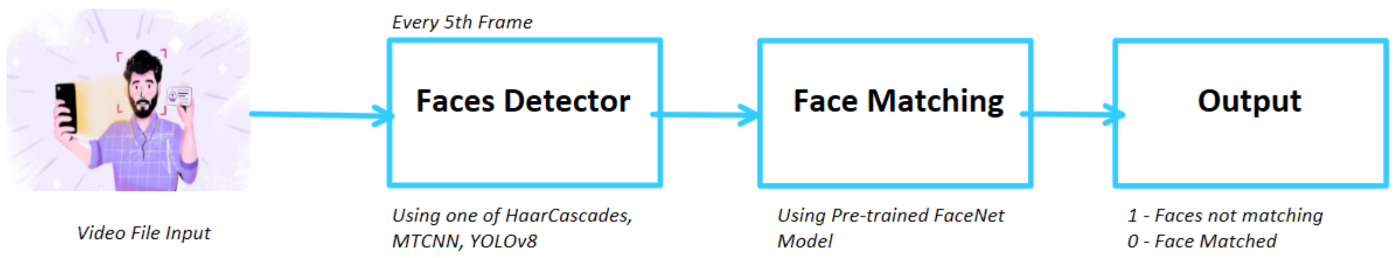
FACE VERIFICATION (Take Home Assignment – Veriff)

Problem – Given a MP4 Video file with one or more persons appearing up, detect and classify if the faces are of same one person or different faces are visible.

Approach –

First Approach: Face Detect + Face Matching
The main goal of the app is to identify if there are more than one person seen in a video. In a nutshell, if we found more than 2 faces in a frame, we can say another person might be in the video. If less than 2 faces are being detected in a frame then it is safe and we can continue looking in next frame. And if we find two faces in a given frame, we would then pass it to next step – Face Matching where we predict if two faces are of same person or not. This will ensure that both faces are of same person (person itself and their ID Card).



Technologies Tested-

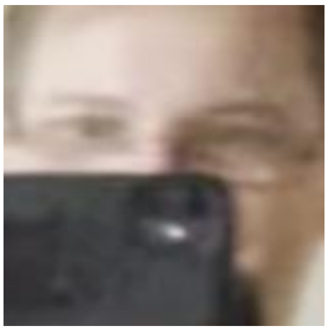
	Face Detector	Face Matching Model
1	*OpenCV - HaarCascades* basics of face detection using Haar Feature-based Cascade Classifiers	*FaceNet* TensorFlow implementation of the face recognizer described in the paper "FaceNet: A Unified Embedding for Face Recognition and Clustering"
2	*MTCNN* one of the most popular and accurate face detection tools today. It consists of 3 neural networks connected in a cascade.	

Advantages/Disadvantages –

FaceDetectors-

OpenCV – Haar based Cascade Classifiers are also ML based method that does detect faces swiftly. However, it can miss some angled faces and also can miss faces with some obstruction.

MTCNN – This is the most accurate in detecting all faces in given frames. Below example of face detection displays the accuracy of face detector which *OpenCV-Face Detection* model fails to detect the same.



Major disadvantage is that it takes quite some time to detect faces in an image compared to method 1.
Total Run times are compared as below-
Video veriff12.mp4 (16 secs)
MTCNN *takes* **1:01**
OpenCV *takes* **0:34**
MTCNN can be preferred for accuracy and some corner cases like above mentioned sample, while simple video can be easily predicted using OpenCV's Face Detection method.

Face Matching Model –

Utilizing *Facenet* Embeddings for Face Verification:

1. Encoding Faces:

   Facenet is used to extract embeddings from facial images. These embeddings represent the unique features of each face in a high-dimensional space.

2. Distance Calculation:

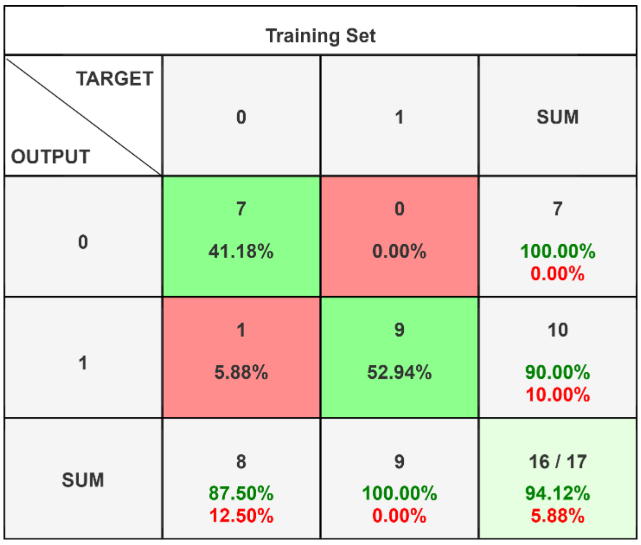
   The Euclidean or cosine distance between the embeddings of two faces is computed. If the distance is below a certain threshold, the faces are considered to belong to the same person.

