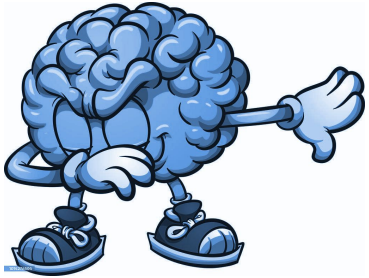
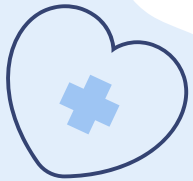
DS50 : Prédication tumeur cérébrale



Osman Gaygusuz, Gaspard Rochu, Eléanore Renaud, Théo Gouin, Elise Albrecht, Jérémie Kimenau



Sommaire

01 Contexte

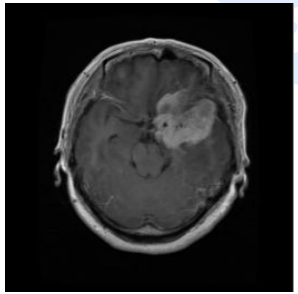
- Les tumeurs cérébrales sont de tailles et localisations variées :
  - Rendent les diagnostics complexe
  - Nécessitent l'expertise de neurochirurgiens

- Système automatisé permet:
  - Accélérer et fiabiliser l'analyse des tumeurs
  - Palier un manque de spécialistes et de connaissances

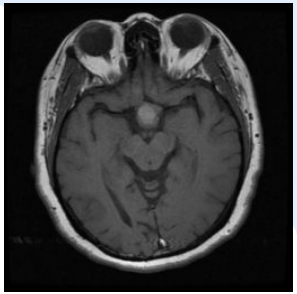
02 Traitement des données

Le dataset

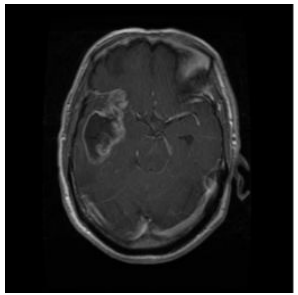
- Dataset de classification
- 3 tumeurs à identifier :
  - Gliome
  - Meningiome
  - Tumeur hypophysaire (*Pituitary)*
  - Pas de tumeur
- 2700 images d'entrainement
- 400 images de test



Meningiome



Pituitaire



Gliome

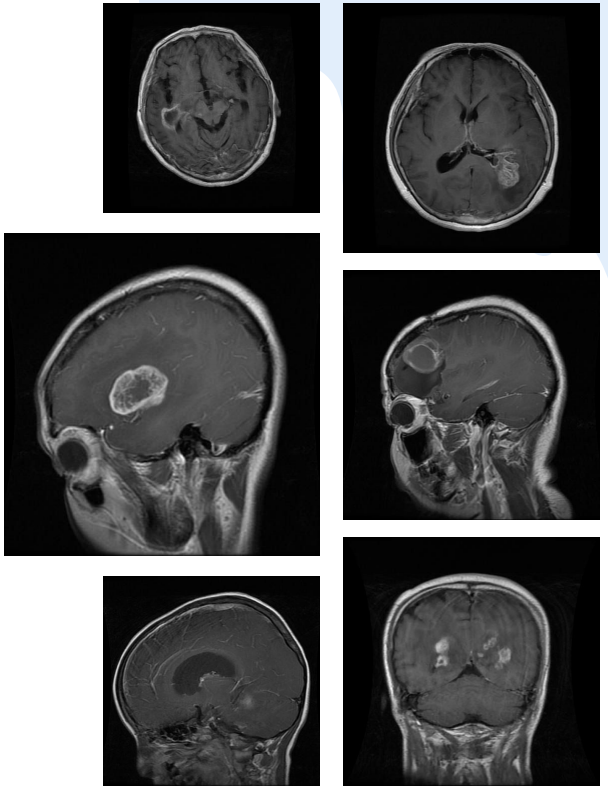
Pas de tumeur

02 Traitement des données

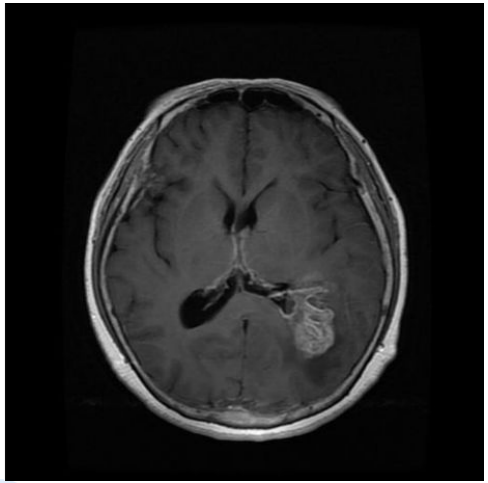
Les images

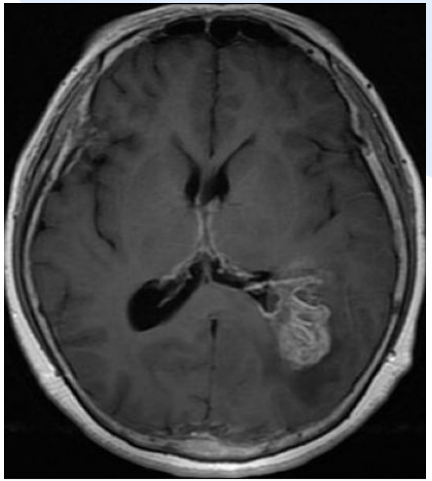
Images :

