruyant.

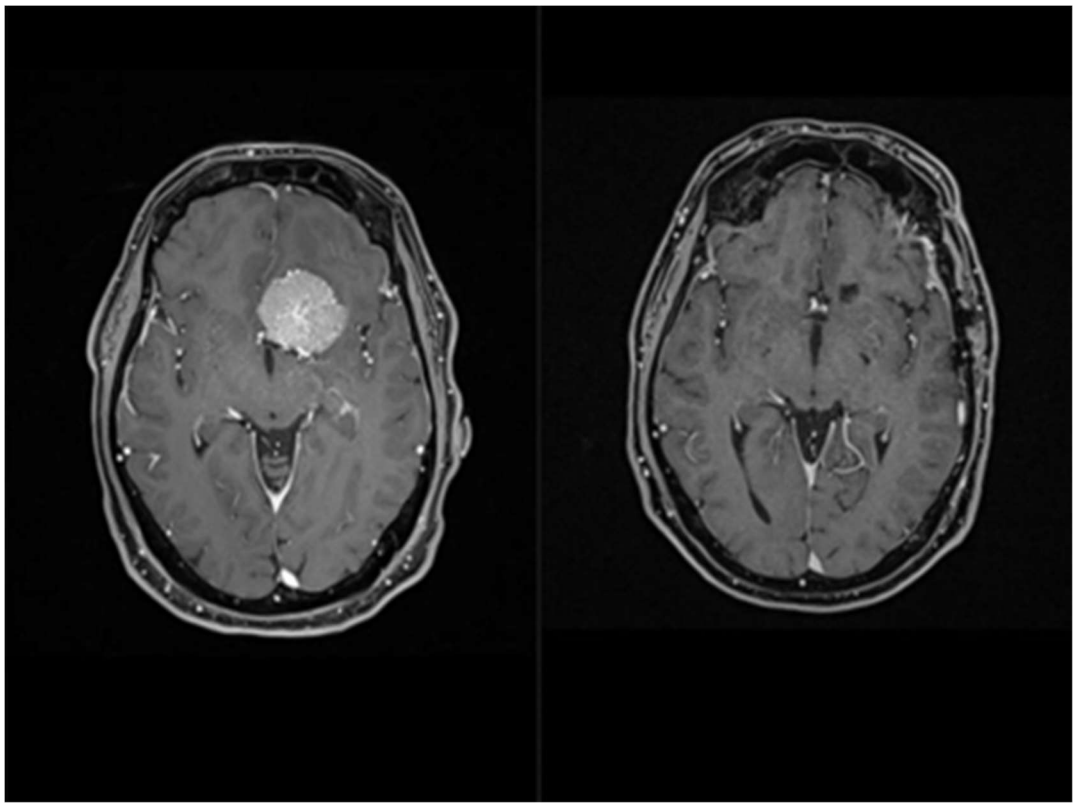


Figure 3:IRM pour la tumeur

7. Pourquoi pratiquer une IRM cérébrale ?

L'IRM cérébrale est effectuée à des fins diagnostiques. Il s'agit d'un examen de choix pour toutes les pathologies cérébrales. Elle est notamment prescrite :

- Pour déterminer la cause de maux de tête.
- Pour évaluer le débit sanguin ou la présence de caillots de sang au cerveau.
- En cas de confusion, de trouble de l'état conscience (provoqué par exemple par des maladies comme Alzheimer ou Parkinson).
- En cas d'hydrocéphalie (accumulation de liquide céphalo-rachidien dans le cerveau).
- Pour déceler la présence de tumeurs, d'infections, ou encore d'abcès.

- En cas de pathologies démyélinisantes (comme la sclérose en plaques), pour le diagnostic ou la surveillance.
- En cas d'anomalies faisant suspecter une atteinte du cerveau.

8. Conclusion

Dans ce chapitre, nous avons présenté en détail tout ce qui concerne les tumeurs cérébrales. Nous avons également introduit les concepts de base liés à principes d'acquisition d'images médicales basées sur l'IRM et qui La méthode la plus utilisée par rapport aux autres car elle est très utile pour surveillance du cerveau.

Dans notre travail, nous nous intéressons aux technologies d'apprentissage En profondeur (deep learning). Nous avons vu que le deep learning permet ordinateur pour créer des concepts complexes à partir de concepts plus simples. Les détails sont le sujet du chapitre suivant.

Chapitre 2 : Introduction à l'apprentissage en profondeur

1. Introduction

Nous débutons ce chapitre par une définition détaillée de l'apprentissage en profondeur dans laquelle nous présenterons une brève définition de l'algorithme d'apprentissage. Dans le cadre de ce chapitre, on parlera surtout de l'importance et le fonctionnement du deep learning. Pour finir nous citons rapidement, par la suite, les différents types de modèles utilisant des architectures d'apprentissage en profondeur.

2. Qu'est-ce que le Deep Learning ?

Le deep learning ou apprentissage profond peut être considéré comme un sous-ensemble de l'apprentissage automatique. C'est un domaine basé sur l'apprentissage et l'amélioration par lui-même en examinant les algorithmes informatiques. Alors que l'apprentissage automatique utilise des concepts plus simples, le deep learning fonctionne avec des réseaux de neurones artificiels, conçus pour imiter la façon dont les humains pensent et apprennent. Jusqu'à récemment, les réseaux de neurones étaient limités par la puissance de calcul et étaient donc limités en complexité. Cependant, les progrès de l'analyse du Big Data ont permis des réseaux de neurones plus vastes et sophistiqués, permettant aux ordinateurs d'observer, d'apprendre et de réagir à des situations complexes plus rapidement que les humains. Deep learning a aidé à la classification des images, à la traduction de la langue et à la reconnaissance vocale. Il peut être utilisé pour résoudre tout problème de reconnaissance de formes et sans intervention humaine.

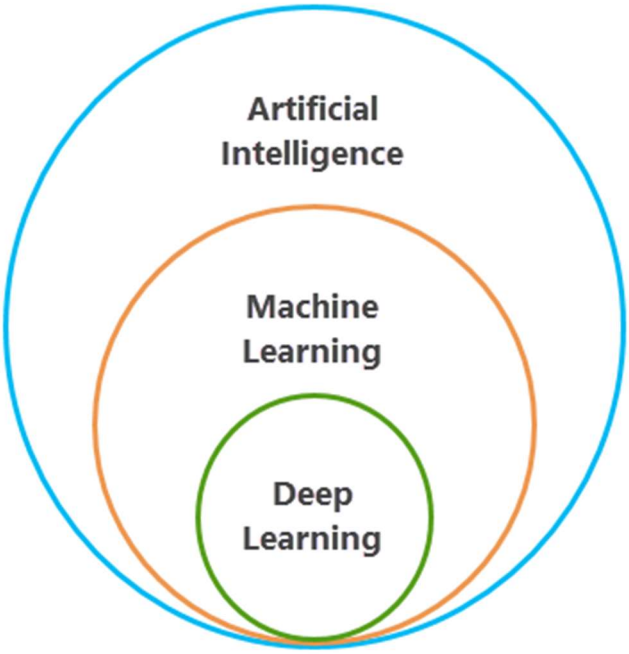


Figure 4: Définition de Deep Learning

3. L’importance de Deep Learning

- L’apprentissage automatique fonctionne uniquement avec des ensembles de données structurées et semi-structurées, tandis que l’apprentissage en profondeur fonctionne avec des données structurées et non structurées.
- Les algorithmes d’apprentissage en profondeur peuvent effectuer efficacement des opérations complexes, tandis que les algorithmes d’apprentissage automatique ne le peuvent pas.
- Les algorithmes d’apprentissage automatique utilisent des échantillons de données étiquetés pour extraire des modèles, tandis que l’apprentissage en profondeur accepte de gros volumes de données en entrée et analyse les données d’entrée pour extraire les caractéristiques d’un objet.
- Les performances des algorithmes d’apprentissage automatique diminuent à mesure que le nombre de données augmente ; donc pour maintenir les performances du modèle, nous avons besoin d’un apprentissage en profondeur.

4. Comment fonctionne le Deep Learning ?

Les réseaux de neurones sont des couches de nœuds, tout comme le cerveau humain est composé de neurones. Les nœuds au sein des couches individuelles sont connectés aux couches adjacentes. On dit que le réseau est plus profond en fonction du nombre de couches dont il dispose. Un seul neurone du cerveau humain reçoit des milliers de signaux d’autres neurones. Dans un réseau neuronal artificiel, les signaux voyagent entre les nœuds et attribuent des poids correspondants. Un nœud pondéré plus lourd exercera plus d’effet sur la couche de nœuds suivante. La couche finale compile les entrées pondérées pour produire une sortie. Les systèmes d’apprentissage en profondeur nécessitent un matériel puissant car ils traitent une grande quantité de données et impliquent plusieurs calculs mathématiques complexes. Cependant, même avec un matériel aussi avancé, les calculs de formation en apprentissage profond peuvent prendre des semaines.

