FaceFeel : Real-time Facial Emotion Detection System

A PROJECT REPORT BY KARTIK SONI (E23CSEU2113)

A blue and red text on a black background

Description automatically generated

SUBMITTED TO

SCHOOL OF COMPUTER SCIENCE ENGINEERING AND TECHNOLOGY, BENNETT UNIVERSITY

GREATER NOIDA, 201310, UTTAR PRADESH, INDIA

April 2025

# DECLARATION

I hereby declare that the work which is being presented in the report entitled " FaceFeel : Real-time Facial Emotion Detection System " for the subject CSET301: Artificial Intelligence and Machine learning , is an authentic record of my own work carried out during the period from JAN, 2025 to April, 2025 at School of Computer Science and Engineering and Technology, Bennett University Greater Noida.

The matters and the results presented in this report has not been submitted by me/us for

the award of any other degree elsewhere.

Signature of Candidate

**KARTIK SONI (E23CSEU2113)**

# ACKNOWLEDGEMENT

I would like to take this opportunity to express our deepest gratitude to our mentor, **Dr. Nitin Arvind Shelke** for guiding, supporting, and helping me in every possible way. I was extremely fortunate to have him as our mentor as he provided insightful solutions to problems faced by us thus contributing immensely towards the completion of this capstone project. We would also like to express our deepest gratitude to VC, DEAN, HOD, faculty members and friends who helped us in successful completion of this capstone project.

Signature of Candidate

KARTIK SONI (E23CSEU2113)

### Introduction

Welcome to FaceFeel!

FaceFeel is an innovative application that brings emotion recognition technology to your desktop. Using advanced computer vision and deep learning techniques, FaceFeel can detect human faces and identify their emotional expressions in real-time.

### What is FaceFeel?

FaceFeel is a user-friendly desktop application that uses your computer’s webcam to analyze facial expressions and classify them into seven core emotional states:

-😠 Anger

– 😫 Disgust

- 😨 Fear

- 😄 Happiness

- 😢 Sadness

- 😲 Surprise

- 😐 Neutral

### 

### Key Features

* Real-time Detection: Instantly analyze and display emotions as they change
* Simple Interface: Easy-to-use interface with intuitive controls
* Non-invasive: Works with standard webcam hardware
* Privacy-focused: All processing happens locally on your device

### 

### Getting Started

1. Ensure your webcam is connected and functioning
2. Launch the FaceFeel application
3. Click “Start Detection” to begin emotion recognition
4. Position your face within view of the camera
5. Your detected emotion will display on screen
6. Click “Stop Detection” when finished

### 

### System Requirements

* Python 3.7 or higher
* Webcam or camera device
* Required libraries: TensorFlow, OpenCV, NumPy, PIL, Tkinter

### 

### Executive Summary

FaceFeel is a real-time facial emotion detection application that utilizes computer vision and machine learning techniques to identify and classify human emotions from facial expressions. The application uses a webcam to capture video input, processes each frame to detect faces, and then applies a trained neural network model to classify the emotional state of the detected faces. The system can recognize seven distinct emotional states: Anger, Disgust, Fear, Happiness, Sadness, Surprise, and Neutral.

### 

### ntroduction

Facial emotion recognition has significant applications in various domains including human-computer interaction, mental health monitoring, customer experience analysis, and security systems. This project aims to develop a user-friendly application that can detect emotions in real-time, providing instant feedback on the emotional state of the user or subject.

### 

### Technical Architecture

The FaceFeel application is built using the following technologies:

1. Python - The core programming language used for development

2. OpenCV - For image processing and face detection capabilities

3. TensorFlow/Keras - For loading and utilizing the pre-trained deep learning mode

l 4. Tkinter - For creating the graphical user interface

### 

### System Components

#### 1. Face Detection Module

The application uses OpenCV’s Haar Cascade classifier (haarcascade\_frontalface\_default.xml) to detect faces in the video stream. This is a machine learning-based approach where a cascade function is trained from numerous positive and negative images, and then used to detect objects in other images.

#### 

#### 2. Emotion Recognition Model

The core of the system is a Convolutional Neural Network (CNN) model trained to recognize emotions from facial expressions. The model was pre-trained and is loaded from the file facefeel\_model.keras. It classifies detected faces into seven emotional categories: - Anger - Disgust - Fear - Happiness - Sadness - Surprise - Neutral

#### 

#### 3. User Interface

The application features a simple and intuitive graphical user interface built with Tkinter that includes: - A video display showing the camera feed with bounding boxes around detected faces - A label displaying the currently detected emotion - Start and Stop buttons to control the detection process

### 

### Workflow

1. Initialization: The application loads the pre-trained emotion recognition model and establishes a connection to the webcam.
2. Face Detection: For each frame captured by the webcam, the application:
   * Converts the image to grayscale
   * Uses the Haar Cascade classifier to detect faces
   * Draws a blue rectangle around each detected face
3. Emotion Recognition:
   * Extracts the detected face region
   * Resizes it to 48x48 pixels (the input size expected by the model)
   * Converts to grayscale
   * Normalizes pixel values to the range [0,1]
   * Feeds the processed face image to the CNN model
   * Retrieves the predicted emotion with the highest confidence score
4. Display: The application displays the processed video feed with facial detection rectangles and the predicted emotion label.

