Emotion Detection from Facial Expressions

A
Project Report
submitted in partial fulfillment of the requirements for the award of the degree of

MASTER'S
in
COMPUTER APPLICATIONS

by

Name	Roll No.
SANTISWARUP NAYAK	500120092
ANWESHA SWARUP	500125984
NANDINI SRIVASTAVA	500125511
VANSHIKA CHAUDHARY	500125855
TEJENDRA Ku. PANDA	500125917

Under the guidance of
Mr. Deen Mohamad

School of Computer Science, UPES
Bidholi, Via Prem Nagar, Dehradun, Uttarakhand

March – 2025

CANDIDATE'S DECLARATION

I/We hereby certify that the project work entitled **"Emotion Detection from Facial Expressions"** in partial fulfillment of the requirements for the award of the Degree of **MASTER IN COMPUTER APPLICATIONS** in **COMPUTER SCIENCE** with specialization in **DATA SCIENCE AI/ML**, submitted to the Data Science Cluster, School of Computer Science, UPES, Dehradun, is an authentic record of my/our work carried out during a period from **February, 2025** to **May, 2025** under the supervision of **Mr. Deen Mohamad, Designation and Affiliation**.

The matter presented in this project has not been submitted by me/us for the award of any other degree of this or any other University.

Name	Roll No.
SANTISWARUP NAYAK	500120092
ANWESHA SWARUP	500125984
NANDINI SRIVASTAVA	500125511
VANSHIKA CHAUDHARY	500125855
TEJENDRA Ku. PANDA	500125917

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: 28th March 2025

Name of Guide
Mr. Deen Mohamad

Acknowledgement

We wish to express our deep gratitude to our guide Mr. Deen Mohamad, for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions. We sincerely thanks to our respected Dr. Virendra Kadyan, Head Department of Computer Science, for his great support in doing our project in Emotion Detection from Facial Expressions. We are also grateful to Dean SoCS UPES for giving us the necessary facilities to carry out our project work successfully. We also thanks to our Course Coordinator, Mr. Mrinal Maji and information during the completion of this project.

We would like to thank all our friends for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our parents who have shown us this world and for every support they have given us.

Name	Roll No.
SANTISWARUP NAYAK	500120092
ANWESHA SWARUP	500125984
NANDINI SRIVASTAVA	500125511
VANSHIKA CHAUDHARY	500125855
TEJENDRA Ku. PANDA	500125917

Abstract

Emotion detection using deep learning has gained significant attention due to its applications in human-computer interaction, security, and psychological analysis. This project presents an emotion detection system that utilizes Convolutional Neural Networks (CNNs) to classify facial expressions into various emotional categories. The system follows a structured pipeline, including face detection, image preprocessing, and emotion classification.

Face detection is implemented using OpenCV's Haar Cascades, MTCNN, or RetinaFace, ensuring robustness across different lighting conditions and facial orientations. To enhance the quality of low-light images, preprocessing techniques such as Histogram Equalization, CLAHE, and Retinex-based methods are applied, improving visibility and feature extraction. The core of the system is the CNN-based emotion classification model, trained on datasets like FER-2013 and AffectNet. Various architectures, including VGGNet, ResNet, and EfficientNet, are explored to balance accuracy and efficiency.

For real-time implementation, OpenCV is used for video capture, and threading is incorporated to optimize performance. The trained model is deployed to classify emotions in real-time video streams, making it suitable for interactive applications. The project also considers ethical concerns related to privacy and bias in emotion detection models.

Through extensive testing and fine-tuning, the proposed system aims to achieve high accuracy in recognizing emotions under diverse conditions. The report provides a comprehensive overview of the methodologies employed, datasets used, and potential real-world applications of the emotion detection system. Future enhancements may include integrating multi-modal emotion recognition and improving model generalization through more diverse datasets. This project serves as a foundation for developing intelligent systems capable of understanding and responding to human emotions effectively.

Contents

Chapter 1

Introduction

Emotions play a crucial role in human communication, decision-making, and overall well-being. With advancements in artificial intelligence and deep learning, there has been a growing interest in developing systems capable of recognizing and interpreting human emotions. Emotion detection is a challenging yet promising field that finds applications in various domains, including healthcare, security, human-computer interaction, and entertainment. The ability of machines to detect emotions can lead to more intuitive user experiences, improved mental health assessments, and enhanced security measures.