Les systèmes d’apprentissage en profondeur nécessitent de grandes quantités de données pour renvoyer des résultats précis ; en conséquence, les informations sont alimentées sous la forme d’énormes ensembles de données. Lors du traitement des données, les réseaux de neurones artificiels sont capables de classer les données avec les réponses reçues à partir d’une série de questions binaires vraies ou fausses impliquant des calculs mathématiques très complexes. Par exemple, un programme de reconnaissance faciale fonctionne en apprenant à détecter et à reconnaître les arêtes et les lignes des visages, puis les parties les plus significatives des visages et, enfin, les représentations globales des visages. Au fil du temps, le programme s’entraîne et la probabilité de réponses correctes augmente. Dans ce cas, le programme de reconnaissance faciale identifiera avec précision les visages avec le temps.

5. Fonctionnement de base des réseaux de neurones :

Les réseaux de neurones (NN) forment la base de l'apprentissage en profondeur, un sous-domaine de l'apprentissage automatique où les algorithmes sont inspirés de la structure du cerveau humain. NN recueille des données, forme eux-mêmes pour reconnaître les modèles dans ces données, puis prédire les sorties pour un nouvel ensemble de données similaires. Les NN sont constitués de couches de neurones. Ces neurones sont les principales unités de traitement du réseau. Nous avons d'abord la couche d'entrée qui reçoit l'entrée ; la couche de sortie prédit notre sortie finale. Entre les deux, existent les couches cachées qui effectuent la plupart des calculs requis par notre réseau.

6. Types de modèles utilisant des architectures d'apprentissage en profondeur

L'apprentissage en profondeur offre une grande précision dans les ensembles de données encombrés. Les algorithmes d'apprentissage en profondeur préfèrent dans les applications de prédiction, de classification et d'identification. L'algorithme d'apprentissage en profondeur a été utilisé avec un réseau de neurones profonds (DNN), un réseau de neurones convolutifs (CNN) et un réseau neuronal récurrent (RNN) et une mémoire à long terme (LSTM) dans ces applications.

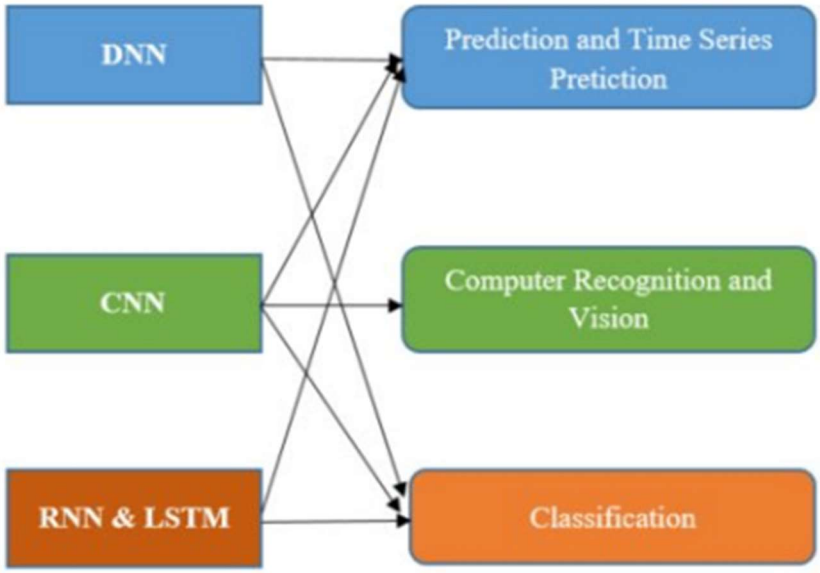


Figure 5:Types de modèles utilisant des architectures d'apprentissage en profondeur

a. Deep Neural Network (DNN)

Un réseau neuronal profond (DNN) est un réseau neuronal artificiel (ANN) avec plusieurs couches entre les couches d'entrée et de sortie. Il existe différents types de réseaux de neurones mais ils ont toujours les mêmes composants : neurones, synapses, poids, biais et fonctions. Ces composants fonctionnent de manière similaire au cerveau humain et peuvent être entraînés comme tout autre algorithme de ML.

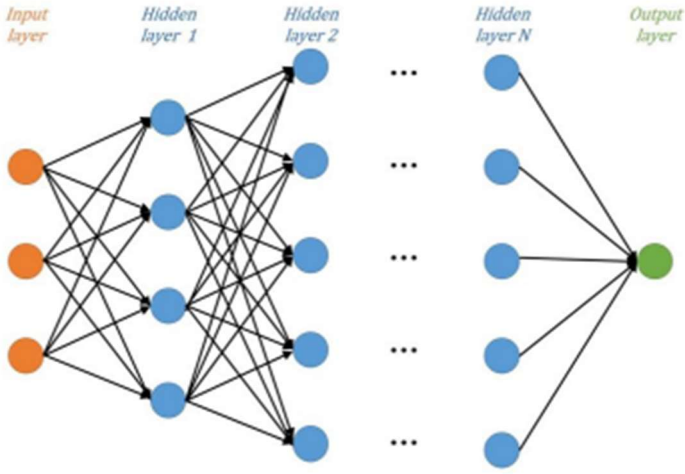


Figure 6:Construction du modèle de réseau neuronal profond -DNN

b. Recurrent Neural Network (RNN) et LSTM

Les réseaux neuronaux récurrents (RNN) et LSTM sont un type puissant et robuste de réseaux neuronaux et appartiennent aux algorithmes les plus prometteurs du moment car ils sont les seuls à avoir une mémoire interne. En raison de leur mémoire interne, les RNN sont capables de se souvenir de choses importantes concernant les données qu'ils ont reçues, ce qui leur permet d'être très précis dans la prédiction de ce qui va suivre. C'est la raison pour laquelle ils sont l'algorithme préféré pour les données séquentielles comme les séries temporelles, la parole, le texte, les données financières, audio, vidéo, météo et bien plus parce qu'ils peuvent former une compréhension beaucoup plus profonde d'une séquence et de son contexte. Algorithmes. Les réseaux neuronaux récurrents produisent des résultats prédictifs dans des données séquentielles que d'autres algorithmes ne peuvent pas produire.

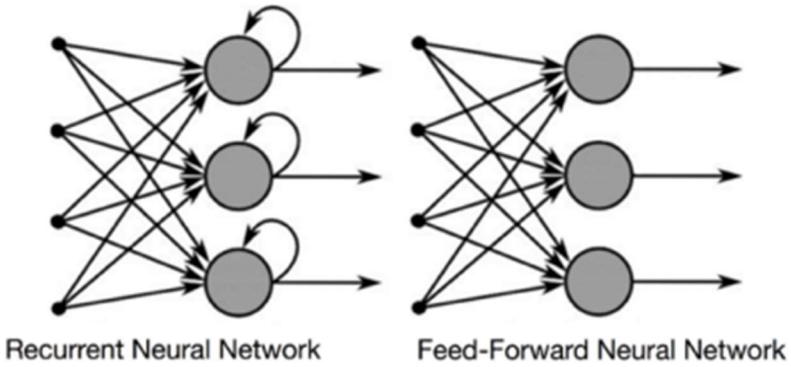


Figure 7:Architecture de RNN pour l'ouverture d'un réseau

c. Convolutional Neural Network

CNN est un réseau neuronal multicouche contenant la convolution, la mise en commun, activation et couches entièrement connectées. Les couches de convolution sont au cœur des CNN et sont utilisées pour l'extraction de caractéristiques. L'opération de convolution peut produire différentes cartes de

caractéristiques en fonction des filtres utilisés. La couche de regroupement effectue une opération de sous-échantillonnage en utilisant le maximum ou la moyenne du voisinage déni comme valeur pour réduire la taille spatiale de chaque carte d'entités. La couche rectifiée non linéaire (ReLU) transforme les données en écrêtant toutes les valeurs d'entrée négatives à zéro tandis que les valeurs d'entrée positives sont passées en sortie. Les neurones d'une couche entièrement connectée sont entièrement connectés à toutes les activations de la couche précédente. Ils sont placés avant la sortie de classification d'un CNN et sont utilisés pour aplatir les résultats avant qu'une prédiction ne soit faite à l'aide de classificateurs linéaires. Lors de la formation de l'architecture CNN, le modèle prédit les scores de classe pour les images de formation, calcule la perte en utilisant la perte sélectionnée.

