**README**

**B19CSE039\_B19CSE045**

This readme file contains all the important occurrences and steps that are involved in this minor project.

**Preprocessing:**

1. Both datasets containing faces with masks on and faces with masks off were imported using OS library and were stored in two lists as a list of arrays
2. All the images were of different sizes and therefore needed to be reshaped into a fixed format for the models to use. For SVM the reshaping was done into (32,32,3) images and for the MLP and random forest the reshaping was done into (128,128,3).  
   The reshaping was done using the interpolation of the closest pixel format using Inter\_Nearest method as it is [relatively faster](https://stackoverflow.com/questions/3112364/how-do-i-choose-an-image-interpolation-method-emgu-opencv) and speed is what we needed since the dataset was considerably large.
3. Then we took a chunk of 20,000 images of the unmasked set(out of 90,000) and all of the images from the masked set.  
   We did this because at 16GB RAM we were unable to process more than this much data and keep it in memory, also the models already seemed to perform incredibly well with only 20k in the unmasked set which increased the overall accuracy but the confusion matrix revealed that the score on masked was only about 50% which was deceptive, so we decided to only used 20k images from the unmasked set
4. We split the dataset into 50-50 for testing and training and used the training set itself for hyperparameter tuning.

**Model Building:**

**RandomForest:**

1. For the random forest classifier it was first run with default values.
2. Then it was run with entropy as the criterion and improvement was observed.

**MLP:**

1. Two MLP’s were used: One I made and one from the sklearn library
2. For the sklearn MLP: All the default values were used and the training set was the hog set
3. For the custom MLP: First two layers have ReLU as the activation function and the output layer had first sigmoid as the activation function(which is considered generally best for binary classification) but it gave very poor accuracy so then Sigmoid was chosen as the activation function.

Optimizer was chosen as ngd with binary\_crossentropy as the loss to be minimized.

**SVM Classifier:**

1. We can Use SVMs whenever there is a 2 class classification problem.
2. Here , did the following preprocessing methods:
3. Conversion of unknown dimension to appropriate 32x32x3 dimension.
4. We used cv2 with interpolation
5. cv2.resize(mask\_images[i], (32,32), interpolation=cv2.INTER\_NEAREST) for this conversion to maintain uniformity.
6. After this conversion, we can convert this to grayscale image using the following:
7. skimage.color.rgb2gray(img)
8. We need to use skimage and it has a function called rgb2gray that helps in this conversion.
9. Further upon this process,
10. The images are converted to hog format.
11. hog(img,pixels\_per\_cell=(12,12),cells\_per\_block=(2, 2),orientations=5)
12. This allows us to complete the preprocessing task.
13. Upon this, we have created a gridsearchcv using sklearn.
14. Their parameters that was chosen are shown below:
15. svm\_1={'kernel':['linear','rbf','sigmoid'],'decision\_function\_shape':['ovo', 'ovr'],'C':[0.1,1,10]}
16. This data was fitted to hog imaged input data and trained.
17. The best parameters were obtained through gs\_svm\_1.best\_params\_ .
18. GridSearch:
19. The decision\_function\_shape was set to be ['ovo', 'ovr'].
20. The value of 'C' was set to be [0.1,1,10].
21. The ‘kernel' values were chosen to be ['linear', 'rbf', 'sigmoid']
22. From the both grid search, we observe the following :
    1. The best parameters: {'C': 10, 'decision\_function\_shape': 'ovo', 'kernel': 'rbf'}
    2. The maximum accuracy obtained : 0.9356180486481599.