Age and Gender Prediction using Face Recognition

A project report submitted in partial fulfillment of the Academic requirements for the award of the Degree of

BACHELOR OF ENGINEERING
IN
INFORMATION TECHNOLOGY

By

CH. SAI TEJA	**(2451-18-737-001)**
T.S. MANOGNYA	**(2451-18-737-024)**
T. PREETHIKAKSHARA	**(2451-18-737-031)**

Under the guidance of
Mrs.J.Sowjanya
Assistant Professor, Dept. of I.T
Maturi Venkata Subba Rao (MVSR) Engineering College
Nadergul, Hyderabad.

(September 2021)

DEPARTMENT OF INFORMATION TECHNOLOGY
MATURI VENKATA SUBBA RAO (MVSR) ENGINEERING COLLEGE
(Affiliated to Osmania University, Hyderabad. Recognized by AICTE)
Nadergul, Balapur Mandal, Hyderabad-501510

2020-2021

MATURI VENKATA SUBBA RAO (MVSR) ENGINEERING COLLEGE

(Affiliated to Osmania University, Hyderabad. Recognized by AICTE)
Nadergul, Balapur Mandal, Hyderabad-501510

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

Certified that this a Bonafide Mini Project work carried out by Mr / Ms.____**CHALLA SAI TEJA**_____ Bearing H.T. No. **2451-18-737-001** in the course Mini Project - I (PW 653 IT) prescribed for B.E (3/4) Sem: **VI** by **Osmania University**, in the Department during the academic year 2020 - 2021.

Faculty in-Charge

Guide Signature

Head of the Department

MATURI VENKATA SUBBA RAO (MVSR) ENGINEERING COLLEGE

(Affiliated to Osmania University, Hyderabad. Recognized by AICTE)
Nadergul, Balapur Mandal, Hyderabad-501510

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

Certified that this a Bonafide Mini Project work carried out by Mr / Ms.___**T.S. MANOGNYA**_____ Bearing H.T. No. **2451-18-737-024** in the course Mini Project - I (PW 653 IT) prescribed for B.E (3/4) Sem: **VI** by **Osmania University**, in the Department during the academic year 2020 - 2021.

Faculty in-Charge

Guide Signature

Head of the Department

MATURI VENKATA SUBBA RAO (MVSR) ENGINEERING COLLEGE

(Affiliated to Osmania University, Hyderabad. Recognized by AICTE)
Nadergul, Balapur Mandal, Hyderabad-501510

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

Certified that this a Bonafide Mini Project work carried out by Mr / Ms. **T. PREETHIKAKSHARA** Bearing H.T. No. **2451-18-737-031** in the course Mini Project - I (PW 653 IT) prescribed for B.E (3/4) Sem: **VI** by **Osmania University**, in the Department during the academic year 2020 - 2021.

Faculty in-Charge

Guide Signature

Head of the Department

ACKNOWLEDGEMENT

We, with extreme jubilance and deepest gratitude, would like to thank **Mrs. J. Sowjanya, Assistant Professor,** Department of Information Technology, Maturi Venkata Subba Rao (MVSR) Engineering College, for her constant encouragement and facilities provided to us to complete our project in time.

With immense pleasure, we record our deep sense of gratitude to our beloved Head of the department **Dr. H. Jayasree** Department of Information Technology, MVSR Engineering College, for permitting us to carry out this project.

We would like to extend our gratitude to **Mrs. J.Sowjanya, Assistant Professor, Dr. D. Shanthi, Associate Professor, Mr. D.Muninder, Assistant Professor**, and **S. Kalyana Chakravarthy, Lab Assistant**, Department of Information Technology, Maturi Venkata Subba Rao Engineering College, for his valuable suggestions and timely help during the course of the project.

We express, from the bottom of my heart, my deepest gratitude to my parents and family for the support, dedication, comprehension, and love.

Finally, we express our heartfelt thanks to each and everyone who directly and indirectly helped us in successful completion of this project work.

CHALLA SAI TEJA
(2451-18-737-001)
T.S. MANOGNYA
(2451-18-737-024)
T. PREETHIKAKSHARA
(2451-18-737-031)

TABLE OF CONTENTS

1. ABSTRACT

Due to its vast applications in many facial analysis challenges, automatic age and gender prediction from face photos have received a lot of interest recently. We show in this project that by learning representations using deep-convolutional neural networks (CNN), we were able to significantly improve age and gender identification using the Caffe Model Architecture of Deep Learning FrameWork. We propose a simplified convolutional net design that may be employed even when learning data is scarce. In the recent Showing potential benchmark for age and gender estimation, we show that our method improves current state-of-the-art methods significantly.

This project covers predicting age and gender, as well as face detection and recognition using a trained model. In internal evaluation, the Caffe framework outperforms TensorFlow by 1 to 5 times. Following the training phase, we'll utilize the **.caffemodel** trained model to generate predictions about new data that hasn't been seen before. For the project code, we'll develop a Python script that uses OpenCV. The trained model will recognize the face and accurately predict the age and gender of the person.

2. INTRODUCTION

Age and gender data are critical for a variety of real-world applications, including social understanding, biometrics, identity verification, video surveillance, human-computer interface, electronic customers, crowd behaviour analysis, online advertising, item recommendation, and many others.

Despite its widespread use, estimating age and gender from face photos is a difficult problem to solve, owing to the numerous sources of intra-class variances in people's facial images, which limits the applicability of these models in real-world applications.

Many strategies for solving the classification problem have been developed in recent years. In recent years, lots of new age and gender prediction studies have been offered.

....

3. SYSTEM ANALYSIS

3.1 Objective:

- To build a machine learning model for age and gender prediction, which is more efficient than the previously worked models.

3.2 Problem Specification:

- Automatic prediction of age and gender from face images has drawn a lot of attention recently, due to its wide applications in various facial analysis problems.
- As there are various works proposed for age and gender prediction, those works lack performance, which leads to inefficiency.

3.3 Proposed Methods:

- Image processing
- Face Detection
- Face alignment
- CNN architecture
- Training Details
- Age group classification
- Gender classification

3.4 Applications:

- Age and gender, two of the key facial attributes, play a very foundational role in social interactions, making age and gender estimation from a single face image an important task in intelligent applications, such as
    - Access control
    - Human-Computer Interaction
    - Marketing intelligence and visual surveillance
      etc.

