Appendix 1: **Daily Activity Tables**

We participated in a comprehensive orientation. After the orientation we went back to our departments and were given information about our departments by our supervisors. Then the topics we will study during our internship were distributed. My topics are Face Detection with AI/DeepLearning and Face Recognition.

Date	Supervisor's Name	Signature
03.07.2023	Ramazan TERZİ	

Since we never had the opportunity to work on the selected topics before internship, today was a training day. All interns were given instructions by supervisors. We decided to spend a few days doing a literature review and also had the opportunity to do a quick research before literature review.

Date	Supervisor's Name	Signature
04.07.2023	Ramazan TERZİ	

I deepened my research towards the technical and theoretical dimensions of FaceNet and then ArcFace technologies. I furthered my research by reading articles "FaceNet: A Unified Embedding for Face Recognition and Clustering" and "ArcFace: Additive Angular Margin Loss for Deep Face Recognition". I mainly used the titles and contexts in these articles.

Date	Supervisor's Name	Signature
05.07.2023	Ramazan TERZİ	

I thoroughly researched and studied the theoretical mathematics behind Face Recognition. Meanwhile, I examined the existing examples from a technical point of view.

Date	Supervisor's Name	Signature
06.07.2023	Ramazan TERZİ	

I researched Loss Function, Ranking Loss, Triplet Loss, FaceNet & ArcFace Architecture, RPN, R-CNN. I've studied the math behind them.

Date	Supervisor's Name	Signature
07.07.2023	Ramazan TERZİ	

I continued my research through the serengil/deepface library and various tutorials. I created a test.ipynb file on Colab and made some tests about its principles of working on a pre-made template.

Date	Supervisor's Name	Signature
10.07.2023	Ramazan TERZİ	

I created a small dataset by taking photos of fellow interns, one biometric and one smiling; 10 Photos total. In order to not have any parameters in these photos, I took them on a white background and zoomed very close to the face. I have thoroughly studied the serengil/deepface Face extraction part. I found out that Deepface uses VGG-Face, Open face, Face Net, FbDeep Face.

Date	Supervisor's Name	Signature
11.07.2023	Ramazan TERZİ	

I worked on the five layer steps of the Layered Architecture. I realized that two convolutions would not be enough for the assigned problem definition. I prepared architectural decals for the model. I researched the use of ready-made models. Basically, there was some code for the model that I looked at the explanations for. It wasn't connect any base that one convolution line specifically examined. I realized that just create the part of inputs and outputs.

Date	Supervisor's Name	Signature
12.07.2023	Ramazan TERZİ	

I learned what Keras representation dropout layer is and how it works. I started working on Face Feature capture with Convolution Layers, which is one of my goals. I created a model on colab/test.ipynb. I combined the face cropping part and the detect part from the whole photo data into a single function. I made a list in order to progress through the Face Recognition pipeline common 5 steps: Detect, Align, Normalize, Represent, Verify.

Date	Supervisor's Name	Signature
13.07.2023	Ramazan TERZİ	

I experimented with technologies I would use to choose which one to use. I prefer to use ResNet -mainly based of VGG- SSD (Single Shot Multibox Detector) with open-cv, Python. I watched some tutorials to learn how open-cv works. I learned more detailed pieces of information such as BGR to HSV, drawing something in the camera screen open-cv, object detection details, and Shi-Tomasi corner detection. I made the foundations to get into practice. I did little mock tests.

Date	Supervisor's Name	Signature
14.07.2023	Ramazan TERZİ	

I made a lot of progress in my experiments on the test.ipynb file I created on Colab because I improved the codes I wrote last week. I completed the stages of detecting a face directly from the photo, drawing a bounding box around the face, making face extractions, and writing a model with keras. While the codes I wrote did not pose a problem when I used ready-made data sets prepared to be used on behalf of famous names or Face Recognition; when I tried to detect photos I took myself, some elements in the photo that were not faces were detected as faces. To solve the problem, I retrained the model using the VGG-Face dataset. Then I took photos of fellow interns in different environments and added them to the dataset I created. The locations where the photos were taken were different, I shot again with a focus on the face. I introduced my growing dataset with my new data to the model and waited for the face bounding box to appear after cropping and extracting. It was highly successful.

