AGE AND GENDER DETECTION USING DEEP LEARNING

A MAJOR PROJECT REPORT

Submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

Submitted By

SHAGUFTHA TAMKINATH	**21UK1A6731**
VAKITI DEEPAK	**21UK1A6733**
PEDDI VILASINI	**21UK1A6737**
KORE KARTHIK	**21UK1A6762**

Under the guidance of

Mrs. K. SOUMYA

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATASCIENCE)

VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) – 506005

2021-2025

AGE AND GENDER DETECTION USING DEEP LEARNING

A UG PHASE -I PROJECT REPORT

Submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

Submitted By

SHAGUFTHA TAMKINATH	**21UK1A6731**
VAKITI DEEPAK	**21UK1A6733**
PEDDI VILASINI	**21UK1A6737**
KORE KARTHIK	**21UK1A6762**

Under the guidance of

Mrs. K.SOUMYA

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) –506005

2021-2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
(DATA SCIENCE)

VAAGDEVI ENGINEERING COLLEGE (WARANGAL)
BOLLIKUNTA, WARANGAL (T.S)-506005 2021-2025

CERTIFICATE OF COMPLETION

UG PROJECT PHASE -I

This is to certify that the **UG Project Phase-1 entitled "AGE AND GENDER DETECTION USING DEEP LEARNING"** is being submitted by: **SHAGUFTHA TAMKINATH (21UK1A6731), VAKITI.DEEPAK (21UK1A6733), PEDDI.VILASINI (21UK1A6737) KORE.KARTHIK (21UK1A6762)** in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2024- 2025.

Project Guide
Mrs. K.SOUMYA
(Assistant Professor)

Head of the Department
Dr. P. MAHIPAL REDDY
(Professor)

EXTERNAL

DECLARATION

We declare that the work reported in the project **"AGE AND GENDER DETECTION USING DEEP LEARNING"** is a record of work done by us in the partial fulfillment for the award of the degree of Bachelor of Technology in Computer Science And Engineering (Data Science), **VAAGDEVI ENGINEERING COLLEGE** (An Autonomous Institution & Affiliated to JNTU Hyderabad) Accredited by NAAC with 'A+' Grade, Certified by ISO 9001:2015 Approved by AICTE, New Delhi, Bollikunta , Warangal-506 005, Telangana, India under the guidance of **Mrs.K.SOUMYA**, Assistant Professor, CSE Department.

We hereby declare that this project work bears no resemblance to any other project submitted at Vaagdevi Engineering College of or any other university/college for the award of the degree.

SHAGUFTHA TAMKINATH	**21UK1A6731**
VAKITI DEEPAK	**21UK1A6733**
PEDDI VILASINI	**21UK1A6737**
KORE KARTHIK	**21UK1A6762**

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. SYED MUSTHAK AHAMED**, Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this **UG PROJECT PHASE -I** in the institute.

We extend our heartfelt thanks to **Dr. P. MAHIPAL REDDY**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the **UG PROJECT PHASE -I.**

We express heartfelt thanks to the coordinator **Mr. E.MAHESH**, Assistant professor, Department of CSE for her constant support and giving necessary guidance for completion of this **UG PROJECT PHASE -I**.

We express heartfelt thanks to the Guide **Mrs. K. SOUMYA,** Assistant professor, Department of CSE for her constant support and giving necessary guidance for completion of this **UG PROJECT PHASE - I.**

Finally, we express our sincere thanks and gratitude to our family members, friends for their encouragement and outpouring their knowledge and experience throughout the project.

SHAGUFTHA TAMKINATH	**21UK1A6731**
VAKITI DEEPAK	**21UK1A6733**
PEDDI VILASINI	**21UK1A6737**
KORE KARTHIK	**21UK1A6762**

ABSTRACT

In today's rapidly advancing digital world, the ability to accurately determine human attributes such as **age** and **gender** through facial image analysis has become crucial for many applications, ranging from security and personalized marketing to automated service systems. This project introduces the ***Age and Gender Detection System Using Facial Image Analysis***, which aims to estimate a person's age group and gender by examining subtle patterns and features in their facial images. The system follows a systematic approach that begins with the **collection of diverse facial images**, continues with precise face detection and enhancement to improve image quality, and then analyzes key facial characteristics to classify individuals based on their age and gender. Leveraging a broad dataset that includes various **lighting conditions**, facial expressions, and ethnic backgrounds, the system is trained to provide robust and reliable predictions. The main goal is to develop a **real-time**, **user-friendly**, and **non-invasive** tool that can be effectively deployed in environments such as digital verification, automated check-ins, smart retail, and crowd monitoring. By offering a fast, accurate, and efficient profiling method, this system supports the creation of smarter, safer, and more adaptive technological solutions.

TABLE OF CONTENTS

LIST OF FIGURES PAGE NO

1. INTRODUCTION

1.1 OVERVIEW

The human face reveals important information about a person's age and gender, making it a valuable source for various applications such as security, personalized services, and digital interactions. Subtle differences in facial structure, skin texture, and expressions can serve as key indicators for estimating age groups and gender categories. This project aims to develop an intelligent, non-invasive system that utilizes facial images to detect age and gender accurately by leveraging advanced image processing and deep learning techniques.

The system captures facial images and processes them to enhance clarity and extract important visual features. These features are then analyzed by a deep learning model trained on a large and diverse dataset of labeled facial images to classify individuals into age and gender categories. Designed to be fast, reliable, and user-friendly, the system is suitable for real-time use and can be deployed in settings such as automated check-ins, personalized advertising, and crowd monitoring. These features are then analyzed by a deep learning model to classify facial images into different age and gender categories. The solution is designed to be user-friendly and accessible, making it ideal for real-time applications in diverse environments.

By combining facial analysis with deep learning, this system demonstrates how technology can enable accurate age and gender detection, support personalized services, and enhance security across various application.

1.2 PURPOSE

The primary purpose of this project is to develop a reliable, non-invasive, and accessible system that can accurately detect a person's age and gender by analyzing facial images using deep learning techniques. Facial features often carry subtle clues related to age and gender that can be automatically identified. However, these indicators are not always easily determined through manual observation.

This system is intended to:

1. **Enable Accurate Detection** of age groups and gender by identifying distinctive visual features present in facial images.
2. **Support Personalized and Automated Services** by providing a fast and efficient

tool useful in areas such as digital verification**,** retail, and crowd analytics.

3. **Leverage AI for Precision**, using deep learning models to improve classification accuracy and minimize human error.

2. PROBLEM STATEMENT

Understanding age and gender accurately is essential in many digital and real-world applications, yet traditional methods often rely on manual input, documentation, or in-person verification, which can be slow, error-prone, or unavailable in real-time contexts. Human facial features hold valuable visual cues related to both age and gender, but interpreting these cues accurately requires expertise and consistency that is not always feasible—especially in automated systems or resource-constrained environments. In the absence of a fast, reliable, and non-intrusive solution, opportunities for personalization, security, and data-driven decision-making are often missed or delayed.

o *Facial features are visible and non-invasive indicators, but are underutilized in current identity estimation systems.*

While facial images contain rich visual cues about a person's age and gender, they are rarely leveraged to their full potential in everyday systems. Traditional methods often depend on manual input or physical documentation, which are not only time-consuming but also prone to errors or misuse. Automating this process with intelligent systems can ensure faster, more accurate, and consistent predictions in both real-time and large-scale applications.

o *There is a lack of AI-based systems specifically designed to analyze facial images for real-time age and gender detection.*

Therefore, there is a clear need for a reliable, non-invasive, and intelligent system that can automatically process and interpret facial images using deep learning. Such a system would not only enhance personalization and automation in digital services but also improve accessibility and efficiency in environments where manual verification is impractical or unavailable.

o *The goal is to build a deep learning-based facial image analysis system for accurate age and gender detection.*

3. LITERATURE SURVEY

3.1 EXISTING PROBLEM

Despite advancements in computer vision and digital identity systems, accurate detection of a person's age and gender remains a challenge in many real-world scenarios. Traditional methods often depend on physical documents, manual verification, or self-declared information, which can be unreliable, time-consuming, or inaccessible—especially in large-scale applications or remote settings. Although human facial features carry distinct markers related to age and gender, these cues are often underutilized in intelligent automated systems.

- *Lack of automation and real-time solution for facial-based identity estimation limits boarder implementation.*

Manual identification of age and gender through facial features requires human expertise, is prone to bias, and varies between observers. This makes the process inconsistent and unsuitable for high-speed or large-scale environments. In applications where instant and accurate demographic information is needed, manual approaches cause bottlenecks and reduces system effectiveness.

