FACE EMOTION DETECTION SYSTEM

MINI PROJECT REPORT

Submitted to the Department of Computer Applications, Bharathiar University

in partial fulfilment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

Submitted by

M.NAGARAJAN (22CSEA59)

Under the guidance of

Dr. S.GAVASKAR, MCA.,Ph.D.,

DEPARTMENT OF COMPUTER APPLICATIONS

BHARATHIAR UNIVERSITY

COIMBATORE – 641 046

DECEMBER – 2023

DECLARATION

I hereby declare that this project titled **"FACE EMOTION DETECTION SYSTEM"** submitted to the Department of Computer Applications, Bharathiar University, Coimbatore is a record of original project work done by **M.NAGARAJAN (22CSEA59)** under the supervision and guidance of **Dr. S. GAVASKAR, MCA., Ph.D., Assistant Professor** Department of Computer Applications, Bharathiar University, Coimbatore and that this project work has not previously formed the basis of the award of the Degree / Diploma / Associate Ship / Fellowship or similar title to any candidate of any university.

Place: Coimbatore Signature of Candidate

Date: M.NAGARAJAN

CERTIFICATE

This is to certify that the project work title **"FACE EMOTION DETECTION SYSTEM"** submitted to Department of Computer Applications, Bharathiar University in partial fulfilment of the requirement for the award of Degree in **Master of Computer Applications,** is a record of the original work done by **M.NAGARAJAN (22CSEA59)** under my supervision and guidance and this project work has not formed the basis of the award of any Degree / Diploma / Associate Ship / Fellowship or similar title to any candidate of any university.

Place:

Coimbatore

Date:

Project Guide **Head of the Department**

Submitted for the University Viva-Voce Examination held on————————

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I express my sincere gratitude to our Professor & Head of the Department **Dr. M. Punithavalli M.Sc., M.Phil., Ph.D.,** Department of Computer Applications, Bharathiar University, Coimbatore for acknowledging our request and permitting me to undertake this project work.

I am forever indebted to **Dr. S. GAVASKAR, MCA.,Ph.D., Assistant Professor** Department of Computer Applications, Bharathiar University for providing valuable suggestions and always been enthusiastically guiding us throughout the project.

Finally, I also extend my special thanks to my family, friends and who have kindly provided necessary support for successful completion of the project and for this moral support.

ABSTRACT

The Face Emotion Detection System (FEDS) is an advanced technological solution designed to analyze and interpret human facial expressions to determine emotional states accurately. Emotions play a crucial role in human communication and interaction, making the ability to detect and understand them valuable in various fields, including human-computer interaction, psychology, and marketing. FEDS leverages computer vision and machine learning techniques to achieve robust and real-time emotion recognition.

The system employs a multi-step process, beginning with face detection using state-of-the-art algorithms. Once faces are identified, the system extracts facial features such as eyes, nose, and mouth using image processing techniques. Feature extraction is followed by a deep learning-based model, which has been trained on a diverse dataset of facial expressions, enabling it to recognize and classify emotions accurately.

FEDS supports the detection of a wide range of emotions, including but not limited to happiness, sadness, anger, fear, surprise, and disgust. The model's training encompasses variations in lighting conditions, facial orientations, and ethnicities, ensuring its adaptability to diverse real-world scenarios.

TABLE OF CONTENTS

1.INTRODUCTION

In an era where technology continues to advance at an unprecedented pace, the intersection of artificial intelligence and human emotion is becoming increasingly crucial. The ability to decipher and respond to human emotions holds significant potential for enhancing various aspects of human-machine interaction, ranging from user experiences in technology interfaces to applications in fields such as psychology, marketing, and security. One key facet of this intersection is the development of the Face Emotion Detection System (FEDS), a sophisticated solution aimed at accurately and dynamically recognizing emotional states through facial expressions.

Human communication is inherently nuanced, with emotions serving as a fundamental element in conveying information and fostering understanding. Recognizing these emotional cues in real-time has far-reaching implications, influencing the design of empathetic and responsive technologies. FEDS, grounded in cutting-edge computer vision and machine learning techniques, emerges as a pivotal innovation in this landscape.

This system is designed to decode the intricate language of facial expressions, offering a comprehensive and instantaneous analysis of emotional states. Through a combination of advanced face detection algorithms, image processing techniques, and a deeply trained neural network, FEDS demonstrates unparalleled accuracy in identifying a spectrum of emotions, ranging from joy and surprise to sadness and anger.

The following sections delve into the technical intricacies and features that define the Face Emotion Detection System. From its real-time processing capabilities to its adaptability across diverse scenarios and its commitment to user privacy, FEDS represents a groundbreaking advancement in the realm of emotion-aware computing. As we explore its inner workings, applications, and potential societal impact, the Face Emotion Detection System stands as a testament to the ongoing synergy between artificial intelligence and our innate ability to express and comprehend emotions.

2.SOFTWARE AND HARDWARE REQUIREMENT ANALYSIS

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended.

2.1 Methods

OpenCV

OpenCV (Open Source Computer Vision Library) is released under a BSD license and hence it is free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.

Caffe

Caffe is a well-known and widely used machine-vision library that ported Matlab's implementation of fast convolutional nets to C and C++. Caffe is not intended for other deeplearning applications such as text, sound or time series data. Like other frameworks mentioned here, Caffe has chosen Python for its API. Caffe perform image classification with convolutional nets, which represent the state of the art. In contrast to Caffe, Deeplearning4j offers parallel GPU support for an arbitrary number of chips, as well as many, seemingly trivial, features that make deep learning run more smoothly on multiple GPU clusters in parallel. While it is widely cited in papers, Caffe is chiefly used as a source of pre-trained models hosted on its Model Zoo site.

TenserFlow

TensorFlow is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API. TensorFlow was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for

the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well.

2.2 PROGRAMMIG/WORKING ENVIRONMENT

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems.

CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

Tkinter

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

OpenCV

OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross platform and free for use under the open-source BSD license. OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch and Caffe.

