Computer Vision for Multiple Sclerosis Detection in Portable MRI Machine

CIS 581 : Computer Vision and Computational Photography

Fall 2021 : Final Report

Aishwarya Wesanekar Anamil Mehta Jal Mahendra Panchal Parimal Mehta

I. Introduction

Magnetic resonance imaging (MRI) is a medical imaging technique used in radiology to form pictures of the anatomy and the physiological processes of the body. An MRI scan is used to investigate or diagnose conditions that affect soft tissue.

Multiple Sclerosis (MS) is a complex inflammatory and degenerative disease of the central nervous system. MS causes demyelinating lesions that can be identified using MRI scans. 3T machines are normally used in a hospital setting to perform these MRI scans. They are expensive and not portable, limiting their reach to remote areas and to places like the emergency room or the intensice care unit. Recent advances in MRI technologies have resulted in the development of portable low-power MRI machines. Hyperfine, Inc is one such company that has developed a portable low-power 64mT 65 W MRI machine. Such portable MRI machines provide the advantage of being usable in non-conventional areas without the need of a shielded environment. Fluid-attenuated inversion recovery (FLAIR) images are more sensitive in lesion detection as compared to T2-weighted MRI images [2].

In this study, we will develop an algorithm to detect MS lesions in FLAIR MRI scans of a portable low-power (64mT) MRI machine and find the accuracy of detected lesions with respect to a manually annotated scan. A similar comparison will then be made using the same algorithm on the paired scans from a 3T machine. In the end, we will present results from the 3T and 64mT scans and discuss future paths to improve detection.

The data set for our project includes the 3T and 64mT MRI scans of five different subjects. We have selected 5 slices from he axial place to detect lesions. All subjects are known to have lesions in their scans. However, the extent of lesions in each subject's scans differs and hence this data set allows us to test our algorithm against scans that exhibit different level of MS stages. This in turn will help us in creating a more robust algorithm that can potentially work for many different data sets of varying sizes and difficulties.

II. Methods

We have obtained paired 3T and 64mT FLAIR scans from 5 subjects having MS lesions. The scans were performed on the same day at the Hospital of University of Pennsylvania. Of all the slices in the scans, we selected 5 slices from the axial view to detect the lesions. Slices were selected where the lesions were most clearly visible.

The overall methodology involves pre-processing the scans to make sure we have equivalent 3T and 64mT scans (registration), and to remove the skull (brain extraction). We then use unsupervised learning on the pre-processed scans to get intensity clustering parameters that are then fed to more Flood fill segmenting algorithm that outputs a mask for the lesions. Once the lesions are segmented, we compare the segmented lesions with the corresponding manually annotated mask for each slice. We use the dice coefficient to quantify the accuracy of lesion detection. The detailed steps are explained as follows:

Brain Extraction

Brain Extraction is the first step that's incorporated in any of the MRI image processing algorithms. This is done to exclude the outer periphery which generally represents the skull, leaving only the region that's occupied by actual brain tissue. This is an essential step as due to the density of the bone structure, the skull can often throw image segmentation algorithms off and decrease the accuracy. In our project, we perform brain extraction by using the open source FSL library's brain extraction

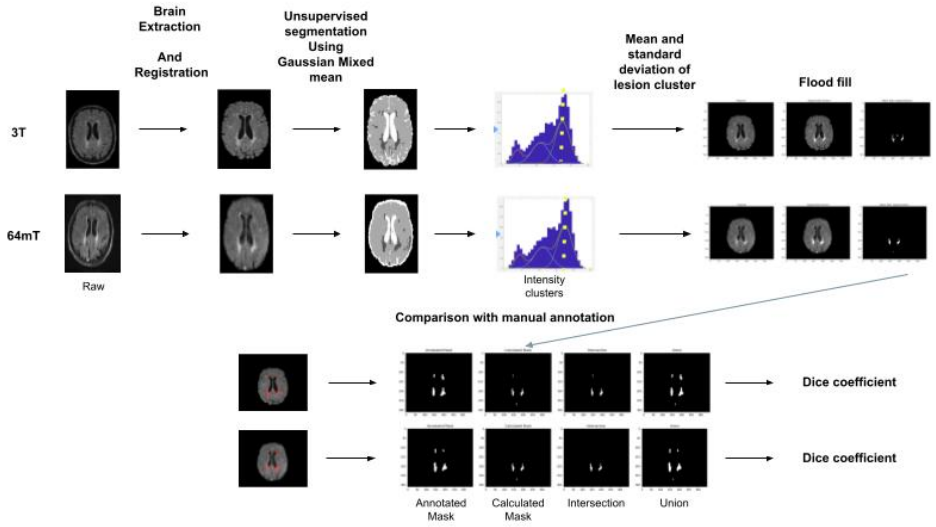


Fig. 1: **Overview** The image shows the steps from raw MRI scans to the calculation of accuracy of lesion detection.

