COMS 4771 COVID Challenge

Mahir Oberai mo2654

May 2020

1 Methods and Algorithms

Firstly, before trying more advanced methods, I simply went through the sklearn classifiers that I was already familiar with. These included Decision Trees, Random Forest Trees, SVN, and Adaboost. Although these methods were not very successful in classifying my validation set, they provided an entrance for me to begin looking for more image classification specific methods. Before this class, I had never trained neural networks and so, after some research, I decided to delve into implementing my first Convolutional Neural Network (CNN) classifier. After training my simple CNN, I saw that I was able to quickly reach around 70 percent validation accuracy by resizing my training set images to very low dimensions and running about 50 epochs. In order to improve this validation accuracy it was evident that I needed to allow the model to train on larger resolutions in order not to lose valuable training information. Furthermore, after more research, I found that the VGG16 architecture of CNN was a proven high accuracy structure for image classification and so I implemented another simple CNN using VGG16. Now, given more time, I would've able to train my VGG16 model and achieve high accuracies given 100 steps per epochs and around 25 epochs (I set model checkpoint to save my best model and an early stopping monitor with a patience of 20 to allow the model to continue training for up to 20 epochs without increasing its validation accuracy). However, I was unable to train a model in time and so my final results are from my simple implementation of CNN.

2 Pre-processing

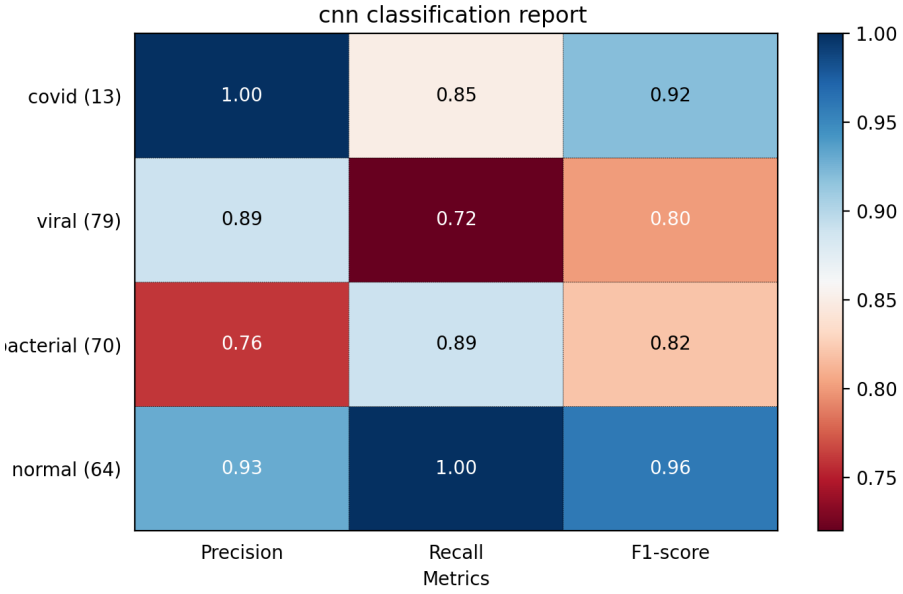
As mentioned, I found that the sizes of the images caused significant changes in the accuracies of my models. At first I was training on tiny dimensions, such as 28x28, just to see the model in action and was still achieving decent validation accuracies. Note: I went through a lot of research trying to see which data augmentation would positively affect my accuracies. For example, I looked into segmentation, marking and such. However, I noticed that for this application and the given data set using the Keras ImageDataGenerator with some parameters like shear range, zoom range, rotation range and horizontal flip,

