# MULTI-FEATURE EVALUATION ON FACIAL DATA USING CONVOLUTIONAL NEURAL NETWORKS

## A CAPSTONE PROJECT REPORT

Submitted in partial fulfillment of the requirement for the award of the Degree of

**BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING**


**BY**

U. Hrushikesh Chowdary (19BCI7073)
Chekitha Swayampu (19BCD7170)
Arikati Kushi (19BCE7104)

**Under the guidance of**

Prof. Lalitha Kumari P




SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
VIT-AP UNIVERSITY
AMARAVATI- 522237


DECEMBER 2022

# CERTIFICATE

This is to certify that the Capstone Project work titled **"MULTI-FEATURE EVALUATION ON FACIAL DATA USING CONVOLUTIONAL NEURAL NETWORKS"** that is being submitted by **U.Hrushikesh Chowdary(19BCI7073),Chekitha Swayampu (19BCD7170), Arikati Kushi (19BCE7104)** is in partial fulfillment of the requirements for the award of Bachelor of Technology, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Dr. Lalitha Kumari P
Guide

**The thesis is satisfactory / unsatisfactory**

**Internal Examiner** **External Examiner**

**Approved by**

**PROGRAM CHAIR** **DEAN**

B. Tech. CSE School of Computer Science and Engineering

# ACKNOWLEDGEMENT

We would like to take this opportunity to acknowledge everyone who has helped us in every stage of this project. Firstly, we are deeply indebted to our guide, Prof. Lalitha Kumari P, for assisting us with the project and providing a working environment. Then, we would like to thank the Dean of Computer Science Engineering VIT-AP, Prof. Dr. Sudha S V for providing us with a wonderful opportunity to make and complete a project.

We would like to express our gratitude to our parents who have boosted us morally and provided continuous support. It is a pleasure to thank my friends for convincing and encouraging us to take on and complete this task in the given time frame. Last but not least, we wish to express our gratitude and appreciation to everyone who has contributed directly or indirectly to the successful completion of this project.

# ABSTRACT

One biometric information technology that is easier to apply and has a wider working area than others, such as fingerprint, iris scanning, signature, etc. is a face recognition system. The most significant facial characteristics that are vital to social interactions are age, mood, and gender. Assessing age, emotion, and gender from facial images is an important task in intelligent applications such as access control, human-computer interaction, enforcement, marketing intelligence, and visual surveillance.

With face detection, the face region is separated from the background. The position and dimension of each detected face are roughly estimated via face detection. Based on the location points, face features such as the eyes, nose, mouth, and facial contours are found. Face detection is performed on live acquired images. Convolutional Neural Networks (CNNs)-based techniques for the classification problem have lately become popular due to their excellent performance in facial analysis. Geometric transformations or morphing are used to normalize the input facial image about geometric features like size and attitude. Faces are usually further normalized concerning photometric properties such as illumination and grayscale. In this research, we propose a novel end-to-end CNN approach for accurate age group and gender identification of unfiltered real-world faces. To distinguish between the faces of various persons and to offer reliable and consistent information about geometrical and photometric variances, feature extraction is carried out after the faces have been normalized geometrically and photometrically.

# TABLE OF CONTENT


| S.No | Title | Page Number |
|---|---|---|
| 1. | Acknowledgment | 2 |
| 2. | Abstract | 3 |
| 3. | Introduction | 5 |
| 4. | Objective | 5 |
| 5. | Requirements | 6 |
| 6. | Literature review<br>6.1 Age and Gender prediction<br>6.2 Facial recognition, expression, and landmarking | <br>7-8<br>8-10 |
| 7. | Methodology | 11-12 |
| 8. | Procedure | 12 |
| 9. | Frameworks<br>9.1 MediaPipe<br>9.2 TensorFlow<br>9.3 OpenCV<br>9.4 NumPy | <br>13-14<br>14-15<br>15<br>16 |
| 10. | Diagrams<br>10.1 Activity Diagram<br>10.2 Block Diagram | <br>17<br>18 |
| 12. | Code | 19-26 |
| 13. | Output | 27-28 |
| 14. | Conclusion | 29 |
| 15. | References | 30 |

# INTRODUCTION

Age, gender, and facial recognition data are crucial for many real-world applications, including crowd behavior research, online advertisements, biometric identity verification, video surveillance, human-computer interaction, electronic customers, and many more. A kind of biometric software called facial recognition creates mathematical maps of human facial traits and stores the information as faceprints. In the contemporary age, facial recognition is fundamental. In many circumstances in everyday life, including ID cards, passports, and other extremely significant identities, facial recognition and identification are used. Nowadays, the most common technique for verifying someone's identity uses face recognition. To stop ID fraud and identity theft, this technology is also implemented in a variety of other professions and sectors. Smartphones may be accessed using facial recognition technology as well. Three skills are necessary for the recognition involved in such tasks: the capacity to identify unfamiliar faces, the capacity to learn new faces, and the capacity to recognize existing faces. Although facial recognition as a concept is not new, this technology has significantly extended as a result of technological advancements throughout time.

# OBJECTIVE

This project aims to develop an algorithm that estimates a person's age, emotion, and gender. In this project, we consider the problem of multi-feature evaluation on facial data using convolutional neural networks for:

- Recognition of Emotion
- Recognition of Gender
- Recognition of Age
- Recognition of Facial Attributes

# REQUIREMENTS

The module's various requirements in the process of recognizing age, gender, and emotion and all the facial attributes are:

- The training of such neural networks requires the use of large datasets. Hence, pre-trained models are used to get accurate results.
- As the dataset is huge, the parameters for the CNN models required are higher, and this training process requires high computational power.

