Human Face Emotion In 3D

Theme-Based Project Report

Submitted in partial fulfilment of the

Requirements for the completion of Mini Project-III

BACHELOR OF ENGINEERING
VI Semester

IN

INFORMATION TECHNOLOGY

By

SHAIK MOHAMMED SAMEER 1602-20-737-168
JARUPLA ARUNA 1602-20-737-125
CHIMMI MAHESH 1602-20-737-143

Under the guidance of
B. Leelavathy
Assistant Professor

Department of Information Technology

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

VASAVI COLLEGE OF ENGINEERING (AUTONOMOUS)

(AFFILIATED TO OSMANIA UNIVERSITY)

HYDERABAD - 500 030

Department of Information Technology

DECLARATION BY CANDIDATES

We, **SHAIK MOHAMMED SAMEER, JARUPLA ARUNA, CHIMMI MAHESH,** bearing hall ticket number,**1602-20-737-168, 1602-20-737-125, 1602-20-737-143,** hereby declare that the project report entitled **"HUMAN FACE EMOTION IN 3D"** under the guidance of B. Leelavathy, Assistant Professor, Department of Information Technology, Vasavi College of Engineering, Hyderabad, is submitted in partial fulfillment of the requirement for the completion of the MiniProject-III, VI semester, **Bachelor of Engineering** in **Information Technology**

This is a record of bonafide work carried out by me and the results embodied in this project report has not been submitted to any institutes.

S. M. SAMEER
1602-20-737-168
J. ARUNA
1602-20-737-125
C. MAHESH
1602-20-737-143

VASAVI COLLEGE OF ENGINEERING (AUTONOMOUS)

(AFFILIATED TO OSMANIA UNIVERSITY)

HYDERABAD - 500 030

Department of Information Technology

BONAFIDE CERTIFICATE

This is to certify that the project entitled "**HUMAN FACE EMOTION IN 3D**" being submitted by **SHAIK MOHAMMED SAMEER, JARUPLA ARUNA, CHIMMI MAHESH,** bearing **1602-20-737-168, 1602-20-737-125, 1602-20-737-143,** in partial fulfillment of the requirements for the completion of MINI PROJECT-III, VI Semester of Bachelor of Engineering in Information Technology is a record of bonafide work carried out by them under my guidance.

Signature of the guide	**Signature of the external examiner**	**Signature of the HOD**
B. Leelavathy		**Dr. K. Ram Mohan Rao**
Assistant Professor		**HOD, IT**

ACKNOWLEDGEMENT

The satisfaction that accompanies that the successful completion of the project would not have been possible without the kind support and help of many individuals. We would like to extend my sincere thanks to all of them. We would like to take the opportunity to express our humble gratitude to **B. Leelavathy**, Associate Professor under whom we executed this project. We are grateful to her guidance, and constructive suggestions that helped us in the preparation of this project. Her constant guidance and willingness to share her vast knowledge made us understand this project and its manifestations in great depths and helped us to complete the assigned tasks. We would like to thank all faculty members andstaff of the Department of Information Technology for their generous help in various ways for the completion of this project. Finally, yet importantly, We would like to express our heartfelt thanks to our Head of the Department **Dr. K. Ram Mohan Rao** Sir.

ABSTRACT

Emotion detection, which is an effortless task for humans, is complex to perform on machines. The interaction between human beings and computers will be more natural if computers are able to perceive and respond to human non-verbal communication such as emotions. This project aims to develop a multi-modal emotion detection webapp that utilizes advanced computer vision, speech, and natural language processing (NLP) techniques to accurately identify and track human emotions in images, text and audio streams. The system will detect and classify multiple emotions simultaneously do image rendering. This project uses CNN to detect emotion from image and RNN to detect emotion from text and speech. It can also generate a 3D model of the image using BFM model and aspose 3d. The system can adapt to individual differences in expressing emotions, ensure privacy and security, and have potential applications in various fields, including healthcare, education, and entertainment.

TABLE OF CONTENTS:

CHAPTER 1

1.1 PURPOSE

Nowadays, more and more intelligent systems are using emotion recognition models to improve their interaction with humans. This is important, as the systems can adapt their responses and behavioral patterns according to the emotions of the humans and make the interaction more *natural*.

Modern 3D modeling provides a level of design depth that rough sketches or 2D designs cannot, such as improved control over details. It also lets engineers explore the physical aspects of a design without surrendering to physical limitations.

1.2 INTENDED AUDIENCE

The intended audience for this project is everyone who wants to detect emotions and do some work on it. This technology can be applied to fields like **security, biometrics, law enforcement, etc., for tracking and surveillance purposes.** It proposes a set of research scenarios of emotion recognition applications in the following domains: **software engineering, website customization, education, and gaming.**

1.3 PRODUCT SCOPE

Now we have designed a website. We are planning to develop an application.

It offers tremendous scope to **human computer interaction, robotics, health care, biometric security and behavioral modeling**.

1.4 PROBLEM DEFINITION

With the recent advancement of computer vision and AI/ML techniques, identification of human faces is no longer a challenging task. However, creating a human face with captured expressions, movements, voice and other features in real time videos is still a challenging task. Design a prototype system with advance techniques of image recognition and AI/ML to identify humans in real time video. The prototype system must render the image of identified person in the video such that the face orientation changes dynamically with the body movement. Effects like face expressions, movements must be captured effectively to give feeling of real human face.

CHAPTER 2

2.1 LITERATURE SURVEY

TensorFlow: A system for large-scale machine learning

Martín Abadi **--** TensorFlow is a machine learning system that operates at large scale and in heterogeneous environments. TensorFlow uses dataflow graphs to represent computation, shared state, and the operations that mutate that state. It maps the nodes of a dataflow graph across many machines in a cluster, and within a machine across multiple computational devices, including multicore CPUs, general-purpose GPUs, and custom designed ASICs known as Tensor Processing Units (TPUs). This architecture gives flexibility to the application developer: whereas in previous "parameter server" designs the management of shared state is built into the system, TensorFlow enables developers to experiment with novel optimizations and training algorithms. TensorFlow supports a variety of applications, with particularly strong support for training and inference on deep neural networks. Several Google services use TensorFlow in production, we have released it as an open-source project, and it has become widely used for machine learning research. In this paper, we describe the TensorFlow dataflow model in contrast to existing systems, and demonstrate the compelling performance that TensorFlow achieves for several real-world applications.

Deep Learning Approaches for Facial Emotion Recognition: A Case Study on FER-2013

Ioannis Hatzilygeroudis **--** Emotions constitute an innate and important aspect of human behavior that colors the way of human communication. The accurate analysis and interpretation of the emotional content of human facial expressions is essential for the deeper understanding of human behavior. Although a human can detect and interpret faces and facial expressions naturally, with little or no effort, accurate and robust facial expression recognition by computer systems is still a great challenge. The analysis of human face characteristics and the recognition of its emotional states are considered to be very challenging and difficult tasks. The main difficulties come from the non-uniform nature of human face and variations in conditions such as lighting, shadows, facial pose and orientation. Deep learning approaches have been examined as a stream of methods to achieve robustness and provide the necessary scalability on new type of data. In this work, we examine the performance of two known deep learning approaches (GoogLeNet and AlexNet) on facial expression recognition, more specifically the recognition of the existence of emotional content.

A study on Image Classification based on Deep Learning and Tensorflow

Mohd Azlan Abu **--**This research study about image classification by using the deep neural network (DNN) or also known as Deep Learning by using framework TensorFlow. Python is used as a programming language because it comes together with TensorFlow framework. The input data mainly focuses in flowers category which there are five (5) types of flowers that have been used in this paper. Deep neural network (DNN) has been choosing as the best option for the training process because it produced a high percentage of accuracy. Results are discussed in terms of the accuracy of the image classification in percentage. Roses get 90.585% and same goes to another type of flowers where the average of the result is up to 90% and above.

LSTM-based Text Emotion Recognition Using Semantic and Emotional Word Vectors

Chung-Hsien Wu --This study proposes a long-short term memory (LSTM)-based approach to text emotion recognition based on semantic word vector and emotional word vector of the input text. For each word in an input text, the semantic word vector is extracted from the word2vec model. Besides, each lexical word is projected to all the emotional words defined in an affective lexicon to derive an emotional word vector. An autoencoder is then adopted to obtain the bottleneck features from the emotional word vector for dimensionality reduction. The autoencoder bottleneck features are then concatenated with the features in the semantic word vector to form the final textual features for emotion recognition. Finally, given the textual feature sequence of the entire sentence, the LSTM is used for emotion recognition by modeling the contextual emotion evolution of the input text. For evaluation, the NLPCC-MHMC-TE database containing seven emotion categories: anger, boredom, disgust, anxiety, happiness, sadness, and surprise was constructed and used. Five-fold cross-validation was employed to evaluate the performance of the proposed method.

