Introduction to Artificial Intelligence

COSC 330

Facial Recognition on a Uniform-Appearance Crowd

Members:

Ahmad Rashid #53924

Yousef Al Nashef #53763

Peter-John Hein #53648

Nawfal Espel #53816

Date: December 6, 2022

Instructor: Dr. Naoufel Werghi

Table of Contents

Introduction

Facial recognition is a widely used technology in the modern world. It is used in our daily lives whenever we use our mobile devices. In the Western world recognition systems world well because individuals dress differently which assists in finding distinguishing features for classification. In the Middle East uniform-crowd appearance is a great challenge. We have used available systems and state of the art technologies to find a solution to this problem.

Problem Statement

Person detection and identification have numerous applications in surveillance, human-computer interaction, and security. Performing this task in a uniform-appearance crowd, often encountered in the Middle East and the gulf-region is a challenge. Up to now, there is no such system that can handle such a task effectively. A proposed solution is to use a camera-equipped drone-based system. This system is invasive, having a nearly zero risk of collision, and most importantly has more flexibility in tracking when it comes to a computer vision-based system. The scope of this project is to develop, implement and validate a deep learning vision model for the proposed solution.

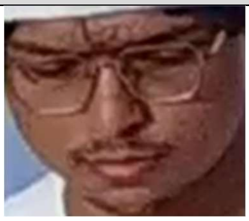
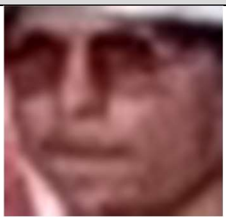
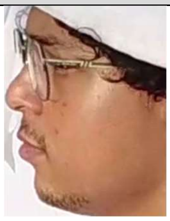
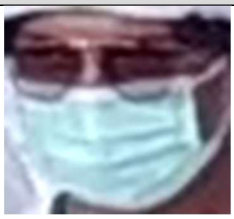
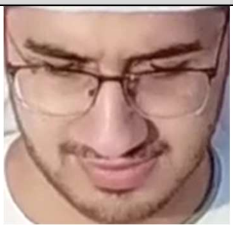
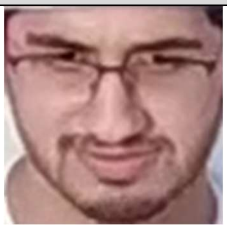
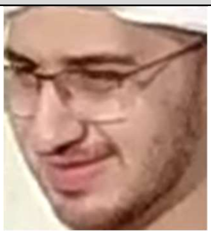
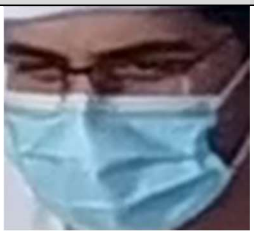
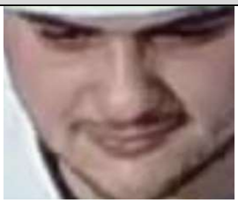
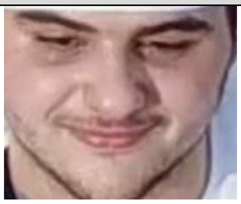
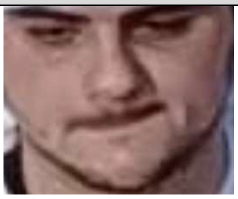
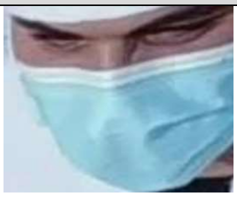
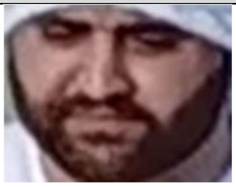
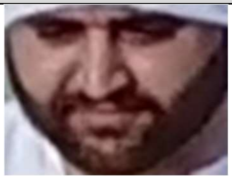
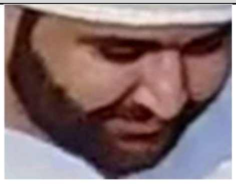
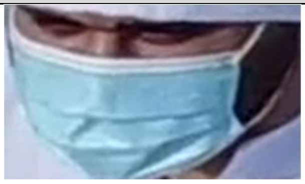
Solution

First, the dataset must be prepared. To get the dataset ready, the class had scheduled multiple sessions, where numerous videos of us wearing uniform clothes and walking as a crowd were recorded. After the dataset was ready, each team member had to choose at least one video with different challenges and annotate himself. Then, the team had to look up libraries and algorithms on face detection and face recognition. After the face detection and recognition is done, the team must validate, analyze, and compare the results gotten from the different challenges.

Annotations

The team used CVAT to do the manual annotations. The team focused on having uniform annotations by having only the face annotated without including the ears, from the chin up until the start of the ghutra, providing us with precise face annotations.

Table 1: Manual Annotations

Peter			
Normal	**Intruder**	**Random**	**Mask**
			
Ahmed			
Normal	**Intruder**	**Random**	**Mask**
			
Yousef			
Normal	**Intruder**	**Compact**	**Mask**
			
Nawfal			
Normal	**Compact**	**Random**	**Mask**
			

Methodology

For each selected video, face detection would be required. First, the frames would need to be extracted. Then, the face detection algorithm would be run on each extracted frame to get the corresponding extracted faces. The team used a face detection algorithm from the retina-face library. The retina-face library provides a function called extract_faces(). The function takes in an image, or one frame from a video in our case, as an input and then outputs at least one image for each extracted face and the bounding box for the face, which will be used later for validation. The team also set the 'align' flag to true when using the function, which transforms the extracted face image upright when necessary. The team had to run this function across all the frames from the videos the team annotated.

Furthermore, after the detection, recognition can be done. The team used a recognition algorithm from the DeepFace library. A query image is used and compared to each extracted face and the DeepFace algorithm will output a cosine indicating the distance between the compared faces. The exact function implemented was find(), which takes the path of the query image and the path of the directory for all the extracted faces. Once completed, the distances for all the faces are returned. All of them are then compared to a threshold value of 0.65, which is the default value for the ArcFace recognition model. All faces with a cosine below this value would be considered a positive result, and all faces above this value would be a negative result.

After the face recognition was done, the team had to validate the results. To validate the results, the team compared the pixel coordinates of the annotations the team did with the pixel coordinates of the face detection that corresponds to the images in the sets of positives and negatives returned by the DeepFace algorithm. The positives would need to be separated into true positives and false positives. A true positive occurs when there is a correct match, the person is the same as in the query image and the cosine distance is below the threshold. A false positive occurs when there is a mismatch, the person is not the same as in the query image, but the cosine distance is below the threshold. A true negative occurs when there is a correct match, the person is not the same as in the query image and the cosine distance is above the threshold. A false negative occurs when there is a mismatch, the person is the same as in the query image, but the cosine distance is above the threshold. The team decided on a margin of error when comparing the pixel coordinates, if there is a greater difference of 20 pixels, it would be considered a mismatch, otherwise, it would be a match.

Challenges

To evaluate the team's face identification algorithm and find its limitations the team needed to input datasets with a wide range of difficulties. To achieve such a range, the team chose four videos which vary in settings. To set a control for our tests, all team members chose at least one unchallenging dataset so that it can be used as a reference point for the other inputs. This meant that the member's face had to be visible for most of the video and the crowd had to be moderately separated. For the other 3 videos the team decided to go with videos that challenge the identification algorithm to see how far they affect its accuracy and precision hence the members went with video settings including intruders, compact formations, masked faces, and random movement.