- *Existing systems do not fully utilize AI or deep learning to analyze facial images for accurate age and gender detection.*

3.2 PROPOSED SOLUTION

- **Image-Based-Diagnosis**
  Develop a system that analyzes high-quality facial images to accurately detect and classify age and gender.
- **Deep Learning-Integration**
  Utilize advanced deep learning models for automatic feature extraction and classification based on facial characteristics.
- **Image-Preprocessing**
  Apply face detection, alignment, and normalization techniques to enhance the quality and consistency of the input data.
- **Non-Invasive & User-Friendly**
  Design the system to be fully non-intrusive, easy to operate, and adaptable for use in both real-time applications and resource-limited environments.

4.THEORITIC AL ANALYSIS

4.1. BLOCK DIAGRAM

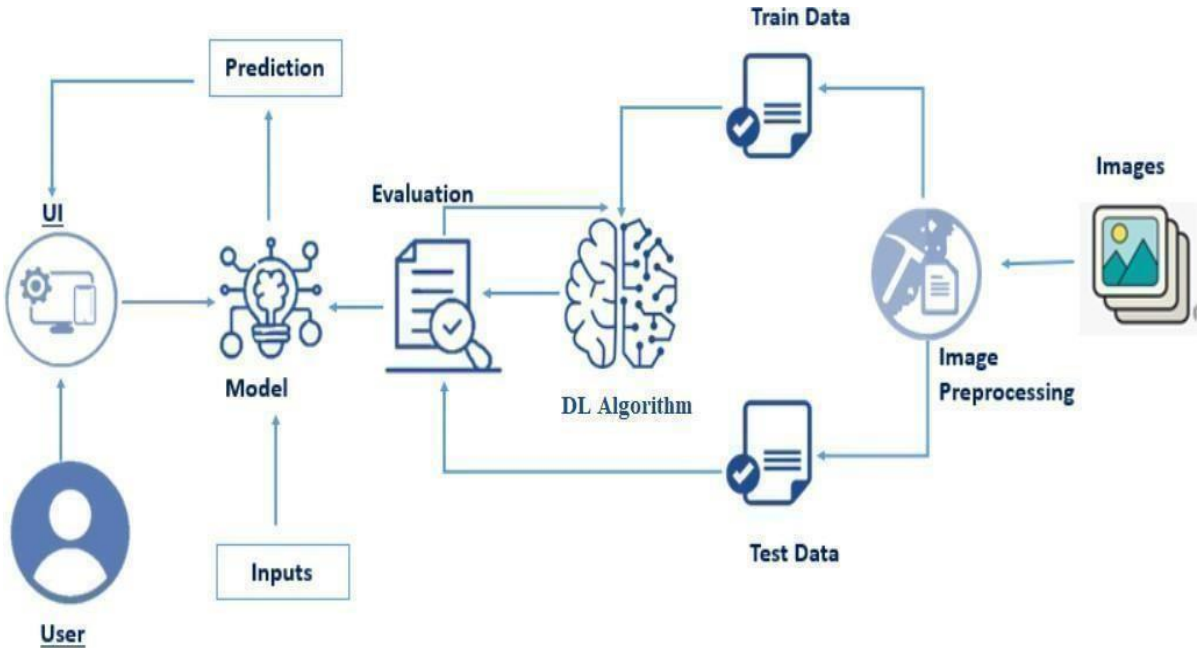


Figure 1: Block diagram

The block diagram represents a system for **"Age and Gender detection Using Deep Learning".**

1. User Interface: User uploads facial images through a simple and accessible interface.
2. Image Preprocessing: Facial images are processed with enhancement, alignment, and normalization techniques.
3. Data Split: The dataset is divided into training, validation, and testing sets for balanced learning.
4. Yolov9: Train the dataset using Yolo version v9.
5. Evaluation: The model's performance is evaluated using accuracy, precision, and other metrics.
6. Prediction: The trained model predicts the age group and gender from new input images.
7. Output: Final results are displayed to the user in a clear and interpretable format.

4.2. SOFTWARE REQUIREMENT SPECIFICATION

Resource Type	Description	Specification/Allocation
Hardware		
Computing Resources	GPU specifications, vs code	NVIDIA V100 GPUs
Memory	RAM specifications	8 GB
Storage	Disk space for data, models, and logs	1 TB SSD
Software		
Frameworks	Python frameworks	Flask
Libraries	Additional libraries	Ultralytics , Yolov9s,
Development Environment	IDE, version control	Google colab
Data		
Data	Roboflow, 2GB, Directories	Roboflow dataset 1000 images

Figure 2: Software requirement specification

The following software and tools were utilized to develop the "Age and Gender Using Deep Learning":

Development Environment

Google Colab

Google Colab serves as the development and execution environment for the age and gender. It provides a cloud-based Jupyter Notebook interface with access to Python libraries and hardware acceleration (GPU/TPU), which is critical for:

Data preprocessing and visualization, Training and fine-tuning deep learning models like ResNet152V2.

Feature Selection and Preprocessing

Data Preprocessing:

Preprocessing ensures high-quality input data for model training. The following steps were implemented:

- **System Design:** Develop a deep learning-based system to analyze facial images for accurate age and gender detection.
- **Dataset Preparation:** Use a dataset of labeled facial images with age groups and gender, applying data augmentation techniques to improve diversity.
- **Model Training and Optimization:** Train and fine-tune a deep learning model to achieve high accuracy in predicting both age and gender.
- **Performance Evaluation**: Assess the system's performance using accuracy, precision, recall, and confusion matrix to ensure reliability across various conditions.

Model Training Tools

- **Roboflow**: Used for organizing and preprocessing facial image datasets, including annotation, augmentation, and format conversion.
- **Deep Learning Model (e.g., AgeNet, GenderNet):**
  Pre-trained models fine-tuned for age and gender classification to ensure efficient and accurate predictions.
- **PyTorch:** A flexible deep learning framework used for training, testing and optimizing model's performance
- **Flask (for UI):** Lightweight web framework used to create a simple interface for users to upload facial images and view predicted age and gender results.

Flask Web Application:

Flask, a Python-based lightweight web framework, was used to create the user interface.

It allows users to upload facial images and displays predicted **age group** and **gender** results.

5. EXPERIMENTAL INVESTIGATIONS

The experimental investigation of the **Age and Gender Detection** system involves designing and conducting a series of tests to evaluate its performance across various facial image scenarios. These investigations assess the model's **accuracy**, **robustness**, and **efficiency** in predicting age and gender in real-time.

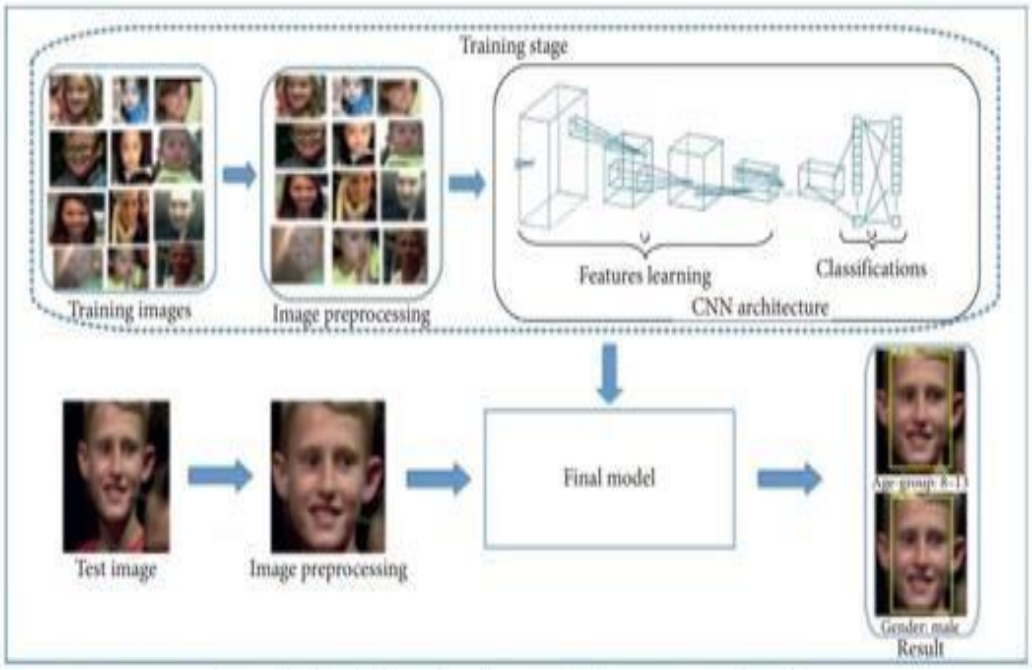


FIGURE 1: The pipeline of our framework for age group and gender.

