Image-based Diagnosis of Type I Chiari Malformation

Elle Buser[*], Emma Hart[†], Ben Huenemann[‡]

July 31, 2021

Abstract

In this study, an atlas-based image registration method and a machine learning method were created to automate the segmentation of the brain stem and cerebellum in a given Magnetic Resonance Imaging (MRI) scan. After identifying these regions of interest, a biomarker, to predict the class of a patient as having or not having Type I Chiari Malformation (CMI), is estimated from Displacement Encoding with Stimulated Echoes (DENSE) MR imaging. In CMI patients, displacement of the brain stem and cerebellum throughout the cardiac cycle is significantly higher than in controls, and information from these regions encoded in associated DENSE MR imaging may provide a more effective basis for diagnosis than current anatomical methods [10]. The process of manually segmenting the cerebellum and brain stem from anatomical MR images to identify these regions of difference is time consuming and expensive, but both the machine learning and atlas-based image registration method yield promising results for automatic segmentation. It was found that...

1 Introduction and Problem Outline

Chiari Malformation type I (CMI) is a condition affecting the skull and brain that is estimated to affect slightly less than 1 in 1,000 people [1]. Although most of these cases are asymptomatic, some can lead to extreme pain and require immediate surgery. In such cases CMI can be accompanied by severe head and neck pain, headaches, dizziness, and impaired vision [3].

Though CMI is typically diagnosed when an individual's cerebellar tonsils (TP) extend more than 5mm below the foramen magnum, many experts recommend rejecting this criterion due to a weak correlation between CMI and TP [3].

A recent study also found that there is a significant increase in neural tissue movement in CMI patients compared to controls, especially in the brain stem and cerebellum [10]. This was found using a dynamic MRI technique called Displacement Encoding with Stimulated Echoes (DENSE): a type of MR imaging that records information about movement in the brain and that is sensitive enough to capture micrometer scale deformations that occur with each heartbeat. Measurements related to the displacement in the brain stem and cerebellum could perhaps become part of a new, more reliable basis for the diagnosis of CMI.

The purpose of this study is to create image segmentation algorithms that automatically identify these regions and conduct further analysis with DENSE MRI data. One of the most limiting steps

[*]Department of Mathematics and Statistics, University of Wyoming, Laramie, WY, USA
[†]Department of Mathematics, Colgate University, Hamilton, NY, USA
[‡]Department of Mathematics, University of Utah, Salt Lake City, UT, USA

in diagnosis based on this new criterion is the identification of the brain stem and cerebellum in an MR image. Currently, this segmentation is done manually: a process both expensive and time consuming [10]. This study compares two distinct methods to automate the segmentation of the brain stem and cerebellum: an atlas-based image registration approach, and a convolutional neural network approach.

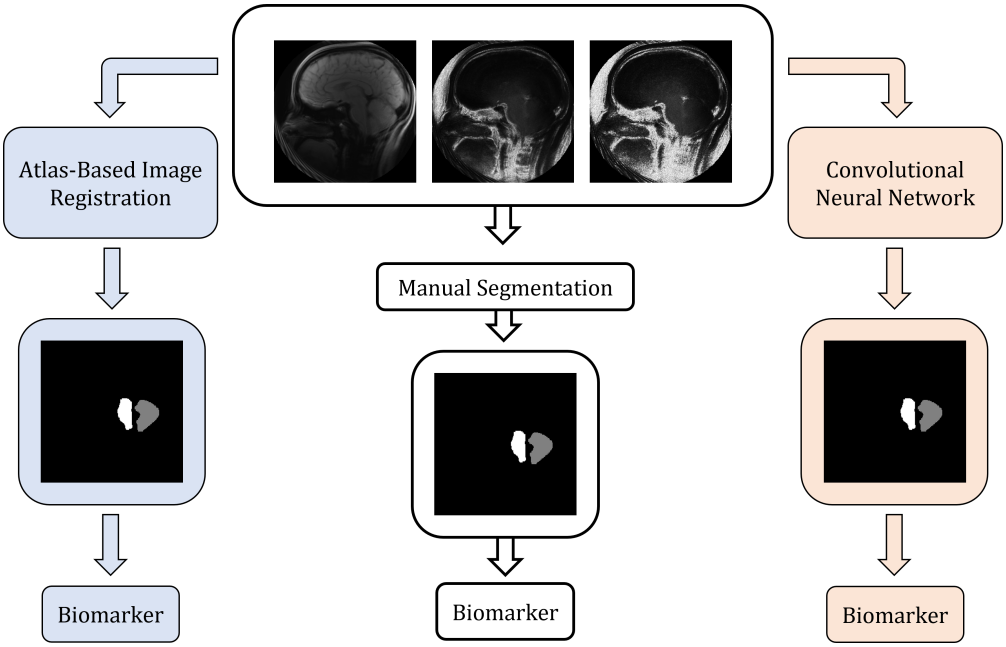

Figure 1: An overview of the setup of this study. The current method of diagnosing Chiari from brain displacement data (center white path) involves a radiologist manually drawing a mask that identifies the brain stem and cerebellum, and computing a biomarker using displacement data from just those regions. An atlas-based algorithm (left blue path) and a machine learning (right red path) approach were developed to automatically segment the cerebellum and brain stem given DENSE MR imaging. Results were compared between the two methods methods and to the results produced from the manual process, that is treated as truth.

The data set that is used to automate this process includes DENSE MR images and masks identifying the cerebellum and brain stem for patients from Emory University Hospital's Department Of Radiology and Imaging Science; a portion have been used in the creation of the machine learning and atlas-based segmentation methods and the rest have been reserved for testing the validity and generalizability of these methods.

In investigating atlas-based registration, our study builds on methods from a MATLAB toolbox called FAIR [8]. Image registration from this toolbox relies on the idea of a template image and a reference image. In the case of segmenting MR images, the reference is the magnitude image that needs segmentation and the template is some magnitude image that has already been segmented. The main idea of this approach is to find some transformation to apply to the template so that it approximately fits the reference.

Another approach is found in convolutional neural networks (CNNs), a category of machine learning. Using MR images from the data set as input, a model is trained in a supervised fashion to output a segmentation with three classes: brain stem, cerebellum, and background. In this study,

a CNN made for biomedical imaging called U-Net[12] is implemented using PyTorch[11], a machine learning library in Python. This network, combined with an optimizer and loss function, loops over a section of the data set, called the training set, to produce an output mask by classifying each pixel as background, brain stem, or cerebellum. This output is then compared to the corresponding known mask. With each iteration, the CNN learns from this comparison and updates its parameters to produce more accurate results. The model can then be tested on images in the data set, not used in training, as a means to see how well it segments new MR images.

The results found are promising; the implementation of both atlas-based registration and machine learning methods are able to segment MR images reasonably well and produce biomarkers very close to manually produced ones.

2 Methods

This section begins with a detailed description of the dataset used, then an introduction to the measures used to evaluate both segmentation and diagnosis. This is followed first by an explanation of the atlas-based segmentation method used, then by an explanation of the machine learning (convolutional neural network) approach.

2.1 Data

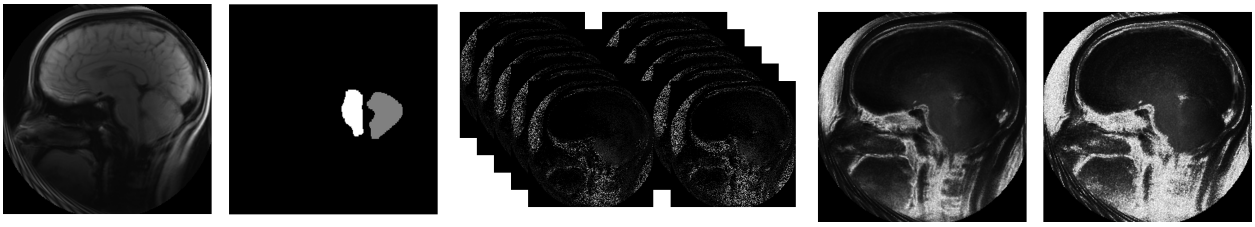

Figure 2: From left to right, an example of one patient's (1) magnitude MRI image; (2) mask; (3) DENSE images representing one cardiac gate each; (4) temporal mean DENSE image; and (5) temporal peak DENSE image.

This study uses 256x256 pixel DENSE MR image data and associated masks for 63 patients, 51 of which have been used in the creation of the machine learning and atlas-based image registration methods and 12 of which have been reserved for testing these methods. As is typical in MR imaging, the DENSE MRI data is stored as a complex-valued array.

The images typically visualized in MRI are based on magnitude, where the grey value of each pixel is created from the magnitude of the complex value in that pixel's position (shown in **Figure 2.1**). The color of each of the pixels in these magnitude images is controlled by the chemical makeup of the material that exists in that position in the brain, and therefore represent the brain's anatomy. Brightness and contrast levels varied across subjects in a way that made segmentation more difficult, so the images' contrast and brightness levels were adjusted using a default function in MATLAB. This function (histeq) normalizes the histograms of gray values associated with each image by linearizing the cumulative distribution function and helps create a more uniform set of images. **Figure 3** compares a sample of original and normalized images. For more explanation of this process, see [5].

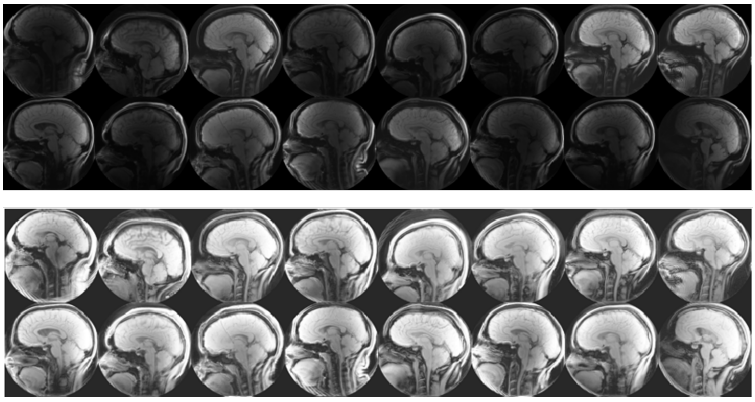

Figure 3: A sample of sixteen magnitude images, original (top) and normalized using histogram normalization (bottom).

