IES Pere Maria Orts

Modelos de Inteligencia Artificial

DeepFace - Detector de sentimientos

Autor:
Kenny Berrones

Profesor:
David Campoy Miñarro

iesperemariaorts

GENERALITAT
VALENCIANA

Índice

1. Introducción

En esta práctica se pide que realicemos diversos experimentos sobre el DeepFace. Esta se trata de una tecnología desarrollada por Facebook en el año 2014, con esta se puede realizar un reconocimiento facial profundo, como puede ser el análisis de emociones.

2. Experimentos

En primer lugar tendremos que instalar esta librería, lo haremos con el siguiente comando:

```
pip install deepface
```

Tras esto tendremos que importar las distintas librerías para esta práctica:

```
from deepface import DeepFace
import pandas as pd
import matplotlib.pyplot as plt
```

2.1. Distintos modelos de reconocimiento facial

Tendremos distintos modelos de reconocimiento de facial, hemos intentado detectar la cara de una foto pero el resultado que hemos obtenido era una gráfica en blanco.

2.2. Comparación de rostros

Podremos comparar dos rostros para ver como de parecidos son, para ello usaremos el método *verify*. En la siguiente gráfica podemos apreciar el resultado de este método, vemos que nos aparece en que el parecido es **False** y un valor decimal.

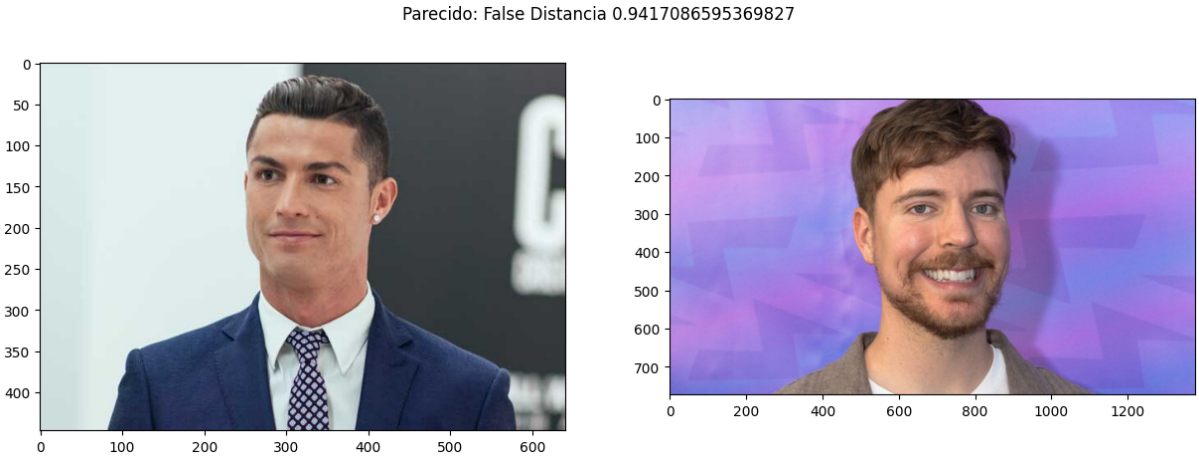


Figura 1: Resultado del método verify

El valor decimal (la distancia) mide la distancia entre dos vectores de caras. Un **valor más bajo** nos indicará una **mayor similaridad**. En este caso vemos que tenemos un valor bastante alto, por lo que ambas imágenes son bastante distintas.

Ahora vamos a usar distintos modelos capaces de comparar rostros, y vamos a determinar que modelo es el mejor, para ello tendremos que quedarnos con el modelo con el que mayor distancia nos indique, tras hacer cambios al código que se nos proporciona:

```
models = ["VGG-Face", "Facenet", "Facenet512", "OpenFace",
"DeepID", "ArcFace", "Dlib", "SFace"]

max_distance = -1
best_idx = -1
for i, model_name in enumerate(models):
    result = DeepFace.verify(img1_path=foto1, img2_path=foto2,
    model_name=model_name)
    fig, axs = plt.subplots(1, 2, figsize=(15, 5))
    fig.suptitle(f'Parecido: {result["verified"]}
    Distancia {result["distance"]:0}')
    axs[0].imshow(plt.imread(foto1))
    axs[1].imshow(plt.imread(foto2))

    if result["distance"] > max_distance:
        max_distance = result["distance"]
        best_idx = i
```

Tras esto hemos identificado que el mejor modelo para comparar rostros es el modelo OpenFace.

