



NOMBRE DEL ALUMNO:

GARCIA SANCHEZ SERGIO JESUS
VALENZUELA BERRELLEZA CESAR JESUS

NOMBRE DEL MAESTRO:

ZURIEL DATHAN MORA FELIX

MATERIA:

INTELIGENCIA ARTIFICIAL

FECHA DE ENTREGA:

29/03/2025

INTRODUCCION

Este proyecto consiste en el desarrollo de un sistema que permite detectar emociones humanas en tiempo real utilizando visión por computadora y redes neuronales convolucionales. Se entrena un modelo con imágenes clasificadas por emociones y se utiliza OpenCV para detectar rostros y predecir emociones con un modelo previamente entrenado.

El sistema está diseñado para identificar cinco emociones fundamentales:

- Angry
- Disgust
- Happy
- Sad
- Surprise

Para ello, se entrena un modelo de red neuronal convolucional (CNN) que aprende a reconocer patrones faciales asociados a cada emoción utilizando un conjunto de imágenes previamente clasificadas.

Dataset: Se uso el dataset FER-2013 del cual solo utilizamos la carpeta de entrenamiento con las clases ya mencionadas

Preprocesamiento:

- Captura de imagen
- Conversion a escala de grises
- Detección de rostros
- Redimensionamiento 48x48 pixeles

Tecnologías y librerías

- Python3
- TensorFlow
- OpenCV
- NumPy

CLASE main:

LIBRERIAS:

- from emotion_model import EmotionModel
- from emotion_detector_app import EmotionDetectorApp
- import os

EXPLICACION:

```
def main():
    # Ruta al dataset organizado por emoción
    data_dir = r"C:\Users\Sergi\Desktop\Emociones\train"
    labels = ['angry', 'disgust', 'happy', 'sad', 'surprise']
    model_path = "emotion_model.h5"

    # Paso 1: Entrenar el modelo
    model_trainer = EmotionModel(data_dir, labels, model_path)
    model_trainer.build_model()
    model_trainer.train()

    # Paso 2: Detectar emociones en tiempo real
    detector = EmotionDetectorApp(model_path, labels)
    detector.run()

if __name__ == "__main__":
    main()
```

CLASE emotion_model:

LIBRERIAS:

- import os
- import numpy as np
- import tensorflow as tf
- from tensorflow.keras.preprocessing.image import ImageDataGenerator
- from tensorflow.keras.models import Sequential
- from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

Explicación:

Propósito: Entrenar un modelo CNN para clasificar emociones a partir de imágenes.

-Métodos:

`build_model()`: Construye la arquitectura del modelo.

```
def build_model(self):
    model = Sequential([
        Conv2D(32, (3, 3), activation='relu', input_shape=(48, 48, 1)),
        MaxPooling2D(2, 2),
        Conv2D(64, (3, 3), activation='relu'),
        MaxPooling2D(2, 2),
        Flatten(),
        Dense(128, activation='relu'),
        Dropout(0.5),
        Dense(len(self.labels), activation='softmax')
    ])
    model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])
    self.model = model
```

`train()`: Entrena el modelo con imágenes y guarda el archivo en formato .h5

```
def train(self, epochs=10, batch_size=32):
    datagen = ImageDataGenerator(rescale=1./255, validation_split=0.2)
    train_gen = datagen.flow_from_directory(
        self.data_dir,
        target_size=(48, 48),
        color_mode='grayscale',
        batch_size=batch_size,
        class_mode='categorical',
        subset='training'
    )
    val_gen = datagen.flow_from_directory(
        self.data_dir,
        target_size=(48, 48),
        color_mode='grayscale',
        batch_size=batch_size,
        class_mode='categorical',
        subset='validation'
    )

    self.model.fit(train_gen, validation_data=val_gen, epochs=epochs)
    self.model.save(self.model_path)
    print(f'Modelo guardado en: {self.model_path}')
```

CLASE: emotion_detector_app

LIBRERIAS:

- import cv2
- import numpy as np
- from tensorflow.keras.models import load_model

Propósito: Detectar emociones en tiempo real usando la cámara web.

