Project Title: Face Detection Using Siamese Model

Team Members:

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

1. <u>Motivation and Significance of the problem</u>

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 with the use of Siamese Networks, which are artificial neural networks provide an efficient solution for similar problem statements.

Convolutional Neural Networks are useful and powerful in most cases but for this project we decided to use Siamese Neural Networks for 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 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, on one hand, Convolutional Neural Networks are not flexible i.e., if the number of people whose faces are required change, the whole model needs to be reconstructed, Siamese Neural Networks, on the other hand, are easy to train and have a fairly simple architecture as shown in the other section of the proposal.

2. <u>Timeline for Project Completion</u>

We are aiming to finish data preparation and pre-processing by midterm. We will further be focusing on model training and evaluation which we will finish by the final presentation.

- March 13th, 2021 – Midterm progress report and oral presentation; Completing report on the data preparation and pre-processing and further building the Siamese model based on our dataset. We will then prepare the presentation for the preliminary results achieved by our team till date and future steps to conclude the project.
- April 15th, 2021 – Final project report and oral presentation; Building the model, training and performance evaluation. Based on the performance of the model, we will re-tune the model if required to achieve better results.

3. <u>Projected workload split between team members</u>

The project will be completed by all the team members in a collaborative manner. The projected workload includes data preparation and pre-processing on which two team members will work. Three team members will be working on model selection, training and evaluation. Equal effort will be put by all the team members towards project report creation and other documentation.

4. <u>Consensus Score for each team member</u>

All the team members contributed equally to the project proposal. Hence, the consensus scores are as follows:

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

5. <u>**Methodological details:**</u>

Model selected and approach

First the data set will be collected and then pre-processed if necessary. Based on the collated dataset, a Siamese model will be trained to perform the face detection. The general algorithm used behind the face detection is as follows:

- The two images to be compared are fed to a single Convolutional Neural Network (CNN).
- The last layer of the CNN produces a fixed size vector (embedding of the image). Since two images are fed, it produces two embeddings as the output i.e., h1 and h2, as shown in the below diagram.
- The Euclidean distance between the vectors is calculated.
- The values then pass through a sigmoid function and a similarity score is produced.

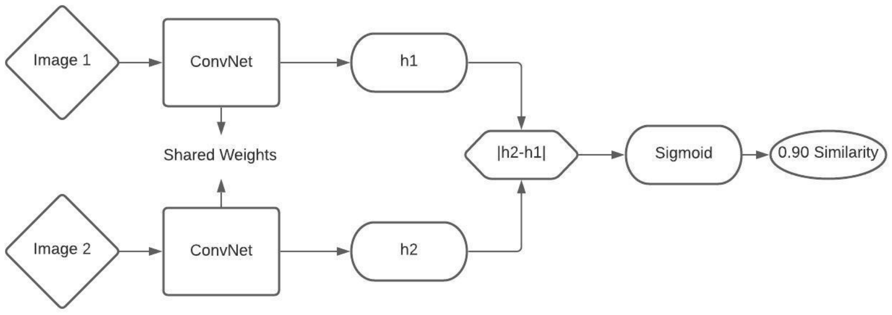

Figure 1: Siamese Model Architecture

- Similarly, the similarity score is calculated between all the stored images in the dataset and the image to be classified. The similarity scores thus calculated are then compared against each other. The pair of images with the highest calculated score is considered as the similar images.

Datasets to be used

We intend to build our own dataset for this project containing the images of our teammates taken with the help of a webcam. These images can be taken with different backgrounds, facial expressions etc. After building this dataset, we divide it into Training, Validation and Test sets using a 60-20-20 split. The training dataset is used to fit the model while the validation set is used to evaluate the model while tuning model hyperparameters. Finally, the test set is used once the model is completely trained and it provides an unbiased evaluation of a final model fit on the training dataset.

Data Pre-processing

We will be using CameraX, Firebase MLKit, and TensorFlow Lite for analysing the images and processing them. To convert the Keras model to TLFLite, we will use TFLiteConverter API. We will retrieve camera frames from the device for image analysis. Using Firebase MLKit, we'll get bounding boxes for all faces present in the camera frame. We will use the FaceNet model to produce face embeddings by cropping the camera frame using the bounding box which we got from Firebase MLKit. The cropped image will be transformed from a Bitmap to a ByteBuffer with normalized pixel values. The ByteBuffer will be fed to the FaceNet model. We will have a customised class that would return us the 128-dimensional embedding for all faces present in the given image. we'll produce face embeddings and compare each of them with a set of embeddings that we already have.

Metrics to assess the results

Since our output is binary {0, 1} in nature, we use the Binary Cross-entropy loss function. Cross-entropy will calculate a score that summarizes the average difference between the actual and predicted probability distributions for predicting class 1. The score is minimized, and a perfect cross-entropy value is 0. The final activation function to be used in our model is the Sigmoid function. Sigmoid results in a value between 0 and 1 which we can infer to be how confident the model is of the example being in the class. The model can be further evaluated using Classification Accuracy. Accuracy is the ratio of number of correct predictions to the total number of input samples.

Model Retraining

To improve the prediction results we may choose to fine tune the model. With fine-tuning, we first change the last layer to match the classes in our dataset, as we have done before with transfer learning. But besides, we also retrain the layers of the network that we want. With fine-tuning we are not limited to retaining only the classifier stage (i.e., the fully connected layers), but we can also retrain the feature extraction stage, i.e., the convolutional and pooling layers. Since our dataset is small and the new dataset would be different from the original one, we can use features of an earlier layer of the convolutional stage, as this will be set in more general patterns than the later layers, and then use a linear classifier in our model.

6. References

https://medium.com/predict/face-recognition-from-scratch-using-siamese-networks-and-tensorflow-df03e32f8cd0

https://en.wikipedia.org/wiki/Siamese_neural_network

https://www.eurekaselect.com/130664/chapter/similarity-functions-for-face-recognitio

https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7

https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/

https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

https://towardsdatascience.com/deep-learning-which-loss-and-activation-functions-should-i-use-ac02f1c56aa8

https://towardsdatascience.com/using-facenet-for-on-device-face-recognition-with-android-f84e36e19761

https://towardsdatascience.com/cnn-transfer-learning-fine-tuning-9f3e7c5806b2