Deep Learning-based
Face Recognition and Detection using Python

Hannah Mariam John, Kevin Reuben
Vellore Institute of Technology

Abstract

Face recognition has received a lot of attention in recent years and is regarded as one of the most promising uses for image analysis. In today's electronic environment, identification and verification have grown to be major issues. Human faces have a variety of variations, including posture, expression, location, and orientation.

Face detection is highly difficult due to a variety of factors, including skin tone, the presence of spectacles or beard, differences in camera gain, lighting, and image resolution. This paper analyses and studies a number of face detection techniques currently in use. This paper's major objective is to present or suggest a method that is a great option for face detection.

Introduction

Face detection and recognition is a challenging problem in computer vision that has a variety of applications, including identity verification, security, surveillance, and human-computer interaction. In this research paper
, we propose a deep learning-based face detection and identification system. We use a convolutional neural network (CNN) for face detection, which is trained on a large dataset of annotated face images.We also go over the difficulties and restrictions of our system, such as the requirement for a substantial amount of training data and the possibility of bias in the recognition procedure. Overall, our research shows that deep learning methods are excellent for detecting and identifying faces, and it also offers potential possibilities for future study.

Background study

Face recognition is a computer technology that uses artificial intelligence (AI) to find and recognise human faces in digital photographs. From basic computer vision methods to developments in machine learning (ML) to increasingly complex artificial neural networks (ANN) and related technologies, face recognition has advanced, leading to continuous performance improvements. It now serves as the foundation for a number of crucial applications, such as face tracking and face analysis.

Applications for face recognition use algorithms and ML to locate people's faces in bigger photos, which frequently include non-facial items like buildings, landscapes, and other human body parts like the feet or hands. One of the simplest aspects to recognise in a face is its eyes, which is often where algorithms for detecting faces begin their search. The computer might then try to identify the iris, mouth, nose, and nostrils. The algorithm does extra tests to verify that it has actually spotted a face once it determines that it has located a facial region.

The algorithms must be trained on huge data sets with hundreds of thousands of both positive and

negative photos in order to ensure accuracy. The algorithm's capacity to identify faces in an image increases with training.

Face recognition techniques might be appearance-based, feature-based, knowledge-based, or template-based. Each has certain benefits and drawbacks.

Evaluation of face detection systems is usually based on the metrics of accuracy, false positive rate, and processing speed. The accuracy of face detection can be affected by several factors such as illumination, pose, and occlusion. Due to the variety of elements, including attitude, emotion, location, and orientation, skin tone and pixel values, the existence of spectacles or facial hair, as well as variations in camera gain, lighting, and image quality, face detection in photographs can be challenging. Deep learning advancements in face detection have been made recently, and they have the advantage of beating conventional computer vision techniques greatly.

Face detection enhances surveillance and aids in the capture of terrorists and criminals. Since there is nothing for hackers to take or alter, like passwords, personal security is also increased.The majority of solutions for face detection and facial recognition are compatible with the vast majority of security software, making integration simple. Electronic identification which used to be done manually by a person was inefficient and frequently inaccurate. Automating the identification process with face detection increases accuracy and reduces processing time.

In recent years, face detection has become an active area of research, and several new algorithms and approaches are being proposed to overcome its challenges. Some of the current research directions include improving the accuracy of face detection in adverse conditions, developing real-time face detection systems, and incorporating facial landmarks and attributes into the detection process.

The use of face recognition has been growing in recent years due to the increasing demand for security and surveillance systems, biometric authentication systems, and face recognition-based human-computer interaction systems. The development of face detection algorithms has been influenced by advances in computer vision, machine learning, and deep learning.

Conventional face detection techniques use hand-crafted features and rule-based classifiers, while modern face detection systems rely on deep learning-based approaches, such as convolutional neural networks (CNNs). Deep learning-based approaches have shown superior performance in detecting faces in complex scenes, including varying illumination conditions, partial occlusion, and different poses.

Research in face detection continues to focus on improving accuracy, speed, and robustness in challenging environments. In conclusion, the field of face detection has seen significant progress in recent years and is expected to continue growing with the increasing demand for biometric-based systems.