Figure 3: Output of the project

4. DATAFLOW DIAGRAM

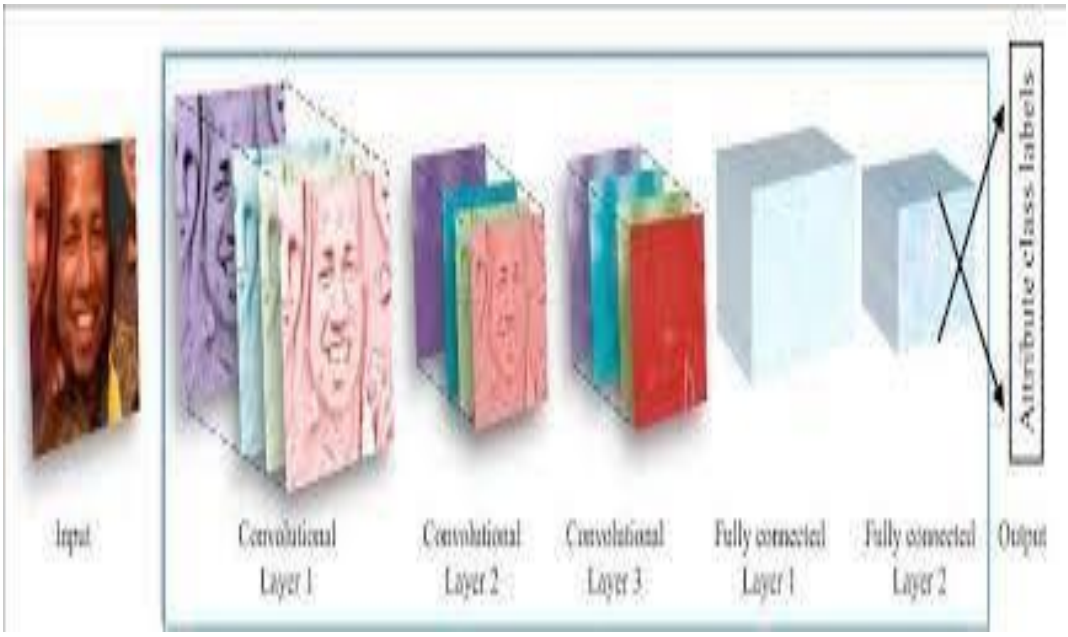
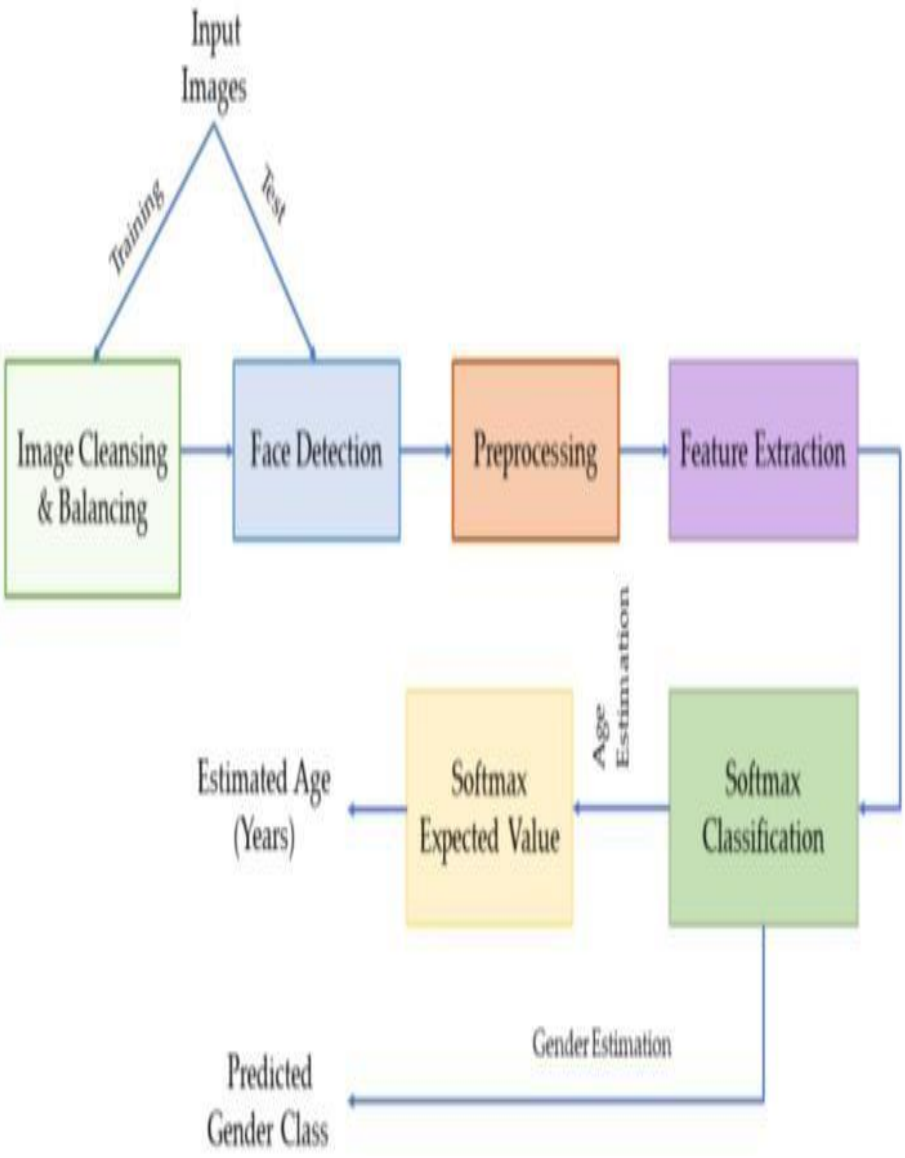


Figure 4: Age and gender detection System Dataflow diagram

Figure 4.1: Flow chart for **Age and Gender Detection Using Deep Learning**



USE CASE DIAGRAM

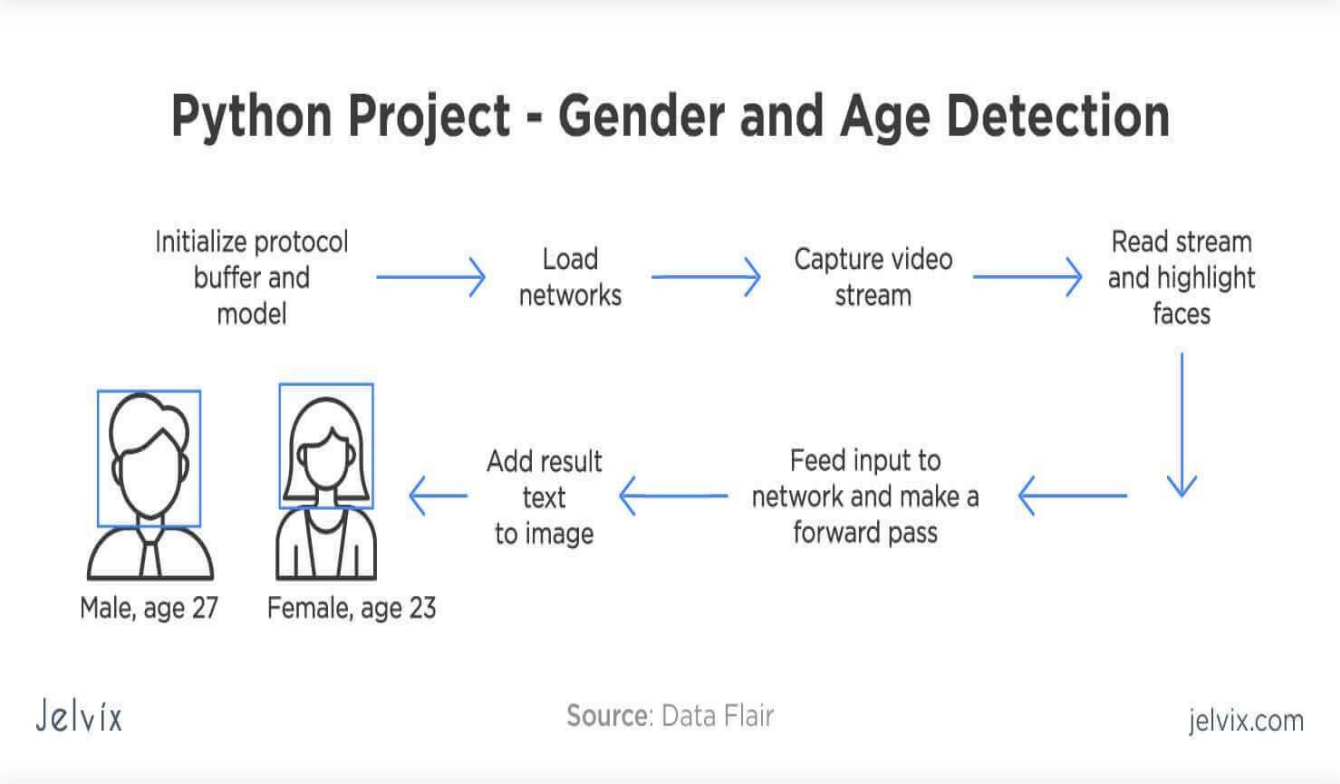


Figure 5: Use case diagram for **Age and Gender Detection Using Deep Learning**

7. FUTURE SCOPE

The block diagram and flowchart will guide the system's development and ensure a structured implementation in Phase-2. The literature survey will continue to inform enhancements to the system by integrating the latest advancements in age and gender recognition techniques. The software design and interface can be further optimized to ensure compatibility with real-time applications and user expectations. In Phase-2 of the project, we will focus on the practical implementation and comprehensive evaluation of the Age and Gender Detection system. Building upon the foundation laid in Phase-1, the next phase will explore several critical areas to ensure the system's effectiveness and usability. Code snippets and full implementation will be included to demonstrate key components such as image preprocessing, model prediction, and user interaction, ultimately resulting in a fully functional and deployable system for accurate age and gender classification in real-world scenarios.