Speech Emotion Recognition using Machine Learning

Navuluri Sainath --The aim of the paper is to detect the emotions which are elicited by the speaker while speaking. Emotion Detection has become a essential task these days. The speech which is in fear, anger, joy have higher and wider range in pitch whereas have low range in pitch. Detection of speech is useful in assisting human machine interactions. Here we are using different classification algorithms to recognize the emotions , Support Vector Machine , Multi layer perception, and the audio feature MFCC, MEL, chroma, Tonnetz were used. These models have been trained to recognize these emotions (Calm, neutral, surprise, happy, sad, angry, fearful, disgust). We got an accuracy of 86.5% and testing it with the input audio we get the same.

2D-to-3D image conversion by learning depth

Meng Wang --Among 2D-to-3D image conversion methods, those involving human operators have been most successful but also time-consuming and costly. Automatic methods, that typically make use of a deterministic 3D scene model, have not yet achieved the same level of quality as they often rely on assumptions that are easily violated in practice. In this paper, we adopt the radically different approach of “learning” the 3D scene structure. We develop a simplified and computationally-efficient version of our recent 2D-to-3D image conversion algorithm. Given a repository of 3D images, either as stereopairs or image+depth pairs, we find k pairs whose photometric content most closely matches that of a 2D query to be converted. Then, we fuse the k corresponding depth fields and align the fused depth with the 2D query. Unlike in our original work, we validate the simplified algorithm quantitatively on a Kinect-captured image+depth dataset against the Make3D algorithm. While far from perfect, the presented results demonstrate that online repositories of 3D content can be used for effective 2D-to-3D image conversion.

2.2 RELATED WORK-

Many of today's new and innovative artificial intelligence (AI) applications use CNN-based deep learning technology to capture, interpret, and analyze various kinds of video, audio, and text data. A convolutional neural network is a type of deep learning algorithm that is most often applied to analyze and learn visual features from large amounts of data. While primarily used for image-related AI applications, CNNs can be used for other AI tasks, including natural language processing and in recommendation engines.

CHAPTER 3

3.1 Software Requirements –

Software requirements –

- Python >=3.5
- PyCharm IDE
- GPU
- Tensorflow

Front end:

HTML & CSS:

- HTML is the markup language used to structure the content of web pages, defining elements like headings, paragraphs, links, and images.
- CSS is a stylesheet language used to style the HTML elements, controlling aspects such as layout, colors, fonts, and spacing.
- HTML and CSS work together to create visually appealing and well-structured web pages, with HTML providing the content and structure, and CSS handling the presentation and styling.

Back-end:

CNN :

A Convolutional Neural Network (CNN) is a type of artificial neural network commonly used for image and video analysis. CNNs are designed to process data with a grid-like topology, such as images or time-series data. A CNN typically consists of multiple layers, including convolutional layers, pooling layers, and fully connected layers. In the convolutional layers, filters are applied to the input data to extract features. The pooling layers down sample the output of the convolutional layers to reduce the spatial dimensions of the input data. Finally, the fully connected layers take the output of the convolutional and pooling layers and produce a prediction or classification. CNN have several advantages over traditional image processing techniques, as they can learn to automatically extract features from raw data, reducing the need for manual feature engineering. They have proven to be highly effective for a wide range of tasks, including image classification, object detection, and image segmentation.

Tensor flow:

- TensorFlow is an open-source machine learning framework developed by Google.
- It provides a comprehensive ecosystem of tools, libraries, and resources for building and deploying machine learning models.
- TensorFlow supports both deep learning and traditional machine learning algorithms, offering flexibility, scalability, and high-performance computation on a variety of hardware platforms. It is widely used in research and industry for tasks such as image and speech recognition, natural language processing, and recommendation systems.

3.2 Hardware Requirements –

- Laptop with camera
- GPU is preferable or else high CPU
- Microphone Enable System

CHAPTER 4 –

4.1 USE CASE DIAGRAM

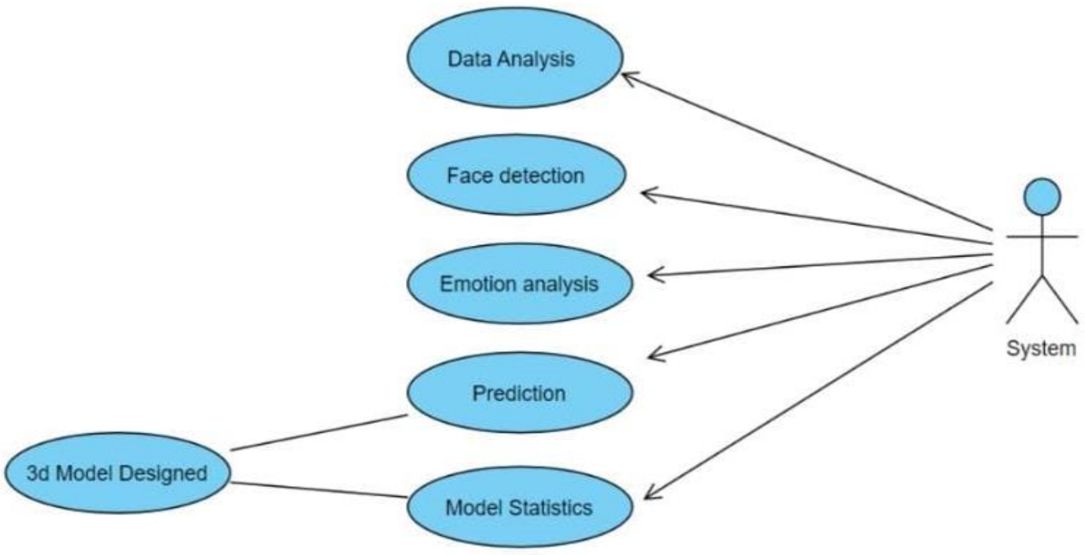


Fig 1: Use Case Diagram For Image Emotion Detection

SPEECH EMOTION **TEXT EMOTION**

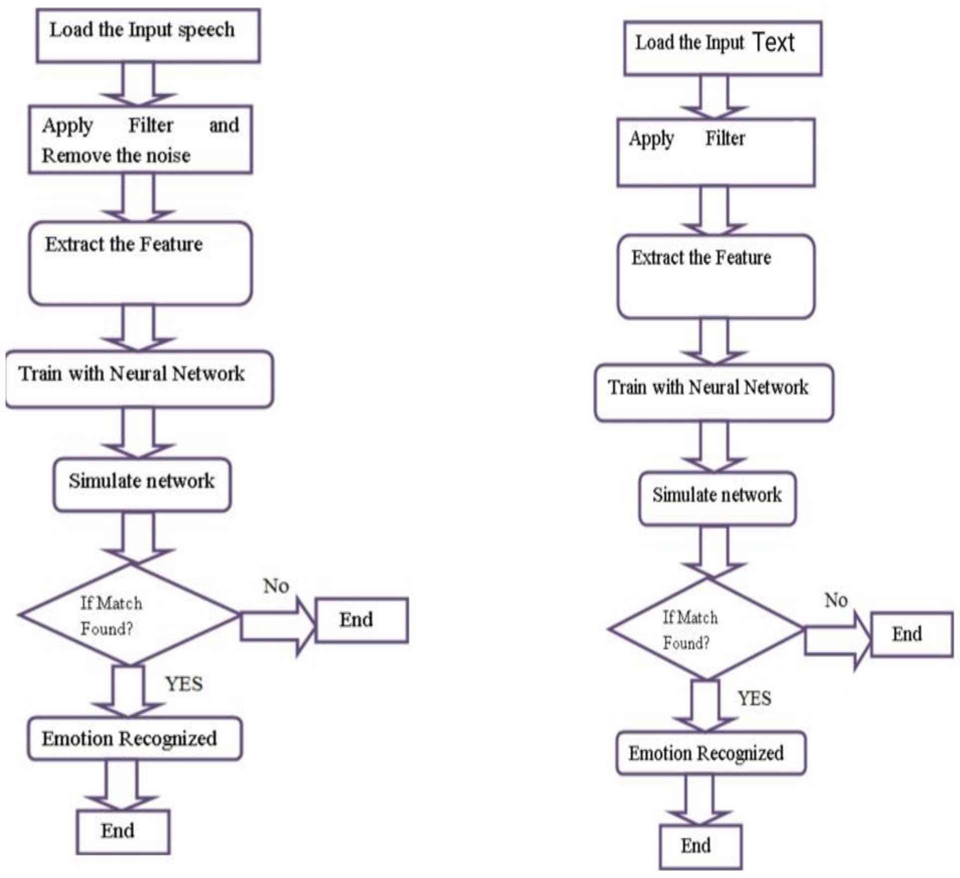


Fig 2: Flow Chart for speech and text emotion detection

4.2 IMPLEMENTATION-

4.2.1- Modules --

The various intents the face and signature classifier is trained on are –

4.2.1.1 Upload image –

If no image is uploaded then a page will be redirected to the same screen.