Date	Supervisor's Name	Signature
17.07.2023	Ramazan TERZİ	

After the tests I made on Colab/test.ipynb and the codes I wrote, the more I added photos to the data set I created and enlarged it, the more it started to label different objects as faces beside the real faces it detects. I decided to write a Face-Detection from scratch, including passing VSCode Local to Notebook and writing and training the model myself. I finished the parts of Bounding Box face detection, labeling people/faces with Python label me, classification & regression (coordinates bounding box), localization-loss function, and class-binary cross entropy/face class. I created my model with Keras functional API.

Date	Supervisor's Name	Signature
18.07.2023	Ramazan TERZİ	

During the coding phase, I first got an open-cv error. I solved this problem by rebuilding the library locally in my notebook. I also saw that TensorFlow installation was causing this problem and I edited the packages. I created a dataset by obtaining 30 frame images from a real-time video. At this stage, I had to re-scan the literature, mainly GitHub. To put it briefly, I actively conducted a literature review throughout the entire process. I combined the dataset I created with the frames I bought and the dataset I created with the photos of my intern friends and labeled it again. I examined the Matplotlib collected samples section. Datasets and training of the model etc. I divided the test, train, and val files I created for the situations and made the training testing and validating partition. I finished the model coding of the structure I built from scratch. I analyzed the collected samples using Matplotlib and divided the data into training, testing, and validation sets. I applied image enlargement techniques to enlarge the dataset and made it approximately 30 times larger. In addition, I have collected and annotated a large number of images of the face from the frames that are very important for training the deep learning model. I will focus on fine-tuning the model using insights from the validation set and making final evaluations on the intact test set. I need to try to do some regularization or I'll change my neural network architecture.

Date	Supervisor's Name	Signature
19.07.2023	Ramazan TERZİ	

I continued my project using Bounding Box Augmentation for object detection Albumentations.ai. My starting point for this was pascal_voc. This is briefly [x_min, y_min, x_max, y_max] and minimum values indicate left corners and maximum values indicate right corners. Then I extracted the values by normalizing these coordinates. With the support of open-cv and JSON, I load test images and annotations, extract coordinates (coordinates to vector), and rescale then I have completely finished the match image resolution part.

Date	Supervisor's Name	Signature
20.07.2023	Ramazan TERZİ	

I learned that the random brightness function is inside the augmentation function. I started this part by adopting the SSD Architecture, which is the same model used in TensorFlow object detection. I decided to take advantage of the capabilities of a famous architecture, VGG-16, and integrate it into my model. I have made the necessary adjustments to tailor it to our specific needs. Specifically, my object detection model includes a classification model and a regression model. Given that VGG-16 primarily functions as a classification model, I have excluded the last classification layers to be replaced with custom layers tailored to my needs. To achieve this I effectively removed unnecessary layers by setting the "include top" parameter to false in the VGG-16 model. After the configuration, I evaluated the VGG model. When running this code for the first time, I needed additional requirements and this solved my problems. I proceeded by running the code and examining the performance of the VGG model.

Date	Supervisor's Name	Signature
21.07.2023	Ramazan TERZİ	

After completing the VGG-16 integration and setting the "include top" to remove the last layers, I ran the code to inspect the VGG model. The VGG model is quite important with around 14.7 million parameters. I trained the entire model from scratch (with VGG's own data set) and used the available knowledge from the image classification tasks as I fine-tuned it. The model consists of multiple convolutional neural network layers including convolutional and max pooling layers, providing valuable features for my object detection model. The final layers are represented as "none" and "512", which will adapt to fit the input image sizes and channels. This flexibility in the architecture increased its robustness, allowing me to seamlessly integrate VGG-16 into my object detection model. Using point summaries in Keras, we can get useful information about various neural networks. I proceeded to build my neural network and run the code accordingly. The first step involves determining the input layer, which is very important when building a neural network. I made sure that VGG-16 was seamlessly integrated into my model to optimize its performance in the next steps. I have completed the face detection with face detect rectangle functionality, and it is now fully operational. Additionally, it is effectively functioning with the camera. After finishing the Detection part, I switched to the Recognition part. This time, I continue by using the serengil/deepface library by not writing a model from scratch. My remaining time was spent with a serious TensorFlow problem.

