ECE 4250 Final Project Report

Kaggle Team Name: Sam Hong

May 16, 2020

Introduction

For the final project of this class, I developed a method of automatically segmenting a brain MRI scan into anatomical regions of interest. In Milestone 1, I loaded the MRI volumes and extracted the middle coronal slices for all training and validation MRIs and segmentations; in Milestone 2, I implemented a four-parameter geometric registration used to resample the manual segmentations and computed the Jaccard index. In the final submission, I made critical changes to improve automatic segmentation.

Changes Made

Denoising/Filtering

I wanted to filter the training and validation images before computing optimized parameters in case there were issues with noise. I looked through the restoration module within *scipy* to find functions that could help improve quality of the images. I tested both the bilateral filter and the non-local means filter. A bilateral filter is an edge-preserving and noise reducing filter that averages pixels based on their spatial closeness and radiometric similarity, and the non-local means filter replaces the value of a pixel by an average of a selection of other pixels values. I chose the non-local means filter because the averaging is performed only for pixels that have patches close to the current patch. Other denoising algorithms can often result in a blurry image, but the non-local means did a good job restoring original definition.

Prior to applying transformation, I first estimated the average noise standard deviation across color channels then filtered the moving images with *skimage.restoration.denoise_nl_means* to eliminate noise. This filter took a little bit of extra time and although to the human eye it can be difficult to notice a difference, I definitely saw an improvement in my score.

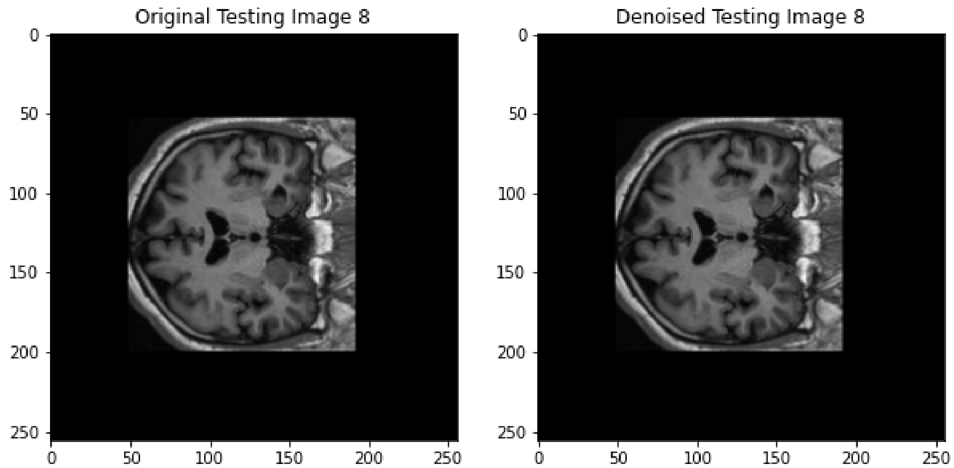

Figure 1: Before and after denoising.

Affine Transformations

I decided adding parameters to the preexisting geometric transformation would help optimize the Jaccard index in the end. Instead of using a fixed scale parameter, I converted that to scale in x-direction and scale in y-direction, increasing the parameter count from four to five.

To implement the transformation, I switched from using OpenCV *warpAffine* function to skimage *warp* function. This allowed me to scale an image in both directions and not have to worry about generating transformation matrices. Although I tested different interpolation methods, I didn't see too much of a difference in performance, so I stuck with bilinear interpolation, which is default for the function.

Optimizing

For Milestone 2, I used *scipy.optimize.fmin*, which minimizes a function using the downhill simplex algorithm. I switched to using *scipy.optimize.minimize* to try different minimization algorithms. After comparing Nelder-Mead, Powell, and BFGS, I decided BFGS had better performance and ended up using it.

Loss Function

My original loss function was a mean squared error (MSE) function, which is calculated as the average of the squared differences between the predicted and actual values. While this function is the default loss to use for regression problems, I decided that a different loss function could tackle some of its downsides. One major disadvantage with MSE is the fact it is prone to outliers it uses the concept of using the mean in computing each error value.

I tried out different loss functions, including mean absolute error loss, but did not see a large change in performance. However, I found many resources online suggesting that an effective way to measure image quality degradation is Structural Similarity Index (SSIM).

The difference between MSE/MAE and SSIM is that the former estimates absolute errors. However, SSIM is a perception-based model that considers image degradation as perceived change in structural information, which is the idea that the pixels have strong inter-dependencies especially when they are spatially close. These dependencies carry important information about the structure of the objects in the visual scene. I used *skimage.measure.compare_ssim*, which took a long time, but I definitely saw an increase in my Dice score.

