SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA - 751024

May 2024

A PROJECT REPORT

on

"MOODMUSIC: A MOOD BASED MUSIC RECOMMENDER SYSTEM"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN

COMPUTER SCIENCE AND ENGINEERING

BY

Sattwik Sen	2105403
Mayukh Patra	21051226
Satyam Raj	21051252
Soumya Kanti Datta	21051262
Srijan Agrawal	21051263

UNDER THE GUIDANCE OF

ABHISHEK RAJ

Table of Contents

Project Overview

This project combines facial emotion detection with music recommendations, creating an interactive system that suggests music based on a user's emotional state. The system can work with both real-time webcam input and uploaded images.

System Components

1. Core Components

- Facial Emotion Detection Model
- Music Recommendation Engine
- Image Processing Pipeline
- User Interface Components (Webcam and File Upload)

2. File Structure

```
project/
│
├── Camera.py              # Real-time webcam detection with music
recommendations
├── realtimedetection.py   # Standalone real-time emotion detection
├── Upload.py              # Image upload and processing functionality
├── requirements.txt       # Project dependencies
└── Additional Files
    ├── facialemotionmodel.h5    # Pre-trained model weights
    ├── facialemotionmodel.json  # Model architecture
    └── music_data.csv           # Music dataset with mood mappings
```

Technical Implementation

1. Neural Network Architecture

The emotion detection model uses a Convolutional Neural Network (CNN) with the following architecture:

- Input Layer: 48x48x1 (grayscale images)
- Multiple Convolutional Layers:
  - Conv2D layers (128 → 256 → 512 → 512 filters)
  - MaxPooling2D layers
  - Dropout layers (0.4 rate)
- Dense Layers:
  - 512 units → 256 units → 7 units (output)

- Output: 7 emotion classes (angry, disgust, fear, happy, neutral, sad, surprise)

2. Emotion-to-Mood Mapping
Emotion Mapping:
- Angry → Calm
- Disgust → Calm
- Fear → Calm
- Happy → Happy
- Neutral → Happy
- Sad → Sad
- Surprise → Energetic

Detailed Code Implementation

1. MoodMusicRecommender Class

```
class MoodMusicRecommender:
    def __init__(self, model_path: str, music_data_path: str):
        self.model = load_model(model_path)
        self.mood_music = pd.read_csv(music_data_path)
        self.emotion_mapping = {
            0: 'Calm',   # angry
            1: 'Calm',   # disgust
            2: 'Calm',   # fear
            3: 'Happy',  # happy
            4: 'Happy',  # neutral
            5: 'Sad',    # sad
            6: 'Energetic'  # surprise
        }

    def get_recommendations(self, emotion_idx: int, num_songs: int = 5):
        mood = self.emotion_mapping[emotion_idx]
        mood_songs = self.mood_music[self.mood_music['mood'] == mood]
        if mood_songs.empty:
            raise ValueError(f"No songs available for the mood '{mood}'.")
        return mood_songs.sample(n=min(num_songs, len(mood_songs)))[['name',
'artist', 'mood']]
```

Key Features: - Constructor initializes model and music dataset - Emotion to mood mapping dictionary - Recommendation function with configurable number of songs - Error handling for empty mood categories

2. Neural Network Implementation

```
model = Sequential([
    Input(shape=(48, 48, 1)),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Dropout(0.4),
    Conv2D(256, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Dropout(0.4),
    Conv2D(512, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Dropout(0.4),
    Conv2D(512, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Dropout(0.4),
    Flatten(),
    Dense(512, activation='relu'),
    Dense(256, activation='relu'),
    Dense(7, activation='softmax')
])
```

3. Image Processing Functions

```
def extract_features(image):
    feature = np.array(image)
    feature = feature.reshape(1, 48, 48, 1)
    return feature / 255.0

def process_image(image_path):
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
    return gray, faces
```