DENSE imaging encodes displacement information in the phase of each complex number corresponding to the intensity of movement associated with each pixel position. For each subject many DENSE MRI scans were created corresponding to gates that occur over the patient's cardiac cycle (shown in **Figure 2.3**). The number of DENSE images varies per patient based on their heartbeat. The visual interpretation of these phase images is less intuitive than the magnitude images, mapping movement rather than anatomy.

Generally, areas of large movement in brain tissue appear lighter than those of small movement, but in areas of especially high movement – especially outside the brain where cerebrosprinal fluid (CSF) moves much more quickly than the micrometer deformations in the brain tissue – phase wrapping occurs. Because there is no distinction between the gray scale color of a pixel stored with a phase ϕ and $\phi + 2\pi$, for example, they appear the same. These regions with high movement appear as random noise because the movement is beyond the smaller scale for which the phase encoding was calibrated. For more on phase wrapping, see [7].

In this study, displacement data from the many scans were summarized into two types of DENSE image data: a mean DENSE image, showing the mean value per pixel over time (**Figure 2.4**), and a peak DENSE image, showing the maximum value per pixel over time (**Figure 2.5**), following the practices developed in [10].

This representation of the DENSE MRI data over time helps reduce the regions of random noise and summarize over the patient's whole cardiac cycle. For more discussion of this choice and of other ways to represent this data, see [10].

The masks associated with the magnitude images classify each pixel as being part of the background, the cerebellum, or the brain stem, and were drawn by researchers in Emory's Department of Radiology and Imaging Science (shown in **Figure 2.2**).

2.2 Notation

The machine learning approach and the atlas based approach, as described in this paper, are based on different conceptions of the MR image data.

In perhaps a more generally intuitive way, the machine learning approach treats the images as discrete sets of data. The input, $I \in \mathbb{R}^{256\times256\times3}$, is made of 3 channels: the normalized magnitude, peak DENSE, and mean DENSE images. Each a 256x256 matrix with entries corresponding to the

gray values.

The atlas-based approach treats an image as an interpolated function, $\mathcal{I} : \Omega \to \mathbb{R}$ where the domain $\Omega \subseteq \mathbb{R}^2$ corresponds to coordinates of pixels in the image grid. The range of this function, called on a specific 256x256 grid, creates the discrete 256x256 pixel images. To distinguish these understandings, calligraphic letters have been used to denote continuous functions while regular script denotes discrete. This notation, and the notation of other equations related to the atlas-based method, are formatted in the style of FAIR [8]. Refer to this MATLAB toolbox and the associated textbook for a more complete discussion of image registration.

These two understandings of images lend also themselves to two conceptions of the outputted masks. In machine learning, the output mask is a discrete 256x256 probability matrix which predicts a class $k \in \{0, 1, 2\}$ for each pixel where 0 corresponds to background, 1 to cerebellum, and 2 to brain stem.

Analogously, in the continuous function sense, let C, B, and T be the sets representing the cerebellum, brain stem, and the total union between them respectively. Then let $\chi_C, \chi_B : \Omega \to \{0, 1\}$ be the characteristic functions for the cerebellum and brain stem respectively. Masks that display these regions will hence be defined as $\mathcal{M} : \Omega \to \{0, 1, 2\}$ where

$$\mathcal{M}(x) = \chi_C(x) + 2\chi_B(x). \tag{1}$$

In addition to this, define $\chi_T : \Omega \to \{0, 1\}$ where $\chi_T = \chi_{C \cup B}$ to be the characteristic function of T for convenience.

Using this notation, the template mask will be referred to as $\mathcal{M}_\mathcal{T}$ and the reference mask as $\mathcal{M}_\mathcal{R}$.

2.3 Evaluation Measures

This project maintains two interrelated goals: to accurately segment the cerebellum and brain stem in a given magnitude MR image, and to produce a biomarker from that segmentation that can help diagnose CMI from DENSE MR imaging. Radiologist drawn masks that identify what displacement data to average in the biomarker biomarkers are treated as the gold standard. It may be that these algorithmic methods can produce better, more consistent masks, but that lies outside the scope of this study. To balance the dual concerns of producing masks and biomarkers that match manually produced ones, Dice similarity indices and biomarkers were considered at each step.

2.3.1 Dice Index

To evaluate the success of both the machine learning and atlas-based image registration as segmentation methods, we use the Dice similarity index. Given two subsets of Ω, A and B, the Dice index is computed as

$$D_{\text{Dice}}[A, B] = \text{Dice}[\chi_\text{A}, \chi_\text{B}] = 2 \int_\Omega \frac{\chi_\text{A}(\text{x})\chi_\text{B}(\text{x})}{\chi_\text{A}(\text{x}) + \chi_\text{B}(\text{x})} \text{dx}, \tag{2}$$

where $\chi_A(x), \chi_B(x) : \Omega \to \{0, 1\}$ represent the characteristic functions for the sets A and B, respectively. This results in a measure that ranges from 0, in the case of two disjoint regions, to 1, in the case of identical regions [9].

We use this measure primarily to evaluate mask similarity and determine if algorithmically predicted masks reasonably match radiologist drawn masks. Our masks define three sets of special

interest: the set of pixels identified as the brain stem, the set identified as the cerebellum, and the set identified as background. Computing the Dice similarity for the cerebellum and brain stem, especially, between a true mask and predicted mask for the same brain quantifies how well the segmentation method is working; in both the machine learning and atlas-based approached, algorithms and parameters were chosen to maximize this number.

2.3.2 Biomarkers

Diagnosis is predicted with spatially averaged, temporally maximized biomarkers for both the brain stem and cerebellum. For example, once a segmentation is identified for the brain stem, the value of each pixel classified as a part of this region from the peak displacement DENSE image data is averaged (or analogously, for the cerebellum). When we refer to biomarkers throughout this paper, we refer to these displacement averages across the brain stem and cerebellum. Building on the results of [10], the biomarkers produced from manual segmentation will be treated as the truth, and our aim is to match these with our automated approaches.

2.4 Atlas-Based

This section includes discussion of the way a template image is chosen, the parameters and functions used in image registration (associated with the toolbox FAIR [8]), and examples of how the method comes together.

The development of the atlas based image registration method involved primarily working with magnitude images. The final tool uses 51 of the magnitude images in a bank of templates to which a new reference can be compared to generate a mask for the template. In development of the method, references with known masks were used and similarity measures between known and predicted masks were calculated to inform the methods and parameters chosen.

Atlas-based segmentation, in FAIR language, refers to a template and a reference image as functions that map coordinates to gray-scale values. Then the template is $\mathcal{T} : \Omega \to \mathbb{R}$ and the reference is $\mathcal{R} : \Omega \to \mathbb{R}$ where $\Omega \subseteq \mathbb{R}^2$ corresponds to coordinates of pixels in the image grid. This means that the goal of the image registration is to find some transformation $y : \Omega \to \mathbb{R}^2$ such that

$$\mathcal{T}[y](x) \approx \mathcal{R}(x), \quad \text{for all} \quad x \in \Omega. \tag{3}$$

Note that $\mathcal{T}[y](x) = \mathcal{T}(y(x))$ is written this way to emphasize that the function y is an input as well.

The main problem that FAIR solves is finding that function y by minimizing the joint objective functional

$$\mathcal{J}_{\text{atlas}}[y] = D_{\text{SSD}}[\mathcal{T}[y], \mathcal{R}] + \alpha \mathcal{S}[y], \tag{4}$$

where D_{SSD} is the distance measure, α is the regularization parameter, and $\mathcal{S}$ is the regularizing functional. These will be described in further detail later in this section.

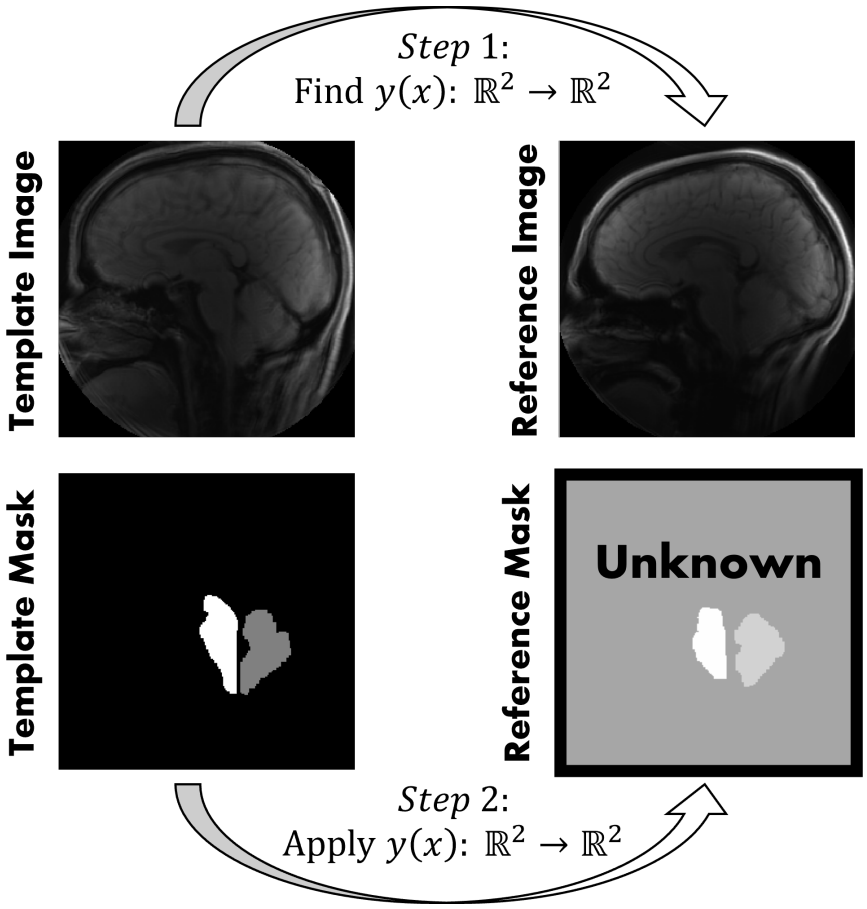

Figure 4: Finding the transformation between template and reference and then applying that transformation to the masks

With this goal in mind, a program was setup with the reference image as input. This program is separated into four parts: find an appropriate template to compare to, setup the image registration parameters, perform the image registration, and then analyze the results.

