Modélisation par apprentissage profond pour contribuer au traitement de l'imagerie médicale, application pour une classification binaire et multi-classe d'une tumeur cérébrale

Hasinavalona H. S. E.[1], Randriamitantsoa P. A.[2], Randriamitantsoa A. A[3]

Laboratoire de Recherche en Télécommunication, Automatique, Signal et Images (LR-TASI)

École Doctorale en Sciences et Techniques de l'Ingénierie et de l'Innovation (ED-STII)

Équipe d'Accueil Doctorale Télécommunication, Automatique, Signal et Images (EAD-TASI)

Université d'Antananarivo

BP 1500, Ankatso – Antananarivo 101 – Madagascar

[1]navalonahenintsoa@gmail.com, [2]rpauguste@gmail.com, [3]andriau23@gmail.com

Résumé

La vision par ordinateur est l'un de domaine qui progresse le plus rapidement grâce à l'apprentissage profond et aux nombres des données circulant aujourd'hui. Nombreux sont les domaines d'application, parmi eux, la conduite d'une voiture autonome, les classifications d'image, les reconnaissances faciales, les domaines des arts, les domaines de l'environnement et les domaines de la santé. L'algorithme en vision artificielle utilise la segmentation d'image pour connaitre chaque élément qui constitue l'image en combinant l'algorithme de classification d'image et la localisation d'objet ainsi que d'autre algorithme. Parmi les domaines mentionnés, l'imagerie médicale prend une place majeure en terme de recherche en vision artificielle. Le cerveau est l'organe principal, centre de l'activité motrice du corps humain. Très délicat et complexe, le diagnostic du cerveau fait l'objet de nombreuses recherches et études. Plusieurs méthodes telles que l'IRM, le scanner. Dans le diagnostic clinique et le traitement des tumeurs cérébrales, la lecture manuelle d'images consomme beaucoup d'énergie et de temps, l'acquisition devant être répétée autant de fois qu'il y a de coupes ce qui entraîne une fatigue du patient. L'IRM dure 30 à 1h, générant plusieurs images inutiles qui ralentit le traitement et fatigue les patients. Une solution pour contribuer à cette technique est l'utilisation d'un modèle en apprentissage profond qui entraînera plusieurs images médicales, complètera les données manquantes lors de l'IRM ou d'un scanner et testera l'image provenant d'un IRM ou d'un scanner pour aider les médecins à diagnostiquer.

L'objectif du présent travail est de créer un algorithme d'apprentissage profond robuste pour contribuer à la classification binaire et multi-classe d'une tumeur de cerveau. L'algorithme est créé dans son intégralité à partir de zéro. Toutes les techniques de régularisation pour supprimer le sur apprentissage et les techniques d'optimisation pour trouver rapidement les paramètres ont été testés et implémentés pour avoir un algorithme robuste. Pour l'ensemble de données, notre méthode peut atteindre une précision maximale, pour la validation, de 96,88% pour la classification binaire, 93,38% pour la classification multi-classe et 98,37% pour la classification binaire en utilisant la technique de l'apprentissage par transfert.

Mots clés : *réseaux de neurones, apprentissage profond, imagerie médicale, algorithme, robuste, vision par ordinateur, classification binaire, classification multi-classe, segmentation.*

Abstract

Computer vision is one of the fastest growing fields thanks to deep learning and the sheer numbers of data circulating today. There are many areas of application, including autonomous car driving, image classification, facial recognition, art, environmental and health domains. The computer vision algorithm uses image segmentation to know each element that makes up the image by combining the image classification algorithm with object localisation and other algorithms. Among the fields mentioned, medical imaging takes a major place in terms of computer vision research. The brain is the main organ, the centre of motor activity in the human body. The diagnosis of the brain is very delicate and complex and is the subject of much research and study. Several methods such as MRI, CT scan. In the clinical diagnosis and treatment of brain tumours, the manual reading of images consumes a lot of energy and time, as the acquisition has to be repeated as many times as there are slices, which leads to patient fatigue. MRI takes 30 to 1 hour, generating many unnecessary images which slows down the treatment and makes the patients tired. One solution to help with this technique is the use of a deep learning model that will train multiple medical images, fill in missing data from MRI or CT scans and test the image from an MRI or CT scan to help doctors make a diagnosis. The objective of the present work is to create a robust deep learning algorithm to assist in the binary and multi-class classification of a brain tumour. The algorithm is created in its entirety from scratch. All regularisation techniques to remove overfitting and optimisation techniques to find parameters quickly have been tested and implemented to have a

robust algorithm. For the dataset, our method can achieve a maximum validation accuracy of 96.88% for binary classification, 93.38% for multi-class classification and 98.37% for binary classification using the transfer learning technique.