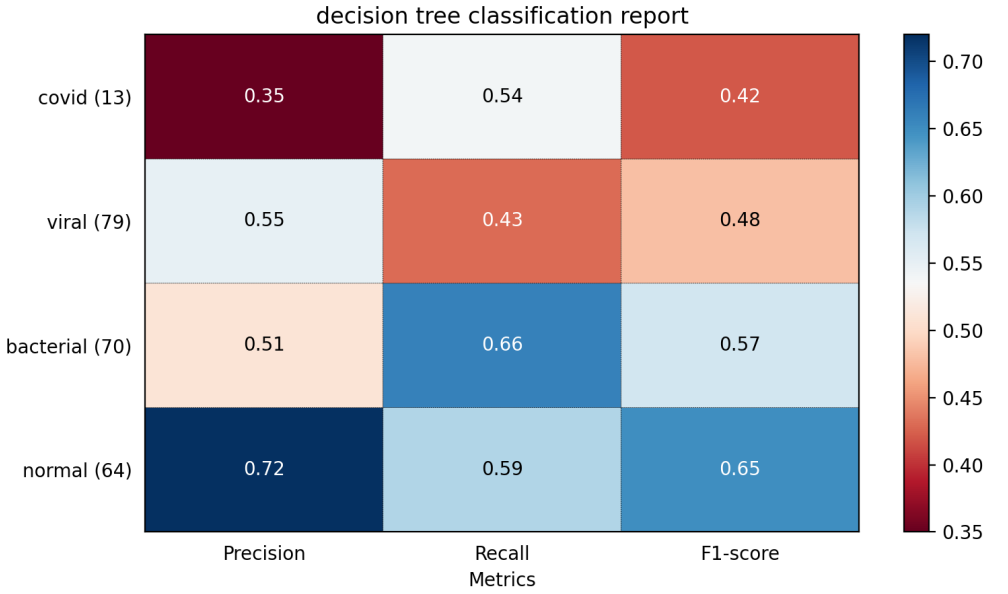
as well as resizing my images to 128x128 and maintaining their RGB channels, I was able to breach the full credit benchmark of 80 percent evaluation score. For my final result I used a cnn that loaded images as grayscale since I thought the RGB original format might be less useful for xray images.