2.4.1 Choosing $\mathcal{T}$

Perhaps the most critical factor in the success of the segmentation is picking the most appropriate template image. Choosing a template that most closely matches the overall shape and brightness of the reference helps the registration perform its best. Our program was created to loop through all of the other MR images and pick the one with the lowest Sum of Squares Distance (SSD).

In **Figure 5**, the image registration was run with one reference against all template images to see how well the SSD works as an initial prediction. Note that a lower SSD is desirable since that means that the two images are more similar. Drawing a least squares line shows a rough correlation between the SSD and how well the image registration performed. It is far from a linear correlation, but at least the top SSD correctly indicated the best similarity in this case. One option to help with this inaccuracy is to take the top 20 patients and average the result together. This way it is less dependent on the outliers.

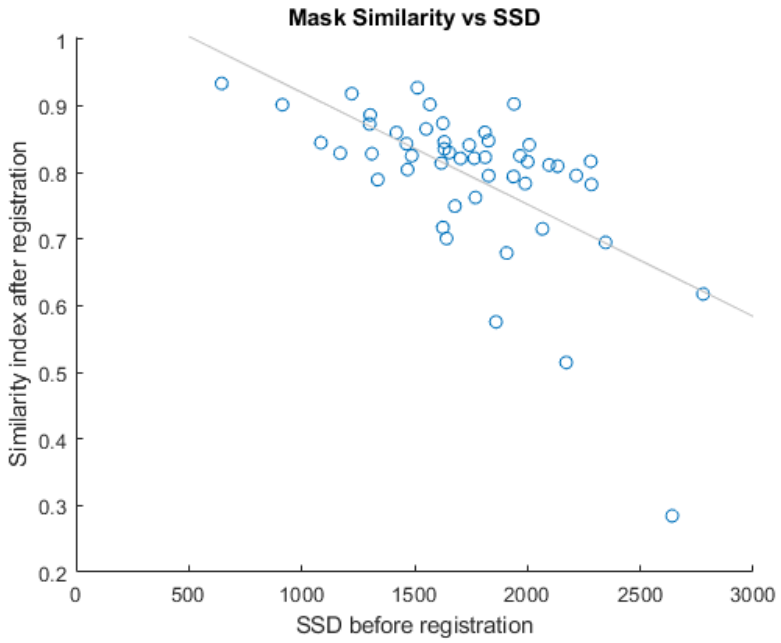

Figure 5: Scatter plot of the SSD between an example reference image and all template images. The mask similarity is measured using the Dice index and a least squares line is drawn for analysis. The correlation coefficient R, R^2, and p-value were found to be -0.6152, 0.3785, and 1.996e-6, respectively. ? DO THESE SEEM REASONABLE? I WORRY THAT I AM FORGETTING NUANCES OF THESE MEASURES, BUT I USED [R,P] = corrcoef()

2.4.2 Setting Up the Registration

Once the template is chosen, the program begins image registration. The most fundamental design and parameter choices of this registration are explained below:

Pre-registration: When performing image registration on two images, the first step is to apply a rough transformation to align them. One natural instinct in this regard is to translate and rotate the template. This would be a rigid transformation. There are certain advantages to keeping it simple like this, but more flexibility can be added by making it an affine transformation — which allows for additional linear transformations such as shearing and scaling. Our study made use of affine transformations because of this flexibility.

In more mathematical terms, this is where the transformation $y : \Omega \to \mathbb{R}^2$ is found. Using affine transformations,

$$y(x) = \begin{bmatrix} w_1 & w_2 \\ w_4 & w_5 \end{bmatrix} x + \begin{bmatrix} w_3 \\ w_6 \end{bmatrix} \tag{5}$$

where w_i are weights that parameterize the transformation.

Distance: After applying an affine transformation, the algorithm tries to minimize the distance measure. There are many options for how to pick this distance measure, but the most common is

Sum of Squares Distance (SSD) where you add the square of each pixel value:

$$D_{\text{SSD}}[\mathcal{T}[y], \mathcal{R}] = \frac{1}{2}\int_{\Omega}(\mathcal{T}[y](x) - \mathcal{R}(x))^2 dx. \tag{6}$$

The SSD measure penalizes different brightness levels. As a result, altering the contrast of the images through normalization can make the SSD measure perform much better.

An alternative to SSD is the Normalized Gradient Field (NGF) measure. The NGF measure is based on the changes in intensity that are found in an image. Because of this, minimizing the NGF measure results in aligning the edges of an image. This also means that an image and it's inverse would produce an NGF of 0 because all of the edge values are the same (whereas SSD would find them vastly different).

Applying the NGF measure to the MR images, the main problem was that the edges of the brain stem and the cerebellum are not very well defined. This means that the image registration focused too much on aligning the skull and other parts of the brain that are less important. It may be possible to alter the image to make the NGF more extreme in that region, but SSD proved to be adequate so that was used instead.

Regularizer: When only focusing on the distance measure, the image registration program can come up with some unrealistic scenarios. Sometimes it compresses the template image to fit within some small region of the reference. Other times it may create visible folds in the image. One way to counteract this is by introducing a regularizer.

The main regularizers that FAIR supports are elastic and hyperelastic [4]. Both of these support the same idea but vary in severity. An elastic regularizer will try to keep the image registration close to the original orientation by introducing linear strain to any change. A hyperelastic regularizer does this with a nonlinear strain equation so that it penalizes greater deviations.

Because of this, the elastic regularizer is only accurate for smaller displacements. For anything larger, the elastic regularizer risks becoming inaccurate and causing folding in the transformation grid. A hyperelastic regularizer, however, would be stronger and prevent more of the folding. Either of these regularizers would work for the Chiari data, but hyperelastic was used since seems to remove more of the edge cases.

Regularization Parameter: When deciding what transformation to make, there is a trade-off between whether to minimize the distance or follow the regularizer. This parameter controls the balance between allowing for displacement between the original and transformed image, to better the distance measure, or to penalizing displacement, to preserve the general shape of the image throughout the registration [8]. Note that in **Equation 4**, α corresponds to the regularization parameter. This coefficient was chosen to be 500 in our final algorithm, as is a common default choice for this kind of problem, but could be further refined for better results.

Multilevel Optimization: As common in FAIR, Gauss Newton optimization was used with this joint objective function to generate a transformation from template to reference. Because this objective function for a 256x256 pixel image contains many local minima, and the convergence of this Gauss Newton to an appropriate minimum is very dependent on an initial guess, a multilevel approach was used. This involves creating coarser representations of our images, by changing the grid for which our interpolated image function is called, that contain fewer pixels: in our case,

three additional representations, with 32x32, 64x64, and 128x128 pixels each. The optimization is applied in a way that moves up through these representations, applying optimization on the coarsest first and using the found transformation as the initial guess for the image at the next level.

2.4.3 Examples

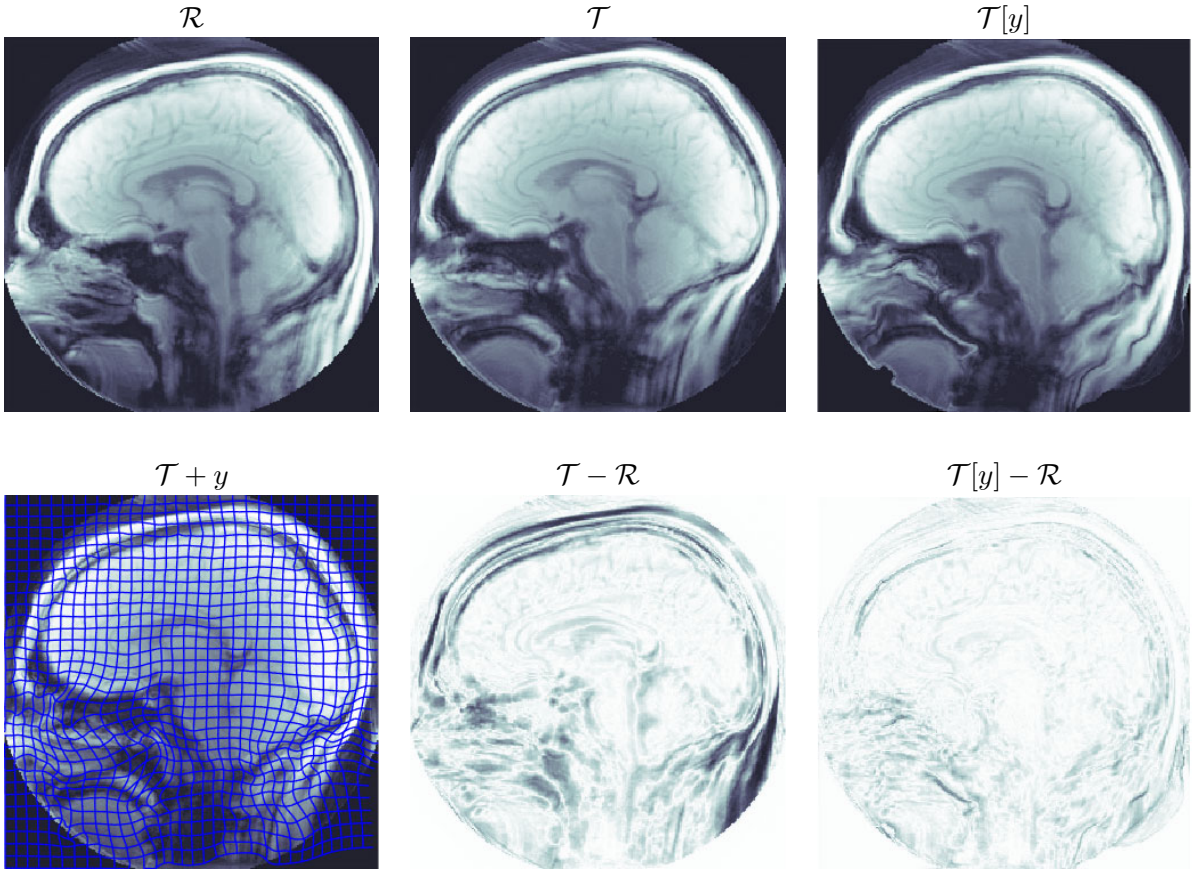

Figure 6: Image registration run with the parameters described above. The first row images from left to right are the reference, template, and transformed template. On the second row, there is the template with the transformation grid, original difference in SSD, and transformed difference in SSD. The runtime for this example was around 40 seconds