This project focuses on building an emotion detection system using Convolutional Neural Networks (CNNs). The goal is to classify facial expressions into different emotional categories by processing images or video streams. The system follows a structured pipeline, beginning with face detection, followed by image preprocessing, and finally emotion classification using a deep learning model.

Face Detection

Face Detection Face detection is the first and most crucial step in emotion recognition. It involves identifying and extracting the face region from an image or video frame. There are several techniques available for this task:

Haar Cascades

A traditional method that is fast but less accurate, especially in poor lighting or varying facial orientations.

MTCNN (Multi-task Cascaded Convolutional Networks)

A deep learning-based method that provides higher accuracy but requires more computation.

RetinaFace

A state-of-the-art face detection model known for its robustness in handling occlusions and varying lighting conditions. The detected face is then passed to the next stage for further processing.

Image Preprocessing

Preprocessing plays a significant role in improving the quality of input images, particularly in low-light environments. Several techniques are employed to enhance visibility and extract meaningful facial features:

Histogram Equalization

Adjusts the contrast of the entire image to enhance visibility.

CLAHE (Contrast Limited Adaptive Histogram Equalization)

Improves local contrast while reducing noise.

Retinex-based Methods

Mimic human visual perception to enhance images in low-light conditions.

Deep Learning-based Enhancement

Uses neural networks (e.g., U-Net) to restore image quality dynamically. By refining the input images, the system ensures better accuracy in emotion classification.

Emotion Classification Using CNNs

Once the preprocessed face images are ready, they are fed into a CNN-based emotion classification model. The model is trained on large datasets such as:

FER-2013

A widely used dataset containing thousands of labeled facial expressions.

AffectNet

A more diverse dataset that helps improve recognition across different demographics. To optimize performance, different CNN architectures such as VGGNet, ResNet, and EfficientNet are explored, balancing accuracy and computational efficiency. The trained model then classifies emotions such as happiness, sadness, anger, surprise, and fear in real-time.

Applications and Future Scope

Emotion detection systems have numerous applications. They can be used in mental health monitoring to detect signs of stress or depression, in customer service to analyze user satisfaction, and in surveillance systems to identify potential security threats. In the future, integrating multi-modal data such as voice and physiological signals can further enhance accuracy and real-world applicability.

This project aims to contribute to the growing field of emotion recognition by developing a robust, efficient, and scalable solution for real-world applications.

1.1 History

The study of emotions has fascinated researchers for centuries, with early theories attempting to classify human emotions and their expressions. Charles Darwin, in The Expression of the Emotions in Man and Animals (1872), proposed that emotions are universal and can be identified through facial expressions. This idea was later expanded by Paul Ekman, who identified six basic emotions—happiness, sadness, anger, surprise, fear, and disgust—that are recognizable across cultures.

With the rise of computers in the late 20th century, early emotion detection methods relied on handcrafted features, analyzing facial landmarks such as eyebrow movement and mouth curvature. The introduction of OpenCV and algorithms like Haar Cascades enabled automated face detection, though these methods struggled with variations in lighting and pose.

The deep learning revolution in the 2010s significantly improved emotion detection. Convolutional Neural Networks (CNNs) eliminated the need for manual feature extraction, allowing models to learn patterns directly from data. Large datasets like FER-2013 and AffectNet helped train these models, while advanced architectures like VGGNet, ResNet, and EfficientNet enhanced accuracy. Simultaneously, MTCNN and RetinaFace improved face detection, making real-world applications more reliable.

Today, emotion detection is widely used in human-computer interaction, healthcare, security, and marketing. Future advancements will focus on multi-modal emotion recognition, combining facial expressions, voice, and physiological signals for improved accuracy and inclusivity.

1.2 Requirement Analysis

Functional Requirements

These define what the system must do to achieve emotion detection effectively.

Face Detection

- Detect faces in images and video streams.
- Support multiple detection methods (Haar Cascades, MTCNN, RetinaFace) for accuracy and speed balance.
- Handle variations in lighting, pose, and occlusion.

Image Preprocessing

- Enhance low-light images using Histogram Equalization, CLAHE, or Retinex-based methods.
- Resize and normalize detected faces for consistent model input.
- Remove noise while preserving facial features for better classification.

Emotion Classification

- Utilize CNN-based models (VGGNet, ResNet, EfficientNet, or custom CNN) to classify facial expressions.
- Train on large datasets (FER-2013, AffectNet) for robust recognition.
- Provide real-time predictions with optimized inference speed.

