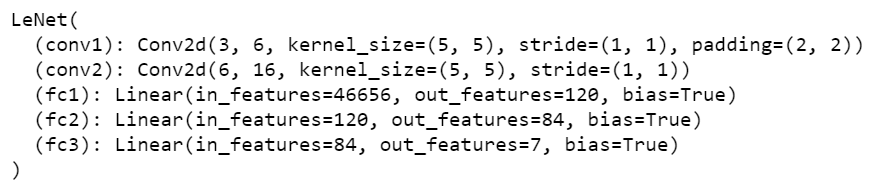
Elick Coval HW7

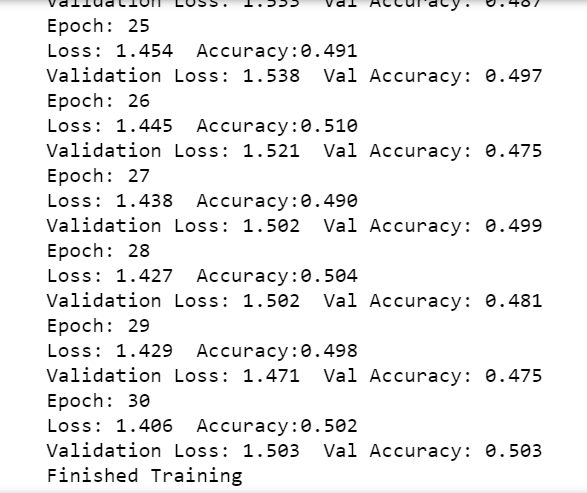
4/17/2020

Report

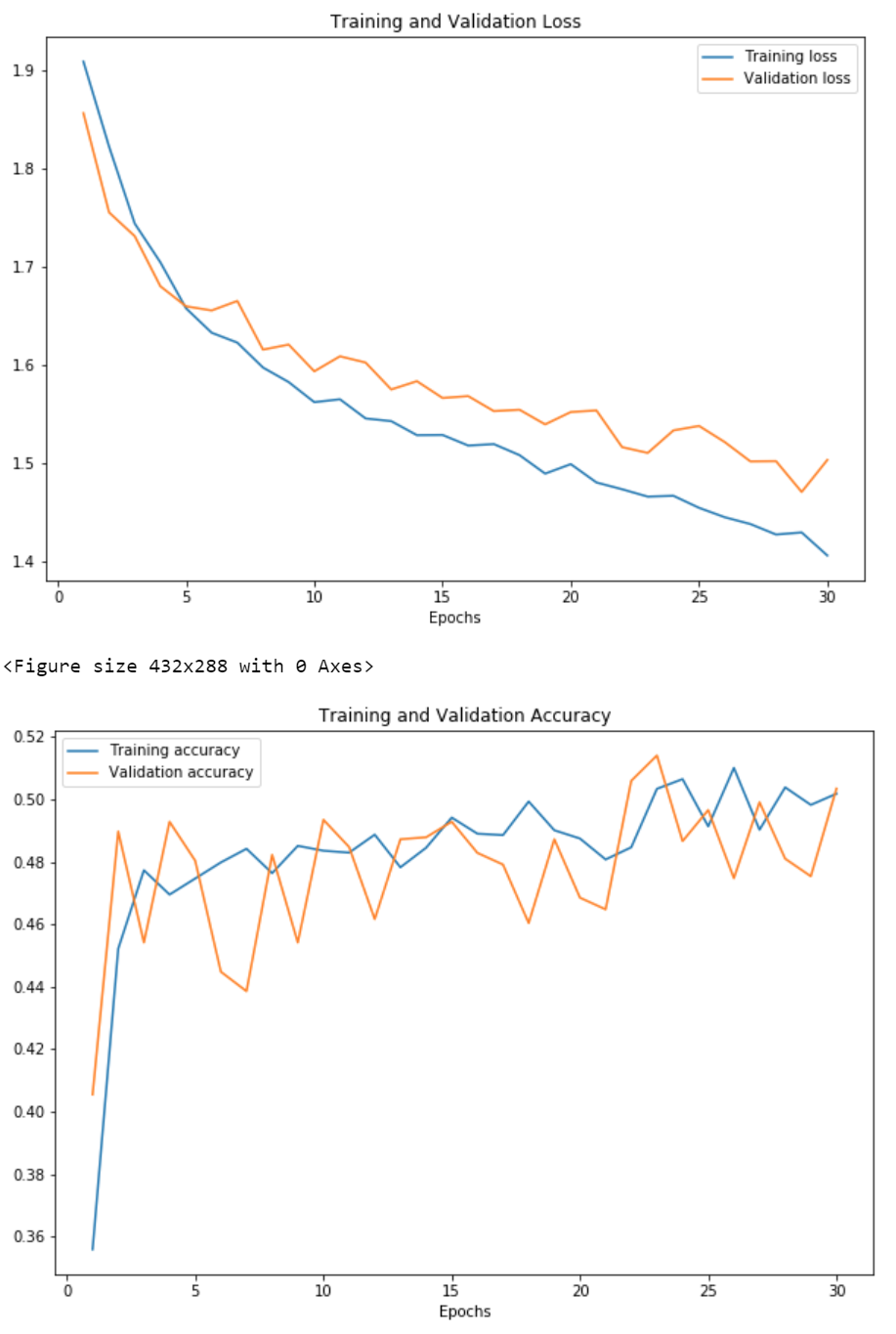
The first thing that was needed was to define a convolutional neural network. We start with a LeNet architecture, which is the more straightforward one usually used for handwriting character and optical recognition. Next, a loss function and optimizer are needed so we define those and print the result which shows the layer functions.



