Detección de emociones en rostros

Natalia Yue Gallardo Pretel
nataliayue@correo.ugr.es

Pablo Martín Palomino
pablomarpa@correo.ugr.es

Pablo Reyes Sousa
sousareyp@correo.ugr.es

Resumen

Este trabajo presenta el diseño e implementación de un sistema de análisis facial en tiempo real. Utilizando como base la arquitectura YOLOv8, se realiza una evaluación en cascada formada por dos etapas: localización del rostro y clasificación de emociones en todas las caras detectadas. También se explora otros modelos con intenciones similares a nuestro objetivo.

1. Introducción

El objetivo de este proyecto es el diseño e implementación de un sistema de análisis facial, capaz de detectar y clasificar emociones humanas en tiempo real mediante la webcam. El sistema debe ser capaz de localizar rostros (permitiendo detección múltiple) e indentificar la emoción que subyace en los mismos.

Se busca alcanzar un alto grado de confianza, ya que, al permitir a las máquinas intepretar cómo se siente una persona, estas simulen un comportamiento más empático con el usuario.

Más allá del Aprendizaje Automático, nos centramos en:

- **Evaluación de modelos en cascada:** El modelo tiene una fase inicial en la que se reconoce la localización de los rostros y una segunda etapa en la que se realiza la detección de emociones

- **Optimización en tiempo real:** Determinar una tasa de muestreo equilibrada, de manera que el proceso sea fluido pero que el consumo de recursos no sea extremadamente costoso.

2. Background

[9] [15] Para realizar el trabajo, ha sido clave el uso de la arquitectura de YOLO (You Only Look Once), presentada en 2016. La diferencia fundamental de esta arquitectura respecto a otras de su época como R-CNN, la cual detectaba posibles zonas de información en la imagen y después ejecutaba un clasificador en esas zonas; es que YOLO reúne toda la ejecución en una única red convolucional (1).

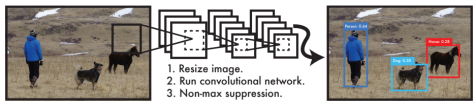

Figura 1: Sistema de Detección YOLO. Procesamiento de imágenes con YOLO. Redimensiona (1) la entrada a 448×448, (2) ejecuta una única red convolucional y (3) umbraliza las detecciones resultantes por la confianza del modelo.

Internamente, YOLO divide la entrada en una malla $S \times S$ cuyo centro se encarga de detectar el objeto. Dentro de la malla, cada celda predice las cajas de detección y la confianza de estas cajas (refleja cómo de seguro está el modelo de que haya algo en esa caja). A cada caja se le asocia 5 parámetros: las coordenadas del centro, sus dimensiones y la confianza, la cual se define por $Probabilidad(Objeto) * IOU_{pred}^{truth}$, donde IOU es el cociente de la interseccion del área predicha con el área real y la unión de ambas áreas.

Cada celda también predice clases de probabilidad $Probabilidad(Clase_i|Objeto)$. Multiplicando estos dos números obtenemos la confianza de una clase de dada una caja, prediciendo así el objeto. **??**

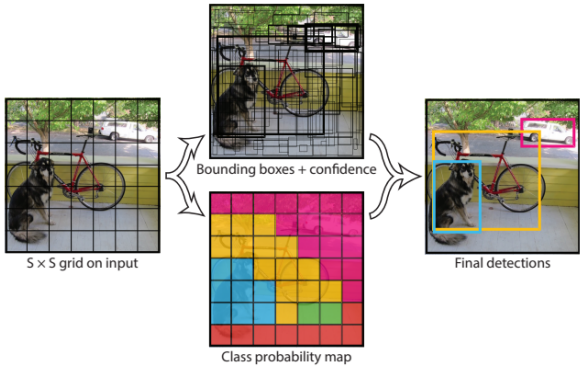

Figura 2: Funcionamiento modelo de detección

Si bien YOLO fue un avance en el ámbito de detección en imágenes en tiempo real (mayor rapiudez que otros modelos), tenía más problemas de localización de objetos pero a cambio daba menos falsos positivos en detec-

tar background (malinterpretar objetos como background). Otro problema de YOLO (original) es en la detección de objetos pequeños.