2.3. Identificación de rostros

Se podría dar el caso de que tenemos muchos rostros en un directorio, y nosotros tenemos una foto y queremos ver si dicha foto pertenece a alguna persona del directorio, para ello podemos emplear la función *find*. En este caso vamos a buscar a un famoso dentro de un directorio de imágenes, concretamente buscaremos a Brad Pitt dentro del directorio **fotos**, obtenemos estos resultados:

```
24-11-03 15:44:35 - Searching ./brad_pitt_busqueda.jpg in 5 length datastore
24-11-03 15:44:38 - find function duration 2.426966905593872 seconds
[             identity                                hash  target_x  \
0  fotos/brad_pitt.jpg  7b447307a03d6cbebd633b0534866846dfc99dfa        39

   target_y  target_w  target_h  source_x  source_y  source_w  source_h  \
0        93       242       242       185       332       713       713

   threshold  distance
0       0.68  0.465268  ]
```

Figura 2: Resultado en el caso de que la persona que buscamos se encuentra en el directorio de imágenes

En el caso de que la persona que buscamos no se encuentre en el directorio de imágenes no aparecerá nada.

Creo que está función es muy interesante para casos reales como: Acceso a domicilios particulares mediante reconocimiento facial.

2.4. Análisis de atributos faciales

Finalmente vamos a analizar las emociones de las personas dada una foto, para ello usaremos la función *analyze*, esta función nos devuelve una lista con los resultados de las emociones de la imagen, además de otros datos, en la siguiente imagen podemos observar lo que nos devuelve:

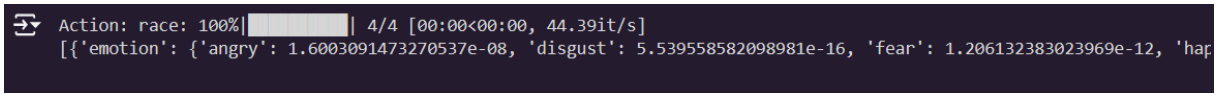
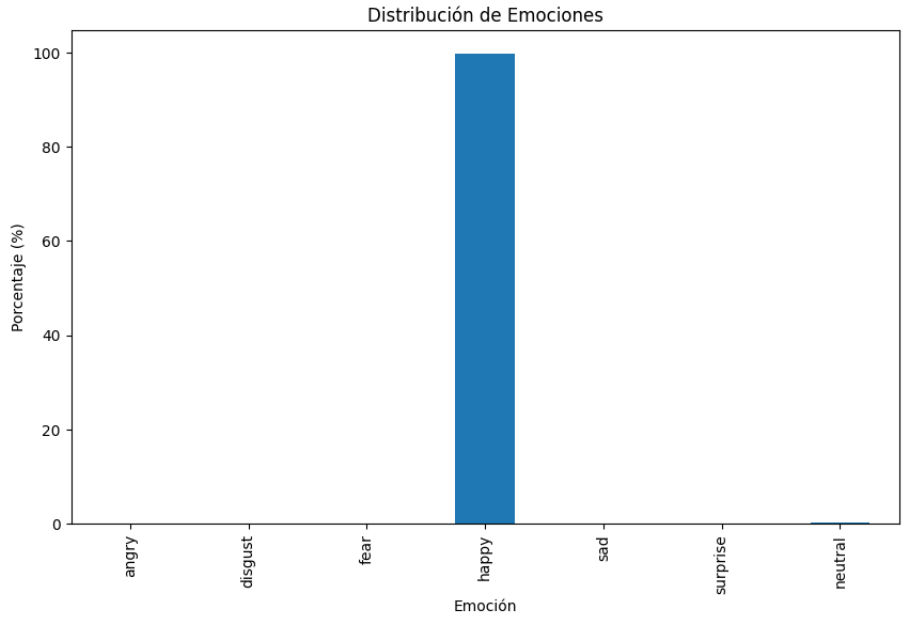