4.2.1.2– detect emotions–

User has to upload image then the system will predict the emotion using trained model.

4.2.1.3– 3D Modelling–

After predicting emotion the image will be send to BFM model which will make it into 3d.

4.2.2.– Algorithm used –

CNN: A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data. There are other types of neural networks in deep learning, but for identifying and recognizing objects, CNNs are the network architecture of choice.

>> Data Analysis:

- Image converting into grayscale using CV2
- Resizing gray Image to 48x48 pixels
- Reshaping image using Numpy

>> Face Detection:

- Detecting faces using “haarcascade_frontalface_default.xml”.
- Cropping image to face shape.

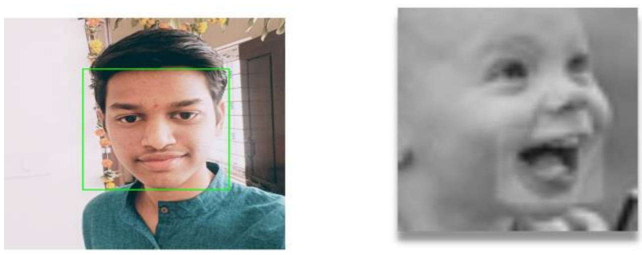


Fig 3: Image cropping and grayscale(48x48) generator

>> Emotion Analysis:

- Loading model structure using json file.
- Predicting emotion using CNN trained model (model.h5)

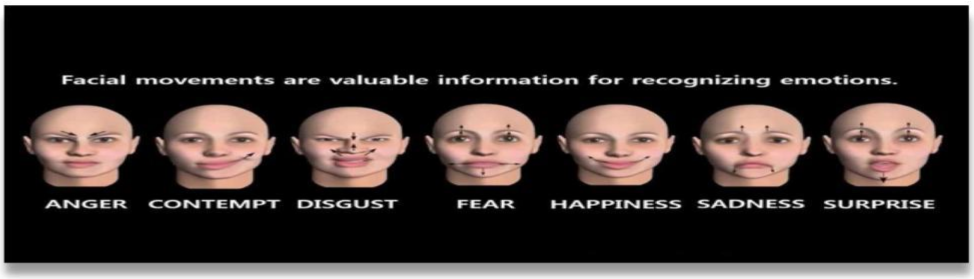

Facial movements are valuable information for recognizing emotions.
ANGER
CONTEMPT
DISGUST
FEAR
HAPPINESS
SADNESS
SURPRISE

Fig 4: Facial movements

>> 3D Model :

- Loading BFM model for making 3d face using torch, aspose-3d and mediapipe.

