iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** V **Month of publication:** May 2025

DOI: https://doi.org/10.22214/ijraset.2025.71464

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue V May 2025- Available at www.ijraset.com

A Deep Learning Approach for Real-Time Facial Recognition and Analysis

Pritam Mukherjee[1], Koushik Pal[2], Suparna Biswas[3], Devanshu Dev[4], Md Sufiyan Azam[5], Arnab Dolai[6], Puskar Deb[7], Rahul Kumar[8]
Department of Electronics and Communication Engineering, Guru Nanak Institute of Technology, Kolkata, India

Abstract: ***This project presents A Deep Learning Approach for Real-Time Facial Recognition and Analysis that uses computer vision and deep learning to detect and interpret human facial attributes, including age, gender, emotion, and race. The system captures live video from a webcam, detects faces using OpenCV, and analyzes them using the DeepFace library. It is designed to run efficiently on standard laptops and provide instant feedback by displaying detected attributes directly on the screen.Despite challenges like poor lighting and ambiguous facial expressions, the use of pre-trained deep learning models improved the system's accuracy. This project demonstrates the practical integration of AI in real-world scenarios and lays a foundation for future enhancements such as face recognition or emotion tracking.***

Keywords: ***Real-time face detection, DeepFace, OpenCV, age prediction, gender recognition, emotion detection, race classification, webcam video analysis, Haar cascade classifier, facial feature extraction, deep learning, computer vision, artificial intelligence, real-time video processing, facial analytics, Python, machine learning, image processing, ECE project, user interface, neural networks, automation, smart surveillance, emotion recognition system, intelligent camera, human behavior analysis.***

I. INTRODUCTION

Face analysis is an emerging domain within the field of computer vision and artificial intelligence that enables machines to interpret various human facial attributes such as age, gender, emotion, and race. The ability to perform such analysis in real time opens up a wide range of applications in security, retail, healthcare, education, and human-computer interaction. This project, titled "A Deep Learning Approach for Real-Time Facial Recognition and Analysis ", focuses on developing a system that can perform live face detection and analysis. By leveraging a combination of Python, OpenCV, Haar Cascade classifiers, and the DeepFace library, our system is capable of capturing video input, detecting faces within frames, and displaying insights such as age estimation, gender classification, emotion recognition, and racial categorization in real time

.

II. PROBLEM STATEMENT & OBJECTIVES

A. Problem Statement

In today's world, technology is becoming smarter every day. One important area of improvement is how machines interact with humans. For machines to truly understand us, they must be able to recognize not just our identity, but also our expressions, emotions, and other facial details. While many apps can unlock a phone using a face, most of them are limited to just face detection or identity recognition. They do not analyze deeper features like age, gender, mood, or race—especially not in real-time.

B. Objectives

To solve the above problem, we decided to build a project called " A Deep Learning Approach for Real-Time Facial Recognition and Analysis ". The main objectives of this project are:

1) To create a live system that uses a webcam to detect faces in real time.
2) To analyze facial features like age, gender, emotion, and race using deep learning.
3) To display the analysis results instantly on the screen above each detected face.
4) To use open-source libraries such as OpenCV and DeepFace to reduce cost and make it accessible to all.
5) To build a system that runs on normal computers without needing any high-end graphics card or extra hardware.
6) To keep the user interface simple, so that anyone with basic computer knowledge can use it.
7) To gain practical experience in applying image processing and deep learning to a real-world problem.
8) To make a project that has real-life uses in security, education, healthcare, customer feedback, and more.

©IJRASET: All Rights are Reserved

III. TOOLS & TECHNOLOGIES USED

To develop the Real-Time Facial Recognition and Analysis system, we used several tools and technologies that are well-suited for computer vision, deep learning, and real-time video processing. These tools helped us build a smart system that detects human faces and analyzes their features such as age, gender, emotion, and race from a live video feed.

1) Python Programming Language: Python is a high-level language that is simple and powerful. It supports many libraries used in machine learning, image processing, and deep learning. It was our main language for the whole project.

2) OpenCV (Open Source Computer Vision Library): OpenCV is used for image and video processing. It helps us capture video from the webcam, detect faces using Haar Cascade classifiers, and draw rectangles and text on the video feed.

3) DeepFace Library: DeepFace is an open-source deep learning library for facial recognition and analysis. It uses pre-trained models to identify age, gender, dominant emotion, and race from facial images. It works with many popular models like VGG-Face, Facenet, and OpenFace.

