ENEL 645 – Title: Data Mining and Machine Learning

Final Project Report

Project Title: Face Detection Using Siamese Model

Team Members:

Chetana Bijoor	**30111018**
Sanyam	**30118797**
Prabhleen Kaur	**30120534**
Neha Singh	**30109044**
Taruneesh Sachdeva	**30114398**

1. Motivation and Significance of the problem

Face Recognition has become one of the most common features used in mobile applications. Studies show that the newer generation laptops and desktops will also have this feature to ease the lives of people. Although recognizing faces is a challenging job but Siamese Networks (artificial neural networks) provide an efficient solution for the similar problem statements. Convolutional Neural Networks are useful and powerful in most cases but for this project we decided to use Siamese Neural Networks because of the following reasons:

• Siamese Neural Network (SNN) allows the user to choose only a handful of images from each class or from a person as compared to Convolutional Neural Networks (CNN) in which we need a large number of images belonging to different classes.

• Secondly, SNN doesn't classify images but instead learns the similarity functions which are aimed to evaluate the similarity between two objects. Some of the similarity functions correspond to some other previously explained measures, such as cosine distance, while others are statistical or probabilistic, or rely on fuzzy logic.

• Finally, the Siamese Neural Networks, are easy to train and have a simple architecture as compared to the Convolutional Neural Networks, which are not flexible i.e., if the number of people whose faces need to be recognized changes, then the whole model needs to be reconstructed.

2. Defining dataset and pre-processing:

To create training dataset, all five team members added 5 images per person. To increase the training dataset, data augmentation is used using open cv and python. Three types of operations are applied on each image, i.e., clockwise 45-degree rotation, anti-clockwise 45-degree rotation, and adding noise to the images, thus resulting in total of 20 images per person.

For the test dataset, 18 images are collected from the 5 team members. After collecting the test images, data augmentation is done and two types of operations are applied to the images i.e., clockwise 45-degree rotation, and anti-clockwise 45-degree rotation, thus resulting in the total test data set of 54 images. So, in total 100 images are added as the training dataset and 54 images are added as a test dataset.

After defining the datasets, the images are loaded and resized for the defined dimension of (128, 128) pixels using Python Imaging Library. This results in the shape of each image as (128, 128, 3). To create an actual training dataset, the defined training images are then converted to NumPy arrays such that each image is paired with every other image and a label is assigned to each pair. Since training dataset consists of 100 images, the above-mentioned combination results in training dataset of 10,000 pair of images.

The label assigned to each pair of images is dependent on if the images belong to the same class or not. If both images in a pair belongs to the same class, then the label assigned is 1, else the label assigned is zero. The images stored in above defined NumPy arrays are x1.npy and x2.npy files. Each of the 'x1.npy' and 'x2.npy' contains an image from each pair, such that nth image from each file, makes the nth pair with corresponding label stored at the nth position in 'y.npy' file. Thus, as a result, we have 2 sets of images, X1 and X2 of shape (10000, 128, 128, 3) and 10000 labels.

The above obtained NumPy arrays and labels are then shuffled and divided into the training and the validation dataset such that the size of the training dataset to the validation dataset is 8000:2000. The defined model is then trained on this dataset for 100 epochs with the aim of minimizing the validation loss.

3. Methodology

Model implementation methodology used to build the model is precisely described in the below image:

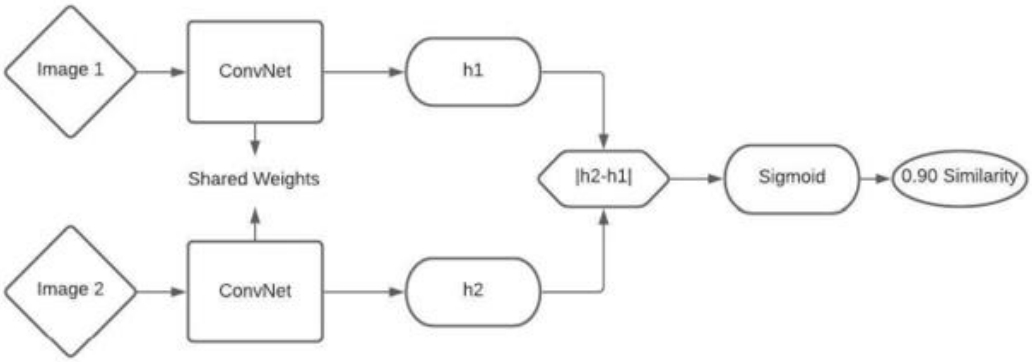

Figure 1: Siamese Model Architecture

Siamese Model for the face detection is built using the sequential model approach. Four convolution layers along with the two max pooling layers are used for building the model, with 'LeakyReLu' as the activation function. Two instances of the model are then constructed with the exact same hyperparameters and same input layers. The outputs from the two instances of the models for each image in a pair of input images is then applied to the dense layer with 'sigmoid' as the activation function. The output of the sigmoid function is the similarity score between the two images, based on which our model learns the similarity between the two images.