3. Estado del Arte

El reconocimiento automático de caras y emociones faciales ha sido un área de investigación activa durante las últimas décadas, impulsada por aplicaciones en interacción humano–computador, vigilancia inteligente, análisis de comportamiento y sistemas afectivos. En los últimos años, el auge del aprendizaje profundo ha permitido avances significativos tanto en la detección facial como en el reconocimiento de emociones, especialmente mediante redes neuronales convolucionales.

3.1. Detección de caras

Los primeros enfoques para la detección de caras se basaban en métodos clásicos de visión por computador. Uno de los más representativos es el algoritmo de Viola–Jones [14]. A pesar de su eficiencia, este método presenta limitaciones importantes frente a variaciones de pose, iluminación y oclusiones.

Con la introducción del aprendizaje profundo, surgieron modelos basados en CNN que mejoraron notablemente la robustez y precisión. Entre ellos destaca MTCNN, una arquitectura en cascada compuesta por tres redes que realiza de forma conjunta la detección y alineación facial [16]. Posteriormente, detectores de una sola etapa como RetinaFace lograron resultados de alta precisión incluso en escenarios complejos, como caras pequeñas o con poses extremas [2].

Paralelamente, detectores genéricos de objetos como YOLO (You Only Look Once) han sido adaptados exitosamente a la detección facial o de personas en general mediante entrenamiento específico [9]. Versiones como YOLOv8 incorporan mejoras en el *backbone*, cabezas desacopladas de detección y mayor eficiencia computacional [6], lo que permite alcanzar un equilibrio favorable entre precisión y velocidad, especialmente en aplicaciones en tiempo real.

3.2. Reconocimiento de emociones faciales

El reconocimiento de emociones faciales (Facial Emotion Recognition, FER) ha evolucionado desde enfoques basados en la extracción manual de características, como Local Binary Patterns (LBP) o Histogram of Oriented Gradients (HOG), combinados con clasificadores tradicionales [11]. Aunque estos métodos sentaron las bases del campo, su capacidad de generalización es limitada.

Los enfoques modernos se basan mayoritariamente en CNN profundas entrenadas sobre imágenes faciales alineadas. Arquitecturas como VGG, ResNet o Inception han sido ampliamente utilizadas para la clasificación de emociones [7]. El desarrollo de conjuntos de datos como FER2013 [4] y AffectNet [8] ha facilitado el entrenamiento y la evaluación de estos modelos, proporcionando etiquetas categóricas de emociones básicas y, en algunos casos, anotaciones continuas.

No obstante, el reconocimiento de emociones sigue siendo un problema desafiante debido a la ambigüedad emocional, el ruido en las etiquetas y la variabilidad entre individuos y contextos culturales [7].

3.3. Enfoques integrados y multitarea

Una tendencia reciente en el estado del arte consiste en integrar la detección facial y el reconocimiento de emociones en una única arquitectura [7]. Estos enfoques multitarea permiten compartir representaciones internas, reduciendo la complejidad del sistema y mejorando la eficiencia computacional.

3.4. Limitaciones y desafíos actuales

A pesar de los avances logrados, persisten diversos desafíos en el reconocimiento automático de emociones faciales, como la dificultad para reconocer emociones sutiles o compuestas, la sensibilidad a condiciones de iluminación adversas, oclusiones y variaciones étnicas, así como la limitada capacidad de generalización entre distintos conjuntos de datos [7]. Además, existe un compromiso inherente entre precisión y latencia que resulta crítico en aplicaciones en tiempo real.

3.5. Tendencias recientes y modelos emergentes

De forma reciente, se han anunciado nuevas evoluciones de la familia YOLO orientadas a mejorar la eficiencia y la facilidad de despliegue en dispositivos de borde y entornos con recursos limitados. Entre ellas destaca YOLO26, una arquitectura aún en desarrollo que propone un rediseño extremo a extremo del proceso de detección, eliminando etapas tradicionales de post-procesamiento como la supresión no máxima (Non-Maximum Suppression, NMS).