Results

For each case, the team found the metrics on precision, recall, F1, and accuracy to analyze the results. Precision is a metric that illustrates a percentage of how many out of the total positives are true positives. Recall is a percentage of faces that are recognized out of all faces that should have been recognized as you. F1 is a weighted average of precision and recall. Accuracy considers all the recognitions and is the percentage of all correct classifications of positives and negatives. Accuracy should be taken with a grain of salt because of how the classes are unevenly distributed. The large offset of true negatives gives a false sense of results which is why F1 should be preferred in this case. A figure is provided on the formulas used to get the metrics used for analysis.

$$\boldsymbol{recall} = \frac{\boldsymbol{true\_positives}}{(\boldsymbol{true\_positives} + \boldsymbol{false\_negatives})}$$

$$\boldsymbol{precision} = \frac{\boldsymbol{true\_positives}}{(\boldsymbol{true\_positives} + \boldsymbol{false\_positives})}$$

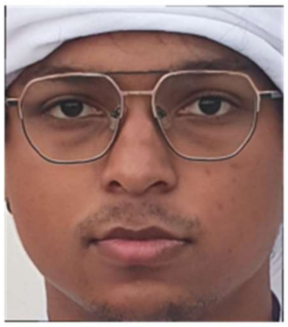
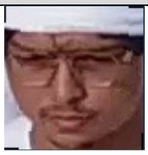
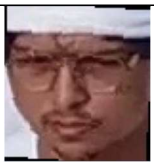
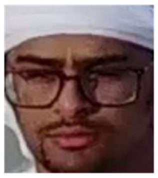
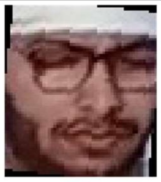
$$\boldsymbol{accuracy} = \frac{(\boldsymbol{true\_positives} + \boldsymbol{true\_negatives})}{(\boldsymbol{true\_positives} + \boldsymbol{true\_negatives} + \boldsymbol{false\_positives} + \boldsymbol{false\_negatives})}$$

$$F1 = 2 \times \frac{precision \times recall}{(precision + recall)}$$

Peter

Normal Challenge:

Table 2: Peter normal challenge results

Query Image		
		
	Images	**Cosine Value**
True Positive 1		0.48915291415314976
True Positive 2		0.4291882645873416
False Positive 1		0.6460638141299603
False Positive 2		0.630940445335768

True Negative 1		0.877667549608466
True Negative 2		0.8916154552293568
False Negative 1		0.9249097851910646
False Negative 2		0.7518203403466369

Table 3:Peter normal challenge performance measurement

Performance Measurement	
Metric	**Value**
Total Positives	571
Total Negatives	15666
True Positives	529
False Positives	42
True Negatives	15469
False Negatives	197
Precision	93.316%

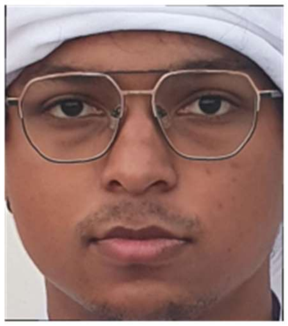
Recall	92.644%
F1 Score	72.865%
Accuracy	98.528%

Analysis

The normal challenge resulted in a very high accuracy of 98.2%. This means that overall, the predictions with regards to both positives and negatives made by the DeepFace algorithm was correct. Furthermore, the total number of false positives and false negatives (incorrect classifications) was somewhat low. This suggests that the precision and the recall should be high and thus a high F1 score, which is the case. The normal challenge presented little difficulty for both detection and recognition. The only challenging part was because the camera-drone was far away from the subject for many of the initial frames, which lead to several false negatives. Another factor leading to the false negative could be because of the orientation of the subject in the query images compared to the orientation of the subject in the video. As seen in the results section, the query image has the subject facing forward and the false negatives have the subject looking slightly downward. The reason for the false positives is due to the uniformity between the different subjects. Not only did each subject wear a ghutra as per the problem statement, but the query subject also wore glasses, and every face in the false positives also wore glasses.

Intruder Challenge:

Table 4: Peter intruder challenge results

Query Image


	Images	Cosine Value
True Positive 1		0.5778420629950678
True Positive 2		0.60521722899488
False Positive 1		0.6426925498444807
False Positive 2		0.5896623743033043
True Negative 1		0.9699522599348025
True Negative 2		0.934356739483596
False Negative 1		0.9374400587884486
False Negative 2		0.8597901615311911

False Negative 3		0.9459201787080707

Table 5: Peter intruder challenge performance measurements

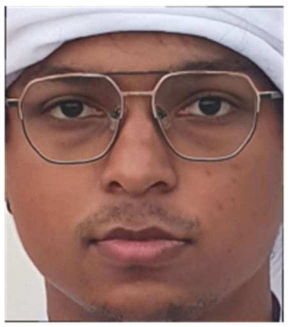
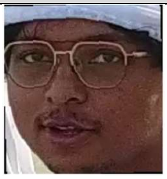
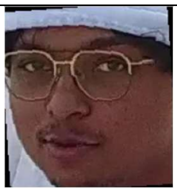
Performance Measurement	
Metric	**Value**
Total Positives	95
Total Negatives	11941
True Positives	66
False Positives	29
True Negatives	11224
False Negatives	717
Precision	69.474%
Recall	8.429%
F1 Score	15.034%
Accuracy	93.802%

Analysis

The intruder challenge presented difficulty because the subject would be infiltrating the other subject which should make detection and recognition harder. This time, the accuracy was 93.8%, which is misleading because there is a large offset between the number of positives and negatives. The precision was 69% which indicates a low number of false positives. The recall, however, was extremely low with a value of 8.43% indicating many false negatives. Furthermore, the total amount of positives is low as many of the negatives should have been classified as positives. The closeness as well as the uniformity of the subjects must have confused the recognition system leading to the low scores. The sporadic movement of the subject could also have caused poor recognition because the subject has been occluded and hidden for many of the false positive frames. The reason for the many false negatives is because the subject was significantly far away from the drone-camera, so the detected image of the subject was blurry when compared to the query image. In addition, the orientation and movement of the subject, where he partially hidden by other subjects, also lead to the misclassifications.

Random Challenge:

Table 6: Peter random challenge results

Query Image		
		
	Images	**Cosine Value**
True Positive 1		0.30278186154029274
True Positive 2		0.3532057076963553

False Positive 1		0.6493840833177065
False Positive 2		0.6310832885683069
True Negative 1		0.8490110333963538
True Negative 2		0.904345230090156
False Negative 1		0.6721265821605416
False Negative 2		0.6871992253653312

Table 7: Peter random challenge performance measurements

Performance Measurement	
Metric	**Value**
Total Positives	375

Total Negatives	9639
True Positives	359
False Positives	16
True Negatives	9261
False Negatives	378
Precision	95.733%
Recall	38.711%
F1 Score	64.568%
Accuracy	96.065%

Analysis

The random challenge had the subject as the main objective in the scenario. With the subjects all walking around in circles, there was many chances for the main subject to be occluded or hidden. The orientation of the subject plays a significant role in the results because he was seen from the side, not front as in the query image, for many of the frames. The challenge resulted a very high precision, 95%, and accuracy, 96%, indicating good overall predictions and a low number of false positives. The recall, however, was slightly below 50% meaning that a there was a good number of false negatives. The F1 score was therefore at an even 64%. The large number of false negatives is mainly because of the orientation of the subject. As seen in the results, the false negatives were all looking to the left and even away, while the query image had the subject facing forward. The false positives were affected because of the orientation as well. As the subject was facing to the side most of the time, the recognizer predicted other faces to have a better similarity instead.

