Instituto Tecnológico de Culiacán

TECNOLÓGICO NACIONAL DE MÉXICO

Carrera: Ingeniería en Sistemas Computacionales

Materia: Inteligencia Artificial

Profesor: Zuriel Dathan Mora Félix

Trabajo: Adquisición y preprocesamiento de imágenes para la clasificación de emociones en personas

Grupo:

11:00 PM – 12:00 PM

Integrantes:

Ojeda López Luis Enrique

Saucedo Rodríguez Roberto Carlos

Paso 1: Adquisición de imágenes para la clasificación de emociones en personas

Como primer paso, nosotros buscamos un dataset que tuviera imágenes de las distintas emociones en la plataforma de Kaggle, encontramos muchos, pero con el inconveniente que se encontraban en blanco y negro. Finalmente encontramos el siguiente dataset *FANE: Facial Expression & Emotion Dataset*.

FANE: Facial Expressions & Emotion Dataset

FANE es un dataset de imágenes para clasificar expresiones faciales y emociones. Originalmente contaba con 9 categorías, con un total de 16,913 imágenes, sin embargo, por conveniencia, solo utilizamos las categorías siguientes:

- happy (feliz – 1922 imágenes)
- angry (enojado– 1774 imágenes)
- sad (triste – 2732 imágenes)
- surprise (sorpresa – 1854 imágenes)

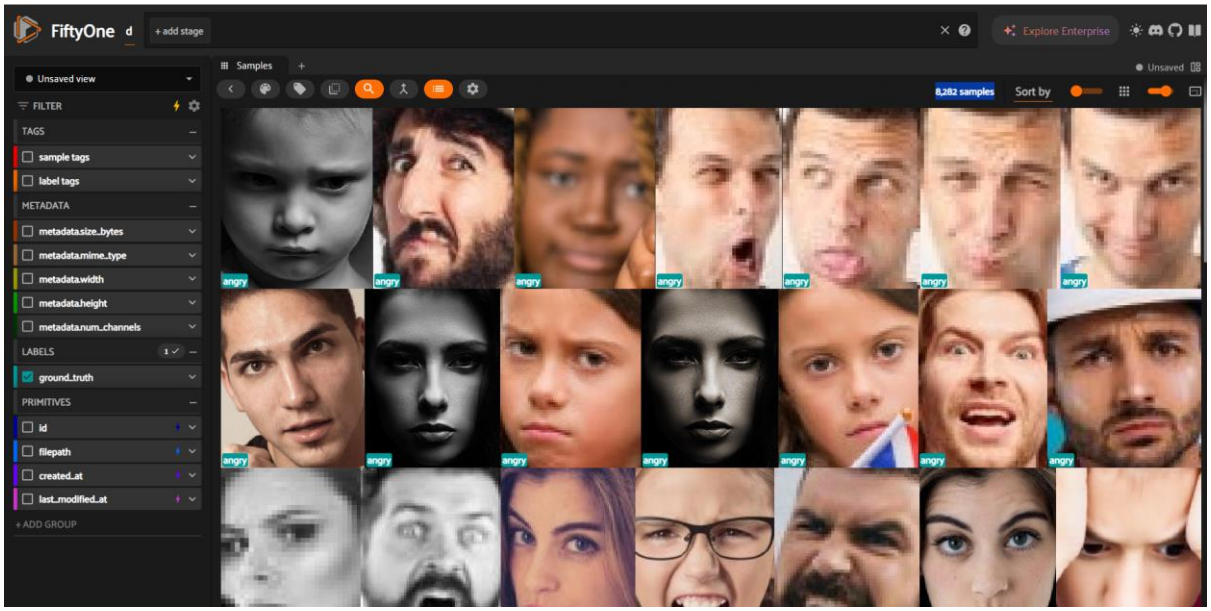
Total: 8282 imágenes resultantes

Paso 2: Conexión a FiftyOne

Como segundo paso, usamos la herramienta recomendada FiftyOne la cual nos ayuda a mostrar de manera visual nuestro conjunto de imágenes y filtrar por etiquetas. Instalamos primero **FiftyOne** en Python asegurándonos que coincidiese con una versión compatible al igual que actualizar otros entornos como mongoDB para evitar ciertos errores.

Primero hicimos una versión de script sencilla para conectarnos a fiftyone, en ella desactivamos primero las validaciones al cargar los nombres con *fo.config.database_validation = False* para evitar errores por estructuras de carpetas. Después realizamos la carga del dataset original con la ruta de este y las fuimos clasificando con .from_dir, le colocamos un nombre al dataset y también, si ya está procesado lo sobrescribimos con *overwrite = True*.