- Vue du dessus
- Vue de côté
- Vue de derrière…
- Différentes tailles
- Différentes proportions
- Différentes orientations



02 Traitement des données





Recadrage des images afin de n’avoir que la partie intéressante pour le modèle

02 Traitement des données

495x619

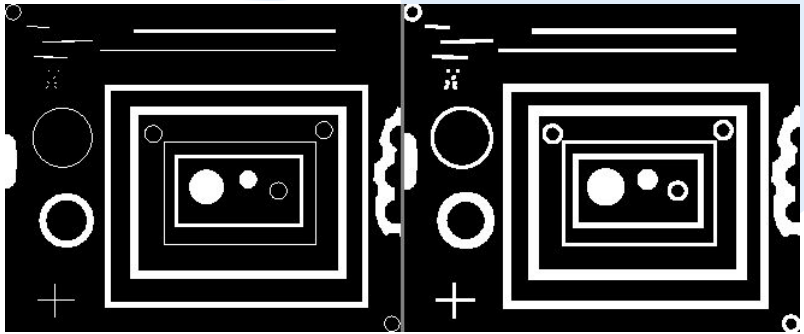
224x224 (DenseNet/ResNet)
240x240 (EfficientNet)

Redimensionnement des images pour les adapter aux entrées des modèles

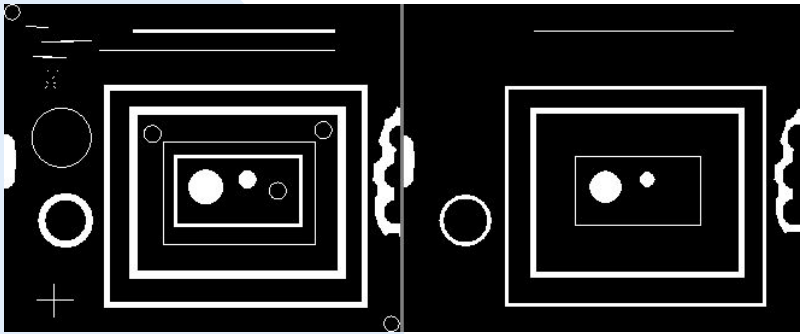
02 Traitement des données

Dilatation des images:

- Objectif: Remplir les potentiels petits trous dans l'image
- Conséquences: Effet de grossissement des traits



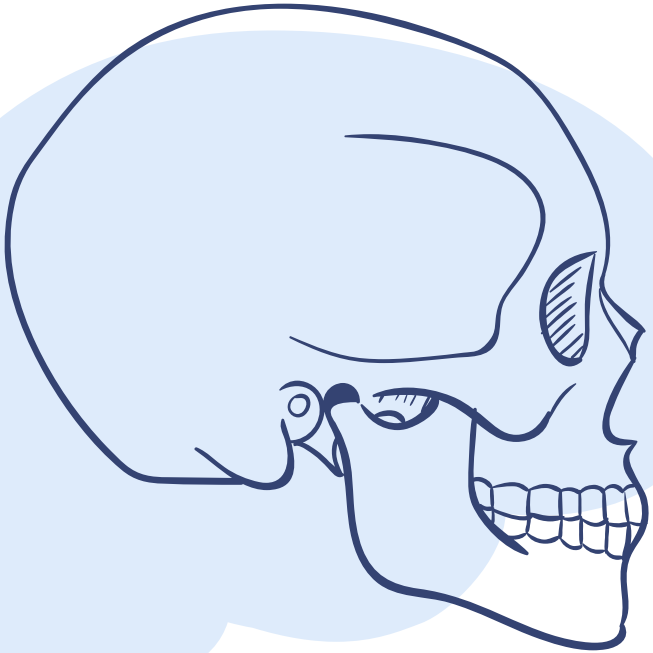
Source: https://www.mathworks.com/



Source: https://www.mathworks.com/

Erosion des images:

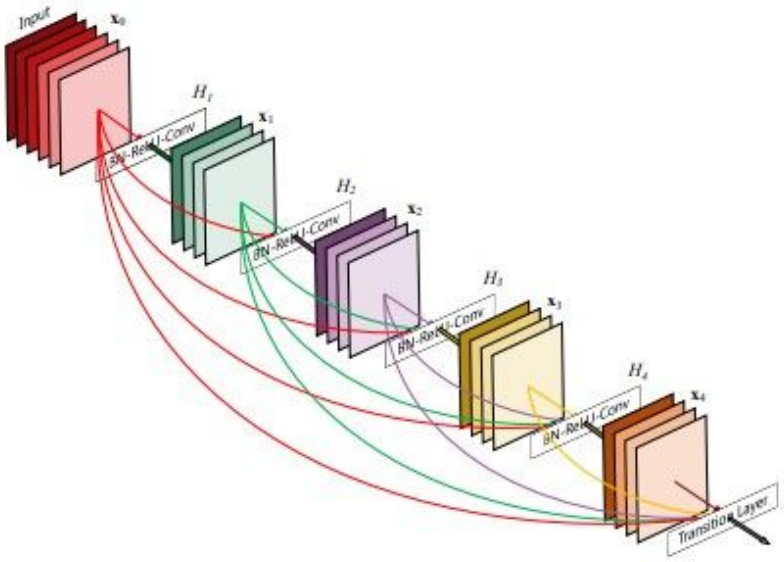
- Objectif: Supprimer les potentiels pixels flottants
- Conséquences: Effet d'amincissement des traits