Mask Challenge 1:

Table 8: Peter mask challenge 1 results

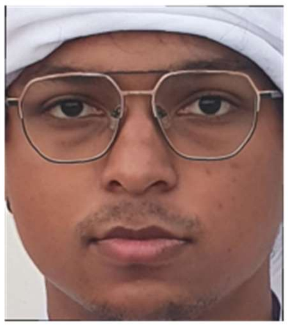
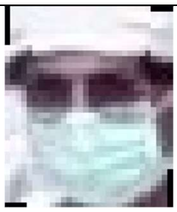
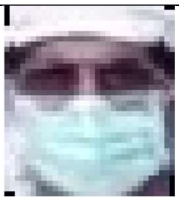
Query Image		
		
	Images	**Cosine Value**
True Positive 1	-	-
True Positive 2	-	-
False Positive 1	-	-
False Positive 2	-	-
True Negative 1		0.9573672816990993
True Negative 2		0.9635749381861359
False Negative 1		0.9627367909989106
False Negative 2		0.9577537897002338

Table 9: Peter mask challenge 1 performance measurements

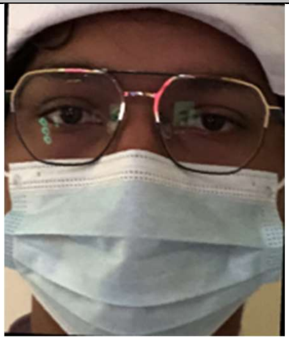
Performance Measurement	
Metric	**Value**
Total Positives	0
Total Negatives	5294
True Positives	0
False Positives	0
True Negatives	5004
False Negatives	290
Precision	-
Recall	0.0%
F1 Score	-
Accuracy	94.522%

Analysis

The mask challenge was separated into two parts. In the first, a query image with the subject not wearing a mask was tested, and the second had the subject wear a mask. The first set of results had no metrics for precision and F1. Zero positives were predicted by DeepFace, meaning that all detected faces had cosines above the threshold. This action is correct, because all the detected faces had distinguishing blue masks on, but the query image did not have this feature. As there were no positives, every case belonged to the negatives and calculating a precision was not possible. The only available metric is now the accuracy which is still high because the dataset is offset with respect to the negatives. This means that all the negatives were still classified correctly, causing the high accuracy. Having zero positives means that DeepFace could not recognize a person without a mask to the person wearing a mask.

Mask Challenge 2:

Table 10: Peter mask challenge 2 results

Query Image		
		
	Images	**Cosine Value**
True Positive 1		0.055843019319277376
True Positive 2		0.05626261197131477
False Positive 1		0.061154068331070 62
False Positive 2		0.06448961416136356

True Negative 1		0.6693731771435003
True Negative 2		0.6904036133932776
False Negative 1	-	-
False Negative 2	-	-

Table 11: Peter mask challenge 2 performance measurements

Performance Measurement	
Metric	**Value**
Total Positives	5275
Total Negatives	19
True Positives	290
False Positives	4985
True Negatives	19
False Negatives	0
Precision	5.498%
Recall	100%
F1 Score	10.422%
Accuracy	5.837%

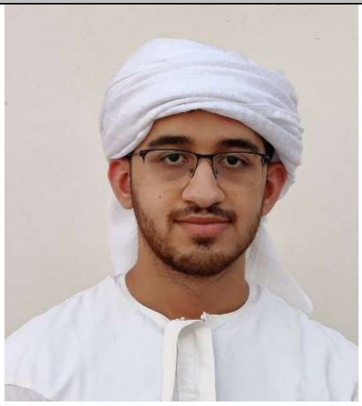
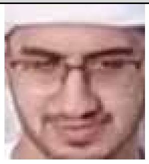
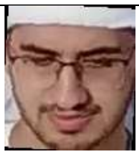
Analysis

The second test had the query subject wear a mask and seemed more promising. This test resulted in very low scores in every category except recall. The recall was 100% because there were zero false negatives, but this only happened because most of the detected faces were classified as positives. The uniformity between all the faces caused all of them to have cosines below the threshold. With all the faces having masks on, Deepface could not find enough distinguishing features to differentiate the faces. As almost everyone was classified as a positive, the precision score took a hit. In addition, only 19 total negatives were reported, and each face had a large portion not full hidden by the mask and was very clear.

Ahmed

Normal Challenge:

Table 12: Ahmed normal challenge results

Query Image		
		
	Images	**Cosine Value**
True Positive 1		0.5316036296050217
True Positive 2		0.5343108707083903

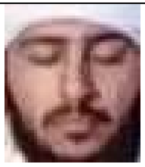
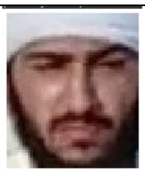
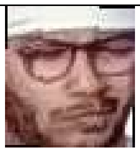
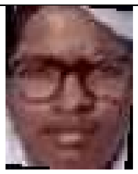
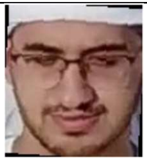
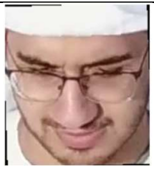
False Positive 1		0.6466227457701834
False Positive 2		0.6455344055824611
True Negative 1		0.8545062023643755
True Negative 2		0.9551626240433582
False Negative 1		0.6605004130111163
False Negative 2		0.6541573312729722

Table 13: Ahmed normal challenge performance measurements

Performance Measurement	
Metric	**Value**
Total Positives	763
Total Negatives	15467
True Positives	712
False Positives	51
True Negatives	15453

False Negatives	14
Precision	93.316%
Recall	98.072%
F1 Score	95.635%
Accuracy	99.600%

Analysis

In the normal challenge, we can see that our system did not face difficulties with detecting and recognizing people. The 99.6% accuracy shows us that the system has classified the positive and negative cases almost perfectly. We can see that the system has high precision and recall scores, resulting in a high 95.635% F1 score. The F1 score shows us that there is only a small number of misidentifications in the normal challenge, where we have only 51 false positives, and 14 false negatives. The false positives were a result of someone having a similar face structure and features as can be seen in the false positive case 1 and 2 shown in the table, most of the false positive cases where on the edge of the threshold having a cosine distance of 0.64 or more. The false negatives were all very close to the threshold having a cosine of 0.7 or lower. This misclassification was mostly due to the lower quality of the face images or the face orientation being down or sideways, which provides a false negative when compared to our query image when the face is looking forward.

Intruder Challenge:

Table 14: Ahmed intruder challenge results

Query Image		
	Images	**Cosine Value**
True Positive 1		0.4326557509088137
True Positive 2		0.6057107203363272
False Positive 1		0.6347904833663711
False Positive 2		0.6221328160309522
True Negative 1		1.0030884903901813
True Negative 2		0.9908037033226844

False Negative 1		0.9686740742804856
False Negative 2		0.9758550682658034

Table 15: Ahmed intruder challenge performance measurements

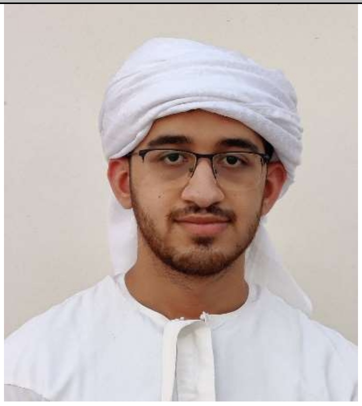
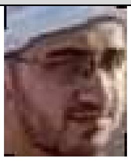
Performance Measurement	
Metric	**Value**
Total Positives	271
Total Negatives	15434
True Positives	251
False Positives	20
True Negatives	14800
False Negatives	634
Precision	92.620%
Recall	28.362%
F1 Score	43.425%
Accuracy	95.836%

Analysis

In the intruder challenge, where the subject infiltrated the crowd. We can see that our system faced some difficulties in classifying people. The 95.836% accuracy shows that most people were classified to the correct categories, negative or positive. But the amount of misclassification was high. Having 20 false positives and 634 false negatives. Which resulted in 92.62% precision and a low 28.362% recall giving us F1 score of 43.425%. The high precision score shows us that we correctly identified most of our positive cases, while the low recall score shows us that we have a high number of false negatives. The false positives were usually from cases where people had similar facial features and wore glasses similar to the query image. Additionally, the high number of false negatives is mostly due to face orientation being very different from the query image, a person obstructing the view of the subject when he was infiltrating into the crowd, or from the blurry images from the initial 300 frames of the video when the subject was far away from the camera. From the table, we can see in the first false negative that the image is very blurry, making it very difficult to be recognized. In the second false negative case, we can see a person obstructing view of the subject making it difficult to recognize the subject.