AGE AND GENDER DETECTION USING DEEP LEARNING

A UG PHASE -II PROJECT REPORT

Submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

Submitted By

SHAGUFTHA TAMKINATH	**21UK1A6731**
VAKITI DEEPAK	**21UK1A6733**
PEDDI VILASINI	**21UK1A6737**
KORE KARTHIK	**21UK1A6762**

Under the guidance of

Mrs. K. SOUMYA

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) – 506005

2021-2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

VAAGDEVI ENGINEERING COLLEGE (WARANGAL)

BOLLIKUNTA, WARANGAL (T.S)-506005

2021-2025

CERTIFICATE OF COMPLETION

INDUSTRY ORIENTED UG PHASE -II

This is to certify that the **UG Project Phase-1 entitled "AGE AND GENDER DETECTION USING DEEP LEARNING"** is being submitted by: **SHAGUFTHA TAMKINATH (21UK1A6731), VAKITI DEEPAK (21UK1A6733), PEDDI VILASINI (21UK1A6737),** and **KORE KARTHIK (21UK1A6762)** in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2024-2025,is a record of work carried out by them under the guidance and supervision.

Project Guide

Mrs. K. SOUMYA
(Assistant Professor)

Head of the Department

Dr. P. MAHIPAL REDDY
(Professor)

EXTERNAL

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. SYED MUSTHAK AHAMED**, Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this **UG PROJECT PHASE -II** in the institute.

We extend our heartfelt thanks to **Dr. P. MAHIPAL REDDY**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the **UG PROJECT PHASE -II.**

We express heartfelt thanks to the coordinator **Mr.E.MAHESH**, Assistant professor, Department of CSE for her constant support and giving necessary guidance for completion of this **UG PROJECT PHASE -II**.

We express heartfelt thanks to the Guide **Mrs. K. SOUMYA,** Assistant professor, Department of CSE for her constant support and giving necessary guidance for completion of this **UG PROJECT PHASE -II.**

Finally, we express our sincere thanks and gratitude to our family members, friends for their encouragement and outpouring their knowledge and experience throughout the project.

SHAGUFTHA TAMKINATH	**21UK1A6731**
VAKITI DEEPAK	**21UK1A6733**
PEDDI VILASINI	**21UK1A6737**
KORE KARTHIK	**21UK1A6762**

TABLE OF CONTENTS

LIST OF FIGURES PAGE NO

1. INTRODUCTION

Accurate age and gender detection plays a vital role in improving personalization, user security, and demographic analysis across various platforms. Traditional approaches often involve manual assessments or documentation, which can be time-consuming and inconsistent. The rise of AI and image processing has enabled the development of smart, contactless alternatives. These systems can offer rapid predictions without the need for human intervention. This shift opens new doors for automation in real-world applications.

The human face contains subtle patterns that indicate age and gender, such as facial structure, Skin tone and texture. However, identifying these cues manually is prone to subjectivity and errors. Automatic systems offer a more consistent and scalable solution for analyzing such features. These systems can be especially useful in environments with high user interaction. They reduce human effort while improving overall system intelligence.

This project focuses on building an intelligent system to detect age and gender using facial images. The approach uses digital image analysis to extract key visual features from uploaded images. It then predicts the age group and gender with high accuracy and speed. The system is designed to be non-invasive, easy to use, and adaptable to real-time environments. This makes it ideal for public systems, security, and user experience platforms.

This project demonstrates how deep learning can enhance real-time facial analysis systems. It supports the shift toward automated and intelligent profiling tools across industries. the practical application of deep learning in enhancing facial image recognizing System and sets the foundation for further advancements in intelligent health systems.

UG Project Phase-II involves the complete development and testing of the system designed during Phase-I. It includes the practical implementation, evaluation, and refinement of the model. This phase also covers a detailed analysis of the application's advantages, limitations, and potential areas for future improvement. Final conclusions are drawn based on the system's performance and outcomes. Overall, Phase-II translates theoretical design into a functional and tested solution.

2. CODE SNIPPETS

2.1 PYTHON CODE

Here we will train the dataset in Google colab such that it detects the Age and Gender of humans.

Prepare and load the labeled dataset (images with age and gender annotations) in Google Colab.

Follow the sequence given below to execute the project practically.

- Collection of Data (Images of human faces labeled with age and gender).
- Create test, train, valid sets using roboflow.
- Importing necessary libraries.
- Download the pretrained weights.
- Train and Test model using VGG16.
- Evaluate the model using accuracy and loss graphs.
- Save the trained model
- Build the Application
- Create HTML webpages
- Create app.py file
- Detect Age and Gender of Humans from uploaded images.

IMPORTING THE NECESSARY LIBRARIES

To implement the **Age and Gender Detection** system using deep learning and computer vision, several Python libraries are essential. These libraries provide the necessary functions for image capture, preprocessing, model loading, prediction, and serving results via a web interface. Below is a code snippet showing the libraries used in this project:

```
import pandas as pd
import cv2 # Import cv2 directly instead of as cv
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from tqdm.notebook import tqdm
warnings.filterwarnings('ignore')
%matplotlib inline

import tensorflow as tf
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D, Input
```

```
[12] 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"
```

```
[13] MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
     ageList = ['(0-2)', '(4-6)', '(8-12)', '(15-20)', '(25-32)', '(38-43)', '(48-53)', '(60-100)']
     genderList = ['Male', 'Female']
```

```
[14] import os
```

```
ariables  Terminal                                                    11:21 AM  Python 3
```

Figure 1: Importing necessary Libraries

SPLITTING THE DATASET:

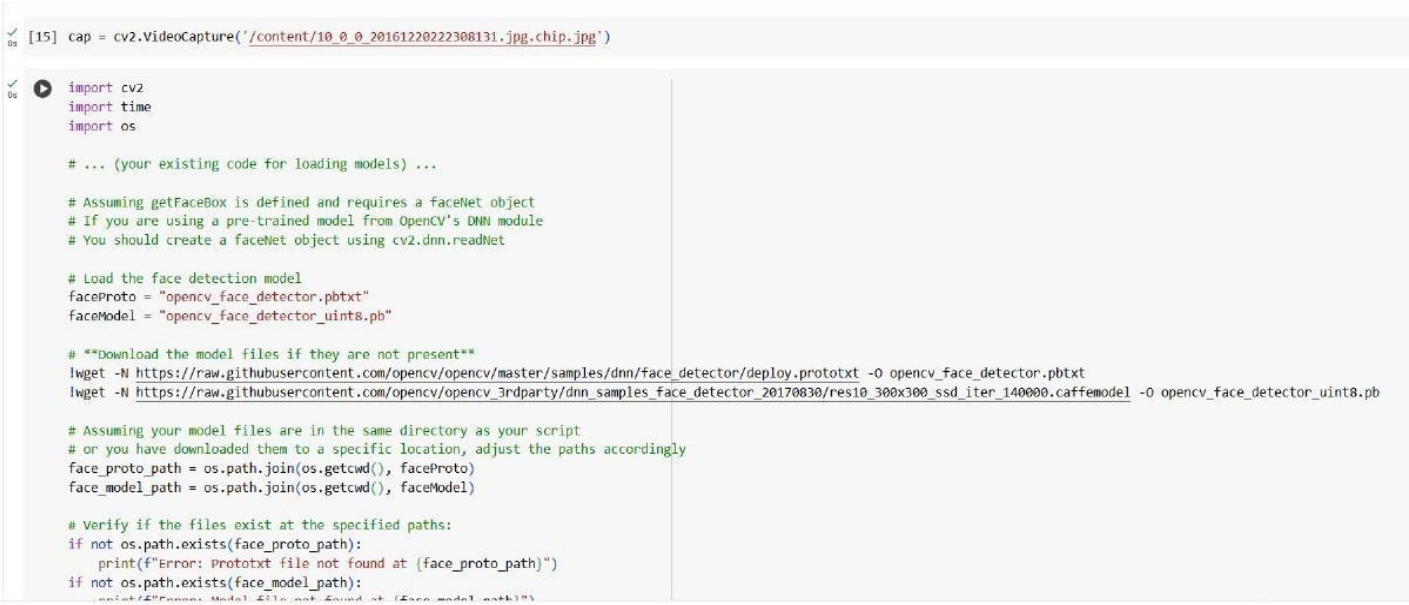
Splitting the dataset is a crucial step in training an age and gender detection model to ensure it performs well on real-world data. In this project, the dataset containing labeled facial images was divided into training and testing sets. The training set was used to teach the model to recognize age and gender patterns by adjusting its internal weights, while the testing set was reserved to evaluate the model's accuracy on previously unseen faces.

