**Face-Recognition**

Training Siamese model to create a one-shot neural network model using Face-Net and MTCNN as the backbone. Yolov5 and OpenCV are used alongside to classify faces in the webcam feed. We are trying to implement face recognition with very little memory usage by switching between yolov5 and MTCNN when necessary. The target of this repository is to use run facial recognition in the background to ensure user privacy and protection while leaving more than enough memory for the user to continue using the device for personal uses.

## **Algorithm for Training Siamese Model:**

We are using the Olivetti dataset which contains images of faces of 40 different people with 10 images for each person. We use this dataset and resize each face to 128×128 image which is then normalized and fed into pre-trained InceptionResnetV1 (vggface2 dataset weights) which returns embeddings of shape 1×512. These embeddings represent the encoded features for an image. We then take embeddings of any two images and feed them into the Siamese model where the model returns a value between 0 and 1 which represents how similar are the images that were fed. We are only training the Siamese model as the InceptionResnetV1 is already trained

## **Working System:**

MTCNN takes up most of the memory if it directly feeds the webcam feed. To solve this problem, we are using the Yolov5 model to detect if there are any people in the webcam feed and if there is any change in the number of people in the webcam feed. If there is a change we are then using MTCNN network to detect faces to create boxes for the faces which are used to crop and resize the faces to the size of 128×128 which is then normalized and fed into InceptionResnetV1 giving us target embeddings. Similarly, these image embeddings are previously saved for reference images that are present in each class in the database while initializing the models so that they can be loaded directly in the future. This target and reference embeddings are fed into the Siamese model which finds the similarity between the target and each reference image. The reference image with the highest similarity that crosses the minimum threshold is the predicted class. While using the Yolov5 model there might be some slight error in accuracy giving us extra boxes with low person probability which might trigger MTCNN frequently causing a drop-in fps and increased memory usage. To solve this problem, we are using a cooldown timer to check if the change is real or if it's due to an error caused by the low accuracy of the Yolov5 model. We are also using IOU to match the boxes generated by Yolov5 and MTCNN so that the classified classes are matched with their respective bounding boxes.