Literature Survey

Enhancing the precision of object detection in the field of computer vision and precognition, including the detection of the human face and eyes, is one of the most difficult and unpleasant challenges. Researchers from all around the world are working with respect to this field, enabling you to use the well-found objects in a variety of packages.
An overview on the major human face recognition techniques that apply mostly to frontal faces, advantages and disadvantages of each method are given here.

- Hidden Markov Models have been successfully applied to speech recognition and action recognition in recent years. Ali et al. used this technique to investigate the recognition and detection performance of a one dimensional HMM for gray scale images. Faces were intuitively separated into parts like the eyes, mouth, nose, etc., which are connected to hidden Markov model states. Images should be transformed into either 1D temporal sequences or 1D spatial sequences since HMMs need a one-dimensional observation sequence and they are two-dimensional. Each image of height H and width W Is divided into overlapping blocks of height L. The amount of overlap between consecutive blocks is taken into consideration. For face recognition, the images in the training set represent faces of different people taken under different illumination conditions and for face detection, each individual in the database is represented by a HMM face model. The observation vector consists of Karheunen Loeve Transform(KLT) coefficients. A band sampling technique was used to recover a spatial observation sequence from a face image and a 1D vector series of pixel observations served as the representation for every face image. A block of L lines makes up each observation vector, and M lines overlap each other when new observations are made. An observation series is first sampled from an unidentified test image. It is then compared to each HMM in the model face database. The match with the highest likelihood is selected as the best match and the relevant model reveals the identity of the test face.
  This approach allows fast implementation due to the breaking of the face template into short observation vectors. Thus, the proposed model provides satisfactory recognition rates but it is computationally very complex as the size of the observation vectors is very large. Moreover, the algorithm is limited to frontal and upright faces and has poor performance for low-intensity photos. It is possible to improve the research to accommodate intensity and pose variant facial images.

- Face recognition technology based on CNN (Convolutional Neural Networks) has emerged as the primary technique used in the field of face recognition with the development of deep learning.

  Neural networks can be divided into two kinds- biological and artificial neural networks. An artificial neural network is a data model that processes information and is

similar in structure to the synaptic connections in the brain. Neural network is composed of many neurons, the output of the previous neuron can be used as the input of the latter neuron. This unit is also called the Logistic regression model. When many neurons are linked together and layered, the structure is called a neural network model.

Wang et al. studied the basic principles of Artificial CNN and constructed the convolutional and down-sampled layers of CNN by using the corresponding functions in Opencv to process the images. At the same time, the basic principles of multi-layer perceptrons were studied using the Python library. To simplify the CNN model, the convolution and sampling layers were combined into a single layer. Thus, the CNN design contains the following layers of structure- two convolution plus sampling layers, a fully connected layer and a softmax classification layer. The first layer is the face image which is a grayscale image. The face data set after the face image which is collected and processed is the input of the convolution neural network. The first layer after the input layer is the first convolutional and downsampling layer. After the convolution operation is performed, the image is down sampled to the maximum. The input to the second convolution plus sample layer is the output of the first convolution plus sample layer. The size of the input image in both the cases is calculated. Similar to the operation of convolution plus sampling layer in the first layer, the image is convolution processed first and the subsequent image under maximum downsampling is the resulting image.

The advantages of the proposed system include faster processing, automation of the identity, breach of privacy, massive data storage, best results, enhanced security, real time face recognition of students in schools and colleges, employees at corporate offices, smartphone unlock and many more in day to day life while some of the disadvantages include the costing, poor image quality which may limit the effectiveness of this system,poor reliability for varying face angles, massive storage requirement for the system to work effectively.

- Smitha and Varsha et al. proposed and implemented a new face detection method which combines the Skin color detector and the Template matching method. In the first stage, the skin color detector uses the YCbCr color space to detect skin and non-skin regions. In the YCbCr, Y channel represents the luminance component of the images, and both Cb and Cr represent the chrominance components of the images. Skin regions consist of faces or non-faces, while non-skin regions consist of only non-faces. The disadvantage of the skin color detector is that it detects non-face regions in addition to faces because it only detects skin pixels. The result of the skin color detector is applied to the template matching method. Template matching involves the segmentation of images. There are various methods for image segmentation

such as template matching, edge detection, clustering, boundary detection, thresholding, texture matching. But in the proposed method, only edge detection and template-matching have been used. The Sobel edge detector is used to detect the edges. The trained images are used in template matching, making it easy to find the location of eyes, nose, mouth etc., thereby allowing us to detect the faces and remove the non faces. The results have proved that the proposed method has detected very few non faces than the skin color detector. Thus, it has improved the face detection process by reducing the false positives (non faces detection).

