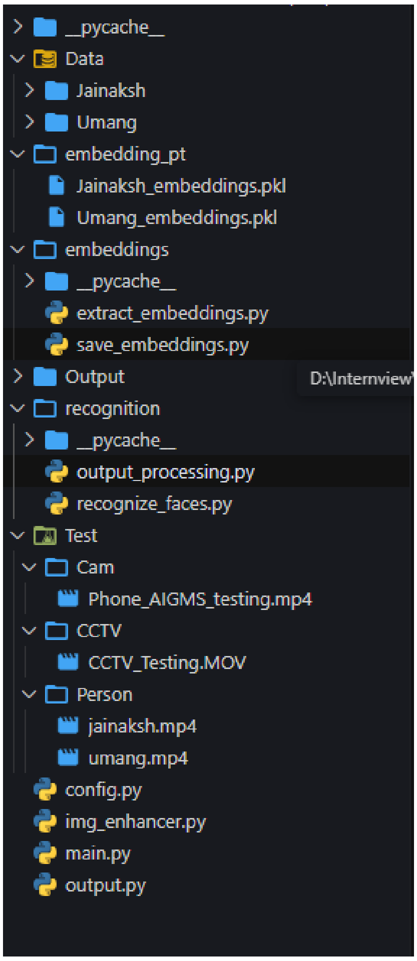
Maharishi Industries Pvt. Ltd. Interview Project Report

Task – Face Recognition

Project Overview

Implementing face recognition pipeline using Deep learning pretrained models. With goal to detect, extract face embedding and match with them with the input data to recognized person based on library of face data available. The project is structured in modular and segregated function.

Project File Structure



```
> __pycache__
v Data
  > Jainaksh
  > Umang
v embedding_pt
    Jainaksh_embeddings.pkl
    Umang_embeddings.pkl
v embeddings
  > __pycache__
    extract_embeddings.py
    save_embeddings.py
> Output                      D:\Interview
v recognition
  > __pycache__
    output_processing.py
    recognize_faces.py
v Test
  v Cam
      Phone_AIGMS_testing.mp4
  v CCTV
      CCTV_Testing.MOV
  v Person
      jainaksh.mp4
      umang.mp4
  config.py
  img_enhancer.py
  main.py
  output.py
```

- **Data/**: Contains the image dataset used for training the model. Images are grouped by person (e.g., Jainaksh and Umang folders).
- **embeddings/**: Contains scripts for extracting embeddings from images.
- **Embedding_pt/**: Stores the embeddings of each person in .pkl format.
- **Output/**: Stores the output files, as Annotated Video output.
- **recognition/**: Contains scripts for recognizing faces in videos using the extracted embeddings and processing the output.
- **Test/**: Contains videos used for testing, including videos of personals and video files from different sources (e.g., CCTV, Cam).
- **config.py**: Configuration file that contains paths and parameters (confidence Thresold) used in the project.
- **main.py**: Main script to execute the pipeline.

Tech stack utilized –

1. Python	2. Pillow	3. InceptionResnetV1
4. Tensorflow	5. Computer Vision	6. MTCNN
7. PyTorch	8. Deep Learning	9. FaceNet

How to Use -

1. Installing Dependencies
    a. With Python => 3.9
    b. Run in terminal pip install torch facenet-pytorch opencv-python Pillow tqdm
2. Adding Images to dataset
    a. In Data directory
        i. Create a New Folder for new Person Name and Drop all Images related to person
        ii. For existing Person drop all Images related to person into his/her name associated directory
    b. Adding Testing Data
        i. In Test Directory Drop the File within any associated directory or just into the Test directory
3. Extracting Embedding from Dataset
    a. Run main.py
        i. Check all persons in data directory are having their own embedding_pt.
4. Recognize Face in Test Video
    a. Once the main.py is running
        i. It will iterate after having all the faces pkl. Data with it over the Test data and detect face and recognize based on similarity with existing and Annotate the video

