Automatic Segmentation of Brain MRIs

ECE 4250 Final Project Report

Christine Mayer
clm293

May 17 2019

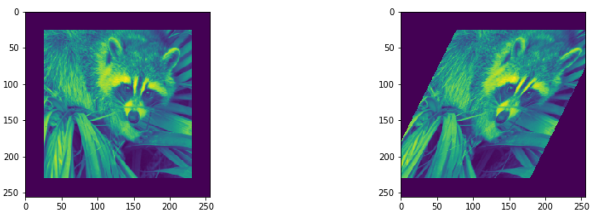
Introduction

The goal of this project is to implement an algorithm that automatically segments brain MRI's into anatomically significant regions of interest. This is an area of ongoing research at the intersection of the medical and image processing fields which will be increasingly utilized for diagnostic and treatment evaluation purposes as the reliability of the technology continues to improve.
In particular, the focus is on identifying the regions of cerebral cortex and cerebral white matter at the middle coronal slice. At the end of the document, I visualize the steps that outline my algorithm.

Optimization

The first step of the algorithm involved aligning the MRIs to be segmented with the training set MRIs, which we had an auto segmentation of. This process is known as registration – it consisted of a transformation function, an objective function, and an optimizer. Once the MRI image was registered to each of the training MRIs, we used the training manual segmentations to predict the correct label for the new MRI's segmentation. Each of these auto segmentations was then compared to the manual segmentation of the same MRI via the Jaccard index to evaluate the function's performance.

The first step I took to improve upon the results of my initial auto segmentation algorithm was to add more parameters to the transformation function in my registration process. Initially, the moving image was changed by an affine transformation involving four parameters: scale, rotation, horizontal translation, and vertical translation. These four parameters were optimized to minimize the sum of squared differences between the two images, but with additional specifications a more accurate registration can be achieved. I split the scale parameter into vertical and horizontal scale to account for an image that was stretched to a different degree along each axis. I added a horizontal shearing parameter, which displaces each point in the x direction by a distance proportional to its signed distance from the y axis down the center of the image. The vertical shearing parameter is analogous, visualized below:



(a) Image Zoomed Out for Visualization

(b) Same Image, Horizontal Shear Factor = 0.5

Figure 4: Shearing

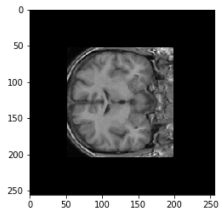
Next, I considered the function used to find the most optimal transformation parameters for the registration. In my initial implementation, I used the scipy function optimize.fmin, but this method ran into issues getting stuck in local minima. Further, it struggled to optimize as the number of parameters increased, such that my Jaccard indices actually declined when more specifications were added to the transformation. I experimented with several other optimization methods, eventually settling on the Broyden–Fletcher–Goldfarb–Shanno method, which utilizes gradient descent and hill climbing. This algorithm, provided in the scipy.minimize package, resulted in marginally better Jaccard indices that others I tried in the package.
Once I had the optimal parameters that would register the validation and training MRIs, the next step of the project was to transform each of the training segmentations with these parameters. Then, at each pixel I selected the most frequent training label out of the six training segmentations to be included in the auto segmentation. To optimize this part of the process, I decided to use a nearest neighbor interpolator rather than linear for the segmentaitons. This was a crucial step because each value in the segmentation is a key in a dictionary, so it is important that the auto segmentation contain only those values. Previously, I had used a linear interpolator and then rounded off the result to the nearest integer, but the nearest neighbor approach worked better.

Results

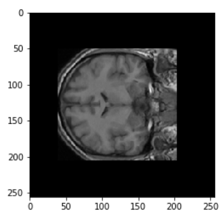
My best result over many attempts at optimization was a Dice coefficient of 0.69295.

Ideas for Future Improvements

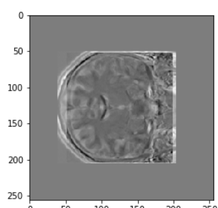
The best ideas for solving this problem involve higher level machine learning techniques. However, I did consider some more ideas at this level that could provide incrementally better results. One approach would be to weight each of the training images differently (when combining their registered segmentations into one), with weights corresponding to their relative similarities in terms of intensity values. Another idea is to implement a better tie-breaking strategy when deciding the most-frequent training label for each pixel. Currently, I arbitrarily select from the modes of each pixel. A better method might involve using a patch-based approach and looking at the surrounding pixels. A final area for improvement would be further adjustments to the optimizing function to approach the absolute minimum of the SSD of the pixels.



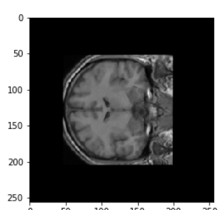
(a) First Brain MRI: Fixed Image



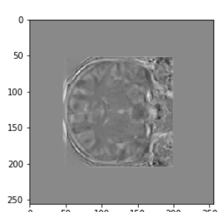
(b) Second Brain MRI: Moving Image



(c) Difference Between the MRIs

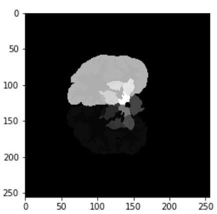


(d) Moving Image Registered to Fixed Image

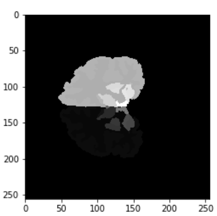


(e) Difference Between MRI 1 and MRI 2 Registerd to MRI 1

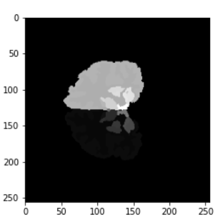
Figure 1: Image Registration Process



(a) MRI 1 Manual Segmentation

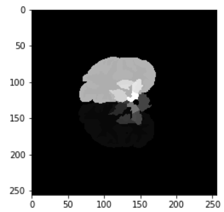


(b) MRI 2 Manual Segmentation

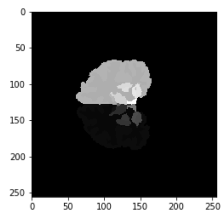


(c) Segmentation 2 Registered to Segmentation 1

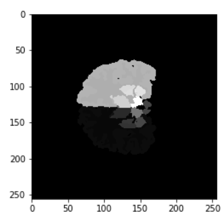
Figure 2: Segmentations Registered



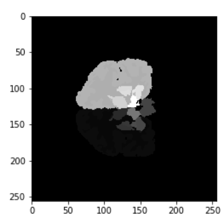
(a) First Training Segmentation, Registered



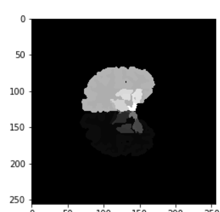
(b) Second Training Segmentation, Registered



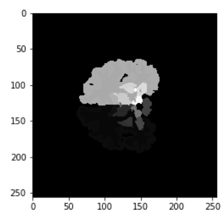
(c) Third Training Segmentation, Registered



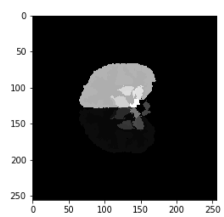
(d) Fourth Training Segmentation, Registered



(e) Fifth Training Segmentation, Registered



(f) Sixth Training Segmentation,
Registered



(g) Automatic Segmentation for
the MRI – MFTL

Figure 3: Automatic Segmentation Process