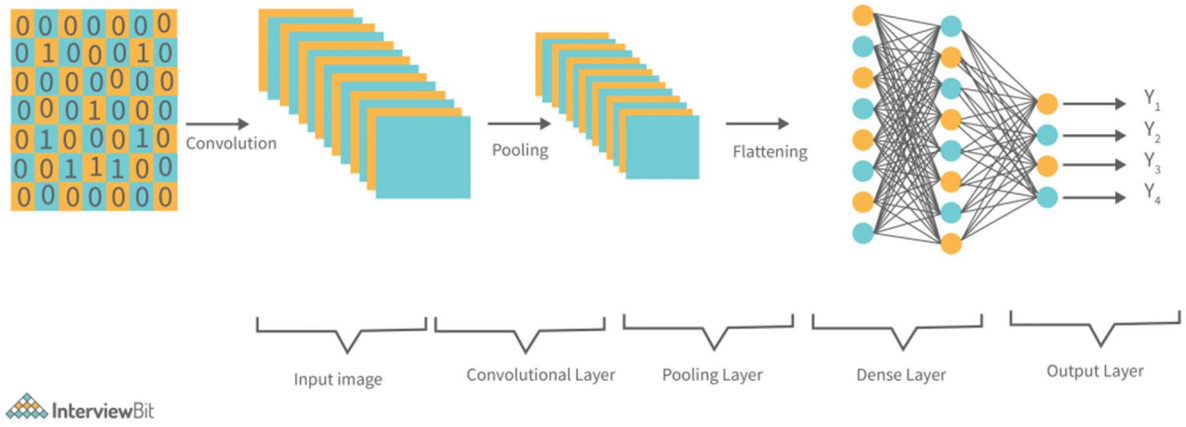


Figure 8:structure de CNN

i. Les Différentes couches de CNN

Il existe quatre couches principales opérations illustrées dans le CNN à savoir :

- **La couche de convolution :**

La couche de convolution est la couche la plus importante de l'ensemble du réseau neuronal et constitue toujours au moins leur première couche, c'est un outil mathématique et la partie principale de cette couche est le filtre. L'opération de convolution utilisé un jeu de filtres, Son but est de repérer la présence d'un ensemble de caractéristiques (Features) dans les images reçues en entrée, très utilisé en retouche d'image, car il permet d'en faire ressortir l'extraction des caractéristiques à partir des images d'entrées, an d'appliquer un bon filtre. Une convolution prend simplement en entrée une image et un filtre, effectue un calcul, puis renvoie une nouvelle image appelée une carte d'activation, ou feature map. Le filtre est caractérisé par une petite matrice de nombres (appelée noyau ou filtre), cette matrice est passée sur l'image entrée à n de la transformer en fonction des valeurs du filtre. Les valeurs de carte d'activation suivantes sont calculées selon la formule suivante : $G[m\,;\,n] = (f\,h)[m\,;\,n]$

Telle que f l'image entrée et h son filtre. Les valeurs m et n sont des indices utilisés pour parcourir l'image.

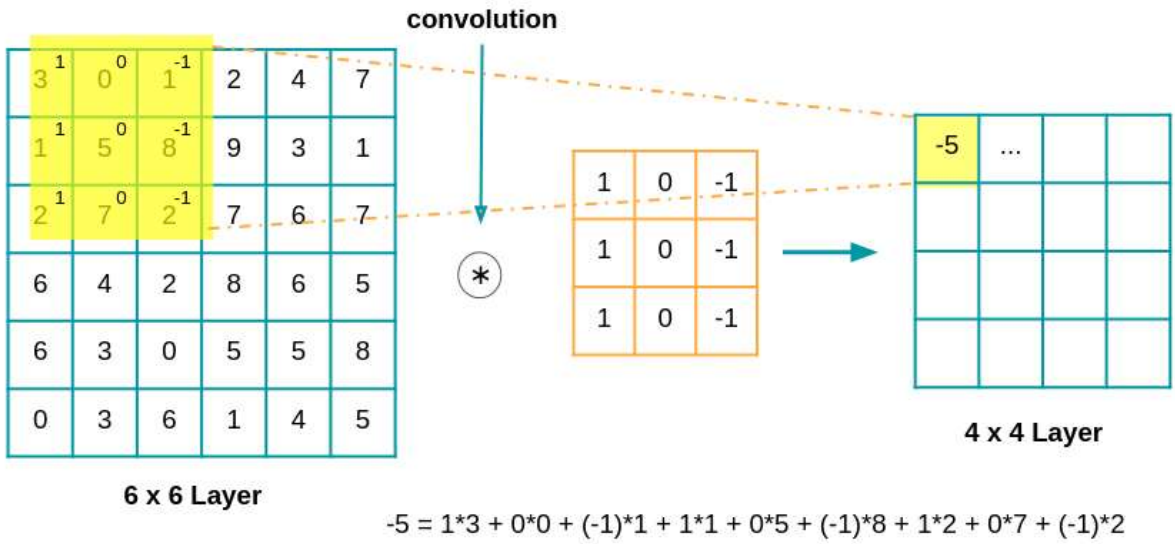


Figure 9:Couche de convolution

- **Couches de correction (ReLU)**

Acronyme de RectiedLinear Unit (unité linéaire rectifiée), c'est fonction d'activation utilisée après chaque opération de convolution, très couramment utilisée. Elle remplace toutes les valeurs pixels négatives sont mises à zéro. La fonction Relu est interprétée par la formule : f(x) = max(0 ; x) il existe plusieurs fonctions d'activation comme La correction par tangente hyperbolique, la correction par la fonction sigmoïde, la correction par la tangente hyperbolique saturante. Cependant, la correction Relu est préférable, car il en résulte la formation de réseau neuronal plusieurs fois plus rapide, sans faire une différence significative à la généralisation de précision .

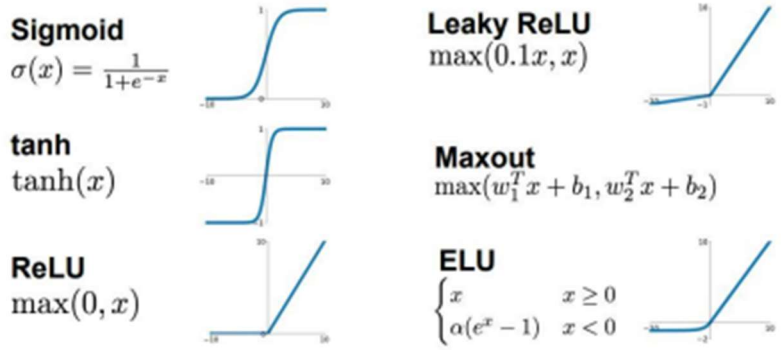


Figure 10:Fonctions d'activation

- La couche de d'union (pooling)

Un autre outil très puissant utilisé par les CNN s'appelle le Pooling. C'est une forme de sous-échantillonnage de l'image, qui consiste à réduire progressivement la taille de l'image en ne gardant que les informations les plus importantes. Il existe plusieurs types de couches de pooling, tel que : Le

max pooling qui revient à prendre la valeur maximale de la sélection C'est le type le plus utilisé, car il est rapide à calculer (immédiat). Le mean pooling soit la moyenne des pixels de la sélection : on calcule la somme de toutes les valeurs et on divise par le nombre de valeurs. Le sum pooling, c'est la somme sans avoir divisé par le nombre de valeurs (on ne calcule que leur somme). Nous nous focaliserons sur l'étude de la couche d'union par maximum, car elle est dans de nombreux cas plus performante que la couche d'union par moyenne. Dans ce cas on définit un voisinage spatial ; par exemple, une fenêtre 2 2 dans la carte de caractéristiques(featuremap)et de prendre le plus grand élément dans cette fenêtre. La figure 2.9 montre un exemple d'un filtre de max pooling de taille 2*2.