-Métodos:

preprocess_face(face): Preprocesa la imagen del rostro para predecir.

```
def preprocess_face(self, face):
    face = cv2.resize(face, (48, 48))
    face = face.astype('float32') / 255.0
    face = np.expand_dims(face, axis=0)
    face = np.expand_dims(face, axis=-1)
    return face
```

run(): Captura la cámara y muestra la emoción detectada en pantalla.

```
def run(self):
    cap = cv2.VideoCapture(0)
    print("Presiona 'q' para salir.")
    while True:
        ret, frame = cap.read()
        if not ret:
            break
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

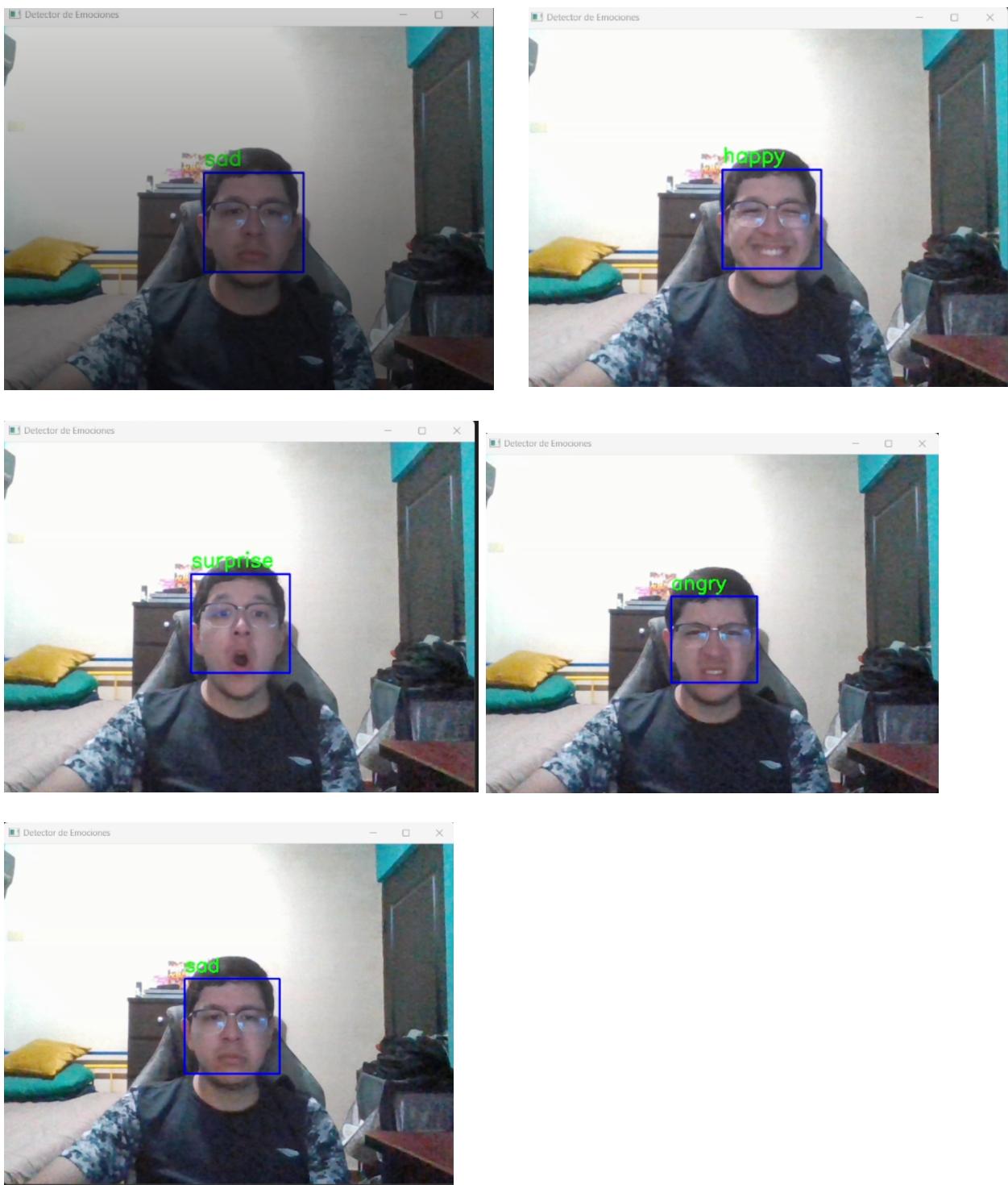
        faces = self.face_cascade.detectMultiScale(gray, scaleFactor=1.3,
minNeighbors=5)
        for (x, y, w, h) in faces:
            face_img = gray[y:y+h, x:x+w]
            processed = self.preprocess_face(face_img)
            prediction = self.model.predict(processed)[0]
            emotion = self.labels[np.argmax(prediction)]
            confidence = round(np.max(prediction) * 100, 2)

            cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)
            cv2.putText(frame, emotion, (x, y-10),
cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2)

        cv2.imshow('Detector de Emociones', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cap.release()
    cv2.destroyAllWindows()
```

Pruebas:



Link:

<https://drive.google.com/file/d/1BqosEur9f39s-XgPKBMzr1wgGxZUa6pc/view?usp=sharing>