4. Real-time Camera Processing (Camera.py)

```
# Initialize components
haar_file = cv2.data.haarcascades + 'haarcascade_frontalface_default.xml'
face_cascade = cv2.CascadeClassifier(haar_file)
webcam = cv2.VideoCapture(0)
labels = {0: 'angry', 1: 'disgust', 2: 'fear', 3: 'happy',
          4: 'neutral', 5: 'sad', 6: 'surprise'}
```

```
while True:
    ret, im = webcam.read()
    if not ret:
        print("Failed to grab frame")
        break

    gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)

    for (x, y, w, h) in faces:
        face = gray[y:y+h, x:x+w]
        face_resized = cv2.resize(face, (48, 48))
        features = extract_features(face_resized)
        prediction = model.predict(features)
        emotion_idx = prediction.argmax()

        # Draw rectangle and emotion label
        cv2.rectangle(im, (x, y), (x+w, y+h), (255, 0, 0), 2)
        cv2.putText(im, labels[emotion_idx], (x, y-10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255, 255, 255), 2)

    cv2.imshow('Emotion Detection', im)
    key = cv2.waitKey(1)
    if key == ord('q'):
        break
```

5. Image Upload Implementation (Upload.py)

```
def upload_image():
    Tk().withdraw()
    file_path = filedialog.askopenfilename(
        title="Select an Image",
        filetypes=[("Image Files", "*.jpg;*.jpeg;*.png")]
    )
    return file_path if file_path else None

def print_recommendations(recommendations_df):
    if recommendations_df is not None:
        recommendations_str = recommendations_df.to_string(index=False)
        lines = recommendations_str.splitlines()
        max_line_length = max(len(line) for line in lines)
        print('+' + '-' * (max_line_length + 2) + '+')
        for line in lines:
            print(f'| {line.ljust(max_line_length)} |')
        print('+' + '-' * (max_line_length + 2) + '+')

def main():
    recommender = MoodMusicRecommender('facialemotionmodel.h5',
'music_data.csv')

    while True:
        image_path = upload_image()
        if image_path is None:
            print("No image selected. Exiting...")
            break

        gray, faces = process_image(image_path)
        if len(faces) == 0:
            print("No faces detected in the image.")
            continue

        for (x, y, w, h) in faces:
            face = gray[y:y+h, x:x+w]
            face_resized = cv2.resize(face, (48, 48))
            features = extract_features(face_resized)
            prediction = model.predict(features)
            emotion_idx = prediction.argmax()

            recommendations = recommender.get_recommendations(emotion_idx)
            print(f"\nDetected emotion: {labels[emotion_idx]}")
            print("\nMusic Recommendations:")
            print_recommendations(recommendations)

        if input("\nProcess another image? (yes/no): ").lower() != 'yes':
            break
```

Usage Instructions

1. Installation

```
pip install -r requirements.txt
```

Required dependencies: - tensorflow - keras - pandas - numpy - opencv-contrib-python - jupyter - notebook - tqdm

2. Running the Application

For Real-time Webcam Detection:
python Camera.py

- Press ‘c’ to capture and analyze
- Press ‘q’ to quit

For Image Upload:
python Upload.py

- Select image through file dialog
- View results and recommendations
- Choose to process another image

For Basic Emotion Detection:
python realtimedetection.py

- Press ‘q’ to quit

Testing Considerations

Unit Testing Areas

1. Emotion Detection:
   - Face detection accuracy
   - Emotion classification accuracy
   - Input image preprocessing
2. Music Recommendation:
   - Mood mapping accuracy
   - Recommendation randomization
   - Empty mood category handling

Integration Testing

1. Camera Integration:
   - Webcam initialization
   - Frame capture timing
   - Memory management
2. User Interface:
   - File dialog functionality

    - Error message display
    - User input handling

<u>Limitations and Constraints</u>

1. Face Detection:
    - Works best with front-facing faces
    - Requires adequate lighting
    - Single face processing in upload mode
2. Emotion Detection:
    - Limited to 7 basic emotions
    - May be affected by image quality
    - Requires clear facial features
3. Music Recommendations:
    - Limited by available songs in database
    - Fixed mood mappings
    - Random selection within mood categories

<u>Future Enhancement Possibilities</u>

1. Technical Improvements:
    - Multiple face detection and processing
    - Enhanced emotion detection accuracy
    - Real-time music playback integration
2. Feature Additions:
    - Custom mood mappings
    - User feedback integration
    - Playlist generation
    - Music preference learning