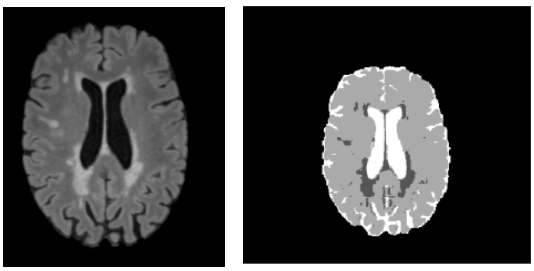
tool (BET). This tool helps us extract the brain by deleting the non brain tissue from the scan. More details about brain extraction can be found in Appendix I.

Registration

This is an important component of pre-processing as it ensures that the 3T and 64mT images being compared are equivalent in nature. This is because all the raw MRI scans are 3D representations that are put together using multiple scans/slices of the brain. In case of 3T images, the higher capacity of these machines ensure a higher resolution and hence the complete 3D scan includes roughly 256 slices. On the other hand, in case of the low powered 64mT scans - there are only 36 slices for a subject. For reference, these 'slices' can be thought of as different layers that when put on top of each other would make a 3D model of a brain.
We perform registration to align the two scans so that the selected slice in each scan corresponds to the same brain region.
For this, we use an open-source tool called ITK-SNAP. ITK-SNAP is a software application used to segment structures in 3D medical images. It provides semi-automatic segmentation using active contour methods, as well as manual delineation, image navigation, and other supporting utilities. We used the automatic registration feature to register the 64mT images with the 3T images using image centers. After this, we manually adjusted the z axis(depth) until the 3T and 64mT slices matched each other in appearance. After this, we have chosen 5 relevant slices from each of our 5 subjects and use them to detect lesions and make comparisons.

Gaussian Mixture modeling

Gaussian mixture modelling (GMM) is a probabilistic technique that assumes that all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. One can think of mixture models as generalizing k-means clustering to incorporate information about the co-variance structure of the data as well as the centers of the latent Gaussians. We have utilized scikit-learn based GMM functions to build our own code that helps our algorithm to determine segmenting thresholds in an unsupervised format.



(a) Brain extracted (b) Segmented

Fig. 2: P23 3T MRI Scans

As can be seen in the figure above, a brain extracted MRI scan generally have the following regions -

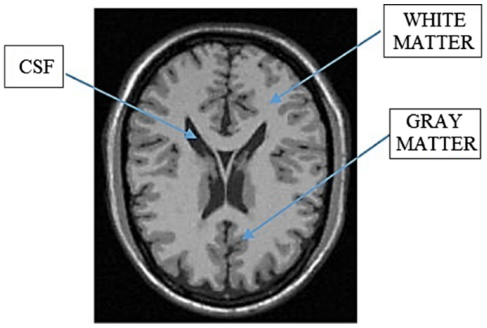
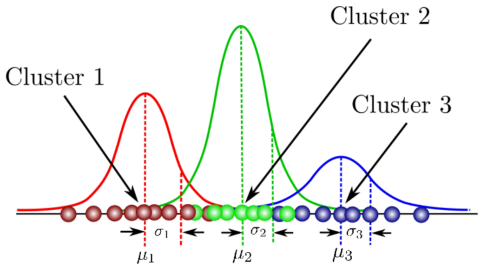


Fig. 3: Analogy of the brain

- Gray matter - Gray matter is the region associated with movements and voluntary actions. This is usually shaped like a butterfly and is easily distinguishable both in the original MRI scan as well as the segmented example where it is coloured white
- White matter - White matter is the brain tissue that surrounds the gray matter. It is also quite distinguishable and is coloured gray in the segmented image
- In case the subject is suffering from MS, the MS lesions will show up in their MRI scans as light coloured lobes/ patches around the gray matter of the brain. In almost all cases, these patches show up over the white matter and are easily distinguishable due to their higher pixel value as compared to the rest of the MRI. These are shown in the segmented sample image in a dark gray tone.