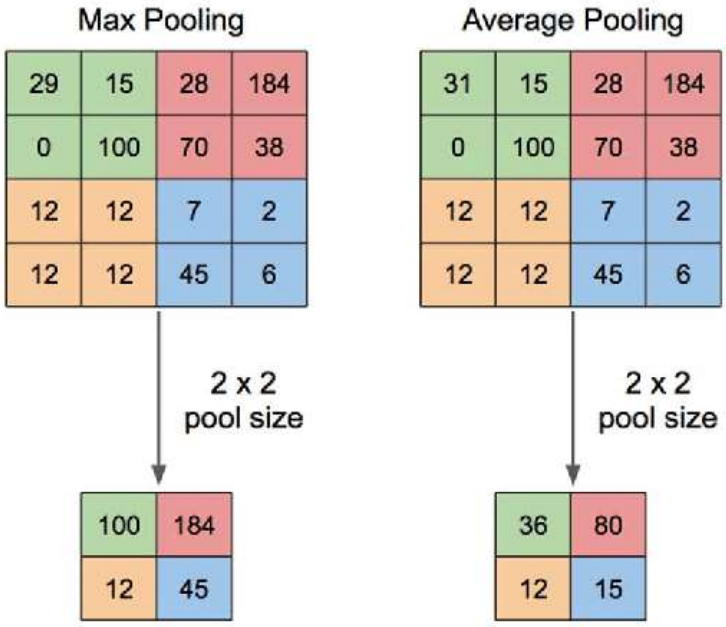


Figure 11:Couche pooling (Max pooling & Average pooling)

- Couche entièrement connectée (FC)

Après plusieurs couches de convolution et de max-pooling, le max raisonnement de haut niveau dans le réseau neuronal se fait via des couches entièrement connectées. La couche entièrement connectée (Fully Connected layer) est une traditionnelle perception multicouche (Multi Layer Perceptron) Cette couche est utilisée pour associer les différents motifs et d'en déduire la classe. Comme son nom l'indique, la matrice de connexion est entièrement connectée. Utilisant une fonction d'activation notamment appelée softmax dans la couche de sortie. Le terme entièrement connecté implique que chaque neurone dans la couche précédente est connecté à chaque neurone sur la couche suivante. Le but de la couche entièrement connectée est de pouvoir utiliser ces fonctions pour classer l'image d'entrée dans différentes classes en fonction de l'ensemble de données d'apprentissage.

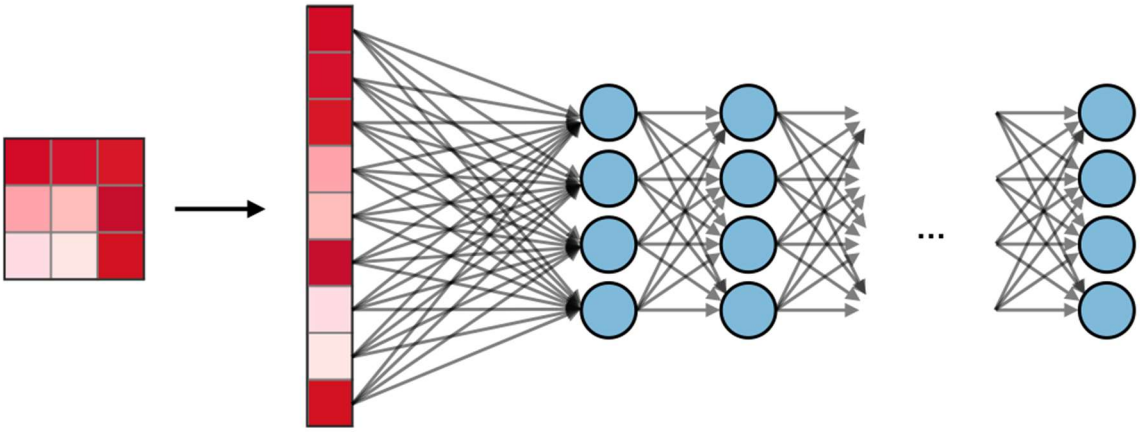


Figure 12:Couche entièrement connectées

ii. Les avantages de CNN

- L'utilisation des CNN pour l'apprentissage en profondeur est devenue de plus en plus populaire en raison de trois facteurs importants.
- Les CNN éliminent le besoin d'extraction manuelle des fonctionnalités
- Les fonctionnalités sont apprises directement par le CNN.
- Les CNN produisent des résultats de reconnaissance de pointe.
- Les CNN peuvent être recyclés pour de nouvelles tâches de reconnaissance, ce qui vous permet de vous appuyer sur des réseaux préexistants.
- Les CNN offrent la possibilité de calculer automatiquement des cartes de caractéristiques (feature maps), évitant ainsi à l'utilisateur d'effectuer des calculs de caractéristiques lourds

7. Conclusion

Dans ce chapitre, nous avons vu premièrement ce que le Deep Learning, et Pourquoi nous avons choisi le deep learning, ensuite nous avons également parlé des trois types de modèles utilisant des architectures d'apprentissage en profondeur, nous avons parlé dans ce chapitre également sur quelques explications sur les réseaux de neurone convolutifs, et les couches principales nous avons terminé ce chapitre par les avantages de CNN. Dans le chapitre suivant, nous avons collecté et prétraité les images d'IRM afin de construire un model CNN pour la classification des patient soit ont une tumeur cérébrale ou non.

Chapitre 3 : Construction du model CNN

1. Environnement de travail
   a. DataSpell

DataSpell est un environnement de développement intégré (IDE) développé par JetBrains, spécifiquement conçu pour le travail avec des données scientifiques. Il s'adresse principalement aux data scientists et aux analystes de données. Voici quelques-unes de ses caractéristiques principales :

- **Support des notebooks** : DataSpell prend en charge les notebooks Jupyter, ce qui facilite l'exécution de code, la visualisation des résultats et la manipulation des données en temps réel.
- **Intégration de Python** : Il offre un support avancé pour Python, avec des fonctionnalités comme l'auto-complétion, la gestion des environnements virtuels et des outils de débogage.
- **Visualisation des données** : L'IDE permet d'intégrer des bibliothèques de visualisation comme Matplotlib et Seaborn, et offre des outils pour afficher et analyser graphiquement les données.
- **Exploration des données** : Avec des outils pour naviguer et manipuler les bases de données, les fichiers CSV, et d'autres formats de données courants.
- **Gestion de l'environnement scientifique** : Il s'intègre bien avec des bibliothèques de science des données comme NumPy, Pandas, Scikit-learn, et TensorFlow.

   b. Jupyter

Jupyter est une plateforme open source utilisée principalement pour l'écriture, l'exécution et le partage de code dans des notebooks interactifs. Les **notebooks Jupyter** sont largement utilisés dans les domaines de la science des données, de l'apprentissage automatique, de la recherche et de l'éducation. Voici quelques points clés à propos de Jupyter :

- **Notebooks interactifs** : Un notebook Jupyter permet d'écrire du texte explicatif (en Markdown), du code (en Python, R, Julia, etc.), d'exécuter ce code et de voir les résultats directement dans le même document.
- **Langages pris en charge** : Jupyter est principalement associé à Python, mais il prend en charge de nombreux langages via des "kernels". Le **kernel** le plus populaire est celui de Python (IPython), mais il existe des kernels pour d'autres langages comme R, Julia, Scala, et plus encore.

- **Visualisation des données** : Il permet de visualiser des graphiques et des résultats directement dans les cellules du notebook en intégrant des bibliothèques de visualisation comme Matplotlib, Seaborn, Plotly, etc.
- **Reproductibilité** : Les notebooks peuvent être partagés et rejoués, ce qui est utile pour la reproductibilité des résultats dans les recherches scientifiques.
- **Extensions et widgets** : Jupyter peut être enrichi avec des extensions et des widgets interactifs qui permettent de personnaliser l'expérience utilisateur, comme l'ajout de boutons, de barres de défilement, et d'autres éléments d'interface.
- **Environnement flexible** : Les notebooks Jupyter peuvent être exécutés localement ou via des plateformes cloud comme Google Colab, ce qui rend l'environnement très flexible et accessible.

c. Python

Python est un langage de programmation polyvalent, interprété et open source, très populaire pour son simplicité et sa **lisibilité**. Voici un aperçu de Python en bref :

- **Facile à apprendre et à utiliser** : Python a une syntaxe claire et simple, ce qui le rend accessible même aux débutants.
- **Polyvalence** : Utilisé dans divers domaines, notamment le développement web, la science des données, l'intelligence artificielle, l'automatisation, la finance, les jeux, et bien plus encore.
- **Langage interprété** : Python est exécuté directement sans compilation préalable, ce qui facilite le débogage et le développement rapide.
- **Large écosystème de bibliothèques** : Python dispose d'une immense collection de bibliothèques et de modules pour faciliter des tâches spécifiques comme :
    1. **NumPy, Pandas** : Manipulation et analyse de données
    2. **Matplotlib, Seaborn** : Visualisation de données
    3. **Scikit-learn, TensorFlow, PyTorch** : Apprentissage automatique et intelligence artificielle
    4. **Django, Flask** : Développement web
- **Communauté large et active** : Python bénéficie d'une grande communauté d'utilisateurs et de développeurs qui créent constamment des ressources, des bibliothèques, et fournissent du support.