2.3 REQUIREMENTS TO RUN THE APPLICATION

Supportive Operating Systems: The supported Operating Systems for client include: Windows 2010, windows 2008, windows 2007

2.3.1 Software Requirements:

The Software Requirements in this project include:

a. Python

b. OpenCV framework

c. Sqlite

d. MS Excel

Software requirements deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed. OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 7 million. The library is used extensively in companies, research groups and by governmental bodies. As an asynchronous event driven framework

2.3.2 Hardware Requirements:

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibilitylist (HCL), especially in case of operating systems. An HCL lists tested, compatible, and

sometimes incompatible hardware devices for a particular operating system or application. Face detection in a fixed image without special hypothesis is a difficult problem due to the high variability of the shape to detect. Many techniques of detection and face recognition have been developed in recent years and many of which are very efficient. Among these methods, we find the method of Viola and Jones were studied in this work. The aim of our work is to study this method to implement on the CPU with C / C + +, then acceleration was presented with OpenCV. Subsequently, a second implementation in FPGA was presented Keywords— Face detection, method of Viola/Jones, C/C++, OpenCV, FPGA. I. INTRODUCTION Face detection is a new computer technology that determines the locations and sizes of human faces in images.

Components	**Minimum**	**Recommended**
Processor	Intel Core i3-2100 2nd generation	Intel Core i7 5th generation
RAM	4GB	8GB
Camera	HD 720p Webcam	Full HD 1080p Webcam
Disk	128Gb	512Gb

3.FLOW DIAGRAM

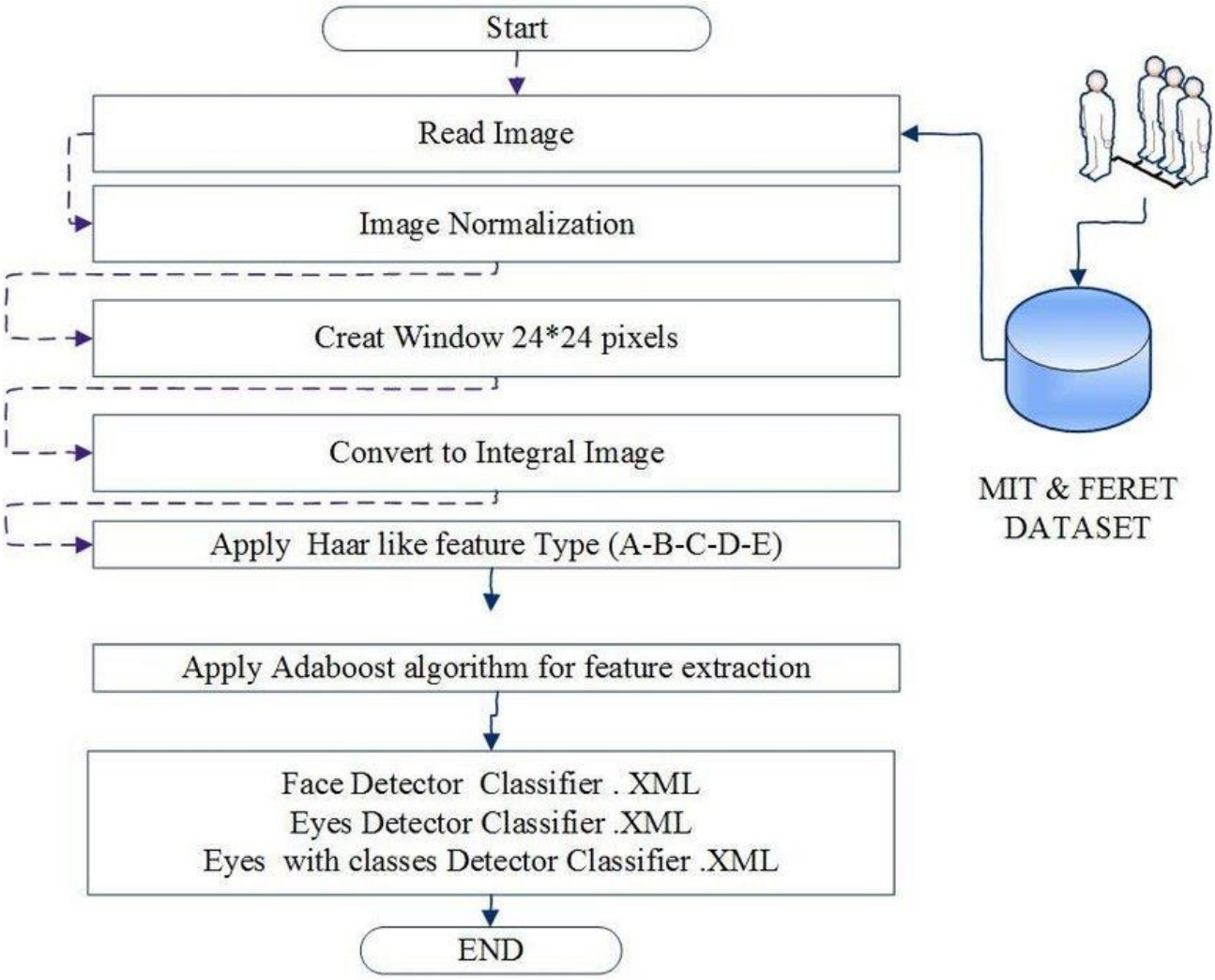


FIGURE 3.1

4.SYSTEM TESTING

System testing is the process of existing software with the intent of finding and ultimately correcting errors. This fundamental philosophy does not change for web applications, because web-based systems and applications reside on a network and interoperate with many different operating systems, browsers, hardware platforms, and communication protocols, the search for errors represents a significant challenge for web applications.

Testing objectives there are several rules can serve as testing objectives. They are

1. Testing is a process of executing a program with the intent of finding an error.

2. A good test is one that has a high probability of finding an undiscovered error.

3. A successful test is one that uncovers and undiscovered error.

The above are the objectives for a good test procedure. A successful test is that in which no errors are found. The objective is to design tests that systematically uncover different classes of errors and to do so with the minimum amount of time and effort. The unit is the testing changes made in an existing or new program. This test is carried out during the programming and each module is found to be working satisfactorily. In the data entry form web applicants, details are stored in the database.