### 

### Implementation Details

The implementation follows a straightforward approach: 1. The video capture is handled using OpenCV’s VideoCapture class 2. Face detection is performed on each frame using the Haar Cascade classifier 3. Detected faces are preprocessed to match the input requirements of the emotion recognition model 4. The model predicts the emotion for each face 5. Results are displayed in real-time through the Tkinter interface

Code Architecture

The main application consists of several key components:

# Model loading  
model = load\_model('model/facefeel\_model.keras')  
emotion\_labels = ['Anger', 'Disgust', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral']  
  
# Video capture initialization  
cap = cv2.VideoCapture(0)  
  
# GUI components  
window = tk.Tk()  
label = Label(window) # For video display  
emotion\_label = Label(window, text="Emotion: ") # For emotion text display  
  
# Core detection function  
def start\_detection():  
 # Process frames from video capture  
 # Detect faces using Haar Cascade  
 # Process each face and predict emotion  
 # Update display with results  
  
# Control functions  
def stop\_detection():  
 # Release resources and close application

### 

### Training the Model

While the provided code uses a pre-trained model, the typical process for training an emotion recognition model involves: 1. Data Collection: Gathering a diverse dataset of facial expressions labeled with emotions 2. Data Preprocessing: Standardizing images through face detection, alignment, resizing, and normalization 3. Model Architecture: Designing a CNN architecture suitable for image classification tasks 4. Training: Optimizing model parameters using labeled training data 5. Validation: Testing performance on separate validation data to ensure generalization 6. Fine-tuning: Adjusting hyperparameters to improve accuracy and performance

The model expects 48x48 pixel grayscale images as input, suggesting it may have been trained on a standard emotion recognition dataset like FER-2013.

### 

### Limitations and Future Improvements

Based on the current implementation, potential limitations and areas for improvement include: 1. Performance: The application might experience reduced frame rates on lower-end hardware due to the computational demands of running both face detection and emotion recognition in real-time. 2. Accuracy: The emotion recognition model’s accuracy depends on its training data. It may perform better on certain demographics or in specific lighting conditions. 3. Multiple Face Handling: The current implementation processes all detected faces, but the emotion display shows only the most recently processed face’s emotion.

Future enhancements could include: - Support for tracking and displaying emotions for multiple faces simultaneously - Adding confidence scores alongside emotion predictions - Implementing emotion history tracking and statistics - Adding options to adjust detection sensitivity - Including a feature to record and save sessions

### 

### Applications and Use Cases

The FaceFeel application has potential applications in various domains: 1. Human-Computer Interaction: Enhancing user experience by adapting interfaces based on detected emotions 2. Mental Health Monitoring: Assisting in emotional state tracking for therapeutic purposes 3. Education: Supporting teachers in gauging student engagement and emotional responses 4. Market Research: Analyzing consumer reactions to products or advertisements 5. Security and Surveillance: Identifying potentially concerning emotional states in security contexts

## 

## User Guide

### 

### Installation

1. Clone the repository or download the source code
2. Install required dependencies:

* pip install opencv-python tensorflow pillow numpy

1. Ensure you have the pre-trained model file facefeel\_model.keras in the model directory

### Running the Application

1. Execute the main script:

* python facefeel\_detect.py

1. When the application window appears, click “Start Detection”
2. Position yourself in front of the camera
3. Your detected emotion will display on screen
4. Click “Stop Detection” to end the session and close the application

### Troubleshooting

* No camera feed: Ensure your webcam is connected and not being used by another application
* Poor detection: Improve lighting conditions and ensure your face is clearly visible
* Performance issues: Close other resource-intensive applications

## 

## Conclusion

FaceFeel demonstrates the effective application of computer vision and deep learning techniques to create a practical tool for real-time emotion recognition. The system successfully integrates face detection, emotion classification, and user interface components to provide instant feedback on emotional states detected from facial expressions.

The modular design of the application allows for future enhancements and adaptations to specific use cases, making it a versatile foundation for emotion-aware systems.

## 

## References

1. OpenCV Documentation: https://docs.opencv.org/
2. TensorFlow Documentation: https://www.tensorflow.org/api\_docs
3. Facial Expression Recognition research: https://arxiv.org/abs/1804.08348
4. FER-2013 Dataset: https://www.kaggle.com/datasets/msambare/fer2013
5. Haar Cascade Classification: <https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html>

## Appendix A: Future Development Roadmap

### Short-term Improvements (1-3 months)

* Add multiple face tracking with individual emotion labels
* Implement emotion confidence scores display
* Create emotion history tracking and basic statistics
* Improve UI with emotion icons and better visualization

### Medium-term Goals (3-6 months)

* Build a more robust model with improved accuracy across demographics
* Add video recording and playback with emotion overlay
* Develop batch processing for recorded videos
* Create documentation and developer API

### Long-term Vision (6+ months)

* Integrate with other applications via SDK
* Develop cross-platform mobile versions
* Add emotion-based recommendation systems
* Explore gesture and posture recognition integration

## 

## Appendix B: Technical Specifications

### Model Architecture

The emotion recognition model uses a Convolutional Neural Network architecture: - Input: 48x48 grayscale images - Multiple convolutional layers with ReLU activation - Max pooling layers for dimensionality reduction - Dropout layers to prevent overfitting - Fully connected layers - Softmax output layer with 7 classes

### Hardware Requirements

* Processor: Dual-core CPU or better
* RAM: 4GB minimum, 8GB recommended
* Camera: Standard webcam (720p resolution or higher recommended)
* Graphics: Integrated graphics sufficient, dedicated GPU recommended for optimal performance