3.5 MODULES AND THEIR FUNCTIONALITIES:

Module-1: Data preparation

In this step, we clean the images and store them in a format that can be used by Caffe. We will write a Python script that will handle both image pre-processing and storage.

Module-2: Model definition

In this step, we choose a CNN architecture and we define its parameters in a configuration file with extension **.prototxt.**

Module-3: Solver definition:

The solver is responsible for model optimization. We define the solver parameters in a configuration file with extension **.prototxt.**

Module-4: Model training

We train the model by executing one Caffe command from the terminal. After training the model, we will get the trained model in a file with the extension **.caffemodel.**

4. REQUIREMENT SPECIFICATIONS

4.1 HARDWARE REQUIREMENTS:

Hardware Tools	Minimum Requirements
Processor	i5 or above
Hard Disk	10GB
RAM	8GB
Monitor	17” colored
Mouse	Optical
Keyboard	122

4.2 SOFTWARE REQUIREMENTS:

Software Tools	Minimum Requirements
Platform	Windows, Linux (or) MacOS
Operating System	Windows, Linux (or) MacOS
Technology	Machine Learning-Python
Scripting Language	Python
IDE	PyCharm (and jupyter)

5. LITERATURE SURVEY

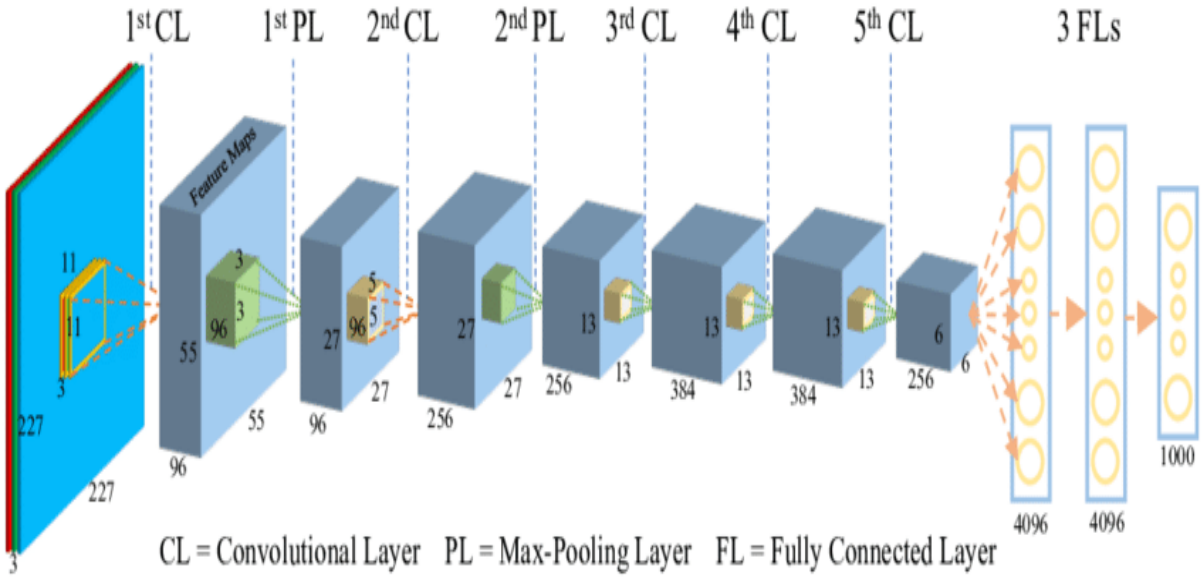
Age and Gender Prediction using Face Recognition is done using an attentional convolutional network, Adience benchmark, and Caffe Model.

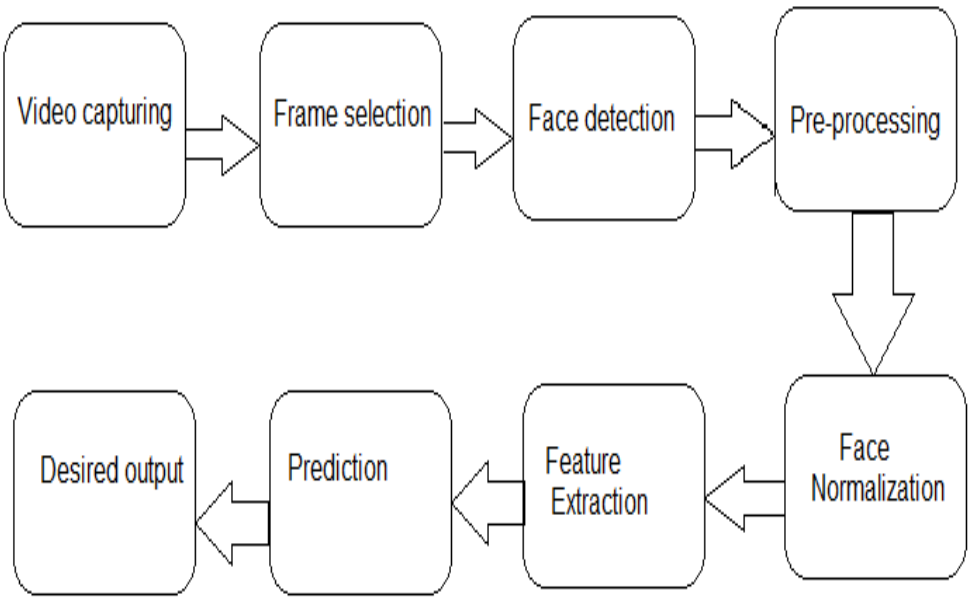
Face detection is widely employed in biometrics as part of (or in conjunction with) a facial recognition structure. Face recognition is also useful for selecting areas of interest in photographs. Marketers are interested in face detection in advance. Any face that walks by can be detected using a webcam that can be connected to a television. The technology then calculates the face's race, gender, and age range. Once the data has been compiled, a sequence of announcements specific to the detected race, gender, and age can be played.

The model above is used to fine-tune networks for three tasks: apparent face detection, age estimation, and gender recognition.