Keywords : *Neural network, deep learning, medical imaging, algorithm robust, computer vision, dataset, Regularization, Batch, classification.*

1 Travaux existants

Il existe plusieurs études sur la détection des tumeurs cérébrale, utilisant l'IRM qui ont été effectuées auparavant. L'ondelette discrète Transformée (DWT), Transformée en ondelettes continue (CWT), et les méthodes SVM (Support Vector Machine) sont utilisées pour détecter les tumeurs cérébrales [1]. Cette méthode obtient des résultats suffisamment élevés pour détecter les tumeurs cérébrales ; cependant, il y a encore des faiblesses dans le calcul. [2] Même si les résultats sont assez élevés, la configuration du modèle créé n'est pas incluse et le jeu de données n'est pas expliqué. [2][3]

La classification des tumeurs cérébrales à l'aide de méthodes d'apprentissage automatique a déjà été étudiée par les chercheurs, surtout ces dernières années. Le développement de l'intelligence artificielle et des nouvelles technologies basées sur l'apprentissage profond a eu un grand impact dans le domaine de l'analyse des images médicales, notamment dans le domaine du diagnostic des maladies (Mehmood et al. 2020, 2021 ; Yaqub et al. 2020). En parallèle, de nombreuses études ont été menées sur la détection et la multi-classification des tumeurs cérébrales à l'aide de CNN. [4]

Les chercheurs suivants ont adopté des modèles CNN pré-entraînés en utilisant une approche d'apprentissage par transfert pour la classification des tumeurs cérébrales. Par exemple, Çinar et Yildirim (2020) ont utilisé une forme modifiée du modèle CNN ResNet-50 pré-entraîné en remplaçant ses 5 dernières couches par 8 nouvelles couches pour la détection de tumeurs cérébrales. Ils ont obtenu une précision de 97,2 % en utilisant des images IRM avec ce modèle CNN modifié. [4]

S. Kevin Zhou, Hayit Greenspan, Dinggang Shen ont écrit des livres concernant l'apprentissage profond pour l'analyse des images médicales [5]. À part plusieurs articles écrits par des chercheurs comme celui de Ali ARI, Davut Hanbay [6], Justin Paul [7], plusieurs compétitions ont été réalisés afin d'avoir le maximum de

précision pour une quantité de données limitées.

2 Algorithme et implémentation

2.1 Principes de l'apprentissage profond

Définition 01 :

L'apprentissage machine est un domaine de l'intelligence artificielle qui consiste à programmer une machine pour que celle-ci apprenne à réaliser des tâches en étudiant des exemples de ces dernières. Ces exemples sont représentés par des données que la machine utilise pour développer un modèle. [8] Un modèle comme une fonction du type

$$f(x) = wx + b \qquad (01)$$

L'objectif est de trouver les paramètres $\boldsymbol{w}$ et $\boldsymbol{b}$ qui donnent le meilleur modèle possible, c'est-à-dire le modèle qui s'ajuste le mieux à nos données, pour cela, on programme dans le modèle, un algorithme d'optimisation qui va venir tester différentes valeurs de w et b jusqu'à obtenir la combinaison qui minimise la distance entre le modèle et les points (ou qui minimise les erreurs entre la sortie du modèle et la réponse attendue).

Définition 022 :

L'apprentissage profond est un domaine de l'apprentissage machine dans lequel, au lieu de développer un des modèles de l'apprentissage machine, alors on développe à la place des réseaux de neurones artificiels.

Le réseau de neurones reçoit des paramètres de données en entrée, traite ces données avec des autres paramètres par des calculs complexes et retourne des résultats, parfois une prédiction. Un réseau de neurones est donc une succession de calcul plus ou moins complexe pour apprendre une sortie correcte.

2.1.1 Fonction d'activation

Dans le domaine des réseaux de neurones artificiels, la fonction d'activation est une fonction mathématique appliquée à un signal en sortie d'un neurone artificiel. Le terme de "fonction d'activation" vient de l'équivalent biologique "potentiel d'activation", seuil de stimulation qui, une fois atteint entraîne une réponse du neurone. [9]

Quand on construit un modèle d'apprentissage profond, l'un des choix à faire est le choix de la fonction d'activation, utilisée dans les couches cachées. Un choix le plus populaire en apprentissage profond est la fonction ReLU

$$a(z) = \max(0, z) \qquad (02)$$

En utilisant la fonction d'activation ReLU, le réseau de neurones apprendra souvent

beaucoup plus rapidement, elle remplace donc toutes les valeurs négatives reçues en entrées par des zéros.

Un désavantage de la fonction ReLU est que sa dérivée devient nulle lorsque l'entrée est négative ce qui peut empêcher la rétropropagation du gradient. On peut alors introduire une version appelée Leaky ReLU définie par :

$$a(z) = \max(\epsilon z, z) \; pour \; tout \; réel \; z \qquad (03)$$

Le paramètre ϵ est un réel strictement positif et inférieur à 1

$$\epsilon \in]0,1[$$

La dérivée est alors égale à ϵ lorsque x est strictement négatif, ce qui permet de conserver la mise à jour des poids d'un perceptron utilisant cette fonction d'activation. Pour une classification binaire, on va utiliser des fonctions redresseuses comme activation, c'est-à-dire des fonctions ReLU, et la fonction sigmoïde en sortie.