Real-Time Implementation (Optional)

- Process real-time video streams using OpenCV and threading.
- Display detected faces with their corresponding emotions on-screen.
- Enable potential web-based or mobile integration.

Non-Functional Requirements

These define the quality and performance expectations of the system.

Performance

- Face detection and emotion classification should execute in real time (30 FPS for video).
- Model inference time should be optimized for efficiency.
- GPU acceleration (CUDA, TensorRT) may be used for deep learning models.

Accuracy and Reliability

- The system should achieve high accuracy on benchmark datasets.
- It must generalize well to different lighting conditions, angles, and ethnicities.
- Fine-tuning may be required to reduce false positives.

Security and Privacy

- No sensitive user data should be stored without consent.
- Compliance with ethical guidelines regarding facial recognition.
- If used in real-world applications, ensure GDPR or similar privacy standards.

Scalability

- The system should be scalable for deployment in different environments (desktop, cloud, mobile).
- Support model updates for improved performance over time.

Hardware and Software Requirements

Hardware

- **Processor:** Intel i5/i7 or equivalent (for development).
- **GPU:** NVIDIA RTX 2060 or higher (for deep learning).
- **RAM:** Minimum 8GB (16GB recommended).
- **Camera:** HD webcam (for real-time applications).

Software & Libraries

- **Programming Language:** Python
- **Libraries:** OpenCV, TensorFlow/Keras, PyTorch, MTCNN, RetinaFace
- **OS:** Windows, Linux, or macOS
- **Development Tools:** Jupyter Notebook, VS Code

1.3 Main Objective

The primary objective of this project is to develop an emotion detection system that utilizes deep learning and computer vision to accurately classify human facial expressions into various emotional categories. By leveraging Convolutional Neural Networks (CNNs) and face detection algorithms, the system aims to analyze facial features and recognize emotions in real time or from static images.

Key Goals

Accurate Emotion Recognition

- Implement CNN-based models (VGGNet, ResNet, EfficientNet) to classify emotions like happiness, sadness, anger, surprise, fear, and neutral expressions.
- Train the model on diverse datasets (FER-2013, AffectNet) to improve accuracy and generalization.

Robust Face Detection

- Use OpenCV's Haar Cascades, MTCNN, or RetinaFace for efficient face detection.
- Ensure the system performs well under different lighting conditions, angles, and occlusions.

Image Preprocessing for Low-Light Conditions

- Enhance image quality using techniques like CLAHE, Histogram Equalization, and Retinex-based methods.
- Improve visibility in dark environments for better emotion classification.

Real-Time Processing (Optional)

- Capture live video feeds using OpenCV and process frames dynamically.
- Optimize performance through threading and GPU acceleration for faster inference.

Scalability and Deployment

- Develop a system that can be integrated into applications, including web-based, mobile, and security systems.
- Ensure efficient performance across different hardware configurations (desktop, cloud, and edge devices).

Ethical Considerations & Privacy

- Address ethical concerns related to facial recognition and ensure responsible AI usage.
- Comply with data privacy regulations and prevent misuse of personal data.

Conclusion

By achieving these objectives, the project will contribute to human-computer interaction, psychological analysis, security applications, and AI-driven emotion recognition systems.

1.4 Sub Objectives

1. Face Detection & Preprocessing

- Implement Haar Cascades, MTCNN, and RetinaFace for detecting faces accurately in various lighting and pose conditions.
- Enhance image quality using CLAHE and Retinex-based techniques to improve emotion recognition in low-light environments.

2. Emotion Classification using Deep Learning

- Train CNN-based models (VGGNet, ResNet, EfficientNet) to classify facial expressions into different emotion categories.
- Use data augmentation techniques (flipping, rotation, brightness adjustments) to improve model generalization.

3. Dataset Selection & Training

- Utilize diverse datasets like FER-2013 and AffectNet to ensure a well-balanced training dataset.
- Apply transfer learning to leverage pre-trained models for improved accuracy and reduced training time.

4. Real-Time Implementation & Optimization (Optional)

- Integrate the model with OpenCV for real-time emotion detection in video streams.
- Optimize inference speed using multi-threading and GPU acceleration to achieve real-time processing.

5. Deployment & Scalability

- Develop a web-based application using Flask/Django for easy accessibility and interaction.
- Ensure cross-platform compatibility for desktop, mobile, and cloud environments.