Hence, the main reason behind using GMM is to exploit the different pixel values of certain regions on the MRI scan to identify them as different regions. Gaussian mixture models helps us plot the intensity of these pixel values to give us a single Gaussian curve which can be broken down to be made up of multiple Gaussian curves.



(a) Clustering based on gaussian curves

Fig. 4: Gaussian curves

These smaller Gaussian curves represent a different region in an image and can be used to find the ideal threshold values and tolerances to separate out different regions in order to segment the lesions.

During our analysis and trials, we noticed the best results were derived when the threshold was set up to be the mean + 1 standard deviations.

GMM gives the mean and standard deviation of each group. When the mean was used as the threshold and the standard deviation as tolerance to get the lesion region, it was seen that the output was not quite accurate.
Through GMM, the group with maximum mean

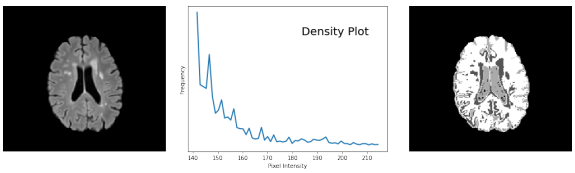


Fig. 5: Histogram Analysis of Lesion Segment

intensity was separated out and then the histogram was plotted. In Figure 5, the rightmost image shows the segmentation of the image through GMM. The histogram is plotted for the group where the lesion regions are present. Through the histogram, it can be concluded that the mean of the group is actually lower than what should have been the actual threshold - this can be reasoned out due to the probabilistic prediction used by GMM.
Hence, the threshold for the next step is taken as the mean + one standard deviation and the tolerance is taken as the standard deviation directly.

Flooding

Flood filling is a technique in which all pixels with similar intensities can be identified in an image. This also lets an user manipulate the identified pixels to make it standout in the result. Hence the flood filling algorithm is really useful and allows us to segment out just the lesions from the brain extracted MRI images in form of masks.
This technique is used in combination with the GMM part of our algorithm. After GMM has been able to identify the different regions and their threshold values, the region of interest (lesions) is chosen. Because we know that the pixel intensity of the lesions is the highest in an MRI image, it becomes easy to chose the cluster of interest as the cluster representing the lesions will naturally have the highest mean for its Gaussian curve. Once we find out the cluster of interest, we use its mean and standard deviation to calculate the threshold

and tolerance that will be then be used to segment the lesions to create a mask. These values are then passed on to the flood filling algorithm.
The flood filling algorithm takes two inputs from the GMM function. The first one of these being a threshold value and the second one is a tolerance value. Both these values signify the pixel intensity value on a scale from 0 to 255. For this function, our code finds out all pixels that have an intensity value that is more than the passed threshold and stores their coordinates. These coordinates are then passed onto the flood fill algorithm which is made to loop over each pixel coordinate where it does a breadth first search. In this process, the function loops over all neighboring pixels that lie within the defined tolerance of the threshold. This is done in a wildfire way where the algorithm would keep looking for neighboring neighbors until it cant find any more pixels that have an intensity within the defined tolerance of the threshold. Our code sets all such neighboring pixels to an intensity of 255 and the rest to 0. This process is repeated for all the stored coordinates and ends up giving us a mask of the MRI where only the detected lesions are shown and every other pixel is turned black.

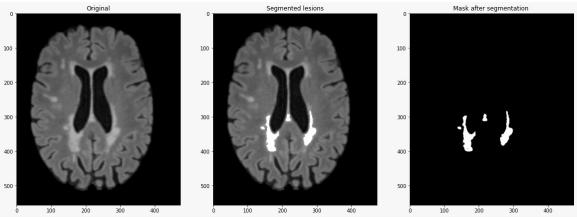


Fig. 6: Flood filling

These masks are then used to compare against a known annotated mask that has been manually created by us to check for accuracy of the detected lesions.

Annotated Masks and Dice coefficient

Annotated Masks are created manually. Here, we use the ITK-SNAP open source software to manually go into each slice of a 3T or a 64mT MRI scan and color the lesions by hand. The process of annotation is fairly common when it comes to testing the robustness of MRI lesion segmenting algorithms. It generally involves a qualifies radiologist manually going through all slices of an MRI and annotating / colouring the lesions by hand. These coloured lesions are then converted into black and white masks which then serve as the ground truth. Hence the annotated masks are then used to analyze the accuracy of a segmenting algorithm. In our project, we annotated the masks ourselves instead of a radiologist and used these to perform the final analysis. All 3T and 64mT masks that are created using our algorithm are then compared against the ground truth.