2.1.2 Régression logistique

Il s'agit d'un algorithme d'apprentissage que l'on utilise lorsque les étiquettes de sorties y dans un problème d'apprentissage supervisé soit $\{0,1\}$

En régression logistique, on peut utiliser une fonction sigmoïde, qui résout la sortie entre $\{0,1\}$

Car une fonction sigmoïde est représentée comme suit :

$$\sigma(z) = \frac{1}{1+e^{-z}} \qquad (04)$$

Si

$$\begin{Bmatrix} z \; plus \; grand \; , \sigma \; tend \; vers \; 1 \\ z \; plus \; petit, \sigma \; tend \; vers \; 0 \end{Bmatrix}$$

Pour une représentation plus simple, ci-après un modèle avec une couche d'entrée, une couche cachée et une couche de sortie :

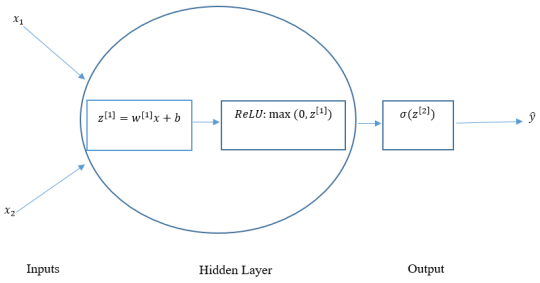


Figure 01: *Aperçu d'un modèle de réseaux avec une couche connectée et une couche de sortie*

2.1.3 La fonction perte

Pour évaluer la qualité d'un modèle de réseau de neurones, c'est-à-dire trouver les valeurs $\boldsymbol{w}$ et $\boldsymbol{b}$ qui correspondent à notre modèle minimisant la fonction cout.

L'objectif est de faire comprendre au modèle que sa prédiction est satisfaisante ou non. [15]

Il y a plusieurs fonction coût comme :

- L'erreur quadratique

$$\mathcal{L}(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2 \quad (05)$$

Il s'avère qu'on peut utiliser cette fonction, mais en régression logistique, on n'utilise pas cette méthode car lorsqu'on doit apprendre les paramètres, on trouvera que le problème d'optimisation devient non convexe, et la descente de gradient peut ne pas trouver l'optimum global. Donc ce qu'on va utiliser dans la régression linéaire est en fait la fonction de perte cross entropy. [15]

- Cross entropy

$$\mathcal{L}(\hat{y}, y) = -(y log\hat{y} + (1 - y)\log(1 - \hat{y})) \quad (06)$$

Notons que l'objectif est que cette fonction de perte soit la plus petite possible

Si $y = 1$ $\mathcal{L}(\hat{y}, y) = -y log\hat{y}$

Si $y = 0$ $\mathcal{L}(\hat{y}, y) = -\log(1 - \hat{y})$

La fonction de perte est définie pour un seul exemple d'apprentissage $(\boldsymbol{x^1}, \boldsymbol{y^1})$, donc il mesure la performance sur un seul exemple d'apprentissage.

2.1.4 La fonction de coût

La fonction de cout est la somme de la fonction perte qui mesure la performance sur les jeux de données en entier $(x^1, x^1), \ldots\ldots, (x^m, y^m)$, cette fonction est définie par :

$$J(\omega, b) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) \quad (07)$$

$$J(\omega, b) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)} log\hat{y}^{(i)} + (1 - y^{(i)})\log(1 - \hat{y}^{(i)})] \quad (08)$$

On peut trouver les valeurs de $\boldsymbol{w}$ et $\boldsymbol{b}$

$$\omega = \omega - \alpha\frac{dJ(\omega, b)}{d\omega} \quad (09)$$

$$b = b - \alpha\frac{dJ(\omega, b)}{db} \quad (10)$$

α est le taux d'apprentissage

2.1.5 Etape à suivre pour développer et entrainer un modèle des réseaux de neurones

Pour développer et entrainer des réseaux de neurones artificiels, alors on répète les étapes suivantes jusqu'à trouver la valeur de $\boldsymbol{w}$ et $\boldsymbol{b}$ qui minimise la fonction coût :

- Forward propagation : on fait circuler les données de la première

couche jusqu'à la dernière afin de produire une sortie $\hat{y}$

- Calculer la fonction coût : on calcule l'erreur entre cette sortie $\hat{y}$ et la sortie de référence y que l'on désire avoir
- Backward propagation : on mesure comment cette fonction coût varie par rapport à chaque couche de notre modèle, on applique l'algorithme de la descente de gradient afin de corriger chaque paramètre du modèle.

2.2 Les réseaux de neurones à convolution

Définition 033 :

Les réseaux de neurones à convolution désignent une sous-catégorie de réseaux de neurones : ils présentent donc toutes les caractéristiques et les différentes méthodologies listées ci-dessus. Cependant, les CNN sont spécialement conçus pour traiter des images en entrée. [10] L'architecture d'un réseau de neurones à convolution est formée par une succession de blocs de traitement pour extraire les caractéristiques discriminant la classe d'appartenance de l'image des autres. Un bloc de traitement se compose d'une à plusieurs :

- Couches de convolution (CONV) qui traitent les données d'un champ récepteur
- Couches de correction (ReLU)
- Couches de pooling (POOL), qui permet de compresser l'information en réduisant la taille de l'image intermédiaire (souvent par sous-échantillonnage).

