# ESCUELA POLITÉCNICA NACIONAL

# FACULTAD DE INGENIERÍA DE SISTEMAS

# RECUPERACIÓN DE LA INFORMACIÓN

## Integrantes:


- KEVIN MALDONADO
- RAQUEL ZUMBA


**DOCENTE:** IVAN CARRERA

**FECHA DE ENTREGA:** 12-08-2024

# 1 Introducción

El objetivo de este proyecto es diseñar y desarrollar una máquina de búsqueda que facilite la recuperación de información visual mediante consultas basadas en imágenes. A diferencia de los motores de búsqueda tradicionales que dependen de texto para encontrar resultados, este sistema se enfoca en la similitud visual, lo que permite a los usuarios buscar imágenes comparables dentro de una base de datos predefinida. Este tipo de tecnología tiene el potencial de revolucionar la forma en que interactuamos con grandes volúmenes de datos visuales, proporcionando una herramienta poderosa para diversas aplicaciones prácticas.

El desarrollo del sistema se abordará en varias fases, cada una con un enfoque específico. Estas fases incluyen la adquisición de datos, el preprocesamiento de imágenes, la extracción de características relevantes, la indexación eficiente de las mismas, y finalmente, el diseño y evaluación del motor de búsqueda.

# 2 Análisis detallado de la evaluación del sistema

## 2.1 Conjunto de data para el entrenamiento y testeo

En la ilustración 1 se puede observar la cantidad de imágenes que se va a tomar para el conjunto de entrenamiento y testeo.

```
# Cargar y dividir el dataset usando índices absolutos
(train_dataset, test_dataset), dataset_info = tfds.load(
    name='caltech101',
    split=[f'train[:{train_size}]', f'train[{train_size}:]'],
    with_info=True,
    as_supervised=True,
    data_dir='C:\\DATARIIIB'
)

# Contar imágenes en las nuevas divisiones
num_train_images = tf.data.experimental.cardinality(train_dataset).numpy()
num_test_images = tf.data.experimental.cardinality(test_dataset).numpy()

print(f'Número de imágenes en el conjunto de entrenamiento: {num_train_images}')
print(f'Número de imágenes en el conjunto de prueba: {num_test_images}')
```

```
Número de imágenes en el conjunto de entrenamiento: 2447
Número de imágenes en el conjunto de prueba: 612
```

*Ilustración 1 Numero de imágenes para entrenamiento y testeo*

## 2.2 Numero de clases de la data Caltech

El número de clases del dataset Caltech es de 102 como se muestra en la ilustración 2.

Obtener el numero de clases

```
[11] # Obtener el número de clases del conjunto de datos
     num_classes = dataset_info.features['label'].num_classes
     # Imprimir el número de clases
     print(f'Número de clases en el dataset: {num_classes}')
```

```
Número de clases en el dataset: 102
```

*Ilustración 2 Numero de clases*

## 2.3 Numero de imágenes por cada categoría

En la ilustración 3 se muestra el total de imágenes por cada clase, de esta manera podemos visualizar si la data esta equilibrada.

```
[12] # Inicializar un diccionario para contar el número de imágenes por clase
     class_counts = {i: 0 for i in range(num_classes)}

     # Contar las imágenes por clase en el conjunto de entrenamiento
     for image, label in tfds.as_numpy(train_dataset):
         class_counts[label] += 1

     # Contar las imágenes por clase en el conjunto de prueba
     for image, label in tfds.as_numpy(test_dataset):
         class_counts[label] += 1

     # Obtener los nombres de las clases
     class_names = dataset_info.features['label'].int2str

     # Imprimir el número de imágenes por clase
     for class_id, count in class_counts.items():
         class_name = class_names(class_id)
         print(f'Clase: {class_name}, Número de imágenes: {count}')
```

```
Clase: airplanes, Número de imágenes: 651
Clase: anchor, Número de imágenes: 33
Clase: ant, Número de imágenes: 36
Clase: background_google, Número de imágenes: 368
Clase: barrel, Número de imágenes: 33
Clase: bass, Número de imágenes: 49
Clase: beaver, Número de imágenes: 40
Clase: binocular, Número de imágenes: 28
```