6. Ethical Considerations & Privacy Protection

- Comply with data privacy regulations to protect user information and prevent misuse of facial data.
- Address dataset biases to ensure fair and non-discriminatory emotion detection across diverse demographics.

1.5 Pert Chart Legend

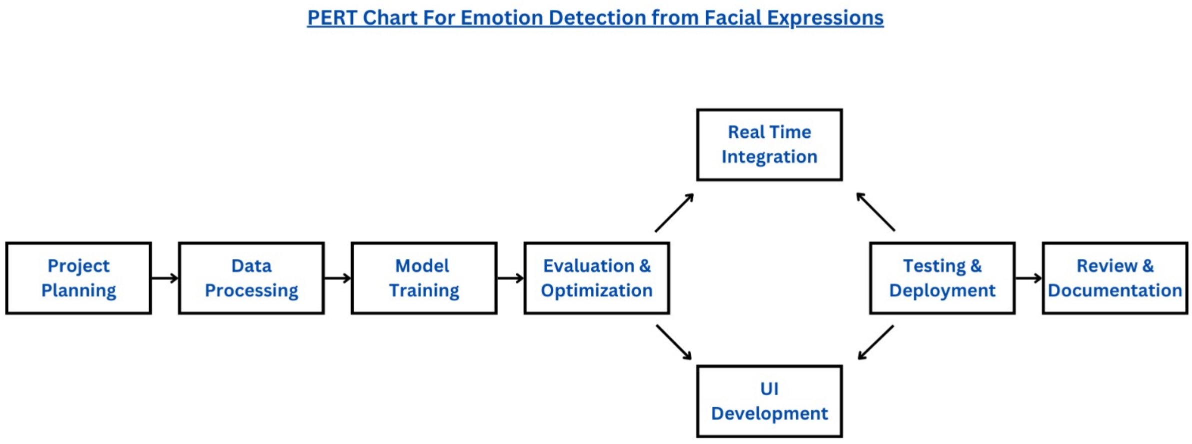

Figure 1.1: Pert Chart

Chapter 2

System Analysis

2.1 Existing System

Emotion detection has been an area of research in computer vision and artificial intelligence for many years. Several existing systems use traditional and deep learning-based approaches to analyze facial expressions and classify emotions. However, these systems have limitations in accuracy, real-time performance, and adaptability under various conditions.

1. Traditional Methods for Emotion Detection

Facial Action Coding System (FACS)

- A manual system that categorizes facial movements into Action Units (AUs) to interpret emotions.
- Requires human expertise, making it inefficient for large-scale applications.

Handcrafted Feature-Based Approaches

- Methods like Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG) extract features from facial images.
- Limited accuracy due to their inability to handle variations in lighting, pose, and occlusions.

2. Machine Learning-Based Approaches

Support Vector Machines (SVM) and Random Forest

- Used with extracted facial features for classification.
- Performance is highly dependent on feature extraction quality, making them less effective in dynamic environments.

Hidden Markov Models (HMM)

- Used for sequential emotion recognition from video data.
- Computationally expensive and requires significant training data for accurate predictions.

3. Deep Learning-Based Emotion Detection

CNN-Based Models (VGGNet, ResNet, EfficientNet)

- Learn hierarchical features directly from raw facial images.
- Provide higher accuracy than traditional methods but require large datasets and high computational power.

Pre-Trained Models & Transfer Learning

- Models trained on large datasets (e.g., FER-2013, AffectNet) are fine-tuned for emotion recognition.
- Performance may still drop in real-world scenarios due to dataset bias and uncontrolled conditions.

4. Challenges in Existing Systems

Sensitivity to Lighting & Pose Variations

- Many models fail in low-light environments or when faces are not front-facing.

Real-Time Performance Issues

- High computational demand affects real-time emotion detection speed.

Bias in Datasets

- Existing datasets may lack diversity, leading to biased predictions across different demographics.

Privacy & Ethical Concerns

- Storing and analyzing facial data raises concerns about user privacy and consent.

2.2 Motivations

Emotion detection plays a crucial role in human-computer interaction, psychological analysis, and security applications. The ability to recognize emotions from facial expressions can significantly enhance AI-driven applications, making them more adaptive, responsive, and human-centric. However, existing solutions face challenges that motivate the need for a more robust and accurate emotion detection system.