Les blocs de traitement s'enchaînent jusqu'aux couches finales du réseau qui réalisent la classification de l'image et le calcul de l'erreur entre la prédiction et la valeur cible :

- Couche « entièrement connectée » (FC), qui est une couche de type perceptron ;
- Couche de perte (LOSS)

Donc pour notre couche de convolution $[l]$, les différents paramètres et hyper-paramètres sont :

- Pour l'entrée, la taille est :

$$n_H^{l-1} \times n_W^{l-1} \times n_c^{l-1} \qquad (11)$$

- La taille du filtre f est : $f^{[l]}$
- Le padding est : $p^{[l]}$
- Le stride est : $s^{[l]}$
- Pour la sortie la taille est :

$$n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]} \quad (12)$$

Où

$$n_H^{[l]} = \frac{n_H^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \quad (13)$$

Et

$$n_w^{[l]} = \frac{n_w^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \quad (14)$$

- Nombre de filtre :$n_c^{[l]}$
- Pour chaque filtre, la taille est : $f^{[l]} \times f^{[l]} \times n_c^{[l-1]}$
- Activation : $a^{[l]}$: $n_H^{[l]} \times n_w^{[l]} \times n_c^{[l]}$
- Weights : $f^{[l]} \times f^{[l]} \times n_c^{[l-1]} \times n_c^{[l]}$
- Bias : $n_c^{[l]}$

Illustrons à l'aide de la figure suivante la structure d'une couche de convolution

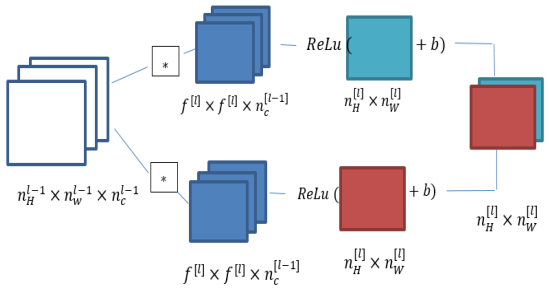


Figure 02: *Structure d'une couche de convolution*

3 Réalisation du modèle

Le choix des hyper-paramètres dans un apprentissage profond est une étape très importante pour optimiser l'algorithme et rendre robuste. Ces choix résident sur :

- Les nombres des couches
- Les nombres des filtres
- Les choix sur les tailles des filtres
- Quelle fonction d'activation à utiliser pour les différentes couches
- Les nombres de couches connectées
- « L'optimizer » à utiliser et les hyper-paramètres des optimiseurs comme β_1, β_2
- Le taux d'apprentissage

3.1 Les jeux des données

3.1.1 Structure des données

Les données d'entrainement et de validation ont été téléchargées sur le site KAGGLE [11]. Les données sont classées en deux parties : une partie contenant les tumeurs et une partie sans une présence de tumeur. Les images ressemblent à

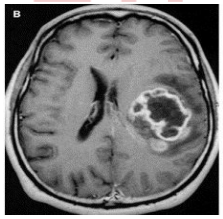


Figure 03 *: Image en présence d'une tumeur*

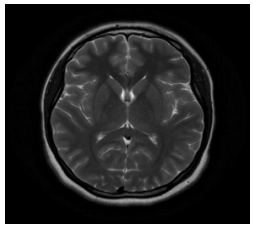
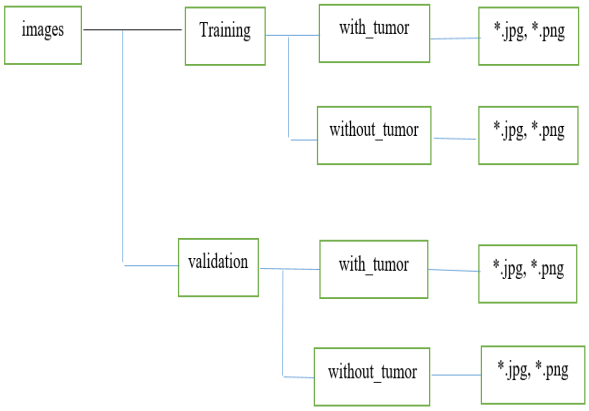


Figure 04 : *Image d'un cerveau normal*

Ci – après la structure des données sur notre machine



Nous avons 3000 images appartenant aux deux classes.

Comme nos données sont limitées, nous avons décidé de diviser les données d'entrainement et les données de validation en 80% et 20%

Tableau 1 : Division des données

Données d'entrainement :80%	Données de validation : 20%

3.1.2 Augmentation des données

Une insuffisance de données risque le non-apprentissage au niveau du modèle, car le modèle n'arrive même pas à apprendre des données. Cette insuffisance provoque aussi un surentrainement du réseau, car le modèle s'entraine sur le peu de données d'entrainement et a une bonne précision au niveau de l'entrainement, mais n'arrive pas à reconnaitre une nouvelle image qu'il n'a pas étudiée.

Une solution pour résoudre à cela est l'augmentation d'image. Les différentes techniques sont : Symétrie axiale, Rotation, Recadrage aléatoire, Changement de couleur, Addition de bruit, Zoom, Changement de contraste [12].

Ci-après une représentation d'une image après une augmentation :

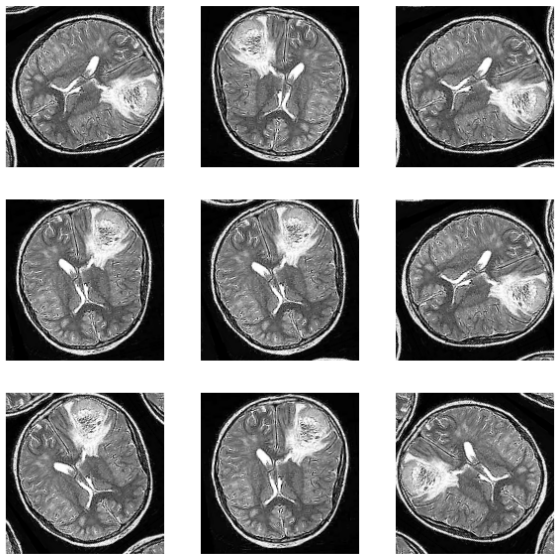


Figure 05 : *Une image augmentée*

3.2 Choix sur les nombres et les tailles des filtres

Lors de la conception de l'architecture d'un modèle de réseaux de neurones, l'efficacité d'un modèle réside sur le choix des hyper-paramètres. Parmi ces hyper-paramètres, il y a les choix sur les filtres à utiliser comme, comment choisir entre les filtres de taille 1×1 ou 3×3 ou 5×5 et bien d'autres. Le

réseau de démarrage ou inception network permet donc d'utiliser tous ces choix, cela rend l'architecture compliquée, mais elle fonctionne remarquablement bien.