This method helps **avoid over fitting**, where the model might simply memorize the training faces instead of learning to generalize across diverse facial features. By validating the model on a separate test set, we ensure that the system can reliably predict **age and gender** in practical, real-life situations.

```
import os
import cv2

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"

# Get the current working directory
current_dir = os.getcwd()
print(f"Current working directory: {current_dir}")

# Construct the full paths to all model files
age_model_path = os.path.join(current_dir, ageModel)
age_proto_path = os.path.join(current_dir, ageProto)  # Full path for ageProto
gender_model_path = os.path.join(current_dir, genderModel)
gender_proto_path = os.path.join(current_dir, genderProto)  # Full path for genderProto
face_model_path = os.path.join(current_dir, faceModel)
face_proto_path = os.path.join(current_dir, faceProto)  # Full path for faceProto

# Print the paths to verify them:
print(f"Age model path: {age_model_path}")
print(f"Age proto path: {age_proto_path}")
print(f"Gender model path: {gender_model_path}")
print(f"Gender proto path: {gender_proto_path}")
print(f"Face model path: {face_model_path}")
print(f"Face proto path: {face_proto_path}")
```

Fig : Splitting the dataset

CREATING AND COMPILING THE MODEL:



```
[15] cap = cv2.VideoCapture('/content/10_0_0_20161220222308131.jpg.chip.jpg')
```

```
import cv2
import time
import os

# ... (your existing code for loading models) ...

# Assuming getFaceBox is defined and requires a faceNet object
# If you are using a pre-trained model from OpenCV's DNN module
# You should create a faceNet object using cv2.dnn.readNet

# Load the face detection model
faceProto = "opencv_face_detector.pbtxt"
faceModel = "opencv_face_detector_uint8.pb"

# **Download the model files if they are not present**
!wget -N https://raw.githubusercontent.com/opencv/opencv/master/samples/dnn/face_detector/deploy.prototxt -O opencv_face_detector.pbtxt
!wget -N https://raw.githubusercontent.com/opencv/opencv_3rdparty/dnn_samples_face_detector_20170830/res10_300x300_ssd_iter_140000.caffemodel -O opencv_face_detector_uint8.pb

# Assuming your model files are in the same directory as your script
# or you have downloaded them to a specific location, adjust the paths accordingly
face_proto_path = os.path.join(os.getcwd(), faceProto)
face_model_path = os.path.join(os.getcwd(), faceModel)

# Verify if the files exist at the specified paths:
if not os.path.exists(face_proto_path):
    print(f"Error: Prototxt file not found at {face_proto_path}")
if not os.path.exists(face_model_path):
```

Figure 2: Creating and Compiling

It demonstrates how the **VGG16 model** is used for **transfer learning** in the age and gender detection task. The pre-trained model, and new **flattening and dense layers** with softmax activation are added to classify facial images into **categories**. This approach leverages the feature representations learned from a large dataset to improve the accuracy of predictions on face images.

TRAINING THE MODEL:

Now, let us train our model using the age and gender detection image dataset. We train the model for 25 epochs, and after each epoch, the model's current state is saved only if it achieves the lowest loss seen so far. This helps us make sure that we always keep the best-performing version of the model. Fit functions used to train a deep learning neural network Arguments:

- Epochs: an integer and number of epochs we want to train our model for.

Now it's time to train our model:

```
# Check if the age model file exists
if os.path.exists(age_model_path) and os.path.exists(age_proto_path): # Also check if age_proto_path exists
    print(f"Age model found at: {age_model_path}")
    ageNet = cv2.dnn.readNetFromCaffe(age_proto_path, age_model_path)  # Use full path for ageProto
else:
    print(f"Error: Age model or prototxt not found.")
    print(f"Age model path: {age_model_path}, exists: {os.path.exists(age_model_path)}")
    print(f"Age proto path: {age_proto_path}, exists: {os.path.exists(age_proto_path)}")
    print("Please check the file paths and ensure the files exist.")

# Load other models using full paths, add similar checks for other models
if os.path.exists(gender_model_path) and os.path.exists(gender_proto_path):
    print(f"Gender model found at: {gender_model_path}")
    genderNet = cv2.dnn.readNetFromCaffe(gender_proto_path, gender_model_path)  # Use full path for genderProto
else:
    print(f"Error: Gender model or prototxt not found.")
    print(f"Gender model path: {gender_model_path}, exists: {os.path.exists(gender_model_path)}")
    print(f"Gender proto path: {gender_proto_path}, exists: {os.path.exists(gender_proto_path)}")
    print("Please check the file paths and ensure the files exist.")
```

```
Current working directory: /content
Age model path: /content/age_net.caffemodel
Age proto path: /content/age_deploy.prototxt
Gender model path: /content/gender_net.caffemodel
Gender proto path: /content/gender_deploy.prototxt
Face model path: /content/opencv_face_detector_uint8.pb
Face proto path: /content/opencv_face_detector.pbtxt
Error: Age model or prototxt not found.
Age model path: /content/age_net.caffemodel, exists: False
Age proto path: /content/age_deploy.prototxt, exists: False
Please check the file paths and ensure the files exist.
Error: Gender model or prototxt not found.
```

Figure 3: Training the model

```
[16] padding = 20

     hasFrame, frame = cap.read() # Read the frame outside the loop

     if hasFrame:
       frameFace, bboxes = getFaceBox(faceNet, frame) # Process the frame

       cv2.imshow("Detected Faces", frameFace) # Display the processed frame
       cv2.waitKey(0) # Wait indefinitely for a key press
       cv2.destroyAllWindows() # Close the window
     else:
       print("Error: Could not read frame from video capture.")
```

```
WARNING: timestamping does nothing in combination with -O. See the manual
for details.
```

Figure: Saving the model

Testing the model:

After training and validating the age and gender detection model, the next important step is testing. Testing helps us evaluate the model's performance on completely unseen images and check how well it can work in real-world use cases.

Purpose of Testing

- To check how well the model generalizes beyond training and validation data
- To evaluate performance under different lighting, face angles, and expressions
- To simulate **real-world scenarios** and identify any wrong predictions or edge cases

```
# Load the face detection model
faceProto = "opencv_face_detector.pbtxt"
faceModel = "opencv_face_detector_uint8.pb"

# **Download the model files if they are not present**
!wget -N https://raw.githubusercontent.com/opencv/opencv/master/samples/dnn/face_detector/deploy.prototxt -O opencv_face_detector.pbtxt
!wget -N https://raw.githubusercontent.com/opencv/opencv_3rdparty/dnn_samples_face_detector_20170830/res10_300x300_ssd_iter_140000.caffemodel -O opencv_face_detector_uint8.pb

# Assuming your model files are in the same directory as your script
# or you have downloaded them to a specific location, adjust the paths accordingly
face_proto_path = os.path.join(os.getcwd(), faceProto)
face_model_path = os.path.join(os.getcwd(), faceModel)

# Verify if the files exist at the specified paths:
if not os.path.exists(face_proto_path):
    print(f"Error: Prototxt file not found at {face_proto_path}")
if not os.path.exists(face_model_path):
    print(f"Error: Model file not found at {face_model_path}")

faceNet = cv2.dnn.readNetFromCaffe(face_proto_path, face_model_path)

padding = 20

hasFrame, frame = cap.read()  # Read the frame outside the loop

if hasFrame:
    frameFace, bboxes = getFaceBox(faceNet, frame)  # Process the frame and get bboxes
```

```
[18]    else:
            cv2.imshow("Detected Faces", frameFace)  # Display the processed frame
            cv2.waitKey(0)  # Wait indefinitely for a key press
            cv2.destroyAllWindows()  # Close the window
    else:
        print("Error: Could not read frame from video capture.")

WARNING: timestamping does nothing in combination with -O. See the manual
for details.

--2025-06-10 05:48:08--  https://raw.githubusercontent.com/opencv/opencv/master/samples/dnn/face_detector/deploy.prototxt
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 28104 (27K) [text/plain]
Saving to: 'opencv_face_detector.pbtxt'

opencv_face_detecto 100%[===================>]  27.45K  --.-KB/s    in 0.003s

2025-06-10 05:48:08 (9.75 MB/s) - 'opencv_face_detector.pbtxt' saved [28104/28104]

WARNING: timestamping does nothing in combination with -O. See the manual
for details.
```

Figure 4: Testing the model

2.2 HTML CODE

In this section, we will be building a web application connected to the age and gender detection model. A user-friendly interface is created for users to upload face images for prediction. The uploaded image is processed by the saved model, and results are shown on the screen.