The above model defined is then trained on the training dataset of images as described above. Once the model is trained, it is then tested on the test data. The test data is a set of 54 images belonging to the different classes. All the test images are passed to the trained model to calculate the similarity score. For each image, the similarity scores are calculated corresponding to each image in the training dataset. Since, there are total of 100 images in training dataset, each test image is compared against each training image and a similarity score is calculated. Thus, for every test image 100 similarity scores are calculated. Based on the similarity scores thus obtained, the image is then classified into the class, against which the highest similarity score is observed.

Training of the model

Model is trained with the different batch size and epochs. The best score is achieved with a batch size of 6 and 100 epochs. Further, a patience with value 10 is used along with the monitoring of the minimum validation loss. Further, the performance of the model is also noted by varying the split ratio of training and validation data, and the model performed best when the ratio of training and validation data is 8000:2000.

4. Results

Training dataset consists of a pair of images with each pair having a label with it. If both the images in a pair belong to the same class, then its label is 1, else its label is zero. The images are stored as NumPy arrays in x1.npy and x2.npy files. Each of the 'x1.npy' and 'x2.npy' contains an image from each pair, such that nth image from each file, makes the nth pair with corresponding label stored at the nth position in 'y.npy' file.

These three files are successfully getting created on the execution of the program. Based on these files, the model is trained and tested. We tested our model for 54 test images. Corresponding results shows the best confidence scores of each image along with the class into which the image is classified.

Predicted vs Actual images along with confidence scores

Below image shows the labels of the class to which image belongs (First Label) against the label of the image predicted by the model (Second Label). The confidence score is the score calculated by the model between the predicted image and the actual image.

```
IMAGE 2 is 2 with confidence of 0.8362138271331787  IMAGE 4 is 3 with confidence of 0.5029616355895996
IMAGE 2 is 2 with confidence of 0.6828592419624329  IMAGE 4 is 4 with confidence of 0.8199065327644348
IMAGE 0 is 0 with confidence of 0.856453001499176   IMAGE 4 is 1 with confidence of 0.45158079266548157
IMAGE 0 is 4 with confidence of 0.5170267224311829  IMAGE 1 is 2 with confidence of 0.6533607244491577
IMAGE 4 is 3 with confidence of 0.45496025681495667 IMAGE 1 is 2 with confidence of 0.7311225533485413
IMAGE 4 is 2 with confidence of 0.435358464717865   IMAGE 1 is 1 with confidence of 0.6144034266471863
IMAGE 1 is 3 with confidence of 0.5158643126487732  IMAGE 1 is 3 with confidence of 0.659569501876831
IMAGE 1 is 0 with confidence of 0.5769051313400269  IMAGE 3 is 2 with confidence of 0.6999387145042419
IMAGE 3 is 3 with confidence of 0.721704363822937   IMAGE 3 is 3 with confidence of 0.5100961327552795
IMAGE 3 is 3 with confidence of 0.6368738412857056  IMAGE 3 is 2 with confidence of 0.7472109794616699
IMAGE 0 is 4 with confidence of 0.6924153566360474  IMAGE 3 is 3 with confidence of 0.6873370409011841
IMAGE 3 is 1 with confidence of 0.4345649182796478  IMAGE 0 is 1 with confidence of 0.863449215888977
IMAGE 3 is 3 with confidence of 0.7542691230773926  IMAGE 0 is 2 with confidence of 0.6173081994056702
IMAGE 2 is 2 with confidence of 0.44226354360580444 IMAGE 3 is 3 with confidence of 0.5495609045028687
IMAGE 2 is 2 with confidence of 0.6952904462814331  IMAGE 3 is 3 with confidence of 0.6111175417900085
IMAGE 1 is 1 with confidence of 0.7839048504829407  IMAGE 3 is 0 with confidence of 0.5781177878379822
IMAGE 1 is 1 with confidence of 0.615827739238739   IMAGE 3 is 3 with confidence of 0.5765010714530945
IMAGE 4 is 4 with confidence of 0.6341765522956848  IMAGE 2 is 1 with confidence of 0.4948279857635498
IMAGE 2 is 4 with confidence of 0.5415031313896179  IMAGE 2 is 2 with confidence of 0.6773273348808289
IMAGE 2 is 2 with confidence of 0.7906201481819153  IMAGE 2 is 4 with confidence of 0.575417697429657
IMAGE 2 is 3 with confidence of 0.3494266867637634  IMAGE 2 is 2 with confidence of 0.7531541585922241
IMAGE 2 is 2 with confidence of 0.8006615042686462  IMAGE 1 is 1 with confidence of 0.3350374102592468
IMAGE 0 is 0 with confidence of 0.7146273851394653  IMAGE 1 is 2 with confidence of 0.5337969660758972
IMAGE 0 is 0 with confidence of 0.5635249614715576  IMAGE 1 is 1 with confidence of 0.5341705679893494
IMAGE 0 is 4 with confidence of 0.5950427651405334  IMAGE 1 is 1 with confidence of 0.44557514786720276
IMAGE 0 is 0 with confidence of 0.6448196172714233  IMAGE 4 is 4 with confidence of 0.792978048324585
IMAGE 4 is 3 with confidence of 0.4139559268951416  IMAGE 4 is 0 with confidence of 0.5701799392700195
```