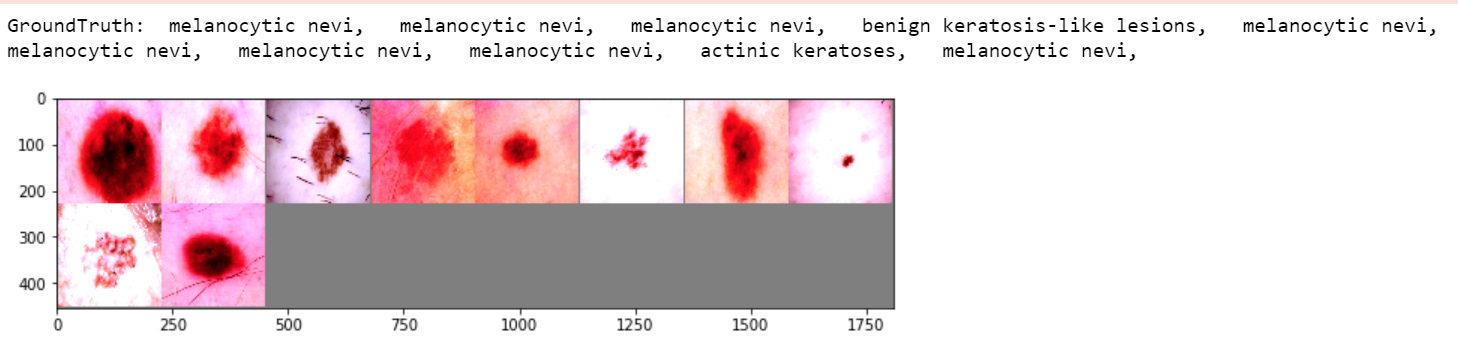
Then we had to define some helper functions to evaluate the training process and then start training the network. This took about an hour on an Intel i7-7700 CPU, hearing all the difficulties with the length I decided to shorten from 50 to 30 epochs. Although the real problem comes later with the more complex ResNet.



Next, we plot the training and validation loss curves, my curves diverged more than the example for both loss and accuracy.