Este enfoque end-to-end, introducido inicialmente en versiones previas experimentales de YOLO, busca simplificar el pipeline de inferencia, reducir la latencia y facilitar la integración en sistemas de producción. Asimismo, YOLO26 incorpora innovaciones en el proceso de entrenamiento, como el uso del optimizador MuSGD, un método híbrido inspirado en avances recientes en la optimización de modelos de lenguaje de gran escala, con el objetivo de mejorar la estabilidad y la velocidad de convergencia durante el entrenamiento.

Aunque los modelos YOLO26 aún no han sido publicados oficialmente y sus resultados deben considerarse preliminares, estas propuestas reflejan una tendencia clara en el estado del arte hacia arquitecturas más simples, eficientes y orientadas a la implementación directa en dispositivos de bajo consumo. Estas líneas de investigación refuerzan la

relevancia de enfoques basados en detectores unificados y eficientes no solo de emociones claro esta, sino de objetos.

La información sobre esta futura arquitectura se puede consultar de forma más detallada aquí: https://docs.ultralytics.com/es/models/yolo26/#overview

3.6. Posicionamiento del presente trabajo

Este trabajo propone una arquitectura basada en YOLOv8 para la localización facial, integrada en un flujo de trabajo en cascada con otro modelo YOLOv8 para el reconocimiento de emociones. Aunque las soluciones de un solo paso son el estándar en velocidad, el enfoque modular adoptado en este proyecto facilita el ajuste independiente de los pesos de los modelos, garantizando una gran robustez ante variaciones en la pose y la iluminación.

4. Métodos

Se optó por implementar una arquitectura en cascada con dos modelos independientes debido entre otros motivos a la carencia de un dataset unificado que cubriera ambas necesidades.

La mayoría de los conjuntos de datos disponibles se limitan a la detección de rostros en entornos complejos o a la clasificación de emociones en imágenes ya recortadas, pero rara vez combinan una localización precisa mediante bounding boxes con un etiquetado emocional fiable en escenarios no controlados.

Bajo esta premisa, el sistema se ha estructurado en dos fases difrenciadas y secuenciales: detección de rostros y clasificación emocional.

4.1. Detección y Localización facial

Esta primera etapa se centra en la extracción de regiones de interés a partir de las imágenes (los vídeos son secuencias de fotogramas). Para ello, se usa el modelo `YOLOv8`. Concretamente, trabajamos con la versión "Small", pues está diseñada para equilibrar velocidad y precisión.

Los pasos realizados son los siguientes:

1. El modelo procesa imágenes de tamaño 640×640, pues el valor para el que están optimizados los modelos de la arquitectura `YOLOv8`. Aunque la webcam usada tenga otra resolución, la librería `ultralytics` redimensionará las entradas. También se escalan los valores de los píxeles al rango $[0, 1]$ antes de pasarlos a la red.

2. Una vez detectado el rostro, el sistema extrae las coordenadas de la *bounding box* y realiza un proceso de *clamping*, pues necesitamos la cara completa para la detección.

El modelo de esta primera fase es llamado `YOLOv8_face`

4.2. Clasificación Emocional

Una vez localizado el rostro, el objetivo es interpretar la expresión facial. Para ello, usamos el recorte obtenido en la primera fase como entrada de `YOLOv8_emotion`.

De nuevo tenemos el mismo preprocesamiento automático de `YOLO`, es decir, reescalado automático de intensidad e imágenes de 640×640.

La salida de esta segunda fase consiste en:

- Categoría de la emoción: índice entero que corresponde con la emoción detectada.

- Puntuación de confianza: Para cada categría, valor que indica cómo de seguro está el modelo de que el rostro pertenece a dicha categoría.

4.3. Qué es y cómo funciona YOLOv8 [5]

YOLOv8 fue lanzada por Ultralytics en enero del 2023. Se trata de un modelo de detección de objetos en "una sola etapa"tal y como indica su nombre. Esto se consigue procesando la imagen a través de una red neuronal profunda.

