Abstract:

The steps detailed in this report were taken with the intent of modifying cardiovascular magnetic resonance (CMR) images to lower the difficulty in diagnosing hypertrophic cardiomyopathy. The ability to view hypertrophy of the ventricular heart muscles can be of consequence in preventing further cardiovascular disease and heart failure. So, it is vital for us to be able to process these images in a way that assists the viewer. **Figure 1** shows the original images used in processing for hypertrophic cardiomyopathy. Image 1 and Image 2 are CMRs of sick individuals with hypertrophic cardiomyopathy. Image 3 is a CMR of a healthy individual, for comparison. Obviously, these images are not perfectly clear and could use some processing. So, the algorithms we will use to process them are 1) Crop and Interpolate, 2) Add Contrast, 3) Edge Detection, and 4) Classification Using Neural Network.

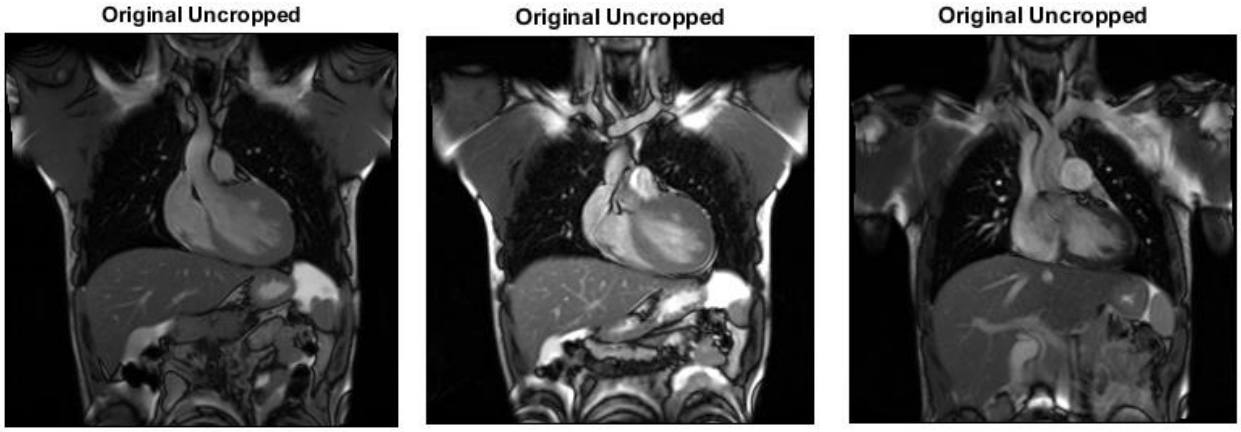

Figure 1: From left to right: Sick (Image 1), Sick (Image 2), Normal (Image 3)

Methods and Results:

Crop and Interpolate:

The first step to clearing up the images was to crop them to focus on the heart. After cropping, I noticed the images were somewhat pixelated, so interpolation was the next logical step to enhance the images. The code used for cropping and interpolating can be found below in **Figure 2**. It turned out that the images were fairly standardized, and the cropping dimensions did not need to change for each image.

```
figure(1);
subplot(131); imshow(I0); title("Original Uncropped")

I0 = imcrop(I0, [81 60 130 100]);

subplot(132); imshow(I0); title("Original Cropped")

I_BC = imresize(I0, 16, "bicubic");

subplot(133); imshow(I_BC); title("Bicubic")
```

Figure 2: Using imcrop() to crop, and imresize() to interpolate.

In **Figure 3, Figure 4, and Figure 5** you can see that the images have become much clearer and less pixelated. The interpolation algorithm works by resizing the image and then using data from surrounding pixels to derive information about what is between those pixels resizing to fill in the missing data. This can be used to enhance the quality of any image, to a degree, so it has many uses in medical imaging. These results were quite impressive, although these figures might not be large enough to really get the idea. Zooming into the images may help.

Figure 3: Image 1 cropped and interpolated.

Figure 4: Image 2 cropped and interpolated.

Figure 5: Image 3 cropped and interpolated.