- **Multi-paradigmes** : Python prend en charge la programmation procédurale, orientée objet et fonctionnelle.
- **Portabilité** : Il fonctionne sur presque toutes les plateformes (Windows, macOS, Linux).

2. Bibliothèques utilisées

Numpy (np) : Utilisé pour les opérations numériques et la manipulation de tableaux.

Torch (torch) : La bibliothèque principale pour le deep learning avec PyTorch.

Torch Datasets :

- Dataset : Classe de base pour créer des datasets personnalisés.
- DataLoader : Utilisé pour charger les données par lots.
- ConcatDataset : Permet de combiner plusieurs datasets.

Glob (glob) : Utilisé pour récupérer des chemins de fichiers correspondant à un motif spécifique.

Matplotlib (plt) : Utilisé pour créer des graphiques et visualiser des données.

Métriques Scikit-learn :

- confusion_matrix : Permet de calculer la matrice de confusion.
- accuracy_score : Permet de calculer la précision du modèle.

OpenCV (cv2) : Utilisé pour le traitement d'images et de vidéos.

Seaborn (sns) : Utilisé pour visualiser des matrices de confusion avec des heatmaps.

Sélection de Modèles Scikit-learn :

- train_test_split : Permet de diviser les données en ensembles d'entraînement et de test.

torch.nn : Ce module de PyTorch fournit des classes pour construire des blocs de réseaux de neurones, comme des couches convolutives, des couches entièrement connectées, des fonctions d'activation, etc.

torch.nn.functional : Ce module contient des fonctions utilisées pour appliquer diverses opérations dans les réseaux de neurones, comme les fonctions d'activation (ReLU, Sigmoid, etc.), les convolutions, le pooling, et d'autres opérations.

```python
import numpy as np
import torch
from torch.utils.data import Dataset, DataLoader, ConcatDataset
import glob
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, accuracy_score
import cv2
import seaborn as sns
from sklearn.model_selection import train_test_split
import torch.nn as nn
import torch.nn.functional as F
```

Figure 13:Bibliothèques utilisées

3. Dataset des images

L'ensemble de données IRM utilisé dans ce travail est un ensemble de données open source accessible au public qui devient étiqueté en deux leçons avec une tumeur (Oui) et sans tumeur (Non).

L'ensemble de données est des données catégorisées qui ont été rassemblées par le biais d'examinateurs médicaux et comprenant des radiologues, des médecins et partagées sur le net. De plus, de nombreuses recherches ont été effectuées sur cet ensemble de données.

La base de données contient au total 253 images de MR cérébrales de différentes formes et longueurs. Cet ensemble de données a en outre étiqueté en deux types avec tumeur et sans tumeur, 155 images appartenaient à la classe tumorale et 98 images MR appartiennent à l'élégance normale. La forme de l'ensemble de données est hétérogène et l'épine dorsale des images est également non uniforme. Le jeu de données à le plus souvent le format jpeg, mais peu de photos sont au format .png. La figure 1 illustre l'ensemble de données MR selon leurs étiquettes, sans tumeur montrée dans une catégorie avec l'étiquette « Non » et avec tumeur ont été montrées dans l'autre avec étiquette oui.

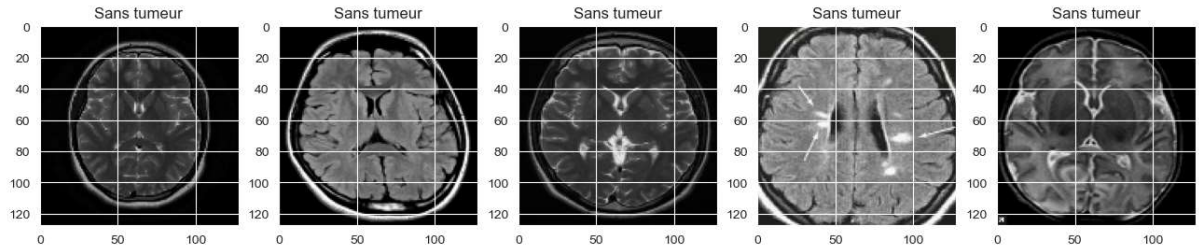


Figure 14:Echantillion des images sans tumeur

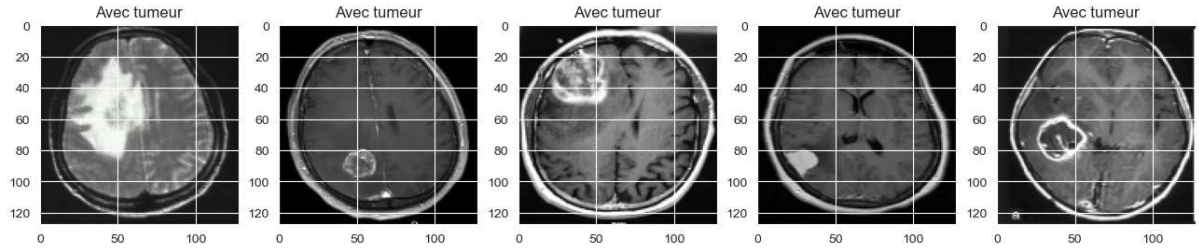


Figure 16:Echantillion des images avec tumeur

```python
class TumorDT(Dataset):
    def __init__(self):
        with_tumor = []
        no_tumor = []
        for f in glob.iglob("./data/yes/*.jpg"):
            img = cv2.imread(f)
            img = cv2.resize(img,(128,128))
            b, g, r = cv2.split(img)
            img = cv2.merge([r,g,b])
            img = img.reshape((img.shape[2],img.shape[0],img.shape[1]))
            with_tumor.append(img)
        for f in glob.iglob("./data/no/*.jpg"):
            img = cv2.imread(f)
            img = cv2.resize(img,(128,128))
            b, g, r = cv2.split(img)
            img = cv2.merge([r,g,b])
            img = img.reshape((img.shape[2],img.shape[0],img.shape[1]))
            no_tumor.append(img)
        with_tumor = np.array(with_tumor,dtype=np.float32)
        no_tumor = np.array(no_tumor,dtype=np.float32)
        with_tumor_label = np.ones(with_tumor.shape[0], dtype=np.float32)
        no_tumor_label = np.zeros(no_tumor.shape[0], dtype=np.float32)
        self.images = np.concatenate((with_tumor, no_tumor), axis=0)
        self.labels = np.concatenate((with_tumor_label, no_tumor_label))
    def __len__(self):
        return self.images.shape[0]
    def __getitem__(self, index):
        item = {'image': self.images[index], 'label':self.labels[index]}
        return item
    def normalize(self):
        self.images = self.images/255.0
```

Figure 15:Dataset de images

4. Prétraitement des images

Le traitement des images est une étape cruciale dans le développement de modèles d'apprentissage automatique, notamment pour la détection de tumeurs. Une des techniques essentielles est la **normalisation**, qui consiste à ajuster les valeurs des pixels d'image, généralement comprises entre 0 et

255, dans une plage plus standardisée, comme 0 à 1. Cette approche facilite l'apprentissage et la convergence des modèles en réduisant les risques de disparités dans l'échelle des données d'entrée. Dans notre classe TumorDT, nous chargons les images à partir de deux répertoires, un contenant des images avec des tumeurs et l'autre sens. Chaque image est redimensionnée à 128x128 pixels, et les canaux de couleur sont réorganisés avant d'être convertis en un tableau NumPy. Après avoir constitué les ensembles d'images et d'étiquettes, nous appliquons la normalisation en divisant chaque valeur de pixel par 255.0, assurant ainsi que toutes les données d'entrée sont dans une plage cohérente, ce qui est essentiel pour optimiser les performances de nos modèles de classification.

5. Définition du modèle

Nous créons ce modèle en utilisant un Réseau de Neurones Convolutif (CNN), qui est particulièrement adapté aux tâches de classification d'images. D'abord, nous définissons le modèle convolutionnel à travers une séquence de couches, en utilisant **nn.Conv2d** pour appliquer des convolutions à l'image d'entrée, et **nn.Tanh()** pour introduire une non-linéarité via la fonction d'activation tangente hyperbolique. Ensuite, nous réalisons un sous-échantillonnage avec **nn.AvgPool2d**, ce qui permet de réduire la taille de l'image tout en préservant ses caractéristiques importantes. Chaque étape est conçue pour extraire des informations essentielles de manière progressive et hiérarchique.