Y por último hicimos la conexión a FiftyOne para su visualización con sesión = fo.launch_app(preprocessed_dataset) y usamos *.wait()* para mantener la sesión abierta hasta que nosotros decidamos cerrarla.



Dataset cargado con éxito en FiftyOne

Paso 3: Preprocesamiento de imágenes

Una vez logramos la conexión de FiftyOne y poder observar las imágenes del dataset etiquetadas, procederemos al preprocesamiento de las imágenes.

Nosotros utilizamos **albumentations** para un preprocesamiento adecuado al igual que cv2 y os.

```
print("Dataset cargado correctamente con", len(dataset), "imágenes")

# Borrar dataset si existe para evitar error de nombre no disponible
if fo.dataset_exists("emociones_con_albumentations"):
    fo.delete_dataset("emociones_con_albumentations")

preprocessed_dataset = fo.Dataset(name="emociones_con_albumentations")

output_dir = r"C:\Users\luiso\OneDrive\Escritorio\Tec2025\Inteligencia_Artificial\U4\albumentations_output"
```

Indicamos con que nombre se debe de guardar el resultado y en donde, además de verificar que, si existe, se elimine la carpeta anterior, para volver a crearla.

```
transform = A.Compose([
    A.RandomBrightnessContrast(p=0.5),
    A.HorizontalFlip(p=0.5),
    A.Rotate(limit=45, p=0.7),
    A.Resize(128, 128)
])
```

Posteriormente, indicamos las transformaciones que realizaremos a cada imagen individualmente, por ejemplo, el cambio de brillo y el contraste con una probabilidad de 0.5, voltear horizontalmente la imagen con probabilidad de 0.5, rotar en un límite de hasta 45 grados con probabilidad de 0.7 y redimensión de la imagen a 128x128 pixeles y *A.Compose nos permite aplicar todas las transformaciones juntas*.

```
for sample in dataset:
    rel_path = os.path.relpath(sample.filepath, start=dataset_dir)
    new_path = os.path.join(output_dir, rel_path)
    os.makedirs(os.path.dirname(new_path), exist_ok=True)
```

Para cada imagen en el dataset, obtenemos su ruta relativa a la carpeta original, para mantener la estructura de las carpetas. Construimos la ruta completa donde se guardará la imagen procesada en la carpeta de salida y creamos la carpeta destino si no existe.

```
image = cv2.imread(sample.filepath)
if image is None:
    print(f"No se pudo cargar la imagen: {sample.filepath}")
    continue

image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
augmented = transform(image=image)
transformed_image = augmented["image"]
cv2.imwrite(new_path, cv2.cvtColor(transformed_image, cv2.COLOR_RGB2BGR))
```

Leemos la imagen con OpenCV, si no se puede leer, se salta para evitar errores.

Convertimos la imagen en BGR (formato que utiliza OpenCV) a RGB (formato que utiliza Albumentations). Se aplica las transformaciones definidas y convierte la imagen trasformada de nuevo a BGR para guardarla con OpenCV en la ruta destino.

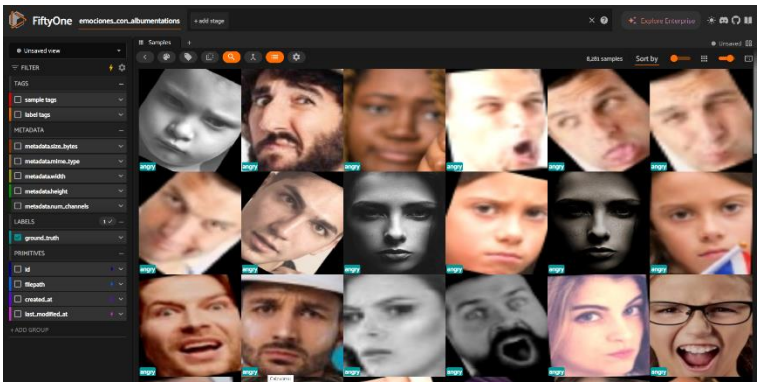
```
new_sample = fo.Sample(filepath=new_path)
if "ground_truth" in sample:
    new_sample["ground_truth"] = sample["ground_truth"]
preprocessed_dataset.add_sample(new_sample)
```

Se crea un nuevo **Sample** para el dataset preprocesado con la ruta de la nueva imagen. Si el **Sample** original tiene etiqueta **ground_truth** la copia al nuevo **Sample**. Y añade el resultado en el dataset.

```
print("Preprocesamiento con Albumentations completado.")

session = fo.launch_app(preprocessed_dataset)
session.wait()
```

Por último, se abre la App visual de **FiftyOne** para explorar el dataset preprocesado junto con *session.wait()* como se mencionó en el paso 2.



Resultado final.