These parameters control how a transformation is found in each registration. **Figure 6** shows one of the best performing cases in terms of segmentation. In this example, the SSD-based template search algorithm was able to find an image with a close SSD that also worked well in the registration. The main points of interest in this figure are in the second row.

Looking at this particular transformation grid, most of the deformation happened around the front end of the skull. This is also seen in the SSD difference between the template and the reference (row 2 column 2). The darker regions around the front of the skull show the regions that were originally unaligned. After the transformation, the SSD was minimized resulting in the transformed

SSD difference image (row 2 column 3). The fact that this image is mostly white shows that the transformation generally performed well.

Once the transformation is found, the next step is to apply the same transformation to the mask of the template image. This way the transformed template mask can become an indicator for where the reference mask might be. In **Figure 7**, the original template mask and the transformed template mask are compared to the manually segmented reference mask to see how well the registration did.

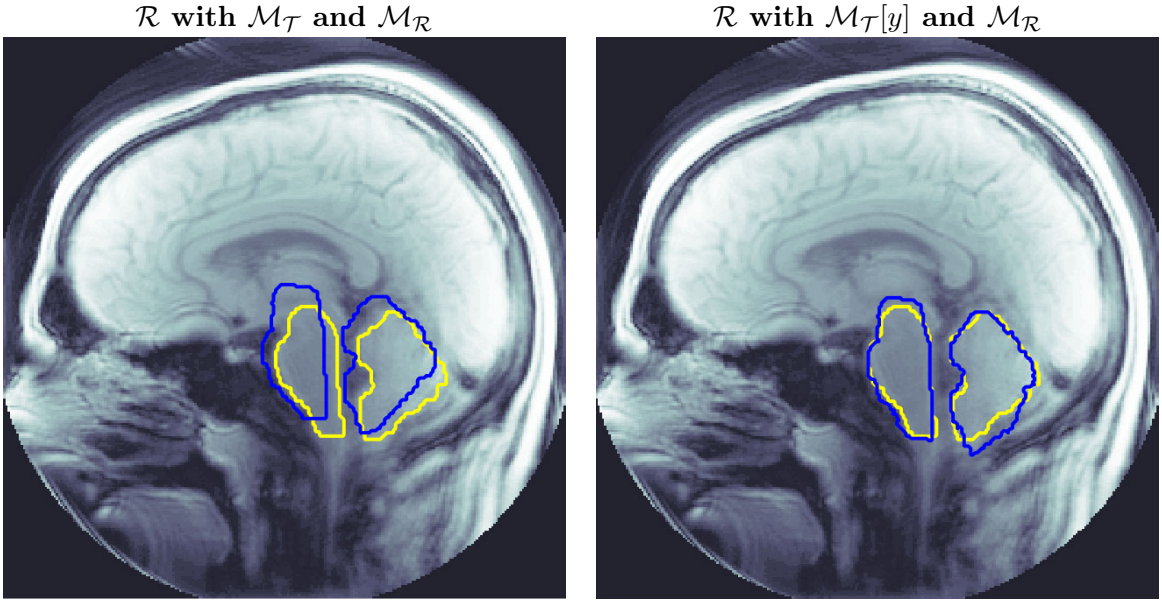

Figure 7: The first image is the reference and reference mask (yellow) with the template mask on top (blue); the second image is the same except with the blue outline representing the transformed template mask. The transformation, y, applied to the template mask is the same as **Figure 6** and the dice similarities are in **Table 1**.

To evaluate this numerically, the Dice similarities were calculated for each region of the masks as shown in **Table 1**. In this case, most of the difference came from the cerebellum which showed around a 9% increase.

In order to reduce the significance of the edge cases for how well the registration performs, another program was created to average the registration across multiple templates. This is aimed at lessening the importance of choosing the template by it's SSD. Then the choice in mask is less dependent on the quality of any individual mask.

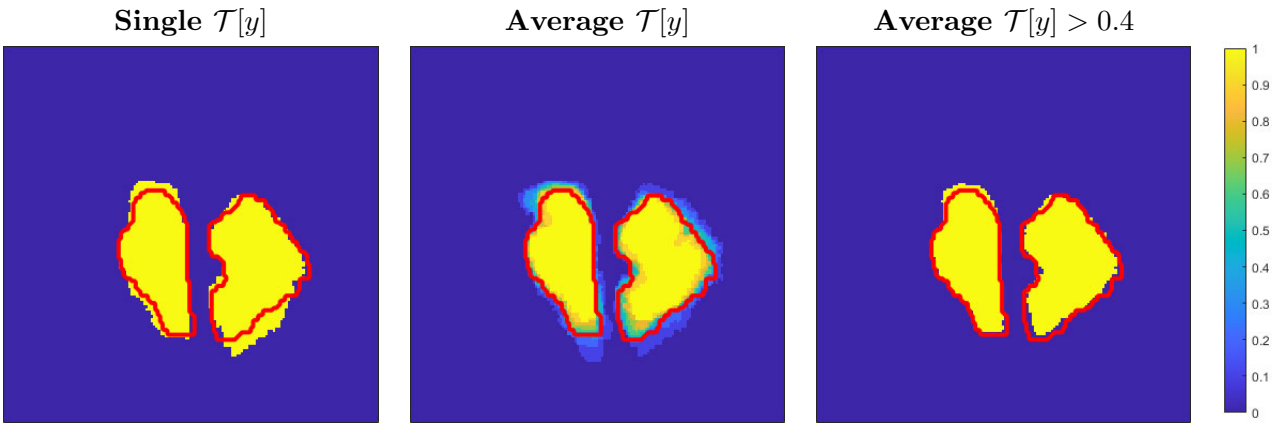

Figure 8: The left shows the single registration from before, the middle is the average of all the pixels with $n = 15$ patients, and the right is the same figure after using a threshold of 0.6. This is the same mask as **Figure 6** and the similarity is shown under the average column of **Table 1**.

To be more specific, the program inputs n as the number of patients to average. Then it calculates the SSD for all patients in the training set and takes the top n of them. These images are then summed together and divided by n to get the average mask. Finally, a threshold is used to convert this average transformed mask into a binary mask again. An example of both the average transformed mask and the binary version is shown in **Figure 8**. To see how we chose the values of n and threshold, see the beginning of section 3.1.

In **Table 1**, this average method is shown to be slightly better than the single registration. The reason for this seems to be that the original mask that was selected based on SSD seemed to overshoot in certain parts. Averaging this with other patients lessened the severity of this by eliminating outliers.

	Original	Transformed	Average
Brain stem	0.734	0.896	0.928
Cerebellum	0.659	0.910	0.936
Full Mask	0.701	0.902	0.932

Table 1: Dice similarities of the masks from **Figure 7** before and after the transformation along with the average version from **Figure 8**.

Although this example shows the strengths of image registration with our chosen parameters, there are other patients in the data set where the registration is weaker. One of these is shown in **Figure 9**, **Figure 10**, and **Table 2**. The single SSD that was chosen was better than the average so including those other samples reduced the final similarity.

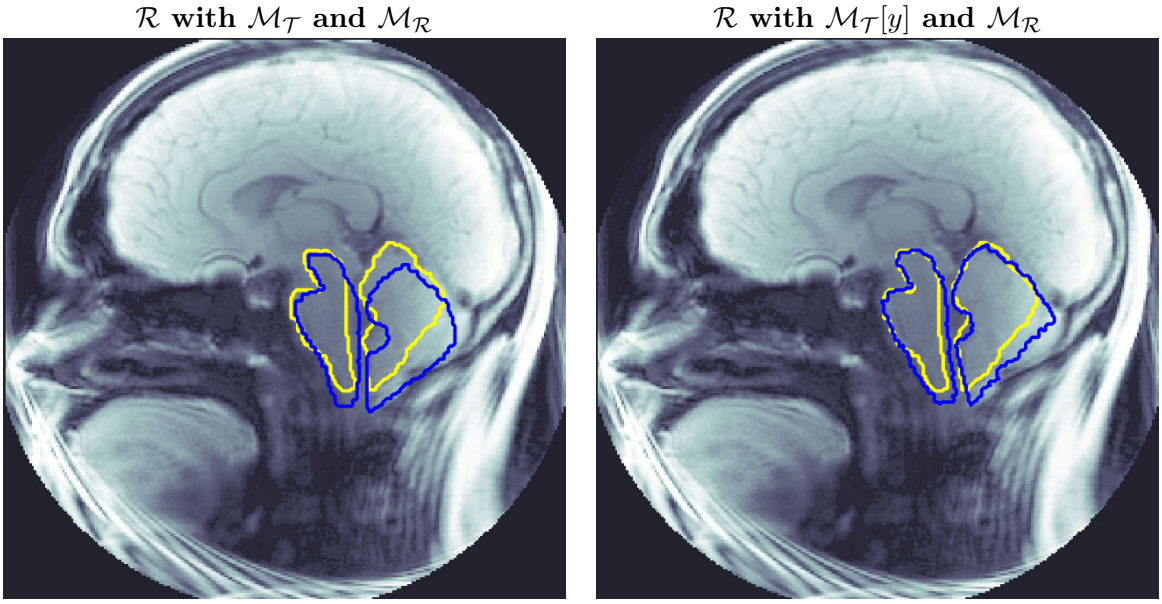

Figure 9: The first image is the reference and reference mask (yellow) with the template mask on top (blue); the second image is the same except with the blue outline representing the transformed template mask. The dice similarities are in **Table 1**.