4.1 DATABASE CONNECTIVITY TEST

While connecting the database through my web application I have encountered with a host error for database connectivity which tells there is no existence of the host. Then I rectified the error by changing the host and username information.

4.2 Databases:

Olivetti database: • Face images taken between April 1992 and April 1994. • There are ten different images of each of 40 distinct people. • There are 400 face images in the dataset. • Face images were taken at different times, varying lighting, facial express and facial details. • All face images have black background. • The images are gray level. • Size of each image is 64x64. • Image pixel values were scaled to [0, 1] interval. • Names of 40 people were encoded to an integer from 0 to 39. [2] 3.2.2. Fer 2013 database: The data consists of 48x48 pixel grayscale images of faces. The faces have been automatically registered so that the face is

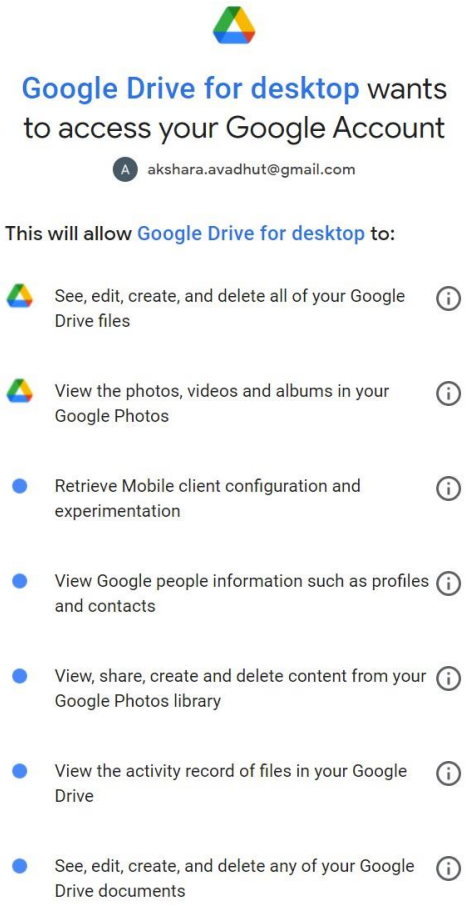
centered and occupies about the same amount of space in each image. The task is to categorize each face based on the emotion shown in the facial expression into one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral). The training set consists of 28,709 examples and the public test set consists of 3,589 examples.

```
from google.colab import drive
drive.mount('/content/drive/')

Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleuser

Enter your authorization code:
```

Allow access through the link.

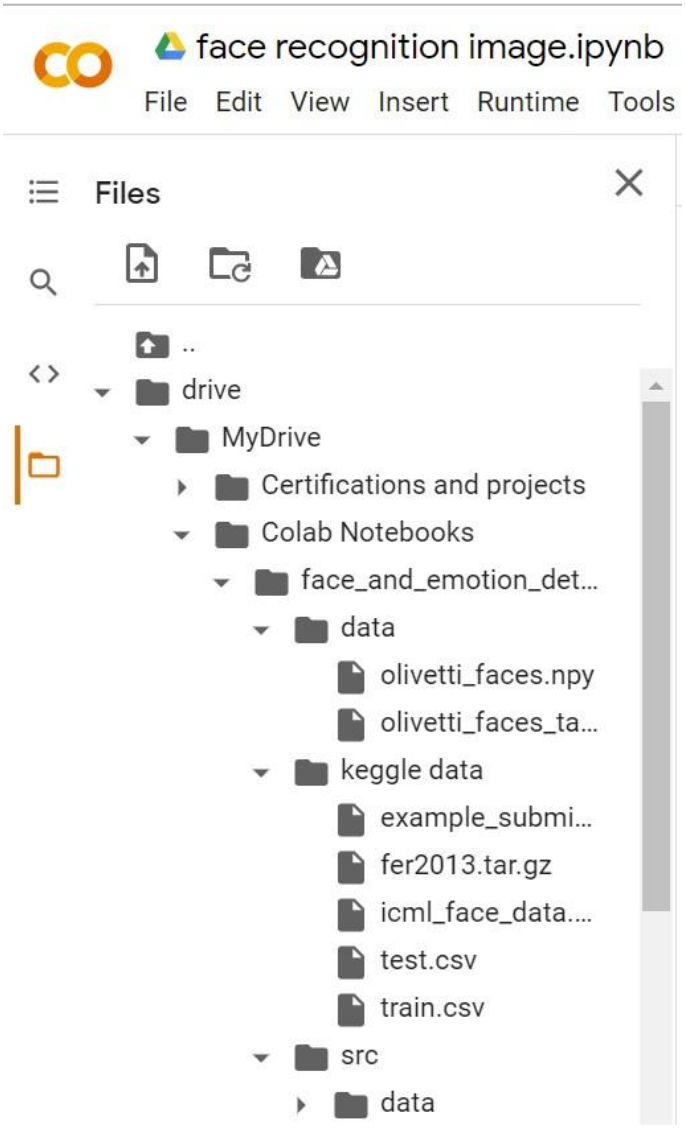


Drive mounted for data access.

```
from google.colab import drive
drive.mount('/content/drive/')

Mounted at /content/drive/
```

Files are accessible.



4.3 IMAGE PROCEESING

```
In [29]:
plt.figure(figsize=[16,16])
for i in range(36):
    img = test_images[i,:,:,0]
    p_dist = cnn.predict(img.reshape(1,48,48,1))
    k = np.argmax(p_dist)
    p = np.max(p_dist)

    cam = GradCAM(cnn, k)
    heatmap = cam.compute_heatmap(img.reshape(1,48,48,1))

    plt.subplot(6,6,i+1)
    plt.imshow(img, cmap='binary_r')
    plt.imshow(heatmap, alpha=0.5, cmap='coolwarm')
    plt.title(f'{emotions[test_labels[i]]} - ({emotions[k]} - {p:.4f})')
    plt.axis('off')
plt.tight_layout()
plt.show()
```