## Setup for implementation

**Windows Users:**

- python-OpenCV - Install using pip.
- pip install NumPy
- pip install matplotlib

**Linux / Mac Users:**

- pip3 install NumPy or apt-get install python3-numpy. We might need to use the command apt-get to install python3-pip.
- pip3 install matplotlib or apt-get install python3-matplotlib.
- apt-get install python3-OpenCV

# LITERATURE REVIEW

## 6.1 AGE AND GENDER PREDICTION

### 6.1.1 IN THE OPEN DOMAIN

Nowadays, most people use a variety of domains, online dating services, and social networking sites due to the quick spread of internet resources. These factors have led to many issues, including age and gender misrepresentation. [1] Recent studies have demonstrated the value of knowing the demographics of internet users; it has been established that internet users frequently inflate their sex, looks, age, and educational background. The goal of this work is to develop a classifier that can predict the deceiver's gender and age using machine learning. This study focuses on dishonest behavior. [1]Researchers used various individual and combined classifier algorithms, including the support vector machine, naive Bayes, and decision tree, to classify and analyze the text using open-source machine learning (J48).

The amount of fraud on the internet will be reduced thanks to this, especially in areas like online dating and job interviews. The SVM classifier produces the best results for gender prediction utilizing features (CFG), with an accuracy of 82.81%. However, by combining CFG and an SVM classifier, the best age prediction was made with an accuracy of 83.2%.

The gender classification resulted in only modest improvements over the baseline in the age and gender experiment in short open domain deceptive messages. [1]The findings of the age prediction, however, were subpar. Age and gender prediction are difficult jobs when trying to identify fraud in open-domain data since the overall result showed that the classifier did not do very well when attempting to do so.

### 6.1.2 MULTI-STAGE LEARNING

First, a saliency detection network with an encoder-decoder is presented to extract the interest area while minimizing interference from the complicated backdrop. [4]The Saliency detection network, in particular, is adapted from the convolutions semantic segmentation network, which can categorize each

pixel into two classes, "people" and others, and we only use the "people" regions to complete the prediction. [4]Second, a prediction network based on the VGG18-Net is suggested to determine the image's ultimate gender and age. On many classification problems, the DNN-based approaches have produced impressive results. Convolutional neural networks have been successfully applied to challenging computer vision applications, in particular.

Even though CNNs can accurately portray images, it is challenging to train two conventional big-scale CNNs sufficiently with a little dataset. The large storage requirements of the big-scale network models make them inconvenient for mobile devices.

## 6.2 FACIAL RECOGNITION, EXPRESSION, AND LANDMARKING

### 6.2.1 INDEPENDENT SPATIAL CHANNEL ATTENTION AND COMPLEMENTARY CONTEXT INFORMATION-BASED FACIAL EXPRESSION

[2]The use of attention mechanisms is currently popular in FER to address issues like occlusion and position variation. Both local and global contexts can be captured through it. Different strategies have been used recently to use attention. In this study, an end-to-end architecture for FER is suggested. It includes SCAN, a cutting-edge spatial channel attention net that acquires local and global attention per channel for each spatial location without requesting any data from the landmark detectors. [2]Even though CNN does a great job of portraying the image. With a small dataset, it is challenging to sufficiently train a traditional big scale CNN. The large storage requirements of the big-scale network models make them inconvenient for mobile devices.

Though the gain may not look pronounced, it is to be noted that the local patches input to SCAN is of size around 512*5* 5. The spatial size of 5 *5 is small.

It is also found that in our experiment the absence of SCAN brings down the performance of the model by 1.2 to 1.5% across the datasets.

### 6.2.2 EXPRESSION RECOGNITION USING FUZZIFIED PSEUDO-ZERKIN MOMENTS

[3]To test the efficacy and effectiveness of the suggested fuzzy facial expression detection system, we have chosen 414 facial photos from RaFD databases that were taken under the best possible lighting conditions. Therefore, each of the six fundamental facial expressions has 69 different photos. They start by examining various databases of various face photos to locate regions of membership functions (support sets). [3]A chromosome is then formed using the parameters of the membership functions Gaussian, 7-Shape, and S-Shape. We choose three characteristics with a total of five membership functions.

[3]This experiment's main goal is to discern apart emotions from partially obscured facial expressions. The human inference system was chosen as the inspiration because of this. With the help of structural characteristics and PZM, this fuzzy inference method produces a face expression identification system that is resistant to a variety of lighting conditions, partial occlusion, rotation, and noisy facial images.

The proposed technology now accurately captures five different facial emotions. However, when comparing the method of employing structural elements of the face, the recognition rate did not increase and almost reached the same accuracy. [3]Therefore, the suggested facial recognition system's accuracy cannot be improved by combining textural and structural characteristics.

### 6.2.3 FACE RECOGNITION ISSUES AND ALTERNATIVES

Over the past ten years, trustworthy facial recognition algorithms have developed quickly. Holistic features and local feature methods are two categories into which the conventional face recognition algorithms can be divided. Additionally, the holistic group can be broken down into linear and nonlinear projection techniques. The linear projection appearance-based approaches, such as principal component analysis, linear discriminate analysis, 2DPCA, Linear regression classifier, and independent component analysis, have produced good results in many applications.

Focuses on the manifold's local structure. These techniques map the face onto the eigenface images' linear subspace. The distance from the face is parallel to the mean image plane. Therefore, a probabilistic interpretation of Mahalanobis distances may be used easily.

These methods may fail to adequately represent faces when large variations in illumination facial expressions and other factors occur. Applying kernel-based nonlinear methods does not produce a significant improvement compared to linear methods. LLE, LLP, and LBP brought simple and effective ways to describe neighboring changes in face description. Subspace approaches were applied in OCV- and SVM-based methods. Preserving the local structure between samples is the domain of NPP and ONPP methods. The problem is that it is still unclear how to select the neighborhood size or assign optimal values for them.

# METHODOLOGY

Multi-feature evaluation on facial data using convolutional neural networks (CNNs) is a machine learning approach that uses CNNs to analyze and extract features from facial data. This can be used for a variety of tasks, such as facial recognition, emotion detection, and gender classification and detection of different facial attributes.

**Data collection:** A dataset of facial images is collected and labeled with the relevant features that need to be analyzed. For this, we will use the pretrained models for giving better accuracy and such models are already preprocessed ensuring that they are in the proper format for analysis. This may involve resizing the images, applying filters to remove noise, and performing other image processing tasks.