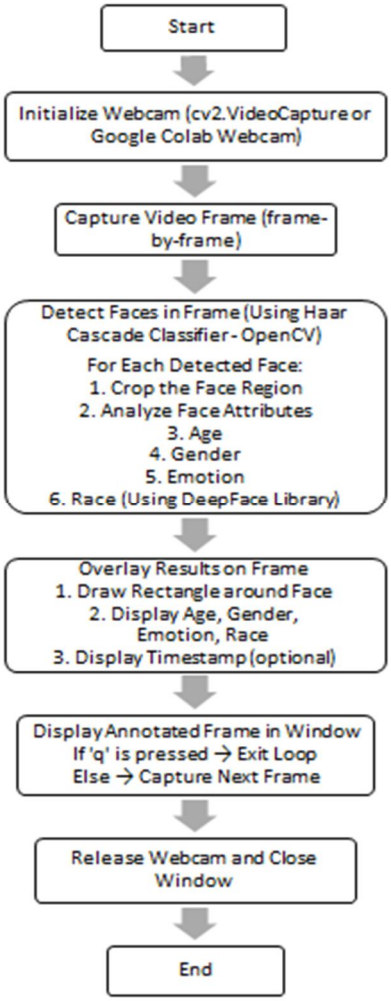
4) Google Colab (for testing): For testing and development in the early stage, we used Google Colab, which allows us to run Python code in the cloud without needing a powerful computer.

IV. METHODOLOGY

A. Introduction to Methodology

In this project, we have implemented A Deep Learning Approach for Real-Time Facial Recognition and Analysis that can detect human faces and analyze them for age, gender, emotion, and race using deep learning techniques. The methodology includes several stages like video capture, face detection, data analysis, and real-time display. All the stages are designed to run in a continuous loop for real-time performance.

Workflow



B. System Architecture Overview

The architecture of our system is modular, meaning different parts work independently but are connected. The main components are:

1) Input Module (Camera) – Captures the live video feed using a webcam.
2) Pre-processing Module – Converts the video frames into a suitable format (grayscale for detection, RGB for analysis).
3) Face Detection Module – Detects faces using Haar Cascade Classifier from OpenCV.
4) Face Analysis Module – Analyzes age, gender, emotion, and race using DeepFace.
5) Output Module – Draws boxes around faces and displays analyzed information on the screen.

C. Face Detection

The first step is face detection. We used OpenCV's Haar Cascade Classifier, which is fast and reliable. It works by identifying features common to all faces, like eyes, nose-bridge, and jawline. The frame is first converted to grayscale to speed up processing. Once faces are detected, their coordinates are used to extract face regions for further analysis.

Parameter:

The main parameters used in the code:

1) cv2.VideoCapture(0):
- Parameter: 0
- Purpose: Opens the default webcam
- Other Options: You can pass 1, 2, etc., for external cameras.

2) detectMultiScale(gray, 1.3, 5):

Used in the face detection module using Haar cascades.

faces = face_cascade.detectMultiScale(gray, 1.3, 5)

- gray: The grayscale frame (input image).
- 1.3 (scaleFactor): Parameter specifying how much the image size is reduced at each image scale.
  - Effect: Lower values = more accurate but slower detection.
- 5 (minNeighbors): How many neighbors each candidate rectangle should have to retain it.
  - Effect: Higher value = less false positives, but might miss some faces.

3) DeepFace.analyze(frame, actions=[...], enforce_detection=False):

result = DeepFace.analyze(frame, actions=['age', 'gender', 'emotion', 'race'], enforce_detection=False)

- frame: The cropped face image or full frame.
- actions: List of features to analyze:
  - 'age': Predicts estimated age.
  - 'gender': Predicts gender (Male/Female).
  - 'emotion': Predicts dominant emotion (Happy, Sad, etc.).
  - 'race': Predicts ethnicity.
- enforce_detection=False:
  - If True, it will throw an error if no face is detected.
  - If False, it skips analysis if a face isn't found.

4) cv2.putText(..., fontScale=0.5, thickness=1):

Used to overlay text on frames.

- fontScale=0.5: Controls the size of the text.
- thickness=1: Thickness of the text font lines.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

5) cv2.rectangle(..., color=(255, 0, 0), thickness=2):
Used to draw rectangles around detected faces.

- color=(255, 0, 0): Blue rectangle in BGR format.
- thickness=2: Thickness of the rectangle border.