Figura 3: Return de la función analyze

Para esta parte hemos usado la siguiente imagen de Brad Pitt:

Figura 4: Imagen de Brad Pitt empleada para esta parte

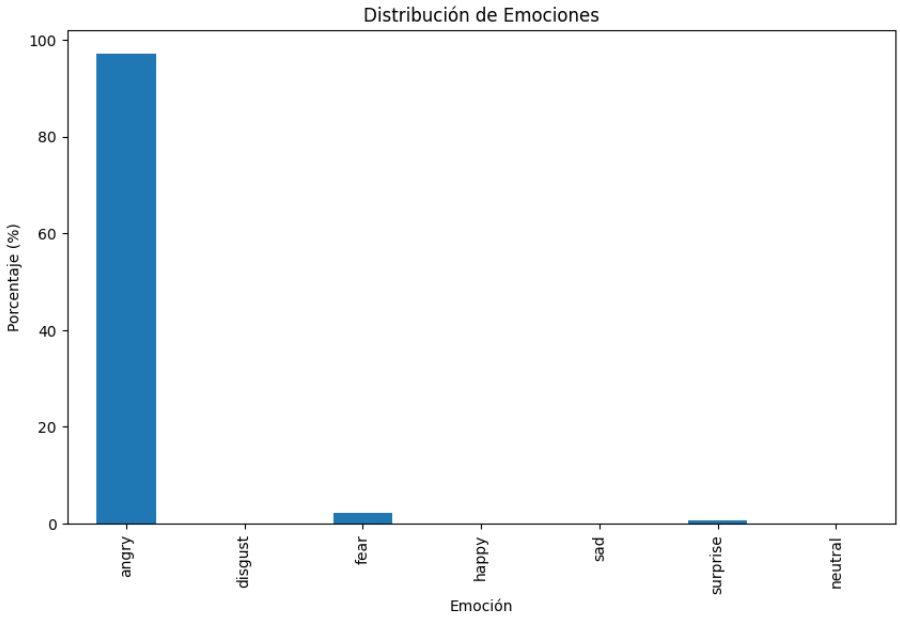
Como se puede apreciar en la imagen anterior se ve a Brad Pitt y se le ve contento, ahora si decidimos mostrar una gráfica de barras obtenemos lo siguiente:



Como se puede apreciar, el modelo ha funcionado correctamente a la hora de analizar las emociones del actor. Ahora vamos a probar con una persona que se encuentra enfadado, para ello vamos a usar una imagen del actor Jake Gyllenhaal en la película Nightcrawler:

Figura 5: Jake Gyllenhaal en la película Nightcrawler

Tras realizar el análisis de las emociones obtenemos la siguiente gráfica:



Vemos que el modelo funciona correctamente, ya que afirma con casi un 100 % de que la emoción predominante es la ira.

Esta función de la librería también la encuentro muy útil, ya que se podría emplear para muchas cosas útiles como podría ser detección de depresión en insitutos por ejemplo.

3. Conclusiones

En esta práctica hemos explorado cómo funciona el reconocimiento facial con la librería *DeepFace* y las posibilidades que ofrece para analizar similitudes, identificar rostros y hasta interpretar emociones. A través de diferentes modelos de comparación facial, logramos identificar cuál es el más preciso y, con el análisis de atributos faciales, pudimos ver cómo esta tecnología es capaz de reconocer emociones con bastante acierto.

Más allá de lo técnico, esta librería tiene aplicaciones muy útiles en el mundo real: desde acceso seguro mediante reconocimiento facial hasta herramientas que podrían ser usadas en salud mental para analizar el estado emocional. Aunque hemos encontrado algunas limitaciones, los resultados muestran que el reconocimiento facial está avanzando rápido y se vuelve cada vez más accesible para distintas aplicaciones.