*Ilustración 3 Numero de imágenes por clase*

## 2.4 Búsquedas similares de las imágenes

En la ilustración 4 se realiza una búsqueda de las imágenes más similares a una imagen de consulta en un conjunto de datos utilizando el modelo de vecinos más cercanos (Nearest Neighbors). Primero, selecciona una imagen específica para la consulta y extrae sus características. Luego, estas características se utilizan para buscar en el conjunto de características preprocesadas utilizando un modelo de vecinos más cercanos entrenado con el algoritmo `ball_tree`. Finalmente, se identifican y muestran las imágenes más similares a la imagen de consulta junto con sus distancias, lo que permite evaluar la similitud entre las imágenes.

```
# Seleccionar la imagen con índice 10
query_index = 10

# Obtener las características y la imagen de consulta
query_features = test_features_flat[query_index]
query_img = test_imgs_flat[query_index]

# Redimensionar las características de la imagen de consulta para la búsqueda
query_features = query_features.reshape(1, -1)

# Encontrar los vecinos más cercanos en el conjunto de prueba
from sklearn.neighbors import NearestNeighbors
nn_model = NearestNeighbors(n_neighbors=10, algorithm='ball_tree').fit(test_features_flat)
##nn_model = NearestNeighbors(n_neighbors=5, algorithm='brute').fit(train_features_flat)

distances, indices = nn_model.kneighbors(query_features)
```

*Ilustración 4 Obtención de características*

De acuerdo con lo que se explicó en la sección anterior se puede ver que en la búsqueda se muestra las imágenes similares.

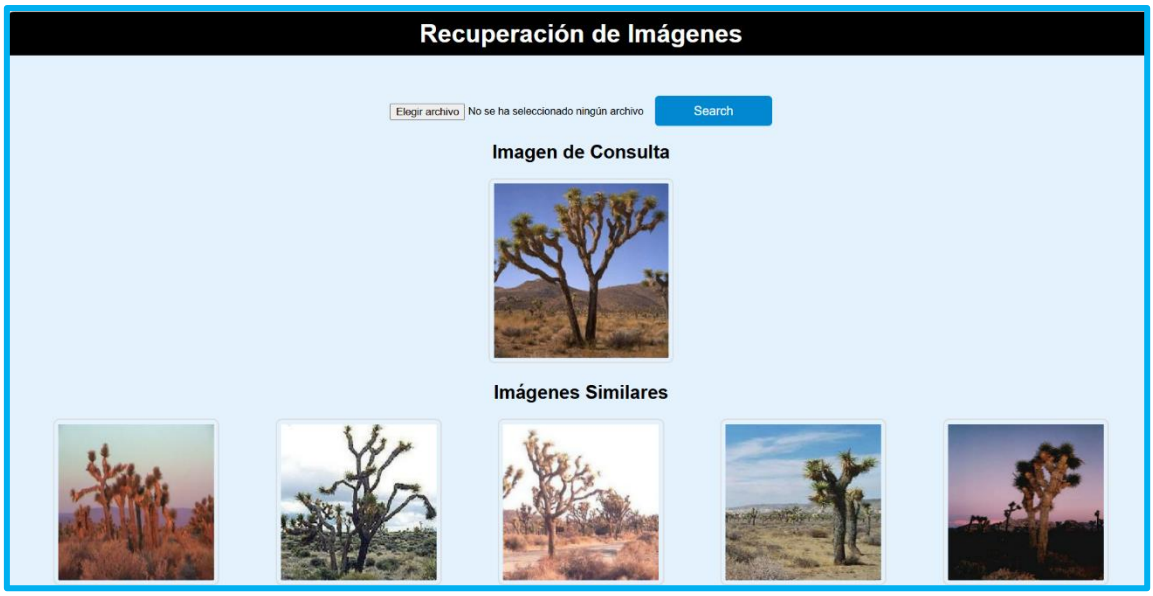
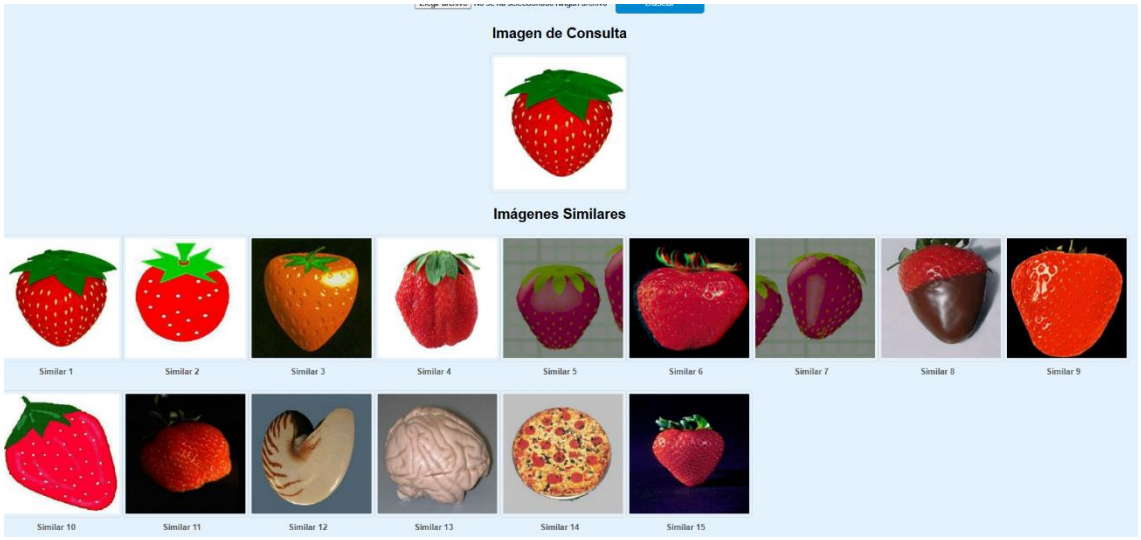