**CNN model development:** A CNN model is developed and trained to analyze the facial data. This typically involves defining the architecture of the CNN, including the number and size of the convolutional and pooling layers, as well as the type of activation functions and optimizers used.

**Model loading:** The CNN model is loaded with the pretrained models on facial data using an appropriate training algorithm. This typically involves iteratively adjusting the weights and biases of the model based on the errors made during the training process.

**Model evaluation:** The performance of the trained CNN model is then evaluated on a separate dataset to ensure that it is able to accurately classify the features of interest in the facial data. This may involve calculating metrics such as accuracy, precision, and recall.

**Model fine-tuning:** If the performance of the CNN model is not satisfactory, it can be fine-tuned by adjusting its hyperparameters or by adding or removing layers from the model architecture. This process is iterated until the desired performance is achieved.

**Model deployment:** Once the CNN model has been trained and fine-tuned to an acceptable level of performance, it can be deployed for use on real-world facial data.

# PROCEDURE

Initially, all the pre-trained modules have been loaded into the model. Then the camera starts and recognizes the face ahead of it. If the face is detected then first of all the emotions are predicted then the gender of the respective face. After this, it is time for the detection of age using DNN. Later, several facial features like eyebrows, beard, and nose are detected using CNN and along with it the status of eyes and mouth are also mentioned. In addition, the number of eye blinks is also calculated. At the last, all the features are mentioned in the results providing a green line for outlining the features and the perimeter of the face.

# FRAMEWORKS

## 9.1 MEDIAPIPE

Google created the cross-platform Mediapipe library, which offers fantastic, easily available solutions for computer vision applications. As a prebuilt Python package, MediaPipe provides ready-to-use yet configurable Python solutions. It is built once and can be deployed anywhere. PyPI offers the MediaPipe Python package for Windows, macOS, and Linux. It is an open source that is free. Some of the solutions in Mediapipe are Face detection, Face mesh, Iris, Hands, Pose, and Hair segmentation.

### 9.1.1 MediaPipe Face Detection

MediaPipe Face Detection is an incredibly quick face detection tool that supports multiple faces and six landmarks. Because of the detector's super-realtime performance, it can be used in any live viewfinder application where an accurate facial region of interest is needed as an input for other task-specific models, including 3D facial keypoint estimation, facial features or expression classification, and face region segmentation. Previous to using Mediapipe's face detection model, the model must first be initialized. To do this, use the short syntax mp.solution.face detection. Afterward, use the facial detection method with the appropriate inputs.

- **min_detection_confidence:** This argument also takes an integer value, which is in the range [0.0,1.0]. The default value is 0.5, or 50% confidence, meaning that our model must be at least 50% certain that a face is present to detect it; otherwise, it won't do anything.

### 9.1.2 MediaPipe Face Mesh

MediaPipe Face Mesh's solution estimates 468 3D face landmarks on real-time devices. With only a single camera input and no dedicated depth sensor required, it uses machine learning (ML) to infer the 3D facial surface.

Additionally, the solution comes with the Facial Transform module, which fills the gap between usable real-time augmented reality (AR) applications and face landmark estimation. To estimate a face transform within that space, it creates a metric 3D space and uses the positions of the face landmark screens.

## 9.2 TENSORFLOW

TensorFlow is an open-source, Python-compatible toolkit for numerical computation that accelerates and simplifies the implementation of neural networks and machine learning algorithms. TensorFlow calculations are represented by using domain-specific dataflow graphs. The actions that these neural networks carry out on multidimensional data arrays, known as tensors, are where the name TensorFlow originates. By combining a variety of deep learning and machine learning models and techniques (also known as neural networks), TensorFlow makes them useful. Its flexible architecture enables simple computation deployment across a wide range of platforms, including CPUs, GPUs, and desktops.

Google developed and maintains it, and it was made available under the Apache 2.0 open-source license. Despite having access to the underlying C++ API, the API is ostensibly for the Python programming language. Text-based apps, image identification, voice search, and many more technologies all utilize TensorFlow.

- **from TensorFlow.Keras.preprocessing.image import img_to_array**
  Utilities for image preprocessing and instance to an Augmentation

PIL is the Python Imaging Library which provides the python interpreter with image editing capabilities. Converts a PIL image instance.

- **from tensorflow.keras.models import load_model**
  Loads a model saved via model.save()
- **from tensorflow.keras.models import model_from_json**
  Parses a JSON model configuration string and returns a model instance.

## 9.3 OPENCV

Open-Source Computer Vision Library is how OpenCV is formally referred to. In applications that involve artificial intelligence (AI) and deep learning (DL), the term computer vision (CV) is frequently used and heard. Giving a machine the ability to perceive the world as we do is the essence of the phrase. It was created for computer vision applications and to speed up the use of machine learning and artificial intelligence in consumer products. Python, C++, Java, and many other programming languages are supported by OpenCV. More than 2500 optimized algorithms are available in the collection, including a wide range of both traditional and cutting-edge computer vision and machine learning techniques. These algorithms can be used to find similar images in a database, remove red eyes from flash-taken photos, track eye movements, recognize scenery, and create overlay markers. It can recognize faces, objects, and even human handwriting in images and videos.