To Configure for your Directories edit Config.py

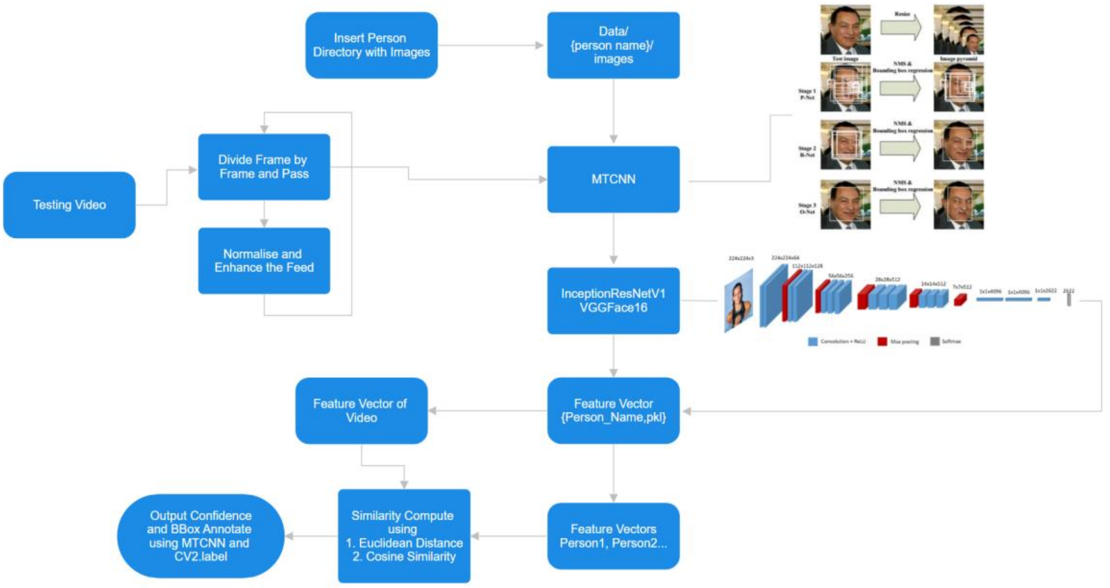
```
CONFIG = {

  'dataset_dir': 'Data/',  # Dataset containing person images

  'embeddings_dir': 'embeddings/',  # Directory to save embeddings

  'output_video_path': 'Output/output_video.mp4',  # Output video path

  'input_video_path': 'Test/Cam/Phone_AIGMS_testing.mp4',  # Input video path

  'annotated_video_path': 'Output/annotated_video.mp4',  # Annotated video output path

  'device': 'cuda' if torch.cuda.is_available() else 'cpu',  # Device configuration

  'confidence_threshold': 0.5  # Threshold for face recognition

}
```

Methodology

The system consists of the following main steps:

1. **Dataset Preparation**: Images of individuals (Jainaksh, Umang, etc.) are collected in specific folders with nomenclature {person name}.
2. **Embedding Extraction**: The images are processed using a pre-trained face detection model (MTCNN) and a face recognition model (InceptionResnetV1) to generate embeddings that uniquely represent each person's face.
3. **Face Recognition**: When a new video is provided, faces in the video are detected and compared with the pre-computed embeddings to identify the person. The recognition is based on the cosine similarity/Euclidean distance between the face embeddings.
4. **Video Annotation**: Once the faces are recognized, the video is annotated with the name of the person and a confidence score.

Work Flow


Insert Person Directory with Images
Data/ {person name}/ images
Testing Video
Divide Frame by Frame and Pass
Normalise and Enhance the Feed
MTCNN
Resize
Test image
Image pyramid
Stage 1 P-Net
Stage 2 R-Net
Stage 3 O-Net
NMS & Bounding box regression
InceptionResNetV1 VGGFace16
224x224x3
224x224x64
112x112x128
56x56x256
28x28x512
14x14x512
7x7x512
1x1x4096
1x1x4096
1x1x2622
2622
Convolution + ReLU
Max pooling
Softmax
Feature Vector of Video
Feature Vector {Person_Name,pkl}
Output Confidence and BBox Annotate using MTCNN and CV2.label
Similarity Compute using 1. Euclidean Distance 2. Cosine Similarity
Feature Vectors Person1, Person2...