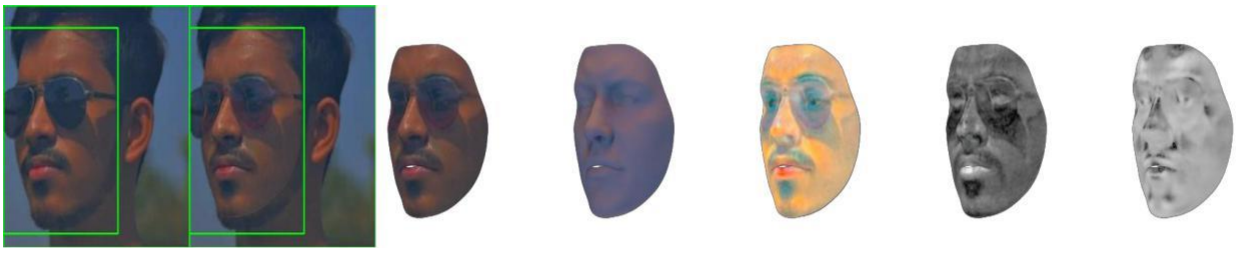


Fig 5: 3D Mesh images

CHAPTER 5 –

5.1 CODE

APP.PY

```
from flask import Flask,render_template,request,redirect
from os.path import join, dirname, realpath
from werkzeug.utils import secure_filename
import os
import numpy as np
import cv2
from keras.preprocessing import image
from keras.models import load_model,model_from_json
import tensorflow as tf
from scipy.stats import zscore
import pandas as pd
import librosa
import pickle
from  keras.preprocessing.text import Tokenizer
from keras.utils import pad_sequences
from optimizer import Optimizer
from config import Config
UPLOAD_FOLDER = join(dirname(realpath(__file__)), 'static/')
ALLOWED_EXTENSIONS = {'jfif', 'png', 'jpg', 'jpeg'}

def mel_spectrogram(y, sr=16000, n_fft=512, win_length=256, hop_length=128,
window='hamming', n_mels=128, fmax=4000):
    mel_spect = np.abs(librosa.stft(y, n_fft=n_fft, window=window, win_length=win_length,
hop_length=hop_length)) ** 2
        mel_spect = librosa.feature.melspectrogram(S=mel_spect, sr=sr, n_mels=n_mels,
fmax=fmax)
    mel_spect = librosa.power_to_db(mel_spect, ref=np.max)
    return mel_spect

def frame(x, win_step=128, win_size=64):
    nb_frames = 1 + int((x.shape[2] - win_size) / win_step)
    frames = np.zeros((x.shape[0], nb_frames, x.shape[1], win_size)).astype(np.float32)
    for t in range(nb_frames):
       frames[:,t,:,:] = np.copy(x[:,:,(t * win_step):(t * win_step + win_size)]).astype(np.float32)
    return frames

def get_key_speech(value):
    dictionary={'neutral':0,'calm':1,'happy':2,'sad':3,'angry':4,'fear':5,'disgust':6,'surprise':7}
    for key,val in dictionary.items():
          if (val==value):
            return key

def get_key_text(value):
    dictionary={'happy':0,'angry':1,'love':2,'sad':3,'fear':4,'surprise':5}
    for key,val in dictionary.items():
          if (val==value):
            return key

def recommend(value):
    if (value=='happy' or value=='Happy'):
        ans="Watch a funny movie or TV show to keep the positive emotions flowing.\n\
Meet up with friends and family to spend quality time together.\n\
Plan a fun activity, such as going to a theme park or playing a game outdoors.\n\
Listen to upbeat music and dance to your favorite songs.\n\
Practice gratitude by writing down three things you are thankful for each day."
    elif (value=='angry' or value=='Angry'):
        ans="Take a deep breath and count to 10 to calm down before reacting.\n\
Go for a run or engage in physical activity to release pent-up energy.\n\
Identify the source of your anger and express your feelings in a constructive manner.\n\
Practice relaxation techniques, such as meditation or deep breathing.\n\
Consider seeking help from a therapist or counselor if your anger is interfering with your daily
life."
```

```
    elif (value=='fear' or value=='Fear'):
          ans="Identify the source of your fear and challenge negative thoughts with positive ones.\n\
Practice relaxation techniques, such as deep breathing or visualization exercises.\n\
Seek support from a trusted friend or family member to talk about your fears.\n\
Engage in activities that make you feel safe and secure, such as spending time with a pet or in nature.\n\
Consider seeking help from a therapist or counselor if your fears are interfering with your daily life."
    elif (value=='disgust' or value=='Disgust'):
        ans="Identify any underlying beliefs or values that may be contributing to your disgust.\n\
Consider whether there may be cultural or societal influences shaping your reaction.\n\
If appropriate, try to reframe the situation or object in a more positive light to reduce your disgust.\n\
Experiment with different coping strategies, such as humor or cognitive reappraisal, to help regulate your emotions."
    elif (value=='surprise' or value=='Surprise'):
        ans="Negative:\n\
Reach out to a trusted friend or family member to talk about your feelings.\n\
Take a deep breath and try to stay calm.\n\
Seek professional help, such as therapy or counseling, if your surprise is causing distress or interfering with your daily life.\n\
Positive:\n\
Express gratitude for the surprise and the positive feelings it brings.\n\
Share your excitement with friends and family to spread the positive vibes.\n\
Reflect on the surprise and the positive feelings it brings to cultivate a sense of happiness and contentment."
    elif (value=="love" or value=="Love"):
        ans="Spend quality time with loved ones to nurture your relationships.\n\
Practice acts of kindness, such as writing a thoughtful note or doing something nice for someone.\n\
Do something you enjoy to boost your mood and show yourself some love.\n\
Give back to your community by volunteering or donating to a charitable cause.\n\
Practice self-care by taking care of your physical and emotional needs."
    elif (value=='sad' or value=='Sad'):
        ans="Take a break and allow yourself to feel your emotions without judgment.\n\
Write down your thoughts and feelings in a journal to process them.\n\
Reach out to a trusted friend or family member to talk about your feelings.\n\
Engage in self-care activities, such as taking a relaxing bath or medication.\n\
Seek professional help, such as therapy or counseling, if your feelings persist."
    elif (value=='calm' or value=='Calm'):
          ans="Take a break from technology and spend time in nature to connect with your surroundings and promote a sense of calm.\n\
Practice gratitude by focusing on the positive aspects of your life and expressing appreciation for the people and experiences that bring you joy.\n\
Use positive self-talk to reinforce feelings of calm and inner peace.\n\
Consider engaging in hobbies or activities that bring you peace and tranquility.\n\
Connect with loved ones and spend time in meaningful conversations or shared experiences."
```

```
    elif (value=='Neutral' or value=='neutral'):
        ans="Consider trying out a new hobby or activity that you've always been interested in but haven't had the chance to pursue.\n\
Take some time to reflect on your current goals and priorities, and make adjustments as needed.\n\
Connect with friends or loved ones and make plans to spend quality time together.\n\
Practice mindfulness or meditation to increase your awareness and presence in the moment.\n\
Engage in regular physical exercise or movement to boost your mood and energy levels."
    else:
        ans=" "
    return ans

def predictText(value):
    Textmodel = model_from_json(open("text_model.json", "r").read())
    Textmodel.load_weights('text_model.h5')
    with open('tokenizer.pickle','rb') as handle:
        tokenizer = pickle.load(handle)
    sentence_lst=[]
    sentence_lst.append(value)
    sentence_seq=tokenizer.texts_to_sequences(sentence_lst)
    sentence_padded=pad_sequences(sentence_seq,maxlen=80,padding='post')
    ans=Textmodel.predict(sentence_padded)
    ans1 = np.argmax(ans, axis=1)
    k=get_key_text(ans1)
    return k

app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 1
app.secret_key = 'sameer'


@app.route('/')
def main():
    return render_template('index.html')


@app.route('/Imagepredict',  methods=['POST', 'GET'])
def uploadFile():
    if request.method == 'POST':
        if 'uploaded-file' not in request.files:
            return redirect(request.url)
        uploaded_img = request.files['uploaded-file']
        if uploaded_img.filename == '':
            return redirect(request.url)

        uploaded_img.save('static/file.jpg')
        img1 = cv2.imread('static/file.jpg')
        gray = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
```

```
                cascade = cv2.CascadeClassifier(os.path.join(cv2.data.haarcascades,
'haarcascade_frontalface_default.xml'))
        faces = cascade.detectMultiScale(gray, 1.1, 3)
        for x,y,w,h in faces:
            cv2.rectangle(img1, (x,y), (x+w, y+h), (0,255,0), 2)
            cropped = img1[y:y+h, x:x+w]
        cv2.imwrite('static/after.jpg', img1)
        try:
            cv2.imwrite('static/cropped.jpg', cropped)
        except:
            pass
        try:
            image = cv2.imread('static/cropped.jpg', 0)
        except:
            image = cv2.imread('static/file.jpg', 0)
        image = cv2.resize(image, (48,48))
        image = image/255.0
        image = np.reshape(image, (1,48,48,1))
        Imagemodel = model_from_json(open("emotion_model1.json", "r").read())
        Imagemodel.load_weights('model.h5')
        prediction = Imagemodel.predict(image)
        label_dict = {0:'Angry',1:'Disgust',2:'Fear',3:'Happy',4:'Neutral',5:'Sad',6:'Surprise'}
        prediction = list(prediction[0])
        img_index = prediction.index(max(prediction))
        final_prediction=label_dict[img_index]
        rec=recommend(final_prediction)
        outimg=final_prediction+".gif"
        return render_template('Imagepredict.html', data=final_prediction,im=outimg,inf=rec)


@app.route('/contact')
def main2():
    return render_template('contact.html')


@app.route('/about')
def main3():
    return render_template('about.html')


@app.route('/run-code', methods=['POST'])
def run_code():
    Imagemodel = model_from_json(open("emotion_model1.json", "r").read())
    Imagemodel.load_weights('model.h5')
    emotion_dict = {0: "Angry", 1: "Disgusted", 2: "Fearful", 3: "Happy", 4: "Neutral", 5: "Sad",
6: "Surprised"}
    cap = cv2.VideoCapture(0)
    while True:
        ret, frame = cap.read()
        if not ret:
            return render_template('index.html')
        facecasc = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = facecasc.detectMultiScale(gray,scaleFactor=1.3, minNeighbors=5)
        for (x, y, w, h) in faces:
            cv2.rectangle(frame, (x, y-50), (x+w, y+h+10), (255, 0, 0), 2)
            roi_gray = gray[y:y + h, x:x + w]
           cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray, (48, 48)), -1), 0)
            prediction = Imagemodel.predict(cropped_img)
            maxindex = int(np.argmax(prediction))
                              cv2.putText(frame,     emotion_dict[maxindex],     (x+20,     y-60),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0, 255), 2, cv2.LINE_AA)
        cv2.imshow('Video', cv2.resize(frame,(1600,960),interpolation = cv2.INTER_CUBIC))
        if cv2.waitKey(1) & 0xFF == ord('q'):
            return render_template('index.html')


@app.route('/TextEmo')
def textEmo():
    return render_template('TextEmo.html')


@app.route('/textPred',methods=['post'])
def textPred():
    sentence=request.form['input1']
    k=predictText(sentence)
    rec=recommend(k)
    out=k+".gif"
    return render_template('textPred.html',data=k,outimg=out,sent=sentence,inf=rec)


@app.route('/SpeechEmotion',methods=['POST', 'GET'])
def speechEmo():
     if request.method == 'POST':
        if 'uploaded-audio' not in request.files:
             return redirect(request.url)
        uploaded_aud = request.files['uploaded-audio']
        if uploaded_aud.filename == '':
             return redirect(request.url)

        uploaded_aud.save('static/audio.wav')
        Speechmodel = model_from_json(open("speech_model_json.json", "r").read())
        Speechmodel.load_weights('speech_emo_weights.h5')
        s = []
        sample_rate = 16000
        max_pad_len = 49100
        win_ts = 128
        hop_ts = 64
        path_="static/audio.wav"
        X, sample_rate = librosa.load(path_,duration=3,offset=0.5)
        sample_rate = np.array(sample_rate)
        y = zscore(X)
        if len(y) < max_pad_len:
            y_padded = np.zeros(max_pad_len)
            y_padded[:len(y)] = y
            y = y_padded
        elif len(y) > max_pad_len:
            y = np.asarray(y[:max_pad_len])
        s.append(y)
        mel_spect = np.asarray(list(map(mel_spectrogram, s)))
        x = frame(mel_spect, hop_ts, win_ts)
        x = x.reshape(x.shape[0], x.shape[1] , x.shape[2], x.shape[3], 1)
        preds = Speechmodel.predict(x)
        preds=preds.argmax(axis=1)
        preds=get_key_speech(preds)
        rec=recommend(preds)
        outputimg=preds+".gif"
        return render_template('SpeechEmotion.html', data=preds,im=outputimg ,inf=rec)


@app.route('/aud1')
def aud1():
    return render_template('aud1.html')


@app.route('/predict',  methods=['POST', 'GET'])
def predict():
    if request.method == 'POST':
        if 'upload-3dfile' not in request.files:
            return redirect(request.url)
        uploaded_img = request.files['upload-3dfile']
        if uploaded_img.filename == '':
            return redirect(request.url)
        uploaded_img.save('static/3dfile.jpg')
    config = Config()
    config.fillFromDicFile('C:/Users/user/Desktop/nextface/optimConfig.ini')
    config.device = 'cpu'
    config.path = 'C:/Users/user/Desktop/nextface/baselMorphableModel/'
    outputDir = 'C:/Users/user/Desktop/nextface/static/output/'
    optimizer = Optimizer(outputDir ,config)
    imagePath = 'C:/Users/user/Desktop/nextface/static/3dfile.jpg'
    optimizer.run(imagePath)
    return render_template('predict.html')


@app.route('/upload-audio', methods=['POST', 'GET'])
def upload_audio():
    audio_file = request.files['audio']
    audio_file.save('static/audio1.wav')
    pass


@app.route('/RealTimeSpeech', methods=['POST', 'GET'])
def realtimespeech():
    Speechmodel1 = model_from_json(open("speech_model_json.json", "r").read())
    Speechmodel1.load_weights('speech_emo_weights.h5')
    s = []
    sample_rate = 16000
    max_pad_len = 49100
    win_ts = 128
    hop_ts = 64
    path1="static/audio1.wav"
    X, sample_rate = librosa.load(path1,duration=3,offset=0.5)
    sample_rate = np.array(sample_rate)
    y = zscore(X)
    if len(y) < max_pad_len:
        y_padded = np.zeros(max_pad_len)
        y_padded[:len(y)] = y
        y = y_padded
    elif len(y) > max_pad_len:
        y = np.asarray(y[:max_pad_len])
    s.append(y)
    mel_spect = np.asarray(list(map(mel_spectrogram, s)))
    x = frame(mel_spect, hop_ts, win_ts)
    x = x.reshape(x.shape[0], x.shape[1] , x.shape[2], x.shape[3], 1)
    preds = Speechmodel1.predict(x)
    preds=preds.argmax(axis=1)
    preds=get_key_speech(preds)
    rec=recommend(preds)
    outputimg=preds+".gif"
    return render_template('RealTimeSpeech.html', data=preds,im=outputimg,inf=rec)

if __name__ == "__main__":
    app.run(debug=True)
```