We use dice coefficient to perform the aforementioned comparisons. A dice coefficient or a dice score is a value between 0 and 1. It is a widely accepted tool that is used to quantify the similarity between a ground truth versus a segmented image. It is defined as the size of the intersection of two segments divided by the total size of the two segmentations. For sake of a better understanding, imagine if we have A (the ground truth segment) and B (segment from an algorithm) that are to be compared. In case of both the masks, the mask is set to white (or 255) where we have a segment of interest and black (or 0) everywhere else. So we have three main categories of interest :-

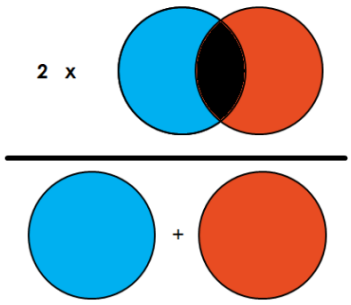


Fig. 7: Dice coefficient

1) Number of positives : This is the total number of pixels that have an intensity of 255 in the mask A
2) Number of true positives : This is the total number of pixels which have the value 255 in both A and B. So it the intersection of the regions of ones in A and B. It is the same as using the AND operator on A and B.
3) Number of false positives : This is the number of pixels that are detected in mask B (i.e. are set to 255) but are black in mask A.

Hence the dice coefficient is defined as :

$$D = \frac{2.|A| \cap |B|}{|A \cup B|} \tag{1}$$

III. Result

To derive a result, we compare each of the annotated masks to their corresponding 3T or 64mT masks created by our unsupervised algorithm. The results are presented in figure 8.

IV. Discussion

Supervised GMMs and CNNs could not be trained for a smoother image segmentation of the

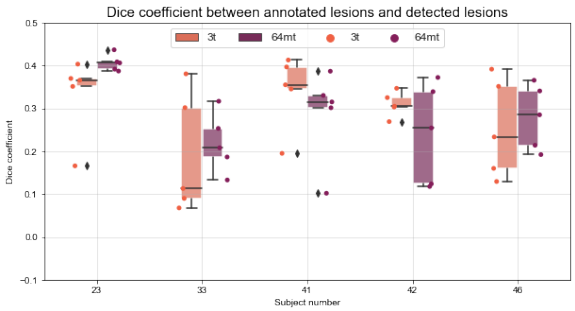
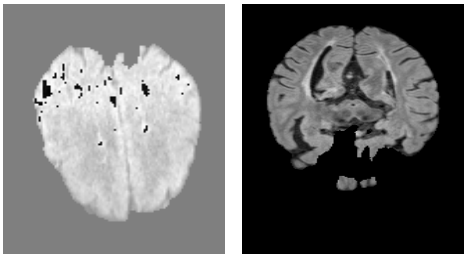


Fig. 8: Dice Coefficient Box plot



(a) 64mT Image Processing (b) Brain Partly Deleted

Fig. 9: DeepBrain Brain Extraction Issues

brain images because of the small dataset. The dataset was the main challenge that was faced in the project. For future scope, contrast filters can be used to pre-process the brain scans and get higher pixel intensity differences. In addition to dice coefficient, the analysis could be done using volumetric lesions. Another approach which could be tried out would be translating the 2D lesion detection into 3D lesion detection by processing on voxels instead of pixels.

V. Conclusion

For the subjects with larger lesions, the analysis on detection and analysis show better similarity indices as opposed to the subjects with smaller lesions. A combination of one or more techniques mentioned in Section IV can be used to improve the segmentation specifically on 64mT images and to achieve better metrics.

VI. Acknowledgement

We would like to express our gratitude towards Thomas Campbell Arnold, Pulkit Khanderwal and Dr. Paul Yushkevich from the Department of Bioengineering, University of Pennsylvania; and Dr. Joel Stein, Department of Radiology, Hospital of University of Pennsylvania for providing the guidance throughout the project.
We would also like to thank Hyperfine, Inc and the staff of the Hospital of University of Pennsylvania for giving us access to the scanned brain images from both 64mT and 3T MRI machines.

VII. Appendix I : Brain extraction methods

Deepbrain

Brain image processing tools using Deep Learning focused on speed and accuracy. We initially used this Python library for brain extraction. However, it did not perform well for 64mT images and cut off parts of the brain as well. Also, the library has not been updated since 2018, so we decided not to move forward with this library.