Los pilares de funcionamiento son:

- **Anchor-Free** Es la parte más novedosa, pues ya no se comparan las imágenes con "plantillas"predefinidas (cajas con tamaño fijo). Esto es debido a que era muy difícil tener plantillas que se pudieran usar para cualquier objeto. En su luga, la red predice de forma matemática el centro del objeto y sus dimensiones.

- **Módulo C2f y Stem:** El modelo sustituye el antiguo módulo C3 por el C2f, que mejora el flujo de información combinando características de alto y bajo nivel mediante más conexiones internas. Además, se modifica la capa de entrada (*Stem*): donde antes se usaba un filtro de 6×6 (que reducía la imagen muy rápido), ahora se emplea una convolución de 3×3, permitiendo conservar mejor la información fina y los detalles desde el inicio.

- **Non-Maximum-Suppression:** Es el algoritmo que se ejecuta al final de la inferencia para filtrar el exceso de detecciones. Dado que la red propone miles de cajas candidatas para un mismo objeto, el NMS selecciona la que tiene mayor puntuación de confianza y elimina (suprime) todas las cajas vecinas que se superponen excesivamente con ella (basándose en el índice IoU), dejando una única detección limpia y precisa por objeto.

- **Mosaic Augmentation:** En vez de entrenar una red mostrándole una foto cada vez, se toman imágenes aleatorias del dataset, se redimensionan y se pegan formando un mosaico de 2×2, obteniendo como resultado

una imagen con 4 objetos mezclados y de distintos tamaños. Esta técnica permite que el modelo se centre en el objeto y no en su entorno (no detectar camellos solamente porque estén rodeados de arena o barcos porque el fondo sea azul) y evita problemas de escala (al componer el mosaico, los objetos se reducnen de tamaño, por lo que la red aprenderá a detectar cosas más pequeás o lejanas). YOLOv8 mejora esta técnica, pues en la realidad no nos vamos a encontrar imágenes "con recortes". Para ello, emplea la siguiente estrategia:

1. Durante las primeras fases de entrenamiento, usa mosaic, permitiendo aprender robutez.
2. En las últimas 10 épocas, deja de ver mosaicos y pasa a ver imágenes reales y completas, para así ajustarse mejor a la realidad.

4.4. Entrenamiento en servidores y ejecución local

No se ha podido usar la herramienta Google Colab ni para el entrenamiento ni para la ejecución debido a:

- **Entrenamiento costoso:** los recursos de Google Colab están limitados, incluso con una licencia, los entrenamientos necesitan más que lo que esta herramienta nos ofrece. Se tuvo que hacer uso de los servidores de la UGR.
- **Tiempo Real:** La arquitectura de red de Colab no ofrece una dirección IP pública persistente, obligando servicios de tunelización intermedios. Esto introduce una latencia de red variable y significativa que es incompatible con un sistema de tiempo real.

5. Experimentos

En esta sección se detalla el protocolo de validación, las métricas empleadas, los experimentos realizados y el análisis de los resultados obtenidos para los modelos de detección de rostros y reconocimiento de emociones.

5.1. Protocolo de Validación Experimental

El proceso de validación se fundamentó en una estrategia de *Transfer Learning* utilizando la arquitectura **YOLOv8s**. El protocolo se dividió en dos fases secuenciales:

1. **Detección de Rostros:** Entrenamiento con una única clase (`face`) para la localización precisa de la región de interés (ROI).
2. **Reconocimiento de Emociones:** Entrenamiento sobre el segundo dataset con 8 categorías: *Anger, Contempt, Disgust, Fear, Happy, Neutral, Sad* y *Surprise*.