Add Contrast:

Some of the images in the dataset were very dark, and some already had contrast. However, adding contrast helped to view any of the images better. The code for adding contrast can be found below in **Figure 6**, and I used the "adapthisteq()" command because it seemed to do the best job.

```
figure(2);

I_BC_hist = adapthisteq(I_BC);

subplot(121); imshow(I_BC); title("Bicubic")
subplot(122); imshow(I_BC_hist); title("AdaptHistEq")
```

Figure 6: Using adapthisteq() to add contrast

The contrast-adding functions work by creating steeper intensity gradients around areas where there are small changes in intensity. Essentially, it makes dark areas darker and light areas lighter. It can have the effect of making objects pop more, which is the effect we want to induce around the heart. This is because the stroke volume tends to show up lighter on a CMR and the ventricular muscle shows up darker. Further differentiating the two can help to measure the thickness of the muscle. As mentioned, some images were lighter than others. For instance, image 2 was already bright enough, but the added contrast still helped to differentiate the ventricular muscle from the stroke volume in the heart. **Figures 7, 8, and 9** display these results. Additionally, this technique would be useful in any application where discernment is key, such as diagnostics.

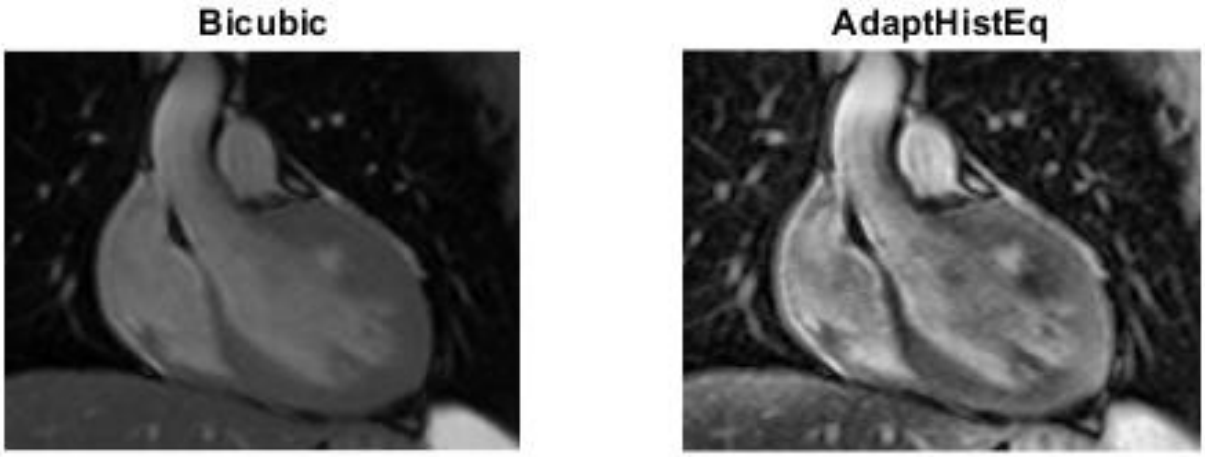

Figure 7: Image 1 with added contrast

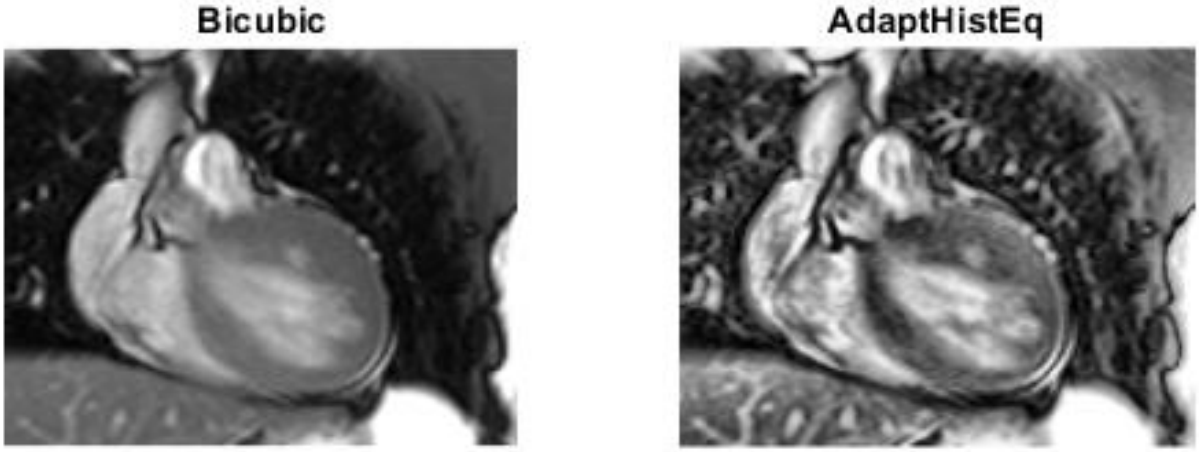

Figure 8: Image 2 with added contrast

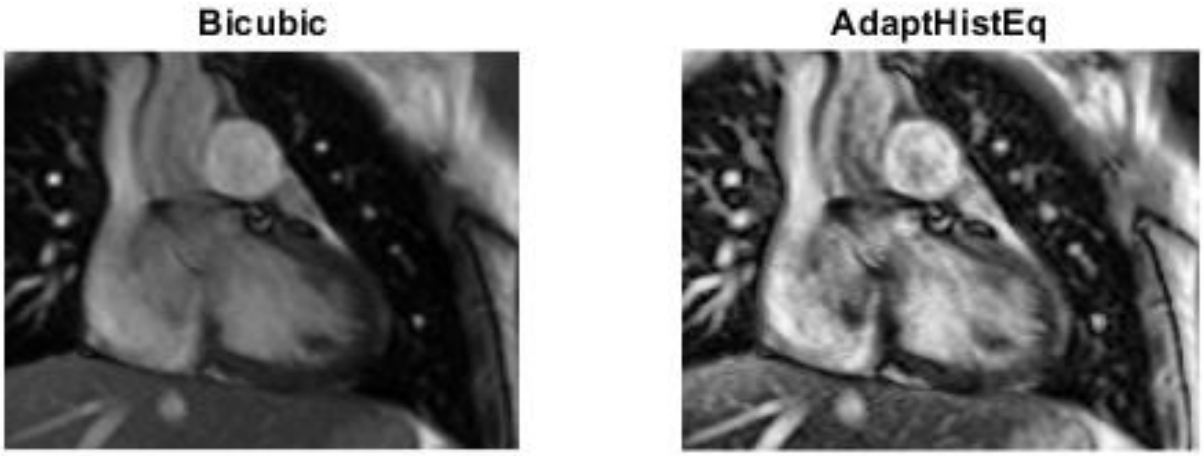

Figure 9: Image 3 with added contrast

Edge Detection:

Next, I utilized edge detection algorithms to try and understand more information about the images that might not already be present. Because our eyes cannot visually separate edges in a binary way like computers can, it would be interesting to see what the computer classified as an edge in these heart images. The code was relatively simple and can be found in **Figure 10** below. I just used the "edge()" command with the specification that it should be "canny" edge detection. The other methods like the Laplace of Gaussian did not seem to produce useful results (only black images), I think just because the baseline used to detect the edges was too low.

```
I_BC_hist_canny = edge(I_BC_hist, 'canny');

figure(3);
subplot(121); imshow(I_BC_hist); title("AdaptHistEq")
subplot(122); imshow(I_BC_hist_canny, []); title("Canny")
```

Figure 10: Code using edge() for edge detection

Figures 11, 12, and 13 show the results of the edge detection. Like with the interpolation images, these can be difficult to see the differences. However, you can notice that the areas which harbor the ventricular muscles appear to have less white dots and more void, especially for the first and second images which have exhibited hypertrophy. So perhaps this kind of technique can be used to measure the average thickness of the ventricular muscle and concretely determine the diagnosis. This method is very useful for accentuating steep intensity gradients, while simultaneously binarizing the image to remove unwanted detail.