03
Modèles utilisés

03 Modèles utilisés

Densenet



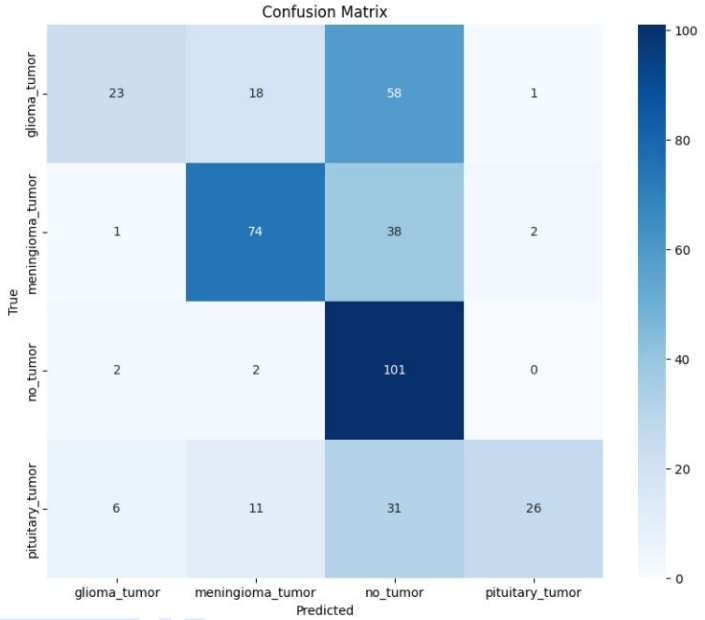
- Architecture utilisant des couches denses → **concaténation par canal**

- Permet de **réduire considérablement le nombre de paramètres** à déterminer → entrainement plus rapide

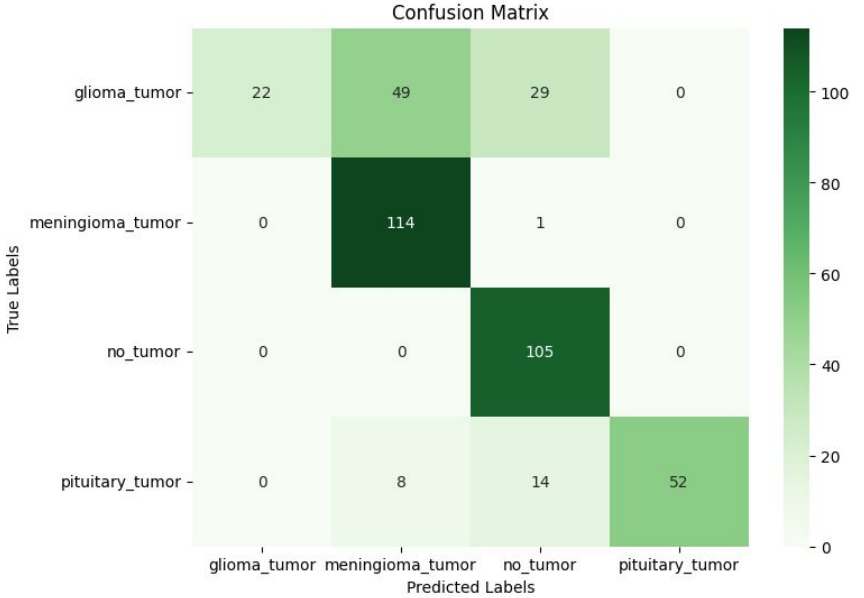
3 entrainement différents:
- modèle implémenté à la main
- modèle implémenté par Keras
- entrainement avec FastAI

