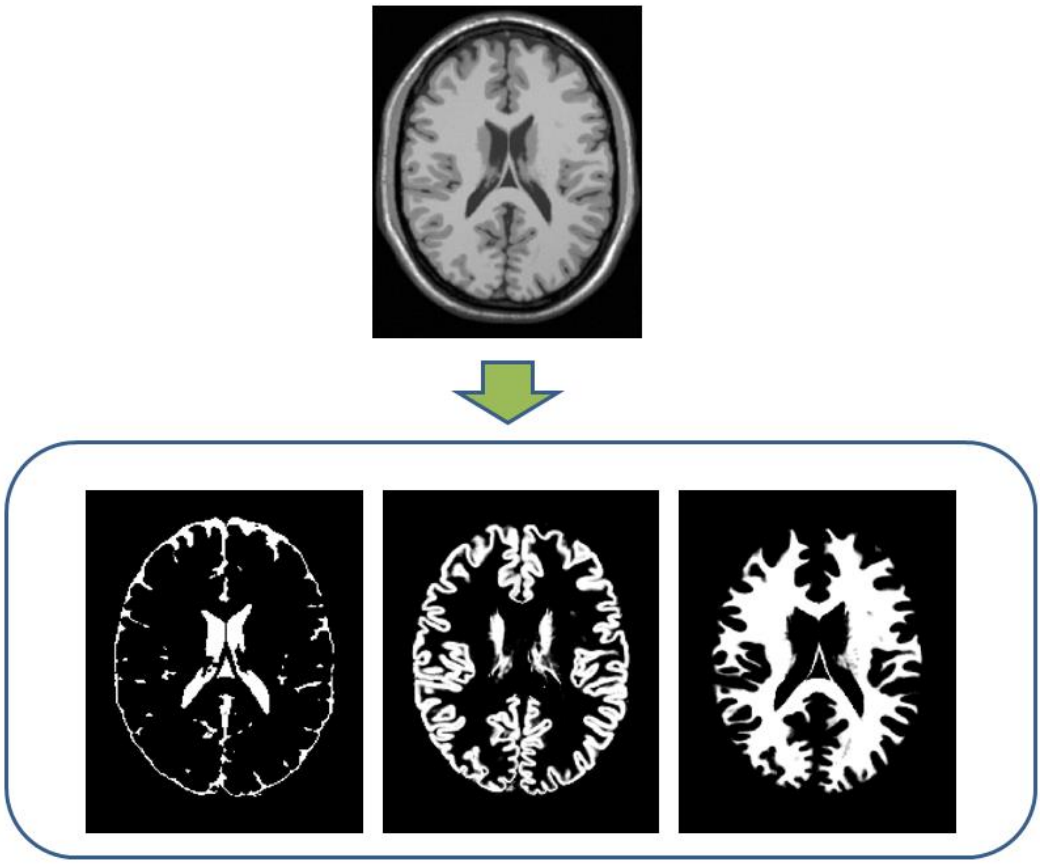
TICV

2020-2021

Atelier Python :

Segmentation d’images cérébrales

Préparé par: Linda Marrakchi



I) Introduction

La segmentation des tissus cérébraux à partir de l'Imagerie par Résonance Magnétique (IRM) présente un grand intérêt pour l'étude de pathologies cérébrales et la compréhension des modifications structurelles associées à ces pathologies. Le cerveau humain est constitué de substance blanche, substance grise et de liquide céphalorachidien comme illustré sur la figure1 suivante :

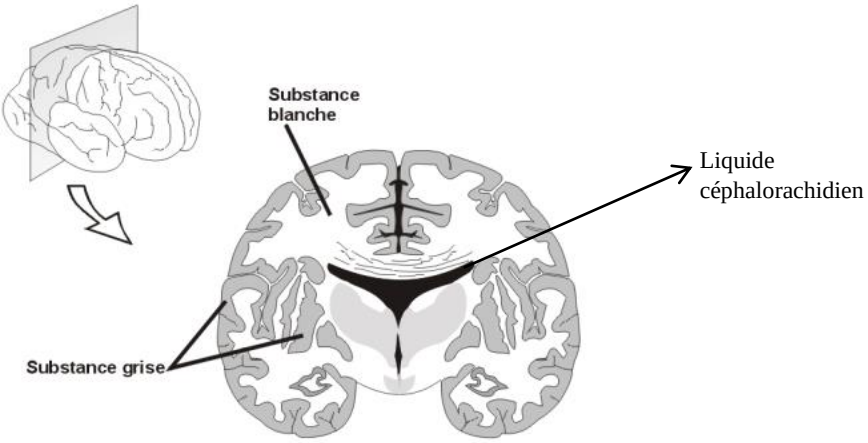


Figure1 : Coupe coronale du cerveau illustrant les différents tissus du cerveau.
Image adaptée de [1]