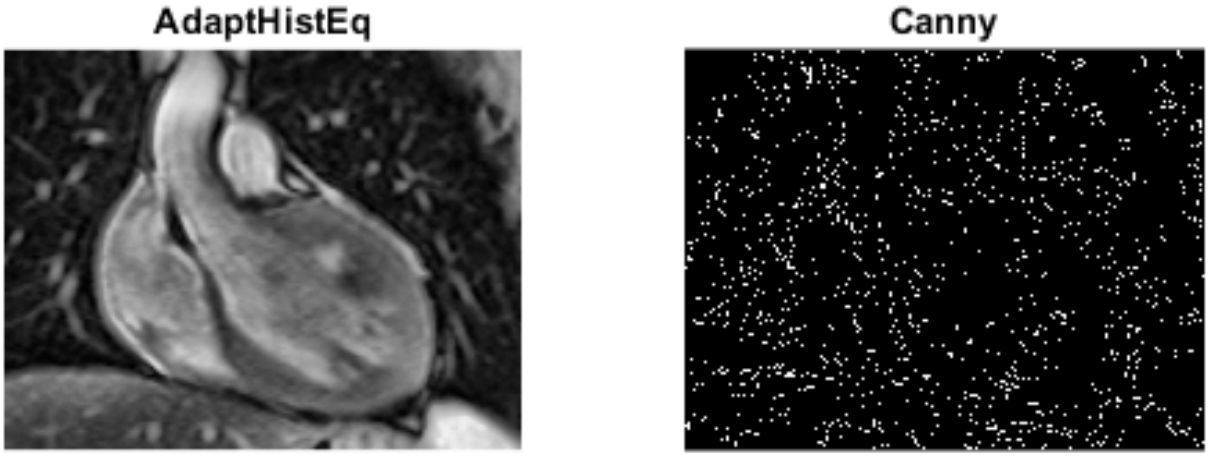

Figure 11: Image 1 edge detection

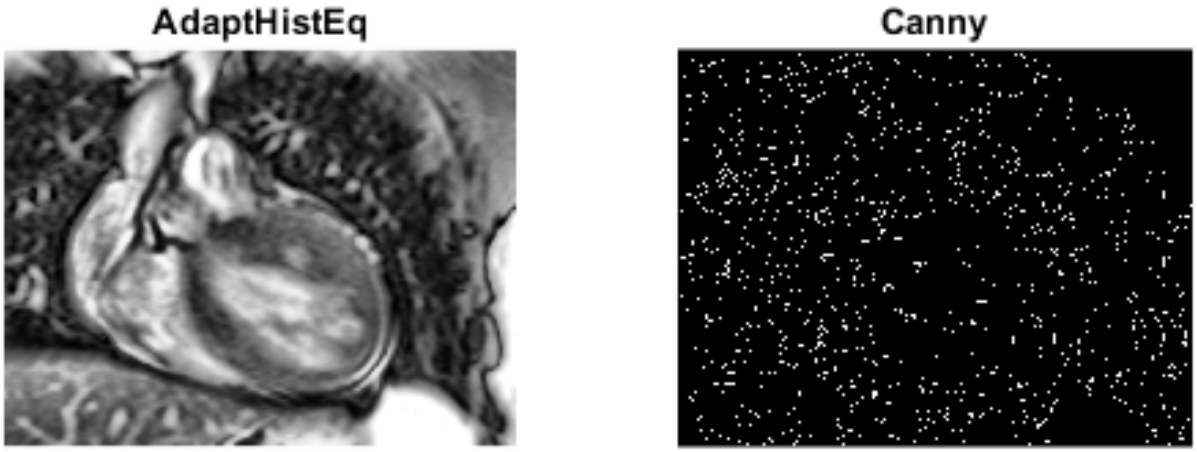

Figure 12: Image 2 edge detection

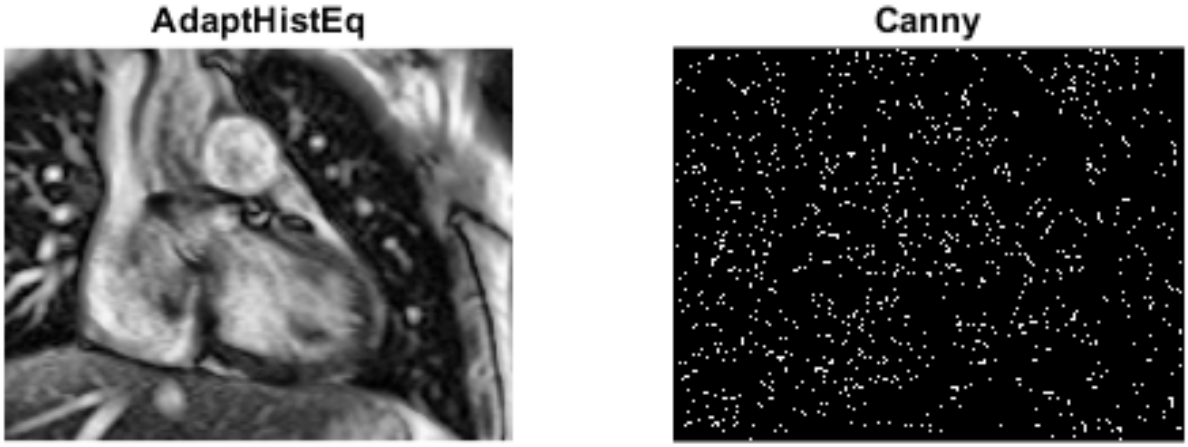

Figure 13: Image 3 edge detection

Classification Using Neural Network:

The last algorithm used to analyze these images was the use of transfer learning and googLeNet's pretrained network to classify new images. I used a total of about 72 images, similar to the ones in this report, to train this network to identify sick vs normal individuals' hearts. The classifications,

training progression, test code, and test results are seen in **Figures 14, 15, 16, and 17** below.

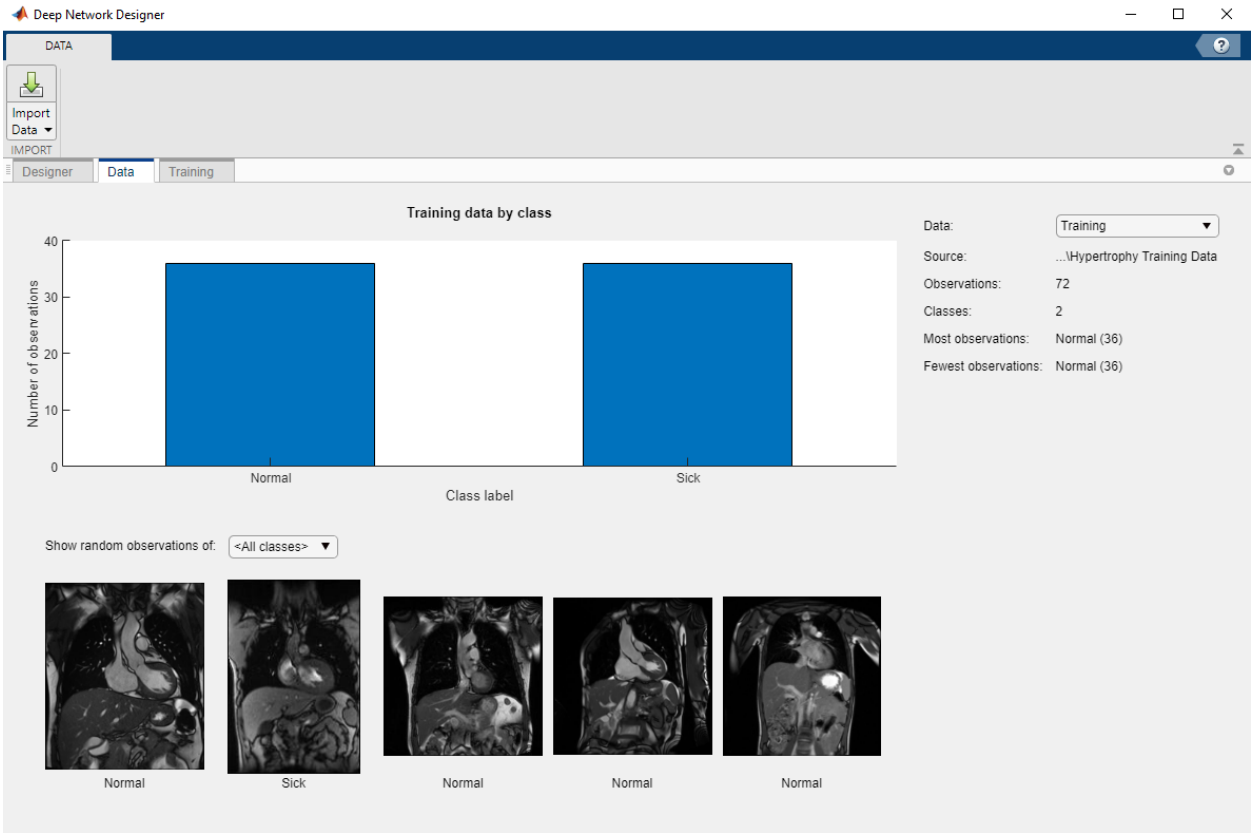

Figure 14: Classifications of images, sick and normal

Figure 15: Training progression of neural network

```
>> I1 = imread('Test Normal\img0002-15.jpg'); I2 = imread('Test Sick\img0007--3.41357.jpg');
>> I1 = cat(3, I1, I1, I1); I2 = cat(3, I2, I2, I2);
>> I1 = imresize(I1, [224 224]); I2 = imresize(I2, [224 224]);
>> [YPred,probs] = classify(trainedNetwork_1,I1); [YPred2,probs2] = classify(trainedNetwork_1,I2);
>> subplot(121); imshow(I1); label = YPred; title(string(label) + ", " + num2str(100*max(probs),3) + "%"); subplot(122); imshow(I2); label =
```

Figure 16: Code for testing the neural network.