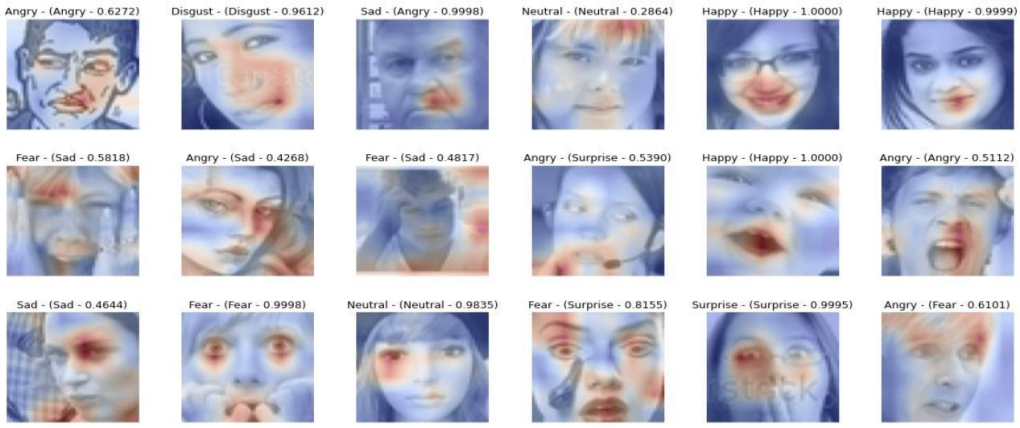


FIGURE 4.3 IMAGE PROCESSING

5.LIMITATIONS

➢ System faces recognition issues if the light conditions are poor

If lighting conditions are poor than it affects the recognition as it recognizes wrong and sometimes don't recognize in poor lighting conditions, so while creating dataset and recognizing it needs proper lighting

➢ Webcam and processor with high specification is required as system consumes a lot of resources.

As this project is doing image processing, so to run it smoothly high specification is required and graphic card is must to run it smoothly otherwise it will be too slow and may hang during its execution.

6.CONCLUSION AND FUTURE IMPLEMENTATION

6.1 CONCLUSION

In conclusion, the Face Emotion Detection System (FEDS) represents a groundbreaking advancement in the intersection of artificial intelligence and human emotion analysis. Its ability to accurately and swiftly recognize a spectrum of emotions through facial expressions has far-reaching implications for human-computer interaction, psychology, and diverse industry applications.

The real-time processing capabilities, high accuracy, and adaptability of FEDS position it as a versatile tool capable of enhancing user experiences and fostering the development of emotionally intelligent technologies. By prioritizing user privacy and incorporating ethical considerations, FEDS sets a standard for responsible deployment of emotion-detection systems in society.

Looking ahead, the future implementation of FEDS holds potential for continuous improvement through expanded and diverse training datasets, multi-modal emotion recognition, and integration into various applications. As technology continues to evolve, FEDS stands as a beacon in the journey toward a more empathetic and responsive technological landscape, contributing to a deeper understanding of human emotion in the digital age.

6.2 FUTURE SCOPE

The Face Emotion Detection System (FEDS) exhibits significant capabilities in emotion recognition, but continuous advancements and refinements can elevate its performance and broaden its scope of applications. Several avenues for future enhancement include:

1. **Fine-tuning Algorithms:** Regular updates and fine-tuning of the underlying algorithms can ensure that FEDS remains at the forefront of emotion detection technology. This involves incorporating the latest advancements in computer vision and machine learning to enhance accuracy and efficiency.
2. **Extended Emotional Spectrum:** Expanding the system's capacity to recognize a broader range of nuanced emotions can provide a more comprehensive understanding of human expressions. This may involve incorporating cultural nuances and subtle emotional cues that contribute to a richer emotional spectrum.

3. **Adaptive Learning:** Implementing adaptive learning mechanisms can enable FEDS to continuously improve its performance based on user interactions and feedback. This iterative learning process ensures that the system evolves to better understand individual and collective emotional expressions over time.
4. **Cross-Modal Integration:** Integrating FEDS with other modalities, such as voice recognition, gesture analysis, or physiological signals, can lead to a more holistic understanding of emotional states. This cross-modal integration enhances the system's accuracy and applicability in diverse scenarios.
5. **Real-time Feedback Mechanism:** Incorporating a real-time feedback mechanism can enable FEDS to adjust its interpretations dynamically. This feature could prove invaluable in scenarios where immediate responsiveness to changing emotional states is crucial, such as in virtual reality environments or interactive technologies.
6. **Edge Computing Optimization:** Further optimizing FEDS for edge computing environments ensures efficient performance on resource-constrained devices, expanding its applicability to a wider array of devices and platforms.
7. **Human-Robot Interaction:** Integrating FEDS into human-robot interaction scenarios can contribute to the development of emotionally intelligent robots. This can enhance the ability of robots to respond appropriately to human emotions, improving the overall quality of human-robot collaboration.
8. **Customization for Specific Industries:** Tailoring FEDS for specific industries, such as healthcare, education, or customer service, can lead to specialized applications that cater to the unique emotional dynamics of each sector.
9. **Continued Ethical Considerations:** As technology evolves, ongoing attention to ethical considerations, privacy standards, and user consent is paramount. Ensuring that FEDS adheres to the highest ethical standards will contribute to its acceptance and responsible use in various contexts.

By pursuing these future enhancements, FEDS can continue to push the boundaries of emotion recognition technology, fostering a deeper understanding of human emotions and enhancing the overall quality of human-machine interactions across diverse domains.

