Pattern Recognition and Machine Learning

Minor Course Project

Project Title

Face mask detection

Team Members

Kotha Pranith Reddy

Manav Gopal

Table of Contents

About Dataset

The drive link for the zipped dataset- https://drive.google.com/drive/folders/1pc22wsjDqUb31sDv2Eq_SHXJg7b7ikD7?usp=sharing

Note: We made the data by ourselves taking the masked and unmasked images from the dataset provided.

Masked Face Recognition Dataset is a data set containing different images of persons in which some are wearing masks and some are not. This dataset aims to study the face recognition problem in the mask occlusive environment to detect the person not wearing a mask.

The zipped dataset contains two folders, “masked” which contains the images of the person wearing a mask and “unmasked” which contains the images of the person not wearing the mask.

The masked folder contains approx 1388 images and the unmasked folder contains 2570 images.

OpenCV(Open Source Computer VIsion Library)

OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection. In this tutorial, we explain how you can use OpenCV in your applications.[3]

- OpenCV is a library of programming functions mainly aimed at real time computer vision.[1,4]
- This was originally developed by Intel and supported by Willow Garage and Itseez
- It is free to use under the open source BSD license.[1]
- The library is cross platform and it focuses mainly on real time image processing.[1,4]
- It has 2500+ advanced optimised algorithms for image processing and machine learning.[2]
- It is used for Facial recognition, image manipulation, tracking objects in videos etc.[2,4]

Preprocessing

We have to unzip the folder and extract all the data from that and after extracting all the dataset should be stored in a newly created new empty folder named “Dataset”.

We used opencv to convert all the images into array and stored the features into X and the labels into y.

We had flattened and reshaped the feature data into size of (len(y),64*64) where 64 is the image size and stored in a variable named data.

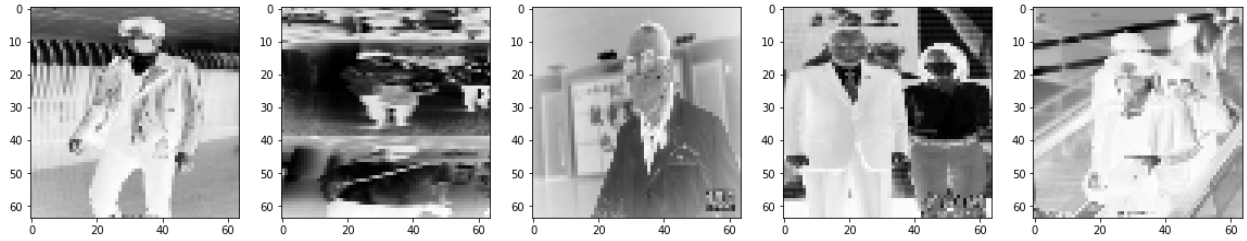
```
data = np.array(X).flatten().reshape(SAMPLE_SIZE, IMG_SIZE*IMG_SIZE)
```

Converted the X and y into a numpy array by reshaping the X using in size(-1, image size=64,image size=64).

Printed the shape of X, y and data.

```
Features, X shape:  (3958, 64, 64)
Target, y shape:  (3958,)
Data shape:  (3958, 4096)
```

Printed some random data using plt.imshow function which takes the array and converts it into image.



Printed the size of the label class in class 0 and class 1, in which the class 0 represents without mask and class 1 represents with mask.

```
No.of Samples: 3958
No.of Without A Mask: 1388
No.of With A Mask: 2570
```

Reshaped the X into total no. of samples and image size*image size.

Train test Split

Splitted the data into a training testing and validation set using a split of 70-20-10. For the splitting of the data we used train_test_split

Classifier Model building

We tried different classifiers and at last used three of the classifiers which are K- nearest neighbour, support vector machines and multi layered perceptron.

We had done a grid search for all the three classifiers to find the best parameters to fit. While performing the grid search, we fitted the grid search model with the validation set obtained from the split.

The parameters taken for the KNN model to pass in the grid search model was `parameters = {'n_neighbors':[1,3,9,15]}`. The obtained best parameter was n_neighbour = 1.

The parameters passed to the grid search for SVM model was `parameters_00 = {'kernel':['rbf','linear','poly'],'C':[0.001, 0.1, 100, 10e5]}`.The obtained best parameter was `{'C': 100, 'kernel': 'rbf'}`

The parameters passed to the grid search for MLP classifier model was `parameters_10 = {'solver': ['lbfgs'], 'max_iter': [100,400,700,1000 ], 'alpha': 10.0 ** -np.arange(1, 10), 'hidden_layer_sizes':np.arange(10, 15)}`. The obtained best parameter was `{'alpha': 0.0001,'hidden_layer_sizes': 13,'max_iter': 1000, 'solver': 'lbfgs'}`

Prediction and Accuracy

Passed the obtained best parameters into all three classifiers and then fitted the model with the training dataset and the made predictions over the testing set using all those three classifiers

Evaluated the classifier score(clf.score) and the accuracy score (accuracy_score).