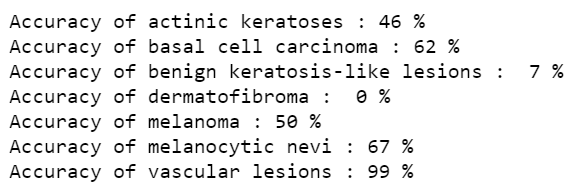
We then moved on to testing, first was displaying images from the test set.



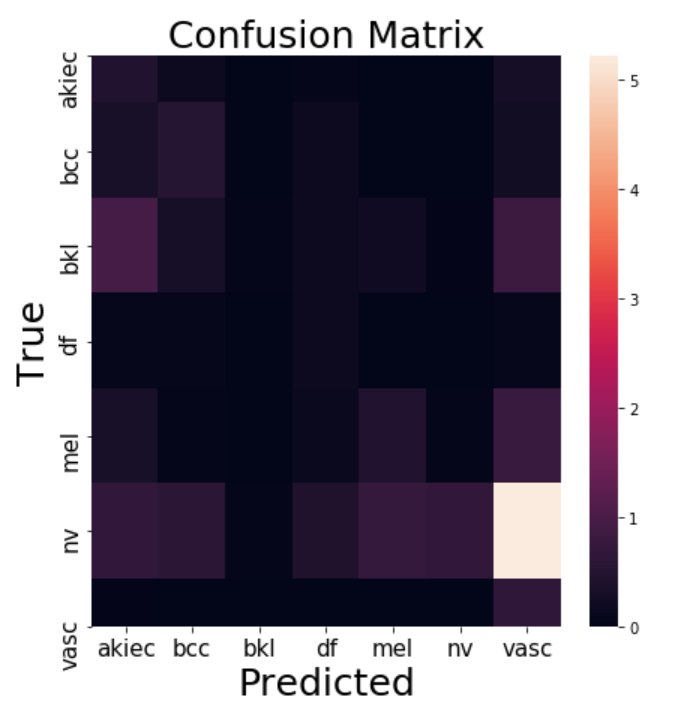
After that was actually checking the performance on the test network, mine came out a hair better than the example.



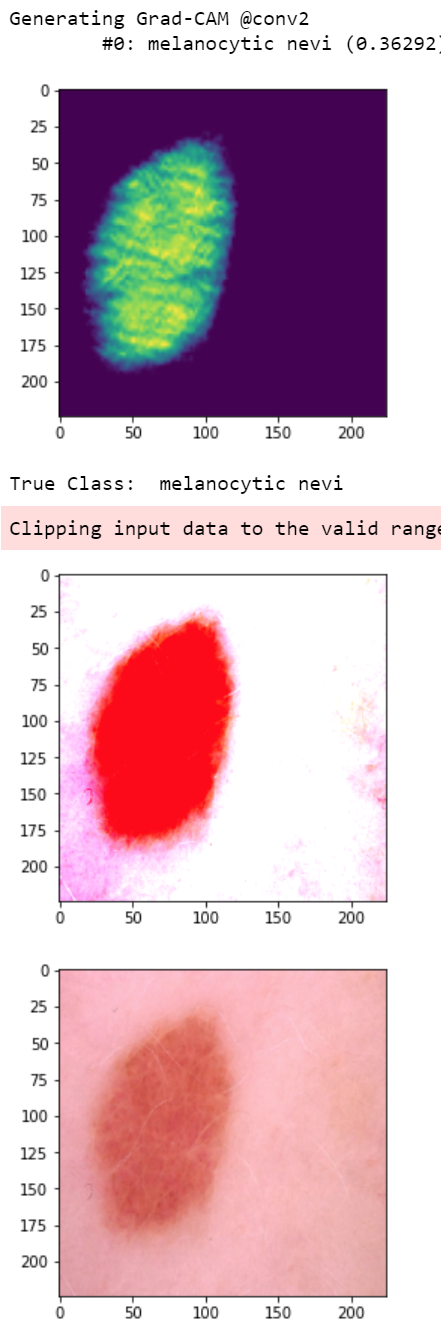
The following step involved checking to see which classes performed better than others.