Introduit par Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke et Andrew Rabinovich dans leur article « Going deeper with convolutions », l'idée de base est que au lieu de devoir choisir la taille des filtres ou de la couche pooling, alors on peut tester toutes les options puis concaténer les sorties. Ci-après le graphe de calcul pour une entrée de taille, prenons comme exemple 28×28

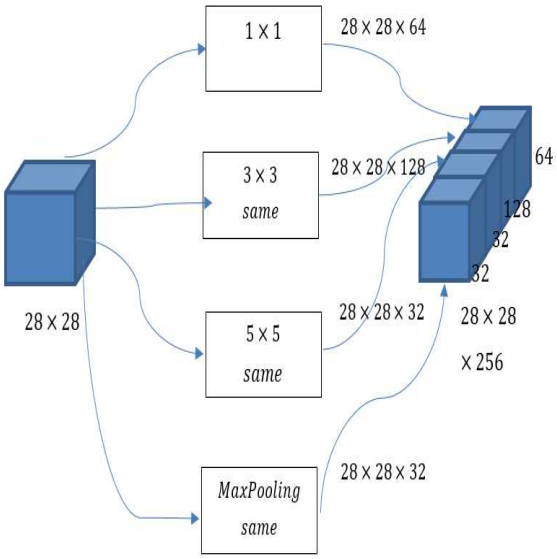


Figure 06 : *Modèle de réseau à inception*

L'utilisation du module d'inception rend l'architecture du réseau plus compliqué, rend le calcul très lent et accapare les ressources.

3.3 Amélioration de l'apprentissage par les techniques de normalisation et de régularisation

Nous avons utilisé les différentes régularisations et normalisation suivantes

La normalisation de lot : La normalisation de lot ou « batch normalization » est une étape qui normalise le lot $\{x_i\}$, avec un choix de paramètres γ, β. En notant μ_B, σ_B^2, la moyenne et la variance de ce qu'on veut corriger du lot.[12]

$$x_i \leftarrow \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta \qquad (15)$$

Dropout : Souvent dans une couche connectée, les nœuds ont des mêmes poids. Le dropout est une technique qui est destinée à empêcher le sur-ajustement sur les données de training en abandonnant des unités dans un réseau de neurones avec une probabilité $p > 0$. Cela force le modèle à éviter de trop s'appuyer sur un ensemble particulier des caractéristiques. [12]

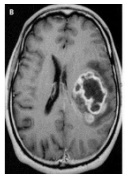
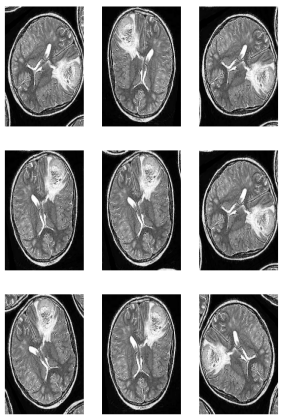
Régularisation de coefficient : Pour s'assurer que les coefficients ne sont pas trop grands et que le modèle ne sur-ajuste pas sur le training set, on utilise des techniques de régularisation sur les coefficients du modèle. [12], Nous avons utilisé une régularisation L2

Arrêt prématuré : L'arrêt prématuré ou early stopping est une technique de régularisation

qui consiste à stopper l'étape d'entraînement dès que le loss de validation atteint un plateau ou commence à augmenter. [12]

3.4 Structure de notre modèle de classification binaire

Tableau 2 : Structure de notre modèle de classification binaire

Image (150,150,3)	
Augmentation des données	
Couche de convolution N°01	Nombre des filtres=64, taille=(5×5), $padding = same$
BatchNormalisation	$BatchNormalization$
Activation	$ReLU$
MaxPooling	Taille (2×2)
Couche convolution N°02	Nombre des filtres=64, taille=(5×5), $padding = same$
BatchNormalisation	$BatchNormalization$
Activation	$ReLU$
MaxPooling	Taille (2×2)
Couche convolution N°03	Nombre des filtres =64, taille=(5×5), $padding = same$
BatchNormalisation	$BatchNormalization$
Activation	$ReLU$
MaxPooling	Taille (2×2)
Couche de convolution N°04	Nombre des filtres=128, taille=(5×5), $padding = same$
BatchNormalisation	$BatchNormalization$
Activation	$ReLU$

MaxPooling	Taille (2×2)

Couche de convolution N°05	Nombre des filtres =256, taille=(3×3), $padding = same$
BatchNormalisation	$BatchNormalisation$
Activation	$ReLU$
MaxPooling	Taille (2×2)

Flatten	$Flatten()$
Régularisation	$Dropout(0.35)$
Couche connectée	$Nombre = 512$
Régularisation	$L2 : 0.01$
Couche de sortie	Nombre de classe = 1, activation=$sigmoid$