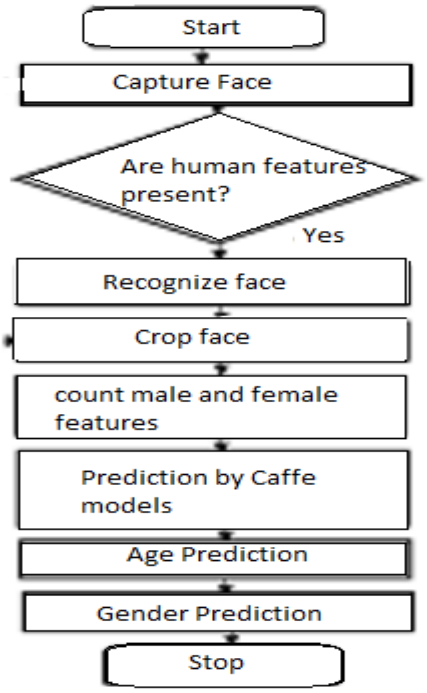
6. DESIGN

6.1 BLOCK DIAGRAM:





6.2 FLOW CHART:



6.3 ALGORITHM

First Layer:

```
class DataAugmentationDoubleLabelsLayer(caffe.Layer):

  def setup(self, bottom, top):
        self._k = 2

  def reshape(self, bottom, top):

     if len(bottom[0].shape) == 4:
        top[0].reshape(self._k*bottom[0].data.shape[0], bottom[0].data.shape[1],
bottom[0].data.shape[2], bottom[0].data.shape[3])
     elif len(bottom[0].shape) == 3:
        top[0].reshape(self._k*bottom[0].data.shape[0], bottom[0].data.shape[1],
bottom[0].data.shape[2])
     elif len(bottom[0].shape) == 2:
        top[0].reshape(self._k*bottom[0].data.shape[0], bottom[0].data.shape[1])
     else:
        top[0].reshape(self._k*bottom[0].data.shape[0])

  def forward(self, bottom, top):

     batch_size = bottom[0].data.shape[0]
     if len(bottom[0].shape) == 4:
```

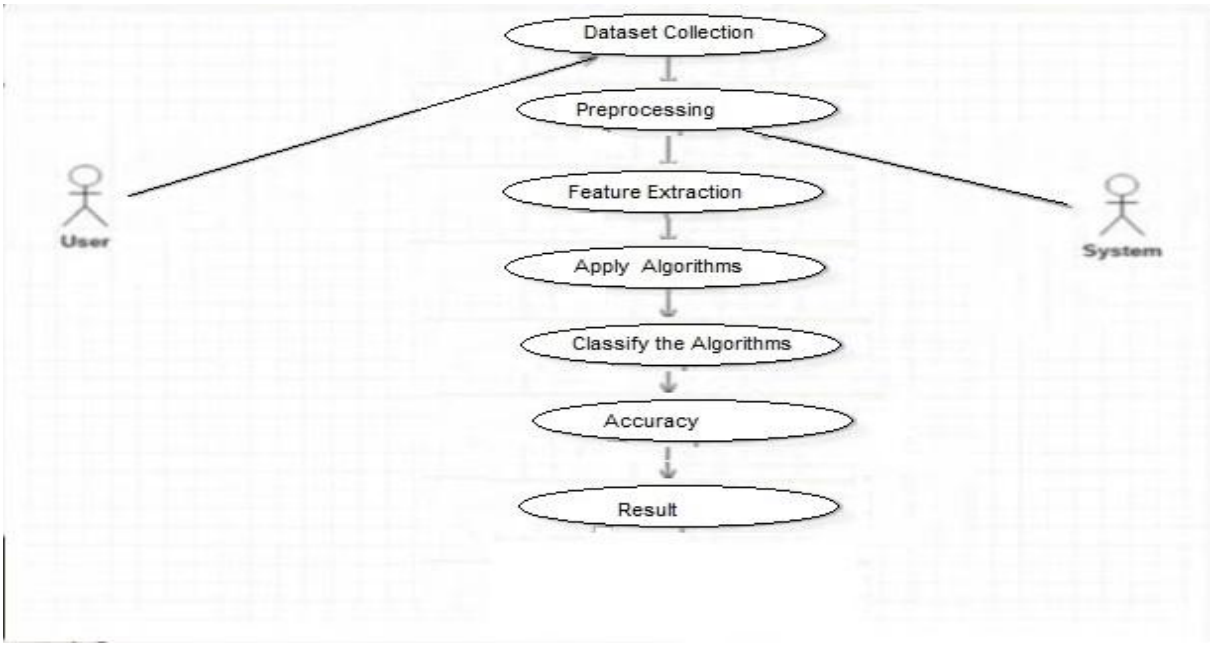
```
            top[0].data[0:batch_size, :, :, :] = bottom[0].data

            for i in range(self._k - 1):
                top[0].data[(i + 1)*batch_size:(i + 2)*batch_size, :, :, :] = bottom[0].data
        elif len(bottom[0].shape) == 3:
            top[0].data[0:batch_size, :, :] = bottom[0].data

            for i in range(self._k - 1):
                top[0].data[(i + 1)*batch_size:(i + 2)*batch_size, :, :] = bottom[0].data
        elif len(bottom[0].shape) == 2:
            top[0].data[0:batch_size, :] = bottom[0].data

            for i in range(self._k - 1):
                top[0].data[(i + 1)*batch_size:(i + 2)*batch_size, :] = bottom[0].data
        else:
            top[0].data[0:batch_size] = bottom[0].data

            for i in range(self._k - 1):
                top[0].data[(i + 1)*batch_size:(i + 2)*batch_size] = bottom[0].data

    def backward(self, top, propagate_down, bottom):

        pass
```