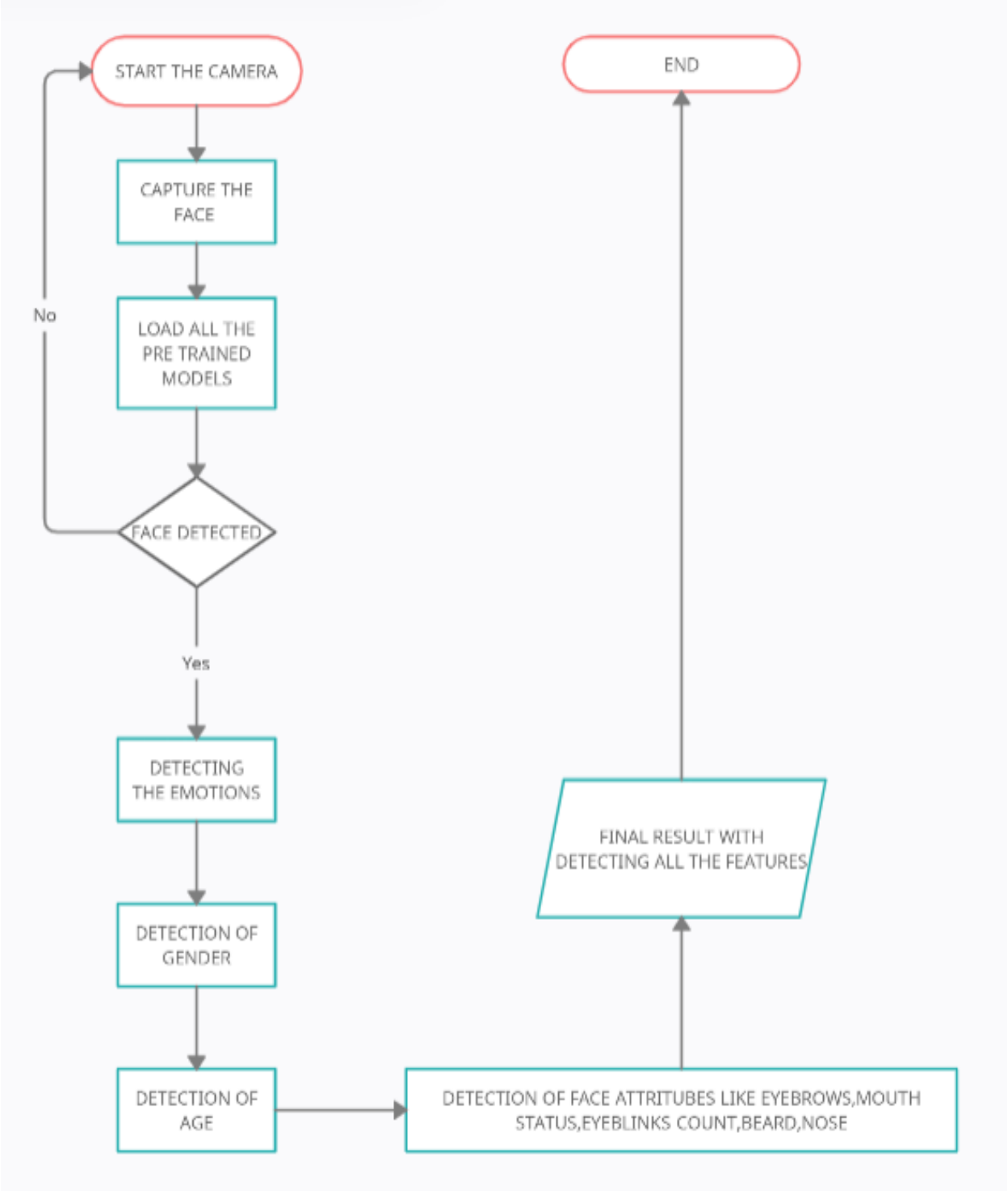
- **Cv2.polylines:** polylines() method is used to draw a polygon on any image.

## 9.4 NUMPY

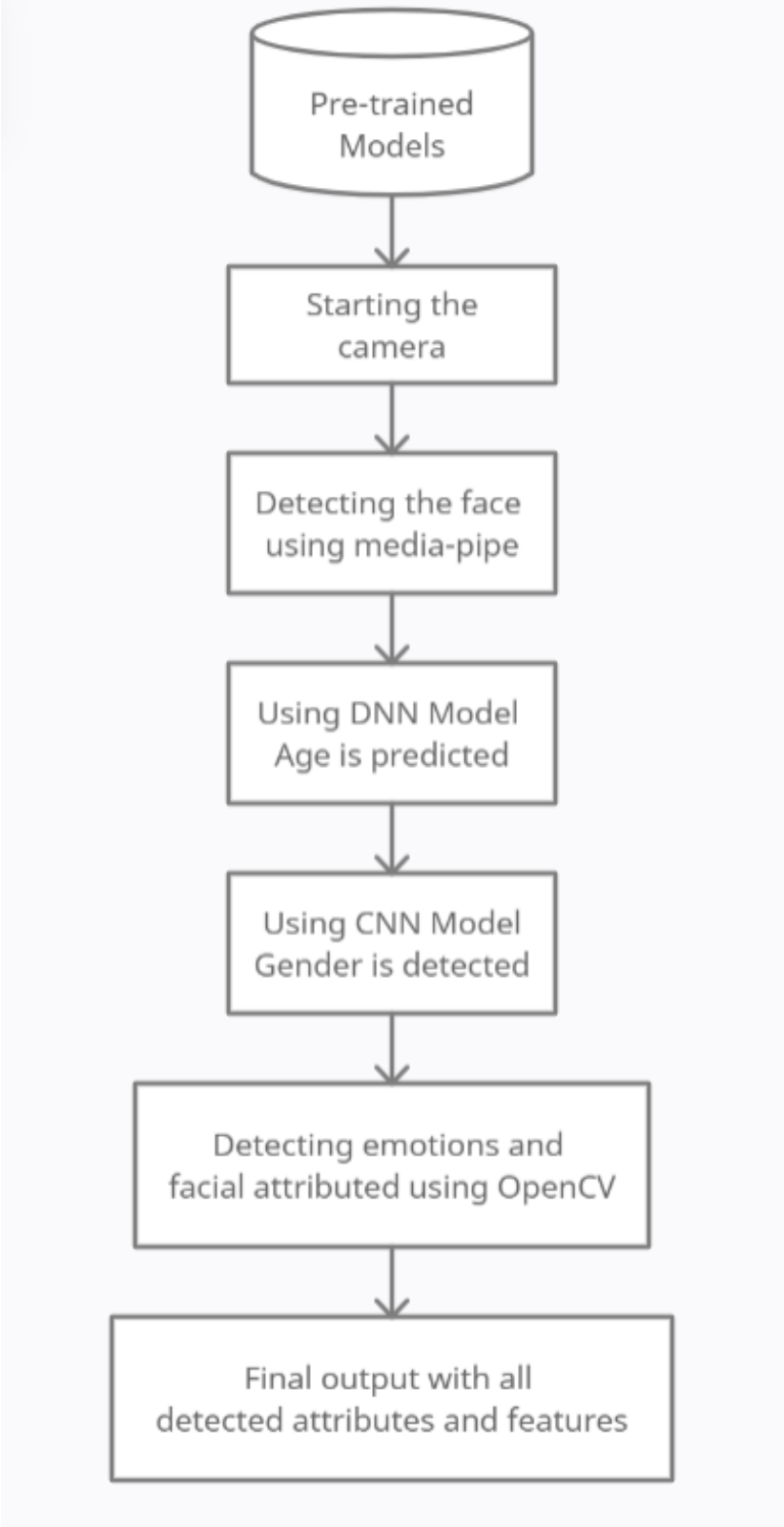
Large, multi-dimensional arrays and matrices are supported by NumPy, a library for the Python programming language, along with a substantial number of high-level mathematical operations that may be performed on these arrays. Numpy is one of the most fundamental Python modules for scientific computing in Python. It is a Python library that offers a multidimensional array object, various derived objects, and a variety of routines for quick operations on arrays, including discrete Fourier, transforms, basic linear algebra, basic statistical operations, shape manipulation, sorting, selecting, I/O, random simulation, and much more. Jim Hugunin originally developed Numeric, the predecessor to NumPy, with assistance from many other programmers. Travis Oliphant developed NumPy in 2005 by heavily altering Numeric to incorporate the capabilities of the rival Numarray.