4 Résultats et discussions

Pour cette simulation, nous avons utilisé l'algorithme du gradient stochastique qui est donc une méthode de descente de gradient itérative optimisée. Cette méthode est la méthode de SGD avec moment qui conserve en mémoire la mise-à-jour à chaque étape de Δw, et calcule la suivante comme une combinaison convexe du gradient actuel et de la modification précédente. Si on note w_n les coefficients obtenus après n itérations, ainsi que Δw_n la $n - ieme$ mise à jour des paramètres, alors

$$\Delta w_{n+1} := \eta \nabla Q_i(w) + \alpha \Delta w_n \qquad (16)$$

$$w_{n+1} := w_n - \Delta w_{n+1} \qquad (17)$$

Notre hyper paramètre taux d'apprentissage $\alpha = 0.01$

Le nombre d'époque est égale à 35, on peut utiliser aussi une méthode de pré-arrêt lorsqu'on atteint un certain seuil de précision pour l'entrainement et la validation.

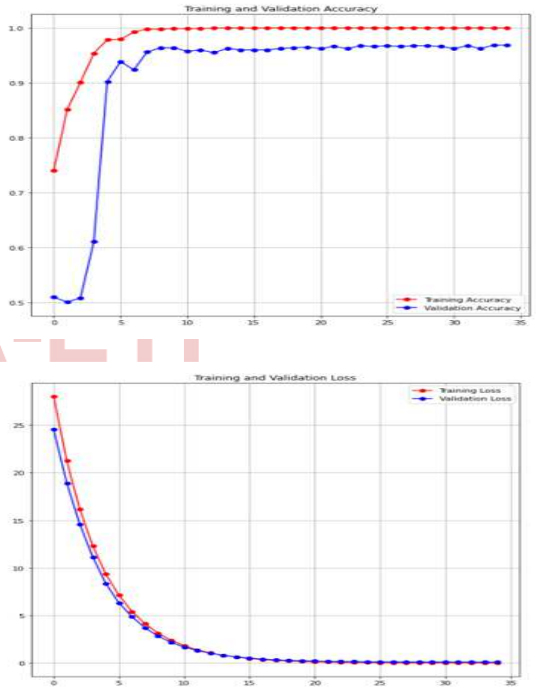


Figure 07 : Résultat de notre modèle de classification binaire, à 35 époque et en utilisant le SGD comme optimizer

La première constatation est que le modèle arrive à s'entrainer correctement, il n'y a pas vraiment d'incohérence au niveau de l'entrainement et au niveau de la validation.

L'écart entre la validation et l'entrainement entre l'époque 0 et 2 est normal, car le modèle commence à apprendre les paramètres, et à partir de la septième époque la précision ne cesse pas de s'accroitre pour l'entrainement et la validation.

La précision pour l'entrainement tend vers **1,0**, en terme de probabilité, et atteint la valeur de **1,0** à partir de la huitième époque, et la précision maximale pour la validation atteint **0,9688.** Une perte ou une explosion de gradient n'est pas constatée, et le modèle n'est pas compliqué, ce qui rend l'apprentissage plus vite.

La bonne configuration de la taille de lots (batch size) nous a supprimé la fluctuation au niveau de notre résultat.

En résumé, le modèle arrive à bien identifier les images de cerveau contenant une tumeur et les images normales d'un cerveau et possède une meilleure précision, sans fluctuation et rapide à apprendre.

5 Multi-class classification

Notre modèle de classification permet de détecter s'il y a une présence ou non d'une tumeur cérébrale dans une image IRM donnée, et compte tenu du résultat précèdent, le modèle arrive à identifier les images avec une meilleure précision. Cette classification est binaire seulement, mais si on veut interpréter les images comme quel type de maladie ou de cancer s'agit-il, alors notre modèle est limité. La solution est d'utiliser une classification multi-class.

Les données ont été téléchargées sur le site KAGGLE [11], et classifiées en données d'entrainement et données de validation, puis subdivisées en quatre type :

Tableau 1: Nombre d'image et structure de données pour la classification multi-class

Type	Nombre de données d'entrainement	Nombre de données de validation
glioma_tumor	826	102
meningioma_tumor	822	111
pituitary_tumor	827	156
no_tumor	395	98

Ci-après une structure globale de notre modèle en utilisant la fonction softmax, avec quatre classes, comme fonction d'activation à la sortie de notre modèle :

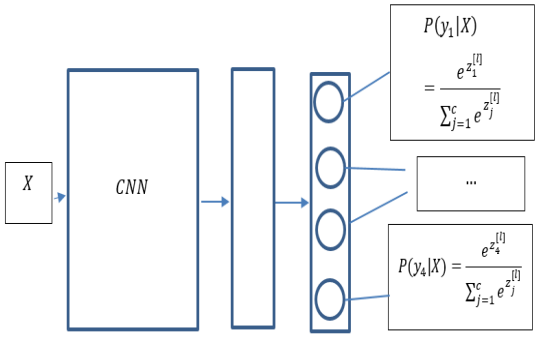


Figure 08 : Structure globale de notre modèle avec la fonction softmax comme fonction d'activation à la sortie