Index.html:

```
{% extends 'base.html' %}
{% block body %}
<script>
  const button = document.getElementById('run-button');
  button.addEventListener('click', () => {
    fetch('/run-code', {method: 'POST'})
        .then(response => response.text())
        .then(text => console.log(text));
  });


  </script>

<style>
    .float-container {
      padding: 100px;
    }
    .float-child {
      padding: 50px;
      float: left;
    }
    .card {

      width: 100%;
    }
    .card-img-top {
      border: 3px solid black;
    }
    .box {
      width: 520px;
      height: 580px;
      background: lawngreen;
      border: 3px solid black;
      text-align: center;
      padding: 30px;
    }
    .btn {
      display: block;
      width: 100px;
      height: 40px;
      background: darkmagenta;
      color: #fff;
      position: relative;
      border-radius: 3px;
      border: 0;
      transition: all 0.3s ease-in-out;
      font-size: 14px;
    }
    .btn:hover {
      background: gold;
      box-shadow: 0 3px 0 0 deeppink;
    }
    .btn1 {
      left: 320px;
      top: -40px;
    }
    .btn2 {
      top: -5px;
      left: 180px;
    }
  </style>

<section id="hero">
  <div class="hero container">
    <div>
      <h5>Welcome to <span></span></h5>
      <h5>Multi Modal 3D Emotion<span></span></h5>
      <h5>A WebApp to detect emotions and make a 3d model <span></span></h5>
      <a href="#carouselExampleIndicators" type="button" class="cta">Let's Start</a>
    </div>
  </div>
</section>

<div class="container-fluid">
     <div   id="carouselExampleIndicators"   class="carousel   slide   carousel-fade"   data-
ride="carousel" data-interval="1000">
    <div class="carousel-inner">
      <div class="carousel-item active">
            <img  class="d-block  w-100"  src="{{  url_for('static',  filename='home1.webp')  }}"
alt="First slide" width=auto
          height="400">
        <h4 style="color:black;text-align: center;">Valid Emotions</h4>
      </div>
      <div class="carousel-item">
             <img  class="d-block  w-100"  src="{{  url_for('static',  filename='home2.jpeg')  }}"
alt="Second slide" width=auto
          height="400">
        <h4 style="color:black; text-align: center;">Emotion Detection</h4>
      </div>
      <div class="carousel-item">
          <img class="d-block w-100" src="{{ url_for('static', filename='11.jfif') }}" alt="Third
slide" width=auto
          height="400">
        <h4 style="color:black; text-align: center;">3D Conversion</h4>
      </div>
    </div>
     <a  class="carousel-control-prev"  href="#carouselExampleIndicators"  role="button"  data-
slide="prev">
      <span class="carousel-control-prev-icon" aria-hidden="true"></span>
      <span class="sr-only">Previous</span>
    </a>
     <a  class="carousel-control-next"  href="#carouselExampleIndicators"  role="button"  data-
slide="next">
      <span class="carousel-control-next-icon" aria-hidden="true"></span>
```

```
          <span class="sr-only">Next</span>
        </a>
      </div>
    </div>

    <div class="float-container">
      <div class="float-child">
        <div class="box">
          <div class="card">
            <div class="view overlay">
              <img
                class="card-img-top"
                src="{{ url_for('static', filename='img.gif') }}"
                alt="Card image cap"
                width="450"
                height="400"
              />
              <h2 class="card-title">Image Emotion</h2>
              <form
                method="POST"
                enctype="multipart/form-data"
                action="{{url_for('uploadFile')}}"
              >
                <input
                  type="file"
                  class="upload"
                  id="myFile"
                  name="uploaded-file"
                />
                <input type="submit" class="btn btn1 submit" value="Submit" />
              </form>
            </div>
          </div>
        </div>
      </div>

      <div class="float-child">
        <div class="box">
          <div class="card">
            <div class="view overlay">
              <img
                class="card-img-top"
                src="{{ url_for('static', filename='camera.gif') }}"
                alt="Card image cap"
                width="450"
                height="400"
              />
              <h2 class="card-title">Real Time Video Emotion</h2>
```

```html
          <form
            method="POST"
            enctype="multipart/form-data"
            action="{{url_for('run_code')}}"
          >
            <input type="submit" class="btn btn2 submit" id="run-button" value="Run" />
          </form>
        </div>
      </div>
    </div>
  </div>
</div>

<div class="float-container">
  <div class="float-child">
    <div class="box">
      <div class="card">
        <div class="view overlay">
          <img
            class="card-img-top"
            src="{{ url_for('static', filename='sppech.webp') }}"
            alt="Card image cap"
            width="450"
            height="400"
          />
          <h2 class="card-title">Speech Emotion</h2>
          <form
            method="POST"
            enctype="multipart/form-data"
            action="{{url_for('speechEmo')}}"
          >
            <input
              type="file"
              class="upload"
              id="myFile"
              name="uploaded-audio"
            />
            <input type="submit" class="btn btn1 submit" value="Submit" />
          </form>
        </div>
      </div>
    </div>
  </div>

  <div class="float-child">
    <div class="box">
      <div class="card">
        <div class="view overlay">
```

```
          <img
            class="card-img-top"
            src="{{ url_for('static', filename='text.webp') }}"
            alt="Card image cap"
            width="450"
            height="400"
          />
          <h2 class="card-title">Text Emotion</h2>
          <form
            action="{{url_for('textEmo')}}"
          >
            <input type="submit" class="btn btn2 submit" value="Run" />
          </form>
        </div>
      </div>
    </div>
  </div>
</div>

<div class="float-container">
  <div class="float-child">
    <div class="box">
      <div class="card">
        <div class="view overlay">
          <img
            class="card-img-top"
            src="{{ url_for('static', filename='list.gif') }}"
            alt="Card image cap"
            width="450"
            height="400"
          />
          <h2 class="card-title">Real Time Speech Emotion</h2>
          <form
            action="{{url_for('aud1')}}"
          >
            <input type="submit" class="btn btn2 submit" value="Start" />
          </form>
        </div>
      </div>
    </div>
  </div>

  <div class="float-child">
    <div class="box">
      <div class="card">
        <div class="view overlay">
          <img
            class="card-img-top"
            src="{{ url_for('static', filename='3d.gif') }}"
            alt="Card image cap"
            width="450"
            height="400"
          />
          <h2 class="card-title">3D Model</h2>
          <form
            method="POST"
            enctype="multipart/form-data"
            action="{{url_for('predict')}}"
          >
            <input
              type="file"
              class="upload"
              id="myFile"
              name="upload-3dfile"
            />
            <input type="submit" class="btn btn1 submit" value="Run" />
          </form>
        </div>
      </div>
    </div>
  </div>
</div>


{% endblock body %}
```

TextPred.html:

```
{% extends 'base.html' %}
{% block body %}
<head>
    <style>
        @import url('https://fonts.googleapis.com/css2?family=Courier+Prime&display=swap');
body {
  margin: 0px;
  padding: 0px;
  font-family: "Courier Prime", monospace;
  background-color: #fdfdfe;
}
header {
  width: 100%;
  justify-content: center;
  align-items: top;
  margin: 0;
}
header h1 {
  font-size: 50px;
  font-weight: 500;
  border-right: 4px solid #000;
  animation: cursor 1s infinite step-end, typing 15s infinite steps(22);
  white-space: nowrap;
  overflow: hidden;
  margin-left: 30px;
}
          display: block;
          width: 400px;
          height: 400px;
          margin-left: auto;
          margin-right: auto;
        }
        .boxb{
          margin: auto;
          width: 50%;
          padding: 10px;
          border: 3px dashed black;
        }
        .recommend{
          padding: 10px;
          border: 3px dashed black;
        }
    </style>
</head>
<body>
    <div class="boxb">
        <header>
        <h1>Text Emotion Detection</h1>
        </header>
                                    <p><h2       style="color:darkblue">Input       Sentence       :       <span
style="color:slateblue">{{sent}}</span></h2></p>
        <h2 style="color:orange ;text-align: center;">Predicted Emotion : {{data}} </h2>
        <img src="{{ url_for('static', filename=outimg)}}" class="img1">
        <div class="recommend">
          <h2 style="color:red ;text-align: center;">Recommendations : </h2>
          <pre style="font-size: medium;">{{inf}}</pre>
        </div>
    </div>
</body>
{% endblock body %}
```