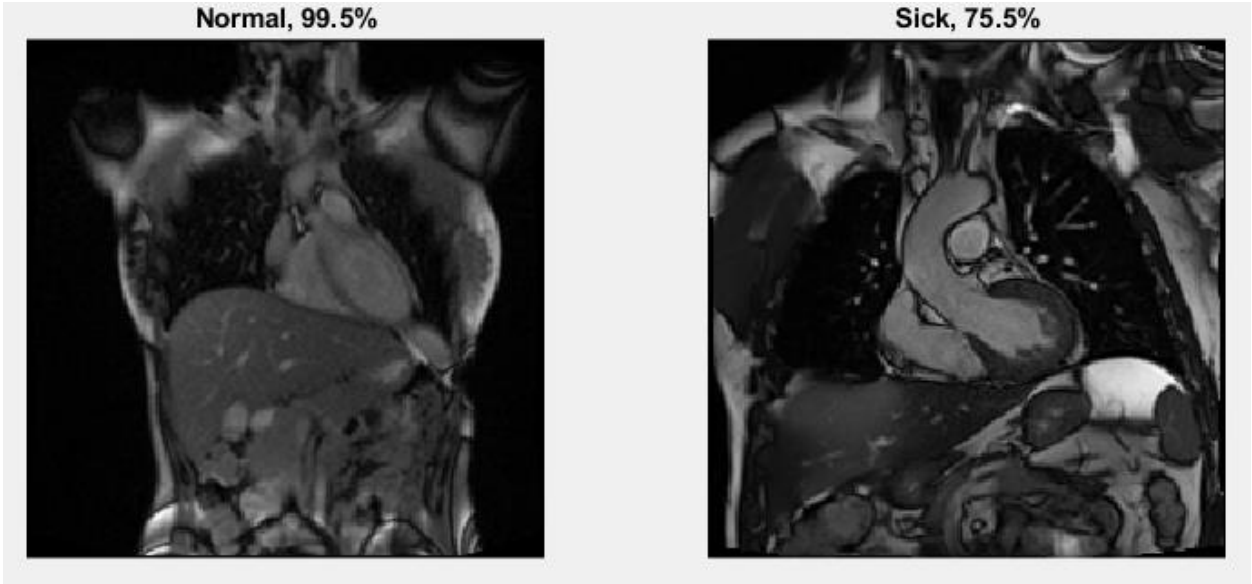

Figure 17: Results of neural network test

The results of the neural network test were rather impressive, with the neural network correctly classifying a couple of test images. These images were kept separate from the training/validation data, of course, to get an accurate result. Although I would like to suspect that the neural network identifies the difference in cardiac muscle, the truth is that it is very difficult to tell what exactly the network is using to make this decision. Regardless, if the accuracy of this network is high enough, it could be useful. It may even be used in a clinical setting. Perhaps, in a day like today, it could be used in addition to a professional medical opinion. The doctor could first analyze the CMR of the patient and propose his professional diagnosis. Then, if the doctor or patient likes, they can consult the neural network for a kind of "virtual second opinion". Thinking way further into the future, neural networks may be able to do the job effectively on their own.

Conclusion:

This project consisted of 3 images and 4 algorithms to process, analyze, and classify images of hypertrophic cardiomyopathy. Through the process, I mostly learned how much can be done with just a few simple lines of code and a good computer. These kinds of analyses are relatively easy once you have learned the theory and coding techniques, but they also hold the power of utilization in the real world, which is the most important part. People of the most esteemed professions can use these methods to aid in their work, which means that this is uplifting news for my own career, as well. Some things I might improve if I were trying to perfect these techniques would be to generalize it more to handle any image of the heart, even ones from different angles. I would also use many more images to train my neural networks, because the validation accuracy of my network was only 73%. However, when I tried many more images, my computer crashed. So, I am happy with the results I got for now.