We then had to create a confusion matrix to compare our model’s predictions with a perfect one, mine looked like this.

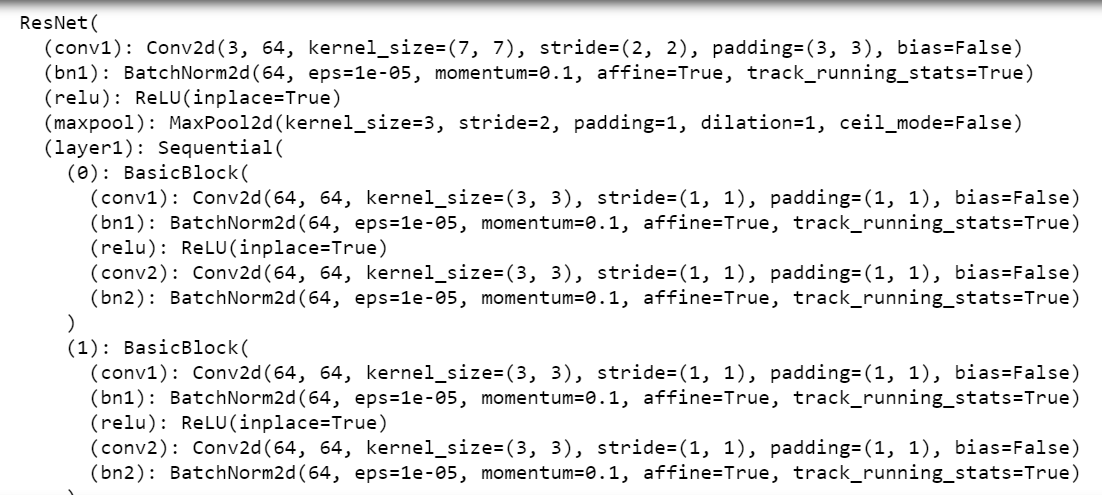


We then were responsible for generating images showing the gradient-weighted class activation mapping. This technique highlights the areas of the image that were highly influential in the decision-making process of our model. I thought this part was really exciting.

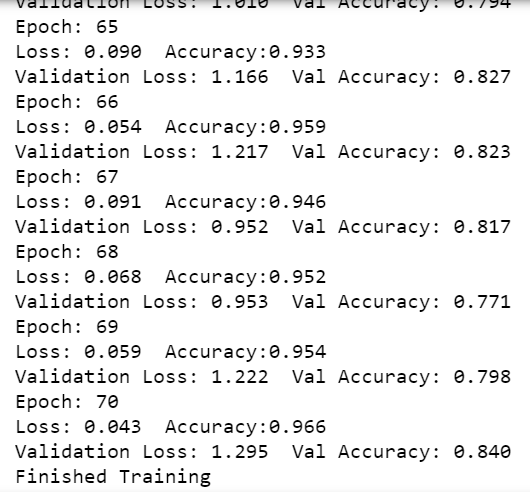


The final accuracy was 63.7% with 36.3% of the images being misclassified, pretty bad performance for something as serious as cancer. Which is why we switch to a deeper, more widely used architecture, ResNet. ResNet contains many more processing layers and a concept called residual blocks, which yields better gradient-flow and learning capacity.