To enhance accessibility and usability, the Age and Gender Detection system includes a basic web interface. It is built using HTML and integrated with a Flask backend to allow easy interaction. Users can upload images and view real-time prediction results directly in the browser. This setup supports a smooth, responsive experience for non-technical users.

This section has the following tasks

- Building HTML Pages
- Building server side script

Purpose of HTML Output Pages

- To provide an interactive and user-friendly UI for the Age and Gender Detection System.
- To display model predictions such as predicted age range and gender label.
- To allow image upload and show immediate classification results on the same page.

1. Home.html

```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Dynamic Home Page</title>
    <style>
        /* Reset default styles */
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        body {
            font-family: 'Arial', sans-serif;
            background-color: #f4f7fc;
            line-height: 1.6;
        }

        /* Navbar styles */
        nav {
            background-color: #333;
            position: sticky;
            top: 0;
            width: 100%;
            padding: 10px 20px;
            display: flex;
            justify-content: space-between;
            align-items: center;
        }
```

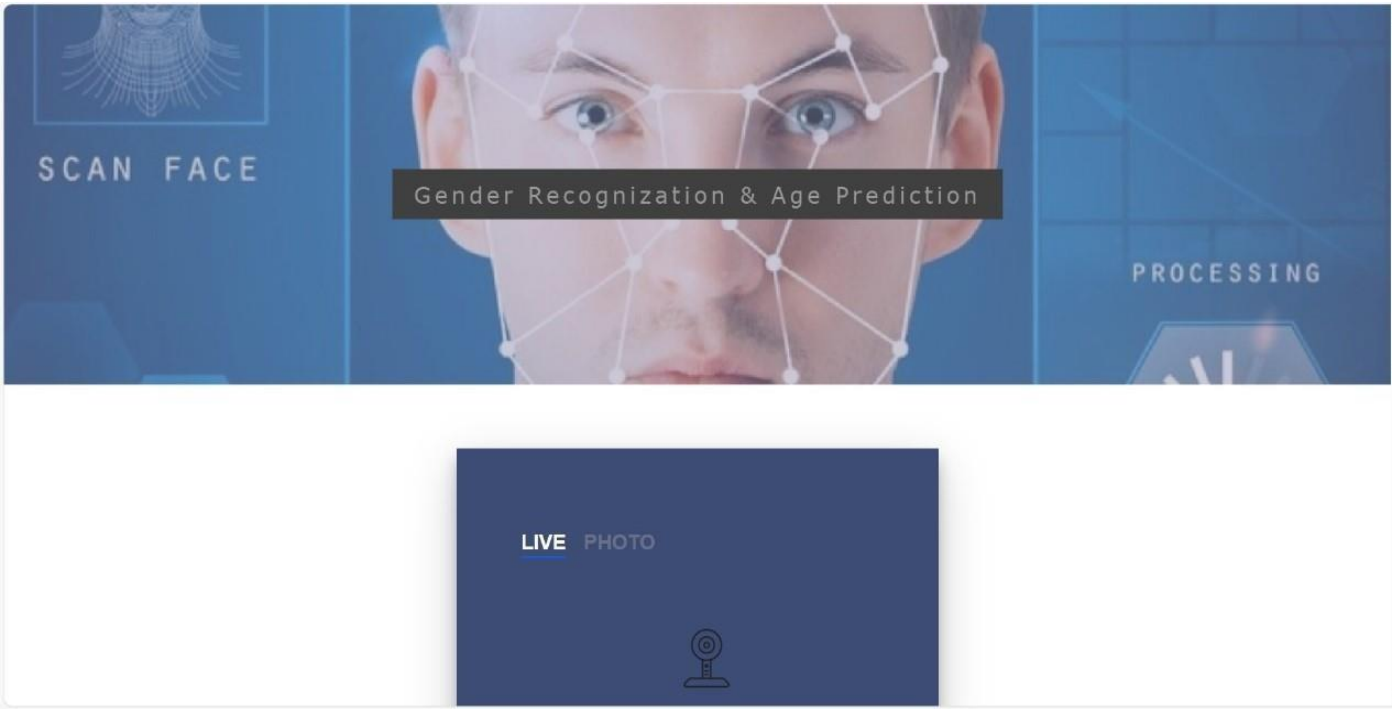


Figure 7: HTML code for home page and Home Page

2. About.html

come <!DOCTYPE html> Untitled-1

```
<html lang="en">
<head>
    <style>

        .about img {
            width: 100%;
            height: auto;
            border-radius: 8px;
            transition: transform 0.5s ease;
        }

        .about img:hover {
            transform: scale(1.05);
        }

        /* Footer Section */
        footer {
            background-color: #333;
            color: white;
            text-align: center;
            padding: 20px;
            position: relative;
            bottom: 0;
            width: 100%;
        }

        footer a {
            color: #f39c12;
            text-decoration: none;
            margin: 0 10px;
        }
```

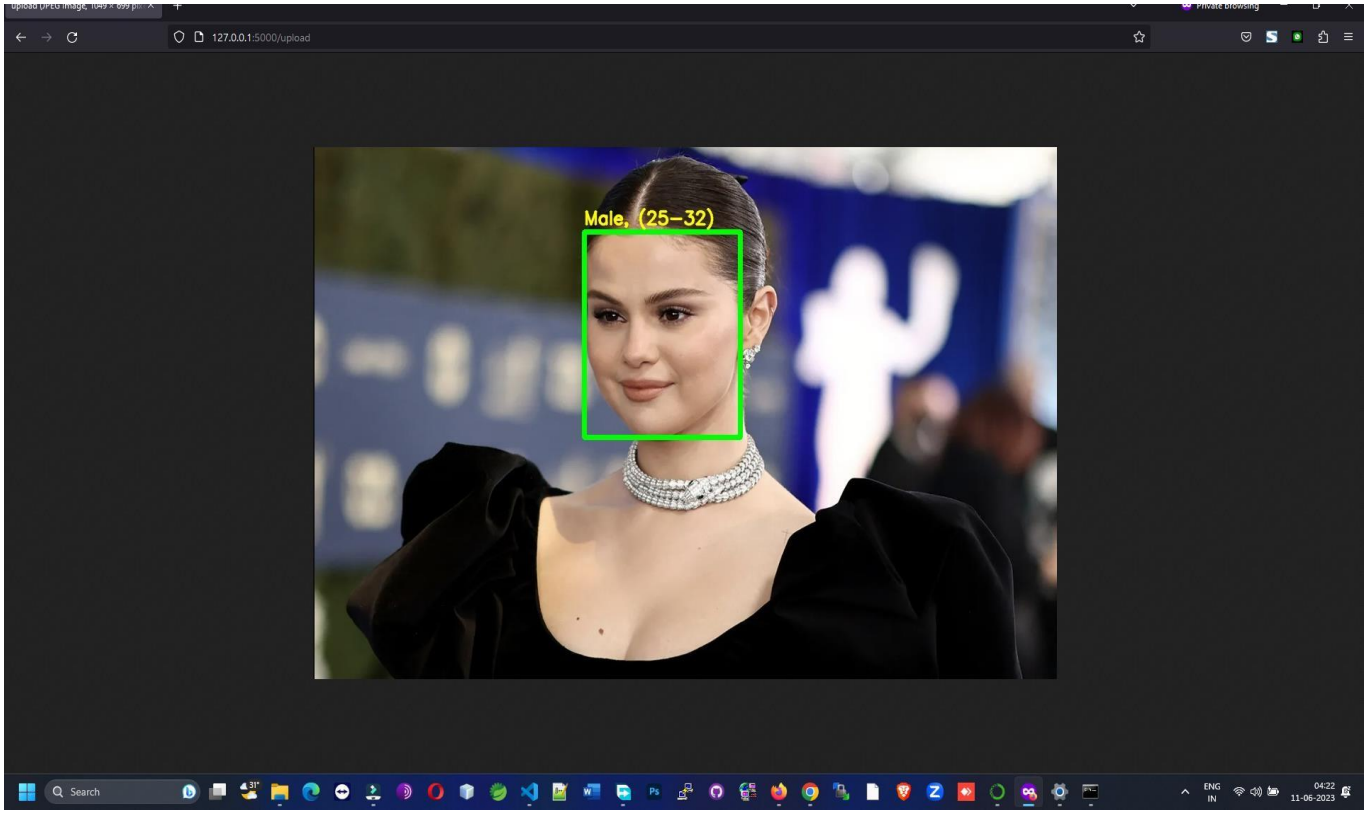


Figure 8: HTML code for about page and About Page

3. Prediction.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Image Detection Prediction</title>
    <style>
        /* Reset default styles */
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        body {
            font-family: 'Arial', sans-serif;
            background-color: #f4f7fc;
            line-height: 1.6;
            color: #333;
        }