- Sammer Aqib Hasmi et al. utilizes a multi-task cascaded convolutional network (MTCNN) which is a framework developed to aid with face detection and face alignment. It includes 3 degrees of CNNs that can identify faces by landmarking areas that include the eyes, nostrils and mouth. The first degree uses a simple CNN to just produce candidate home windows. The second degree refines the candidate home windows through a more complicated CNN. The third degree makes use of another CNN which is more complex than the others to refine the result with all the facial features having been landmarked along with an output of their positions.

  MTCNN is a method used for face detection and alignment based on convolutional neural networks (CNN) which are the type most commonly used to identify patterns in images and videos which means both face detection and alignment can take place at the same time.

  MTCNN has three convolutional networks namely P-Net, R-Net and O-Net. First an image is passed through this model and immediately an image pyramid is to be created by resizing it. An image pyramid is created in order to detect faces in various sizes. So, therefore we create multiple copies of the same image in different sizes. What P-Net does is it returns the coordinates of a bounding box if it detects a face. What R-Net does is it returns the coordinates of newer more accurate bounding boxes with bounding boxes of more confidence levels. After each stage, the boxes with lower confidence are eliminated. Then the image is passed to O-Net after resizing the boxes, O-Net returns three output values, namely, the coordinates of the bounding box, the coordinates of the 5 facial landmarks and the confidence level of each box. And finally, after removing the bounding boxes with lower confidence levels we should only get one bounding box for every face in the image.

  There are many known methods of face detection in today's world such as the Haar Cascade and Viola-Jones. The MTCNN detection method had been compared with the above-mentioned methods and output accuracy levels had been determined with

MTCNN method having the highest accuracy.

- Over the last few decades lots of research and work has been put into face detection as it is a viable way to identify a person without the need of human cooperation. Since lots of methods have been introduced for detection and recognition, Fazian Ahmad, Aaima Najam and Zeeshan Ahmed et al. have evaluated the various face detection methods and provide findings on the most accurate approach based on tests on various face rich databases in terms of subjects, pose, emotions and light.

  AdaBoost classifier is used with Haar and Local Binary Pattern features whereas Support Vector Machine classifier is used with Histogram of Oriented Gradients for face detection evaluation.

  Haar-like features are represented by taking a rectangular part of an image and dividing that rectangle into multiple parts which therefore results in a large set of features, then the boosting algorithm AdaBoost is usd to speed up the process. Also, the number of features obtained are reduced to only the critical features by using the boosting algorithm AdaBoost.

  The LBP operator labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number. The SVM classifier has been used with HOG features for face detection. The SVM captures the dissimilarity between two facial images. To reduce the pose variation and illumination in the images passed to the algorithm, two actions take place before the processing stage to help give a more accurate recognition result. 1)Eyes detection has been used to remove head turn, tilt and slant.2) Histogram equalization has been performed.

  With the use of the five datasets, the performance of these approaches have been evaluated and in the current system, Haar-like features reported relatively well but it has much more false detection than LBP.

- Face recognition is a technique in which the identity of a human being can be identified using one's individual face. The objective of this paper is to find an ideal method to detect and recognize human faces using OpenCV and Python use of multiple datasets containing various appearances of the individual. This method serves in a variety of instances like for security, military, schools, colleges and so on.

  Some of the major techniques already being used include Eigenface which use eigenvectors to represent the different variations in the faces. Graph matching is an algorithm which recognizes objects in an image based on a graph representation extracted from other images from the datasets provided. The Hidden Markov Model divides the face into parts such as eyes, ears, nose, etc and matches by comparing it with the stored dataset.

Template matching is a technique in which the image passed is represented as a two-dimensional array of values which can be compared using Euclidean distance with the single template that represents the whole face.

Tejashree Dhawle, Urvashi Ukey, Rakshandha Choudante et al. concluded that for this arrangement to be successful, firstly the datasets are needed. The test image would be passed through and the face in the image would be encoded using various packages and then these encodings would be passed through a file "recognize_faces_images.py" which contains all the required methods and techniques needed for the process of identification and the use of OpenCV comes to be useful by using its libraries to enhance the outcome or results in the face recognition system. With this system a variety of advantages follow such as faster processing, enhanced security, large data storage, best results. But a few disadvantages follow as well like the cost, high quality/definition cameras are required and the use of poor-quality cameras will reduce the effectiveness of this system.