03 Modèles utilisés

Resultats



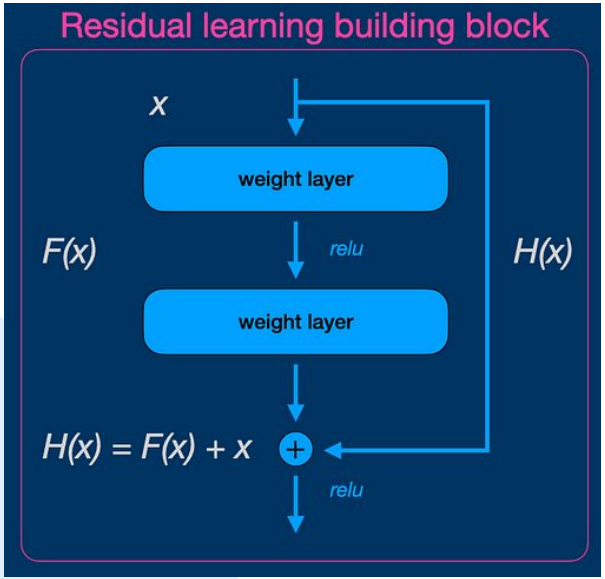
Modèle implémenté à la main

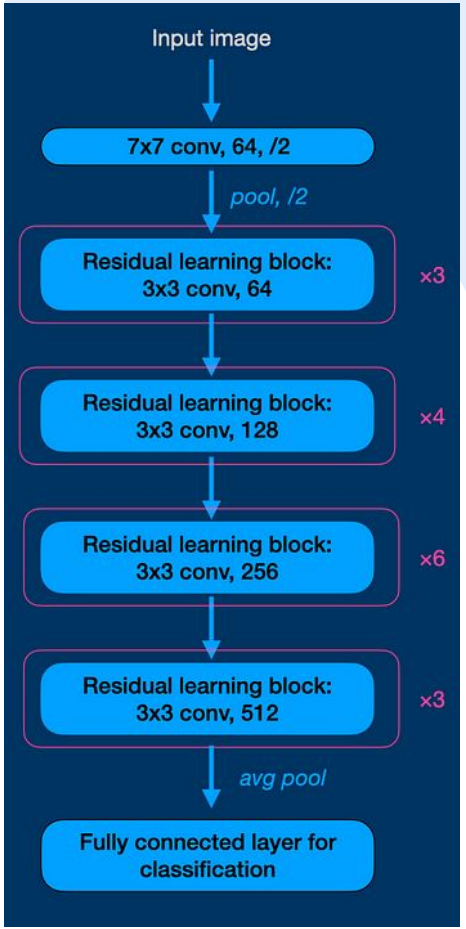


Modèle optimisé par FastAI

03 Modèles utilisés

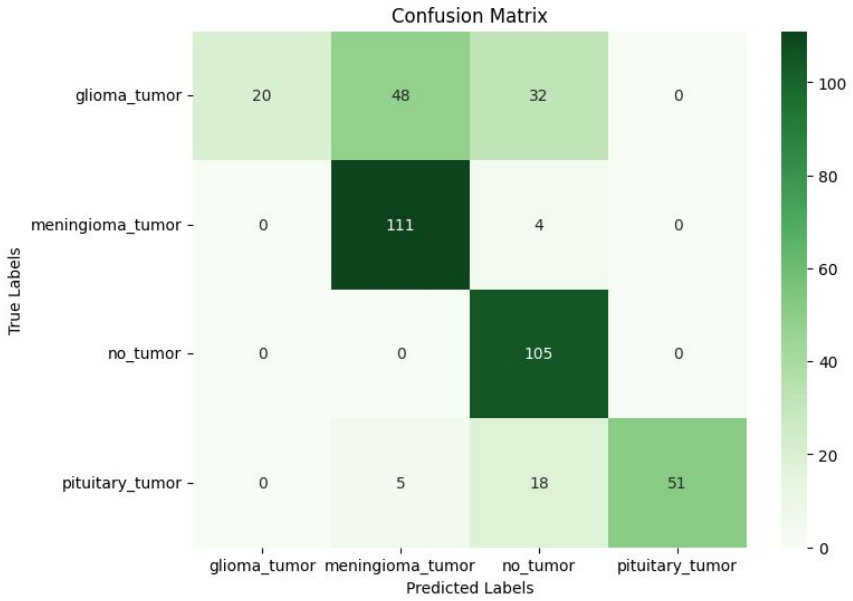
ResNet

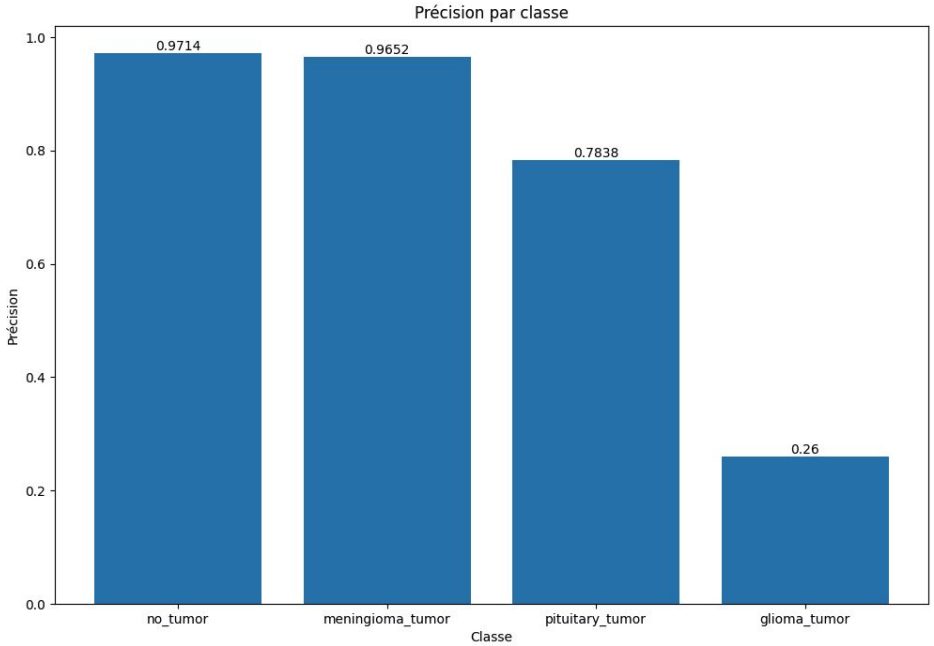




03 Modèles utilisés

ResNet



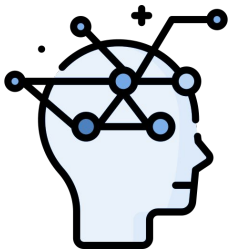


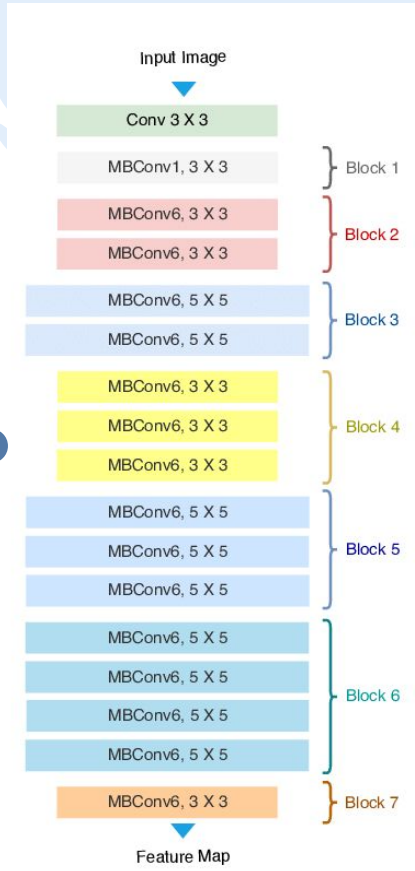
03 Modèles utilisés :