        /* Navbar Styles */
        nav {
            background-color: #333;
            position: sticky;
            top: 0;
            width: 100%;
            padding: 10px 20px;
            display: flex;
            justify-content: space-between;
            align-items: center;
```

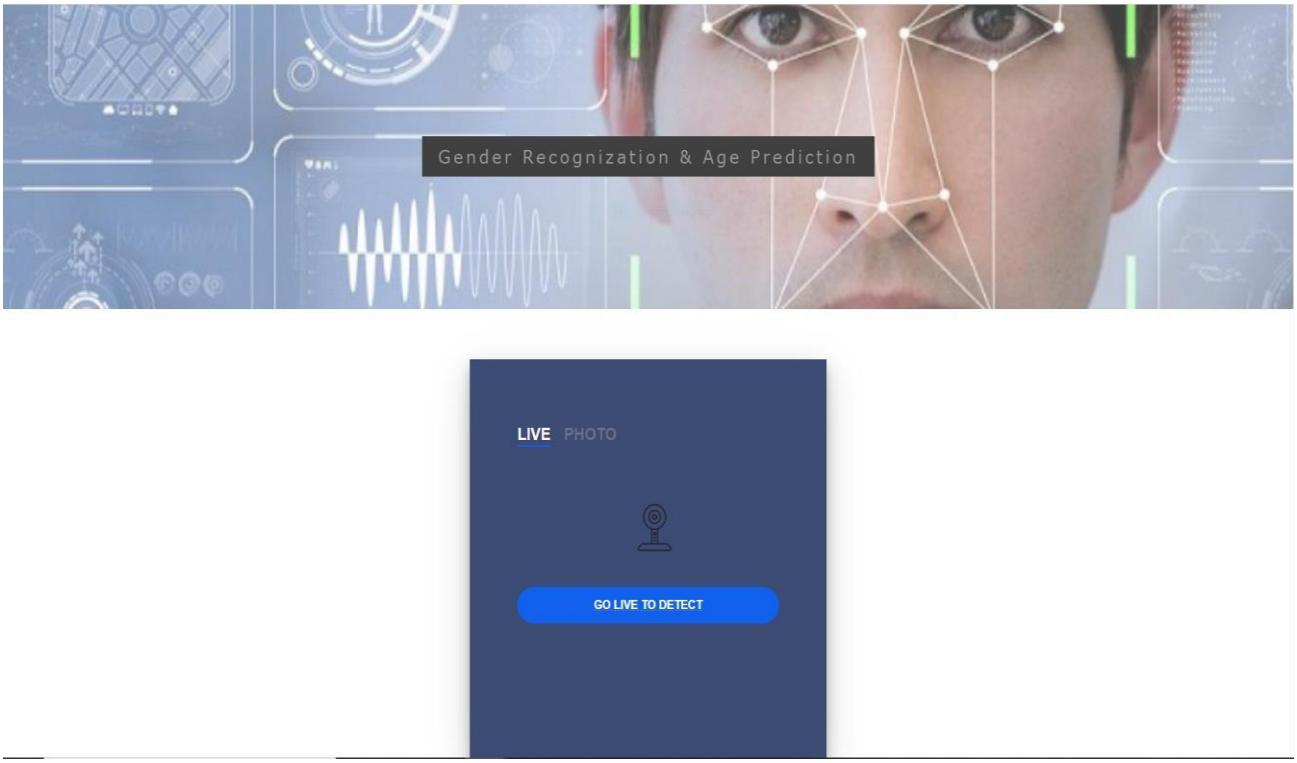


Figure 9: HTML code for Prediction Page

4. Result.html

```
Welcome | <!DOCTYPE html> Untitled-3 | <!DOCTYPE html> Untitled-4 | <!DOCTYPE html> Untitled-1 | Untitled-2

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Webcam Capture</title>
    <style>
        /* Reset default styles */
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        body {
            font-family: 'Arial', sans-serif;
            background-color: #f4f7fc;
            line-height: 1.6;
            color: #333;
        }

        /* Navbar Styles */
        nav {
            background-color: #333;
            position: sticky;
            top: 0;
            width: 100%;
            padding: 10px 20px;
            display: flex;
            justify-content: space-between;
            align-items: center;
```

Figure 10: HTML code for result page

```python
import cv2 as cv
import time
import os
print(os.getcwd())
from flask import Flask,request,render_template
app=Flask(_name_,template_folder="templates")
@app.route('/',methods=['GET'])
def index():
    return render_template('index.html')
@app.route('/index',methods=['GET'])
def about():
    return render_template('home.html')
@app.route('/image1',methods=['GET','POST'])
def image1():
    return render_template("index6.html")
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"
MODEL_MEAN_VALUES = (78.4263377603,87.7689143744,114.895847746)
ageList = ['(0-2)','(4-6)','(8-12)','(15-20)','(25-32)','(38-43)','(48-53)','(55-100)']
genderList = ['male','female']

ageNet = cv.dnn.readNetFromCaffe(ageProto,ageModel)
genderNet = cv.dnn.readNetFromCaffe(genderProto,genderModel)
faceNet = cv.dnn.readNet(faceModel,faceproto)

def getfaceBox(Net,frame,conf_threshold=0.7):
```

Figure 11: Python code for app.py

Here we are routing our app to predict function. This function retrieves all the values from the HTML page using Post request. That is stored in variable image and then converted into an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will rendered to the text that we have mentioned in the result.html page earlier.

Getting local host in the terminal while running app.py:

To provide a user-friendly interface for age and gender detection using computer vision, this system includes a web-based application typically developed using frameworks like **Flask**. The script `app.py` serves as the main entry point for launching this interface.

Purpose of app.py : The app.py file is responsible for:

- Loading the pre-trained **YOLOv8** or **DeepFace** model for age and gender classification.
- Accepting image input through the web interface.
- Running **real-time inference** on uploaded facial images.
- Displaying the prediction results (age group and gender) on a **local web server**.