Random Challenge:

Table 16: Ahmed random challenge results

Query Image		
		
	Images	**Cosine Value**
True Positive 1		0.6212889770074187

True Positive 2		0.5906604688401929
False Positive 1		0.617877574400598
False Positive 2		0.6447438557404195
True Negative 1		0.8456625767512245
True Negative 2		0.7078252702061407
False Negative 1		0.9740360778891115
False Negative 2		0.98634489101639

Table 17: Ahmed random challenge performance measurements

Performance Measurement

Metric	Value
Total Positives	441
Total Negatives	11233
True Positives	376
False Positives	65
True Negatives	10950
False Negatives	283
Precision	85.261%
Recall	57.056%
F1 Score	68.363%
Accuracy	97.019%

Analysis

In the random challenge, where the crowd was moving in a random and unpredictable manner in a small area, we can see that the face detection and recognition preformed decently. The accuracy had a score of 97.019% indicating that in general, most people were classified correctly as either negative or positive. The precision score was 85.261% showing that most of the positive cases were true positives. The 57.056% recall score shows that we have high false negatives. The resulting F1 score from the recall and precision was 68.363%. Most of the 65 false positive cases had cosine distances on the edge of the threshold, these cases mainly occurred as some participants had similar facial features and due to the lower quality of some of the images making it harder for the recognition to judge and classify correctly. Additionally, there was 283 false negatives. From the table, we can see that we got false negatives because of the orientation of the face, because the subject in this video moved randomly, facing left and right, whereas the query image is facing forward, resulting in false negative cases. Another factor was that the subject was not the focus of the video, therefore most of the images where slightly lower quality and blurry reducing the ability to recognize the subject correctly.

Mask Challenge 1:

Table 18: Ahmed mask challenge 1 results

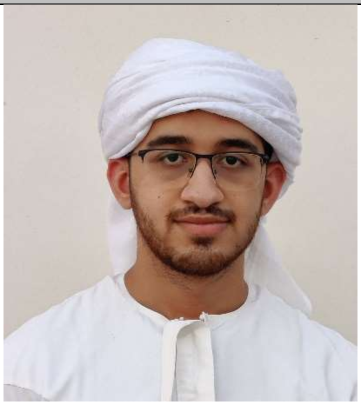
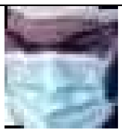
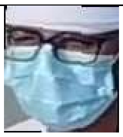
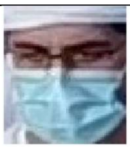
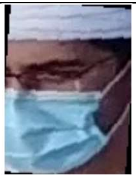
Query Image		
		
	Images	**Cosine Value**
True Positive 1	-	-
True Positive 2	-	-
False Positive 1	-	-
False Positive 2	-	-
True Negative 1		1.0048774595593646
True Negative 2		1.0160089282913516
False Negative 1		0.9990818679590294
False Negative 2		0.9951024336441249

Table 19: Ahmed mask challenge 1 performance measurements

Performance Measurement	
Metric	**Value**
Total Positives	0
Total Negatives	5294
True Positives	0
False Positives	0
True Negatives	4897
False Negatives	397
Precision	-
Recall	0%
F1 Score	-
Accuracy	92.501%

Analysis

In the mask challenge, where everyone in the crowd was wearing a mask, the face recognition could not detect any positive cases when using a query image without a mask. We can see that we have 0 true positives, 0 false positives, and in total, 0 total positives. The recall is 0% as we have 0 true positives, the precision cannot be calculated as we are dividing by 0. Which means that the F1 score also cannot be calculated for this case. From the results, we can see that all the faces were recognized as negative, giving us 397 false negatives, 4897 true negatives, and a total of 5294 negatives. This is because in the query image, the subject is not wearing a mask, while in the video, everyone is wearing a mask. Resulting in a very high cosine distance for every face, making everyone a negative.

Mask Challenge 2:

Table 20: Ahmed mask challenge 2 results

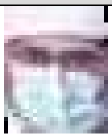
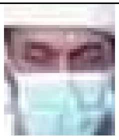
Query Image		
	Images	**Cosine Value**
True Positive 1		0.3479608528131265
True Positive 2		0.28743870897249235
False Positive 1		0.334895992449175
False Positive 2		0.319749514277187
True Negative 1		0.6783345350611745
True Negative 2		0.6642838884357654
False Negative 1	-	-
False Negative 2	-	-

Table 21: Ahmed mask challenge 2 performance measurements

Performance Measurement	
Metric	**Value**

Total Positives	5257
Total Negatives	37
True Positives	397
False Positives	4860
True Negatives	37
False Negatives	0
Precision	7.551%
Recall	100%
F1 Score	14.043%
Accuracy	8.197%

Analysis

In the mask challenge, when the subject is wearing a mask in the query image, we can see that the challenge is still very difficult for the face detection and recognition. We have an accuracy of 8.197%, which means that most people were identified incorrectly. The precision score is 7.551%, which shows that most of the positives are false positives. The recall score is 100%, which tells us that we have no false negatives in this case. From the precision and recall, we get F1 score of 14.043% which is very low. Most people were identified as positives. From the table, we can see that we have 397 true positives, 4860 false positives, giving us 5257 total positives. Additionally, we have only 37 total negatives where all of them are true negatives. The negative cases appeared when people that are not subject had a large portion of their face not covered by the mask giving the face recognition the ability to identify them correctly. Moreover, most of the faces were recognized as positives due to the mask hiding most of their faces similar to the query image where the mask is covering most of the face, resulting in incorrect identification.

Yousef

Normal Challenge:

Table 22: Yousef normal challenge results

Query Image		
	Images	**Cosine Value**
True Positive 1		0.7001535993664798
True Positive 2		0.7194191371697057
False Positive 1		0.6872084935017573
False Positive 2		0.6769661853614386
True Negative 1		0.9916354295343303

True Negative 2		1.0398228247959025
False Negative 1		1.0374760655395243
False Negative 2		1.0311086997285503

Table 23: Yousef normal challenge performance measurements

Performance Measurement	
Metric	**Value**
Total Positives	434
Total Negatives	7908
True Positives	289
False Positives	145
True Negatives	7647
False Negatives	261
Precision	66.589%
Recall	52.545%

F1 Score	58.739%
Accuracy	95.133%

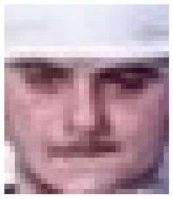
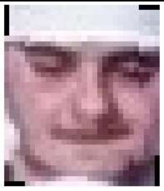
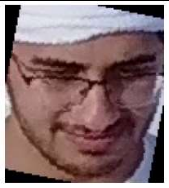
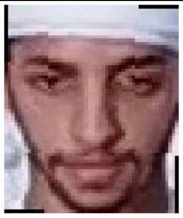
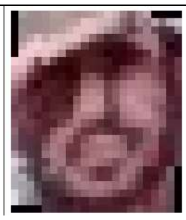
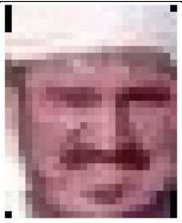
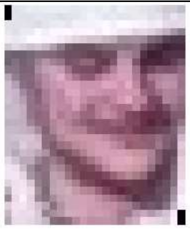
Analysis

In the normal challenge, even though the accuracy is good the F1 score is much worse than expected because of the low precision and recall values. This is because of the high number of false positives and false negatives cases done by the algorithm. As can be seen in the false positive and false negatives examples in the table above, many of the faces misidentified appear pixelated and unclear which could be due to the camera being unfocused or the video being low quality. Because of this, the threshold had to be finely tuned to a cosine value of 0.8. such a high threshold could be the reason why the algorithm failed on identifying them as real negatives or real positives. The algorithm has a great accuracy of 95.133% which means it can detect most of the negative and positive cases correctly.