Output

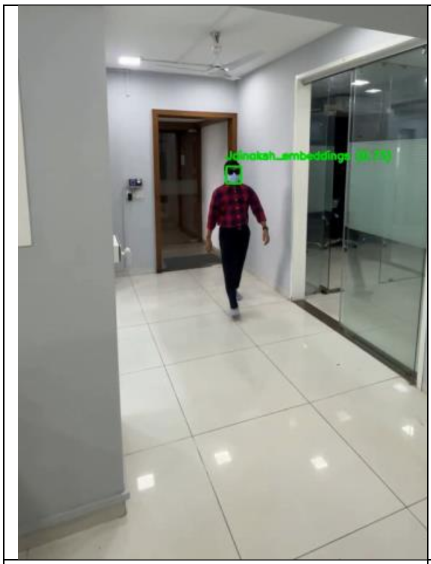
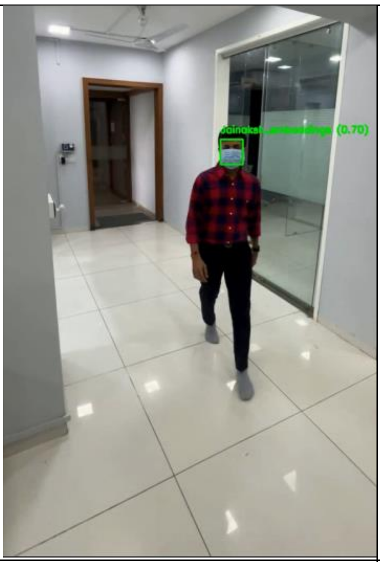
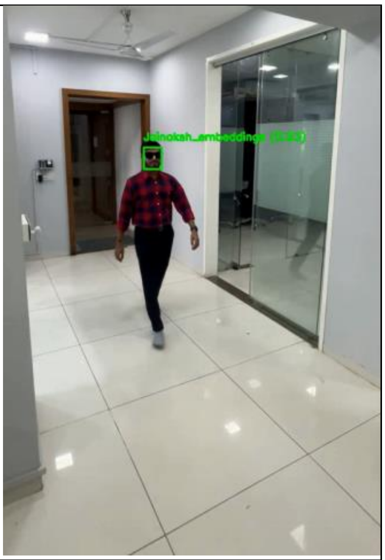
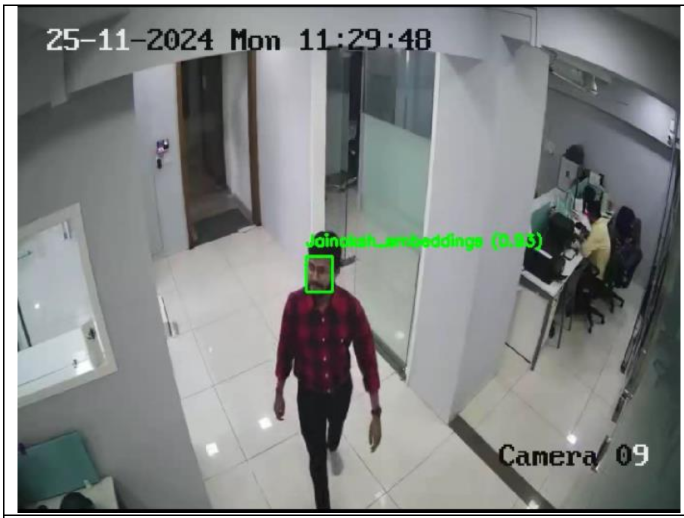
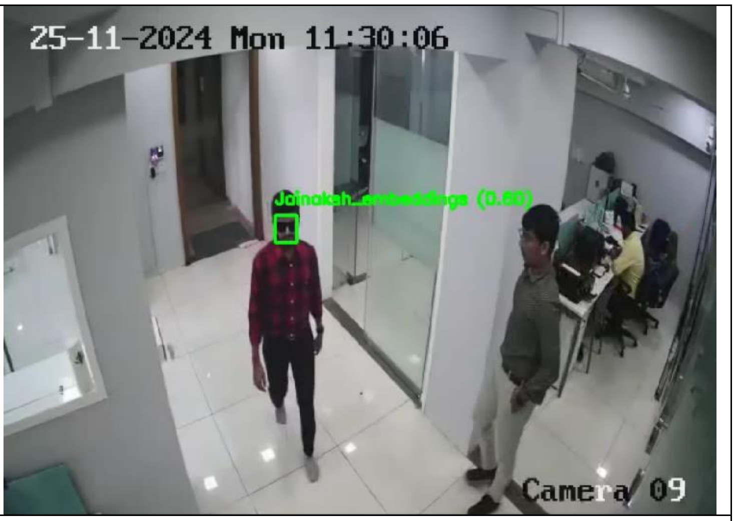
		