Second Layer

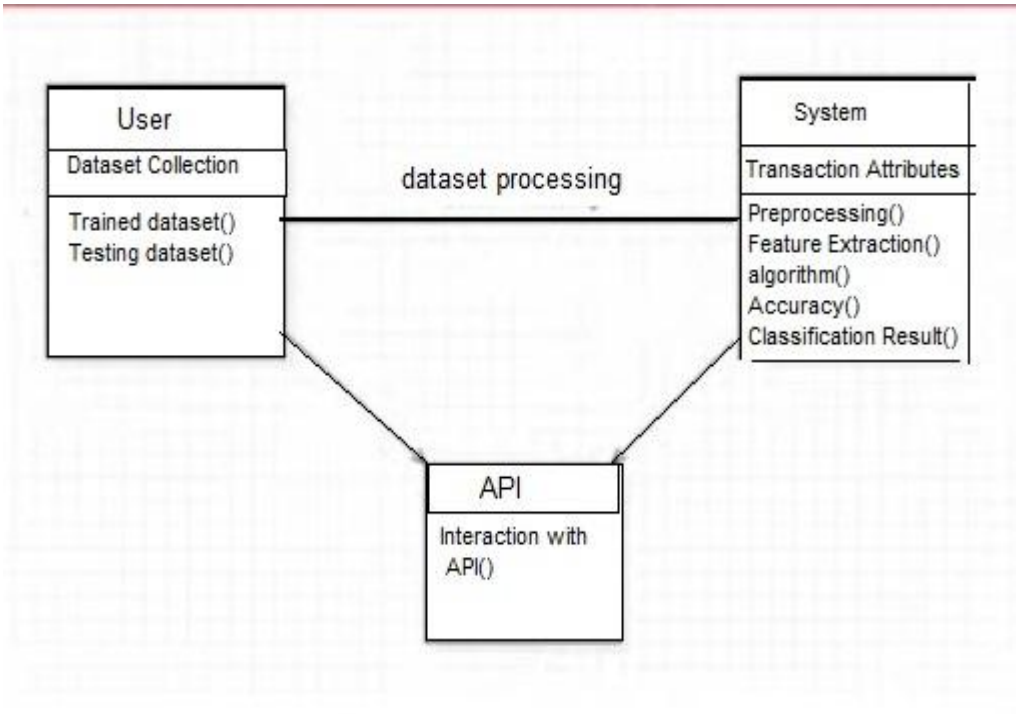
```
class DataAugmentationMultiplicativeGaussianNoiseLayer(caffe.Layer):
    def setup(self, bottom, top):
        pass

    def reshape(self, bottom, top):

        top[0].reshape(2*bottom[0].data.shape[0], bottom[0].data.shape[1],
bottom[0].data.shape[2], bottom[0].data.shape[3])

    def forward(self, bottom, top):

        batch_size = bottom[0].data.shape[0]
        top[0].data[0:batch_size, :, :, :] = bottom[0].data
        top[0].data[batch_size:2*batch_size, :, :, :] =
tools.data_augmentation.multiplicative_gaussian_noise(bottom[0].data)

    def backward(self, top, propagate_down, bottom):

        pass
```