1. Enhancing Human-Computer Interaction (HCI)

- Current AI systems lack emotional intelligence, limiting their ability to provide personalized user experiences.
- Implementing emotion recognition can improve virtual assistants, customer service bots, and educational platforms by adapting responses based on user emotions.

2. Improving Mental Health & Psychological Analysis

- Automated emotion detection can help in early diagnosis of mental health conditions like depression, anxiety, and stress.
- Emotion-aware AI can assist therapists by tracking subtle emotional cues in patients over time.

3. Advancements in Deep Learning & Computer Vision

- Recent improvements in CNN architectures (ResNet, EfficientNet) and face detection methods (MTCNN, RetinaFace) allow for higher accuracy and real-time processing.
- The availability of large-scale datasets (FER-2013, AffectNet) makes it possible to train deep learning models for better generalization.

4. Real-World Applications in Security & Surveillance

- Emotion recognition can enhance security systems by detecting suspicious or unusual behavior in crowded areas.
- Law enforcement agencies can use facial emotion analysis to assess individuals' emotional states in critical situations.

5. Addressing Limitations of Existing Systems

- Many current models struggle with low-light conditions, occlusions, and variations in facial expressions.
- Developing a real-time, efficient, and adaptable emotion detection system can overcome these challenges and provide a scalable solution.

6. Ethical & Privacy Considerations

- With rising concerns over data privacy, there is a need to build secure and ethical AI systems for emotion detection.
- Implementing techniques like on-device processing can reduce privacy risks while ensuring user data protection.

2.3 Proposed System

The proposed emotion detection system aims to overcome the limitations of existing methods by leveraging deep learning, advanced face detection, and real-time processing techniques. This system will accurately classify facial expressions into emotions while ensuring robust performance in diverse lighting conditions, poses, and real-world environments.

1. Face Detection & Preprocessing

- **Detection Methods:** The system will utilize MTCNN or RetinaFace for precise face detection, ensuring improved accuracy in challenging conditions such as occlusions, different angles, and varying lighting.
- **Preprocessing:** Images will be enhanced using CLAHE, Retinex-based methods, and normalization techniques to improve emotion detection in low-light environments.

2. Emotion Recognition Using Deep Learning

- A CNN-based model (e.g., ResNet, EfficientNet, or a custom CNN) will be trained on FER-2013 and AffectNet datasets to classify emotions like happiness, sadness, anger, surprise, disgust, and neutrality.
- Transfer learning will be implemented to enhance model accuracy while reducing training time.
- Data augmentation techniques (rotation, flipping, brightness adjustments) will be applied to improve model generalization.

3. Real-Time Implementation (Optional)

- The trained model will be integrated with OpenCV to process real-time video streams.
- Threading and GPU acceleration will be used to optimize performance for real-time emotion detection.

4. Deployment & Scalability

- The system will be deployable across multiple platforms:
  - Standalone application (PC-based).
  - Web-based application using Flask/Django.
  - Cloud-based API for integration into existing software.
- A lightweight model version will be developed for mobile and edge devices.

5. Ethical Considerations & Privacy Protection

- On-device processing will be explored to minimize privacy risks.
- The system will ensure compliance with ethical AI guidelines and data privacy regulations to prevent misuse.
- Bias in datasets will be mitigated by diverse training data and fairness-aware training techniques.

6. Expected Outcomes

- High accuracy in facial emotion recognition, even in real-world conditions.
- Improved real-time performance with optimized deep learning models.
- A scalable and deployable system suitable for various domains, including HCI, healthcare, security, and marketing.

2.4 Modules

The emotion detection system consists of several key modules that work together to detect human emotions from facial expressions. Each module plays a crucial role in processing the input, analyzing facial features, and classifying emotions. Below are the main modules:

1. Face Detection Module

- Detects human faces in real-time or from an image using OpenCV's Haar Cascade Classifier.
- Can be enhanced by integrating MTCNN or RetinaFace for improved accuracy.

2. Image Preprocessing Module

- Converts the detected face into grayscale and resizes it to a fixed dimension (48×48 pixels).
- Applies Contrast Limited Adaptive Histogram Equalization (CLAHE) to enhance low-light images.
- Normalizes pixel values for better CNN performance.

3. Emotion Detection Model (CNN-based Classification)

- A Convolutional Neural Network (CNN) trained to classify seven different emotions: ['Angry', 'Disgust', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral'].
- The model consists of multiple Conv2D, MaxPooling2D, Flatten, Dense, and Dropout layers to improve accuracy and generalization.