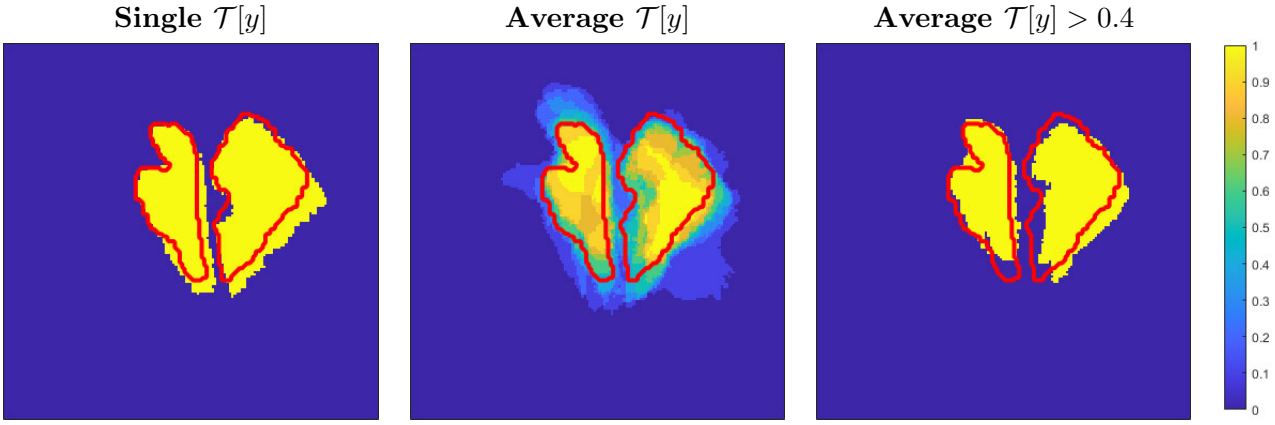

Figure 10: The left shows the single registration from before, the middle is the average of all the pixels with $n = 15$ patients, and the right is the same figure after using a threshold of 0.6. This is the same mask as **Figure 9** and the similarity is shown under the average column of **Table 2**.

	Original	Transformed	Average
Brain stem	0.675	0.845	0.805
Cerebellum	0.805	0.855	0.850
Full Mask	0.728	0.849	0.824

Table 2: Dice similarities of the masks from **Figure 9** before and after the transformation. This also includes the values calculated from the average version of the program that are explained later.

This example chose a decent initial SSD, so the average shown in **Table 2** did not perform well as. However, the average is at most 4% worse similarity in this example so the biomarker results were mostly unaffected. Refer to the results section of the paper for more details.

2.5 Machine Learning

The overarching goal of this approach is to find a relationship between the input DENSE imaging data $I \in \mathbb{R}^{256\times256\times3}$ and the known target segmentation masks $M \in \{0,1,2\}^{256\times256}$ using a subset $\{(I_t, M_t)\}_{t=1,2,\ldots,41}$ containing the first 41 images and masks in $\{(I, M)\}$, defined as training data. Here, M consist of the classes k as defined in Section 2.2.

By training a convolutional neural network (CNN) on $\{(I_t, M_t)\}$, which inputs I and outputs a probability matrix P,

$$\{P \in \mathbb{R}^{256\times256\times3} : P_{i,j,k} \geq 0 \ \forall \ i,j,k \text{ and } \sum_{k=0}^{2} P_{i,j,k} = 1\}, \tag{7}$$

the relationship found between I and M can be represented by a parameterized function

$$f_\theta : \mathbb{R}^{256\times256\times3} \to P, \tag{8}$$

where θ represents the weights in the CNN and the value of $P_{i,j,k}$ is the predicted probability the pixel at position i,j belongs to class k.

To provide an in-depth look at how the model used in this study was trained, basics behind the CNN of choice, and how optimization unfolds are each discussed.

2.5.1 Training the Model

In order to get a successful model, we want the mask from the model prediction P and corresponding target M to be as similar as possible. We can numerically calculate the error between the two using a loss function. This is done by maximizing the pixel probability at the class corresponding to the value of the known mask at the same pixel position. In other words, our goal is to maximize

$$[f_\theta(I)]_{i,j,M_{i,j}} \quad \forall \ i,j, \tag{9}$$

which is equivalent to minimizing

$$-\log\left([f_\theta(I)]_{i,j,M_{i,j}}\right) \quad \forall \ i,j. \tag{10}$$

Summing **Equation 10** above over all pixels i,j for each pair in some set $\{(I_e, M_e)\}_{e=1,2,\ldots,E}$, where E is the size of the set, results in the function

$$\mathcal{J}_{CNN}(\theta) = \min_\theta \frac{1}{256^2} \sum_{e=1}^{E} \sum_{i,j=1}^{256} -\log\left([f_\theta(I)]_{i,j,M_{i,j}}\right), \tag{11}$$

which computes the cross-entropy loss between $[f_\theta(I_E)]_{i,j,:}$ and the probability distribution defined by the standard basis vector $[M_E]_{i,j}$. In this study, the cross-entropy loss compares the mask prediction to the corresponding known mask to calculate how far the model prediction is from the expected output.

Inputting $f_\theta(I)$ and M in the loss function is a forward pass, this is done on both training and validation data. By executing a backwards pass on the calculated loss, gradients can be calculated and used by the optimizer to update model parameters. A backward pass is only executed with respect to the training data.

2.5.2 U-Net: A Convolutional Neural Network

CNNs are a type of neural network which process data that have a "grid like topology," such as image data (a 2D grid of pixels), and utilizes a convolution operator.

Since the data in this study consists of multidimensional arrays of pixels and the goal is to correctly predict which class each pixel belongs to, a CNN is the optimal network to use. While many deep networks require thousands of images, a specific CNN designed for biomedical image segmentation called U-Net can be trained on smaller data sets, such as the one in this study, and will still produce precise probabilities [12]. U-Net consist of 2 main sections: the contracting path on the left side (encoder) and the expansive path on the right side (decoder) as show in **Figure 11** below.

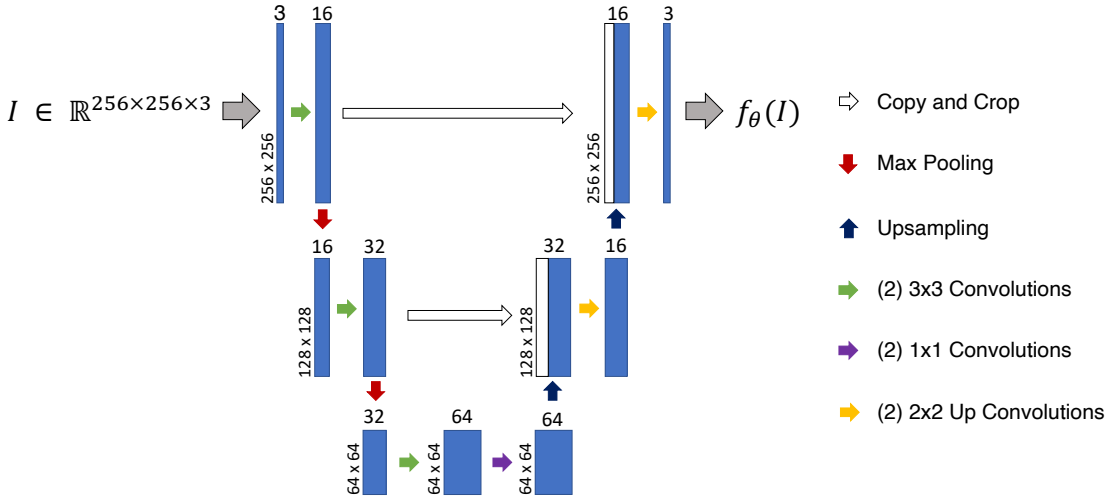

Figure 11: U-Net architecture used in our study based on the figure provided in U-Net: Convolutional Networks for Biomedical Image Segmentation [12]. Here numbers placed on top of the boxes representing image dimension correspond to number of channels.

In the first layer of the contracting path, a 3x3 convolution is applied to each channel increasing the number of total feature channels. This is followed by a max pooling operation to break down the image resolution by decreasing image size to better identify features. A similar convolution structure is applied at each layer doubling the number feature channels. This leads to the bottom most layer where 1x1 convolutions are used maintaining number of channels. Building the image back up, a decoder, or expansive path, increases the image size by replacing the max pooling with upsampling operators. In the same formation of the contracting path, convolutions are applied only these, called up convolutions, are 2x2 and halve the number of feature channels. During the process, high resolution features found during encoding are 'copied and cropped' and combined with the decoded output increasing dimension size and providing information that will be used in the convolution layer to produce more accurate outputs.

While the first version of the network we used only had 1 input channel, 3 were used in the final model: the magnitude, mean DENSE, and peak DENSE images, $I \in \mathbb{R}^{256\times256\times3}$. As the input images were run through the U-Net, the network used information from each channel to assign each pixel a probability for each class. These values are used to calculate the loss and to find the final predicted segmentation.