# DIAGRAMS

## 10.1 ACTIVITY DIAGRAM



## 10.2 BLOCK DIAGRAM



# CODE

```
import mediapipe as mp
import time
import math
import numpy as np


from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
from tensorflow.keras.models import model_from_json

import cv2


close_eye_count =0
Blink_counts =0


Close_frames = 1
FONTS = cv2.FONT_HERSHEY_SIMPLEX
TEXT_COLOR = (0,250, 0)

face_outline=[ 10, 338, 297, 332, 284, 251, 389, 356, 454, 323, 361,
288, 397, 365, 379, 378, 400, 377, 152, 148, 176, 149, 150, 136, 172,
58, 132, 93, 234, 127, 162, 21, 54, 103,67, 109]

LIPS=[ 61, 146, 91, 181, 84, 17, 314, 405, 321, 375,291, 308, 324,
318, 402, 317, 14, 87, 178, 88, 95,185, 40, 39, 37,0 ,267 ,269 ,270
,409, 415, 310, 311, 312, 13, 82, 81, 42, 183, 78 ]

LEFT_EYE =[ 362, 382, 381, 380, 374, 373, 390, 249, 263, 466, 388,
387, 386, 385,384, 398 ]

RIGHT_EYE=[ 33, 7, 163, 144, 145, 153, 154, 155, 133, 173, 157, 158,
159, 160, 161 , 246 ]

LEFT_EYEBROW = [336, 296, 334, 293, 300, 276, 283, 282, 295, 285]

RIGHT_EYEBROW = [70, 63, 105, 66, 107, 55, 65, 52, 53, 46]
```

```
NOSE = [8,240,460]

face_detection =
mp.solutions.face_detection.FaceDetection(model_selection=0,
min_detection_confidence=0.5)
face_mesh = mp.solutions.face_mesh.FaceMesh(min_detection_confidence
=0.5, min_tracking_confidence=0.5)


def detect_eye_mouth_status(face_img, raw_img):
    global close_eye_count, Blink_counts

    rgb_frame = cv2.cvtColor(face_img, cv2.COLOR_BGR2RGB)
    results  = face_mesh.process(rgb_frame)

    if results.multi_face_landmarks:
        img_h, img_w= face_img.shape[:2]
        mesh_coords = [(int(p.x * img_w), int(p.y * img_h)) for p in
results.multi_face_landmarks[0].landmark]
        reRatio, leRatio, mRatio = Open_Close_Ratios(face_img,
mesh_coords, RIGHT_EYE, LEFT_EYE)
        ratio = round((reRatio+ leRatio)/2, 2)

        print("MOUTH RATIO:", mRatio)
        print("Eye RATIO:", ratio)

        eye_threshold = 3.5

        if mRatio > 1.8:
            cv2.putText(raw_img, 'Mouth Closed', (10, 30), FONTS, 1,
TEXT_COLOR, 1)
        else:
            cv2.putText(raw_img, 'Mouth Open', (10, 30), FONTS, 1,
TEXT_COLOR, 1)

        if ratio > eye_threshold:
            cv2.putText(raw_img, 'Eyes Closed', (10, 70), FONTS, 1,
TEXT_COLOR, 1)
        else:
```

```
            cv2.putText(raw_img, 'Eyes Open', (10, 70), FONTS, 1,
TEXT_COLOR, 1)

        if ratio > eye_threshold:
            close_eye_count +=1
        else:
            if close_eye_count>Close_frames:
                Blink_counts +=1
                close_eye_count =0

        cv2.putText(raw_img, f'Total Blinks: {Blink_counts}', (10,
110), FONTS, 1, TEXT_COLOR, 1)



        cv2.polylines(face_img,  [np.array([mesh_coords[p] for p in
LEFT_EYE ], dtype=np.int32)], True, (0,255,0), 1, cv2.LINE_AA)

        cv2.polylines(face_img,  [np.array([mesh_coords[p] for p in
RIGHT_EYE ], dtype=np.int32)], True, (0,255,0), 1, cv2.LINE_AA)

        cv2.polylines(face_img,  [np.array([mesh_coords[p] for p in
LIPS ], dtype=np.int32)], True, (0,255,0), 1, cv2.LINE_AA)

        cv2.polylines(face_img,  [np.array([mesh_coords[p] for p in
face_outline ], dtype=np.int32)], True, (0,255,0), 1, cv2.LINE_AA)

        cv2.polylines(face_img,  [np.array([mesh_coords[p] for p in
LEFT_EYEBROW ], dtype=np.int32)], True, (0,255,0), 1, cv2.LINE_AA)

        cv2.polylines(face_img,  [np.array([mesh_coords[p] for p in
RIGHT_EYEBROW ], dtype=np.int32)], True, (0,255,0), 1, cv2.LINE_AA)

        cv2.polylines(face_img,  [np.array([mesh_coords[p] for p in
NOSE ], dtype=np.int32)], True, (0,255,0), 1, cv2.LINE_AA)

    return face_img, raw_img

def Open_Close_Ratios(img, landmarks, right_indices, left_indices):
```

```
    rh_right = landmarks[246]
    rh_left = landmarks[133]
    rv_top = landmarks[160]
    rv_bottom = landmarks[145]


    lh_right = landmarks[362]
    lh_left = landmarks[387]
    lv_top = landmarks[386]
    lv_bottom = landmarks[374]

    rhDistance = math.dist(rh_right, rh_left)
    rvDistance = math.dist(rv_top, rv_bottom)

    lvDistance = math.dist(lv_top, lv_bottom)
    lhDistance = math.dist(lh_right, lh_left)

    try:
        reRatio = rhDistance/rvDistance
        leRatio = lhDistance/lvDistance
    except:
        reRatio = 0
        leRatio = 0


    mouth_right = landmarks[409]
    mouth_left = landmarks[185]
    mouth_top = landmarks[0]
    mouth_bottom = landmarks[17]

    mhDistance = math.dist(mouth_right, mouth_left)
    mvDistance = math.dist(mouth_top, mouth_bottom)

    try:
        mRatio = mhDistance/mvDistance
    except:
        mRatio = 10

    return reRatio,  leRatio, mRatio
```