6.4 UML DIAGRAMS

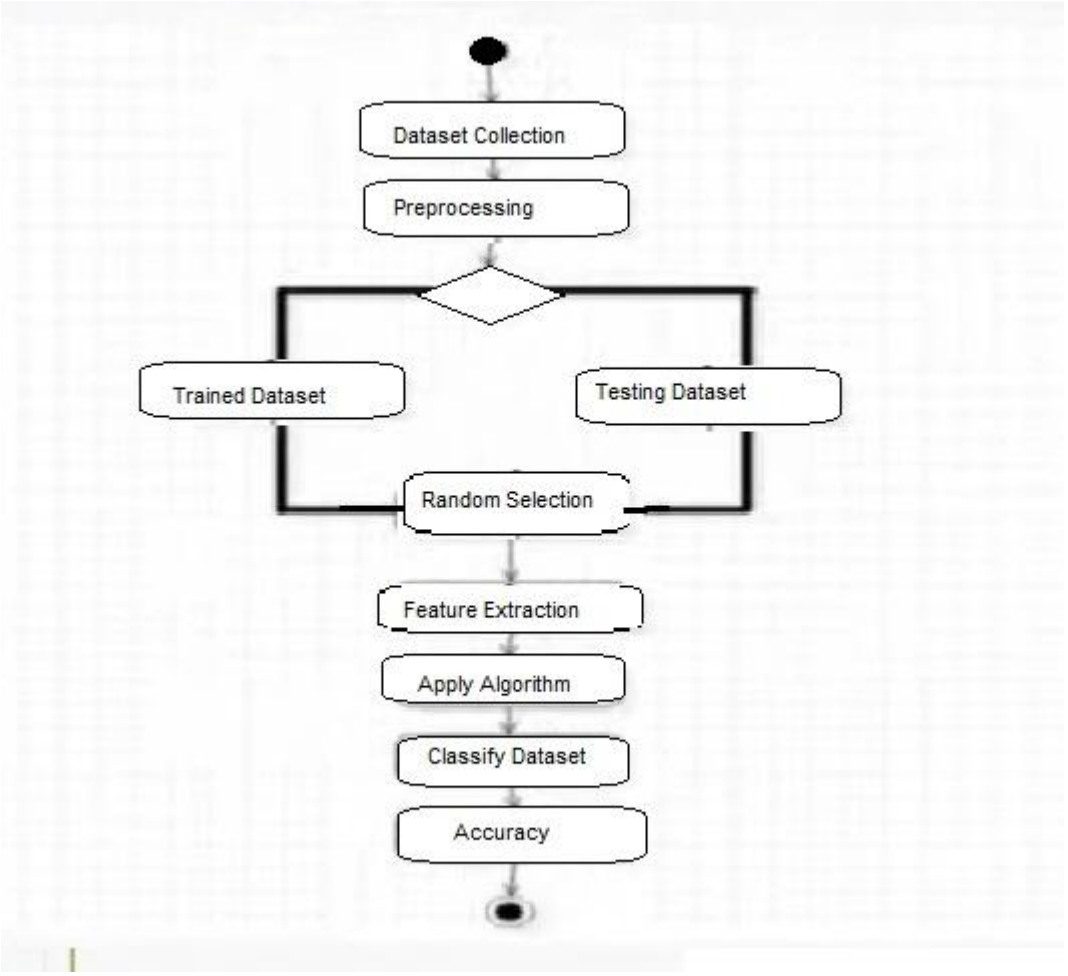
6.4.1 USE CASE DIAGRAM



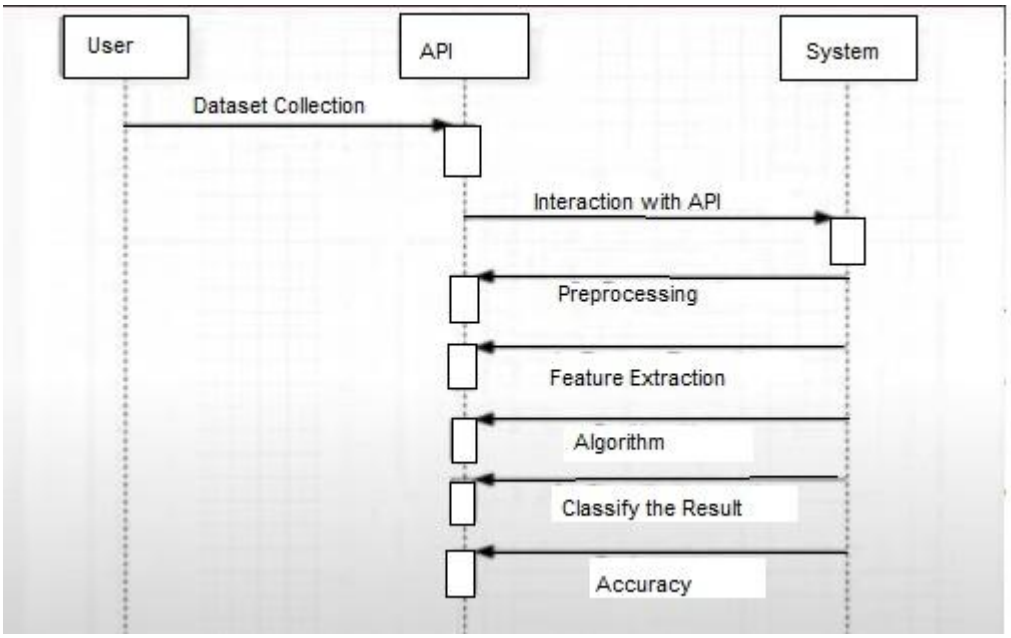
6.4.2 CLASS DIAGRAM



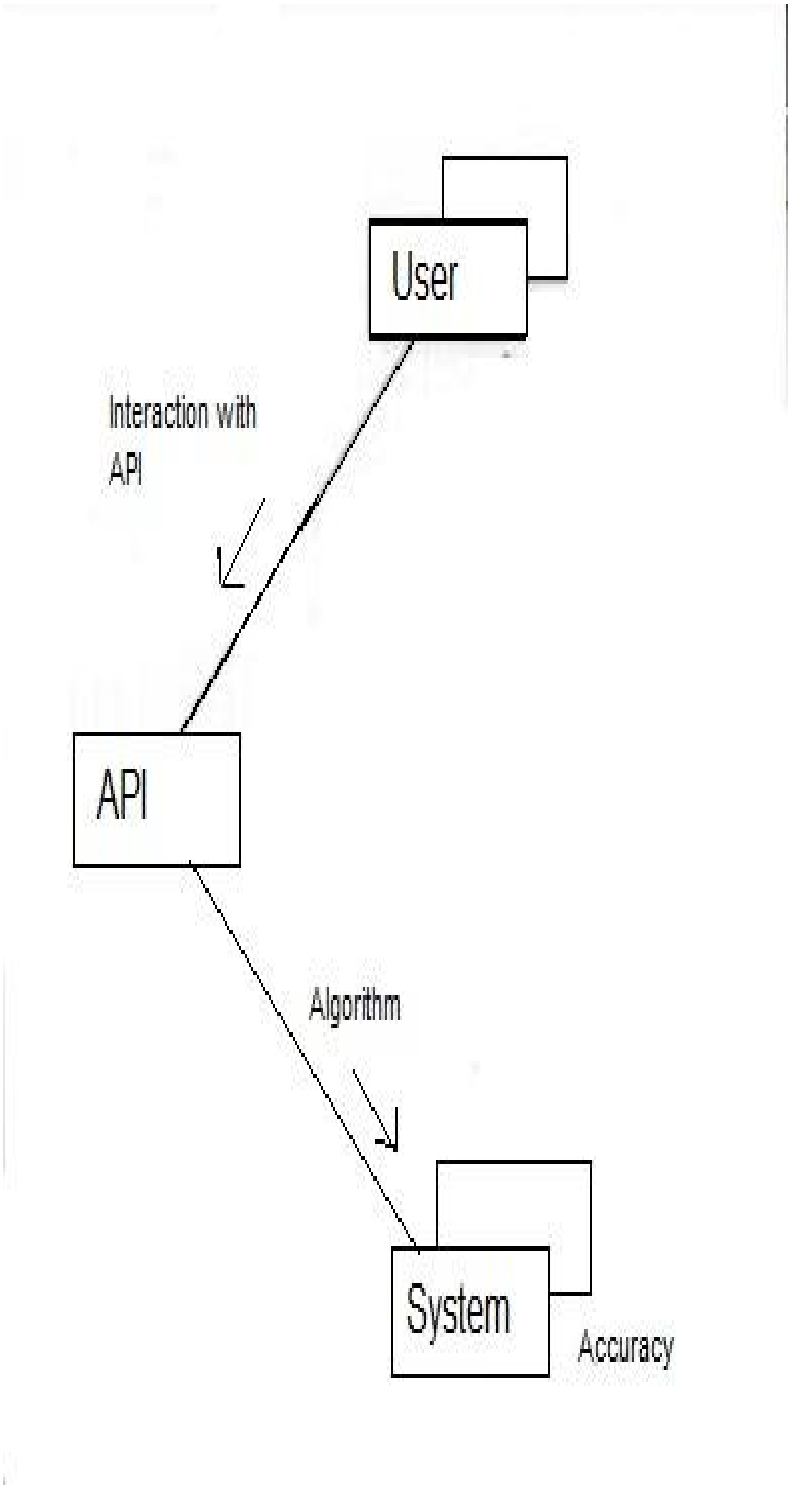
6.4.3 ACTIVITY DIAGRAM



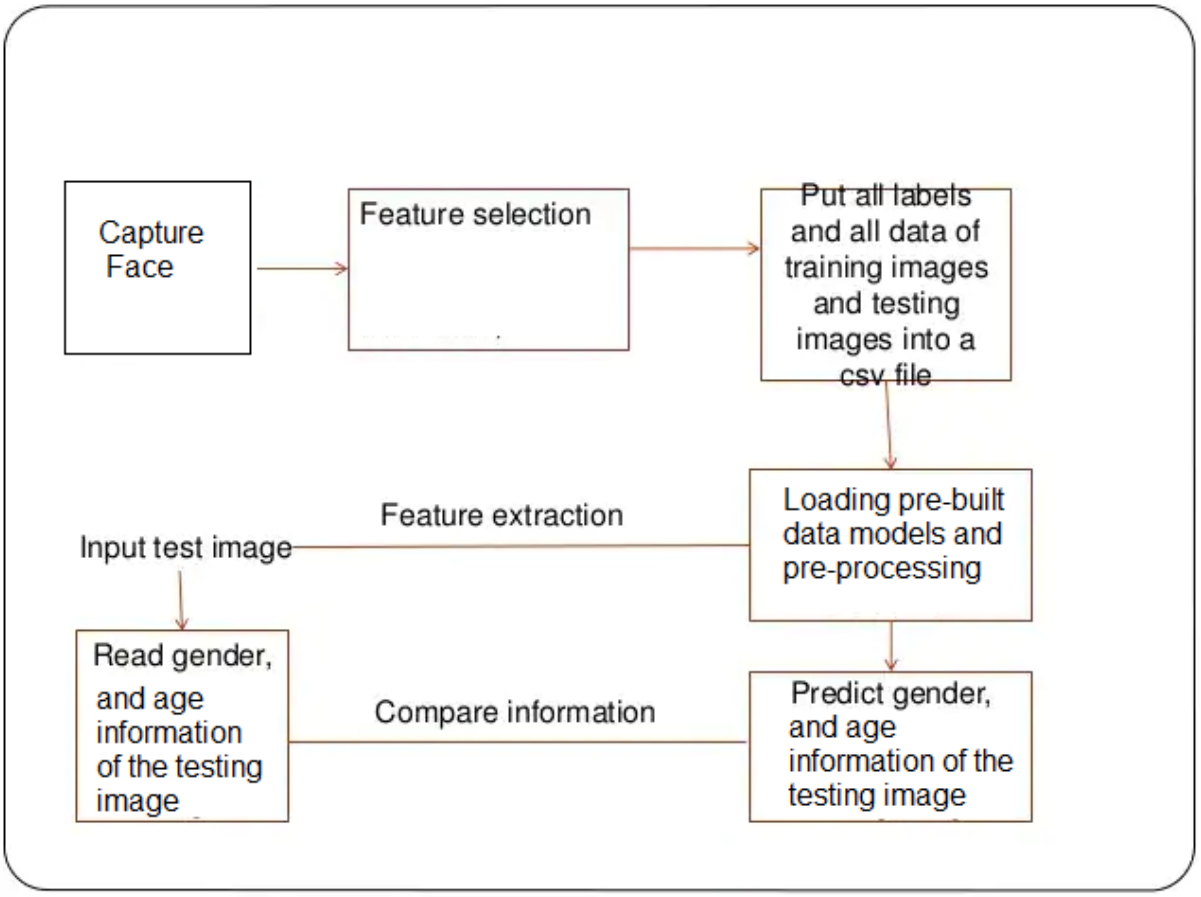
6.4.4 SEQUENCE DIAGRAM



6.4.5 COLLABORATION DIAGRAM



6.5 DATA FLOW DIAGRAM



7. IMPLEMENTATION

age&gender.py:

```
import cv2
import os
def detectFace(net,frame,confidence_threshold=0.7):
   frameOpencvDNN=frame.copy()
   print(frameOpencvDNN.shape)
   frameHeight=frameOpencvDNN.shape[0]
   frameWidth=frameOpencvDNN.shape[1]
   blob=cv2.dnn.blobFromImage(frameOpencvDNN,1.0,(300, 300), [104, 117,
123],swapRB=True,crop=False)
   net.setInput(blob)
   detections=net.forward()
   faceBoxes=[]
   for i in range(detections.shape[2]):
      confidence=detections[0,0,i,2]
      if confidence>confidence_threshold:
         x1=int(detections[0,0,i,3]*frameWidth)
         y1=int(detections[0,0,i,4]*frameHeight)
         x2=int(detections[0,0,i,5]*frameWidth)
         y2=int(detections[0,0,i,6]*frameHeight)
         faceBoxes.append([x1,y1,x2,y2])

cv2.rectangle(frameOpencvDNN,(x1,y1),(x2,y2),(0,255,0),int(round(frameHeight/150)),8)
   return frameOpencvDNN,faceBoxes


faceProto='opencv_face_detector.pbtxt'
faceModel='opencv_face_detector_uint8.pb'
ageProto='age_deploy.prototxt'
ageModel='age_net.caffemodel'