2.5.3 lBFGS Optimization

A limited-memory Broyden–Fletcher–Goldfarb–Shanno (lBFGS) algorithm from the PyTorch library [11] was used to optimize the overall loss function, shown previously in **Equation 11**. lBFGS is a Quasi-Newton method which uses a line search to automatically find the optimal learning rate ϵ [6] which satisfies the strong Wolfe conditions set in the algorithms parameters. This allows for the objective function value to be lowered while keeping ϵ from becoming too small.

The lBFGS algorithm uses the gradient calculated from the loss function and the learning rate found in the line search to update the model weights. The images were then, once again, ran through the U-Net in hopes of a better segmentation.

2.5.4 Experimental Setup

In this method, of the 51 images in the data set, 41 ($\sim$80%) were used for training and the other 10 ($\sim$20%) were set aside for validation. The purpose of the validation data is test how effective the model is on "new" input images not used to update parameters.

The U-Net was setup using code provided by Aman Arora [2]. The network was trained by looping a sequence of using the training images I_t as input to the model, computing the loss between $f_\theta(I_t)$ and M_t, and updating the model with the optimizer for a set number of iterations. In each iteration, the updated model was run with the training data and validation data, separately, to calculate each loss without using the information to change parameters.

Initially, the input data I consisted only of a single channel, the normalized magnitude MR images. This was changed to have 3 channels — one for each of the DENSE images defined above. To compare the cost efficiency of training the model using each path, it was first run using single channel and then again with 3 channels; both with 500 iterations. As a second comparison, the network was trained until the overall minimum validation loss was reached for each channel option. Both experiments used the training and validation data loss and Dice index for each class as a measure of model success.

Using the model which produced the smallest validation loss, segmentations of the validation data were used to find the biomarkers from the phase DENSE images. These were compared to the corresponding biomarkers produced in the atlas-based method and those from the provided masks in M. This was repeated using the test data previously set aside.

3 Results

This section begins with results from the processes of creating the atlas-based and machine learning methods; overall results on the final testing set follow.

3.1 Atlas-Based Image Registration Results

When creating the averaging program, two main parameters were chosen arbitrarily: n and threshold. The n refers to the number of template images to average and the threshold refers to which values should be counted when converting the average image to binary. In order to decide on a more definitive value for these parameters, a program was made to iterate through different possible values and find which performed the best.

This was run over two metrics for how well it performed: Dice similarity and biomarker difference. Both of these metrics compare the atlas-based generated mask (average program) to the manually segmented mask and then average together all of the patients. These produced approximately the same results since the biomarker difference depends on how similar the masks are. A plot showing averaged Dice indices and absolute biomarker error over n and the threshold can be seen in **Figure 12**.

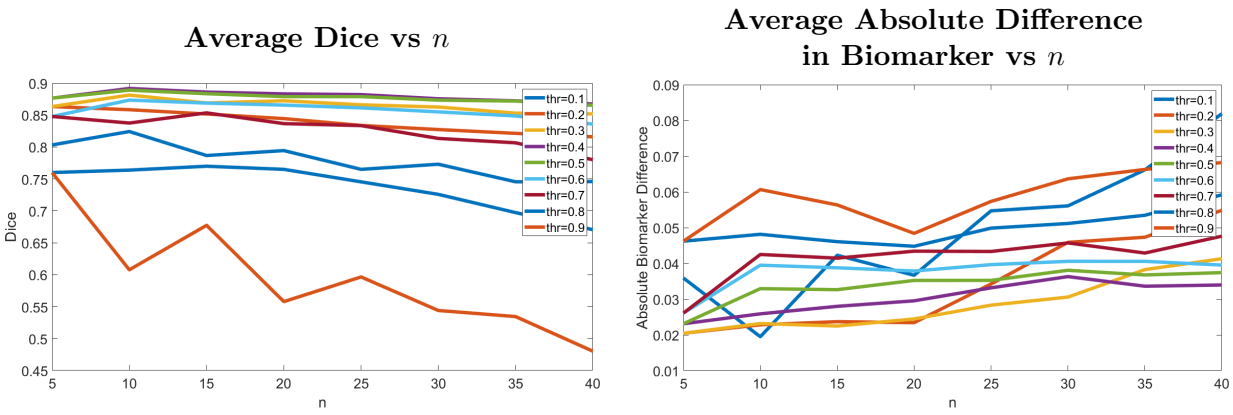

Figure 12: The left image shows the average Dice index plotted against the number of templates used in registration, and the right shows the average absolute difference between predicted and actual biomarker plotted against the number of templates used in registration. For both these figures, the average was computed over all 41 brains in the training set. The different curves correspond to different thresholds chosen for registration.

To choose the threshold and n hyperparameters used when we average registrations, described in 2.4.3, we performed a grid search, first using a wide range of n and threshold values (ranging from 5 to 40 and 10% to 90%, respectively). We chose to complete all further registrations with n=10 and a threshold of 50%, as this maximizes the average Dice index. The maximizer of the Dice index was chosen over the minimizer of the biomarker error because of the ways that biomarkers are subject to large differences given slightly different segmentations. We expect that the behavior of the Dice index average will generalize better to other data.

3.2 Machine Learning Results

First, to find the best model, two numerical experiments were executed. One to compare model efficiency and another to compare success on the validation data set. In total 3 models were considered. Model 1 and Model 2 used only normalized magnitude MR images as input. While Model 1 used 500 iterations, Model 2 used 1000 iterations to ensure the minimum validation loss would be reached. Model 3 used normalized magnitude, mean DENSE, and peak DENSE images as input,

500 iterations, and reached minimum validation loss. Model information and loss results can be seen in **Table 3** while the Dice index for each class can be found in **Table 4**.

	Input Channels	Total Iterations	Model Iteration	Minimum Validation Loss	Training Loss
Model 1	1	500	499	0.0362	0.0192
Model 2	1	1000	595	0.0383	0.0176
Model 3	3	500	309	0.0229	0.0180

Table 3: Final results from three different models. Models with 1 input channel were trained with only normalized magnitude images as input while those with 3 trained with normalized magnitude, mean DENSE, and peak DENSE images as input. Model iteration and training loss correspond to the model saved at the lowest validation loss. Here, iteration counting starts at 0.

	Training			Validation		
	Background	Cerebellum	Brain Stem	Background	Cerebellum	Brain Stem
Model 1	0.9963	0.9223	0.9024	0.9936	0.8345	0.8667
Model 2	0.9964	0.9377	0.8976	0.9938	0.8571	0.8552
Model 3	0.9962	0.9216	0.9116	0.9953	0.8965	0.8919

Table 4: Dice index for each class in Model 1, 2, and 3 calculated from both training and validation data sets.

The following figures show the cross-entropy loss and Dice index for each class over each iteration along with the resulting validation masks produced for Model 1 in **Figure 14**, Model 2 in **Figure 15**, and Model 3 in **Figure 16**. Looking at **Figures 14, Figure 16** and the results in **Table 4**, it can be seen that Model 3 reached higher Dice similarities than Model 1 in 500 iterations. Model 3 also reached higher overall Dice similarities than Model 2 where both models reached minimum validation loss. Model 3 also had the lowest resulting validation cross-entropy loss which can be observed in the quality of the predicted masks when compared with the known validation masks in **Figure 13** conclusion that Model 3 was optimal to use in evaluating the final test data and calculating biomarkers for both validation and test data.

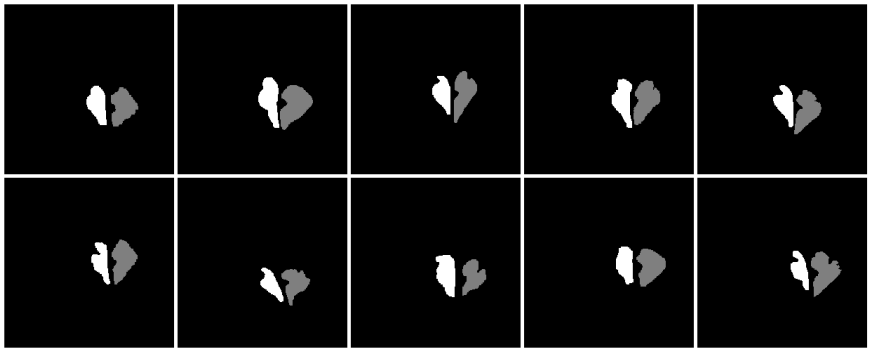

Figure 13: The images above represent the known segmentation masks used in the validation set. The masks shown here and the masks in the below figures are in corresponding order.