Ensuite, nous passons à la construction du modèle fully connected, où chaque neurone est connecté à tous les autres grâce à **nn.Linear**. Cela nous permet de rassembler les informations extraites par les couches convolutives et d'effectuer la classification finale. À la sortie du modèle, nous utilisons une fonction **sigmoïde** pour générer une probabilité entre 0 et 1, correspondant à la probabilité qu'une image contienne ou non une tumeur.

Enfin, pour transformer les probabilités en étiquettes de classe, nous utilisons une fonction de conversion qui applique un seuil de 0,5 pour classer les résultats comme "avec tumeur" ou "sans tumeur". Ce modèle reflète notre approche rigoureuse pour capturer les détails des images médicales et prendre des décisions fiables.

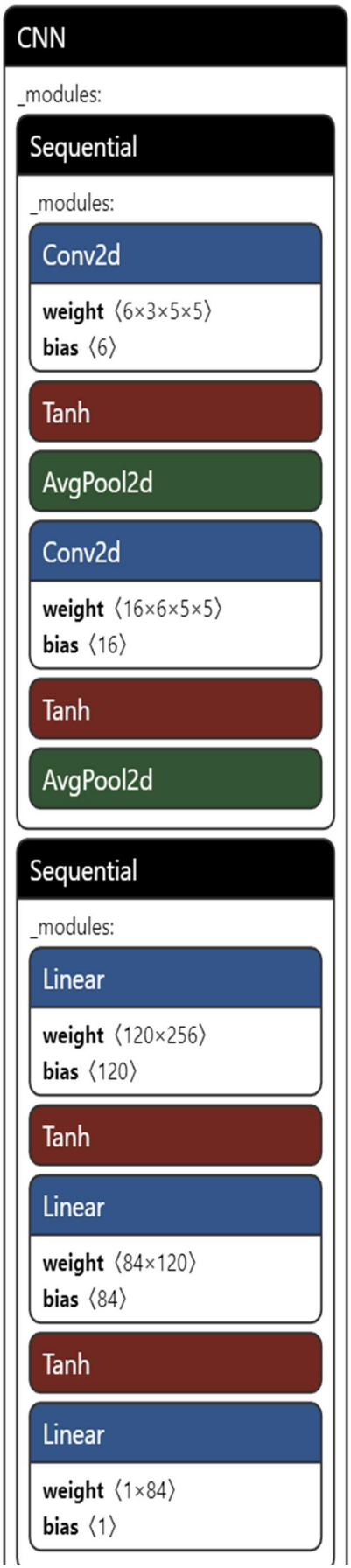


Figure 17:Structure du model CNN

```python

class CNN(nn.Module):
    def __init__(self):
        super(CNN,self).__init__()
        self.cnn_model = nn.Sequential(
            nn.Conv2d(in_channels=3, out_channels=6, kernel_size=5),
            nn.Tanh(),
            nn.AvgPool2d(kernel_size=2, stride=5),
            nn.Conv2d(in_channels=6, out_channels=16, kernel_size=5),
            nn.Tanh(),
            nn.AvgPool2d(kernel_size=2, stride=5))

        self.fc_model = nn.Sequential(
            nn.Linear(in_features=256, out_features=120),
            nn.Tanh(),
            nn.Linear(in_features=120, out_features=84),
            nn.Tanh(),
            nn.Linear(in_features=84, out_features=1))

    def forward(self, x):
        x = self.cnn_model(x)
        x = x.view(x.size(0), -1)
        x = self.fc_model(x)
        x = F.sigmoid(x)
        return x
    def convertOutput(self,scores,threshold=0.50, minimum=0, maximum = 1.0):
        x = np.array(list(scores))
        x[x >= threshold] = maximum
        x[x < threshold] = minimum
        return x
```

Figure 18:Code du model CNN

6. Evaluation du modèle non trainé

L'évaluation de notre modèle de classification d'images est une étape essentielle pour comprendre sa performance et sa capacité à généraliser sur des données inédites. Nous adoptons plusieurs méthodes pour mesurer l'efficacité de notre modèle, en utilisant des métriques standardisées qui fournissent une vue d'ensemble complète.

D'abord, nous utilisons la **précision** pour évaluer la proportion de bonnes classifications parmi l'ensemble des prédictions. Cela nous aide à déterminer dans quelle mesure notre modèle peut identifier correctement les images avec et sans tumeurs. En complément, nous analysons la **matrice de confusion**, qui nous permet de visualiser les faux positifs et les faux négatifs. Cela nous donne des indications précieuses sur les types d'erreurs que notre modèle commet.

```
accuracy_score(y_true, model.convertOutput(outputs))
```

```
0.6285714285714286
```