Compact Challenge:

Table 24: Yousef compact challenge results

Query Image		
	Images	**Cosine Value**

True Positive 1		0.6463975538145637
True Positive 2		0.5493443038357853
False Positive 1		0.6291114758194724
False Positive 2		0.6310867134705227
True Negative 1		1.0566103876496826
True Negative 2		1.0445191650302819
False Negative 1		1.0370154648526249

False Negative 2		1.031040919675371

Table 25: Yousef compact challenge performance measurements

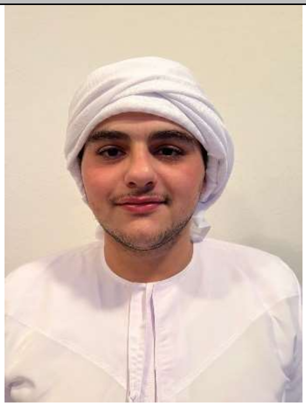
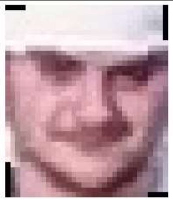
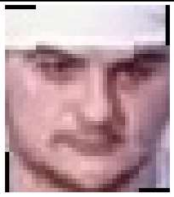
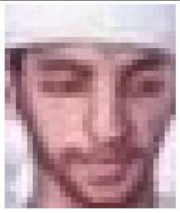
Performance Measurement	
Metric	**Value**
Total Positives	564
Total Negatives	18345
True Positives	553
False Positives	11
True Negatives	18064
False Negatives	281
Precision	98.049%
Recall	66.306%
F1 Score	79.113%
Accuracy	98.455%

Analysis

In the compact challenge, all the people contributing to the video were extremely close to each other which makes increases the probability that faces become hidden or overlap with others in the video. Looking at the table above, the accuracy is above 98% which means that the algorithm was able to correctly identify most of the cases correctly. The precision is also very high because of the very low number of false positives while the recall is at 66% because there are 281 cases of false negatives. Looking at the false negative pictures above, they are unclear and lack details which could be why the algorithm has misidentified them. The F1 score which takes into consideration the precision and recall is at a decent 79% which means the algorithm occasionally misidentifies false positives and false negatives.

Intruder Challenge:

Table 26: Yousef intruder challenge results

Query Image		
		
	Images	**Cosine Value**
True Positive 1		0.7614606229899912
True Positive 2		0.7039396025427851
False Positive 1		0.697785606439542

False Positive 2		0.7444152183932833
True Negative 1		1.033952763187515
True Negative 2		1.0372391350747625
False Negative 1		1.0485350940797917
False Negative 2		1.0302827865141069

Table 27: Yousef intruder challenge performance measurements

Performance Measurement	
Metric	**Value**
Total Positives	333
Total Negatives	15372
True Positives	258

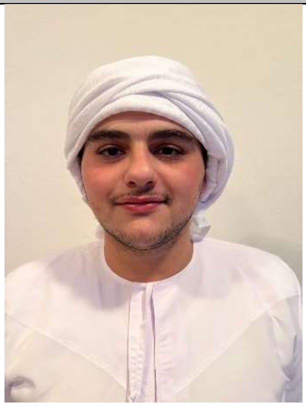
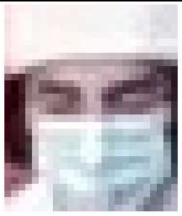
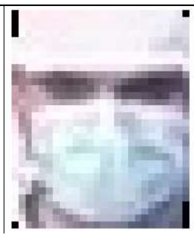
False Positives	75
True Negatives	14850
False Negatives	522
Precision	77.477%
Recall	33.076%
F1 Score	46.361%
Accuracy	96.198%

Analysis

In the intruder challenge, the video used was for a crowd in a compact formation while several intruders were passing through them. The target face was not part of the intruders but was part of the crowd. As can be seen in the table above, the accuracy is 96.198% which is great since it means that for most of the cases the algorithm identifies the person correctly. However, the F1 score is bad since it is below 50%. This is mainly because there are 522 false negatives which is a very high number. The false positives are also at 75 which is a lot when compared to only 258 true positives. The reason for the high number of false negatives and positives could be because of the low quality of the faces extracted and the overlapping of faces due to the compact formation as shown in the two false negative pictures.

Mask Challenge 1:

Table 28: Yousef mask challenge 1 results

Query Image		
		
	Images	**Cosine Value**
True Positive 1	-	-
True Positive 2	-	-
False Positive 1	-	-
False Positive 2	-	-
True Negative 1		1.0498954691967537
True Negative 2		1.0468249440386155
False Negative 1		1.0511389741895671

False Negative 2		1.039684677797931

Table 29: Yousef mask challenge 1 performance measurements

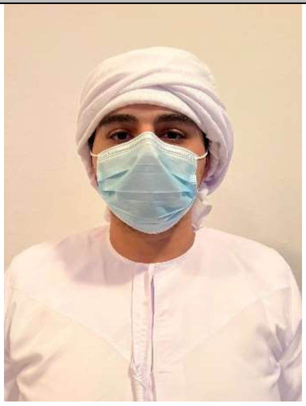
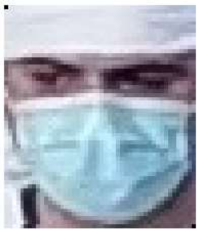
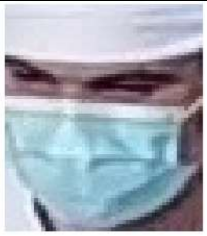
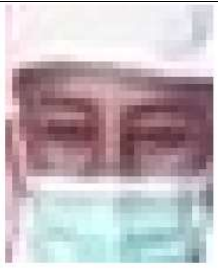
Performance Measurement	
Metric	**Value**
Total Positives	0
Total Negatives	5294
True Positives	0
False Positives	0
True Negatives	4938
False Negatives	356
Precision	-
Recall	0.0%
F1 Score	-
Accuracy	93.275%

Analysis

In mask challenge 1, the query image used was one with no mask to see how the algorithm handles the difference in input. as can be seen in the table, the identification program was not able to identify a single positive and this is because all the faces are wearing masks while the query image is without. Since there were 0 positives, all the faces in the frames were considered negatives since none of them passed the threshold. The accuracy is high but looking at it alone is misleading and the F1, recall, and precision scores have to also be taken into consideration.

Mask Challenge 2:

Table 30: Yousef mask challenge 2 results

Query Image		
		
	Images	**Cosine Value**
True Positive 1		0.6498369614879915
True Positive 2		0.6493865485802156
False Positive 1		0.647338117647146

False Positive 2		0.6485621616654769
True Negative 1		0.717516535619459
True Negative 2		0.7030505006070278
False Negative 1		0.7147985759963305
False Negative 2		0.7022768283302954

Table 31: Yousef mask challenge 2 performance measurements

Performance Measurement	
Metric	**Value**
Total Positives	1103
Total Negatives	4191
True Positives	103
False Positives	1000
True Negatives	3938
False Negatives	253
Precision	77.477%
Recall	28.932%
F1 Score	42.131%
Accuracy	76.331%

Analysis

In the second mask challenge, the query image was changed to one with a mask to see if there is an improvement to the first mask challenge. By looking at the F1 score there is a slight improvement since the algorithm was able to detect 103 true positives however the number of false positives also increased greatly which led to a huge drop in the accuracy. Looking at the false negatives there are still 253 misidentified faces out of 356 which means that the algorithm does not do well with masks regardless of if the query image is with or without masks.