7.APPENDIX

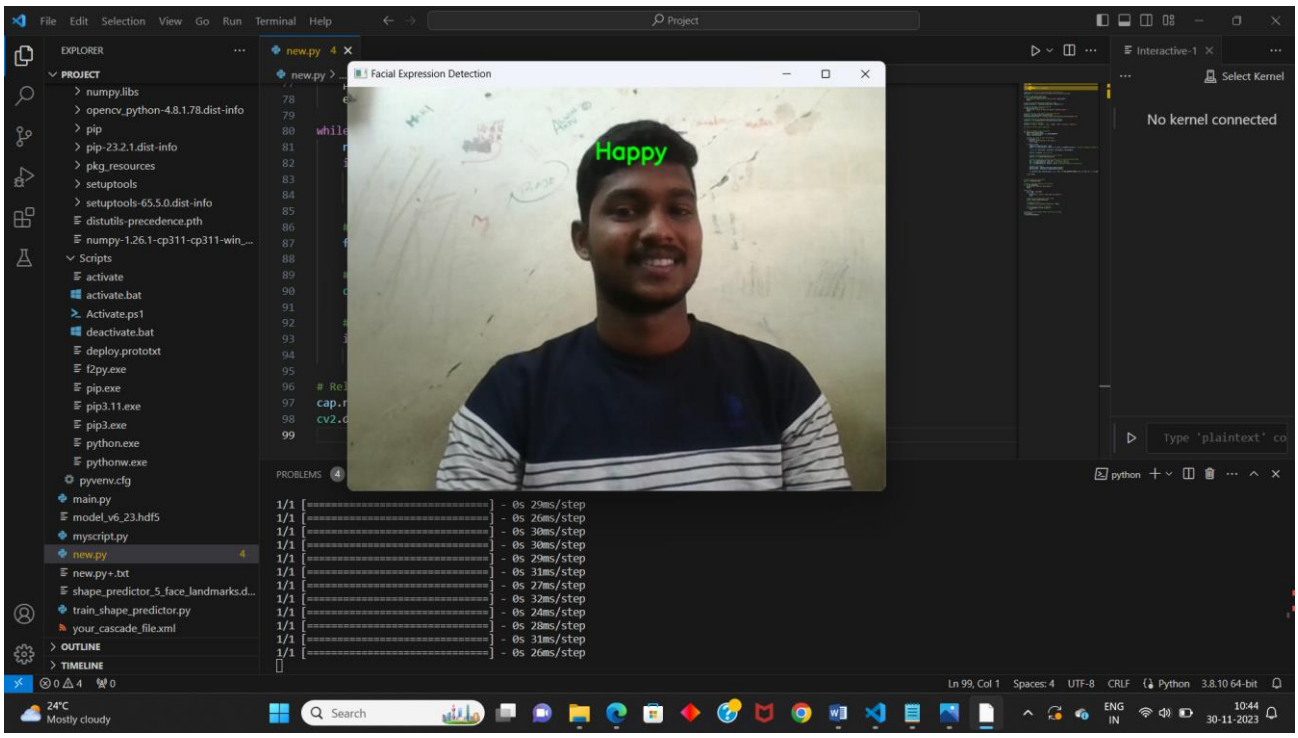


FIGURE 7.1.HAPPY

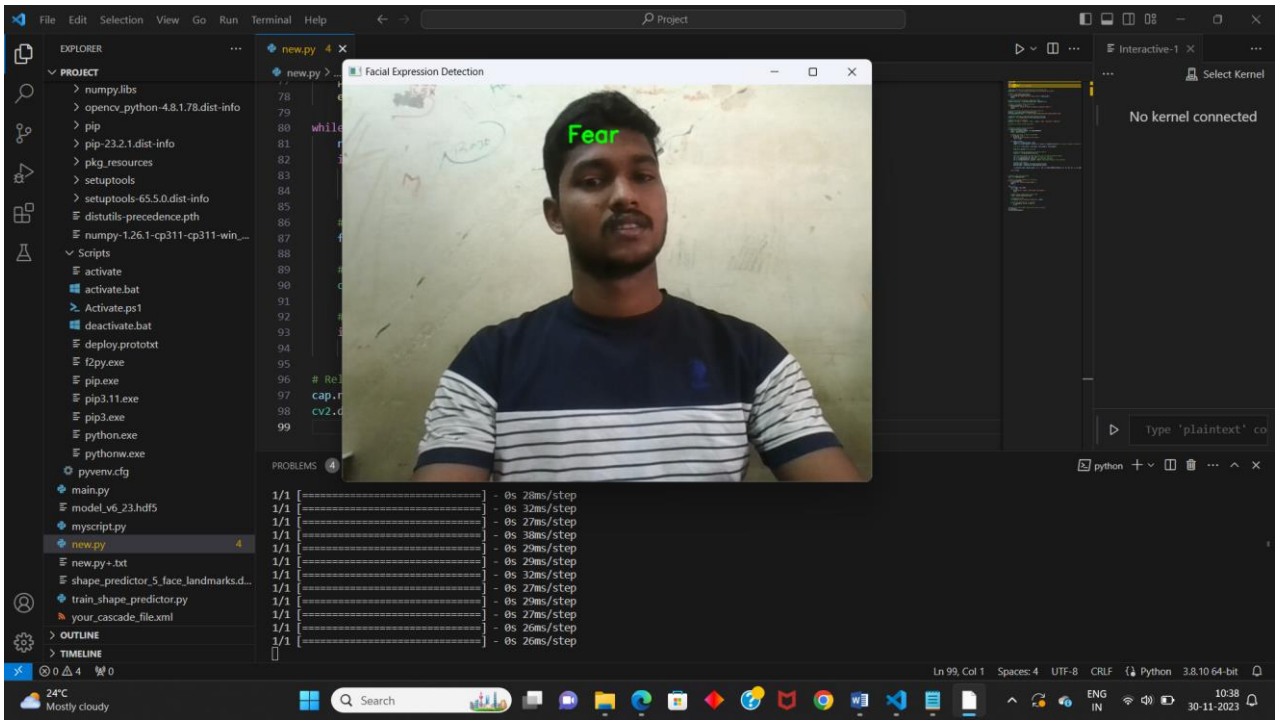


FIGURE 7.2.FEAR

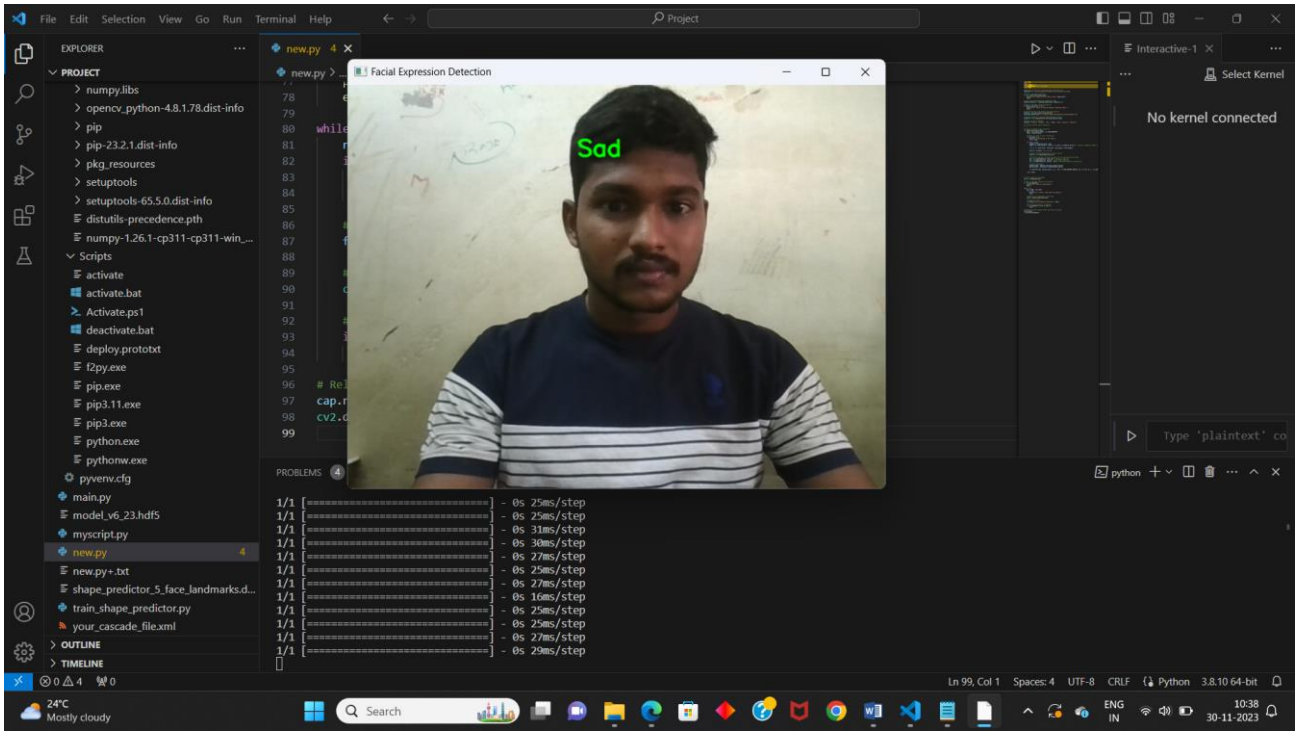


FIGURE 7.3.SAD

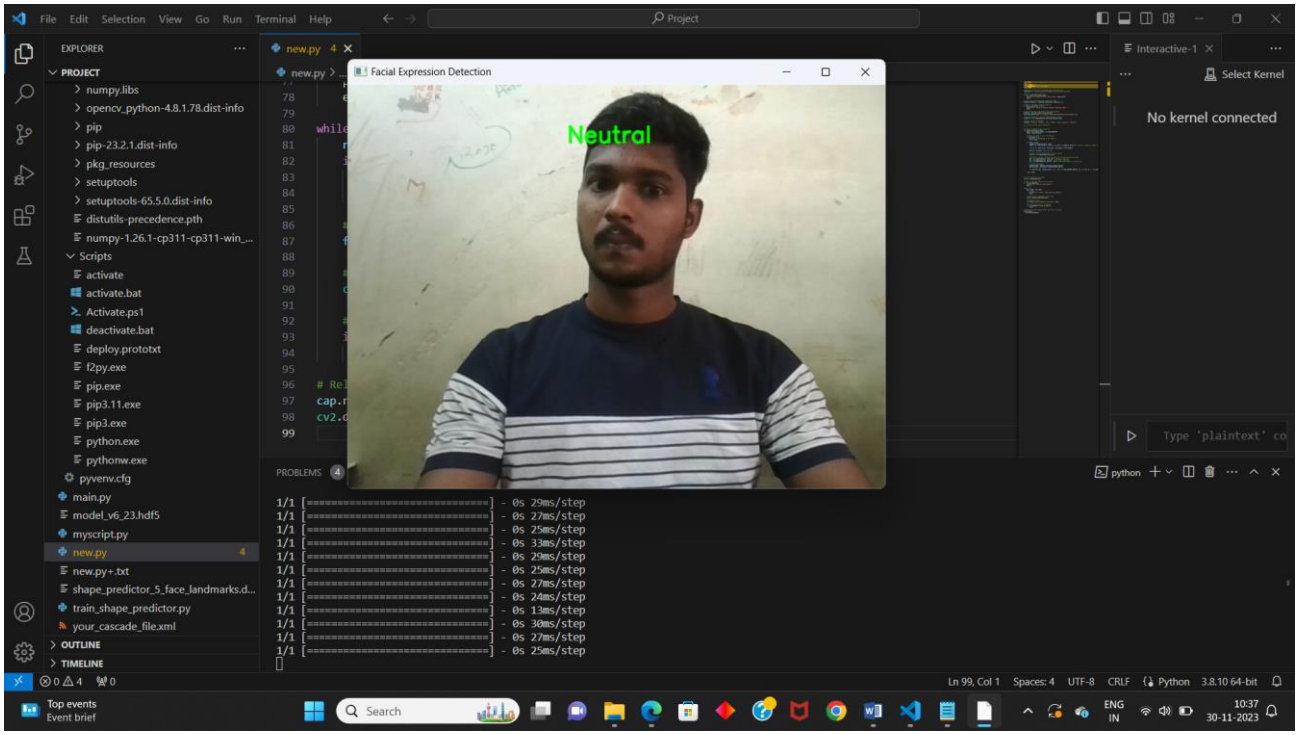


FIGURE 7.4.NEUTRAL

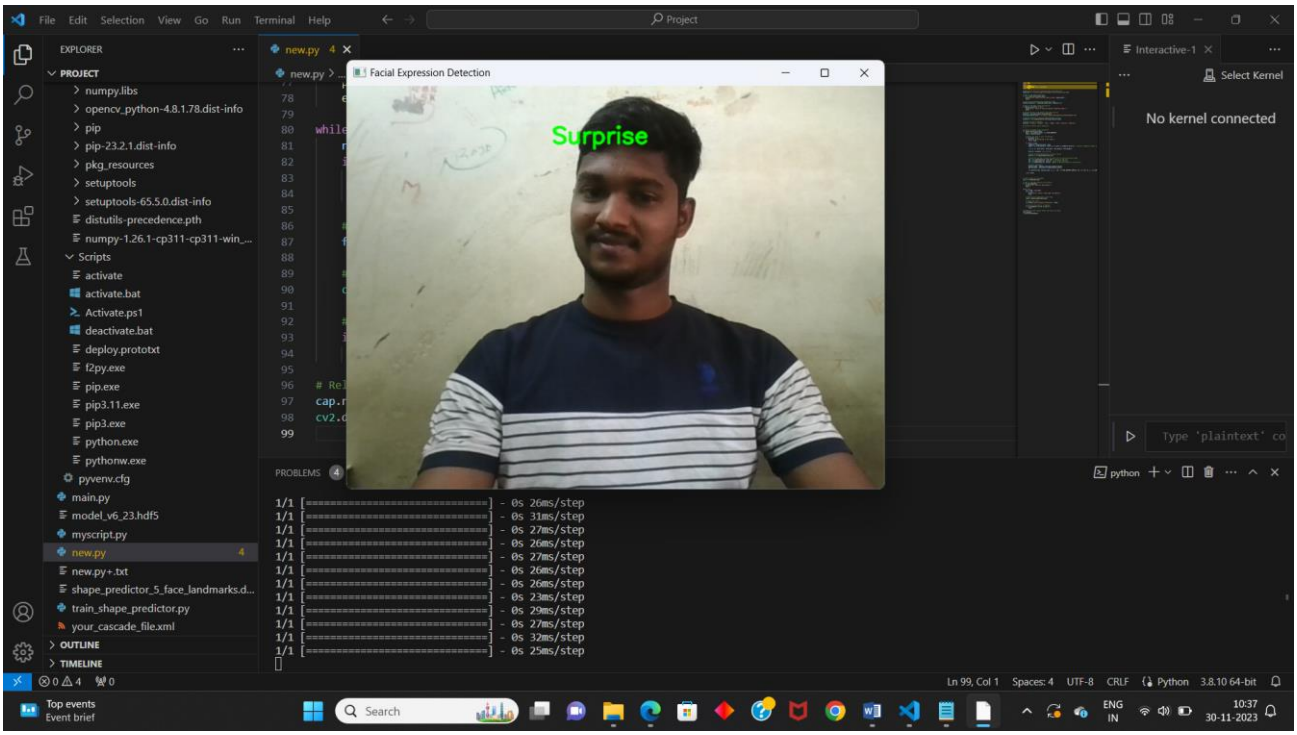


FIGURE 7.5.SURPRISE

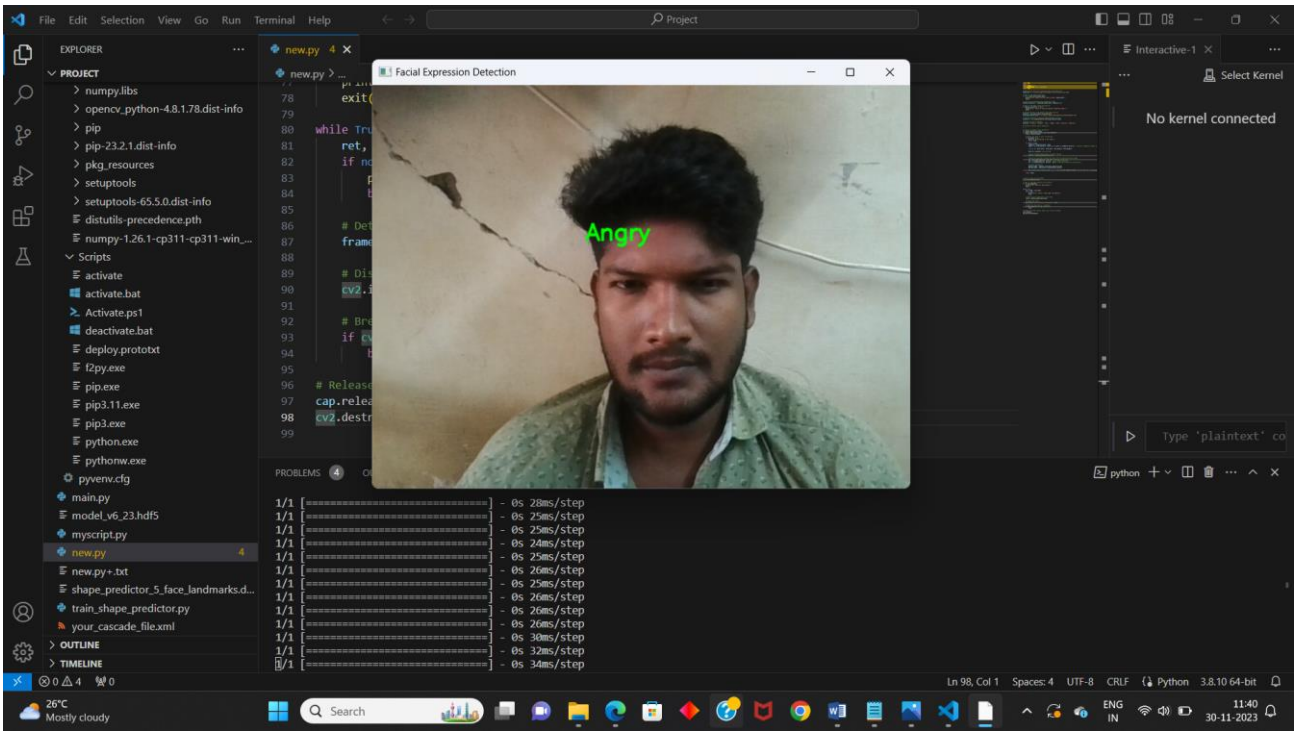


FIGURE 7.6.ANGRY

```
import cv2
import os
import numpy as np
from keras.models import load_model
import dlib

# Replace with the correct absolute path to the model file
model_path = "C:\\Users\\nagar\\Downloads\\Project\\model_v6_23.hdf5"

# Check if the model file exists
if not os.path.exists(model_path):
    print(f"Error: Model file not found at path: {model_path}")
    exit()

# Load the pre-trained emotion recognition model
emotion_classifier = load_model(model_path, compile=False)

# Check if the model loaded successfully
if emotion_classifier is None:
    print("Error: Failed to load the emotion recognition model.")
    exit()

# Initialize dlib's face detector and shape predictor
detector = dlib.get_frontal_face_detector()
existing_predictor_path = "C:\\Users\\nagar\\Downloads\\new_shape_predictor.dat"

# Load the predictor using dlib.shape_predictor directly
```

```
predictor  = dlib .shape_  predictor (existing_predictor_path)

# Define the list of emotions
emotions = ["Angry", "Disgust", "Fear", "Happy", "Sad", "Surprise", "Neutral"]

# Function to detect facial expression
# ...

# Function to detect facial expression
def detect_expression(frame):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = detector(gray)