4. Model Loading and Prediction Module

- Loads the pre-trained CNN model and its saved weights.
- Takes preprocessed facial images as input and predicts the most probable emotion.

5. Real-Time Emotion Detection Module

- Captures live video input from the webcam using OpenCV.
- Continuously detects faces, preprocesses them, and classifies emotions in real-time.
- Displays the detected emotion using an overlay on the video stream.

6. Visualization and Result Display Module

- Annotates the detected face with a bounding box and labels the recognized emotion.
- Uses OpenCV to render the video feed with emotion predictions.

Chapter 3

Implementation/results

```
[19]: import cv2
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import model_from_json
from mtcnn.mtcnn import MTCNN
import matplotlib.pyplot as plt
import os

print(f"OpenCV version: {cv2.__version__}")
print(f"NumPy version: {np.__version__}")
print(f"TensorFlow version: {tf.__version__}")

OpenCV version: 4.11.0
NumPy version: 1.26.4
TensorFlow version: 2.19.0
```

Figure 3.1: 1

```
[20]: '''import tensorflow as tf

      model = tf.keras.models.load_model("Emotion_Detection_final//emotion_model.h5")
      model.summary()'''

[20]: 'import tensorflow as tf\n\nmodel = tf.keras.models.load_model("Emotion_Detection_final//emotion_model.h5")\nmodel.summary()'

[21]: # 1. Face Detection (Using MTCNN - You can switch to RetinaFace if needed)
      EMOTION_MODEL_PATH = "emotion_model.h5"
      EMOTION_LABELS=["Angry", "Disgust", "Fear", "Happy", "Sad", "Surprise", "Neutral"]
      EMOTION_INPUT_SIZE = (48, 48)

[22]: print("[INFO] Initializing MTCNN face detector...")
      try:
          # MTCNN detector might download weights on first run if not cached
          face_detector = MTCNN()
          print("[INFO] MTCNN initialized.")
      except Exception as e:
          print(f"[ERROR] Could not initialize MTCNN: {e}")
          face_detector = None

      [INFO] Initializing MTCNN face detector...
      [INFO] MTCNN initialized.
```

Figure 3.2: 2

```
[1]: print("[INFO] Loading emotion recognition model...")
     try:
         emotion_model = tf.keras.models.load_model(EMOTION_MODEL_PATH)
         print(f"[INFO] Emotion model loaded successfully. Expects input shape: {EMOTION_INPUT_SIZE}")
         # Optional: You can print model summary
         # emotion_model.summary()
     except (IOError, ImportError, ValueError) as e:
          print(f"[ERROR] Could not load emotion model from {EMOTION_MODEL_PATH}: {e}")
          print("[INFO] Ensure the .h5 file exists and you have TensorFlow/Keras installed.")
          emotion_model = None
     except Exception as e:
          print(f"[ERROR] An unexpected error occurred loading the emotion model: {e}")
          emotion_model = None

     [INFO] Loading emotion recognition model...
```

Figure 3.3: 3

```
[2]: def enhance_low_light(image_bgr):
         """Enhances a low-light BGR image using CLAHE on the L channel."""
         try:
             lab = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2LAB)
             l, a, b = cv2.split(lab)
             clahe = cv2.createCLAHE(clipLimit=2.5, tileGridSize=(8, 8)) # Adjusted clipLimit slightly
             cl = clahe.apply(l)
             limg = cv2.merge((cl, a, b))
             enhanced_image = cv2.cvtColor(limg, cv2.COLOR_LAB2BGR)
             return enhanced_image
         except cv2.error as e:
             print(f"[WARN] Could not enhance image with CLAHE: {e}. Returning original.")
             return image_bgr
         except Exception as e:
             print(f"[WARN] General error during enhancement: {e}. Returning original.")
             return image_bgr
```