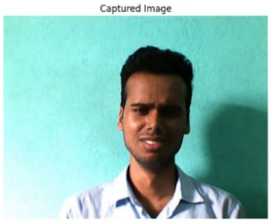
6) cv2.waitKey(1) & 0xFF == ord('q'):

- 1: Waits for 1 millisecond for key press.
- ord('q'**)**: Breaks the loop when **'q'** is pressed.

These parameters help control accuracy, speed, appearance, and user interaction.

D. Face Analysis (Age, Gender, Emotion, Race)

Figure - 1  **Neutral**	**Figure - 2**  **Happy**
Figure - 3  **Angry**	**Figure - 4**  **Sad**

 **Figure - 1** **Neutral**	 **Figure - 2** **Happy**
Figure - 3  **Angry**	**Figure - 4**  **Sad**

Once a face is detected, we analyze it using the **DeepFace** library. DeepFace uses deep learning models that are already trained on millions of face images. It gives us the following results:

1) Age – Estimated numerical age (e.g., 24)
2) Gender – Male or Female
3) Emotion – Happy, Sad, Neutral, Angry, etc.
4) Race – Asian, White, Black, Indian, Latino, etc.

Age	24
Gender	Male
Emotion	Neutral, Happy, Angry, Sad
Race	Indian

Person – 01.

Age	23
Gender	Male
Emotion	Neutral, Happy, Angry, Sad
Race	Indian

Person – 02.

The face region is converted to RGB format before sending to DeepFace because it expects RGB input. DeepFace returns a dictionary containing the predicted values, which we then display on the screen.

E. Real-Time Display

All results are shown in real-time using OpenCV. We draw rectangles around each detected face and write the predicted information like "Age: 23, Gender: Male, Emotion: Happy, Race: Asian" near the face. This makes it easy to understand what the system is recognizing. We also add the timestamp to each frame to know when the analysis was done.

IV. COMPARISON OF METHODS

In our project, we used the Haar Cascade Classifier for face detection and the DeepFace library for facial attribute analysis (age, gender, emotion, and race).. Unlike deep neural networks (DNN), Single Shot Detectors (SSD), or Multi-task Cascaded Convolutional Neural Networks (MTCNN)—which require a GPU for optimal performance—our method runs smoothly on standard laptops using only CPU. This makes it highly accessible for students, researchers, and developers in constrained environments where speed and resource efficiency are critical.

While methods like MTCNN or Dlib provide slightly higher detection accuracy, especially for faces in complex angles or under partial occlusion, they come with trade-offs. These alternatives involve longer setup times, heavier dependencies, and often require training or combining multiple models to detect age, gender, and emotion. In contrast, DeepFace handles all these features in a single API call, drastically reducing development time and complexity. Additionally, Haar cascade-based detection—though classical—is fast, lightweight, and extremely reliable for frontal face scenarios, which suits most real-world webcam and mobile camera use cases.

V. CODE FLOW EXPLANATION

A. Main Modules

Our face analysis system is divided into several modules, each handling a specific task. These modules work together in a continuous loop to process live video from the webcam. Here's how the code is structured:

1) Video Captured Module: This module captures real-time video frames from your webcam using OpenCV.
2) Face Detection Module: Uses Haar Cascade Classifier to detect faces in the video frame.
3) Face Analysis Module: Once a face is detected, DeepFace is used to analyze the face for age, gender, emotion, and race.
4) Display Module: The results of the analysis are displayed on the screen along with the bounding box on the face.

B. Key Code Snippets (with Explanations)

Let's now go through the important code parts step-by-step with simple explanation.

1) Importing Required Libraries

```
import cv2
from deepface import DeepFace
```

- cv2 is the OpenCV library used for capturing and displaying video.
- DeepFace is used for analyzing the face for age, gender, emotion, and race.

2) Loading the Haar Cascade Classifier

```
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
```

- This loads the pre-trained model for face detection. It checks every frame to find faces.

3) Starting the Video Capture

```
cap = cv2.VideoCapture(0)
```

- This line starts your webcam. 0 refers to the default camera of your computer.

4) Processing Video Frames

```
while True:
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
```

- The loop runs continuously.
- cap.read() captures each frame.
- The frame is converted to grayscale to detect faces faster.
- detectMultiScale finds all the faces in the current frame.