Date	Supervisor's Name	Signature
24.07.2023	Ramazan TERZİ	

I installed the Deepface library completely on my computer. I added some code to run it. I combined my Face Detect project with the Deepface library. I got a working version. However, I did not add the face recognition part from the camera. It just detects and labels the face. I set up experimental sets and got some erroneous results while defining and cosine similarity of faces in my own dataset that I wanted the program to recognize in the version it was running. I took notes and made improvements by thinking about the causes and solutions of these errors.

Date	Supervisor's Name	Signature
25.07.2023	Ramazan TERZİ	

I started the reporting process of the whole project and the outputs that I received on my tests in the program the previous day. During this process, I held meetings with my supervisor Ramazan Terzi for feedback. We also attended a seminar given by the Deputy Head of the Digital Transformation Office.

Date	Supervisor's Name	Signature
26.07.2023	Ramazan TERZİ	

I have prepared a report and completed my false negative experimental by getting new test outputs from the program.

Date	Supervisor's Name	Signature
27.07.2023	Ramazan TERZİ	

I have completed my internship exit procedures. I received my Internship Completion certificate and I had my report signed. We had some technical conversations with our supervisor and we attended meetings and seminars.

Date	Supervisor's Name	Signature
28.07.2023	Ramazan TERZİ	

28.07.2023

Face Recognition and Face Detection

Beyzanur YILDIZ

Introduction

This project delves into the innovative world of Computer Engineering and Artificial Intelligence (AI), focusing on Face Detection and Recognition using Deep Learning. We employ convolutional neural networks (CNNs) and various machine learning techniques. This report outlines the project, critical performance indicators, issues encountered, their solutions, and the overall outcomes.

Methodology and Performance Metrics

The project commenced with a literature review of key concepts in AI face detection and recognition, including FaceNet, ArcFace, Loss Function, Ranking Loss, Triplet Loss, RPN, and R-CNN. The mathematical theories were researched, and the technical aspects were examined. We utilized the serengil/deepface library, TensorFlow, and Python's OpenCV library. We created a dataset of images featuring fellow interns. These images were taken against a white background to minimize distracting parameters. A test.ipynb file on Google Colab was set up for experimental development and testing. We implemented a five-step Face Recognition pipeline: Detect, Align, Normalize, Represent, and Verify. This pipeline proved essential in assessing the model's performance. Performance indicators included precision (correct detections vs false positives) and recall (correct detections vs false negatives). Precision assessed the model's reliability, while recall indicated the model's sensitivity in detecting faces. The F1 Score combined precision and recall for a balanced performance metric.

Problematic Issues and Improvement

We encountered a problem where the model began misidentifying non-face objects as faces when we fed it with more photos for training. This issue indicated overfitting, where the model started memorizing the training data instead of learning to generalize from it. To improve the situation, we decided to write a Face-Detection model from scratch using the VGG-Face dataset and Keras functional API. We retrained the model using a mixed dataset of photos from different environments and video frames. By expanding and diversifying the training data, the model improved in generalizing and reduced misidentifications. Another improvement point was increasing the model's robustness by integrating the VGG-16 architecture into our object detection model. It boosted the object detection model with its convolutional and max pooling layers.

Problem Analysis and Improved Solution

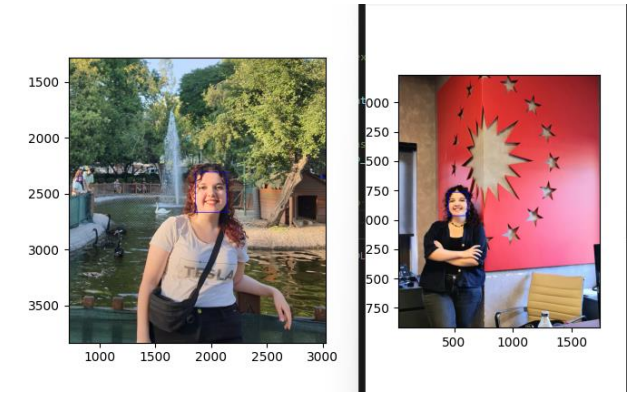
Further analysis revealed that the model could improve by implementing Bounding Box Augmentation using Albumentations.ai. This process involved normalizing the coordinates of bounding boxes and rescaling to match image resolutions. We integrated VGG-16 architecture into our model to enhance its feature

extraction capabilities. Custom layers were added to the VGG-16 model to adapt to our specific requirements. The model was fine-tuned to better detect faces by incorporating image augmentation techniques and insights from the validation set.