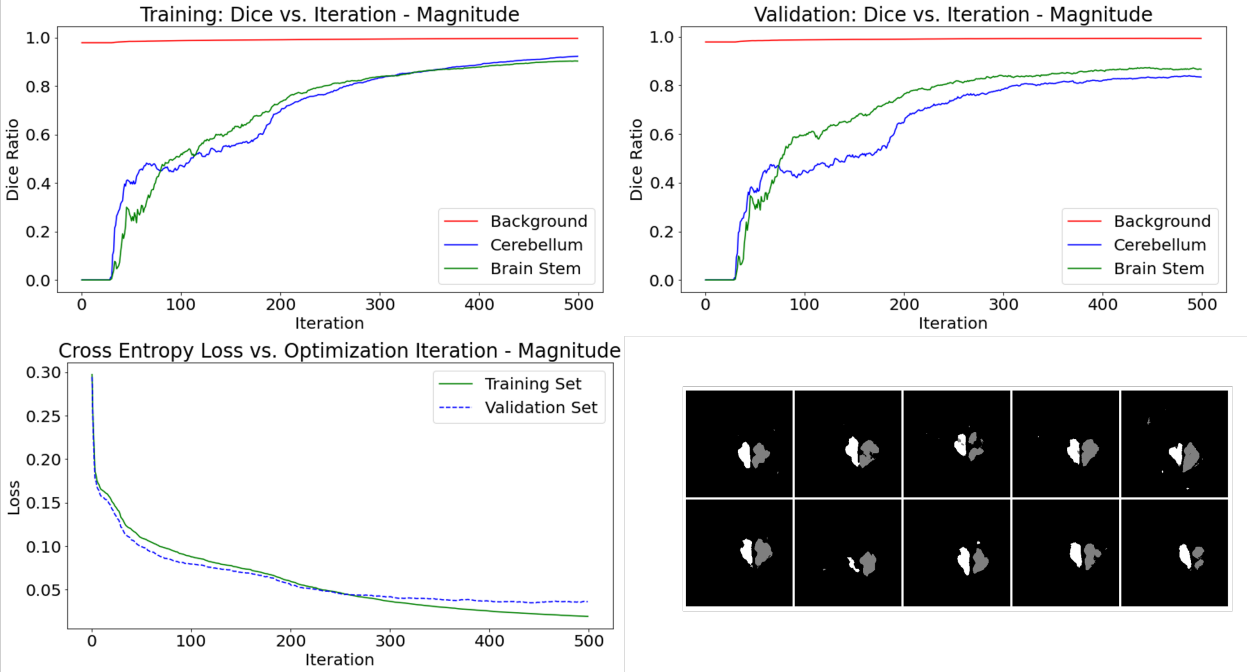

Figure 14: Model 1: The top images show the Dice similarities at each iteration for both training (left) and validation (right) data sets for each class. The bottom left image shows the cross-entropy loss plotted against iterations for both training and validation sets. The bottom right image shows the predicted masks.

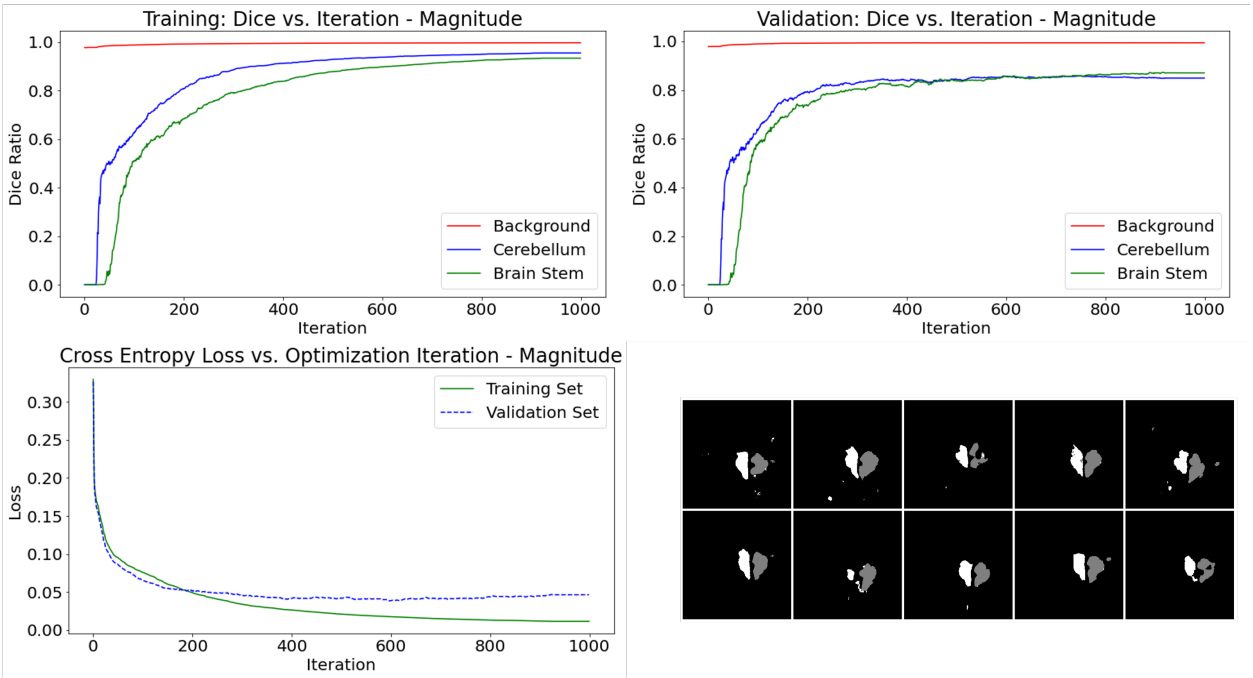

Figure 15: Model 2: Same configuration as above.

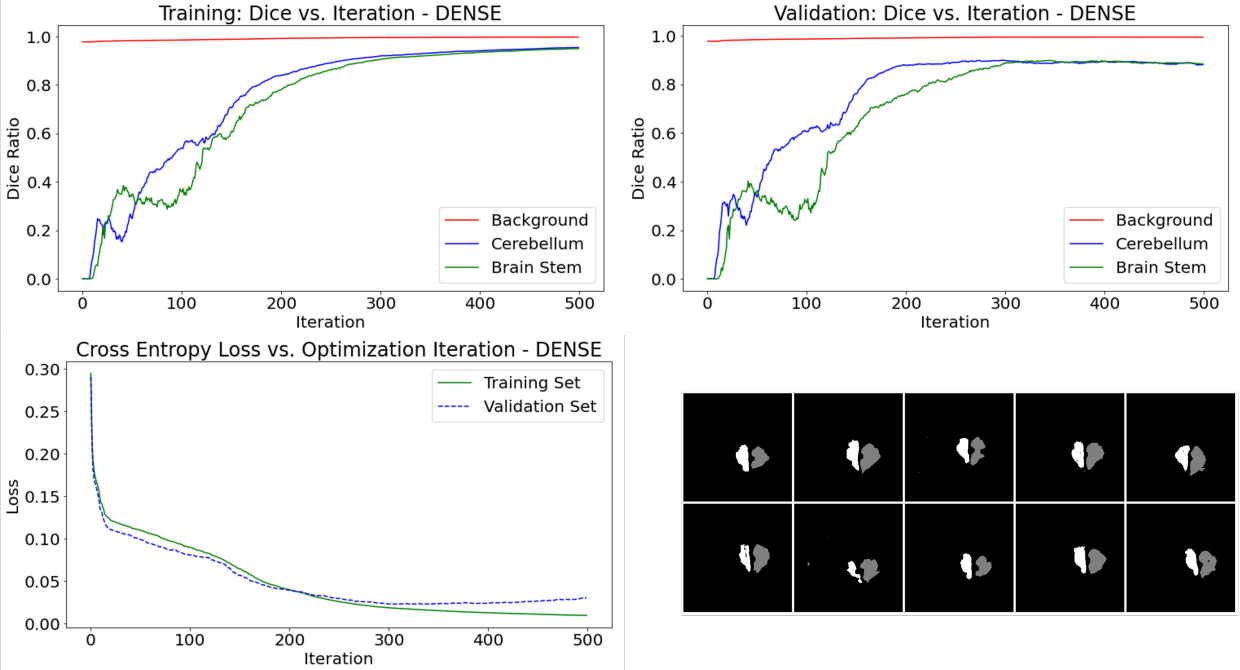

Figure 16: Model 3: Same configuration as above.

3.3 Combined Results

When analyzing atlas-based (AB) and machine learning (ML) in the end, we relied on the Dice and biomarker (BM) metrics again. **Table 5** looks at the samples provided in the testing set and compares these methods to the manually segmented mask (true). The masks that were compared are shown in **Figure 17**.

Test Set Subject #	Total Dice		Brain Stem BM			Cerebellum BM		
	AB	ML	True	AB	ML	True	AB	ML
1	0.76	0.83	2.32	2.37	2.25	1.97	1.31	1.62
2	0.88	0.88	0.69	0.68	0.68	0.47	0.45	0.45
3	0.84	0.86	1.25	1.13	1.15	1.34	1.03	1.22
4	0.86	0.76	1.06	0.99	1.00	0.58	0.53	0.54
5	0.85	0.89	1.02	0.99	0.98	0.64	0.58	0.59
6	0.59	0.71	0.85	0.77	0.76	0.54	0.90	0.60
7	0.91	0.94	0.86	0.84	0.85	0.48	0.45	0.47
8	0.88	0.92	0.89	0.89	0.88	0.48	0.51	0.49
9	0.91	0.88	0.73	0.73	0.74	0.52	0.51	0.51
10	0.87	0.86	1.05	0.97	1.20	1.24	1.07	0.97
11	0.84	0.89	1.48	1.48	1.47	0.99	0.85	0.90
12	0.90	0.87	0.58	0.58	0.59	0.37	0.39	0.37

Table 5: Results of finalized methods on testing set. Total dice is measured over the union of the cerebellum and brain stem and compared against the true mask. This means that the best possible Dice value is 1. Average distance is the average absolute difference between the true and predicted biomarker.

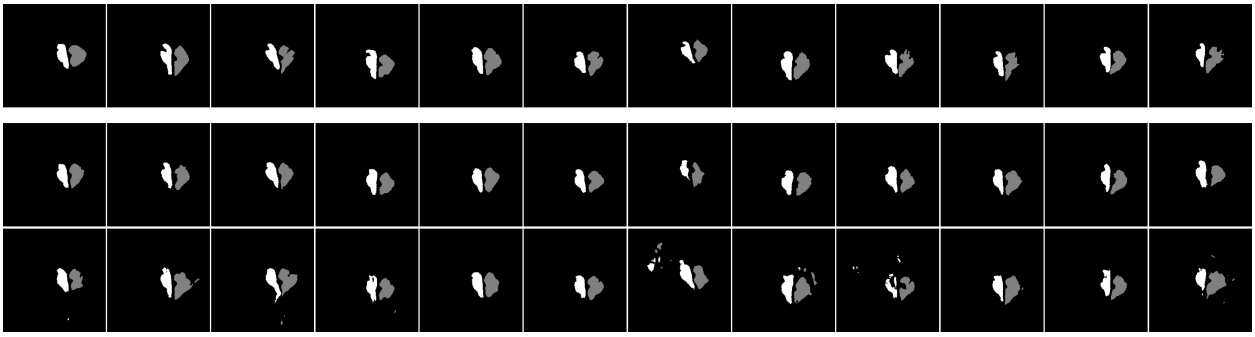

Figure 17: Compared True masks (top), atlas predicted (middle), and machine learning predicted (bottom).