EfficientNetB1

Architecture CNN développé par Google AI : B0 à B7

Compound scaling : optimise les performances en équilibrant profondeur, la largeur et la résolution



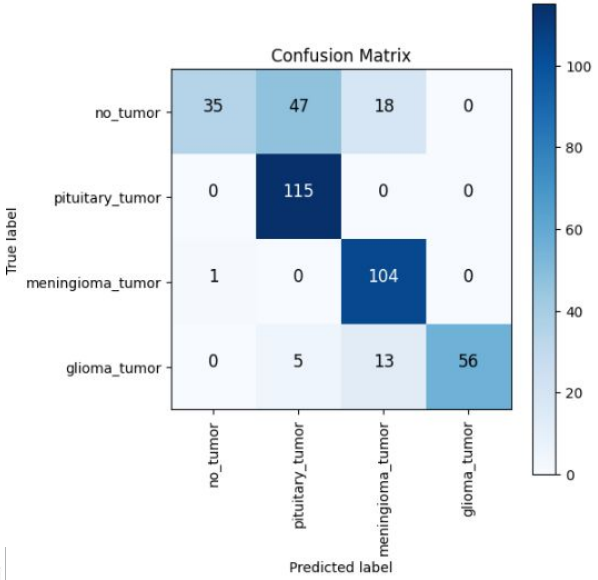


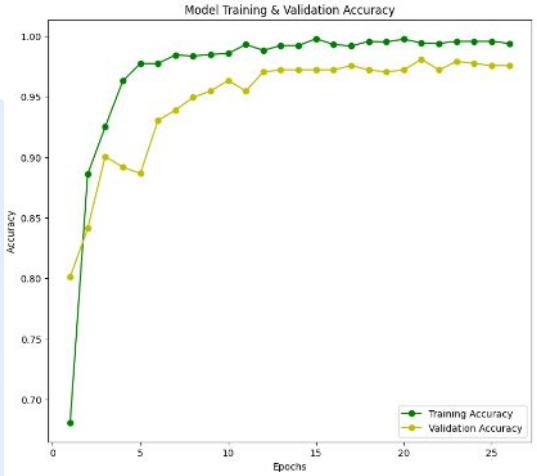
03 Modèles utilisés :