Conclusion

This internship provided an opportunity to delve into AI and Deep Learning, specifically in the field of face detection and recognition. We successfully developed a robust face detection and recognition model using deep learning techniques and various libraries. It effectively detects faces and labels them in real-time. Throughout the project, there was an active literature review process to ensure that the methodologies and technologies utilized were up-to-date and relevant. The project overcame various issues and limitations through innovative solutions, demonstrating the dynamic nature of AI and Deep Learning in solving real-world problems.

Images related to the project can be found in the below page of this report.



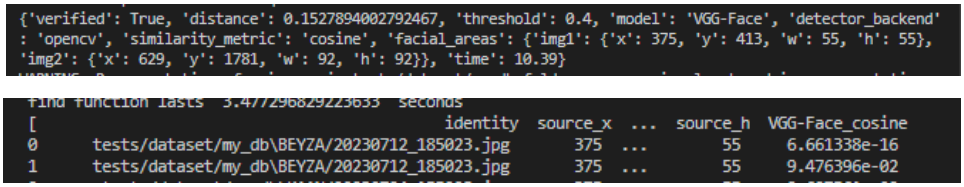


Figure-1: Face Recognition output example with DeepLearning

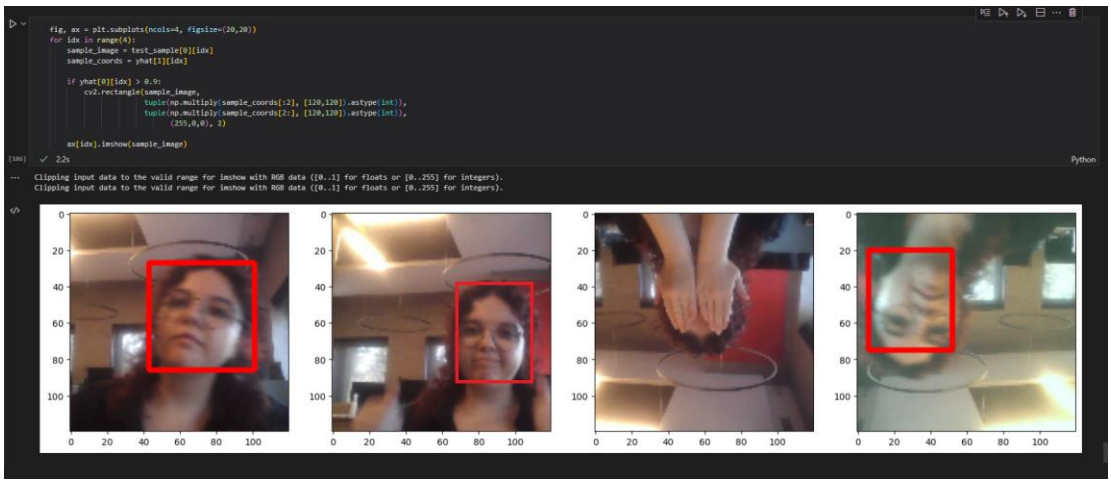
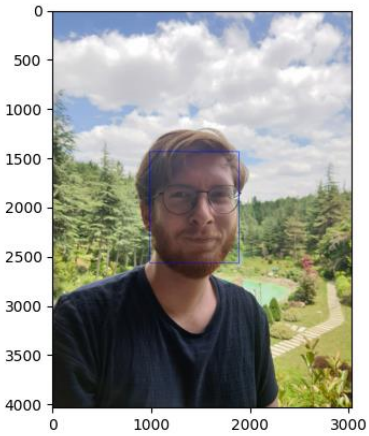
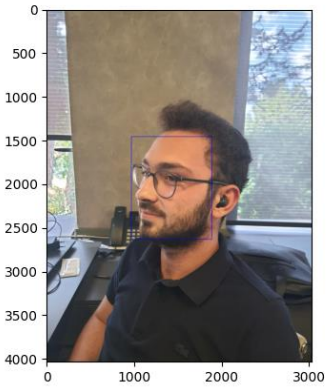