    # Handle case when no faces are detected
    if len(faces) == 0:
        print("No face detected in the frame.")
        return frame

    for face in faces:
        landmarks = predictor(gray, face)
        shape = np.array([[point.x, point.y] for point in landmarks.parts()])  # Convert landmarks to NumPy array

        x , y, w, h = face.left(), face.top(), face.width(),  face.height()

        face_roi = gray[y:y + h, x:x + w]

        # Apply histogram equalization for better results
```

```
        face_roi = cv2.equalizeHist(face_roi)

        # Resize the face ROI to match the input size for emotion detection model
        roi = cv2.resize(face_roi, (48, 48))  # Resize to (48, 48)
        roi = np.expand_dims(roi, axis=-1)  # Add channel dimension
        roi = np.expand_dims(roi, axis=0) / 255.0  # Add batch dimension and normalize

        # Predict the emotion
        emotion_preds = emotion_classifier. Predict (roi) [0]
        emotion_label = emotions[np.argmax(emotion_preds)]

        cv2.putText(frame, emotion_label, (x, y - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2, cv2.LINE_AA)

    return frame



# Capture video from the webcam
cap = cv2.VideoCapture(0)

# Check if the video capture was successful
if not cap.isOpened():
    print("Error: Could not open webcam.")
    exit()

while True:
    ret, frame = cap.read()
```

```
    if not ret:
        print("Error: Couldn't read frame from webcam.")
        break

    # Detect facial expressions in the frame