Figure 3.4: 4

```python
def detect_and_predict_emotions(image_path, enhance_if_lowlight=True):
    """
    Loads an image, detects faces using MTCNN, optionally enhances,
    predicts emotions, and displays the result.
    """
    if not face_detector or not emotion_model:
        print("[ERROR] Required models (MTCNN or Emotion Model) not loaded. Cannot proceed.")
        return

    # Load the image
    img_bgr = cv2.imread(image_path)
    if img_bgr is None:
        print(f"[ERROR] Could not read image: {image_path}")
        return

    print(f"[INFO] Processing image: {image_path}")
    vis_image = img_bgr.copy() # Image for drawing results

    # --- Optional Low-Light Enhancement ---
    if enhance_if_lowlight:
        # Simple check: if average pixel intensity is low, enhance
        # You might need a more sophisticated check
        gray_check = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2GRAY)
        mean_intensity = np.mean(gray_check)
        print(f"[INFO] Average pixel intensity: {mean_intensity:.2f}")
        if mean_intensity < 70: # Threshold for 'low-light' (adjust as needed)
            print("[INFO] Low light detected, applying enhancement...")
```

Figure 3.5: 5

```python
            vis_image = enhance_low_light(vis_image)
            print("[INFO] Enhancement applied.")
        else:
            print("[INFO] Sufficient light, skipping enhancement.")

    # MTCNN expects RGB format
    img_rgb = cv2.cvtColor(vis_image, cv2.COLOR_BGR2RGB)

    # --- Face Detection ---
    print("[INFO] Detecting faces using MTCNN...")
    detections = face_detector.detect_faces(img_rgb)
    print(f"[INFO] Found {len(detections)} face(s).")

    if not detections:
        print("[INFO] No faces detected.")
    else:
        # --- Emotion Prediction Loop ---
        for face_info in detections:
            # MTCNN returns confidence and bounding box [x, y, width, height]
            confidence = face_info['confidence']
            if confidence < 0.90: # Confidence threshold for MTCNN detection
                print(f"[INFO] Skipping face with low confidence: {confidence:.3f}")
                continue

            x, y, w, h = face_info['box']
            # Ensure coordinates are non-negative after potential floating point issues
            x, y = max(0, x), max(0, y)
```

Figure 3.6: 6

```python
            x, y, w, h = face_info['box']
            # Ensure coordinates are non-negative after potential floating point issues
            x, y = max(0, x), max(0, y)

            # Extract face ROI from the *potentially enhanced* BGR image
            face_roi_bgr = vis_image[y : y + h, x : x + w]

            if face_roi_bgr.size == 0:
                print("[WARN] Extracted face ROI is empty, skipping.")
                continue

            try:
                # Preprocess ROI for the emotion model
                face_roi_gray = cv2.cvtColor(face_roi_bgr, cv2.COLOR_BGR2GRAY)
                face_roi_resized = cv2.resize(face_roi_gray, EMOTION_INPUT_SIZE, interpolation=cv2.INTER_AREA)

                # Normalize pixel values (common: 0-1 range)
                face_roi_normalized = face_roi_resized.astype("float32") / 255.0

                # Expand dimensions to match model input (batch_size, height, width, channels)
                # Model might expect (1, 48, 48, 1) if grayscale
                face_roi_expanded = np.expand_dims(face_roi_normalized, axis=-1) # Add channel dim
                face_roi_final = np.expand_dims(face_roi_expanded, axis=0)    # Add batch dim

                # Predict emotion
                predictions = emotion_model.predict(face_roi_final)
                emotion_index = np.argmax(predictions[0])
```

Figure 3.7: 7

```python
            emotion_label = EMOTION_LABELS[emotion_index]
            emotion_confidence = predictions[0][emotion_index]

            # Draw bounding box and label on the visualization image (vis_image is BGR)
            label_text = f"{emotion_label} ({emotion_confidence*100:.1f}%)"
            cv2.rectangle(vis_image, (x, y), (x + w, y + h), (0, 255, 0), 2) # Green box
            # Put text slightly above the box
            text_y = y - 10 if y - 10 > 10 else y + 10
            cv2.putText(vis_image, label_text, (x, text_y), cv2.FONT_HERSHEY_SIMPLEX,
                        0.6, (0, 255, 0), 2) # Green text

            print(f"[RESULT] Face at [{x},{y},{w},{h}]: {emotion_label} ({emotion_confidence:.3f})")

        except cv2.error as e:
            print(f"[WARN] OpenCV error processing face ROI at [{x},{y},{w},{h}]: {e}")
        except IndexError as e:
            print(f"[ERROR] Emotion label index out of bounds. Check EMOTION_LABELS list. Error: {e}")
        except Exception as e:
             print(f"[WARN] Error during emotion prediction for face at [{x},{y},{w},{h}]: {e}")


    # --- Display Result ---
    # Convert final BGR image to RGB for matplotlib display
    vis_image_rgb = cv2.cvtColor(vis_image, cv2.COLOR_BGR2RGB)

    plt.figure(figsize=(10, 8))
    plt.imshow(vis_image_rgb)
```