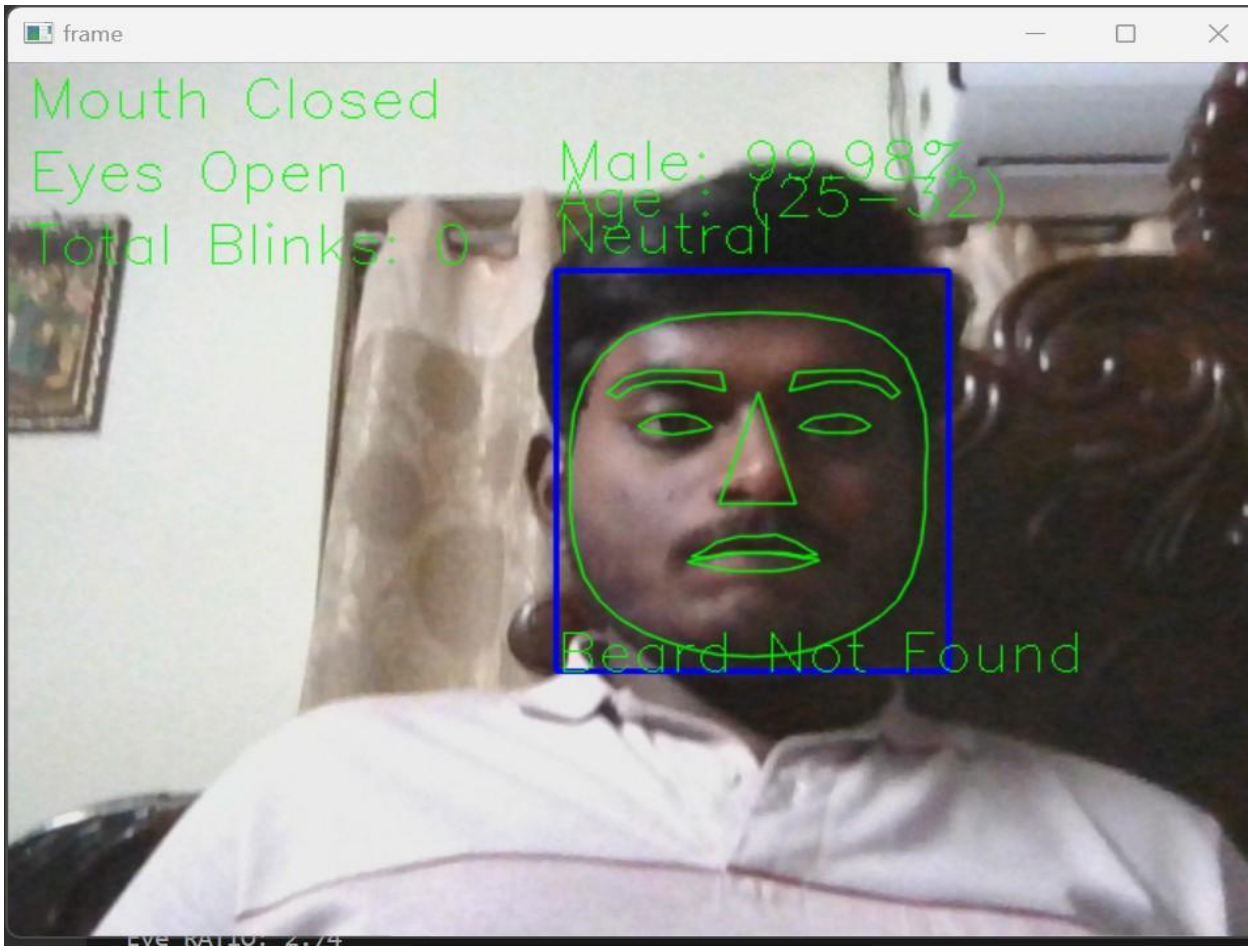
```
classes = ['Male','Female']
model = load_model(r"E:\CAPSTONE\MULTI FEATURE
EVALUATION\models\gender_detection.model")


EMOTIONS_LIST = ["Angry", "Disgust", "Fear", "Happy", "Sad",
"Surprise", "Neutral"]

emotion_json = r"E:\CAPSTONE\MULTI FEATURE
EVALUATION\models\face_emotion_model.json"
emotion_weight = r"E:\CAPSTONE\MULTI FEATURE
EVALUATION\models\face_emotion_model.h5"
with open(emotion_json, "r") as json_file:
    loaded_model_json = json_file.read()
    emotion_model = model_from_json(loaded_model_json)

emotion_model.load_weights(emotion_weight)
print("Emotion Model loaded")



ageList = ['(0-2)', '(4-6)', '(8-12)', '(15-20)', '(25-32)',
'(38-43)', '(48-53)', '(60-100)']

ageProto = r"E:\CAPSTONE\MULTI FEATURE
EVALUATION\models\age_deploy.prototxt"
ageModel = r"E:\CAPSTONE\MULTI FEATURE
EVALUATION\models\age_net.caffemodel"

MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)

ageNet = cv2.dnn.readNet(ageModel,ageProto)

cam = cv2.VideoCapture(0)

while True:
    _, raw_img = cam.read()
    if _:
        start_time = time.time()
        print("Raw Image:", raw_img.shape)
```

```
        x,y,w,h = 0,0,0,0
        img_w = raw_img.shape[1]
        img_h = raw_img.shape[0]

        face_detection_results =
face_detection.process(raw_img[:,:,::-1])

        if face_detection_results.detections:
            for face in face_detection_results.detections:

                print(f'FACE CONFIDENCE: {round(face.score[0], 2)}')
                if face.score[0] < 0.8:
                    continue

                face_data = face.location_data

                x,y,w,h =
int(img_w*face_data.relative_bounding_box.xmin), \

int(img_h*face_data.relative_bounding_box.ymin), \

int(img_w*face_data.relative_bounding_box.width), \

int(img_h*face_data.relative_bounding_box.height)
                break


        if x+y+w+h > 0:
            print("Detected Face Points:", x,y,w,h)
            x = x - 10
            y = y - 40
            w = w + 20
            h = h + 40

            if x<0:
                x = 0
            if y<0:
                y = 0

            face_img = raw_img[y:y+h, x:x+w]
```