genderProto='gender_deploy.prototxt'
genderModel='gender_net.caffemodel'

genderList=['Male','Female']
ageList=['(0-4)','(6-10)','(12-16)','(18-20)','(21-30)','(33-43)','(45-53)','(60-100)']

faceNet=cv2.dnn.readNet(faceModel,faceProto)
ageNet=cv2.dnn.readNet(ageModel,ageProto)
genderNet=cv2.dnn.readNet(genderModel,genderProto)

video=cv2.VideoCapture(0)
padding=20
while cv2.waitKey(1)<0:
   hasFrame,frame=video.read()
   if not hasFrame:
      cv2.waitKey()
      break
```

```
    resultImg,faceBoxes=detectFace(faceNet,frame)

    if not faceBoxes:
        print("No face detected")

    for faceBox in faceBoxes:

face=frame[max(0,faceBox[1]-padding):min(faceBox[3]+padding,frame.shape[0]-1),max(0,f
aceBox[0]-padding):min(faceBox[2]+padding, frame.shape[1]-1)]
        blob=cv2.dnn.blobFromImage(face,1.0,(227,227),[78.4263377603, 87.7689143744,
114.895847746],swapRB=True,crop=False)
        genderNet.setInput(blob)
        genderPreds=genderNet.forward()
        gender=genderList[genderPreds[0].argmax()]

        ageNet.setInput(blob)
        agePreds=ageNet.forward()
        age=ageList[agePreds[0].argmax()]

cv2.putText(resultImg,f'{gender},{age}',(faceBox[0],faceBox[1]-10),cv2.FONT_HERSHEY
_SIMPLEX,0.8,(0,255,255),2,cv2.LINE_AA)
        cv2.imshow("Detecting age and Gender",resultImg)


        if cv2.waitKey(33) & 0xFF == ord('q'):
            break

cv2.destroyAllWindows()
```