Ci-après le résultat de la simulation

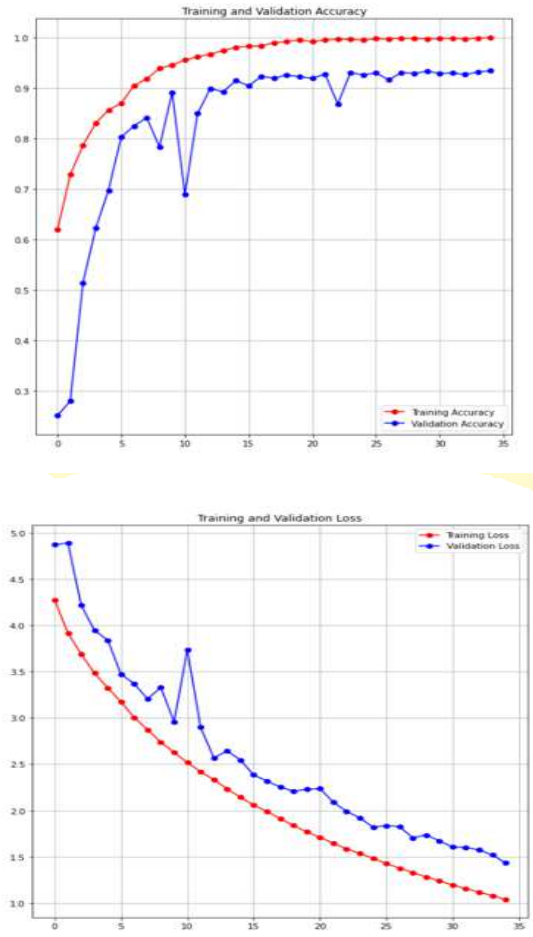


Figure 09 : Résultat de notre modèle de classification multi-classe pour 35 époques et en utilisant la descente de gradient stochastique comme algorithme de gradient optimisé

La première constatation est que le modèle arrive à s'entrainer correctement, il n'y a pas vraiment d'incohérence au niveau de l'entrainement et au niveau de la validation.

La précision pour l'entrainement tend vers **1,0** et atteint la valeur de **1,0** à partir de la dix-huitième époque, et la précision pour la validation atteint **0,9338** à la 35 -ème époque**.**

La précision maximale pour la validation est de 0,9338. Compte tenu de ces résultats, le modèle possède une meilleure précision.

Une perte ou une explosion de gradient n'est pas constatée, et le modèle n'est pas compliqué, ce qui rend l'apprentissage plus vite.

La bonne configuration de la taille de lots (batch size) nous a supprimé la fluctuation au niveau de notre résultat.

En résumé, le modèle arrive à bien identifier les images et capable de classer chaque image dans leurs types correspondants.

Une augmentation des jeux de données peut encore améliorer ce modèle pour avoir plus de précision sur la prédiction des sorties.

6 Apprentissage par transfert

Nous avons créé notre modèle à partir de zéro, en choisissant les différents paramètres et hyper-paramètres, amélioré notre modèle grâce aux différentes techniques pour rendre robuste notre

algorithme. Nous avons 3000 images pour les 2 classes, et notre modèle arrive à s'entrainer correctement, possède une meilleure précision. Cependant, une insuffisance de données peut provoquer un surapprentissage du modèle. La raison est que les données d'entrainement sont très petites et il y a une insuffisance de fonctionnalités communes qui peuvent être extraites même si nous avons fait une augmentation de données.

La solution est de prendre un modèle existant, déjà pré-entrainé et formé sur beaucoup plus de données, c'est le concept de l'apprentissage par transfert.

Nous avons pris le modèle « Inception V3 » [13] pour tester avec nos données, ceci a été préformé sur un jeu de données d'imageNet qui contient 1,4 million d'image dans 1000 classes différentes [14]. Nous avons utilisé seulement quelque couche du modèle, puis modifié la couche de sortie et ajouté plusieurs régularisations pour réduire les surapprentissages.

Voici le résultat tester sur nos données sur 35 époques

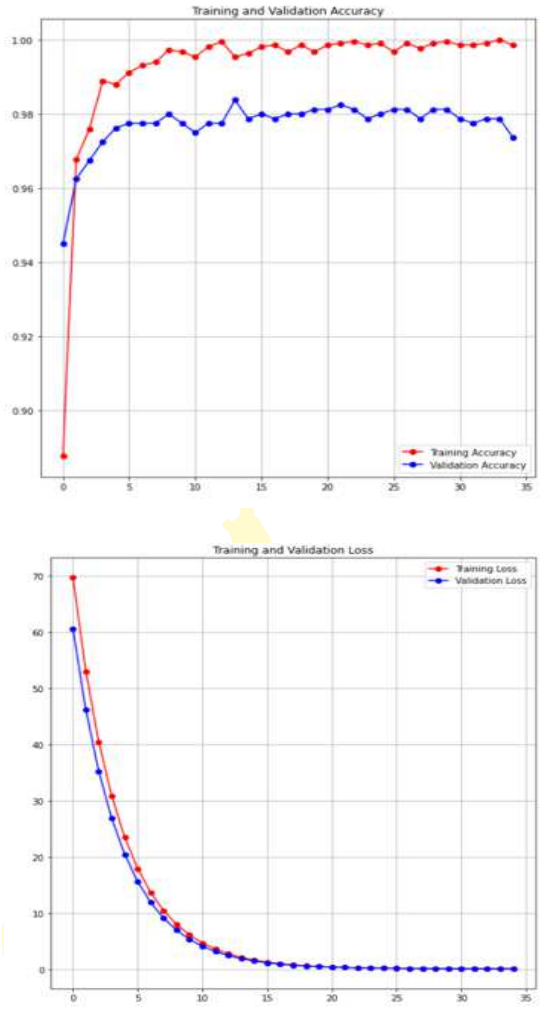


Figure 10 : *Résultat de l'apprentissage par transfert du modèle Inception en utilisant nos jeux de données.*

La première constatation est que en utilisant nos jeux de données, la technique de l'apprentissage par transfert, présente un meilleur résultat.