Figure-2: Face Detection output example with model that I built






```
[57 rows x 6 columns]]
(.venv) PS C:\Users\yildi\DF\deepface> & c:/Users/yildi/DF/deepface/.venv/Scripts/python.exe c:/Users/yildi/DF/deepface/main4.py
1/1 [==============================] - 1s 610ms/step
1/1 [==============================] - 0s 333ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 85ms/step
1/1 [==============================] - 0s 47ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 15ms/step
24/24 [==============================] - 0s 5ms/step
1/1 [==============================] - 0s 94ms/step
1/1 [==============================] - 1s 504ms/step
1/1 [==============================] - 0s 352ms/step
1/1 [==============================] - 0s 174ms/step
1/1 [==============================] - 0s 80ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 19ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 14ms/step
13/13 [==============================] - 0s 5ms/step
1/1 [==============================] - 0s 16ms/step
{'verified': True, 'distance': 0.07998224647182l5, 'threshold': 0.4, 'model': 'VGG-Face', 'detector_backend': 'opencv', 'similarity_metric': 'cosine', 'facial_areas': {'img1': {'x': 2916, 'y': 1925, 'w': 70, 'h': 70}, 'img2': {'x': 2750, 'y': 666, 'w': 62, 'h': 62}}, '
time': 6.24}
```

```
There are  104  representations found in  representations_vgg_face.pkl
find function lasts  1.587933778628174  seconds
[                                identity  source_x  source_y  source_w  source_h  VGG-Face_cosine
0      tests/dataset/my_db\EREN/20230710_131636.jpg       972      1494      1033      1033     2.220446e-16
1      tests/dataset/my_db\EREN/20230710_131639.jpg       972      1494      1033      1033     7.447577e-02
2      tests/dataset/my_db\EREN/20230724_153613.jpg       972      1494      1033      1033     1.877478e-01
3      tests/dataset/my_db\EREN/20230724_153611.jpg       972      1494      1033      1033     1.988702e-01
4      tests/dataset/my_db\EREN/20230724_153614.jpg       972      1494      1033      1033     2.661951e-01
5      tests/dataset/my_db\EREN/20230724_153605.jpg       972      1494      1033      1033     2.767245e-01
6      tests/dataset/my_db\EREN/20230710_091604.jpg       972      1494      1033      1033     2.863325e-01
7      tests/dataset/my_db\EREN/20230710_091603.jpg       972      1494      1033      1033     2.951926e-01
8    tests/dataset/my_db\EREN/20230710_131617(0).jpg       972      1494      1033      1033     3.131192e-01
9    tests/dataset/my_db\KAAN/20230724_155237(0).jpg       972      1494      1033      1033     3.383767e-01
10     tests/dataset/my_db\KAAN/20230724_155225.jpg       972      1494      1033      1033     3.485020e-01
11     tests/dataset/my_db\KAAN/20230724_155230.jpg       972      1494      1033      1033     3.788442e-01
12     tests/dataset/my_db\KAAN/20230724_155247.jpg       972      1494      1033      1033     3.914322e-01,                                identity  source_x  source_y  source_w  source_h  VGG-Face_cosine
0       tests/dataset/my_db\EREN/20230710_131636.jpg      2916      1925        70        70     6.661338e-16
1       tests/dataset/my_db\KAAN/20230724_155237.jpg      2916      1925        70        70     1.501943e-02
2    tests/dataset/my_db\KAAN/20230724_155237(0).jpg      2916      1925        70        70     1.816561e-02
3       tests/dataset/my_db\KAAN/20230724_155227.jpg      2916      1925        70        70     3.696242e-02
4       tests/dataset/my_db\EREN/20230710_131617.jpg      2916      1925        70        70     3.754781e-02
5       tests/dataset/my_db\KAAN/20230724_155235.jpg      2916      1925        70        70     3.946327e-02
6      tests/dataset/my_db\BEYZA/20230710_131826.jpg      2916      1925        70        70     4.274804e-02
7       tests/dataset/my_db\OMER/20230724_153649.jpg      2916      1925        70        70     4.398632e-02
8    tests/dataset/my_db\EREN/20230710_131617(0).jpg      2916      1925        70        70     4.808854e-02
9      tests/dataset/my_db\BEYZA/20230710_131826.jpg      2916      1925        70        70     6.597234e-02