    frame = detect_expression(frame)

    # Display the frame
    cv2.imshow('Facial Expression Detection', frame)

    # Break the loop when 'q' is pressed
    If cv2.waitKey(1) & 0xFF == ord ('q'):
        break

# Release the video capture object and close all windows
cap.release()
cv2.destroyAllWindows()
```

6. BIBLIOGRAPHY

[1] Technical Paper on â_ Image Based Intelligent Attendance Logging System, IEEE Technical Paper by Hary Oktavianto, Hsu Gee-Sern Chung Sheng-Luen, June 30-July 2, 2012, Dalian,China.

[2] Z. Zhou, X. Chen, Y. â“C. Chung, Z. He, T. X. Han, and J. M. Keller, "Activity analysis,summarization, and visualization for indoor human activity monitoring,â IEEE Transactions on Circuits and Systems for VideoTechnology, Volume 18, Issue 11, pp. 1489- 1498, 2008.

[3] OpenCV. Available: http://sourceforge.net/projects/ opencvlibrary/.

[4] Activity zones for context-aware computing, Lecture Notes inComputer Science, Volume 2864/2003, pp. 90-106, 2003.

[5]International Journal of Innovative Research in Computer and communication Engineering (Vol.3 ,Issue 8,Aug 2015)(Automated Attendance System Using Face Recognition.)

[6]International Journal of Computer Applications (0975-8887) Volume 98-No.20,July 2014(Adjacency Matrix based Face Recognition Approach.)

[7]International Journal of Information Technology and KnowledgeManagement July December 2012, Volume 5, No. 2, pp. 361-3639(Face Recognition Techniques: Classification and Comparisons.)