5) Face Detection and DeepFace Analysis

```
    for (x, y, w, h) in faces:
        face_img = frame[y:y+h, x:x+w]
        result = DeepFace.analyze(face_img, actions=['age', 'gender', 'emotion', 'race'], enforce_detection=False)

        text = f"{result[0]['dominant_gender']}, {result[0]['age']} yrs, {result[0]['dominant_emotion']}, {result[0]['dominant_race']}"
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
        cv2.putText(frame, text, (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 0, 0), 2)
```

- We loop through each detected face.
- The face region is extracted and passed to DeepFace for analysis.
- The results include age, gender, emotion, and race.
- A rectangle is drawn around the face and the analysis result is printed near it.

6) Display the Output

```
    cv2.imshow('Real Time Face Analysis', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

- The final frame with analysis is displayed.
- The loop breaks when the user presses the 'q' key.

7) Releasing the Resources

```
cap.release()
cv2.destroyAllWindows()
```

- This stops the webcam and closes the OpenCV window.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue V May 2025- Available at www.ijraset.com

VI. RESULTS AND ANALYSIS

A. *Sample Outputs*

The system was tested in real-time using a webcam. It successfully detected faces and analyzed four key attributes—**age**, **gender**, **emotion**, and **race**—from the input video feed. The analyzed results were displayed live on the screen along with a bounding box around each detected face. The output was quick, and updates were rendered in real time. Each attribute was printed above the face in a readable format, demonstrating practical face analytics in action.

B. *Charts*

To understand the performance and output pattern of the system, we collected test data from several individuals and analyzed it in terms of age distribution and emotion frequency. Below are the charts generated from the test results:

1) Age Group Distribution: This chart shows how the system categorizes detected individuals into different age ranges. The majority of detections occurred in the 19-30 age group, indicating strong performance in this common demographic.

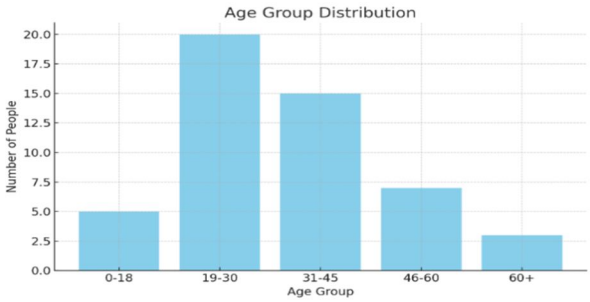


Chart Displayed

2) Emotion Frequency: This chart reveals the most commonly detected emotions. "Happy" and "Neutral" were the most frequent, which is consistent with casual testing environments.