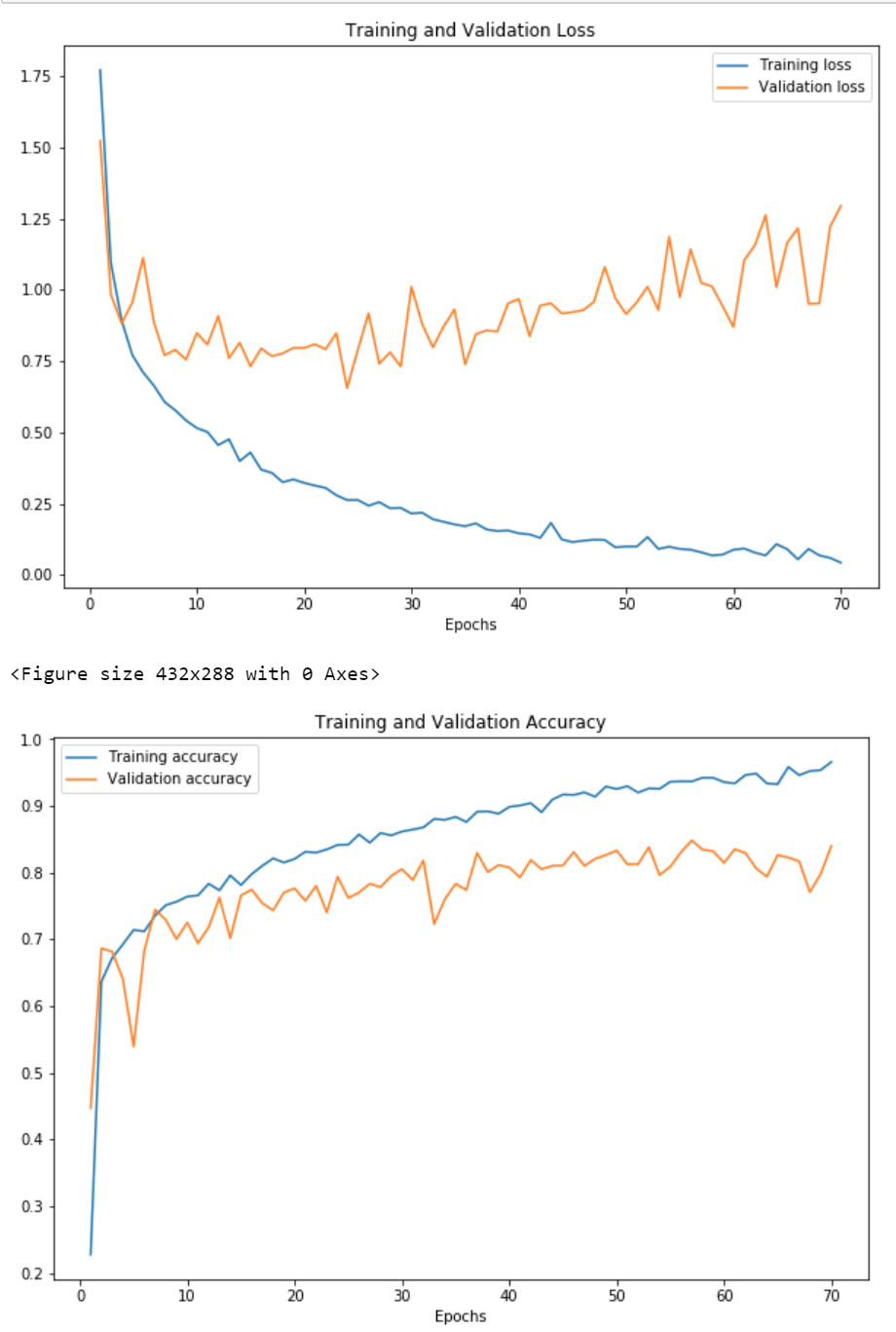
After downloading a pre-trained model and defining a loss function and optimizer for the ResNet.