*Ilustración 5 Imágenes Similares*


*Ilustración 6 Imágenes Similares*

## 2.5 Métricas de evaluación del sistema

En esta búsqueda se puede visualizar que no se tiene todas las imágenes similares debido a que la cantidad de imágenes para esa clase era muy poca, púes de esta manera se tiene una precisión de 0.82, un recall de 0.71 y un F-score de 0.68 como se muestra en la ilustración 7.



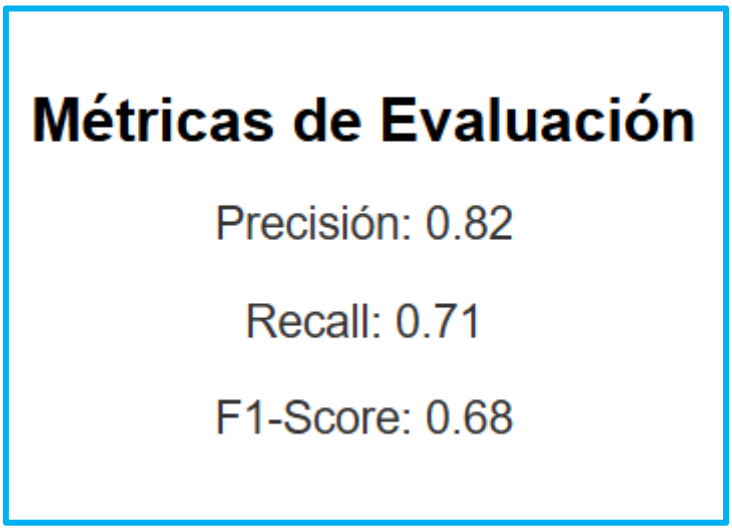




*Ilustración 7 Métricas de evaluación*

### 2.5.1 Precisión:

Un valor de precisión de 0.82 sugiere que, de todas las imágenes que el sistema ha identificado como similares a la imagen de consulta, el 82% son realmente relevantes. Esto indica que el sistema tiene una buena capacidad para evitar falsos positivos, es decir, imágenes que no son realmente similares pero que han sido clasificadas como tales por el sistema. En otras palabras, el sistema es bastante eficaz ya que estamos asegurando que las imágenes recuperadas cumplan con las expectativas de similitud, lo que es crucial en aplicaciones donde la relevancia de las imágenes presentadas es prioritaria.

### 2.5.2 Recall:

Un recall de 0.71 indica que el sistema ha logrado recuperar el 71% de todas las imágenes que son realmente similares a la imagen de consulta. Esto significa que, aunque el sistema es efectivo en identificar una buena parte de las imágenes relevantes, aún deja fuera un 29% de imágenes que deberían haber sido recuperadas (falsos negativos).

### 2.5.3 F1-Score:

El F1-Score de 0.68 refleja un desequilibrio por decirlo así entre las otras métricas. Un F1-Score de este nivel indica que, aunque el sistema tiene una precisión bastante alta y un recall razonable, y que la la combinación de ambos aún puede optimizarse. Un F1-Score más bajo que las métricas individuales de precisión y recall sugiere que ni la precisión ni el recall están en su punto óptimo, y que hay un desequilibrio en cómo el sistema maneja la relevancia y la exhaustividad.