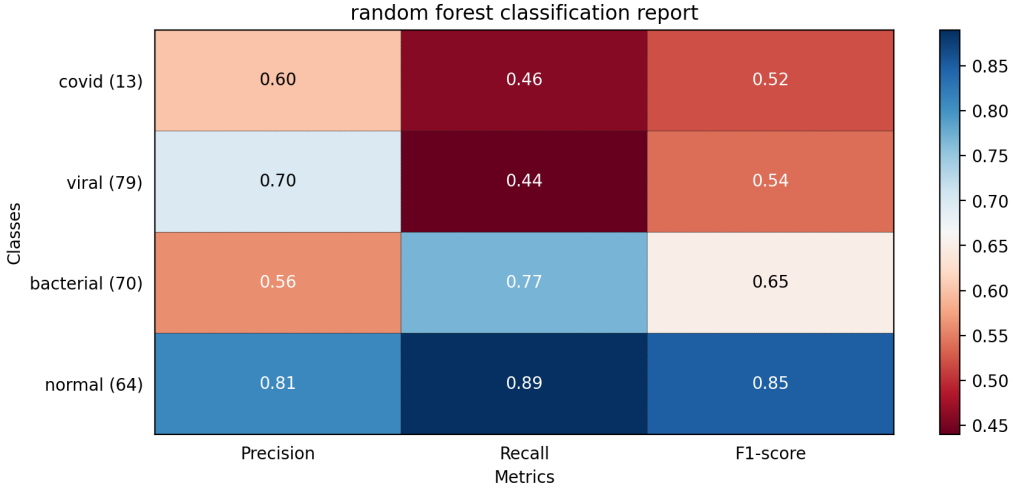
3 Error Analysis

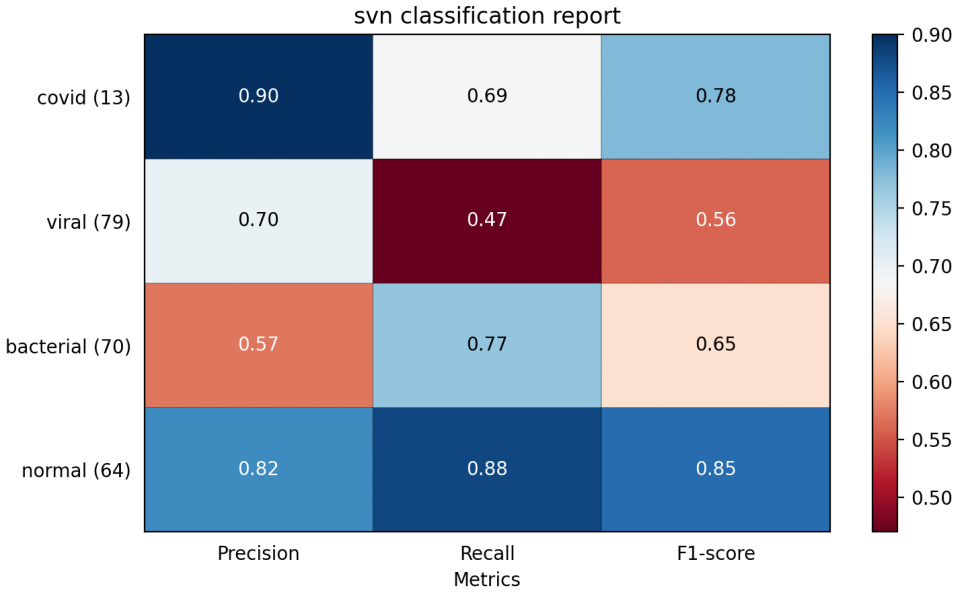
See the classification reports of the various classification methods used on 20 percent validation set below:

As we can see the cnn performed the best with high precision, recall and f1 score on the covid class and high metrics for normal. This would allow a clinical setting to quite accurately classify healthy and covid-infected patients. However, since the viral and bacterial metrics were slightly lower it may be possible for a clinic wanting to diagnose these specific classes to misclassify these and have less accurate diagnoses.




decision tree classification report
covid (13)
viral (79)
bacterial (70)
normal (64)
0.35
0.54
0.42
0.55
0.43
0.48
0.51
0.66
0.57
0.72
0.59
0.65
Precision
Recall
F1-score
Metrics
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35


random forest classification report
Classes
covid (13)
viral (79)
bacterial (70)
normal (64)
0.60
0.46
0.52
0.70
0.44
0.54
0.56
0.77
0.65
0.81
0.89
0.85
Precision
Recall
F1-score
Metrics
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45


svn classification report
covid (13)
viral (79)
bacterial (70)
normal (64)
0.90
0.69
0.78
0.70
0.47
0.56
0.57
0.77
0.65
0.82
0.88
0.85
Precision
Recall
F1-score
Metrics
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50

4 Error Analysis

My CNN model was using a 4 dimensional array shape of the images to classify them, for example, (1127, 128, 128, 1) and used 3x3 kernels, ie. looked at small subsection of the image array at a time. I'm not sure that this method of feature selection would be useful to a human clinician since the pixel values are essentially meaningless without a visual representation. However, they could use these identifed patterns to find more information about the nature of covid and similar or dissimilar it is to other infections in order to prescribe analagous treatments or discard treatments that would have negligible effects.

5 Bugs and Issues

The main issue was obviously time. It took me a lot of time to figure out what kind of shapes (for example of the arrays of images) each classifier took as inputs and such, so data augmentation took a while at first. But mainly, training time. For the VGG16 CNN model, there are multiple layers and each layer has many many neurons where the total trainable parameters were something like 35,698,500. As such, it took about 5 minutes per epoch for 20-ish epochs to train a successful model. Even before reaching this point, I spent a lot of time training less successful models as I was tweaking the parameters and this took a lot of time. In the future, I would spend more time confirming why certain parameters should be to set to certain values and come up with a more systematic method of doing so, so as to not waste a lot of time training non-useful models.