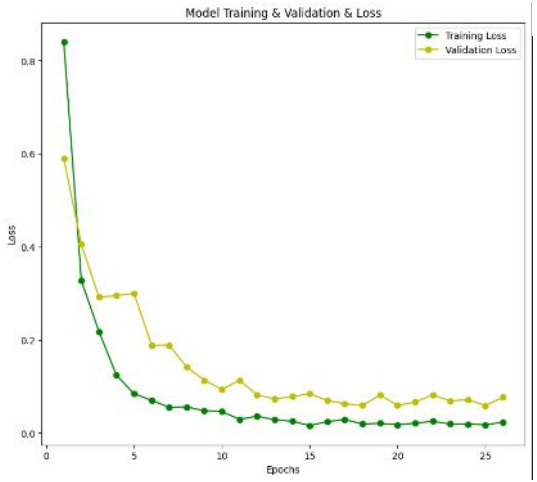
EfficientNetB1

Accuracy pour le training set : 1.0

Accuracy pour le test set : 0.81

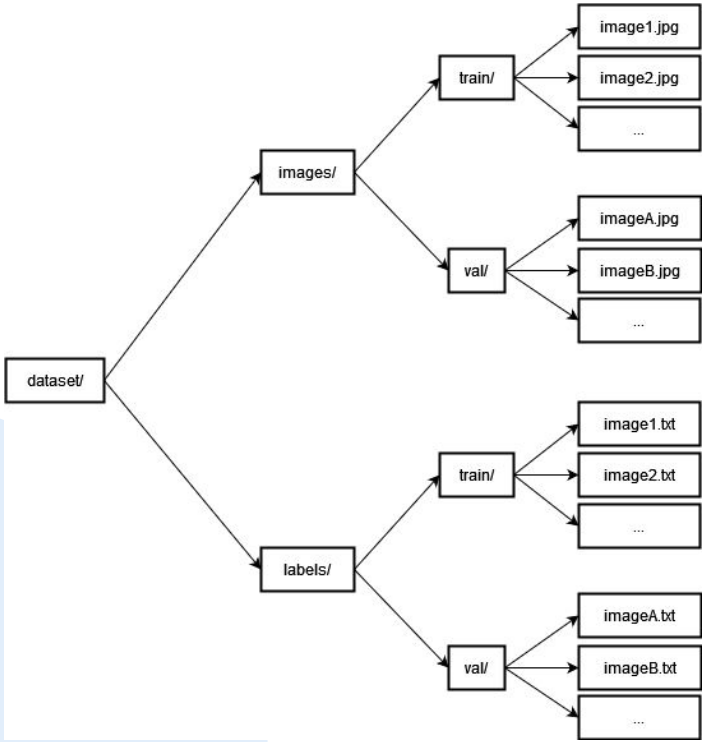






03 Modèles utilisés

YOLO



Mise en forme du dataset

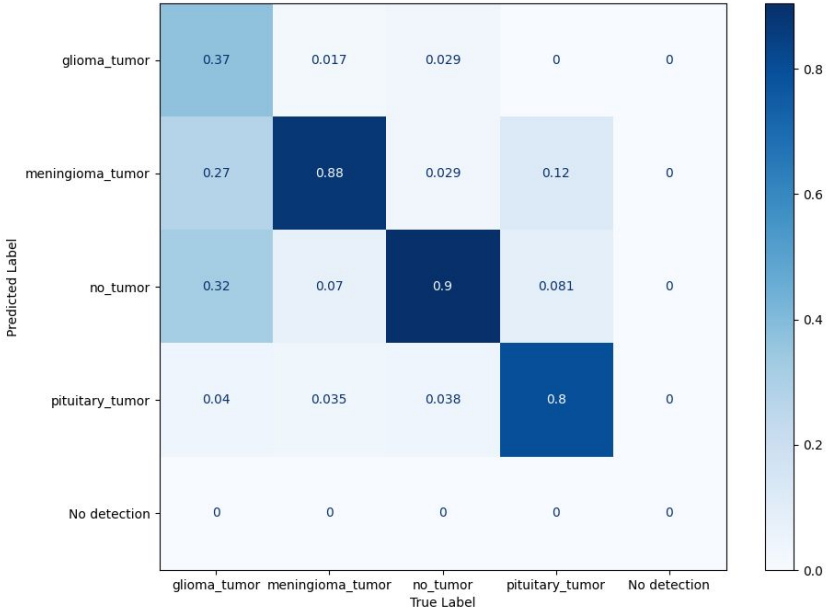
zidane.txt

```
0  0.481719 0.634028 0.690625 0.713278
0  0.741094 0.524306 0.314750 0.933389
27 0.364844 0.795833 0.078125 0.400000
```

03 Modèles utilisés

YOLO

Résultats des tests

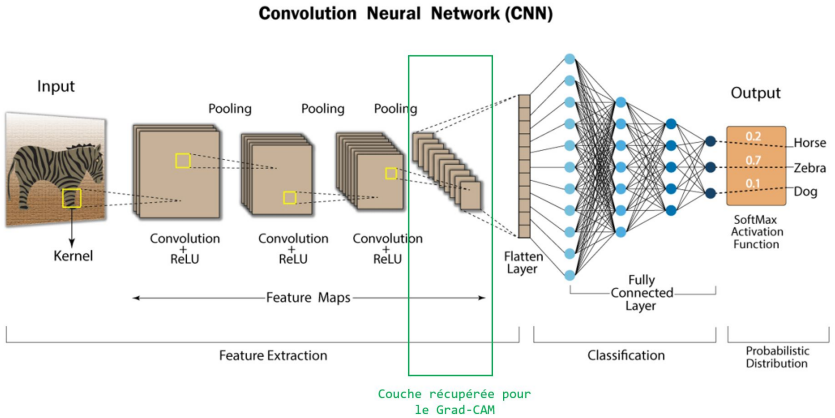


Précision : 77,48%
Recall : 73,75%

Accuracy : 74,1%

03 Modèles utilisés

YOLO avec heatmaps



Modèle depuis lequel on récupère les couches : EfficientNet

Création des heatmaps

Fonctionnement de Grad-CAM (Gradient-weighted Class Activation Mapping) :

1 - Sélection de la dernière couche de convolution (avant les couches entièrement connectées)

2 - Calcul des gradients de la classe d'intérêt et pondération de ces derniers

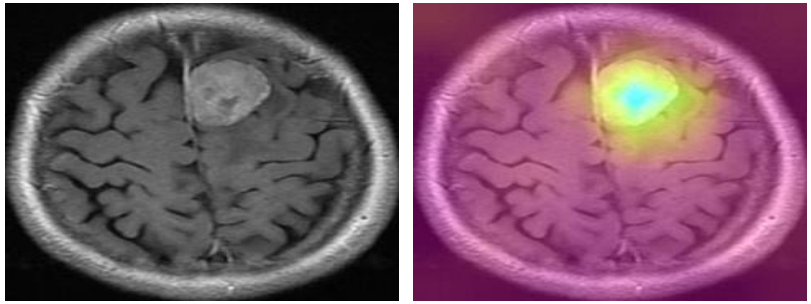
3 - Génération des cartes de chaleur à partir des cartes pondérées