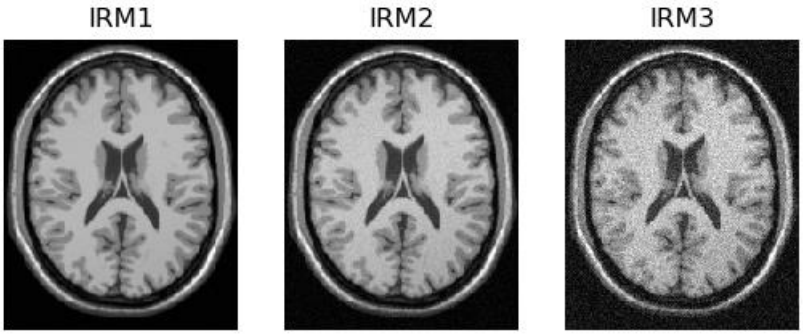
La substance grise se situe soit en périphérie soit à l'intérieur du cerveau. La substance grise périphérique est appelée cortex cérébral et la substance grise à l'intérieur du cerveau forme des structures appelées « noyaux gris centraux ».

L'objectif de ce mini projet à travers cette application inspirée du domaine de la neuroimagerie est d'apprendre à manipuler des images sous python et s'initier à la segmentation d'images.

II) Lecture et affichage d'images :

Vous disposez de trois images du cerveau en coupes axiales : « irm1_cut.jpg », « irm2_cut.jpg » et « irm3_cut.jpg ». Elles représentent toutes le même cerveau simulé du site BrainWeb.

1) En utilisant la fonction « imread » de Opencv, lisez chacune de ces images. Chaque image est une image en niveaux de gris. Choisissez la bonne option pour qu'elle soit lue correctement.
2) Vérifiez que chaque image a été lue correctement en utilisant la fonction « print ». Si l'image n'a pas été lue, la fonction « print » va renvoyer 'None'.
3) A l'aide de la fonction « shape », affichez les dimensions de l'image.
4) Affichez séparément chacune de ces images avec opencv.
5) En utilisant la librairie matplotlib, affichez toutes ces images dans une même fenêtre avec les noms de chaque image (IRM1, IRM2, IRM3) comme suit :



III) Ecriture d'images :

Dans cette section, il s'agit d'apprendre à faire des manipulations simples sur des images et de savoir écrire une image avec opencv. Pour cela vous utiliserez les images suivantes : « csf_cut.jpg », « wm_cut.jpg », « gm_cut.jpg ». Chacune de ces images représente des probabilités d'appartenance à chaque tissu cérébral codées entre 0 et 255. Il s'agit dans cette section de transformer chacune de ces images en image binaire.