Results –

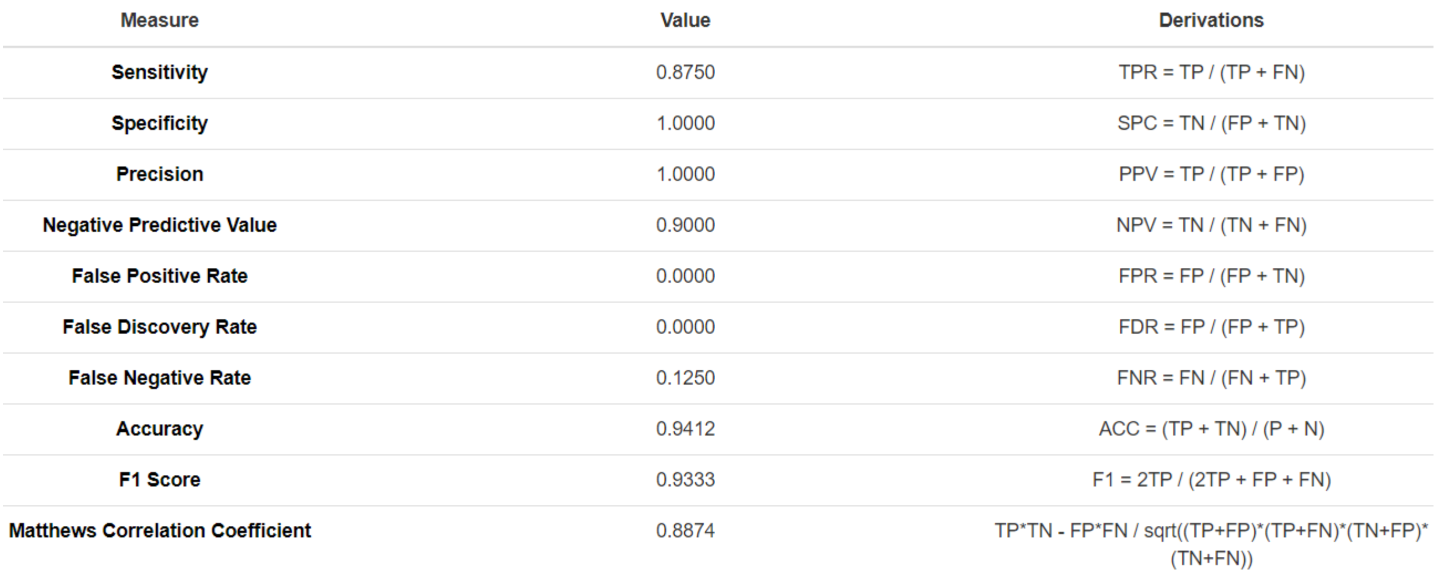
For all video samples provided, following table shows the performance of MTCNN + FaceNet based approach :

Video	Truth	Predicted
veriff1	0	0
veriff2	1	1
veriff3	0	0
veriff4	1	1
veriff5	0	0
veriff6	0	0
veriff7	0	0
veriff8	0	1
veriff9	1	1
veriff10	1	1
veriff11	0	0
veriff12	1	1
veriff14	0	0
veriff16	1	1
veriff17	1	1
veriff18	1	1
veriff19	1	1

Confusion Matrix -



Training Set			
TARGET / OUTPUT	0	1	SUM
0	7 41.18%	0 0.00%	7 100.00% 0.00%
1	1 5.88%	9 52.94%	10 90.00% 10.00%
SUM	8 87.50% 12.50%	9 100.00% 0.00%	16 / 17 94.12% 5.88%



Measure	Value	Derivations
Sensitivity	0.8750	TPR = TP / (TP + FN)
Specificity	1.0000	SPC = TN / (FP + TN)
Precision	1.0000	PPV = TP / (TP + FP)
Negative Predictive Value	0.9000	NPV = TN / (TN + FN)
False Positive Rate	0.0000	FPR = FP / (FP + TN)
False Discovery Rate	0.0000	FDR = FP / (FP + TP)
False Negative Rate	0.1250	FNR = FN / (FN + TP)
Accuracy	0.9412	ACC = (TP + TN) / (P + N)
F1 Score	0.9333	F1 = 2TP / (2TP + FP + FN)
Matthews Correlation Coefficient	0.8874	TP*TN - FP*FN / sqrt((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN))

Final Comments-

1. Overall, this method is robust in predicting and matching faces, hence making this a good baseline approach for Real Implementation.
2. I assumed here that one person will only show one ID Card, and hence max 2 similar faces to track and record. If the idea is that only one person should be tracked from beginning and must focus on one person facial representations only then the discussed method might fail. For example, if I start the video (1 face is detected) and I leave the camera view and another person comes in (again 1 face detected). So this will be considered as safe as only 2 face was detected.
3. Total 3 face mismatches needed to verify as Verification Failed for allowing app to read faces again if not clear in initial frames.

Future Development –

1. If given more time and hundreds of such videos, I would suggest to turn the focus majorly on capturing all True Positives as many as possible as it is more secure way to verify. Also, I would majorly focus on reducing the total run-time. As I expect such app to run and predict in real-time and probably matching the frames as video is being generated.