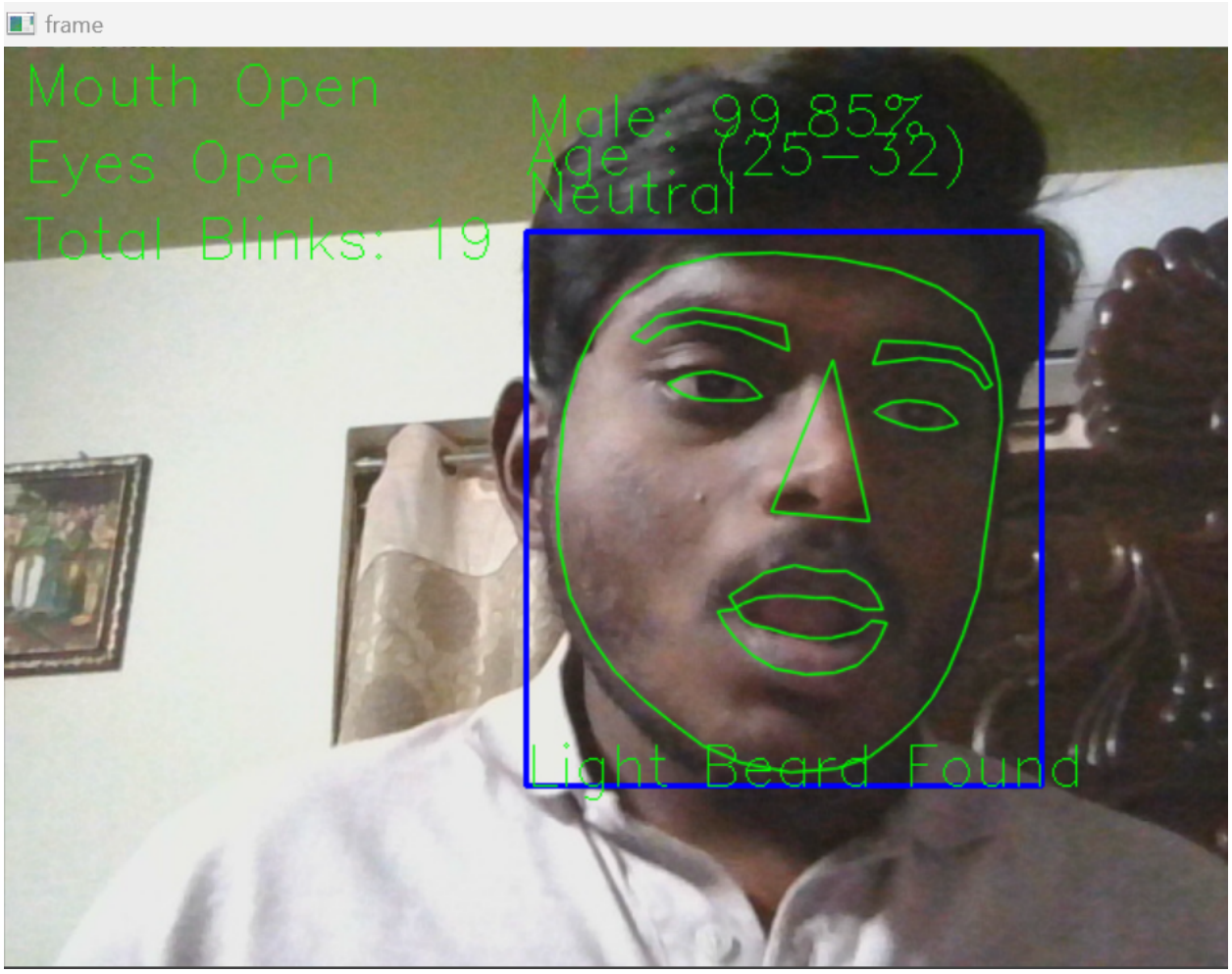
```
            print("Face Image:", face_img.shape)
            cv2.rectangle(raw_img, (x,y), (x+w, y+h), (255,0,0), 2)

            face_crop = cv2.resize(face_img, (96,96))
            face_crop = face_crop.astype("float") / 255.0
            face_crop = img_to_array(face_crop)
            face_crop = np.expand_dims(face_crop, axis=0)

            conf = model.predict(face_crop)[0]
            idx = np.argmax(conf)
            label = "{}: {:.2f}%".format(classes[idx], conf[idx] *
100)
            cv2.putText(raw_img, label, (x, y-50), FONTS, 1,
TEXT_COLOR, 1)


            face_img_gray = cv2.cvtColor(face_img, cv2.COLOR_BGR2GRAY)
            face_resized = cv2.resize(face_img_gray, (48, 48))
            preds = emotion_model.predict(face_resized[np.newaxis, :,
:, np.newaxis])
            pred = EMOTIONS_LIST[np.argmax(preds)]
            cv2.putText(raw_img, pred, (x, y-10), FONTS, 1,
TEXT_COLOR, 1)


            face_processed, raw_img =
detect_eye_mouth_status(face_img, raw_img)
            raw_img[y:y+h, x:x+w] = face_processed


            blob = cv2.dnn.blobFromImage(face_img, 1.0, (227, 227),
MODEL_MEAN_VALUES, swapRB=False)
            ageNet.setInput(blob)
            agePreds = ageNet.forward()
            age = ageList[agePreds[0].argmax()]

            print("Age : {}, conf = {:.3f}".format(age,
agePreds[0].max()))
            cv2.putText(raw_img, "Age : {}".format(age), (x, y-30),
FONTS, 1, TEXT_COLOR, 1)
```

```
            mask = np.zeros_like(raw_img)

            mask = cv2.ellipse(mask, (int((x+(w/2))),
int(y+(h*0.7))),(w//3, w//3), 0, 0, 180, (255,255,255),thickness=-1)

            mask = cv2.cvtColor(mask, cv2.COLOR_BGR2RGB)

            only_beard_part = np.bitwise_and(raw_img, mask)
            hsv_img = cv2.cvtColor(only_beard_part, cv2.COLOR_BGR2HSV)


            low_black = np.array([94, 80, 2])
            high_black = np.array([126, 255, 255])
            MASK = cv2.inRange(hsv_img, low_black, high_black)


            print(cv2.countNonZero(MASK))
            if cv2.countNonZero(MASK) < 110:
                print("Beard Not Found")
                cv2.putText(raw_img, "Beard Not Found", (x, y+h),
FONTS, 1, TEXT_COLOR, 1)
            elif cv2.countNonZero(MASK) < 150:
                print("Light Beard Found")
                cv2.putText(raw_img, "Light Beard Found", (x, y+h),
FONTS, 1, TEXT_COLOR, 1)
            else:
                print("Beard Found")
                cv2.putText(raw_img, "Beard Found", (x, y+h), FONTS,
1, TEXT_COLOR, 1)

        cv2.imshow('frame', raw_img)
        cv2.waitKey(1)


        time_taken = time.time()-start_time
        fps = 1/time_taken
        print("FPS:", fps)
```