Figure 2: Comparison scores of the images

Confusion Matrix

Below confusion matrix shows the true and false labels corresponding to different classes. Class Labels 0,1,2,3,4 corresponds to the images of five persons, i.e., Chetana, Prabhleen, Sanyam, Taruneesh, and Neha, respectively. In the below matrix, cell with the same row and same column represents the correctly classified images of the person corresponding to that label. For instance, in the first cell i.e. (0,0), '4' represents that four images of Chetana are correctly classified as 'Chetana' by the model. However, other numbers in the first row represents the number of images of 'Chetana' incorrectly classified as the person

corresponding to the label equivalent to the column number. For instance, one image of Chetana is classified as 1 i.e., Prabhleen. Similarly, one image as Sanyam and three images as Neha. Similarly, confusion matrix can be read for other class labels.

Class_Labels	0 1 2 3 4
0	[[4 1 1 0 3]
1	[1 6 3 2 0]
2	[0 1 8 1 2]
3	[1 1 2 8 0]
4	[1 1 1 3 3]]

Figure 3: Confusion Matrix

Classification Report

Below image represents the classification. First five rows give the precision, recall, f1-score for the individual classes. 'Support' represents the number of test images belonging to a particular class. Last three rows in the image gives the stats corresponding to the micro, macro, and average precision, recall, and f1-scores. So, according to the micro scores, overall accuracy of the model achieved is 54%.

	precision	recall	f1-score	support
Chetana	0.57	0.44	0.50	9
Prabhleen	0.60	0.50	0.55	12
Sanyam	0.53	0.67	0.59	12
Taruneesh	0.57	0.67	0.62	12
Neha	0.38	0.33	0.35	9
accuracy			0.54	54
macro avg	0.53	0.52	0.52	54
weighted avg	0.54	0.54	0.53	54

Figure 4: Classification Report

Predicted vs Actual Images

Below images show the custom images along side the predicted images. Custom images represent the test dataset and predicted images are the images which got the highest score among all the images of the trained dataset when compared against the passed test image.