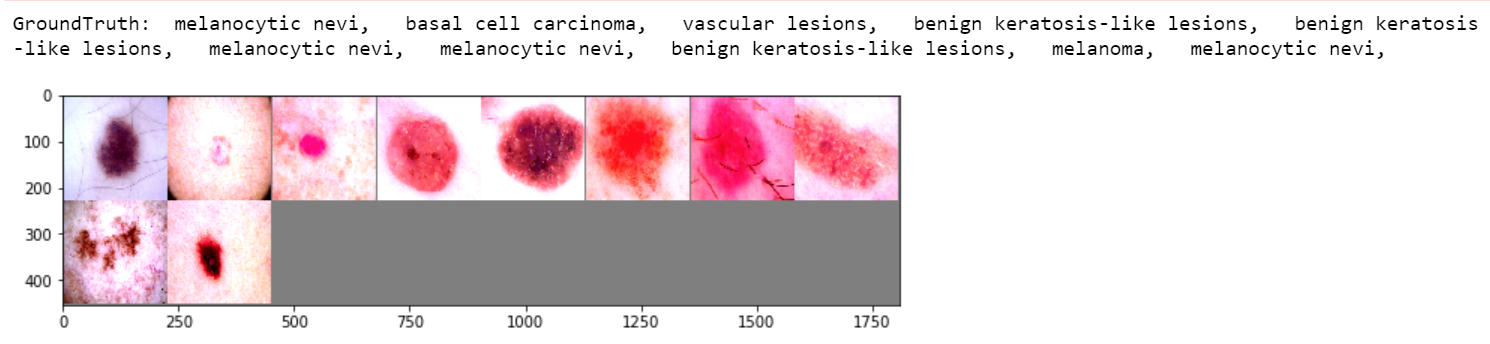
Running the ResNet training didn’t take as long as feared, maybe 2-3hrs.



Plotting the training and validation loss curves show significant overfitting compared to the example as the training loss decreases while the validation loss starts to rise.



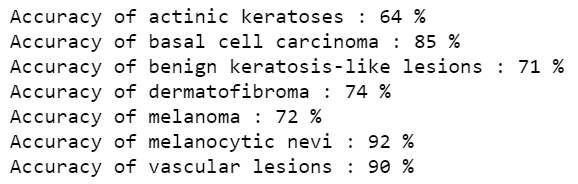
Again, we have to test to see if the network has learnt anything, so we first display images from a test set.



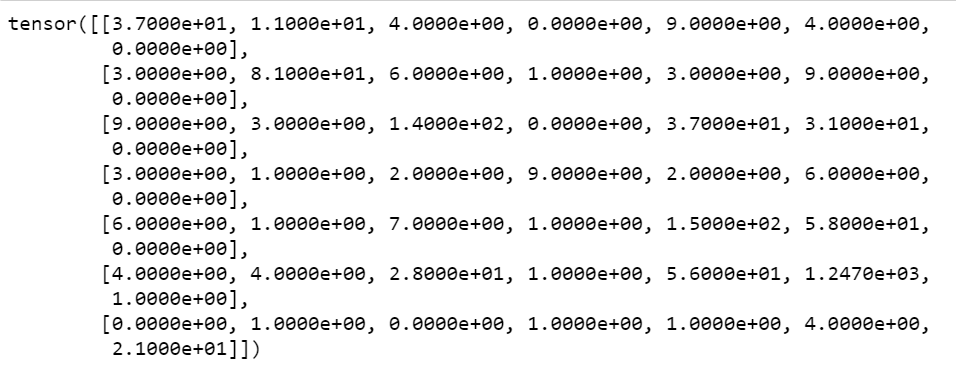
And then check the performance of the network. My network was just 1% short of the example.

