

# TUIA - Computer Vision

## Trabajo Práctico Final

### Sistema de Detección y Clasificación de Razas de Perros

#### **Objetivo General**

Desarrollar un pipeline completo de visión por computadora para la identificación de razas de perros en imágenes. El proyecto abarca desde la creación de un sistema de búsqueda por similitud hasta la implementación de un sistema de detección y clasificación en imágenes complejas, incluyendo el entrenamiento y la optimización de modelos de Deep Learning.

#### **Recursos y Herramientas**

- **Dataset:** [70 Dog Breeds Image Dataset \(Kaggle\)](#)

### Consignas del Trabajo Práctico

El proyecto se divide en cuatro etapas incrementales. Cada etapa construye sobre la anterior.

#### ***Etapa 1: Buscador de Imágenes por Similitud***

El objetivo de esta etapa es crear una aplicación que, dada una imagen de un perro, encuentre las imágenes más similares en el dataset.

##### **1. Creación de la Base de Datos Vectorial:**

- Utilice un modelo pre-entrenado en ImageNet (ej. **ResNet50**) para extraer un vector de características (embeddings) de cada imagen del dataset de perros.
- Indexe estos vectores en una base de datos vectorial para permitir búsquedas eficientes.

##### **2. Desarrollo de la Aplicación en Gradio:**

- Construya una interfaz simple con Gradio que permita al usuario subir una imagen de un perro.
- La aplicación debe procesar la imagen de entrada, extraer su vector de características y usar la base de datos vectorial para encontrar las **10 imágenes más similares** del dataset.
- Muestre la imagen de entrada y las 10 imágenes recuperadas en la interfaz.

##### **3. Clasificación Basada en Similitud y Métrica de Evaluación:**

- A partir de los 10 resultados, implemente una lógica para predecir la raza del perro de la imagen de entrada (ej. por "voto mayoritario" de las razas de los resultados). Muestre la raza más probable en la interfaz.

- Prepare un conjunto de prueba (5-10 imágenes por raza, separadas del conjunto de búsqueda). Para cada imagen de prueba, ejecute la búsqueda y calcule la métrica **NDCG@10 (Normalized Discounted Cumulative Gain)** para evaluar la calidad del sistema de búsqueda.

## **Etapla 2: Entrenamiento y Comparación de Modelos de Clasificación**

En esta etapa, entrenará sus propios modelos para mejorar la representación de las características y la clasificación.

### **1. Entrenamiento de Modelos:**

- **Modelo A (Transfer Learning):** Realice un fine-tuning de un **ResNet18** sobre el dataset de 70 razas de perros.
- **Modelo B (Opcional, recomendado):** Diseñe y entrene un modelo de clasificación convolucional (CNN) propio ("custom") desde cero.

En cualquier caso, explicar las métricas obtenidas (Sensibilidad, Especificidad, Precisión, Exactitud y puntaje F1).

### **2. Integración y Selección en la Aplicación:**

- Actualice la aplicación de Gradio para que los vectores de características se puedan generar usando tanto el **ResNet18** entrenado como su modelo custom.
- Añada un componente de selección (un menú desplegable o combo) en la interfaz que permita al usuario **elegir qué modelo utilizar** para realizar la búsqueda por similitud.

## ***Etapla 3: Pipeline de Detección y Clasificación en Escenas Complejas***

El objetivo es hacer que el sistema funcione en imágenes del mundo real, donde puede haber múltiples objetos además de perros.

### **1. Detección de Objetos:**

- Utilice un modelo **YOLO pre-entrenado de Ultralytics** (ej. yolov8n.pt) para detectar la presencia de perros en una imagen de entrada. **No es necesario entrenar el detector.**
- El sistema debe poder procesar imágenes que contengan otros objetos o múltiples perros.

### **2. Creación del Pipeline Completo:**

- Genere una nueva versión de la aplicación. El flujo de trabajo será:
  1. El usuario sube una imagen compleja.
  2. El modelo YOLO detecta todos los perros y extrae sus "bounding boxes" (las cajas que los rodean).
  3. Para cada perro detectado, el sistema debe recortar la imagen.

4. Cada recorte se pasa al **mejor modelo clasificador de la Etapa 2**.
5. La aplicación debe mostrar la imagen original con los bounding boxes dibujados y una etiqueta con la raza predicha para cada perro.

#### ***Etapla 4: Evaluación, Optimización y Herramientas de Anotación***

Esta etapa final se enfoca en la evaluación del pipeline completo, su optimización y la creación de herramientas auxiliares.

##### ***1. Evaluación del Pipeline:***

- Capture o recopile un **conjunto de prueba de 10 imágenes complejas** (con perros y otros objetos).
- **Anote manualmente** estas 10 imágenes (creando los bounding boxes para los perros y asignando su raza correcta).
- Ejecute su pipeline de la Etapa 3 sobre este conjunto de prueba y mida su desempeño (mAP, IoU, Precision, Recall y F1-Score), explique a detalle el resultado obtenido.

##### ***2. Optimización de Modelos (Elegir una):***

- Aplique una técnica de optimización a su mejor modelo de clasificación y de detección para acelerar la inferencia. Las opciones son:
  - **Cuantización:** Reduzca la precisión de los pesos del modelo (ej. de FP32 a INT8) usando las herramientas de PyTorch.
  - **Exportación a formato optimizado:** Convierta el modelo a ONNX o intente compilarlo con TensorRT.
- Mida y compare la velocidad de inferencia y la precisión del modelo original vs. el modelo optimizado.

##### ***3. Script de Anotación Automática:***

- Cree un script de Python que utilice su pipeline completo (detector + clasificador) para **generar anotaciones de forma automática**.
- El script debe tomar una carpeta de imágenes, procesarlas y exportar las anotaciones de los perros detectados y clasificados en dos formatos estándar: **YOLOv5 (.txt)** y **COCO (.json)**.

### **Entregables**

1. **Código Fuente:** Repositorio de Git (GitHub, GitLab) con todo el código del proyecto, claramente organizado en carpetas. Los repositorios deben ser privados, y sólo se comparten a los profesores de la materia.
2. **Aplicación Funcional:** Instrucciones claras en un archivo README.md para instalar las dependencias y ejecutar la aplicación de Gradio en todas sus versiones.
3. **Informe (PDF):** Un documento de al menos 15 páginas en donde contenga:
  - Las decisiones de diseño tomadas.

- Los resultados de las métricas de evaluación explicadas.
- Comentar los desafíos o problemas encontrados durante la elaboración del trabajo práctico, y cómo fueron abordados.
- Una conclusión general sobre la utilidad de los productos generados en el TP, y las posibilidad de ser aplicados en la vida real, en función de la calidad y resultados obtenidos.

4. ***Fechas estimadas de entrega:***

- ***1ra y 2da etapa: Jueves 27 de Noviembre***
- ***3ra y 4ra etapa: Jueves 11 de Diciembre***