Figure 19:Précision du modèle non trainé

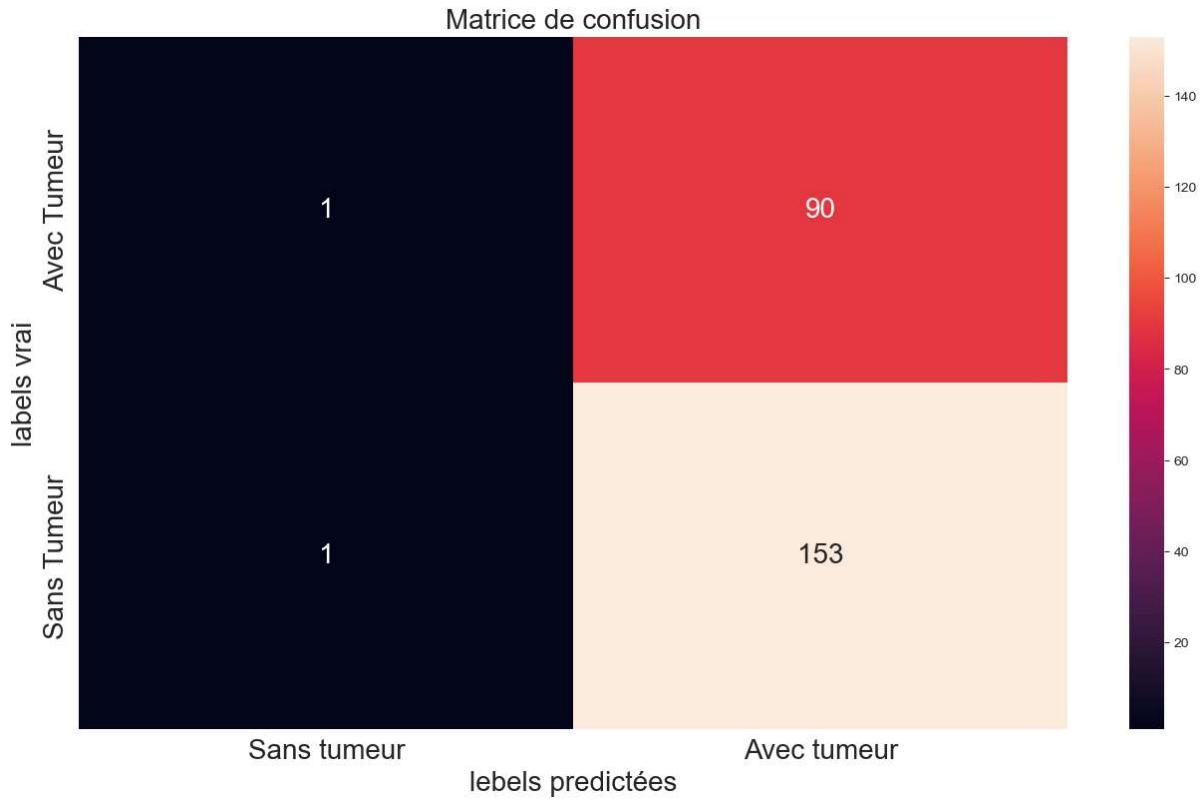


Figure 20:Matrice de confusion du modèle non trainé

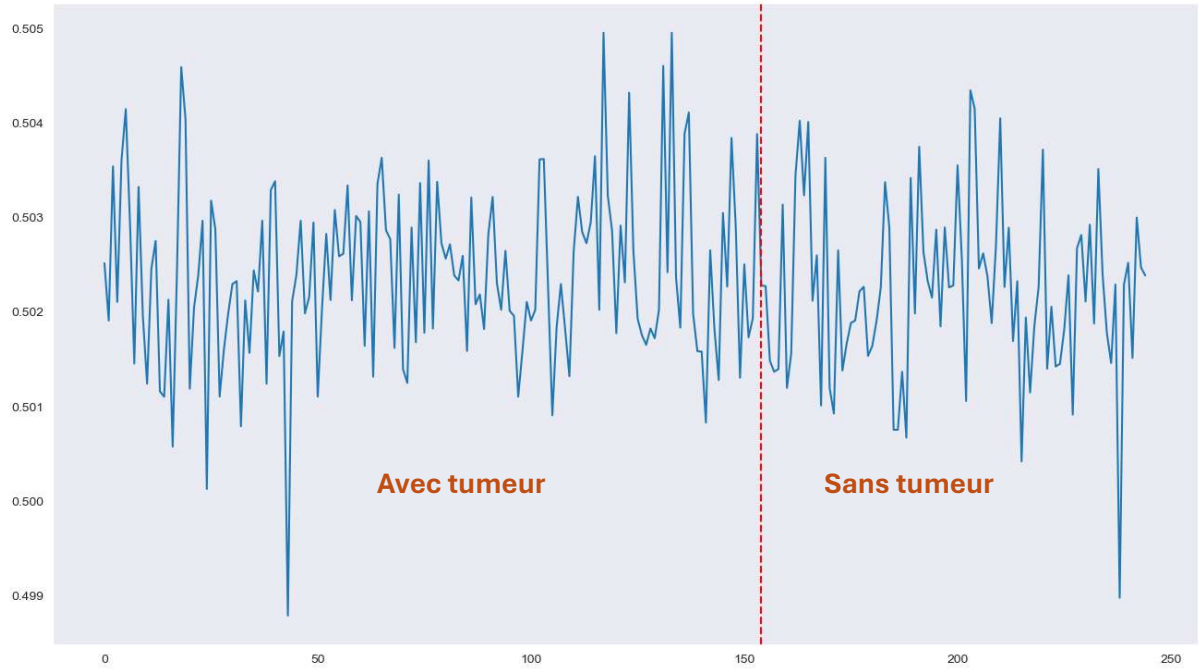


Figure 21:Résultat des prédictions du modèle non trainé

7. Entrainement du modèle

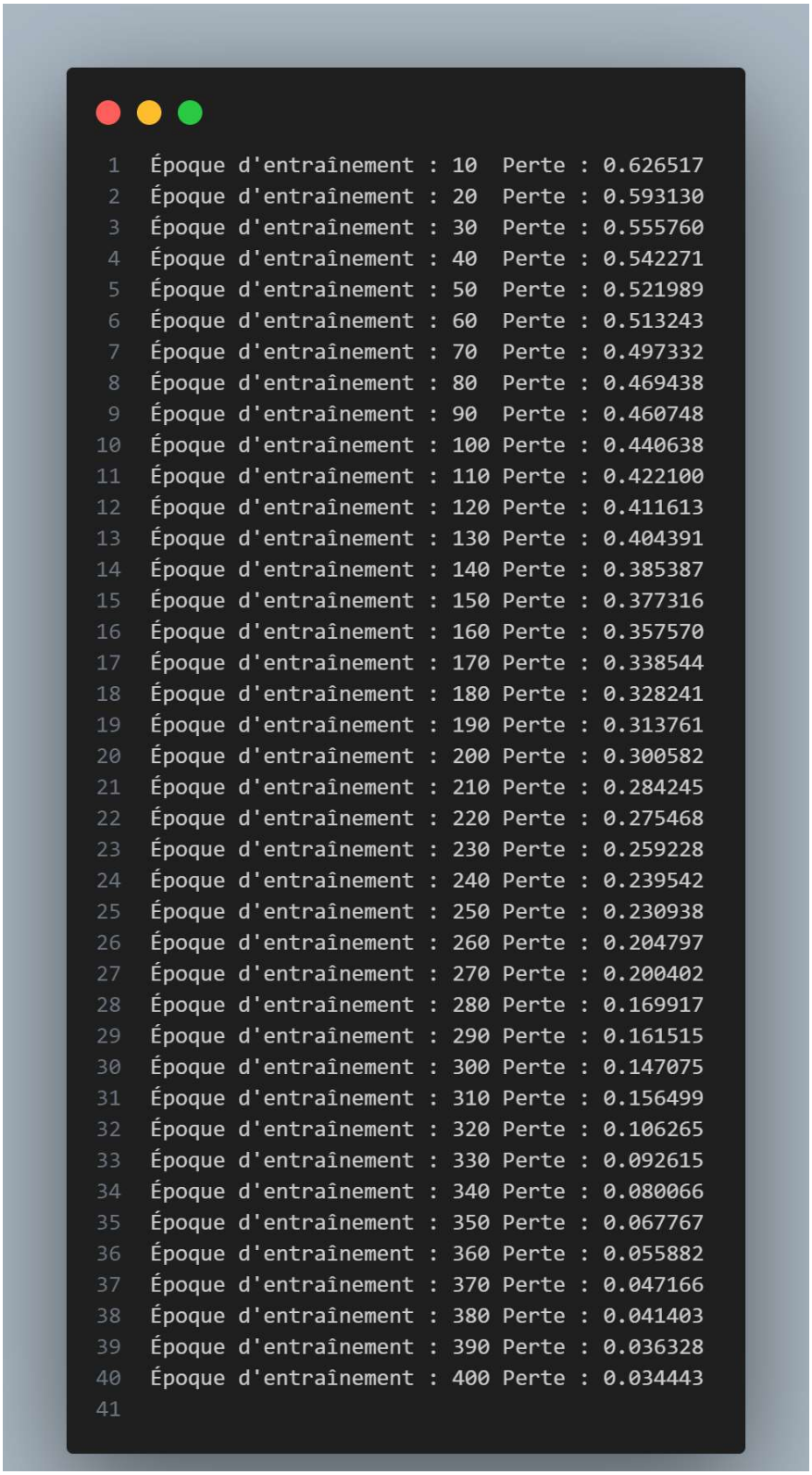
L'entraînement de notre modèle est une phase cruciale qui permet d'ajuster les poids et les biais afin qu'il puisse apprendre à classer efficacement les images. Pour cela, nous utilisons l'algorithme d'optimisation Adam, qui est particulièrement efficace pour gérer les gradients et améliorer la vitesse de convergence. Nous avons défini un taux d'apprentissage (eta) de 0.0001, ce qui est une valeur typique pour garantir un ajustement progressif des paramètres.

Nous avons choisi d'entraîner le modèle sur 400 époques, ce qui nous permet de donner suffisamment de temps au modèle pour apprendre les caractéristiques essentielles des images. Pendant chaque époque, nous passons à travers les données par lots de 32 images, ce qui permet d'optimiser l'utilisation de la mémoire tout en rendant l'entraînement plus efficace.

Au début de chaque itération sur les époques, nous initialisons une liste losses pour enregistrer la perte associée à chaque lot. À chaque lot, nous commençons par réinitialiser les gradients avec optimizer.zero_grad(). Ensuite, nous passons les images à travers le modèle pour obtenir les prédictions y_hat. Pour évaluer la performance du modèle, nous utilisons la fonction de perte BCELoss, qui est appropriée pour les problèmes de classification binaire. Cette fonction mesure la différence entre les prédictions du modèle et les vraies étiquettes.

Après avoir calculé la perte, nous effectuons une rétropropagation avec loss.backward() pour mettre à jour les gradients. Enfin, nous appliquons l'optimiseur avec optimizer.step(), ce qui ajuste les poids du modèle en fonction des gradients calculés.