Most Frequent Training Label

Originally, I used *scipy.stats.mode* to determine the modes at each pixel. I stacked six images together to make a three-dimensional array of size 6 x 256 x 256, and did not check the values of the modes. For the final submission, I checked whether the count of the mode was 1 (meaning there were no repeated intensity values), and if so, I averaged all of the values of the 8 images at the pixel. I also checked if the value of the mode was 0 (indicating black space), and if that were the case, if less than 4 of the 8 images had a value of 0, I took the maximum of the values of the 8 images.

I knew I had to modify the MFTL function because when I plotted the output of this function in Milestone 2, I noticed a lot of spotty edges. Additionally, I wanted to make sure I implemented a tie-breaking procedure to ensure that there weren't more black space than necessary.

Below are the side-by-side comparison of my original MFTL method and the revised one.

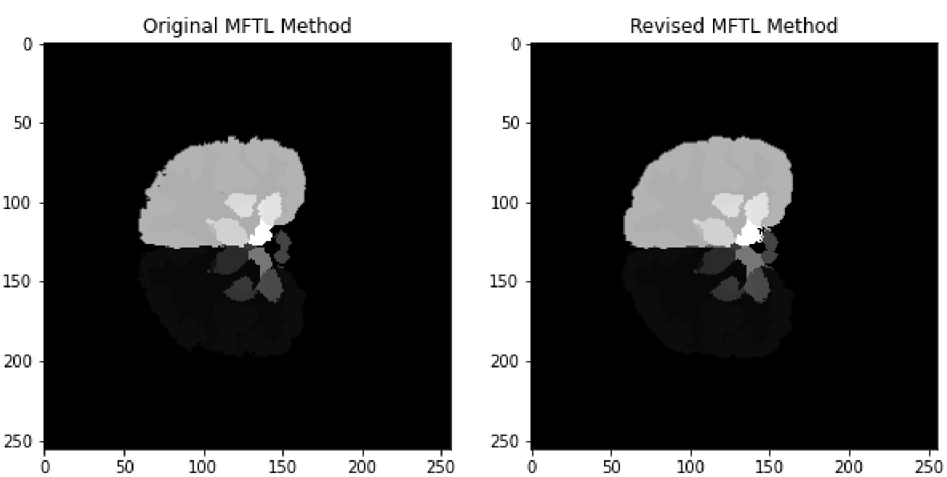

Figure 2: Original MFTL and Revised MFTL

Larger Training Set

I added the validation images to the training images data set so that there would be more images to analyze. I believe the larger sample size contributed to a better score.

Results

Below are the MFTLs of each testing image set.