Nawfal

Normal Challenge:

Table 32: Nawfal normal challenge results

Query Image		
	Images	**Cosine Value**
True Positive 1		0.6118123580116538
True Positive 2		0.6357062355196135
False Positive 1		0.6205372071282105
False Positive 2		0.6321773894642533
True Negative 1		0.7369694415688886
True Negative 2		0.7327045153353742

False Negative 1		0.7092038878219358
False Negative 2		0.7312363322724615

Table 33: Nawfal normal challenge performance measurements

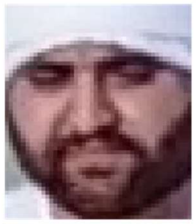
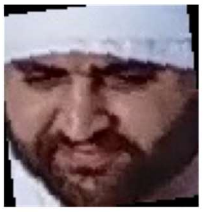
Performance Measurement	
Metric	**Value**
Total Positives	147
Total Negatives	12541
True Positives	120
False Positives	27
True Negatives	12018
False Negatives	523
Precision	81.63%
Recall	18.66%
F1 Score	30.38%
Accuracy	95.67%

Analysis

For the normal challenge, the accuracy was 95.67%, which means that the predictions or classifications of positive and negative cases made by the DeepFace algorithm were correct. We can see that when comparing the true positives (120) and true negatives (12,018) to the total positives (147) and total negatives (12,541) respectively, the number of false positives and false negatives are to some extent low, which resulted in such a high accuracy. From the table above, we can see that the number of false positives (27) compared to the total positives (120) is low, which resulted in a good precision of 81.63%. However, due to the greater number of false negatives (523) compared to the true positives (120), it resulted in a poor recall percentage 18.66%. Because of the bad recall percentage, it also led to a low F1 score of 30.38%. The only difficulties that the normal challenge could have presented were firstly how the face pictures as seen from the results tables above were very pixelated at the beginning frames of the video due to the camera being far away, and secondly how in the video the subject's face can be slightly sideways or downwards as seen in the results table above, whereas in the query image the face is facing forwards. Because of these difficulties, it could have been hard for the recognition algorithm to have picked up on the face features and structure of the subject.

Compact Challenge:

Table 34: Nawfal compact challenge results

Query Image		
	Images	**Cosine Value**
True Positive 1		0.6199404436077335
True Positive 2		0.6406959469369463
False Positive 1	-	-

False Positive 2	-	-
True Negative 1		0.9638622173449238
True Negative 2		0.9641615114278442
False Negative 1		0.9752948440858732
False Negative 2		0.981953783938708

Table 35: Nawfal compact challenge performance measurements

Performance Measurement	
Metric	**Value**
Total Positives	51
Total Negatives	11802
True Positives	51
False Positives	0
True Negatives	11301
False Negatives	501

Precision	100%
Recall	9.24%
F1 Score	16.92%
Accuracy	95.77%

Analysis

As for the compact challenge, we can see similar results with the normal challenge. For the accuracy it was 95.77%, which is very high. In this case however, we can see that there were zero false positives which leads to 100% in precision. For recall however, it got a very low percentage of 9.24% due to how large the amount of false negatives (501) is compared to the true positives (51). This also leads to the F1 score being very low at 16.92%. The normal and compact challenges share almost the same difficulties, however, for the compact challenge there is an additional one. Since in the compact challenge the members of the crowd were close to each other whilst walking, in some frames of the frames my face was not completely visible due to being blocked by other crowd members, as seen in one of the false positive pictures above, which makes it harder for the face recognition to distinguish the face features of subject's face.

Random Challenge:

Table 36: Nawfal random challenge results

Query Image		
	Images	**Cosine Value**
True Positive 1		0.6312432107113216

True Positive 2		0.631681702450941
False Positive 1	-	-
False Positive 2	-	-
True Negative 1		0.9833698351358267
True Negative 2		0.9186856595961499
False Negative 1		0.9974447295748484
False Negative 2		0.9928343194935892

Table 37: Nawfal random challenge performance measurement

Performance Measurement	
Metric	**Value**
Total Positives	236
Total Negatives	9148

True Positives	236
False Positives	0
True Negatives	8698
False Negatives	450
Precision	100%
Recall	34.40%
F1 Score	51.19%
Accuracy	95.20%

Analysis

For the random challenge, we can see that the face recognition performed good. Since the accuracy is 95.20%, we can say that the face recognition algorithm has classified all cases almost correctly to positive and negative cases. In this case, we can also see that the precision is 100%, which means that there are zero false positives, and all the faces that were below the threshold were actually the subject's face. Since there were 450 false negatives compared to the 236 true positives, it led to a bad recall percentage of 34.40%. As for the F1 score, it led to a 51.19% coming from the recall and precision percentages. The large number in false negative for this case can be explained by how in the random challenge, the camera was following on a crowd member who was different from the subject and was moving in a random manner among the crowd. This did not only lead to the subject's face being blocked by other crowd members, but also there were a lot of frames where the subject's face is completely sideways as seen in the results table above.

Mask Challenge 1:

Table 38: Nawfal mask 1 challenge results

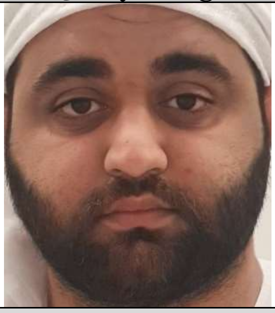
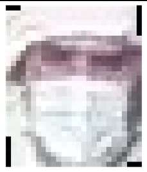
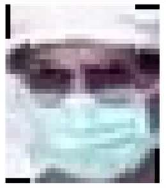
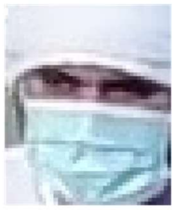
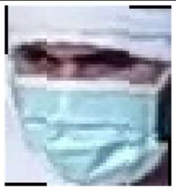
Query Image		
		
	Images	**Cosine Value**
True Positive 1	-	-
True Positive 2	-	-
False Positive 1	-	-
False Positive 2	-	-
True Negative 1		0.9922688087335493
True Negative 2		0.9943119417950202
False Negative 1		0.9929251256052519
False Negative 2		0.9910090723097321

Table 39: Nawfal mask challenge 1 performance measurements

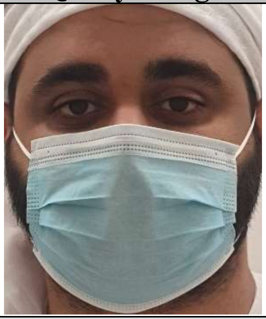
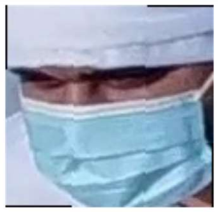
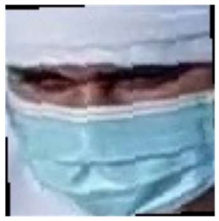
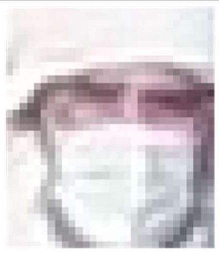
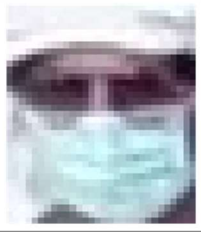
Performance Measurement	
Metric	**Value**
Total Positives	0
Total Negatives	4523
True Positives	0
False Positives	0
True Negatives	4228
False Negatives	295
Precision	-
Recall	0%
F1 Score	0.0%
Accuracy	93.48%

Analysis

For the first mask challenge where the subject was not wearing a mask, even though the accuracy is 93.48%, the face recognition algorithm performed poorly. We can see that the algorithm did not pick on a single positive case, which means the true positives and negatives are both zero. Since the true positives are zero, it leads to the precision, recall, and F1 score all having a percentage of 0% . The reason for the high accuracy percentage, is that the number of actual negative cases are greatly larger than the actual positive ones. And Since the algorithm classified all cases to negative, the accuracy was still high. The high number of false negatives, which is 295, is due to the subject having no mask in the query, whereas in the video all the crowd members were wearing masks. This also leads to everyone having a cosine value greater than the threshold, which also results into zero positive cases.

Mask Challenge 2:

Table 40: Nawfal mask challenge 2 results

Query Image		
		
	Images	**Cosine Value**
True Positive 1		0.4636532677703622
True Positive 2		0.4658894688047247
False Positive 1		0.49963333090378237
False Positive 2		0.4889930320549576

True Negative 1		0.701112670237068
True Negative 2		0.7322649290412339
False Negative 1	-	-
False Negative 2	-	-

Table 41: Nawfal mask challenge 2 performance measurements

Performance Measurement	
Metric	**Value**
Total Positives	4437
Total Negatives	86
True Positives	295
False Positives	4142
True Negatives	86
False Negatives	0
Precision	6.65%
Recall	100%
F1 Score	12.47%
Accuracy	8.42%

Analysis

For the second mask challenge, where the subject was wearing a mask in the query image, the face recognition algorithm performed better than the first mask challenge, but it still faced difficulty. We can see the accuracy in this case is for the first time across all challenges, very low at only 8.42%. The reason for this, is the in this case, there are 4437 total positives, whereas there are only 86 total negatives. Which means that almost everybody had a cosine value lower than the threshold. Due to the large number of false positive at 4142, the precision was very low at 6.65%. As for the recall, it was at 100%, due to there being zero false negatives. For the F1 score however, it was only 12.47% due to the low precision. The reason for the high number of positive cases, is because due to the mask covering most of the faces for the subject in the query image and the crowd members in the video, the face recognition algorithm picked up on the same face features for everybody leading to many misclassifications.