5.2 OUTPUT

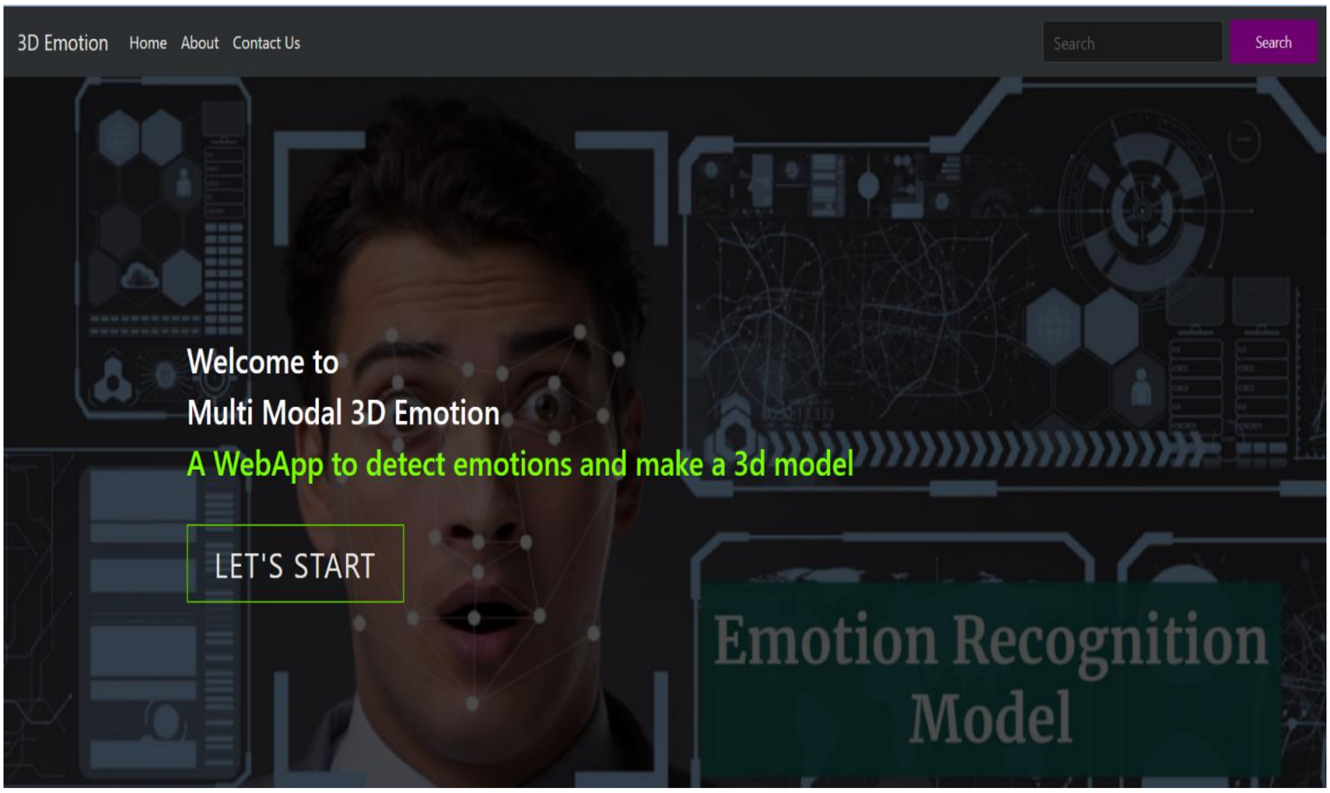
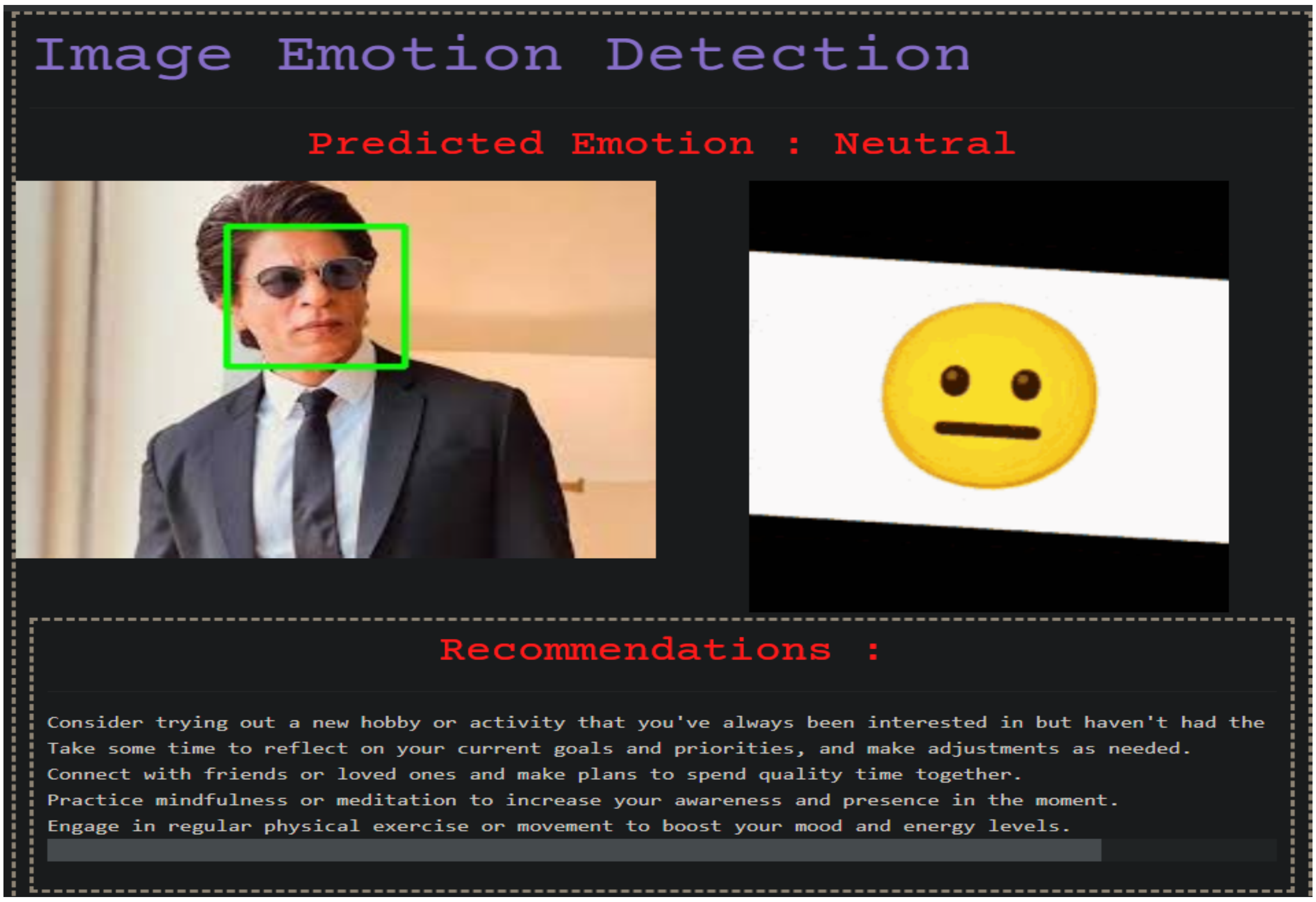
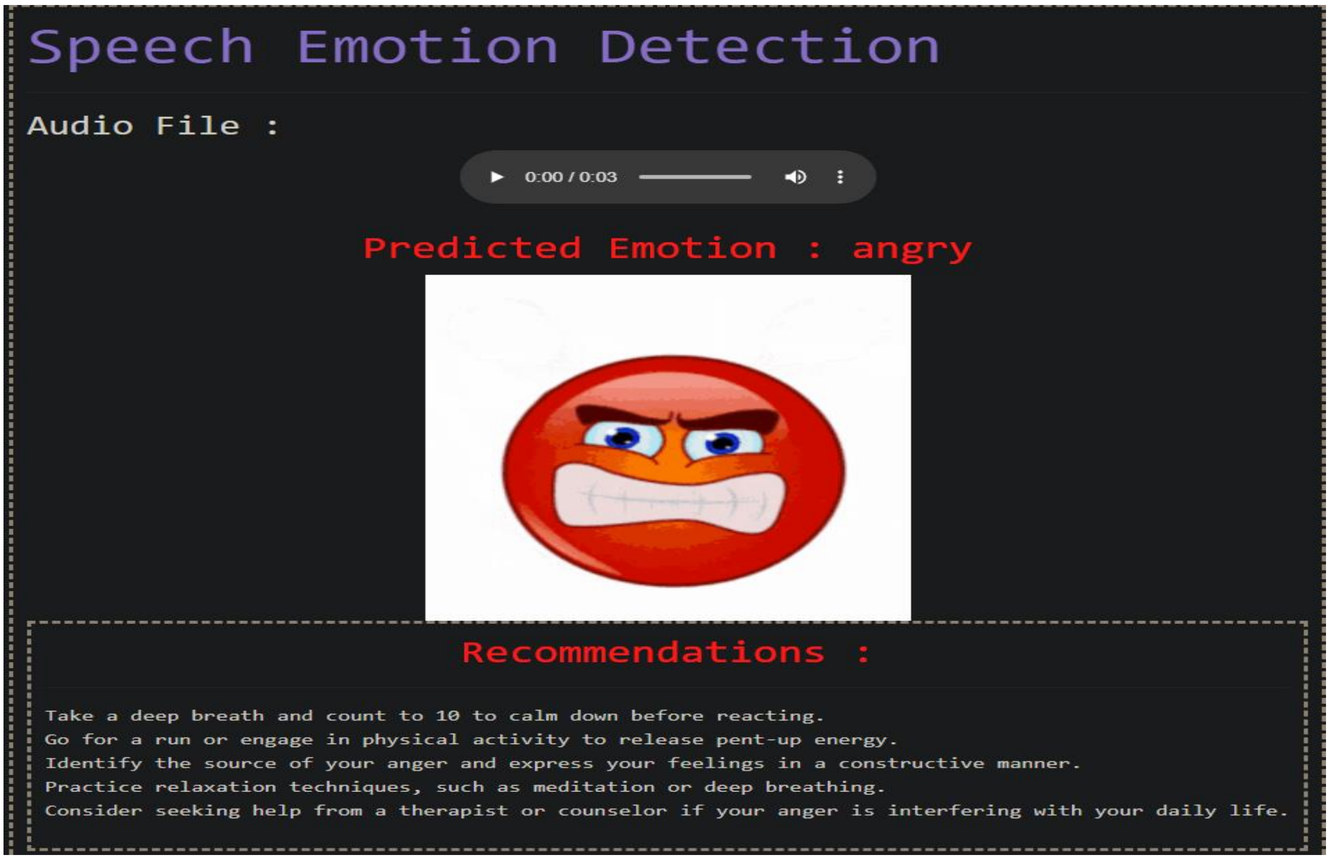