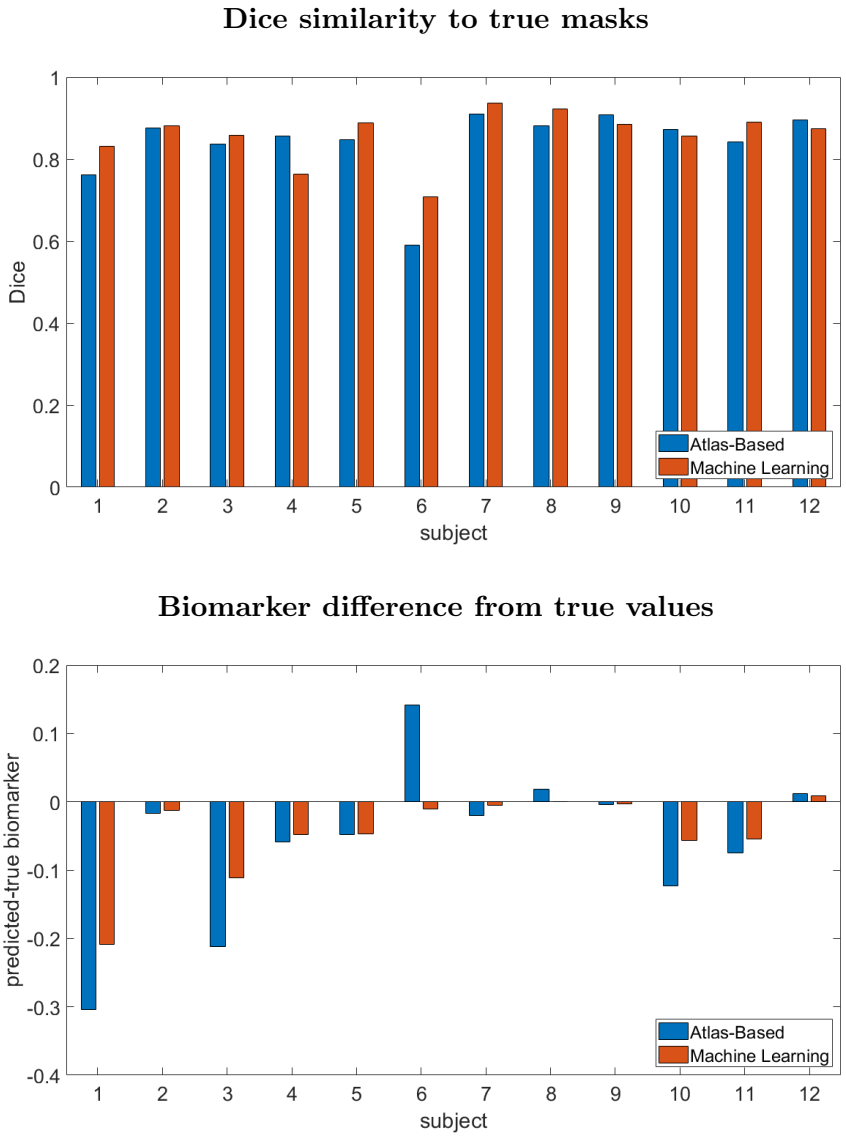

Figure 18: Bar graph displaying atlas-based image registration results (blue) and machine learning results (orange) from testing set in terms of Dice similarity and biomarker error when compared to target (manually generated) results. Note that higher values are better for the Dice similarity while a lower difference is desirable for biomarkers.

The bar chart in **Figure 18** offers more analysis into which method did better on each patient. Here we can see in terms of Dice similarity that machine learning was preferred in eight of the subjects while atlas-based was only preferred in four. This difference becomes even more prominent when the biomarkers are calculated. In terms of biomarkers, machine learning was better than atlas-based on every patient. Some analysis of **Table 5** are shown in **Table 6** and **Figure 19**.

Analytics	Total Dice		Brain Stem BM			Cerebellum BM		
	AB	ML	True	AB	ML	True	AB	ML
Mean:	0.84	0.86	1.06	1.04	1.05	0.80	0.71	0.72
Avg. Distance:	—	—	—	0.04	0.05	—	0.16	0.09
SD :	0.09	0.06	0.47	0.48	0.45	0.49	0.31	0.38

Table 6: Results of finalized methods on testing set. Total dice is measured over the union of the cerebellum and brain stem and compared against the true mask. This means that the best possible Dice value is 1. Average distance is the average absolute difference between the true and predicted biomarker.

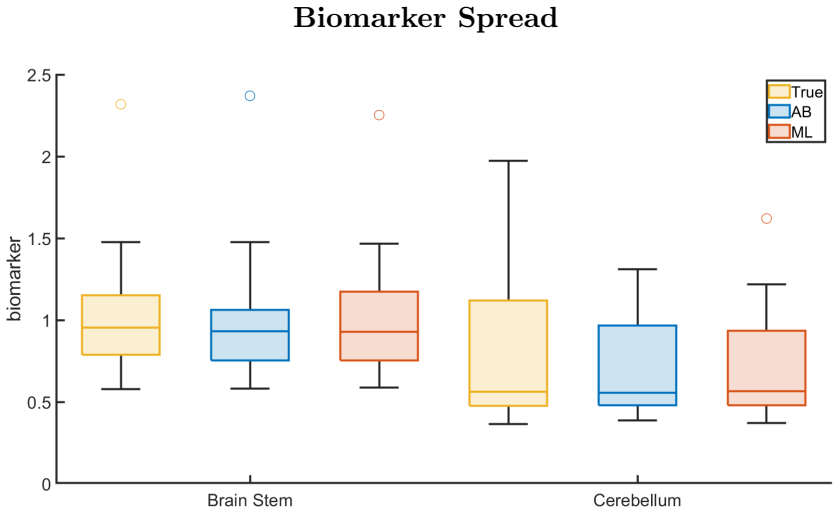

Figure 19: Box plot showing true and predicted biomarkers for the brain stem and cerebellum.

4 Discussion

Despite the small amount of data available to train over, the machine learning segmentation method outperformed the atlas-based segmentation method; the machine learning predicted masks that were more similar to manually drawn segmentations (higher Dice indices), and produced biomarkers that more closely match gold-standard results (lower biomarker differences). Both methods produced more accurate biomarkers for the brain stem than cerebellum. **Figures 18** and **19** and **Tables 5** and **6** present these results in further detail. Though the methods produce comparable results, we anticipate that further work and further data would only widen the performance gap, improving the machine learning segmentation's accuracy more quickly than it would atlas-based.

Further work to improve the machine learning method could include adding a regularizer, similar

to that used in the atlas-method (described in section 2.4.2), that could eliminate non-contiguous pixels from predicted masks, resulting in even better segmentations.

A multi-batch approach, that iteratively creates training and validation sets across non-testing data, could further help improve results, to maximally take advantage of the currently relatively small data set.

Going from here, as more DENSE images are collected, the network can be trained with a larger data set further increasing the accuracy of the segmentations. This would also increase the bank of images used in atlas-based which could potentially lead to better results.

An interesting possibility for further work may also lie in examining the accuracy of radiologist predicted masks. Though they were treated as truth in this study, manually drawn segmentations can vary in perhaps significant ways.

In the case of brain stem and cerebellum segmentation, and the production of biomarkers that can help identify CMI, it seems that the sense of control an atlas-based method offers comes at the cost of accuracy that a machine learning method can reach.

Acknowledgements

We want to sincerely thank Dr. Lars Ruthotto and Justin Smith for their mentor-ship and help throughout the project, as well as Dr. John Oshinski for providing us with this data set and collaborating with us. We would further like to thank all the mentors at Emory's 2021 REU/RET Program and the NSF for making this work possible.

References

[1] AANS. Chiari malformation, July 2021.

[2] A. Arora. U-net: A pytorch implementation in 60 lines of code, Sept 2020.

[3] P. A. Bolognese, A. Brodbelt, A. B. Bloom, and R. W. Kula. Chiari i malformation: Opinions on diagnostic trends and controversies from a panel of 63 international expert. *World Neurosurg*, pages e9–e16, 2019.

[4] M. Burger, J. Modersitzki, and L. Ruthotto. A hyperelastic regularization energy for image registration. *SIAM Journal on Scientific Computing*, 35(1):B132–B148, 2013.

[5] A. Coste. Image processing: Histograms, 2012.

[6] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, Nov. 2016.

[7] T. Lan, D. Erdogmus, S. Hayflick, and J. Szumowski. Phase unwrapping and background correction in MRI. pages 239 – 243, 11 2008.

[8] J. Modersitzki. *FAIR: flexible algorithms for image registration*, volume 6 of *Fundamentals of Algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2009.

[9] A. Monteux. Metrics for semantic segmentation, May 2019.

[10] B. S. T. Nwotchouang, M. S. Eppelheimer, S. H. Pahlavian, J. W. Barrow, D. L. Barrow, D. Qiu, P. A. Allen, J. N. Oshinski, R. Amini, and F. Loth. Regional Brain Tissue Displacement and Strain is Elevated in Subjects with Chiari Malformation Type I Compared to Healthy Controls: A Study Using DENSE MRI. *Annals of Biomedical Engineering*, pages 1–15, Dec. 2020.

[11] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[12] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. pages 1–8, May 2015.