age_deploy.prototxt:

```
name: "CaffeNet"
input: "data"
input_dim: 1
input_dim: 3
input_dim: 227
input_dim: 227
layers {
  name: "conv1"
  type: CONVOLUTION
  bottom: "data"
  top: "conv1"
  convolution_param {
    num_output: 96
    kernel_size: 7
    stride: 4
  }
}
layers {
  name: "relu1"
```

```
  type: RELU
  bottom: "conv1"
  top: "conv1"
}
layers {
  name: "pool1"
  type: POOLING
  bottom: "conv1"
  top: "pool1"
  pooling_param {
    pool: MAX
    kernel_size: 3
    stride: 2
  }
}
layers {
  name: "norm1"
  type: LRN
  bottom: "pool1"
  top: "norm1"
  lrn_param {
    local_size: 5
    alpha: 0.0001
    beta: 0.75
  }
}
layers {
  name: "conv2"
  type: CONVOLUTION
  bottom: "norm1"
  top: "conv2"
  convolution_param {
    num_output: 256
    pad: 2
    kernel_size: 5
  }
}
layers {
  name: "relu2"
  type: RELU
  bottom: "conv2"
  top: "conv2"
}
layers {
  name: "pool2"
  type: POOLING
  bottom: "conv2"
  top: "pool2"
  pooling_param {
    pool: MAX
```

```
    kernel_size: 3
    stride: 2
  }
}
layers {
  name: "norm2"
  type: LRN
  bottom: "pool2"
  top: "norm2"
  lrn_param {
    local_size: 5
    alpha: 0.0001
    beta: 0.75
  }
}
layers {
  name: "conv3"
  type: CONVOLUTION
  bottom: "norm2"
  top: "conv3"
  convolution_param {
    num_output: 384
    pad: 1
    kernel_size: 3
  }
}
layers{
  name: "relu3"
  type: RELU
  bottom: "conv3"
  top: "conv3"
}
layers {
  name: "pool5"
  type: POOLING
  bottom: "conv3"
  top: "pool5"
  pooling_param {
    pool: MAX
    kernel_size: 3
    stride: 2
  }
}
layers {
  name: "fc6"
  type: INNER_PRODUCT
  bottom: "pool5"
  top: "fc6"
  inner_product_param {
    num_output: 512
```

```
  }
}
layers {
  name: "relu6"
  type: RELU
  bottom: "fc6"
  top: "fc6"
}
layers {
  name: "drop6"
  type: DROPOUT
  bottom: "fc6"
  top: "fc6"
  dropout_param {
    dropout_ratio: 0.5
  }
}
layers {
  name: "fc7"
  type: INNER_PRODUCT
  bottom: "fc6"
  top: "fc7"
  inner_product_param {
    num_output: 512
  }
}
layers {
  name: "relu7"
  type: RELU
  bottom: "fc7"
  top: "fc7"
}
layers {
  name: "drop7"
  type: DROPOUT
  bottom: "fc7"
  top: "fc7"
  dropout_param {
    dropout_ratio: 0.5
  }
}
layers {
  name: "fc8"
  type: INNER_PRODUCT
  bottom: "fc7"
  top: "fc8"
  inner_product_param {
    num_output: 8
  }
}
```

```
layers {
  name: "prob"
  type: SOFTMAX
  bottom: "fc8"
  top: "prob"
}
```

gender_deploy.prototxt:

```
name: "CaffeNet"
input: "data"
input_dim: 10
input_dim: 3
input_dim: 227
input_dim: 227
layers {
  name: "conv1"
  type: CONVOLUTION
  bottom: "data"
  top: "conv1"
  convolution_param {
    num_output: 96
    kernel_size: 7
    stride: 4
  }
}
layers {
  name: "relu1"
  type: RELU
  bottom: "conv1"
  top: "conv1"
}
layers {
  name: "pool1"
  type: POOLING
  bottom: "conv1"
  top: "pool1"
  pooling_param {
    pool: MAX
    kernel_size: 3
    stride: 2
  }
}
layers {
  name: "norm1"
  type: LRN
  bottom: "pool1"
  top: "norm1"
  lrn_param {
    local_size: 5
```

```
    alpha: 0.0001
    beta: 0.75
  }
}
layers {
  name: "conv2"
  type: CONVOLUTION
  bottom: "norm1"
  top: "conv2"
  convolution_param {
    num_output: 256
    pad: 2
    kernel_size: 5
  }
}
layers {
  name: "relu2"
  type: RELU
  bottom: "conv2"
  top: "conv2"
}
layers {
  name: "pool2"
  type: POOLING
  bottom: "conv2"
  top: "pool2"
  pooling_param {
    pool: MAX
    kernel_size: 3
    stride: 2
  }
}
layers {
  name: "norm2"
  type: LRN
  bottom: "pool2"
  top: "norm2"
  lrn_param {
    local_size: 5
    alpha: 0.0001
    beta: 0.75
  }
}
layers {
  name: "conv3"
  type: CONVOLUTION
  bottom: "norm2"
  top: "conv3"
  convolution_param {
    num_output: 384
```

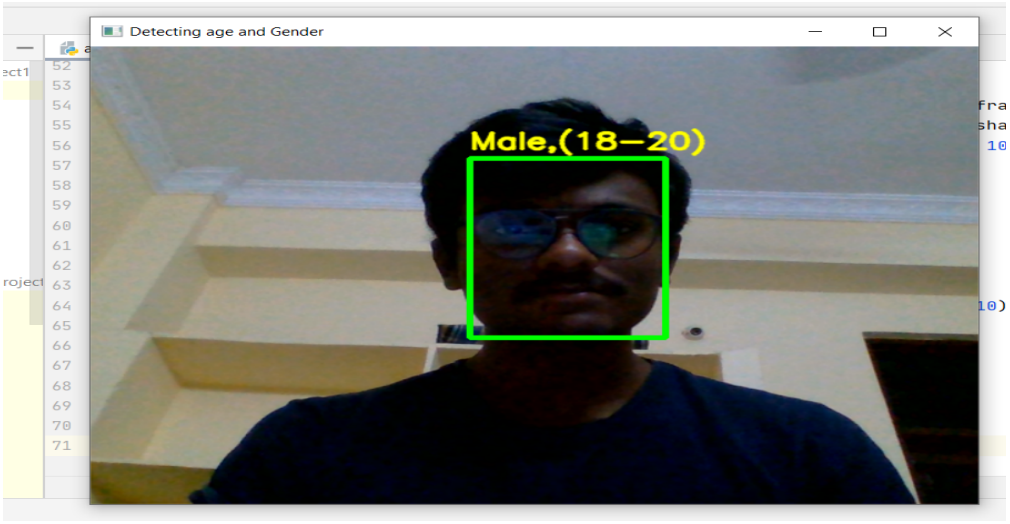
```
    pad: 1
    kernel_size: 3
  }
}
layers{
  name: "relu3"
  type: RELU
  bottom: "conv3"
  top: "conv3"
}
layers {
  name: "pool5"
  type: POOLING
  bottom: "conv3"
  top: "pool5"
  pooling_param {
    pool: MAX
    kernel_size: 3
    stride: 2
  }
}
layers {
  name: "fc6"
  type: INNER_PRODUCT
  bottom: "pool5"
  top: "fc6"
  inner_product_param {
    num_output: 512
  }
}
layers {
  name: "relu6"
  type: RELU
  bottom: "fc6"
  top: "fc6"
}
layers {
  name: "drop6"
  type: DROPOUT
  bottom: "fc6"
  top: "fc6"
  dropout_param {
    dropout_ratio: 0.5
  }
}
layers {
  name: "fc7"
  type: INNER_PRODUCT
  bottom: "fc6"
  top: "fc7"
```