Pour suivre l'évolution de l'entraînement, nous affichons la perte moyenne tous les 10 epochs. Cela nous permet de vérifier si le modèle apprend correctement. Une perte qui diminue au fil des époques est généralement un bon indicateur que le modèle s'améliore et apprend à classer les images de manière plus précise.



```
1  Époque d'entraînement : 10  Perte : 0.626517
2  Époque d'entraînement : 20  Perte : 0.593130
3  Époque d'entraînement : 30  Perte : 0.555760
4  Époque d'entraînement : 40  Perte : 0.542271
5  Époque d'entraînement : 50  Perte : 0.521989
6  Époque d'entraînement : 60  Perte : 0.513243
7  Époque d'entraînement : 70  Perte : 0.497332
8  Époque d'entraînement : 80  Perte : 0.469438
9  Époque d'entraînement : 90  Perte : 0.460748
10 Époque d'entraînement : 100 Perte : 0.440638
11 Époque d'entraînement : 110 Perte : 0.422100
12 Époque d'entraînement : 120 Perte : 0.411613
13 Époque d'entraînement : 130 Perte : 0.404391
14 Époque d'entraînement : 140 Perte : 0.385387
15 Époque d'entraînement : 150 Perte : 0.377316
16 Époque d'entraînement : 160 Perte : 0.357570
17 Époque d'entraînement : 170 Perte : 0.338544
18 Époque d'entraînement : 180 Perte : 0.328241
19 Époque d'entraînement : 190 Perte : 0.313761
20 Époque d'entraînement : 200 Perte : 0.300582
21 Époque d'entraînement : 210 Perte : 0.284245
22 Époque d'entraînement : 220 Perte : 0.275468
23 Époque d'entraînement : 230 Perte : 0.259228
24 Époque d'entraînement : 240 Perte : 0.239542
25 Époque d'entraînement : 250 Perte : 0.230938
26 Époque d'entraînement : 260 Perte : 0.204797
27 Époque d'entraînement : 270 Perte : 0.200402
28 Époque d'entraînement : 280 Perte : 0.169917
29 Époque d'entraînement : 290 Perte : 0.161515
30 Époque d'entraînement : 300 Perte : 0.147075
31 Époque d'entraînement : 310 Perte : 0.156499
32 Époque d'entraînement : 320 Perte : 0.106265
33 Époque d'entraînement : 330 Perte : 0.092615
34 Époque d'entraînement : 340 Perte : 0.080066
35 Époque d'entraînement : 350 Perte : 0.067767
36 Époque d'entraînement : 360 Perte : 0.055882
37 Époque d'entraînement : 370 Perte : 0.047166
38 Époque d'entraînement : 380 Perte : 0.041403
39 Époque d'entraînement : 390 Perte : 0.036328
40 Époque d'entraînement : 400 Perte : 0.034443
41
```

Figure 22:Entrainement du modèle

8. Evaluation du modèle trainé

```
accuracy_score(y_true, model.convertOutput(outputs))
```

```
1.0
```

Figure 23:Précision du modèle trainé

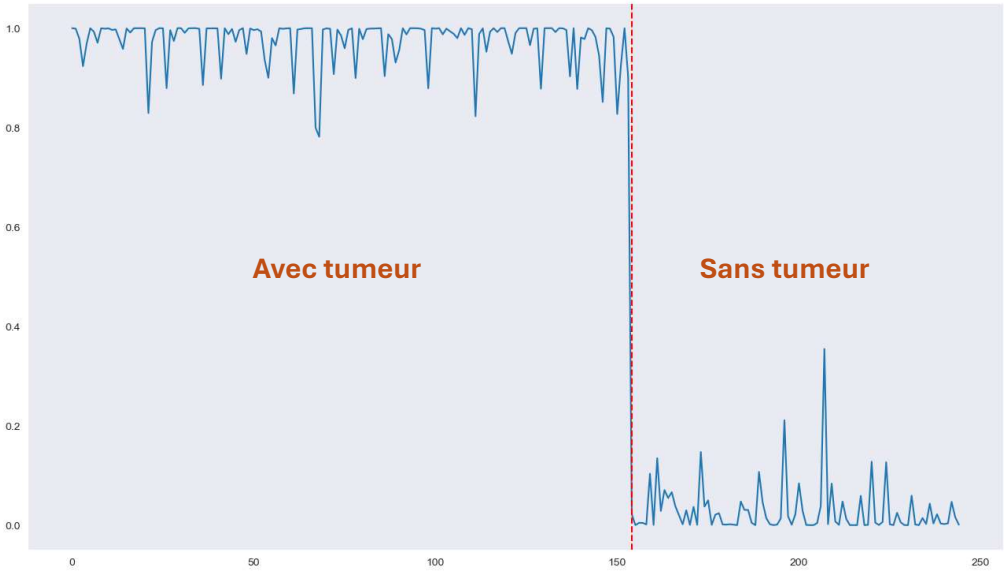


Figure 24:Résultats de prédiction du modèle trainé

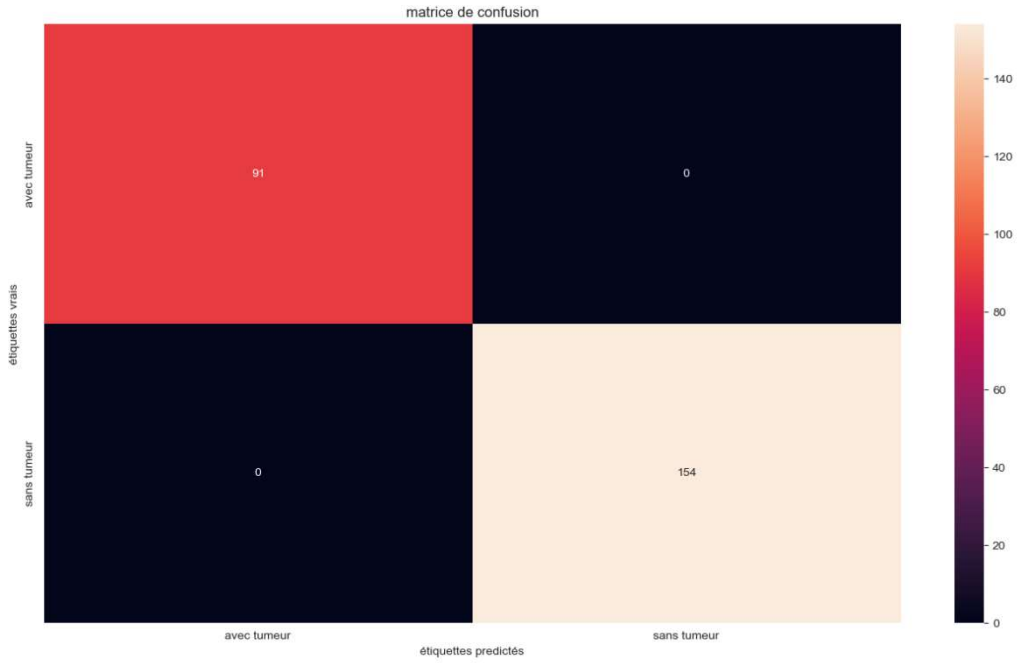


Figure 25:Matrice de confusion du modèle trainé

9. Enregistrement et chargement du modèle

L'enregistrement et le chargement du modèle sont des étapes essentielles pour garantir que notre travail d'entraînement n'est pas perdu et que nous pouvons facilement réutiliser le modèle à l'avenir. Pour cela, nous utilisons la fonction **torch.save** pour sauvegarder notre modèle dans un fichier. Dans notre exemple, nous avons choisi de stocker le modèle sous le chemin PATH='./outputs/model.pt'. Cela nous permet de conserver une copie persistante de l'état du modèle après son entraînement.

L'enregistrement du modèle inclut tous les paramètres appris ainsi que la structure du modèle lui-même, ce qui nous permet de le restaurer ultérieurement sans avoir à réentraîner le modèle. Cette fonctionnalité est particulièrement utile dans des contextes de recherche ou de production où nous souhaitons tester le modèle sur de nouvelles données ou l'intégrer dans une application sans perdre le travail effectué.

Pour charger le modèle sauvegardé, nous utilisons **torch.load(PATH, weights_only=False)**. Cela nous permet de récupérer le modèle exactement tel qu'il était au moment de l'enregistrement. En spécifiant **weights_only=False**, nous assurons que la structure du modèle et ses poids sont tous restaurés, ce qui nous permet de reprendre immédiatement l'utilisation du modèle sans avoir à le reconstruire.

10. Conclusion

En conclusion, notre projet de détection des tumeurs par l'intermédiaire d'un Réseau de Neurones Convolutif (CNN) met en lumière l'importance de l'apprentissage profond dans le domaine de la vision par ordinateur. À travers l'importation des bibliothèques nécessaires, la normalisation des données, la création d'un modèle CNN, et l'évaluation de ses performances, nous avons établi une approche systématique et rigoureuse pour classifier les images médicales.

L'entraînement du modèle, basé sur un taux d'apprentissage optimisé et une fonction de perte adaptée, a permis d'ajuster efficacement les paramètres du réseau pour améliorer sa précision. Les étapes d'enregistrement et de chargement du modèle garantissent que nos efforts sont préservés et que le modèle peut être facilement réutilisé pour des analyses futures.

Cette démarche non seulement renforce notre compréhension des techniques d'apprentissage automatique, mais souligne également leur potentiel dans des applications cliniques, offrant une promesse d'amélioration dans le diagnostic précoce des tumeurs. Les résultats obtenus pourraient ainsi servir de base à de futures recherches et développements, contribuant à l'avancement des solutions technologiques en santé.