The classifier Score and the accuracy score, cross val score and classification report for all the three models are as follows:

KNN:

```
Classifier Score : 0.874055
Accuracy Score : 0.8740554156171285
Cross validation score (cv=5): 0.8756934068880973
Classification report :
                              precision    recall  f1-score   support

                           0       0.99      0.63      0.77       133
                           1       0.84      1.00      0.91       264

                    accuracy                           0.87       397
                   macro avg       0.92      0.81      0.84       397
                weighted avg       0.89      0.87      0.87       397
```

SVM:

```
Classifier score : 0.987406
Accuracy Score : 0.9874055415617129
Cross validation score (cv=5): 0.9838284871470713
Classification report :
                            precision    recall  f1-score   support

                         0       0.98      0.98      0.98       133
```

```
                    1       0.99      0.99      0.99       264

             accuracy                           0.99       397
            macro avg       0.99      0.99      0.99       397
         weighted avg       0.99      0.99      0.99       397
```

MLP:

```
Classifier Score : 0.327456
Accuracy Score : 0.327455919395466
Cross validation score (cv=5): 0.6071441341352846
Classification report :
                    precision    recall  f1-score   support

                    0       0.33      0.98      0.49       133
                    1       0.00      0.00      0.00       264

             accuracy                           0.33       397
            macro avg       0.16      0.49      0.25       397
         weighted avg       0.11      0.33      0.17       397
```

Conclusion

The best accuray we got was by using the Support Vectors Machines followed by K nearest neighbour followed by Multilayered perceptron.

Problem Faced During the Project

- Choosing the best classifier was a challenge for us as all the classifiers do not give better results and we cannot be biased that only this classifier will work in this project so we have to search for different possibilities.
- Increasing the accuracy is also a great challenge of any project and so was in our case, we tried various methods to increase the accuracy like adding grid search, standardising the data etc.

Learning from the Project

- By this project we get to know how to deal with the image dataset effectively and how to do preprocessing over that.

- We learned a new concept of using opencv to develop real time computer vision applications. We learned how to use this library image processing, video capture and analysis for face detection.
- The project provided us a deep knowledge of Neural networks and how we can effectively use various Neural nets in order to find the best accuracy.
- This project taught us to deal with the image classification problems and what are the different classifiers we can use for them.
- We also learned how we can resolve errors encountered during the run.
- In order to increase the accuracy we searched and explored various method in that domain which leads us to gain a lot of information

Contribution of Each Member

We tried to distribute the work among all the three members as much as possible and here are the estimated contributions of each group members:

Kotha Pranith Reddy: Made the dataset for the project by making a zip folder containing two folders with masked and unmasked images dataset. Compared different classifiers in order to find out the best classifier which gives the maximum accuracy. Build the SVM and MLP classifier model. Helped in making the final project report.

Manav Gopal: Done all the preprocessing of the data from extraction of the data, using opencv to convert image data into array and reshaping to splitting the dataset into train, test and validation set.Compared different classifier to increase the accuracy and build the KNN model. Made the document of the project report.

Acknowledgement

We would like to express our sincere gratitude to several individuals for supporting us throughout this course and projects. First, we wish to express our sincere gratitude to our supervisors, Dr. Richa Singh, Dr.Yashaswi Verma and Dr. Romi Banerjee for their enthusiasm, patience, insightful comments, helpful information, practical advice and unceasing ideas that have helped us tremendously at all times to understand the course

throughout and develop interest in machine learning. Their immense knowledge, profound experience and professional expertise in Data Quality Control has enabled us to complete this research successfully. Secondly, also we would like to express our gratitude towards the TA's of this course who continuously helped us to build a proper intuition of the course. Without the support and guidance of TA's and professors, this project would not have been possible. We could not have imagined having a better supervisor in my study.

References

[1] RoboCV Module 2: Introduction to OpenCV and MATLAB, SlideShare, roboVITics club, aug, (2012).
[2] A brief introduction to OpenCV, Thomas Pantels, Lynda.com from Linkedin, jan, (2016)
[3] OpenCV Tutorial, tutorialspoint, eBooks, (2021)
[4] Introduction to OpenCV, Luigi De Russis, oct, (2012)