Figure 3.8: 8

```python
    vis_image_rgb = cv2.cvtColor(vis_image, cv2.COLOR_BGR2RGB)

    plt.figure(figsize=(10, 8))
    plt.imshow(vis_image_rgb)
    plt.title(f"Emotion Recognition Results ({os.path.basename(image_path)})")
    plt.axis('off')
    plt.show()
```

```python
'''# --- Specify Image Path ---
# <<< CHANGE THIS TO YOUR IMAGE FILE >>>
# Try both a normal light and a low light image
# test_image_path = "path/to/your/normal_image.jpg"
test_image_path = "path/to/your/low_light_image.jpg"
# test_image_path = "path/to/your/another_image.png"

# --- Execute ---
if os.path.exists(test_image_path):
    if face_detector and emotion_model: # Make sure models loaded correctly
        detect_and_predict_emotions(test_image_path, enhance_if_lowlight=True)
    else:
        print("[INFO] Cannot run detection, models did not load properly.")
else:
    print(f"[ERROR] Test image file not found at: {test_image_path}")
    print("[INFO] Please update the 'test_image_path' variable in the cell above.")

'''
```

Figure 3.9: 9

Chapter 4

Conclusion

The emotion detection system presented in this project effectively utilizes deep learning techniques to recognize human emotions based on facial expressions. By integrating computer vision and convolutional neural networks (CNNs), the system is capable of detecting faces in real-time, preprocessing images, and accurately classifying emotions into seven categories: Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral.

The project successfully implements key modules, including face detection, image preprocessing, model training, and real-time emotion recognition. The use of OpenCV for face detection and TensorFlow/Keras for deep learning ensures high performance and adaptability. Furthermore, enhancements such as low-light compensation through CLAHE and dropout layers in the CNN model contribute to improved accuracy and robustness.

Despite its effectiveness, the system has certain limitations, including potential misclassification in cases of occlusion, extreme lighting conditions, or subtle facial expressions. Future improvements can focus on integrating advanced models like ResNet or EfficientNet, using larger and more diverse datasets, and incorporating multimodal analysis (e.g., voice or physiological signals) for more accurate emotion detection.

Overall, this project demonstrates the potential of artificial intelligence in human-computer interaction, mental health analysis, and behavioral studies. With further enhancements, it can be applied to real-world scenarios such as emotion-aware chatbots, sentiment analysis in customer service, and mental health monitoring.

Chapter 5

Bibliography

K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.
Keywords: Training; Degradation; Complexity theory; Image recognition; Neural networks; Visualization; Image segmentation.

P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001)*, Kauai, HI, USA, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990517.
Keywords: Object detection; Face detection; Pixel; Detectors; Filters; Machine learning; Image representation; Focusing; Skin; Robustness.

I. T. Gurbuz, M. Lehtonen and A. Belahcen, "Lightning Over-voltages in Nuclear Power Plants," *2019 19th International Symposium on Electromagnetic Fields in Mechatronics, Electrical and Electronic Engineering (ISEF)*, Nancy, France, 2019, pp. 1-2, doi: 10.1109/ISEF45929.2019.9096989.
Keywords: Surges; Lightning; Arresters; Poles and towers; Capacitance; Analytical models; Basic Insulation Level; Lightning; Nuclear Power Plant; Over-Voltage; Surge Arrester; Surge Capacitance.

K. Zhang, Z. Zhang, Z. Li and Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," in *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499-1503, Oct. 2016, doi: 10.1109/LSP.2016.2603342.
Keywords: Face; Face detection; Training; Convolution; Detectors; Computer architecture; Benchmark testing; Cascaded convolutional neural network (CNN); Face alignment; Face detection.

A. Mollahosseini, B. Hasani and M. H. Mahoor, "AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild," in *IEEE Transactions on Affective Computing*, vol. 10, no. 1, pp. 18-31, Jan.-March 2019, doi: 10.1109/TAFFC.2017.2740923.
Keywords: Databases; Computational modeling; Face; Face recognition; Affective computing; Magnetic heads; Affective computing in the wild; Facial expressions; Continuous dimensional space; Valence; Arousal.