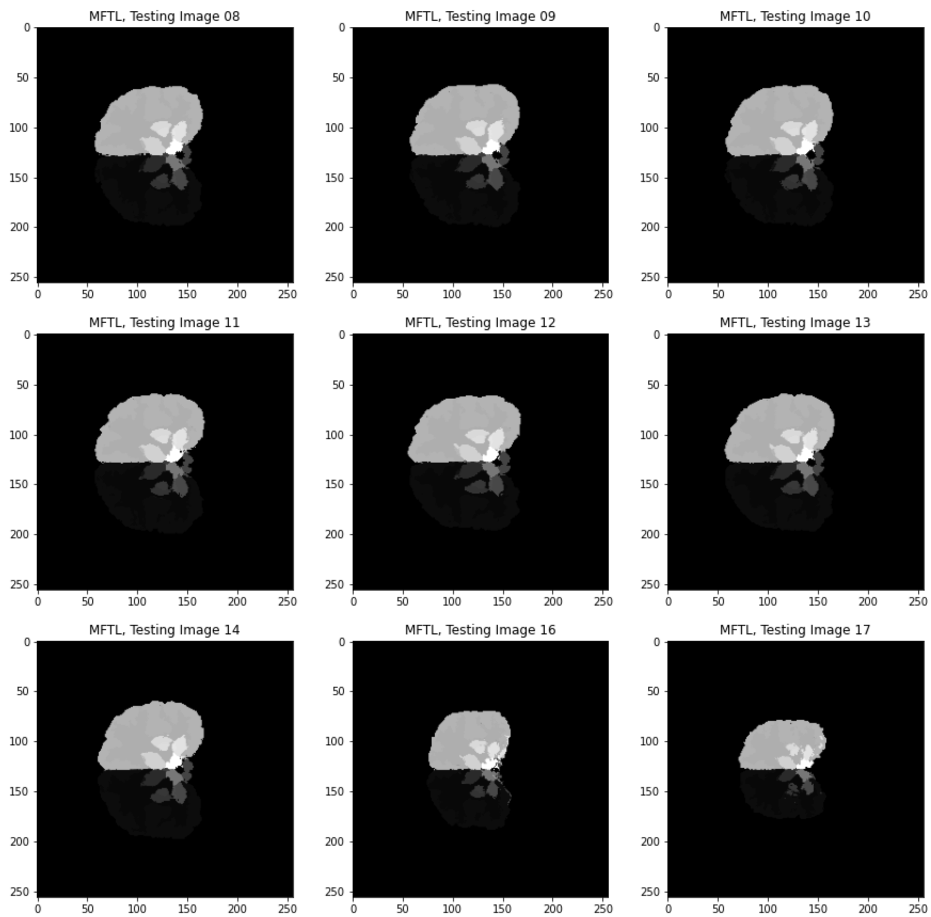

Figure 3: MFTLs of each testing image set.

The following is one set of images that were processed for this project. From left to right, the first image is the testing image 8, the second image is the training image 1, the third image is the training image after denoising, the fourth image is after registration, and the fifth image is the MFTL of all images registered with testing image 8.

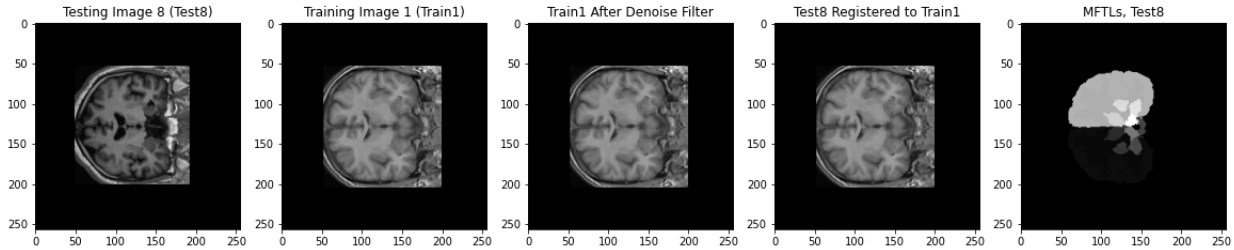

Figure 4: Each step of processing images.

Conclusion

My Jaccard index score computed in Milestone 2 was sufficient enough as a valid submission, but I do believe some changes I made allowed me to increase accuracy. There are definitely many more changes I wanted to implement, such as a weighted fusion strategy, but was unsuccessful. I achieved my personal high Dice score of 71% and am happy with my results.

Ethical Considerations

Medical imaging is the process or technique of creating visual representations of various parts of a human body for both diagnostic, treatment, and clinical purposes. As with any other technologies used in medical research, it is subject to general ethical principles in order to protect the patients' rights.

Medical images of a patient is considered as a patient's personal data; therefore, it is subject to laws that govern confidentiality and patient privacy. This means that the patient and the patient alone as the right to make decisions about how personal information is shared, and the professionals who have access to such information are obligated to hold a patient's medical information in confidence. This also means that with electronic health record increasingly being implemented in many developing countries, it is even more imperative that additional security measures be taken to prevent security breaches and to ensure that only authorized users have access to such electronic information [2].

General ethical principles need to take place when conducting research that involve medical imaging. The researchers need to design clinical investigation that is ethically just and receive informed consent from the research participants. This application of ethical guidance to conduct research involving vulnerable participants can be difficult: for instance, critically ill patients in intensive care units are typically unable to make decision regarding participation. These individuals may be susceptible to the risks of research participation, and this must be taken into account when debating whether a research design is ethically just [3].

Finally, there are new ethical concerns to address when emerging technologies such as artificial intelligence is used in medical imaging. When medical imaging technologists makes a mistake, we want to determine whether the mistake reflects malpractice or malintent. Consequently, if an algorithm contributes to malpractice or is inaccurate, radiologists need to be able to determine the source of error. Additionally, prior to using new models on patient data, the model must be comprehensible to outside viewers [1].

References

[1] J. Raymond Geis et al. "Ethics of Artificial Intelligence in Radiology: Summary of the Joint European and North American Multisociety Statement". In: *Radiology* 293.2 (Oct. 2019). Publisher: Radiological Society of North America, pp. 436–440. ISSN: 0033-8419. DOI: `10.1148/radiol.2019191586`. URL: `https://pubs.rsna.org/doi/10.1148/radiol.2019191586` (visited on 05/15/2020).

[2] Fouzia Ozair et al. "Ethical issues in electronic health records: A general overview". In: *Perspectives in Clinical Research* 6.2 (), pp. 73–76. URL: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4394583/` (visited on 05/15/2020).

[3] Charles Weijer et al. "Ethical considerations in functional magnetic resonance imaging research in acutely comatose patients". In: *Brain* 139.1 (Jan. 2016), pp. 292–299. URL: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5839553/` (visited on 05/15/2020).