Fig. 1 Confidence threshold 0.5 Camera footage Jainaksh Sir with Mask and with Sunglasses	Fig. 2 Confidence threshold 0.5 Camera footage Jainaksh Sir with Mask and without Sunglasses	Fig. 3 Confidence threshold 0.5 Camera footage Jainaksh Sir without Mask and with Sunglasses

Table 1 – Output On a good Quality Camera Feed

 25-11-2024 Mon 11:29:48 Jainaksh_embeddings (0.53) Camera 09	 25-11-2024 Mon 11:30:06 Jainaksh_embeddings (0.60) Camera 09
Fig. 4 Confidence threshold 0.5 CCTV footage Jainaksh Sir without Mask and without Sunglasses	Fig. 5 Confidence threshold 0.5 CCTV footage Jainaksh Sir without Mask and with Sunglasses

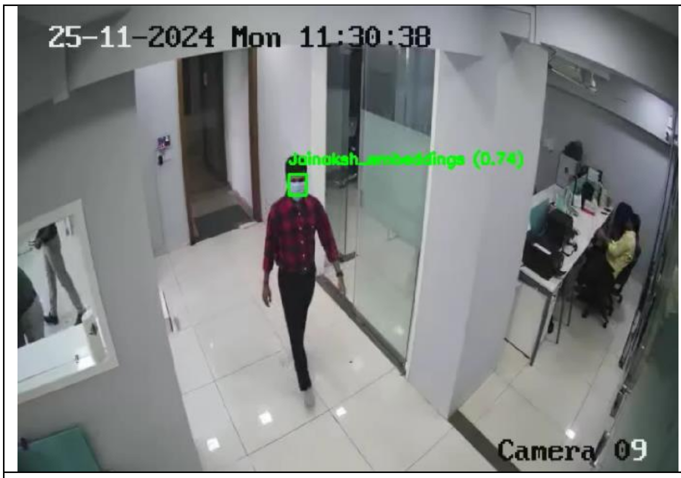
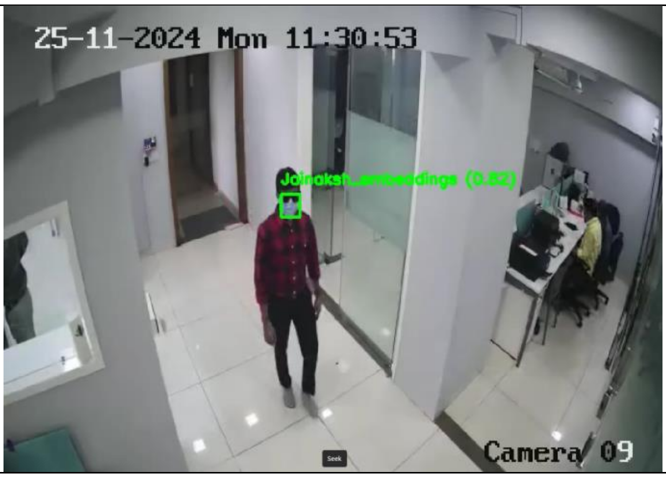
 25-11-2024 Mon 11:30:38 Jainaksh_embeddings (0.74) Camera 09	 25-11-2024 Mon 11:30:53 Jainaksh_embeddings (0.82) Seek Camera 09
Fig. 6 Confidence threshold 0.5 CCTV footage Jainaksh Sir with Mask and without Sunglasses	Fig. 7 Confidence threshold 0.5 CCTV footage Jainaksh Sir with Mask and with Sunglasses

Table 2 – Output On a good Quality CCTV

Improvements

A. Newer Workflow
    a. Collect Images and Annotation using Video recording of
        i. 1 min of person and detection of face as annotation and splitting video into images by frames with best possible quality
    b. Partitioning and Augmenting the Data
        i. To create a automated pipeline
    c. Building Deep Learning model and training
        i. Custom loss function using ArcFace and Optimizer
        ii. Train Neural Network
    d. Testing and Evaluation of Performance with similarity model
        i. with 1 min video or person to build a good dataset
        ii. implementing connection to Realtime feed

B. Improving the Similarity based workflow
    a. With Data quality improvise such case was seen with much inference in the Umang sir face data is having background and other features inference unlike Jainaksh sir data
    b. Trying Other Similarity computation algorithms
    c. Replacing MTCNN and vgg16 with other Face detection model as Dlib, hard cascade and etc.