10     tests/dataset/my_db\TOLGA/20230710_131706.jpg      2916      1925        70        70     7.315493e-02
11   tests/dataset/my_db\TOLGA/20230710_131706(1).jpg      2916      1925        70        70     7.315493e-02
12      tests/dataset/my_db\OMER/20230724_153648.jpg      2916      1925        70        70     7.998225e-02
13      tests/dataset/my_db\KAAN/20230710_131747.jpg      2916      1925        70        70     9.386593e-02
14     tests/dataset/my_db\TOLGA/20230724_153449.jpg      2916      1925        70        70     9.757211e-02
15     tests/dataset/my_db\TOLGA/20230710_131706.jpg      2916      1925        70        70     1.038567e-01
16   tests/dataset/my_db\TOLGA/20230710_131706(1).jpg      2916      1925        70        70     1.038567e-01
17     tests/dataset/my_db\BEYZA/20230712_185023.jpg      2916      1925        70        70     1.233299e-01
18      tests/dataset/my_db\KAAN/20230710_131747.jpg      2916      1925        70        70     1.256462e-01
19      tests/dataset/my_db\OMER/20230724_153648.jpg      2916      1925        70        70     1.299067e-01
20      tests/dataset/my_db\KAAN/20230710_131733.jpg      2916      1925        70        70     1.422180e-01
21     tests/dataset/my_db\TOLGA/20230710_131706.jpg      2916      1925        70        70     1.429856e-01
22   tests/dataset/my_db\TOLGA/20230710_131706(1).jpg      2916      1925        70        70     1.429856e-01
23      tests/dataset/my_db\EREN/20230710_131630.jpg      2916      1925        70        70     1.443466e-01
24      tests/dataset/my_db\KAAN/20230710_131738.jpg      2916      1925        70        70     1.595505e-01
25      tests/dataset/my_db\KAAN/20230710_131743.jpg      2916      1925        70        70     1.640316e-01
26     tests/dataset/my_db\BEYZA/20230712_185023.jpg      2916      1925        70        70     1.725890e-01
27      tests/dataset/my_db\KAAN/20230710_131730.jpg      2916      1925        70        70     1.739390e-01
28      tests/dataset/my_db\KAAN/20230710_131732.jpg      2916      1925        70        70     1.920862e-01
29      tests/dataset/my_db\KAAN/20230710_131747.jpg      2916      1925        70        70     2.174952e-01
30      tests/dataset/my_db\KAAN/20230710_131732.jpg      2916      1925        70        70     2.225168e-01
31     tests/dataset/my_db\BEYZA/20230712_185023.jpg      2916      1925        70        70     2.295564e-01
32      tests/dataset/my_db\KAAN/20230710_131730.jpg      2916      1925        70        70     2.450646e-01
33      tests/dataset/my_db\KAAN/20230710_131738.jpg      2916      1925        70        70     2.468477e-01
34      tests/dataset/my_db\EREN/20230710_131617.jpg      2916      1925        70        70     2.520827e-01
35      tests/dataset/my_db\KAAN/20230710_131738.jpg      2916      1925        70        70     2.617104e-01
36     tests/dataset/my_db\BEYZA/20230708_093404.jpg      2916      1925        70        70     2.625976e-01
37     tests/dataset/my_db\BEYZA/20230711_202309.jpg      2916      1925        70        70     2.687217e-01
38   tests/dataset/my_db\OMER/20230710_131806(0).jpg      2916      1925        70        70     2.694713e-01
39      tests/dataset/my_db\KAAN/20230710_131733.jpg      2916      1925        70        70     2.744710e-01
40      tests/dataset/my_db\KAAN/20230710_131747.jpg      2916      1925        70        70     2.755957e-01
41      tests/dataset/my_db\KAAN/20230710_131733.jpg      2916      1925        70        70     2.790809e-01
```

Figure-3: An example of an output where two different individuals, captured in different environments and with varying variables, yield a false-negative result during recognition, thus providing a true positive and erroneous similarity.

References:

- Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A Unified Embedding for Face Recognition and Clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015.

- Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2019). ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019.

- Serengil, S. (2021). DeepFace: Open-source facial recognition with deep learning. GitHub repository. Retrieved from https://github.com/serengil/deepface.