Ambos modelos fueron entrenados durante 100 épocas con un tamaño de lote (*batch size*) de 16 y una resolución de entrada de 640×640 píxeles. Se utilizó el optimizador **SGD** con una tasa de aprendizaje inicial de 0.001. En caso de querer replicarse el entrenamiento un posible script para lanzar mediante sbatch en servidor sería el siguiente, claro esta este se deberia ejecutar sobre un entorno con Ultralytics instalado y además las rutas de los datos se deberián modificar a donde esten descargados en los .yaml de cada modelo para que los códigos de entrenamiento de cada modelo train_yolo_x.py los ubiquen:

Listing 1. Script de entrenamiento para YOLOv8 en SLURM

```
#!/bin/bash
#SBATCH --job-name=Yolo_emotion
    # Nombre del job
#SBATCH --partition=dios
    # Cola de tu cluster
#SBATCH --gres=gpu:1
    # Numero de GPUs a usar
#SBATCH --time=28:00:00
    # Tiempo maximo (opcional)
#SBATCH --mem=16G
    # Memoria maxima
module load nvhpc/21.3
# Activar Conda desde entorno con
    requisitos como ultralytics en carpeta
    especifica
export PATH="/opt/anaconda/anaconda3/bin:
    $PATH"
eval "$(conda shell.bash hook)"
conda activate /mnt/homeGPU/usuario/entorno
# Ejecutar el script Python
python /mnt/homeGPU/usuario/
    train_yolo_emotion.py
```

5.2. Data Augmentation

Para maximizar la capacidad de generalización de los modelos y mitigar el riesgo de sobreajuste (*overfitting*), se habilitó el parámetro `augment=True` durante el proceso de entrenamiento. Esta configuración activa un flujo de transformaciones aleatorias aplicadas de forma dinámica en cada *batch*:

- **Mosaic Augmentation:** Técnica que concatena cuatro imágenes distintas en una sola composición como ya se ha explicado antes, mejorando la detección de rostros en diversas escalas y contextos espaciales. [12]
- **Transformaciones de Color (HSV):** Ajustes estocásticos en el matiz, saturación y brillo para garantizar robustez ante variaciones en la iluminación ambiental. [12]
- **Transformaciones Geométricas:** Inversiones horizontales (*flips*) y variaciones de escala que permiten

al modelo reconocer caras y expresiones faciales independientemente de la orientación o distancia de la captura. [12]

Gracias a estas técnicas, el modelo no solo aprende las características estáticas del dataset, sino que desarrolla invarianza ante las perturbaciones visuales.

5.3. Métricas de Evaluación [13]

Para cuantificar el rendimiento del modelo, se emplearon las siguientes métricas estándar en la detección de objetos. Estas se basan en la comparación entre las predicciones del modelo y las etiquetas reales, clasificándolas como Verdaderos Positivos (TP), Falsos Positivos (FP) o Falsos Negativos (FN).

- **Intersección sobre Unión (IoU):** Mide la superposición entre la caja delimitadora predicha (B_p) y la real (B_g).

$$IoU = \frac{\text{Área}(B_p \cap B_g)}{\text{Área}(B_p \cup B_g)} \tag{1}$$

- **Precisión (P):** Indica la fracción de detecciones positivas que son correctas.

$$P = \frac{TP}{TP + FP} \tag{2}$$

- **Exhaustividad (Recall, R):** Mide la capacidad del modelo para detectar todas las instancias positivas reales.

$$R = \frac{TP}{TP + FN} \tag{3}$$

- **mAP@n:** La Precisión Media Promedio (*mean Average Precision*) es el promedio de la precisión media (AP) calculada para todas las clases con un umbral de IoU de n. Para una clase dada, el AP se calcula como el área bajo la curva de precisión-exhaustividad $p(r)$:

$$AP = \int_0^1 p(r)\, dr \quad \Rightarrow \quad mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{4}$$

  Donde N es el número total de clases.

- **F1-Score:** Es la media armónica de la precisión y la exhaustividad, proporcionando una métrica única que equilibra ambos valores.

$$F1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{5}$$

- **Funciones de Pérdida:** Durante el entrenamiento se minimiza una función de pérdida compuesta:
  - **Box Loss:** calculada mediante el error de IoU Completo (CIoU), que penaliza la distancia, escala y aspecto de las cajas.
  - **Cls Loss:** Pérdida de clasificación que utiliza Entropía Cruzada Binaria para verificar la exactitud de la categoría predicha.
  - **DFL Loss:** La *Distribution Focal Loss* optimiza la probabilidad de que los bordes de la caja predicha estén cerca de los bordes reales.