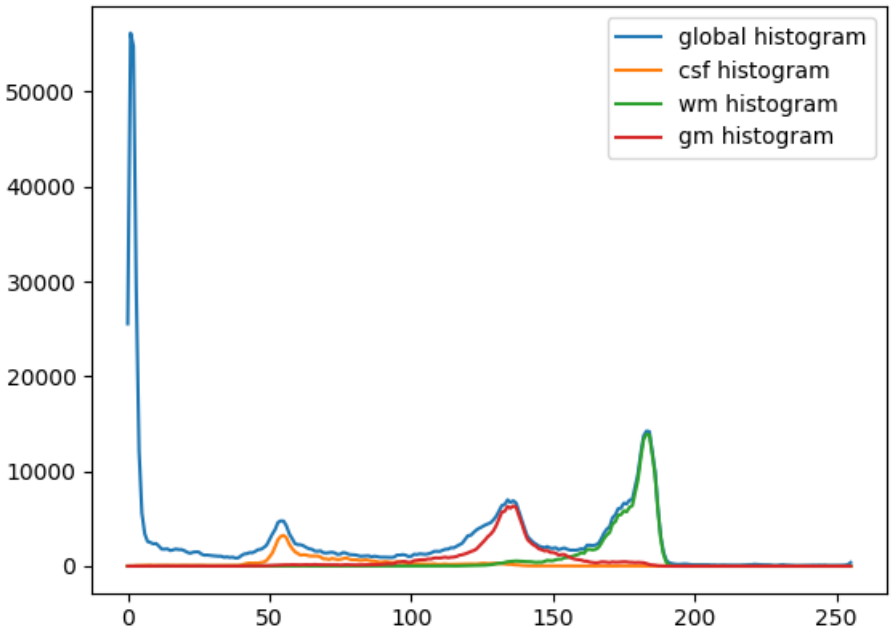
1) Lisez chacune de ces images avec opcncv.
2) Ecrivez une fonction qui s'appelera « binarizing_mask » et qui transforme les niveaux de gris compris entre 0 et 127 en la valeur 0 et les valeurs comprises en 128 et 255 en la valeur 255. Cette fonction doit renvoyer l'image transformée.
3) Appliquez cette fonction à chacune des images que vous avez lues précédemment et écrivez les images résultats. Les images résultats auront respectivement les noms suivants : « csf_mask.jpg », « gm_mask.jpg », « wm_mask.jpg ».
4) En utilisant la librairie matplotlib, affichez les images « irm1_cut.jpg », « csf_mask.jpg », « gm_mask.jpg », « wm_mask.jpg » dans une même fenêtre.

IV) Histogrammes et masques :

Dans cette section, vous allez apprendre à manipuler l'histogramme de l'image. Pour cela vous allez utiliser les images : « csf_mask.jpg », « gm_mask.jpg », « wm_mask.jpg » que vous avez générées. Chacune de ces images représente un masque d'un tissu cérébral. Par exemple « csf_cut.jpg » représente un masque du liquide céphalorachidien du cerveau.

1) Avec la librairie opencv, calculez l'histogramme de l'image « irm1_cut.jpg » et affichez-le avec la librairie matplotlib.
2) Avec la librairie numpy, calculez l'histogramme de l'image « irm1_cut.jpg » et affichez-le avec la librairie matplotlib.
3) Avec la librairie opencv, affichez l'histogramme à l'intérieur de chaque tissu cérébral. Vous utiliserez pour cela l'image « irm1_cut.jp » et le masque de la région d'intérêt.

4) Affichez dans un même graphique l'histogramme global de l'image ainsi que les histogrammes de chaque région d'intérêt comme suit :



V) Segmentation par seuillage :

Dans cette section, il s'agit de partitionner le cerveau en 4 régions d'intérêt en utilisant une technique de seuillage simple :

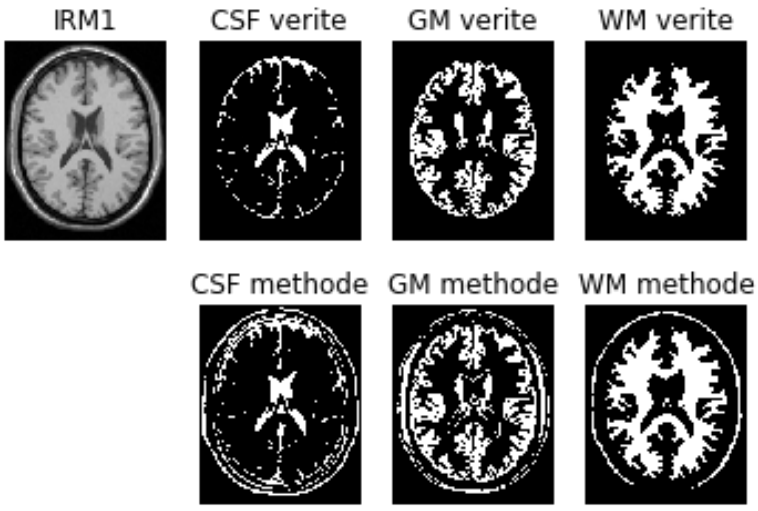
- Le fond (étiquette associée = 0)
- Le liquide cérébrospinal (étiquette associée =50)
- La substance grise (étiquette associée =150)
- La substance blanche (étiquette associée = 200)