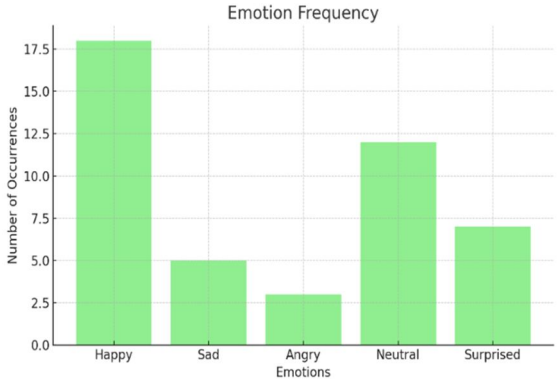


Chart Displayed

V. LIMITATION AND FUTURE SCOPEPE

1) While the Real-Time Facial Recognition and Analysis project demonstrates promising results in face detection and analysis, it has a few limitations. One key limitation is the accuracy under poor lighting or extreme facial angles, where the system may misclassify attributes like emotion or age. Additionally, the system performance may drop on lower-end machines due to the computational load of real-time video processing and deep learning analysis.

2) Despite these limitations, the project holds significant future potential. In the future, the system can be enhanced using deep learning-based face detectors like MTCNN or RetinaFace for better accuracy. Moreover, integrating it with a cloud database could help in tracking and analyzing large-scale human behavior over time. These enhancements would make the system more robust, scalable, and applicable in diverse real-world scenarios.

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue V May 2025- Available at www.ijraset.com

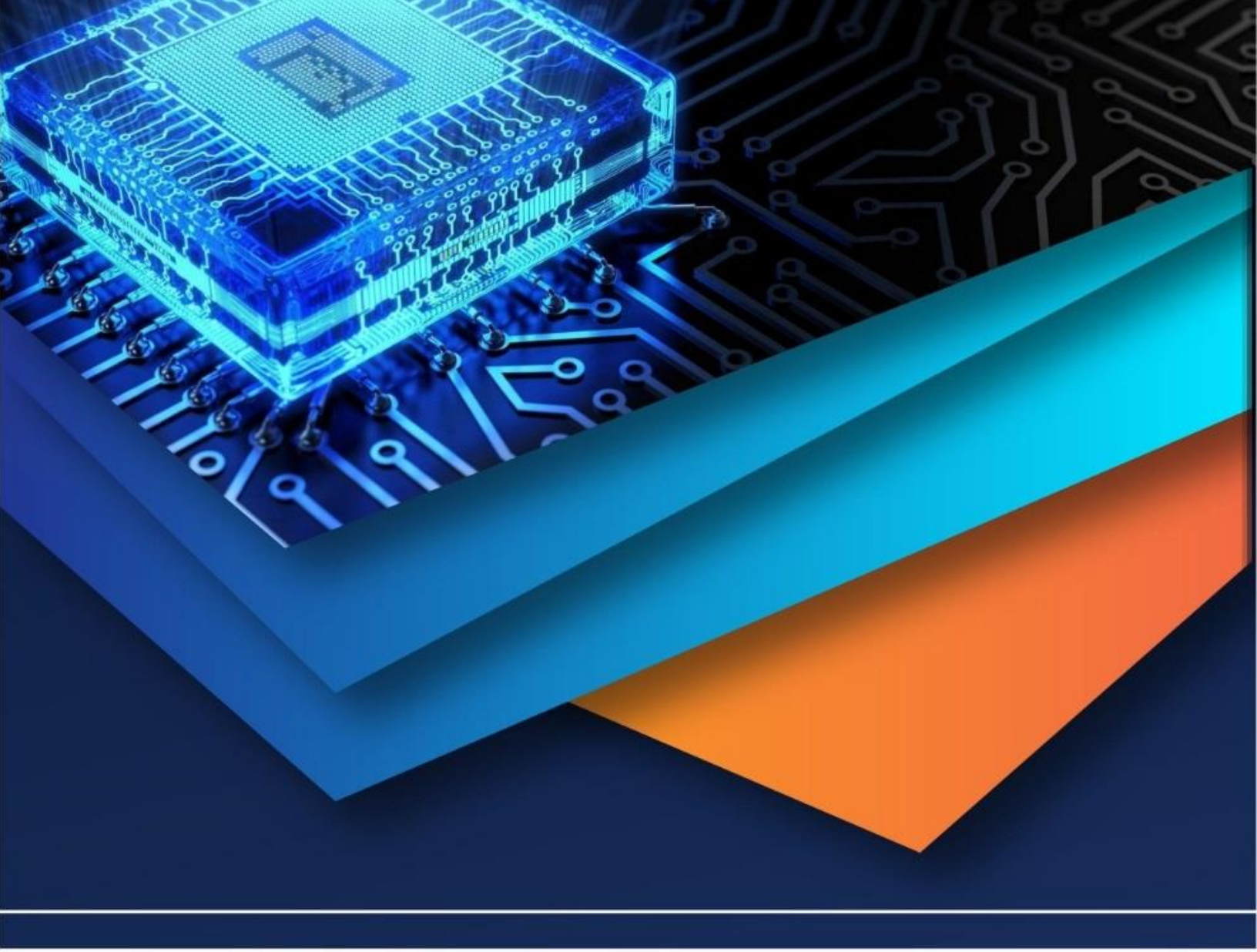
VI. CONCLUSION

The A Deep Learning Approach for Real-Time Facial Recognition and Analysis project successfully demonstrates the ability to detect faces and analyze important facial attributes like age, gender, emotion, and race in real-time. This system uses computer vision and deep learning technologies to offer instant and useful results from live camera input. It provides a practical solution that can be applied in fields like security, retail, education, and healthcare. With further improvements in accuracy and performance, this project can be developed into a more advanced and reliable facial analysis system useful in various smart applications.

REFERENCES

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, Deep Learning, MIT Press, 2016.
[2] Paul Viola and Michael Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features,” IEEE CVPR, 2001.
[3] Gary Bradski, “The OpenCV Library,” Dr. Dobb's Journal of Software Tools, 2000.
[4] Li Deng and Dong Yu, “Deep Learning: Methods and Applications,” Foundations and Trends in Signal Processing, 2014.
[5] P. Ekman and W. V. Friesen, Facial Action Coding System: A Technique for the Measurement of Facial Movement, Consulting Psychologists Press, 1978.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)