5.4. Validación Cualitativa e Implementación Individual

Con el objetivo de complementar el análisis cuantitativo y verificar el desempeño de los modelos en entornos no controlados, se desarrollaron dos herramientas de validación denominadas `pruebaModeloDeteccionCaras.py` y `pruebaModeloDeteccionEmociones.py`.

Estos scripts permiten realizar inferencias individuales sobre muestras de imagen externas, facilitando una auditoría visual de los resultados.

- **Detección de Rostros (`yolov8_face`):** Se validó la capacidad del modelo para localizar la región de interés (ROI) bajo diversas condiciones de iluminación y oclusión, confirmando la estabilidad de la caja delimitadora (*bounding box*).

- **Reconocimiento de Emociones (`yolov8_emotion`):** Se contrastó la capacidad de clasificación del modelo verificando las predicciones.

Los resultados obtenidos mediante estas pruebas empíricas hechas a mano confirman que ambos modelos poseen una robustez y precisión suficientes para su despliegue en aplicaciones de tiempo real. Ambos códigos se adjuntan en la entrega final, permitiendo la ejecución de pruebas independientes sobre cualquier imagen de prueba.

5.5. Dataset

Se han utilizado estos los siguientes datasets:

- Entrenamiento modelo detección de caras: https://www.kaggle.com/datasets/fareselmenshawii/face-detection-dataset/data

- Entrenamiento modelo detección de emociones: https://www.kaggle.com/datasets/fatihkgg/affectnet-yolo-format/data

Face	9133	2590
Background	1166	0
	Face	**Background**

Tabla 1. Matriz de confusión de detección de caras

5.6. Análisis de resultados

5.6.1 Detección de caras

Los resultados obtenidos son bastante óptimos. Observando 3 vemos como el modelo funciona bien en un gran rango de umbral de confianza y como las gráficas del entrenamiento del modelo convergen de manera adecuada conforme pasan las épocas, obteniendo buenos valores de métricas y reduciendo la función de pérdida, el ajuste fino se puede notar en las últimas épocas, bajando mucho más la función de pérdida.

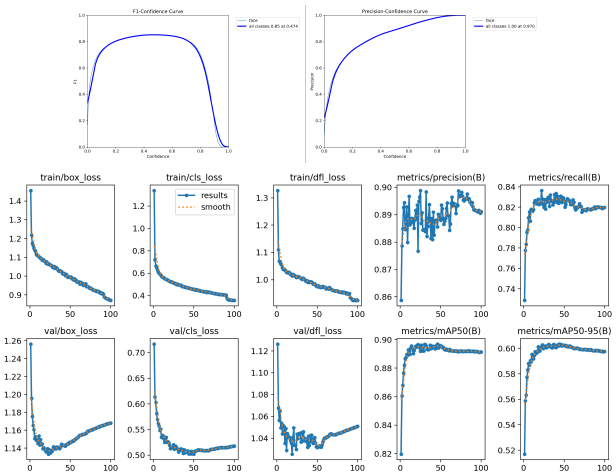

Figura 3: Curvas de resultados de detección de caras

5.6.2 Detección de emociones

Los resultados cuantitativos demuestran una alta eficacia en la clasificación de expresiones faciales. El modelo alcanzó un mAP@0.5 global de 0.828 y un valor F1 máximo de 0.75 con un umbral de confianza de 0.544. El desglose por categorías se presenta en la Tabla 2.

El análisis de la matriz de confusión (4) revela que la clase **Happy** es la que presenta menor ambigüedad, con una precisión casi perfecta. Por el contrario, la clase *Neutral* muestra una tendencia a confundirse con *Sad* y *Anger*, lo cual es explicable debido a la similitud de las microexpresiones faciales en estados de baja activación.