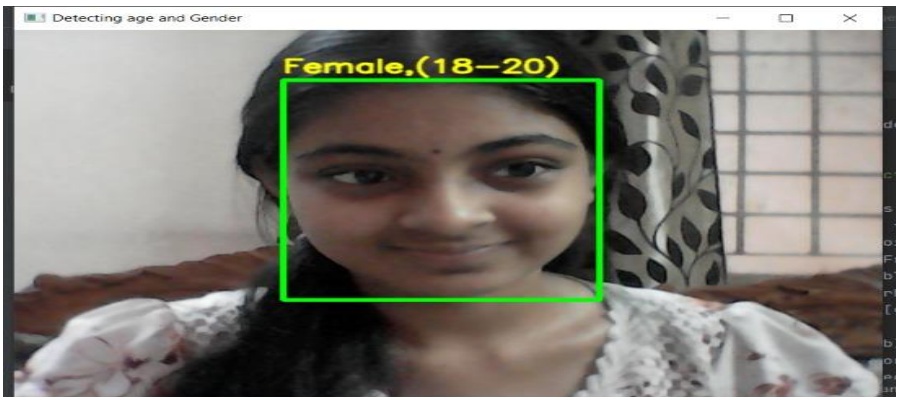
```
  inner_product_param {
    num_output: 512
  }
}
layers {
  name: "relu7"
  type: RELU
  bottom: "fc7"
  top: "fc7"
}
layers {
  name: "drop7"
  type: DROPOUT
  bottom: "fc7"
  top: "fc7"
  dropout_param {
    dropout_ratio: 0.5
  }
}
layers {
  name: "fc8"
  type: INNER_PRODUCT
  bottom: "fc7"
  top: "fc8"
  inner_product_param {
    num_output: 2
  }
}
layers {
  name: "prob"
  type: SOFTMAX
  bottom: "fc8"
  top: "prob"
}
```

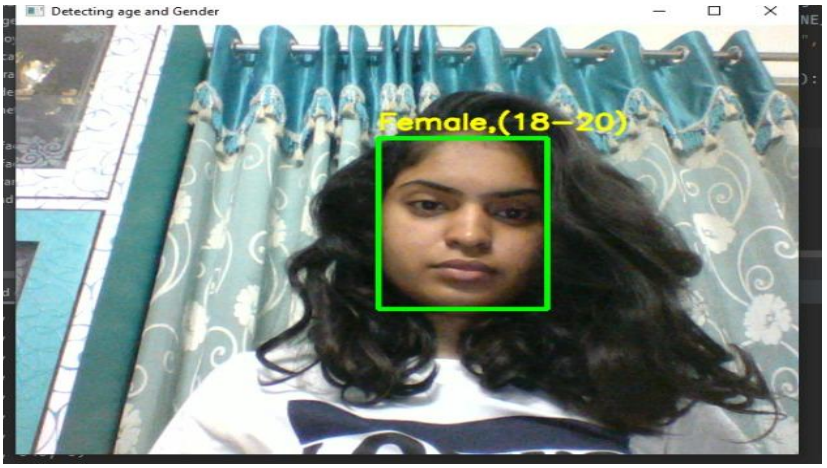
8. TESTING

S.No	Description	Expected Results	Actual Results	Status
1	Face Detection	The face of a person should only be detected	Face has been detected	Success
2	Age Prediction	The age of a person should be predicted after detection of face	Age has been predicted	Success
3	Gender Prediction	The gender of a person should be predicted after detection of face	Gender has been predicted	Success
4	Display of gender and age	Displaying a person's predicted age and gender	Age and Gender has been displayed	Success

9. SCREENSHOTS







10. CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION:

The entire study of gender categorization and age estimation contributions can be used to solve real-time application challenges. Despite the fact that many earlier methods have addressed the difficulties of age and gender categorization, much of this work has concentrated on limited photos acquired in lab settings until recently. Such settings do not sufficiently reflect the appearance variances found in real-world photos on social media platforms and in online archives. Images on the internet, on the other hand, are not only more difficult but also more plentiful. Using Internet data, we investigate how well deep CNN performs on these tasks using an example from the related topic of facial recognition. We present our findings using a lean deep-learning architecture that avoids overfitting due to the scarcity of labeled data. In comparison to some contemporary network architectures, our network is "shallow," lowering the number of parameters and the risk of overfitting. We intentionally add cropped versions of the photographs in our training set to further inflate the size of the training data.

Our findings lead to two crucial conclusions. First, even with the significantly lower size of today's unconstrained image sets labeled for age and gender, CNN may be used to improve age and gender classification results. Second, because our model is so simple, more complex systems with additional training data may potentially be capable of much bettering the findings given here.

FUTURE ENHANCEMENTS:

For future work, we'll look into a more complex CNN architecture and a more reliable image processing approach for estimating exact ages. We can use this project for electronic customers, crowd behavior analysis.

11. REFERENCES

Image Dataset:

https://www.kaggle.com/ttungl/adience-benchmark-gender-and-age-classification

OpenCV:

https://docs.opencv.org/3.4/d6/d0f/group__dnn.html

https://docs.opencv.org/4.5.2/d5/de7/tutorial_dnn_googlenet.html

Literature Survey:

https://towardsdatascience.com/age-detection-using-facial-images-traditional-machine-learning-vs-deep-learning-2437b2feeab2

https://arxiv.org/pdf/2010.03791.pdf#:~:text=Age%20and%20gender%20information%20are,item%20recommendation%2C%20and%20many%20more

APPENDICES:

A. LIST OF ABBREVIATIONS

Caffe: Convolutional Architecture for Fast Feature Embedding
dnn: Deep Neural Network

B. LIST OF FIGURES

Block Diagram
Flow Chart
Data Flow Diagram
Screenshots

C. LIST OF TABLES

A table showing the testing performed during different instances of our project.