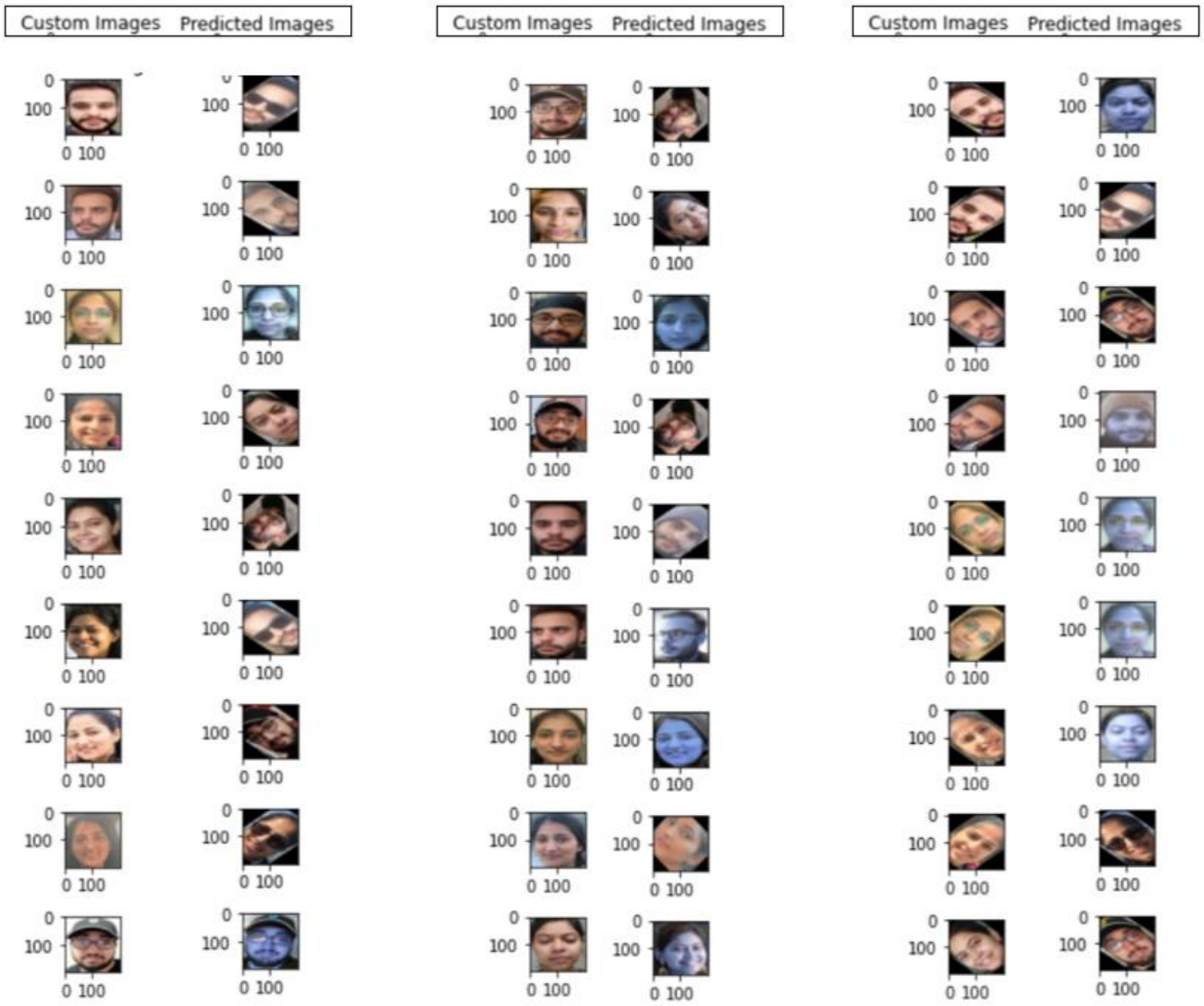

Figure 5: Actual Images vs Predicted Images

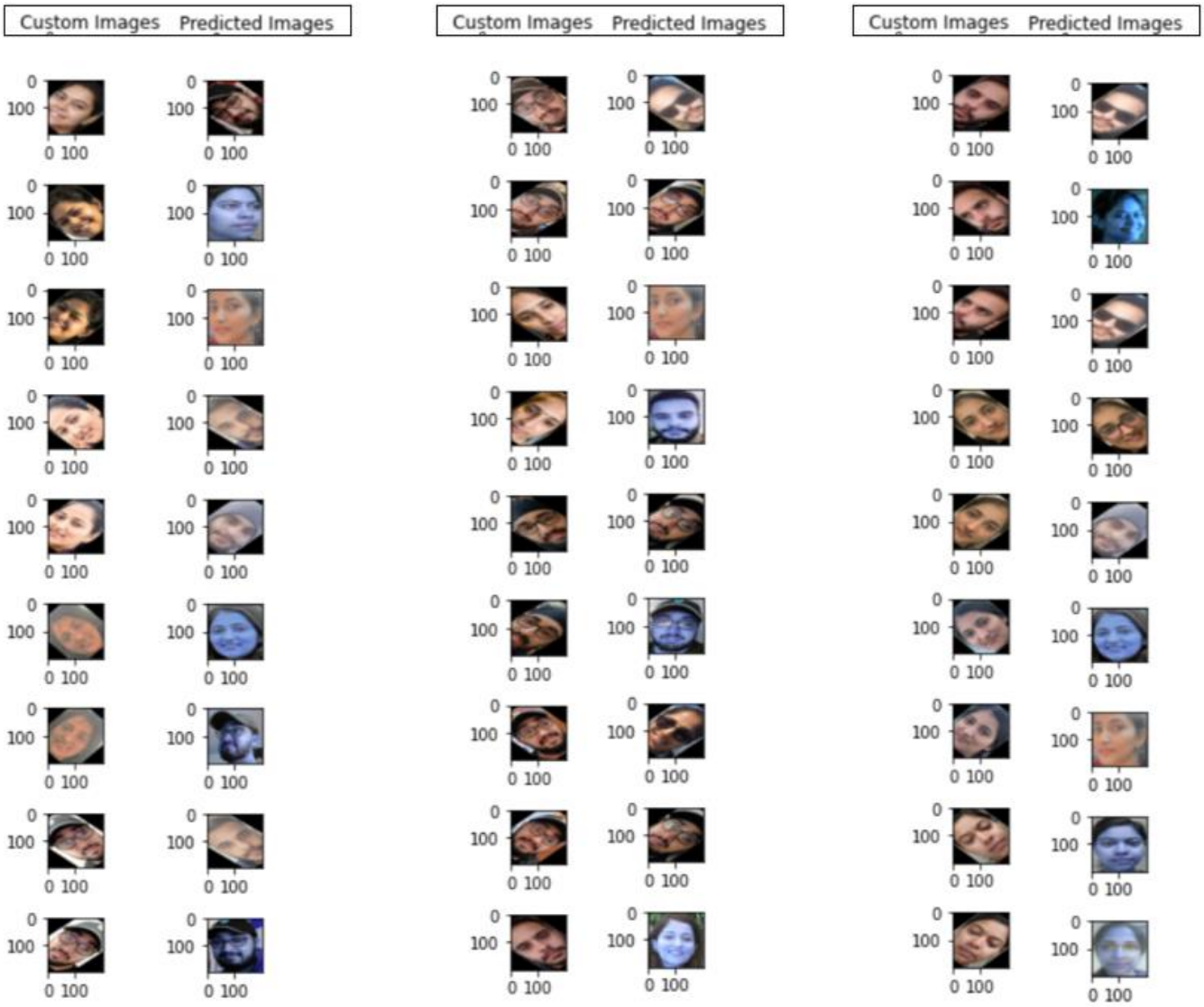

Figure 6: Predicted Images vs Actual Images

Challenges Faced

- The approach followed by the team to implement the face detection is by converting the images to the NumPy arrays. This approach is slow and takes a lot of time in building the NumPy arrays. However, team was able to implement the project using a dataset of around 10,000 pair of images, on which model is trained.
- Google Colab has a limit on the number of GPU runtime resources we can use, thus training a model with different number of possibilities for longer duration was difficult.

GitHub Link

Below is the GitHub link of the source code files and dataset collected:

https://github.com/sanyam01/Enel645_FaceDetection_SiameseModel.git

Below is the description of the relevant the folders on GitHub:

- 'Test_Images' folder contains the test dataset.
- 'augmented_dataset' folder contains the training dataset i.e., 20 images belonging to each team member.
- 'images_dataset' folder contains the unaugment training data i.e., 5 images belonging to each team member.
- 'src_files/Augment_Test_Data.ipynb' and 'src_files/Data_Augment.ipynb' includes the code for the generation of augmented test and the training data, respectively.
- 'src_files/SiameseModel.ipynb' includes the build Siamese model.
- 'src_files/DataPreProcessFile.ipynb' includes the code for the generation of the NumPy arrays required for training the model.