Tabla 2. Rendimiento por Clase del Modelo de Emociones (mAP@0.5)

Emoción	**mAP@0.5**
Happy	0.954
Surprise	0.858
Fear	0.849
Contempt	0.843
Anger	0.826
Disgust	0.807
Sad	0.777
Neutral	0.709
Promedio Global	**0.828**

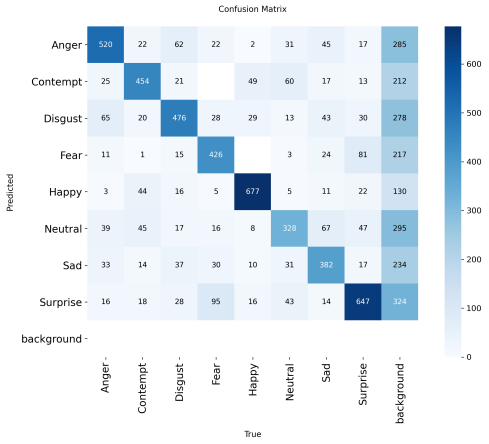

Figura 4: Matriz de confusión de detección de emociones

Las curvas de aprendizaje (5) indican una convergencia estable. Se observa una reducción drástica en la pérdida de clasificación (*Cls Loss*) en las últimas 10 épocas, lo que sugiere que el ajuste fino final fue crucial para la discriminación de categorías complejas. La estabilidad de la métrica mAP por encima de 0.8 confirma la robustez de la arquitectura YOLOv8s para aplicaciones de análisis facial en tiempo real.

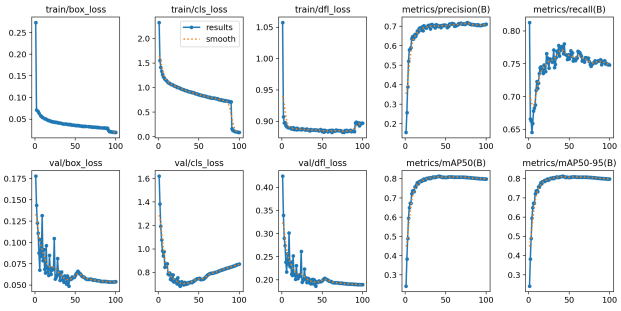

Figura 5: Curvas de aprendizaje de detección de emociones

5.7. Prueba con Retina Face+ DeepFace + VGGFace

5.7.1 RetinaFace [1]

RetinaFace es un modelo de detección de rostros de alta precisión que tiene un buen comportamiento a pesar de cambiar los tamaños de las imágenes, pues realiza procesos de detección jerárquica.

Su arquitectura principal es ResNet y no solo es capaz de dibujar una *bounding box* alrededor de una cara, sino que también calcula puntos clave para los ojos o la boca.

Figura 6: Ejemplo de uso de RetinaFace

La principal ventaja de usar RetinaFace para la detección es la alineación de los rostros, de manera que será más fácil detectar la emoción.

5.7.2 DeepFace, Mini-Xception [10]

DeepFace es una librería de Visión por Computador de python que agrupa varios modelos de inteligencia artificial. Cuando queremos estimar emociones, usa por defecto el modelo VGG-Face.

5.7.3 VGG-Face [3]

VGG-Face es una Red Neuronal Profunda diseñada específicamente para el reconocimiento facial. Debido a que la arquitectura VGG es buena extrayendo características estáticas y estructurales (forma del cráneo, arrugas...) se puede usar para reconocer el género y la edad aproxiamada de una persona.

VGG fue propuesta por Simonyan y Zisserman, de la universidad de Oxford. Destaca por mejorar los modelos anteriores mediante el uso de kernels más pequeños. Además, su preprocesamiento es muy simple, pues solamente hay que restar a cada píxel el valor RGB medio del conjunto de entrenamiento.

Concretamente, usamos la arquitectura VGG-16.