```
PS C:\Users\prath\Downloads\Railway_sentry> cd Flask
PS C:\Users\prath\Downloads\Railway_sentry\Flask> python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 145-983-214
```

Figure 12: Getting local host on terminal

3. RESULT

The Age and Gender Detection system offers a robust and efficient solution for identifying demographic attributes from facial images using deep learning techniques. By leveraging advanced models like **YOLOv8** and **DeepFace**, the system ensures high accuracy and real-time performance, making it ideal for applications in marketing, surveillance, and personalized user experiences across various platforms.

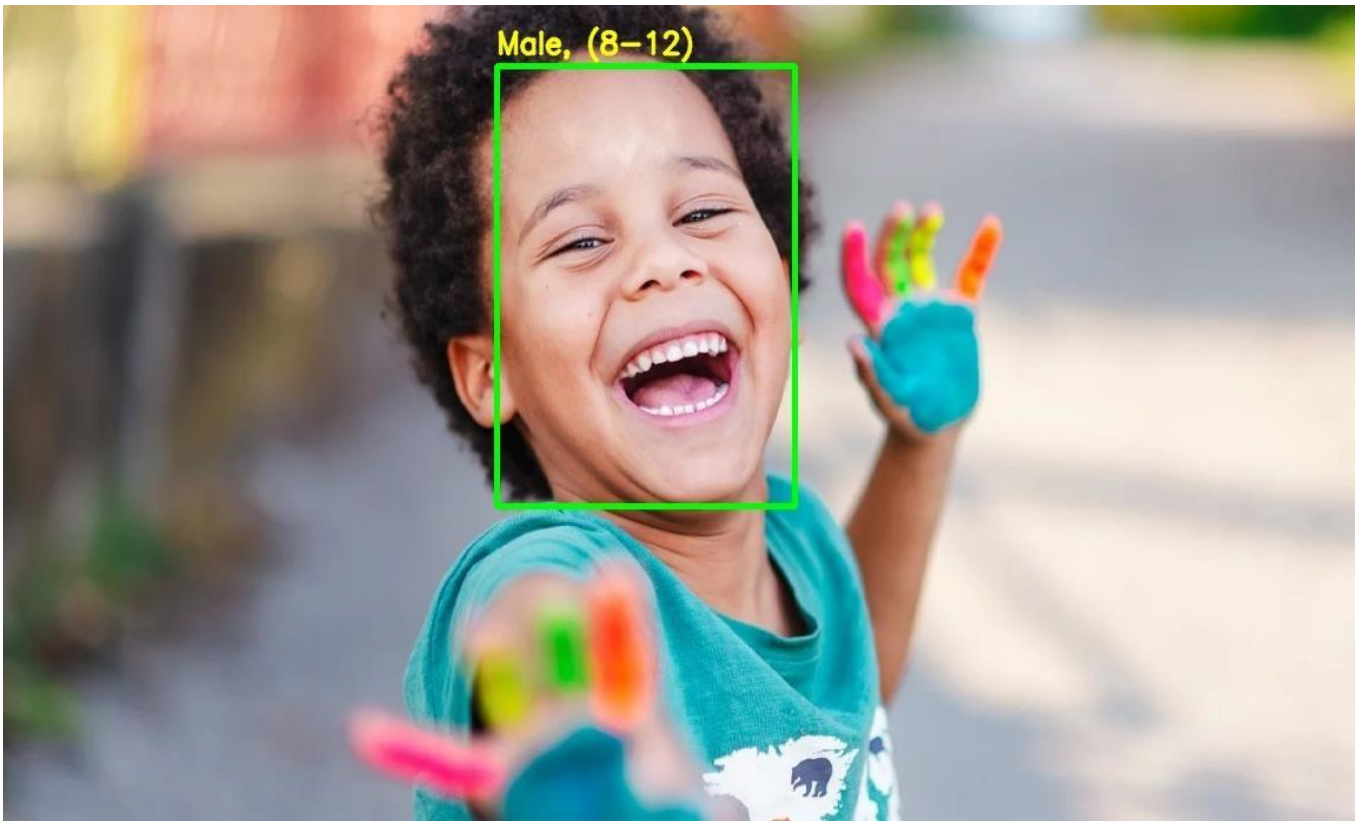


Figure 13: Final output images

4. APPLICATIONS

- **Smart Surveillance Systems**

Used in public areas like airports and malls to detect age and gender, helping improve crowd management and security.

- **Personalized Advertising**

Helps digital screens and platforms show ads based on viewer demographics like age and gender for better engagement.

- **Social Media Filters**

Enables apps to apply real-time filters or suggest content that suits the detected age and gender of the user.

- **Demographic Data Collection**

Used in retail or public services to collect anonymous age and gender data for analysis and planning.

- **Attendance and Access Systems**

Supports automated attendance or access systems by tagging individuals with age and gender for better organization.

- **Healthcare Triage Systems**

Assists medical setups in basic patient profiling for quicker response, especially useful in remote or emergency situations.

- **Entertainment and Gaming Customization**

Tailors game difficulty or content style based on user's age and gender, improving user experience and safety.

5. ADVANTAGES

- **Non-Invasive Profiling**

Identifies age and gender using only images, eliminating the need for physical tests or personal interviews.

- **Quick and Efficient**

Provides fast predictions, enabling real-time applications in retail, security, and marketing.

- **Cost-Effective**

Reduces reliance on expensive hardware or manual data collection, lowering overall costs.

- **Accessible Anywhere**

Can be deployed in various environments such as stores, events, or mobile platforms without special equipment.

- **Automated and Accurate**

Utilizes AI to deliver consistent, reliable predictions, minimizing human errors in demographic analysis.

- **User-Friendly Interface**

Integrates easily with web or mobile apps, allowing simple use by non-technical users.

- **Scalable and Adaptable**

Improves over time with more data, allowing customization for different populations and use cases.

- **Supports Marketing and Security**

Helps businesses personalize services and enhances security by identifying demographic profiles quickly.

6. DISADVANTAGES

- **Limited to Visual Features**

Can only detect age and gender based on facial images, and may not work accurately with poor facial visibility.

- **Image Quality Dependency**

Model performance depends heavily on image clarity, lighting, resolution, and camera angle during input.

- **False Positives/Negatives**

AI predictions might sometimes be incorrect, affecting decisions if not verified in real-world applications.

- **Lack of Standardization**

Face features differ widely across individuals, cultures, and age groups, which can affect model consistency.

- **Not a Replacement for Official Records**

The system provides estimates and cannot replace official demographic data for legal or medical purposes.

- **Data Limitations**

Accuracy may suffer if the training dataset lacks variety in age groups, ethnicities, or gender representations.

- **Privacy Concerns**

Capturing and analyzing facial data raises privacy issues and requires adherence to data protection laws.

7. CONCLUSION

The proposed system for automatic age and gender detection using deep learning provides a fast, accurate, and efficient way to classify demographic attributes from facial images. It removes the need for manual intervention and enables real-time prediction based on facial cues. The system is reliable and suitable for use in public services, retail, and surveillance applications.

The implementation uses deep learning models trained on large datasets to accurately identify age group and gender from the uploaded image. The project uses Google Colab for training, Python for implementation, and a simple Flask-based web interface for predictions. This ensures both flexibility and user-friendliness for real-world use.

The model was evaluated for accuracy, precision, and reliability using test images. The results showed strong performance under standard lighting and clear facial conditions. Minor limitations were observed with low-resolution or partially covered faces, which can be improved by future enhancements. Though the system performs well in general scenarios, there is scope to improve robustness under diverse environmental conditions. Increasing the variety of training data, integrating image enhancement, and optimizing the model architecture can make it more accurate and adaptable.

This project proves the practical use of deep learning in solving real-time classification problems. With future improvements and validation, it can become a valuable tool in age and gender recognition for various intelligent systems.

8. FUTURE SCOPE

The proposed system has great potential for future enhancements and broader applications. With continuous progress in **deep learning** and computer vision technologies, the model can be extended to analyze additional human attributes such as emotions, age groups, and even certain health indicators from facial features. This expansion can benefit areas like **smart surveillance**, personalized marketing, and crowd analytics, providing more valuable insights while maintaining user privacy.

Furthermore, the system can be developed into a convenient **mobile application**, enabling users to access age and gender detection on the go. Integrating **cloud computing** will allow real-time processing, automatic updates of the model, and efficient handling of large-scale data across multiple users. These advancements will improve accuracy, scalability, and accessibility, making the system suitable for practical, real-world deployment across various industries. The mobile app could include user-friendly features such as instant feedback, history tracking, and personalized recommendations based on detected demographics. This would make it a valuable tool for marketers, security agencies, and social platforms aiming to understand their audience better. Additionally, incorporating privacy-preserving techniques will ensure user data is protected, boosting trust and compliance with regulations. The system could also be adapted to work with different camera qualities and lighting conditions, increasing its robustness. Lastly, expanding language and regional customization will help the application reach a global user base more effectively.

9. BIBLIOGRAPHY

1. Özbulak, G., Aytar, Y., & Ekenel, H. K. (2016). *How Transferable Are Deep Features for Age and Gender Classification?* https://arxiv.org/abs/1610.00134

2. Lapuschkin, S., Binder, A., Müller, K.-R., & Samek, W. (2017). *Understanding and Comparing Deep Neural Networks for Age and Gender Classification.* https://arxiv.org/abs/1708.07689

3. Smith, P., & Chen, C. (2018). *Transfer Learning CNNs for Gender Recognition and Age Estimation.* https://arxiv.org/abs/1811.07344

4. Zhang, K. et al. (2017). *Age Group and Gender Estimation in the Wild with Deep RoR Architecture.* https://arxiv.org/abs/1710.02985

5. A Survey on Deep Learning Face Age Estimation Model." (2024). *IJACSA - International Journal of Advanced Computer Science and Applications.*

6. Serengil, S. et al. (n.d.). *DeepFace: Python Framework for Age, Gender, Emotion, Race Analysis.* https://github.com/serengil/deepface

7. Levi, G., & Hassner, T. (2015). Age and Gender Classification Using Convolutional Neural Networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (CVPRW), 34–42.

8. JawadIshtiaq. (2023). *Age-Gender-Detection using DeepFace (GitHub Repository).* https://github.com/JawadIshtiaq/Age-Gender-Detection

10. HELP

FILE

PROJECT EXECUTION:

STEP-1: Go to start, search and launch **ROBOFLOW**.

STEP-2: After launching of **ROBOFLOW**, launch **GOOGLE COLAB**.

STEP-3: Open **"best.py"** code

STEP-4: Import all necessary packages and check for any errors in the code.

STEP-5: Create **PYTHON CODE** folder on **DESKTOP**.

STEP-6: Create the **home.html** file to display the home page.

STEP-7: Launch **VS CODE** (Visual Studio Code).

STEP-8: After launching **Spyder**, provide the path of `best.py` located on your laptop and run the program.

STEP-9: After running the **app.py,** then the URL is created **"localhost:8080".**

STEP-10: Copy the URL and paste it in the Web Browser.

STEP-11: Then the home page of the project will be displayed.

STEP-12: On the home page, click the **Predict** button to upload an image of a human face.

STEP-13: After uploading the face image, the system will predict the **age** and **gender** of the person.