The use of python programming and OpenCV makes it a simple method that can be incorporated by anyone according to their requirement. The discussed system would also be a viable option to many due to its user friendly and cost-efficient system. Hence by the use of python and OpenCV the face recognition system can be designed for various purposes.

Applications of facial recognition technology

Immigration:

Biometric technology makes travelers' journey through an airport seamless and enjoyable. Facial recognition algorithms used by automated terminals and entrance control tourniquets prove efficient for immigration control.
For example, Ljubljana Airport successfully tested facial recognition technology for passengers' identification at the boarding gates, which reduced boarding times at least by 75%.

Healthcare:

Face recognition algorithms have revolutionized and improved many operations in the health sector. These changes can be seen in security systems, patient verification, diagnosis establishment, tailoring treatment, and caretaking robots.

Facial recognition technology has applications in the health field that go far beyond simple surveillance. Face analysis facilitated proper drug management, early identification of some genetic disorders, and pain control.

The following are some examples of how face recognition technology is most frequently used in healthcare:

Patient registration and evaluation

At healthcare facilities, the check-in and check-out procedures are quite important. Face identification technologies streamline and perfect these procedures, occasionally saving lives in the process. A facial recognition technology excludes medical insurance fraud by promptly confirming a sick person's identity and preventing fraudulent identification.

Monitoring and diagnosing the sick

Facial recognition tools are combined with other modern technologies like sentiment analysis in real-time to make more accurate diagnosis . The solutions obtained can be used to get detailed information about a patient, discover pain sources, monitor health conditions, and even define early signs of serious diseases.

Caretaking robots

The ways that some services are provided reflect how healthcare systems are becoming more computerized and automated. Caretaking robots equipped with facial recognition software have been developed to take care of the sick at home or at a hospital. Due to the robot's ability to recognise a person from a past engagement, the technology guarantees a tailored interaction with a patient.

For instance, a caretaking robot who has been looking after an old person for some time may decide whether medication is given as directed.

Steps Involved in Face Recognition

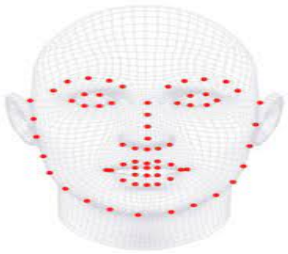
Face recognition is a subject of study where faces are categorized into sets that each belong to a certain person. Using Facebook as an example might make it simpler to comprehend. Facebook used to be able to recognise faces , but at that time, users had to tag specific people by clicking on the image and entering their name. Facebook can now automatically tag each and every person in a photograph. Facial recognition techniques enable this. In Python this task can be achieved using pre-trained convolutional neural networks and OpenCV. The whole program relies heavily on various libraries.So modules paths, face recognition, argparse, picle and os have to be imported into Python Project . First, some images of the person we wish to recognize have to be collected either manually, or by the application of Microsoft's Bing API for searching. The dataset must preferably contain at least 30 images of each person. In the training photographs, no other people should be present.

There are 4 main steps used in face recognition systems:

Face Detection: The detection process involves scanning the image to determine whether any area has full human faces or partial features by noting down the coordinates of each one and drawing a bounding box around it. This step is crucial to the entire process as the technology must catch a face alone or among other people quickly and precisely.

Face Alignments: Normalizing the face in face recognition is the process of aligning and standardizing facial images to a consistent format before analyzing them. This step deals with the