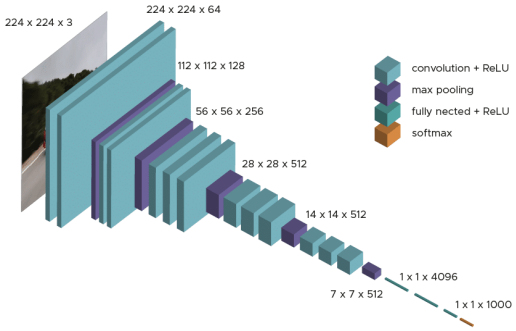

Figura 7: Arquitectura VGG-16

6. Conclusiones

En este trabajo se ha diseñado e implementado un sistema robusto de análisis facial en tiempo real mediante una arquitectura basada en `YOLOv8s`. Los resultados nos han permitido extraer las siguientes conclusiones:

- La detección en dos fases permite abordar el problema de la falta de datasets combinados lo suficientemente diversos, ahora podemos extraer emociones de imágenes en las que aparecen más elementos a parte de una cara. No solo eso, sino que se permite detección múltiple en una sola imagen o fotograma (ya que un vídeo es una secuencia de fotogramas).
- Mientras que para la emoción `happy` es la más distinguible, frente a otras como `sad` y `neutral`
- No se pueden usar los recursos del Google Colab porque el entrenamiento de los modelos es demasiado costoso (se nos retiran los permisos antes de finalizarlo) y la tunelización y conexión entre nuestro hardware y Google requieren un tiempo que no podemos controlar nosotros.

También hemos podido llegar a la conclusión tras comparar con RetinaFace+DeepFace+VGGFace. Que el modelo presentado podría mejorar enormemente incluyendo alineación facial, pues a pesar de que da un buen funcionamiento en general hay casos en los cuales el otro modelo funciona de forma más robusta.

Referencias

[1] David Cochard. Retinaface: A face detection model for high resolution images, 2024. 7

[2] Jiankang Deng, Jia Guo, Yuxiang Zhou, Jinke Yu, Irene Kotsia, and Stefanos Zafeiriou. Retinaface: Single-shot multi-level face localisation in the wild. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5203–5212, 2020. 2

[3] Desconocido. Vgg: ¿qué es este modelo? ¡daniel te lo cuenta todo!, 2022. 7

[4] Ian Goodfellow, Dumitru Erhan, Pierre-Luc Carrier, et al. Challenges in representation learning: A report on three machine learning contests. *Neural Networks*, 64:59–63, 2015. 2

[5] Francesco Jacob Solawetz. What is yolov8? a complete guide, 2024. 3

[6] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Yolov8. *Ultralytics*, 2023. https://github.com/ultralytics/ultralytics. 2

[7] Shan Li and Weihong Deng. Deep facial expression recognition: A survey. *IEEE Transactions on Affective Computing*, 13(3):1195–1215, 2022. 2

[8] Ali Mollahosseini, Behzad Hasani, and Mohammad H. Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing*, 10(1):18–31, 2019. 2

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016. 1, 2

[10] Sefik Ilkin Serengil and Alper Ozpinar. Lightface: A hybrid deep face recognition framework. In *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pages 1–5. IEEE, 2020. 7

[11] Caifeng Shan, Shaogang Gong, and Peter W. McOwan. Facial expression recognition based on local binary patterns: A comprehensive study. *Image and Vision Computing*, 27(6):803–816, 2009. 2

[12] Ultralytics. Train - augmentation settings and hyperparameters, 2024. Documentación oficial de YOLOv8 sobre configuraciones de aumento de datos. 4, 5

[13] Ultralytics. Yolo performance metrics: Choosing the right metrics, 2024. Accedido el 2 de enero de 2026. 5

[14] Paul Viola and Michael 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*, pages 511–518, 2001. 2

[15] Ranjan Sapkota y Manoj Karkee. Ultralytics yolo evolution: An overview of yolo26, yolo11, yolov8, and yolov5 object detectors for computer vision and pattern recognition. *Cornell University*, page 16, 2025. 1

[16] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016. 2