1) Calculez l'histogramme de l'image « irm1_cut.jpg » et affichez-le avec la librairie matplotlib.
2) A partir de l'histogramme de l'image définissez visuellement un seuil inférieur et un seuil supérieur d'appartenance à chaque région d'intérêt. Pour cela vous considérerez que chaque gaussienne forme une classe.
3) Créez une nouvelle image d'étiquettes de même taille que l'image « irm1.jpg ».
4) A l'aide d'une boucle 'for' parcourez l'image d'origine et remplissez chaque pixel de l'image d'étiquettes avec l'étiquette qui lui correspond. Par exemple, un pixel qui se trouve entre 0 (seuil inférieur pour le fond) et 25 (seuil supérieur pour le fond) aura comme étiquette 0 (l'étiquette du fond).
5) Affichez l'image d'étiquettes.
6) Enregistrez l'image d'étiquettes sous le nom « segmentation_seuillage.jpg ».
7) Effectuez toutes ces opérations avec l'image « irm3_cut.jpg ». Que remarquez-vous ?
8) Question bonus : Appliquez à l'image 'irm3_cut.jpg' un filtre correcteur de bruit (par exemple : filtre median, filtre moyenneur, filtre gaussien) et refaites les opérations précédentes. Que remarquez-vous ?

VI) Segmentation par classification :

On se propose dans cette section de segmenter les tissus cérébraux en utilisant une méthode de classification non supervisée (kmeans, GMM, …).

1) Appliquez à l'image « irm1_cut.jpg » une méthode de classification non supervisée pour la segmenter en quatre régions. A chaque pixel de l'image sera attribuée une nouvelle étiquette selon la région à laquelle il appartient. L'image d'étiquettes finale sera sauvegardée sous le nom « segmentation_methode.jpg ». Où « methode » désigne le nom de la méthode que vous avez utilisée. Par exemple le résultat de la segmentation avec kmeans sera appelé « segmentation_kmeans.jpg ».
2) A partir de l'image d'étiquettes que vous avez précédemment créée, générez quatre images de masques contenant les masques de chaque région et sauvegardez les sous le nom « csf_methode.jpg », « gm_methode.jpg » et « wm_methode.jpg ».
3) Affichez dans une même fenêtre les masques obtenus avec votre méthode de classification ainsi que les masques représentant la vérité terrain : « csf_mask.jpg », « wm_mask.jpg » et « gm_mask.jpg » calculés à la section III) comme suit :



VII) Validation :

Cette section vise à introduire quelques critères de validation utilisés en traitement d'images pour évaluer les performances d'une méthode de segmentation d'images. Les masques « csf_mask.jpg », « wm_mask.jpg » et « gm_mask.jpg » calculés à la section III) correspondent à la vérité terrain pour les régions csf, wm et gm que l'on cherche à segmenter. En vous servant de cette vérité terrain ainsi que des masques que vous avez obtenus par votre méthode de classification :

1) Calculez le coefficient de DICE [2] pour chaque région d'intérêt.
2) Calculez pour chaque région d'intérêt le nombre de faux positifs (FP), faux négatifs (FN), vrai positifs (VP) et vrai négatifs (VN) [3].

VIII) Pistes d'amélioration :

On se propose dans cette section d'améliorer les performances des méthodes de segmentation testées précédemment en appliquant des prétraitements et des post-traitements.

1) Filtrage :

Evaluez les performances de votre méthode de classification sur l'image « irm3_cut.jpg ». Appliquez un filtre (median, moyenneur ou gaussien) qui permet de corriger le bruit dans l'image puis évaluez de nouveau les performances de votre méthode de classification sur cette nouvelle image. Que remarquez-vous ?

2) Morphologie mathématique :

Appliquez des opérations morphologiques aux masques obtenus par votre méthode de classification afin de se rapprocher des masques initiaux fourni dans l'énoncé et qui représentent la vérité terrain. Calculez de nouveau les performances de votre méthode après les post traitements.

Ressources à consulter :

1) https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html
2) https://matplotlib.org/3.1.1/tutorials/introductory/pyplot.html
3) https://scikit-learn.org/stable/modules/clustering.html#clustering

Librairies à installer :

Opencv, Numpy, Scikit-learn, Matplotlib

Références :

[1] http://tpelyceeledouxintelligence.e-monsite.com/pages/content/ii-les-mecanismes-de-l-intelligence/presentation-generale-du-cerveau.html

[2] https://fr.wikipedia.org/wiki/Indice_de_S%C3%B8rensen-Dice

[3] https://fr.wikipedia.org/wiki/Matrice_de_confusion