FSL BET

This is the brain extraction tool from the FMRIB Software Library. It is a command line based tool and worked perfectly out of the box. So, this is what we moved forward with.

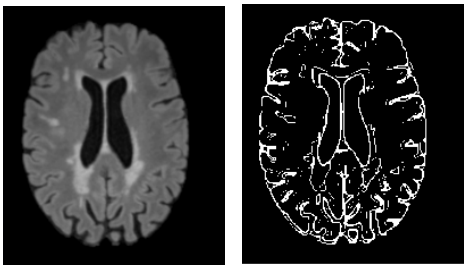
ANTSPyNet

A collection of deep learning architectures and applications ported to the python language and tools for basic medical image processing. This library is being actively worked on, however it is not compatible with Windows, and has a lot of dependencies which need to be downloaded for MacOS and Linux, so we stayed with BET.

VIII. Appendix II : Segmentation methods

Canny Edge Detection

This was part of an initial thought process where the intention was to use canny edge detection to find various bounded areas within the MRI and use the output to segment lesions out. However, on doing some further research we discovered better approaches and hence did not end up using Canny edge in the final version of the project.



(a) Input (b) Output

Fig. 10: Canny edge

Active Contour Segmentation

This is a scikit based algorithm that needed the user to specify a region of interest (in the form of a circle or a square). The active contour algorithm

then uses a 'snake' approach to search for contours. In addition to specifying a region of interest, the user also needs to tune 3 different parameters that vary between different images of the same subject. We found this difficult to tune manually in order to get the right segmentation and found the overall approach finicky.

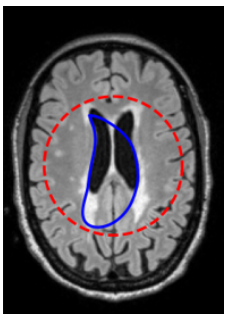


Fig. 11: Active contour segmentation

Otsu thresholding

This is an exhaustively searches for the threshold that minimizes the intra-class variance, defined as a weighted sum of variances of the two classes. Its implementation varies depending on the kind of thresholding procedures that are coded in addition to the algorithm. Out of all the thresholding techniques applied, we got relatively better result for Local thresholding. This technique although promising was not used as we found flood filling to be much more accurate and easier to train in an unsupervised format.

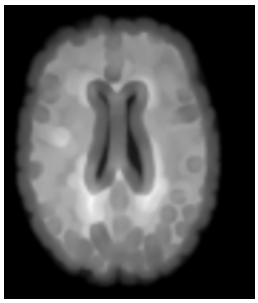


Fig. 12: Local thresholding on Otsu

Slic

SLIC is an unsupervised scikit based algorithm that segments image using k-means clustering in Color-(x,y,z) space. We were able to roughly segment the areas containing the legions but this algorithm required a lot of additional post processing that the other algorithms did not require.

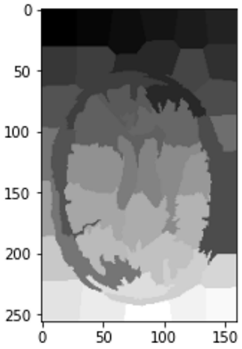


Fig. 13: SLIC

K-means clustering

K-means is an unsupervised algorithm used to identify the different segments of the interest area. It clusters, or partitions the given data into K-clusters or parts based on the K-centroids. In this project, the goal was to find certain groups based on the pixel intensities in the unlabelled data and then assign each pixel a group number. The total number of groups would be K.

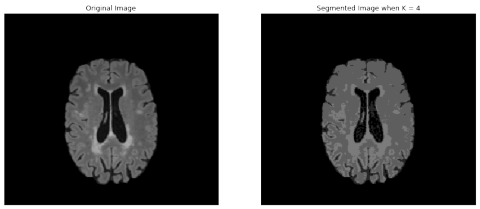


Fig. 14: K-Means Clustering

REFERENCES

[1] Deoni, S., Bruchhage, M. M. K., Beauchemin, J., Volpe, A., D'Sa, V., Huentelman, M., Williams, S. C. R. (2021). Accessible pediatric neuroimaging using a low field strength portable MRI scanner. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.3820626

[2] Arnold, T. C., Baldassano, S. N., Litt, B., Stein, J. M. (2021). Simulated diagnostic performance of ultra-low-field MRI: Harnessing open-access datasets to evaluate novel devices. https://doi.org/10.1101/2021.07.02.21259789