4 - Superposition sur l'image d'origine : formation des cartes de chaleur

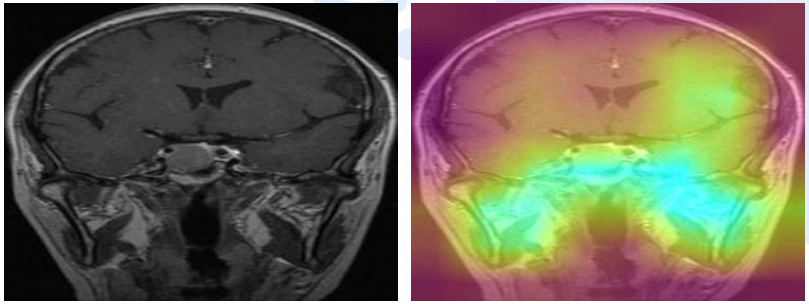
03 Modèles utilisés

YOLO avec heatmaps

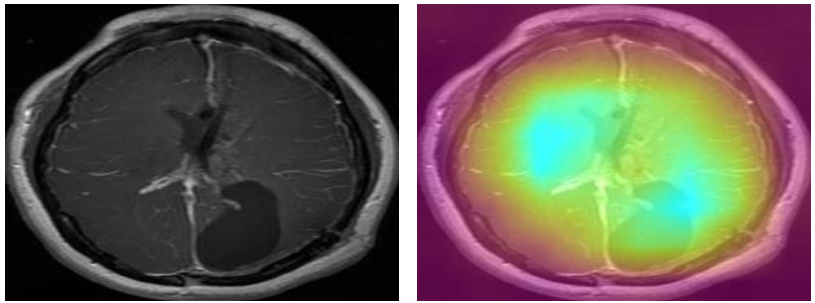
Présentation des heatmaps



Heatmap facilement exploitable



Heatmap moyennement exploitable



Heatmap difficilement exploitable

03 Modèles utilisés

YOLO avec heatmaps

Utilisation de la librairie cv2 (threshold et findContours)

1 - Création d'un masque binaire pour récupérer les zones intéressantes (choix du threshold)

2 - Récupération des bords des zones créées, et création des boîtes avec ces valeurs

3 - Limitation du "bruit" avec des contraintes de taille et de proportions

Récupération des boîtes englobantes

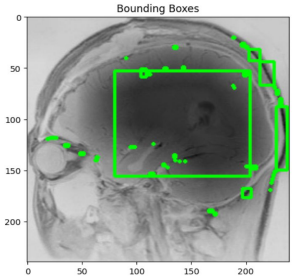
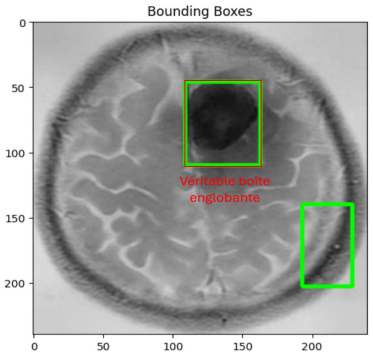
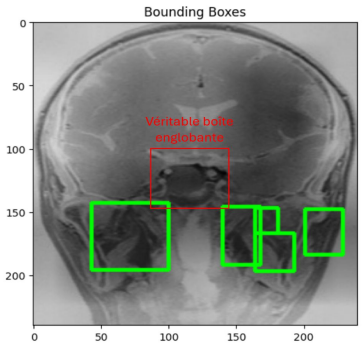


Image avec "bruit"