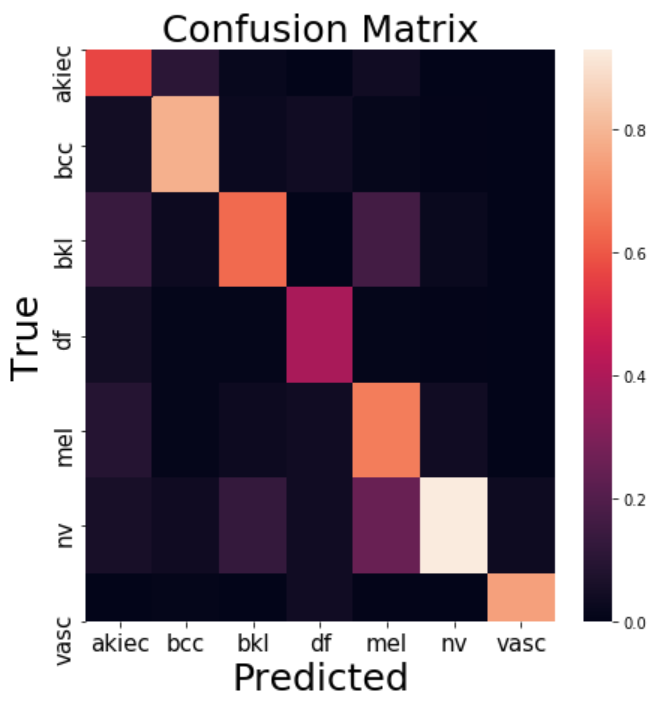
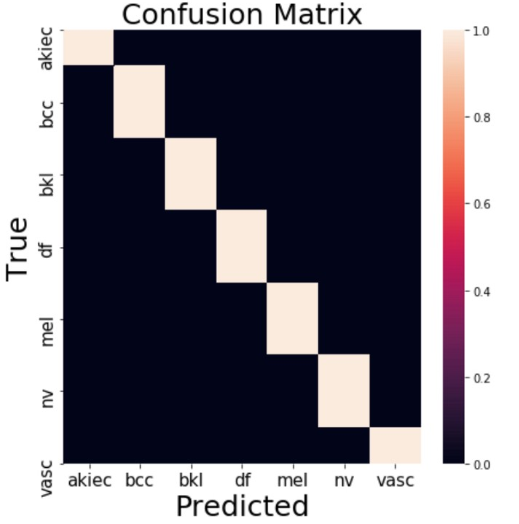


Zeroing in on the different class accuracy, we see much higher percentages for each class, confirming the ResNet does a better job identifying the different classes of lesions.



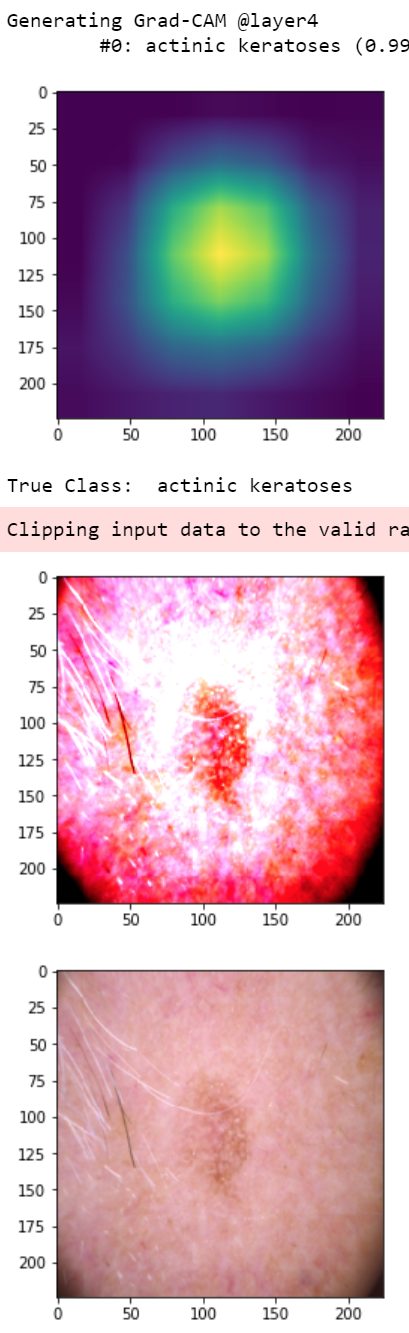
Again, we build a confusion matrix to visually represent which classes were easier/harder to identify by out neural net.



 It is clear the deeper ResNet did a much better job of distinguishing the different lesion classes.

My result vs Ideal

Generating the Grad-CAM, we see that the ResNet focuses on a much more through area of the lesion. The LesNet had much more of a patchwork of highlighted/darker areas, here we see the ResNet seems to start from the middle of the lesion and work its way out with the entirety of the lesion being relevant and the x and y axis all the way to the edges being most important.



I enjoyed this assignment very much, it made neural nets less abstract and more accessible, and I am excited to try this with a different application.