[8]Face Recognition using Line Edge Map Vipin Kumar and Mohit Mehta Department of Electronics and communication, Punjab College of engineering and technology, India Accepted 01 January 2014, Available online 10 January 2014, Vol.2 (Jan/Feb 2014 issue)

[9]International Journal of Advanced Computer Research (ISSN (Print): 2249- 7277 ISSN (Online): 2277-7970) Volume-4 Number-4 Issue-17 December-2014 939 Study of Face Recognition Techniques Sangeeta Kaushik1*, R. B. Dubey2 and Abhimanyu Madan3 38

[10]Face Recognition Using Line Edge Map Yongsheng Gao, Member, IEEE, and Maylor K.H. Leung, Member, IEEE (IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 24, NO. 6, JUNE2002)

[11] C. Kotropoulos and I. Pitas, "Rule-based face detection in frontal views," Proc. Int'l Conf. Acoustics, Speech and Signal Processing, vol. 4, pp. 2537-2540, 1997.

[12] E. Saber and A.M. Tekalp, "Frontal-view face detection and facialfeature extraction using color, shape and symmetry based cost functions," Pattern Recognition Letters, vol. 17, no. 8, pp. 669-680, 1998.

[13] M. Turk, A. Pentland, Eigenfaces for Recognition, Journal of Cognitive Neurosicence, Vol. 3, No. 1, Win. 1991, pp. 71-86 .

[14] Discriminant analysis for recognition of human face images Kamran Etemad and Rama Chellappa

[15] Sanjay singh et.al, "A robust skin color based face detection algorithm", Tamkang Journal of Science and Engineering vol.6, no.4,pp227-234, 2003

[16] Jose M.Chaves-Gonzalez, Miguel A Vega-Rodriguez,Juan A.GomezPulido,Juan M.Sanchez-perez," Detecting skin in face recognition systems: A color spaces study", Digital signal processing, 20(2010) 806-823