Discussion & Conclusion

The team applied and integrated face detection and recognition to identify certain people from a crowd wearing uniform clothes. The team tested the face identification algorithm on various challenges, such as the crowd moving randomly, the crowd wearing masks, the crowd being compact, an intruder infiltrating the crowd, and a normal case where the crowd moved uniformly towards the camera. From the results discussed above, we can see that the identification algorithm performed mostly well on the normal, compact, and random challenges. On the other hand, the identification algorithm did not perform well in the intruder and mask challenges. For the false negative and false positive cases, the identification algorithm mostly misidentified the subjects due to three main causes. Firstly, the images being blurry and pixelated due to the camera being far away. Secondly, the orientation of the faces being very different from the query image. Finally, the faces being hidden by other crowd members or the masks. The facial identification algorithm can work well in general, but it needs to be improved on the cases where the face orientation is different from the query image and when the image is blurry and pixelated.

Appendix

```
# -*- coding: utf-8 -*-
"""FinalDeepFace.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1DBsrmkKq_dQ5Bg20TbEq18GX
KahUjC29
"""

from google.colab import drive
drive.mount('/content/drive')

project_dir = '/content/drive/MyDrive/AI/deepface' #change this as
needed

"""Convert Video to Frames"""

import cv2
challenge_type = "mask"

vidcap = cv2.VideoCapture(project_dir + '/videos/mask.mp4')
success,image = vidcap.read()
count = 0

while success:
  cv2.imwrite(project_dir + '/frames/' + challenge_type +
'/frame%d.jpg' % count, image)     # save frame as JPG file
  success,image = vidcap.read()
  count += 1

"""# Extract faces from the frames using Face Detector"""

!pip install retina-face

#FACE DETECTION
from retinaface import RetinaFace
import os
import matplotlib.pyplot as plt
frame_dir = project_dir + '/frames/' + challenge_type
frames = os.listdir(frame_dir)
detectedFaces = {}

for frame in frames:

  img_path = frame_dir + "/" + frame
  resp = RetinaFace.detect_faces(img_path)
  if type(resp) == dict:
    faces = {}
    for k in resp:
      faces[frame.split('.')[0] + "_" + k] = [int(i) for i in 
resp[k]["facial_area"]]
    detectedFaces[ frame.split('.')[0] ] = faces

    faces_extract = RetinaFace.extract_faces(img_path = img_path, 
align = True)
    count = 1
    for face in faces_extract:
        save_path =  project_dir + '/extracted_faces/'
+challenge_type + "/" + frame.split('.')[0] + '_face_' + str(count) +
'.jpg'
        cv2.imwrite(save_path, cv2.cvtColor(face, cv2.COLOR_RGB2BGR))
        count +=1

import json

json = json.dumps(detectedFaces)

with open(project_dir + "/" + challenge_type +  "_faces.json", "w")
as f:

  f.write(json)

"""# Verify your results"""

!pip install deepface

#Recognition
from deepface import DeepFace

# cosine, euclidean, euclidean_l2

#Query images
# img_path =
'/content/drive/MyDrive/AI/deepface/queries/100053648_Mask_Glasses_Gh
utra_WhiteBackground_front.jpg'
```

```
# img_path =
'/content/drive/MyDrive/AI/deepface/queries/100053648_Glasses_Ghutra_
Morning_front.jpg'
img_path =
"/content/drive/MyDrive/AI/deepface/queries/100053648_Glasses_Ghutra_
Morning_left.jpg"

#Faces Directory
db_path = '/content/drive/MyDrive/AI/deepface/extracted_faces/random'

#Recognition happens here
y = DeepFace.find(img_path, db_path, model_name ="ArcFace",
distance_metric = 'cosine', model = None, enforce_detection = False,
detector_backend = 'retinaface', align = True, prog_bar = True,
normalization = 'base', silent=False)
y

import os
len(os.listdir('/content/drive/MyDrive/AI/deepface/extracted_faces/in
truder' ))

paths = y["identity"]
values = y["ArcFace_cosine"]
positives = []
negatives = []
threshold = 0.65

#DF parser
for i in range(len(paths)):
  name = paths[i].split("/")[-1].split(".")[0]

  if values[i] <= threshold:
    positives.append((name,values[i]))

  else:
    negatives.append((name,values[i]))

"""# Load in old data

"""

#load detected faces
import json
challenge_type = "mask"
result_dir = project_dir + "/" + challenge_type + "_results/" +
challenge_type

with open(result_dir + "_faces.json") as f:
  detectedFaces = json.load(f)

#load positives and negatives
positives = []
negatives = []
import json
result_dir = project_dir + "/" + challenge_type + "_results/" +
challenge_type

with open(result_dir + "_positives.json") as f:
  positives_dict = json.load(f)
  for key, value in positives_dict.items():
    positives.append((key,value))

with open(result_dir + "_negatives.json") as f:
  negatives_dict = json.load(f)
  for key, value in negatives_dict.items():
    negatives.append((key,value))

"""## Used for adjusting threshold"""

#adjusting threshold

#new split
identities = [i for i, k in positives] + [i for i, k in negatives]
values = [k for i, k in positives] + [k for i, k in negatives]
y = pd.DataFrame(
    {'identity': identities,
     'ArcFace_cosine': values
    })

positives = []
negatives = []
threshold = 0.65

#DF parser
for i in range(len(identities)):

  if values[i] <= threshold:
    positives.append((identities[i],values[i]))
  else:
    negatives.append((identities[i],values[i]))

"""# Validation"""

!pip install xmltodict

import xmltodict
import matplotlib.pyplot as plt

with open(project_dir + "/xml_files/" + challenge_type
+"_annotations.xml") as file:
    file_data = file.read() # read file contents

    # parse data using package
    dict_data = xmltodict.parse(file_data)

image_list  =[]
items = dict_data["annotations"]["track"]["box"]

#check false and true positives
true_positives = []
false_positives = []
positive_holder = []

for i in items:
  framenum = "frame" + i["@frame"]
  try:
    detectedFaces[framenum]
  except:
        continue
  else: #match found
    for k, j in positives:
      try:
        detectedFaces[framenum][k]
      except:
        # count+=1
        continue
      else:
        value = detectedFaces[framenum][k]
        detected_x = float(i["@xtl"])
        detected_y = float(i["@ybr"])
        annotated_x = value[0]
        annotated_y = value[3]
        x_error = abs(annotated_x - detected_x)
        y_error = abs(annotated_y - detected_y)
        if x_error < 20 and y_error < 20 :
          true_positives.append([k, (x_error, y_error), j])
          positive_holder.append([k, (x_error, y_error), j])
        else:
          false_positives.append([k, (x_error, y_error), j])
          positive_holder.append([k, (x_error, y_error), j])

found = False

for i,j in positives:
  found = False
  for k,q,l in positive_holder:
    if i == k:
      found = True
  if not found:
      false_positives.append([i, (1000, 1000), j])

#check false and true negatives
true_negatives = []
false_negatives = []
negative_holder = []
for i in items:
  framenum = "frame" + i["@frame"]
  try:
    detectedFaces[framenum]
  except:
        continue
  else: #match found
    for k, j in negatives:
      try:
        detectedFaces[framenum][k]
      except:
        continue
      else:
        value = detectedFaces[framenum][k]
        detected_x = float(i["@xtl"])
        detected_y = float(i["@ybr"])
        annotated_x = value[0]
        annotated_y = value[3]
        x_error = abs(annotated_x - detected_x)