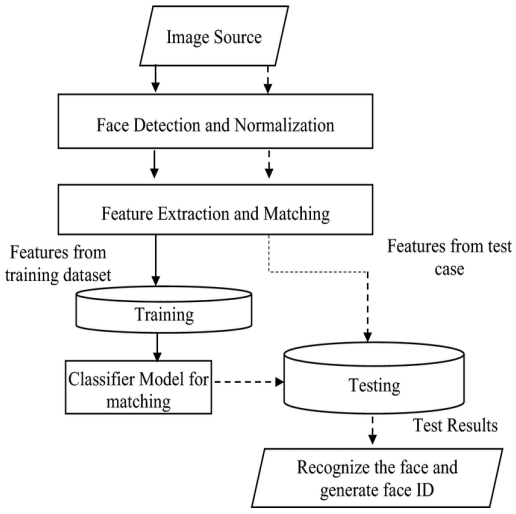
problem that an image might be looking in some other direction instead of looking straight into the camera. There are numerous solutions to this problem. Python's library resolves this by making use of the 68 landmarks present on any face which are the essential feature points such as eyebrows, corners of the mouth, eyes, nose, lips and so on. A machine learning algorithm is trained to locate these landmarks on any face.Then, the face is transformed using affine transformations so that eyes and mouth



are centered as accurately as possible.

Face Extraction:
The Deep Convolutional Neural Network is trained to generate measurements for each face. For the training process, three images are required at a time (the image of a known person, another image of the same person and an image of some other person). This step requires a large data set and a lot of computer power. However, it has to be executed only once.

Face Recognition:
The face being analyzed is compared and matched with one or more faces in our dataset. In order to achieve this, Python's library uses a Support vector machine (SVM). In principle, any other classification algorithm could also be used.



-

Implementation

Implementing a deep learning-based face recognition system using python and the face_recognition library

1. **Installing the face_recognition libraries**

   dlib: Dlib is a modern C++ toolkit that contains machine learning algorithms and tools for creating complex software in C++.

   ```
   # installing dlib

   pip install dlib
   ```

   face recognition: a library that can be installed after dlib

   ```
   # installing face recognition

   pip install face recognition
   ```

   Opencv: for some image pre-processing

```
# installing opencv

pip install opencv
```

Then we import the above mentioned libraries and then load the sample images.

2. Detecting and Locating Faces

The library face_recognition can detect faces on its own and does not require any supplementary code

The face in every picture gets converted into some numerical encoding, and gets stored in a list and all the labels(names of persons) in another list.

3. Sample Image Recognition

The installed library face_recognition is a deep learning-based library, which means it requires only a single picture to be able to train itself to recognise a person

We pass a picture of an unknown person into the face recognition system which then converts the face detected in the picture into an encoding.

After converting the person's Image into encoding, it tries to find the most similar encoding based on the distance parameter. The stored encoding with the least distance from the encoding of an unknown person will be the closest match.

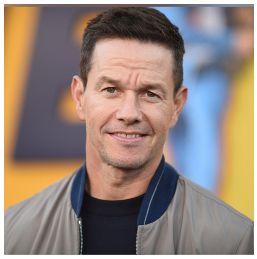
After getting the closest match encoding, we take the index of that encoding from that list and use indexing. We find the detected person's name.

Therefore, we arrive at a correct match of the face in the picture and the picture provided to the face recognition for training.

Then the face recognition system displays the image provided with a rectangular box around the faces in the picture and identifies them with their name below it as well as pulls only the faces of the people provided in the training set.

Results

First Training Set:



Bill Gates, Steve Jobs and Mark Wahlberg

First Sample Input

Image of Donald Trump (unknown), side view of Bill Gates (known), side view of Bill Gates (known).

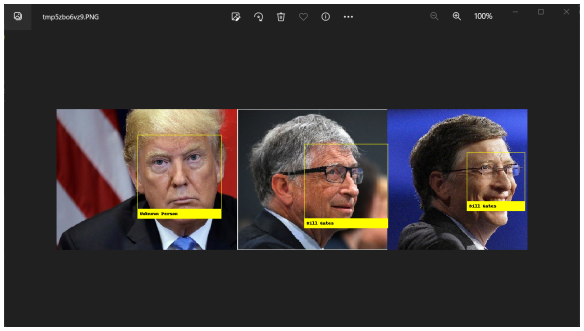
Code:

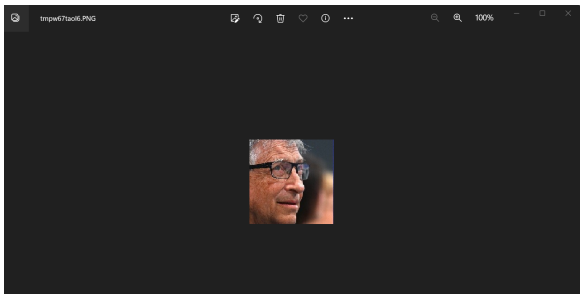
```
import face_recognition
from PIL import Image, ImageDraw


#encoding the face of bill gates
image_of_mark = face_recognition.load_image_file('./img/known/mark wahl.jpg')
mark_face_encoding = face_recognition.face_encodings(image_of_mark)[0]

#encoding the face of steve jobs
image_of_steve = face_recognition.load_image_file('./img/known/Steve Jobs.jpg')
steve_face_encoding = face_recognition.face_encodings(image_of_steve)[0]

#encoding the face of elon musk
image_of_bill = face_recognition.load_image_file('./img/known/Bill Gates.jpg')
bill_face_encoding = face_recognition.face_encodings(image_of_bill)[0]


#  Create arrays of encodings and names
known_face_encodings = [
  mark_face_encoding,
  steve_face_encoding,
  bill_face_encoding
]

known_face_names = [
  "Mark Wahlberg",
  "Steve Jobs",
  "Bill Gates"
]
```

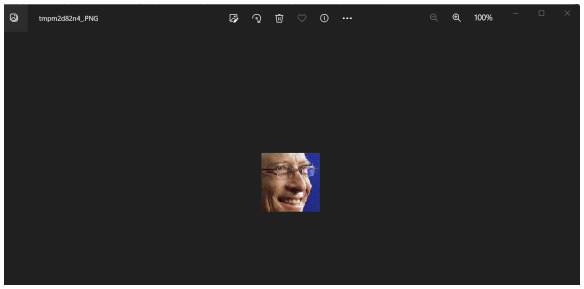
```
# Load test image to find faces in
test_image = face_recognition.load_image_file('./img/unknown/combination.jpg')

# Find faces in test image
face_locations = face_recognition.face_locations(test_image)
#Encoding the faces found in the image
face_encodings = face_recognition.face_encodings(test_image, face_locations)
pil_image = Image.fromarray(test_image)

draw = ImageDraw.Draw(pil_image)

for(top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
  matches = face_recognition.compare_faces(known_face_encodings, face_encoding)

  name = "Unknown Person"

  # If match
  if True in matches:
    first_match_index = matches.index(True)
    name = known_face_names[first_match_index]

  # Draw box
  draw.rectangle(((left, top), (right, bottom)), outline=(255,255,0))
```

```
  # Draw label
  text_width, text_height = draw.textsize(name)
  draw.rectangle(((left,bottom - text_height - 10), (right, bottom)), fill=(255,255,0), outline=(255,255,0))
  draw.text((left + 6, bottom - text_height - 5), name, fill=(0,0,0))

del draw

# Display image
pil_image.show()


# Loop through faces in test image
for(top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
        matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
        name = "Unknown Person"

 # If match
        if True in matches:
                first_match_index = matches.index(True)
                name = known_face_names[first_match_index]
                face_image = test_image[top:bottom, left:right]
                pil_image = Image.fromarray(face_image)

# Display image
                pil_image.show()
```

Output:



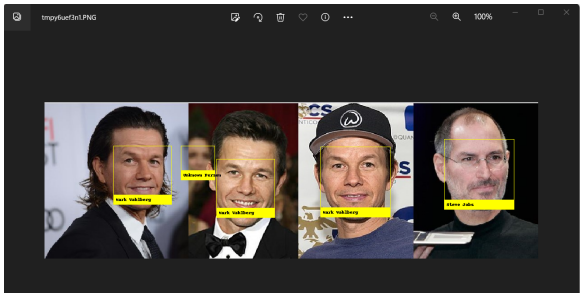


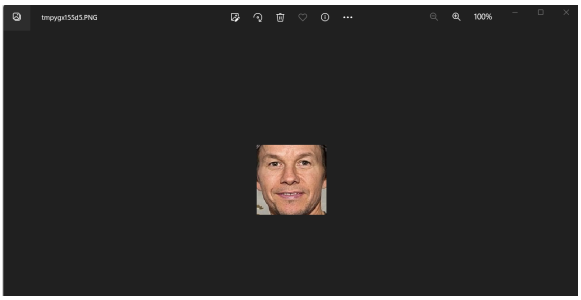


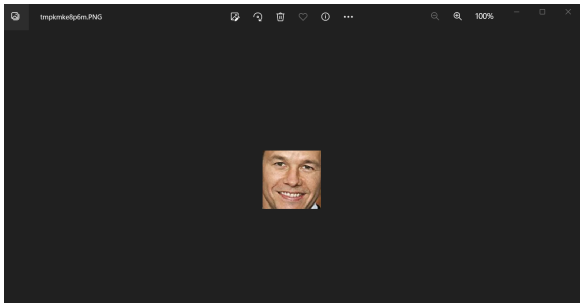
Second Sample Input:

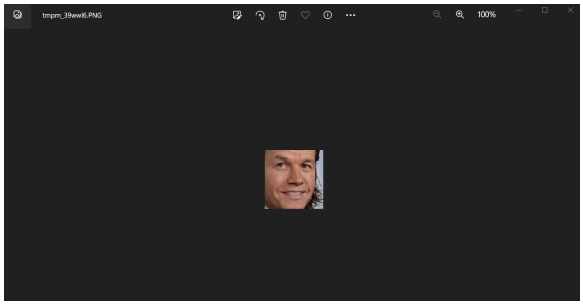
Mark Wahlberg with long hair (known), Mark Wahlberg (known), Mark Wahlberg with a cap (known), Steve Jobs (known)

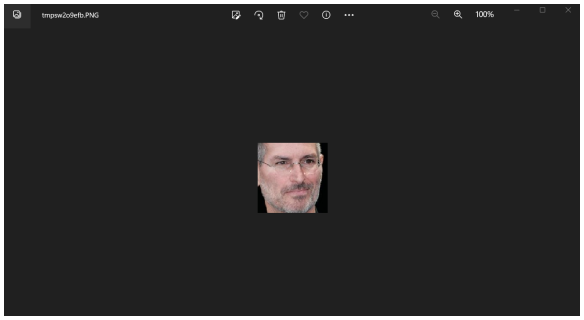
Output:











Challenges to Face Recognition Systems

Poses: Since face recognition systems depend on precise facial traits to identify a person, poses can become a big difficulty as the system might not be able to correctly identify the individual if the person's face is not visible or if the pose of the face differs drastically from the faces the system was trained on.

Illumination: The exposure of light can cause illumination that can significantly affect the performance of facial recognition systems, as changes in lighting conditions can lead to drastic changes in the appearance of facial features.

Low Resolution: Low-resolution pictures contain fewer pixels and less detail than high-resolution pictures, which means that they may not provide enough information for facial recognition systems to accurately detect and extract facial features.

Conclusion:

Face detection and facial recognition are two specific areas of intense research because they improve

communication between humans and robots or computer systems. The face detection and identification package of Python proves to be a fast and dependable tool, creating a system that is both user friendly and cost effective. Python is a high-level programming language, therefore the library can be used in a larger project simply as a face detection (identification) function without the requirement for in-depth theoretical knowledge of the used algorithms. We therefore believe it has a promising future. Further research on the potential of Python libraries for emotion detection would be fascinating. In the field of research on human-machine interfaces, this area is quite vast and intriguing. Using the findings from this study, it is possible to significantly improve the social aspects of robots or software packages that can adapt to the user. Thus, its feedback in human- machine interaction is extremely reliable.

References:

https://pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/

https://easternpeak.com/blog/facial-recognition-use-cases/

DANISH ALI , IMRAN TOUQIR , ADIL MASOOD SIDDIQUI, JABEEN MALIK , AND MUHAMMAD IMRAN , Department of Electrical Engineering, National University of Sciences and Technology, Pakistan

Jie Wang and Zihao Li ,2018 , Research on Face Recognition Based on CNNIOP.

Faizan Ahmad, Aaima Najam and Zeeshan Ahmed, 2019,Image-based Face Detection and Recognition: “State of the Art”, Department of Computer Science & Engineering, Beijing University of Aeronautics & Astronautics Beijing, China

Real-Time Face Recognition and Detection Using Python, 2022,Tayyaba Zamindar, Shradhatai Gangawane, Shital Kalane, Yogita Kalane

Laheeb Mohammad Ibrahim and Ibrahim A Saleh, 2008, Face recognition using artificial intelligence techniques, College of computer science and mathematics, University of mosul, Iraq

https://www.analyticsvidhya.com/blog/2022/04/face-recognition-system-using-python/