Fig 6: Home Page



Consider trying out a new hobby or activity that you've always been interested in but haven't had the
Take some time to reflect on your current goals and priorities, and make adjustments as needed.
Connect with friends or loved ones and make plans to spend quality time together.
Practice mindfulness or meditation to increase your awareness and presence in the moment.
Engage in regular physical exercise or movement to boost your mood and energy levels.

Fig 7: Image Emotion Detection Output



Speech Emotion Detection

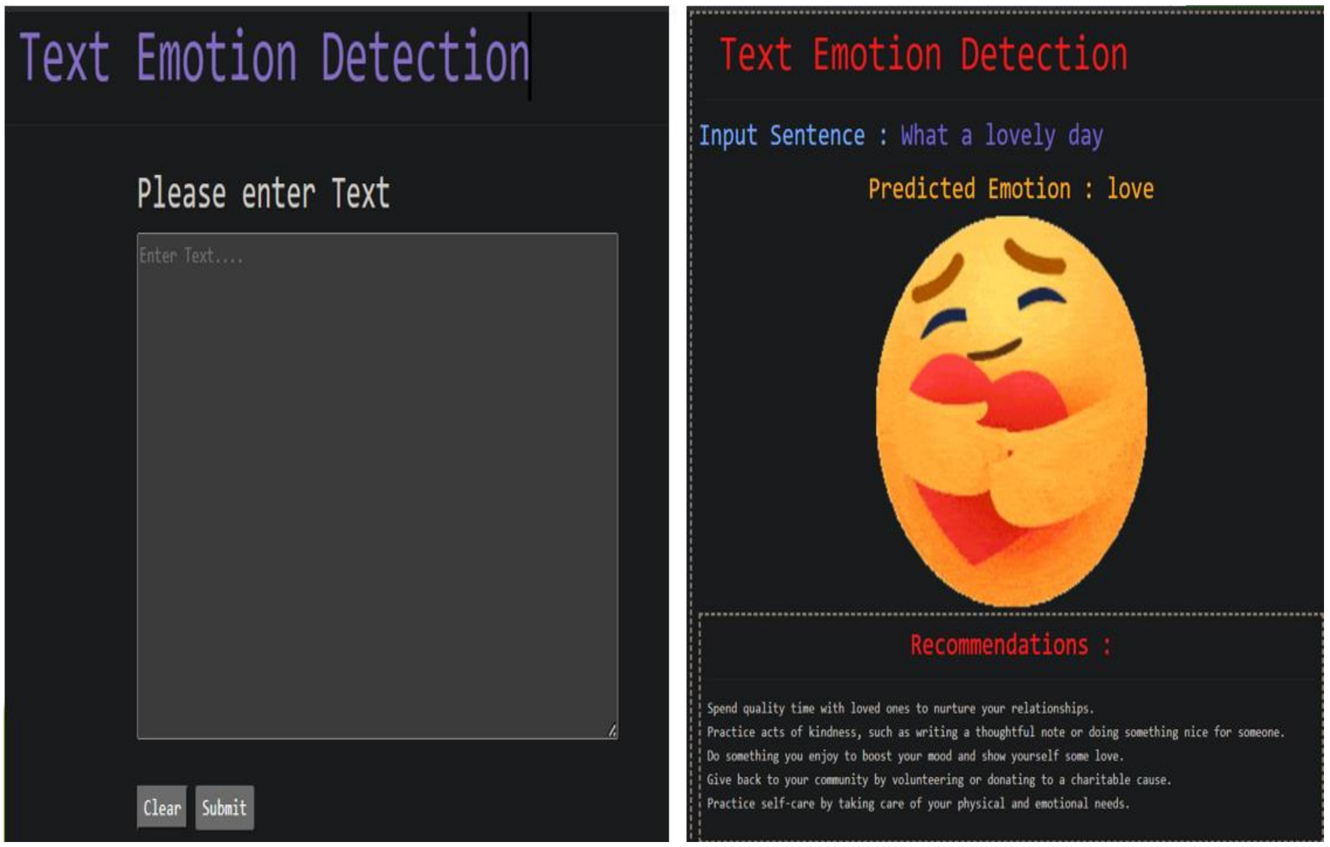
Audio File :

Predicted Emotion : angry

Recommendations :

Take a deep breath and count to 10 to calm down before reacting.
Go for a run or engage in physical activity to release pent-up energy.
Identify the source of your anger and express your feelings in a constructive manner.
Practice relaxation techniques, such as meditation or deep breathing.
Consider seeking help from a therapist or counselor if your anger is interfering with your daily life.