Threshold trop grand

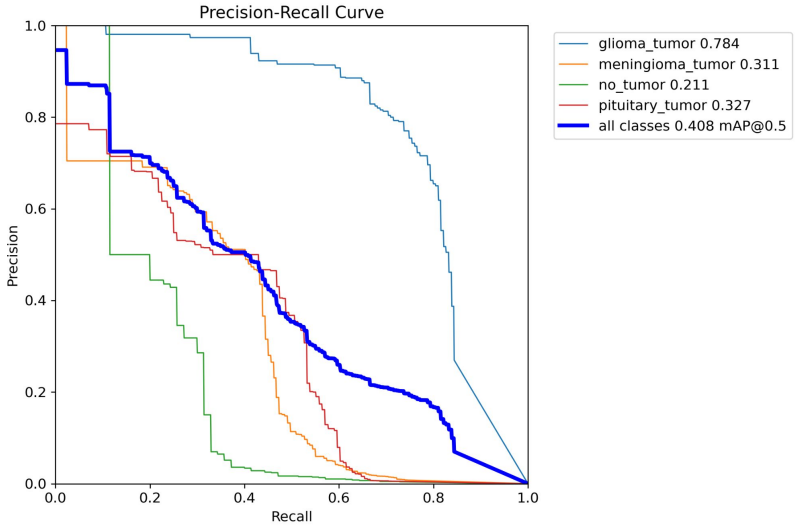


Threshold trop petit

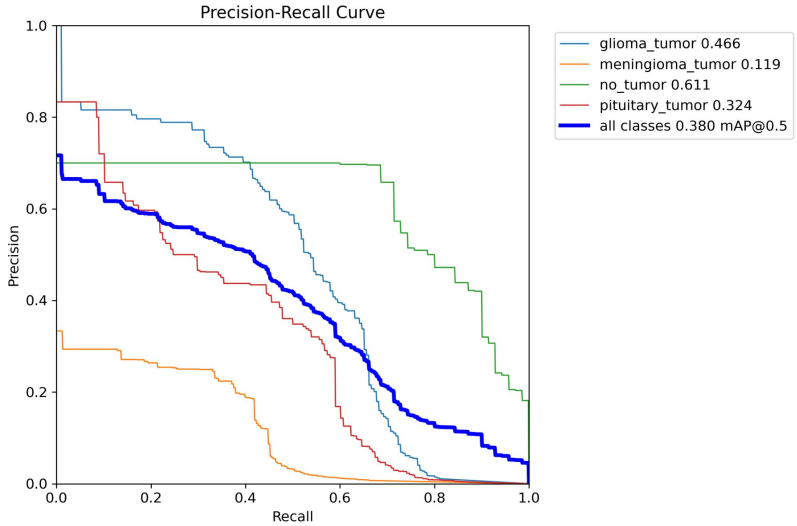
03 Modèles utilisés

YOLO avec heatmaps

Résultats



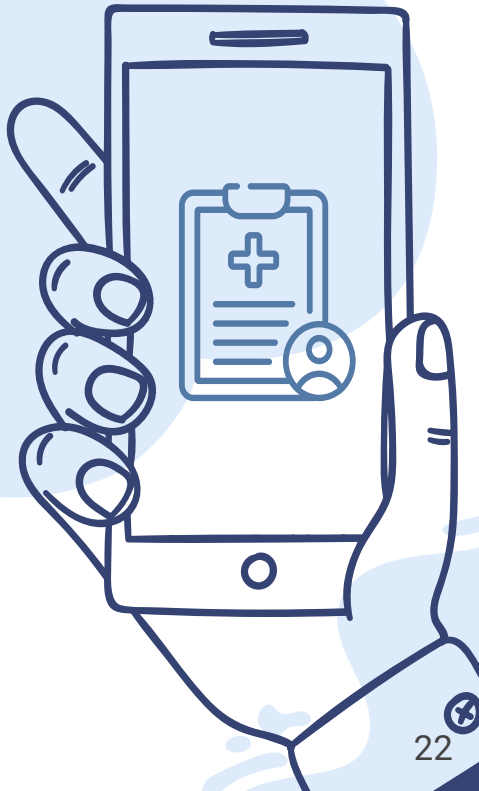
Premier test : on ne garde que la plus grande boîte englobante



Second test : on garde plus de boîtes englobantes

04 Conclusion

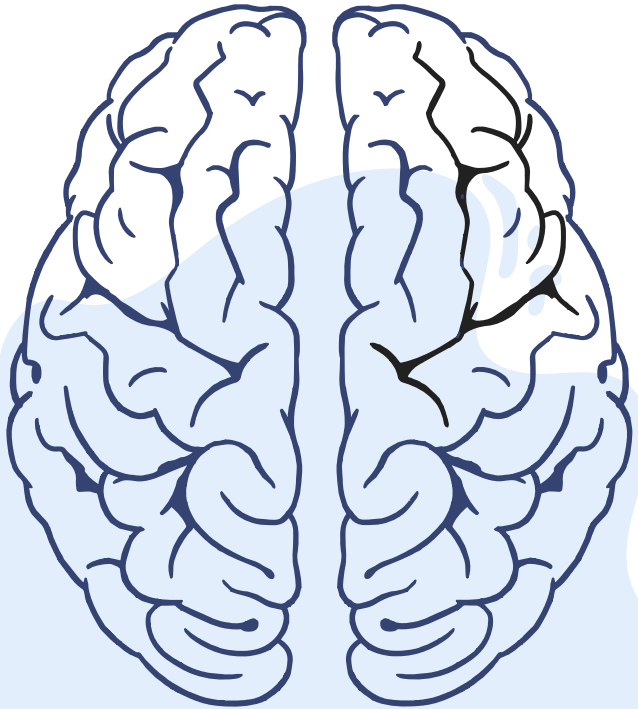
Modèle	Accuracy sur le test set
DenseNet121	75%
DenseNet201	76%
ResNet50	75.4%
ResNet101	74.9%
ResNet152	75.4%
YOLO	74.1%
YOLO avec heatmaps	40,8%
EfficientNetB1	81.7%



04 Conclusion

Amélioration possible

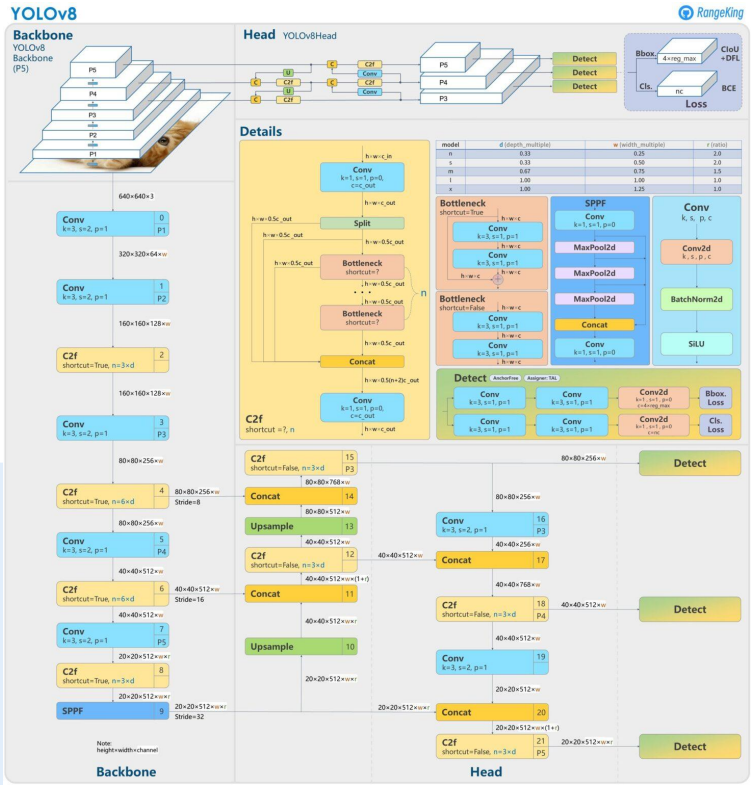
- Augmentation des données (GANs, etc.)
- Enrichissement du dataset
- Optimisation des modèles:
    - Nas
    - Ensemble learning
- Collaboration interdisciplinaire



Questions ?

03 Modèles utilisés

YOLO



Fonctionnement

- Couches de convolution et d'activation

- Blocs résiduels (amélioration de la propagation des gradients)

- Feature Pyramid Network (gestion de la détection multi-échelle)

- Têtes de détection (prédiction des boîtes, des scores et des classes)