- 'src_files/MainFile.ipynb' includes the code for training the model and predicting the images.

For implementation, first run the 'DataPreProcessFile.ipynb' which will create the required NumPy array files. Then, run the 'MainFile.ipynb'.

6. References

- https://medium.com/@enoshshr/learning-similarity-with-siamese-neural-networks
- https://github.com/adityajn105/Face-Recognition-Siamese-Network
- https://github.com/shubham0204/Face_Recognition_with_TF
- https://medium.com/predict/face-recognition-from-scratch-using-siamese-networks-andtensorflow-df03e32f8cd0
- https://en.wikipedia.org/wiki/Siamese_neural_network
- https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7
- https://towardsdatascience.com/siamese-networks-line-by-line-explanation-for-beginners-55b8be1d2fc6
- https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826
- https://towardsdatascience.com/image-augmentation-using-python-numpy-opencv-and-skimage-ef027e9898da

7. Consensus Score

Broadly speaking, Sanyam, Taruneesh, and Neha focused on building the model and Prabhleen and Chetana focused on data collection and pre-preprocessing. Then, a main file is built which is done together as a team. So overall, the project is a collaborative team effort. All the team members contributed equally to the project.

Hence, the consensus scores are as follows:

Name	Consensus Score
Chetana Bijoor	3
Neha Singh	3
Prabhleen Kaur	3
Taruneesh Sachdeva	3
Sanyam	3