La précision pour l'entrainement tend vers **1** et atteint la valeur de **1,0** à la 34 eme époque, et les valeurs de la précision pour la validation tournent autour de **0,96** à **0,9999**.

La précision pour la validation présente un meilleur résultat, les valeurs tournent autour de **0,98**.

La précision maximale pour la validation est de 0,9825. Compte tenu de ces résultats, le modèle possède une meilleure précision.

En résumé, le modèle arrive à bien identifier les images de cerveau contenant une tumeur et les images normales d'un cerveau et possède une meilleure précision, sans fluctuation.

7 Conclusion

Ce travail nous a permis de créer à partir de zéro notre modèle de classification binaire et multi-classe en utilisant les réseaux de neurones avec une technique d'apprentissage profond. Notre modèle de classification binaire présente une meilleure précision en termes d'entrainement et de validation, ces valeurs tendent vers 1 à une certaine époque. Une perte ou une explosion de gradient n'est pas constatée, et le modèle n'est pas compliqué. Notre modèle de classification multi-class présente une meilleure précision au niveau de l'entrainement et au niveau de la validation. Mais la précision pour la validation peut être encore améliorée avec une augmentation de données. En apprentissage profond, il n'est pas toujours possible de généraliser certaines règles d'optimisation ou des choix sur les paramètres et hyper-paramètres. Dans ce travail, nous avons utilisé et élaboré les différentes techniques d'optimisation et de régularisation comme la normalisation de lot, la normalisation des entrées, le dropout, la régularisation de coefficient, l'arrêt prématuré et la technique d'augmentation de données pour rendre notre modèle optimal et robuste. Nous avons utilisé plusieurs combinaisons pour choisir les nombres des couches, les nombres et les tailles des filtres. Nous avons utilisé aussi l'apprentissage par transfert avec nos données, et avons testé la précision de ce dernier. Pour conclure, notre méthode peut atteindre une précision maximale de 96,88% pour la classification binaire, 93,38% pour la classification multi-classes et 98,37% pour la classification binaire en utilisant la technique d'apprentissage par transfert.

8 Bibliographie

[1] M. Gurbină, M. Lascu, and D. Lascu, *« Tumor Detection and Classification of MRI Brain Image Using Different Wavelet Transforms and Support Vector Machines, »* 42nd International Conference on Telecommunications and Signal Processing (TSP), 2019, pp. 505–508, Jul 2019.

[2] Krisna Nuresa Qodri, Indah Soesanti, Hanung Adi Nugroho, *« Image Analysis for MRI-Based Brain Tumor Classification Using Deep Learning »,* Mars 2021.

[3] S.K. Chandra, « *Effective Algorithm For Benign Brain Tumor Detection Using Fractional Calculus,* » TENCON 2018 -

2018 IEEE Region 10 Conference, 2018, pp. 2408–2413, Fev 2019.

[4] Emrah Irmak, « *Multi-Classification of Brain Tumor MRI Images Using Deep Convolutional Neural Network with Fully Optimized Framework* », Avril 2021.

[5] Kevin Zhou, Hayit Greenspan, Dinggang Shen, *« Deep learning for Medical Image Analysis »*, Janv 2017.

[9] *« Fonction d'activation », https://fr.wikipedia.org/wiki/Fonction_d%27activation*, 2021.

[10] Pascal Monasse, Kimia Nadjahi *« Classez et segmentez des données visuelles », https://openclassrooms.com/fr/courses/4470531-classez-et-segmentez-des-donnees-visuelles/5082166-quest-ce-quun-reseau-de-neurones-convolutif-ou-cnn*, 2021.

[11] Ahmed Hamada, *« Brain Tumor Detection », https://www.kaggle.com/ahmedhamada0/brain-tumor-detection,* 2021.

[12] Shervine Amidi, Afshine Amidi, *« Réseaux convolutionnels », https://stanford.edu/~shervine/l/fr/teaching/cs-230/pense-bete-reseaux-neurones-convolutionnels,*2021.

[6] Ali ARI, Davut HANBAY, « Deep learning based brain tumor classification and detection system », Sept 2018.

[7] Justin Paul, thesis, *« Deep Learning for Brain Tumor Classification »,* Mai 2016.

[8] Guillaume Saint-Cirgue, *« Formation deep learning complète », https://www.youtube.com/watch?v=XUFLq6dKQok,* Machine learnia, 2021.

[13] Christian Szegedy, Vincent Vanhoucke, Serge loffe, Jonahtan Shiens, Zhigniew Wojna, *« Rethinking the inception Architecture for computer vision »,* Dec 2015.

[14] Laurence Moroney, *« Convolutional Neural Networks in TensorFlow », https://www.coursera.org/learn/convolutional-neural-networks-tensorflow/home/welcome,* DeepLearning.AI, 2021.

[15] Andrew Ng, *« Convolutional Neural Networks », https://www.coursera.org/learn/convolutional-neural-networks/home/welcome,* DeepLearning.AI, 2021.