Fig 8: Speech Emotion Detection Output



Text Emotion Detection

Please enter Text

Text Emotion Detection

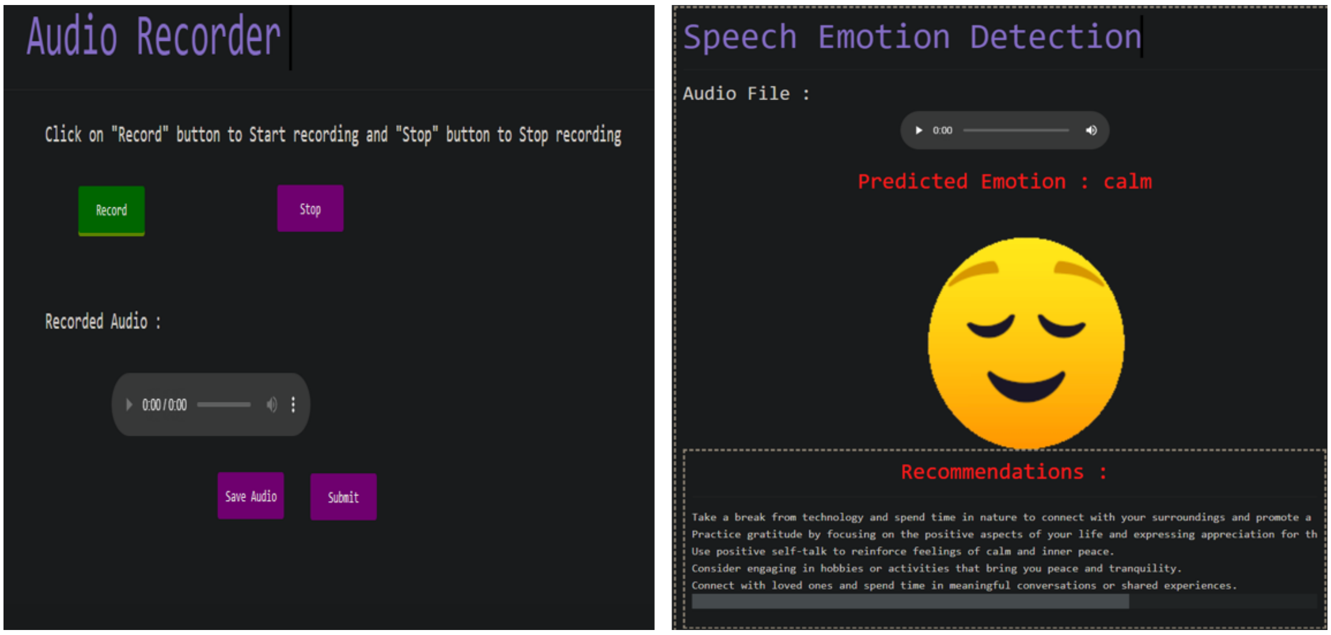
Input Sentence : What a lovely day

Predicted Emotion : love

Recommendations :

Spend quality time with loved ones to nurture your relationships.
Practice acts of kindness, such as writing a thoughtful note or doing something nice for someone.
Do something you enjoy to boost your mood and show yourself some love.
Give back to your community by volunteering or donating to a charitable cause.
Practice self-care by taking care of your physical and emotional needs.

Fig 9: Text Emotion Detection Output



Audio Recorder

Click on "Record" button to Start recording and "Stop" button to Stop recording

Record Stop

Recorded Audio :

0:00 / 0:00

Save Audio Submit

Speech Emotion Detection

Audio File :

0:00

Predicted Emotion : calm

Recommendations :

Take a break from technology and spend time in nature to connect with your surroundings and promote a
Practice gratitude by focusing on the positive aspects of your life and expressing appreciation for th
Use positive self-talk to reinforce feelings of calm and inner peace.
Consider engaging in hobbies or activities that bring you peace and tranquility.
Connect with loved ones and spend time in meaningful conversations or shared experiences.

Fig 10: Audio Record Emotion Detection Output

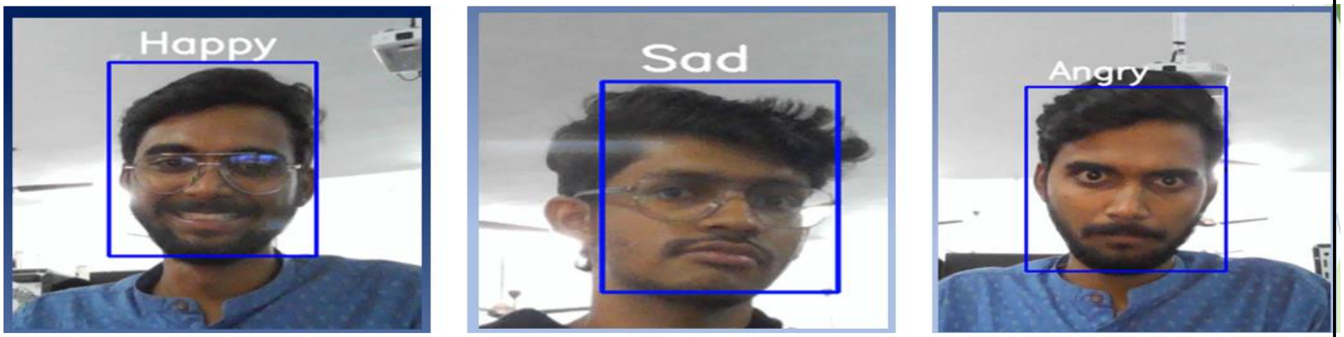


Fig 11: Real Time Emotion Detection Output

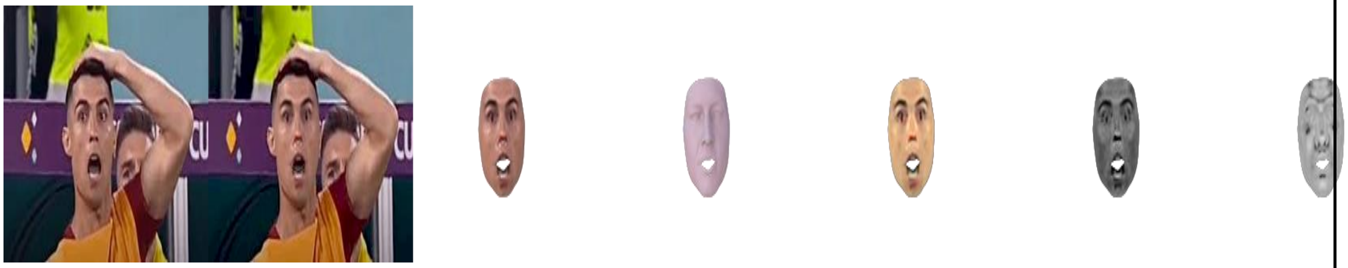


Fig 12: 3D Mesh generator

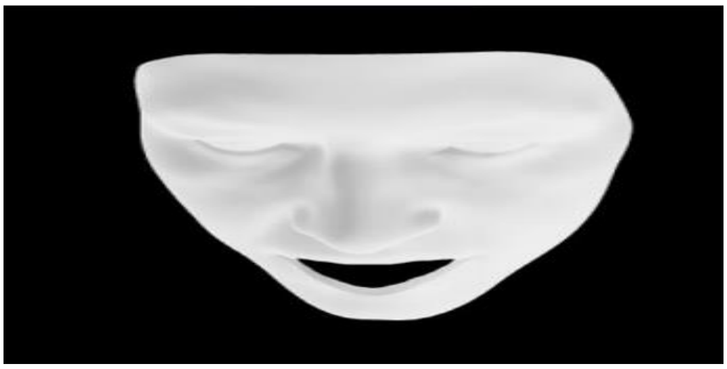


Fig 13: 3D model

CHAPTER 6 –

6.1 CONCLUSION

- In conclusion, this multi-modal emotion detection project has demonstrated the effectiveness of combining different deep learning models for accurately detecting emotions from facial expressions. By utilizing multiple models, including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and LSTM the project achieved a high level of accuracy in identifying emotions (anger, disgust, fear, happiness, sadness, and surprise).
- Moreover, the project goes beyond traditional emotion detection systems by generating a 3D file of the input image. This innovative feature allows for a more immersive and interactive experience for users, enabling them to visualize and explore the emotions detected in a more engaging and dynamic way. The potential applications of this project are numerous, including in fields such as psychology, market research, and human-computer interaction.
- Overall, this project represents a significant advancement in the field of emotion detection, offering a more comprehensive and interactive approach to understanding and analyzing human emotions.

6.2 FUTURE SCOPE

- Improved accuracy: AI is continually evolving, and improvements in these technologies will lead to greater accuracy in detecting emotion from uploaded images.
- The project can be further extended by developing an API.
- Improving 3d model efficiency
- Enhance 3D visualization

6.3 REFERENCES

[1]Abadi, Martín & Barham, Paul & Chen, Jianmin & Chen, Zhifeng & Davis, Andy & Dean, Jeffrey & Devin, Matthieu & Ghemawat, Xiaoqiang. (2016). TensorFlow: A system for large-scale machine learning.

[2] Sarraf, A. (2023). Binary Image Classification Through an Optimal Topology for Convolutional Neural Networks.

[3] Abu, Mohd Azlan & Indra, Nurul Hazirah & Abd Rahman, Abdul & Sapiee, Nor & Ahmad, Izanoordina. (2019). A study on Image Classification based on Deep Learning and Tensorflow.

- https://www.kaggle.com/datasets/msambare/fer2013
- https://faces.dmi.unibas.ch/bfm/bfm2017.html
- https://www.kaggle.com/datasets/uwrfkaggler/ravdess-emotional-speech-audio
- https://www.kaggle.com/datasets/praveengovi/emotions-dataset-for-nlp