# OUTPUT




```
MOUTH RATIO: 3.053545141476458
Eye RATIO: 3.05
Age : (25-32), conf = 0.999
15
Beard Not Found
FPS: 3.770923557573442
Raw Image: (480, 640, 3)
FACE CONFIDENCE: 0.96
Detected Face Points: 250 229 158 158
Face Image: (198, 178, 3)
1/1 [==============================] - 0s 49ms/step
1/1 [==============================] - 0s 44ms/step
```




```
MOUTH RATIO: 1.6817991187600467
Eye RATIO: 2.69
Age : (25-32), conf = 0.980
115
Light Beard Found
FPS: 7.815716016025342
Raw Image: (480, 640, 3)
FACE CONFIDENCE: 0.91
Detected Face Points: 262 142 217 217
Face Image: (257, 237, 3)
1/1 [==============================] - 0s 29ms/step
1/1 [==============================] - 0s 26ms/step
```

# CONCLUSION

In this project, the recognition of emotion, gender, and age has been attempted using a real-time video stream. The proposed method of using CNN's general architecture was successfully implemented. With this attempt, we arrived at a prediction of age within the range of 0-100 years, trained on a significantly large dataset. The gender is predicted to be male or female. Other facial attributes, such as beard detection (Beard found/ light beard found/ No beard found), mouth status detection (open/close), and eye status detection(open/close), are used to identify the emotion of the respective face. Along with all these features and predictions, the eye blink count is also calculated. The proposed method was completed successfully and with greater accuracy. The only difficulty is caused by the brightness of the light. Extremely low background lighting resulted in poor image quality. And this eventually led to an increase in the count of errors in the pre-processing of the image and resulted in comparatively wrong predictions.

# REFERENCES

[1] Emad E.Abdallah, Jamil R.Alzghoul, MuathAlzghool (2020): *" Age and Gender prediction in Open Domain Text"* International Journal of Business Analytics- 2021.

[2] Darshan Gera, S.Balasubramanian (2020*): "Landmark guidance independent spatio-channel attention and complementary context information based facial expression recognition"* Pattern Recognition Letters-2022.

[3] MaryamAhmady, Seyed SaeidMirkamali, BaharehPahlevanzadeh, ElnazPashaei, Ali Asghar RahmaniHosseinabadi, AdamSlowik (2022): *"Facial expression recognition using fuzzified Pseudo Zernike Moments and structural features"* .

[4] JieFang, YuanYuan, XiaoqiangLu, Yachuang Feng (2019): *"Muti-stage learning for gender and age prediction"*.

[5] https://google.github.io/mediapipe/solutions/face_detection.html

[6]https://www.analyticsvidhya.com/blog/2021/07/age-and-gender-detection-using-deep-learning/

[7]https://towardsdatascience.com/real-time-multi-facial-attribute-detection-using-transfer-learning-and-haar-cascades-with-fastai-47ff59e36df0