        y_error = abs(annotated_y - detected_y)
        if x_error < 20 and y_error < 20 :
          false_negatives.append([k, (x_error, y_error), j])
          negative_holder.append([k, (x_error, y_error), j])
        else:
          true_negatives.append([k, (x_error, y_error), j])
          negative_holder.append([k, (x_error, y_error), j])



found = False

for i,j in negatives:
  found = False
  for k,q,l in negative_holder:
    if i == k:
      found = True

  if not found:
      true_negatives.append([i, (1000, 1000), j])

"""Alansari Results for Testing"""

total_positives = len(positives)
total_negatives = len(negatives)
total_true_positives = len(true_positives)
total_false_positives = len(false_positives)
total_true_negatives = len(true_negatives)
total_false_negatives = len(false_negatives)

print("total_positives: ", total_positives)
print("total_negatives: ", total_negatives)
print("total_true_positives: ", total_true_positives)
print("total_false_positives: ", total_false_positives)
print("total_true_negatives: ", total_true_negatives)
print("total_false_negatives: ", total_false_negatives)

try:
  recall = (total_true_positives/ (total_true_positives +
total_false_negatives)) * 100
except:
  recall = "-"

try:
  precision = (total_true_positives / (total_true_positives +
total_false_positives)) * 100
except:
  precision = "-"

try:
  accuracy = ((total_true_positives +
total_true_negatives)/(total_true_negatives + total_false_positives
+  total_false_negatives + total_true_positives)) * 100
except:
  accuracy = "-"
try:
  f1 = (2 * (precision * recall) / (precision + recall))
except:
  f1 = "-"
print("Precision: ", precision," %")
print("Recall: ", recall, " %")
print("F1 score: ", f1, " %")
print("Accuracy: ", accuracy, " %")
```

```
"""# Saving the Data"""

with open(project_dir + "/" + challenge_type + "_results/"
+  challenge_type +  "_results.txt", "w") as f:
  f.write("Precision: " + str(precision) +"%\n")
  f.write("Recall: " + str(recall) + "%\n")
  f.write("F1 score " + str(f1) + "%\n")
  f.write("Accuracy: " + str(accuracy) + "%\n\n")

  f.write("total_positives: " + str(total_positives) + "\n" )
  f.write("total_negatives: " + str(total_negatives) + "\n" )
  f.write("total_true_positives: " + str(total_true_positives) + "\n"
)
  f.write("total_false_positives: " + str(total_false_positives) +
"\n" )
  f.write("total_true_negatives: " + str(total_true_negatives) + "\n"
)
  f.write("total_false_negatives: " + str(total_false_negatives) +
"\n" )

obj = {}

for i,j,k in false_positives:
  obj[i] = (k,j)
import json
# create json object from dictionary
json = json.dumps(obj)

# open file for writing, "w"
```

```
f = open(project_dir + "/" + challenge_type + "_results/"
+  challenge_type + "_false_positives.json", "w")

# write json object to file
f.write(json)

# close file
f.close()
######################################################################
############
obj = {}

for i,j,k in true_positives:
  obj[i] = (k,j)
import json
# create json object from dictionary
json = json.dumps(obj)

# open file for writing, "w"
f = open(project_dir + "/" + challenge_type + "_results/"
+  challenge_type+  "_true_positives.json", "w")

# write json object to file
f.write(json)

# close file
f.close()
######################################################################
############
```

```
obj = {}
for i,j,k in true_negatives:
  obj[i] = (k,j)

import json
# create json object from dictionary
json = json.dumps(obj)

# open file for writing, "w"
f = open(project_dir + "/" + challenge_type + "_results/"
+  challenge_type +  "_true_negatives.json", "w")

# write json object to file
f.write(json)

# close file
f.close()
######################################################################
############

obj = {}
for i,j,k in false_negatives:
  obj[i] = (k,j)

import json
# create json object from dictionary
json = json.dumps(obj)

# open file for writing, "w"
```

```
f = open(project_dir + "/" + challenge_type + "_results/"
+  challenge_type +  "_false_negatives.json", "w")

# write json object to file
f.write(json)

# close file
f.close()
######################################################################
############

obj = {}
for i,j in positives:
  obj[i] = j

import json
# create json object from dictionary
json = json.dumps(obj)

# open file for writing, "w"
f = open(project_dir + "/" + challenge_type + "_results/"
+  challenge_type +  "_positives.json", "w")

# write json object to file
f.write(json)

# close file
f.close()
######################################################################
############
```

```
obj = {}
for i,j in negatives:
  obj[i] = j

import json
# create json object from dictionary
json = json.dumps(obj)

# open file for writing, "w"
f = open(project_dir + "/" + challenge_type + "_results/"
+  challenge_type +  "_negatives.json", "w")

# write json object to file
f.write(json)

# close file
f.close()

"""# **Plotting**"""

!pip install retina-face

import matplotlib.image as mpimg
from retinaface import RetinaFace
# img_path =
'/content/drive/MyDrive/AI/deepface/queries/100053648_Glasses_Ghutra_
Morning_front.jpg'
```

```
img_path =
'/content/drive/MyDrive/AI/deepface/queries/100053648_Mask_Glasses_Gh
utra_WhiteBackground_front.jpg'
faces_extract = RetinaFace.extract_faces(img_path = img_path, align =
True)
for face in faces_extract:
    save_path =  project_dir + "/mask_query" +  '_face.jpg'
    cv2.imwrite(save_path, cv2.cvtColor(face, cv2.COLOR_RGB2BGR))
img = mpimg.imread(save_path)
plt.axis('Off')
plt.imshow(img)
plt.show()

import matplotlib.image as mpimg
for i,j,k in true_positives:
  img = mpimg.imread(project_dir +"/extracted_faces/" +
challenge_type + "/"+ i + ".jpg")
  plt.axis('Off')
  plt.imshow(img)
  plt.show()
  print(i)
  print(k)
  print(j)

import matplotlib.image as mpimg
for i,j,k in false_positives:
  try:
    img = mpimg.imread(project_dir +"/extracted_faces/" +
challenge_type+ "/"+ i+ ".jpg")
    plt.axis('Off')
    plt.imshow(img)
    plt.show()
    print(i)
    print(k)
    print(j)
  except:
    print()

import matplotlib.image as mpimg
for i,j,k in true_negatives:
  img = mpimg.imread(project_dir + "/extracted_faces/" +
challenge_type+ "/"+ i+ ".jpg")
  plt.axis('Off')
  plt.imshow(img)
  plt.show()
  print(i)
  print(k)
  print(j)

for i,j,k in false_negatives:
  try:
    img = mpimg.imread(project_dir + "/extracted_faces/" +
challenge_type+ "/"+ i+ ".jpg")
    plt.axis('Off')
    plt.imshow(img)
    plt.show()
    print(i)
    print(k)
    print(j)
  except:
    print()

"""# Analysis"""

!pip install pandas

import pandas as pd

tp = pd.DataFrame([k for i,j,k in true_positives])
fp = pd.DataFrame([k for i,j,k in false_positives])

"""## Mean and STD"""

tp_mean = round(tp.mean().values[0], 4)
fp_mean = round(fp.mean().values[0], 4)
tp_std = round(tp.std().values[0], 4)
fp_std = round(fp.std().values[0], 4)

print("Mean of true positives: ", tp_mean)
print("Std of true positives: ", tp_std)
print("Mean of false positives: ", fp_mean)
print("Std of false positives: ", fp_std)

"""## Distribution"""

tp[0].plot.kde()
fp[0].plot.kde()

print(float(tp.max()))
print(float(fp.min()))
```

```
"""The max of positives overlaps with the minimums of the negatives.
This overlap makes a single distinguising threshold difficult to
choose

## Using 2 Sigma, which is 95% confidence to get finetuned threshold
"""

threshold = round(tp_mean + 2 * tp_std, 4)
threshold
```