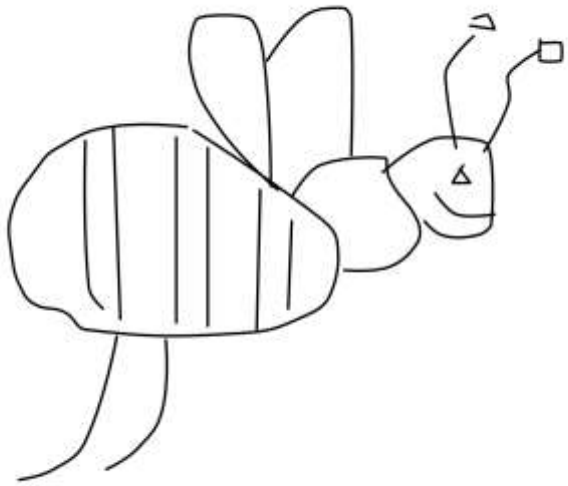
2024

Memoria Práctica 2

RECONOCIMIENTO DE PATRONES

DIEGO GONZÁLEZ OVIAÑO Y PAULA PENA GONZÁLEZ



ÍNDICE

OBJETIVO

El objetivo de esta práctica es el desarrollo de un sistema de aprendizaje profundo para la clasificación de bocetos de objetos cotidianos recogidos en el conjunto de datos un artículo presentado en SIGGRAPH, una de las conferencias más importantes en informática gráfica.

MATERIALES Y MÉTODOS

La presente práctica se llevó a cabo utilizando el conjunto de datos de bocetos recogido y descrito en el artículo presentado en SIGGRAPH. Dicho conjunto contiene 20,000 dibujos únicos, distribuidos en 250 categorías de objetos cotidianos, con un enfoque en las primeras 26 categorías para esta práctica. Los datos están disponibles en formato SVG y PNG en el siguiente enlace: Conjunto de datos de bocetos.

Principales recursos y herramientas:

Lenguaje de programación: Python 3.9
Librerías y herramientas:

- **`ultralytics`** para el manejo y entrenamiento del modelo YOLOv11.
- **`scikit-learn`** para la separación estratificada del conjunto de datos.
- **`opencv-python`** para la visualización de las imágenes durante el proceso de inferencia.
- **`numpy`** y **`pandas`** para el procesamiento de datos y creación de la matriz de confusión.
- **`tqdm`** para el seguimiento del progreso de tareas largas.

Hardware: Una GPU NVIDIA compatible para la aceleración del entrenamiento.

Etapas de desarrollo de la práctica:

1. Estado del Arte:

Se abordó el proyecto como un problema que necesitaba de un conocimiento previo del estado del arte y de las técnicas más efectivas para garantizar el éxito en su resolución. Se estudió la arquitectura de YOLO que proporcionaba buenos resultados en tareas de clasificación, a pesar de que su tarea primaria fuese la detección.

2. Preparación del conjunto de datos:

Una vez resuelto la arquitectura que se iba a utilizar, se organizó el conjunto de datos original en carpetas representando cada categoría. Se desarrolló un script en Python (**`splitter.py`**) para realizar una división estratificada en tres subconjuntos:

- Entrenamiento: 80% de las imágenes, usado para entrenar el modelo.
- Validación: 10% de las imágenes, empleado para evaluar el desempeño del modelo durante el entrenamiento.
- Prueba: 10% restante, utilizado para la evaluación final del modelo.

Los archivos generados (**`train.txt`**, **`validation.txt`**, y **`test.txt`**) contienen las rutas relativas de las imágenes asociadas a cada subconjunto.

3. Entrenamiento del modelo:

Utilizando el script **`train.py`**, se estructuraron los conjuntos de entrenamiento y validación en carpetas organizadas por clase, ya que se trata de la estructura necesaria para el modelo utilizado: YOLOv11 pre-entrenado, ajustado con el conjunto de datos de bocetos mediante *transfer learning*.

Además, al profundizar en la estructura de YOLOv11, se observó que realiza unas transformaciones a los datos de entrada tales como rotaciones, oclusiones, mosaicos, o cambios de iluminación. Debido al contexto de la tarea, no se está interesado en entrenar acorde a oclusiones o mosaicos, ya que es un problema que la red no iba a tener y podría empeorar el resultado.

Se desactivaron los parámetros comentados a continuación: Mosaic, MixUp, Random Erasing y se limitó el número de clasificaciones a una por imagen. Asimismo, se buscó un equilibro entre velocidad y precisión y se usó como archivo base de pesos el modelo small de YOLOv11.

Los hiperparámetros clave del entrenamiento fueron:

- Número de épocas: 200
- Early Stopping: 50
- Tamaño del lote: 8
- Resolución de imágenes: 640 × 640 píxeles
- Dispositivo: GPU para acelerar el proceso.

4. Evaluación del modelo:

Mediante el script **`finalproduct.py`**, se realizaron las siguientes tareas:

a) Inferencia sobre las imágenes de prueba listadas en test.txt.

b) Generación de predicciones para cada imagen, almacenadas en **`etiquetas_estimadas.txt`.**

c) Cálculo de la matriz de confusión basada en las etiquetas reales y las predicciones, guardada en **`matriz_confusion.txt`.**

Adicionalmente, se implementó una funcionalidad para visualizar las imágenes con sus predicciones superpuestas. Se adjunta en la **Imagen 1** los resultados de dicha función.

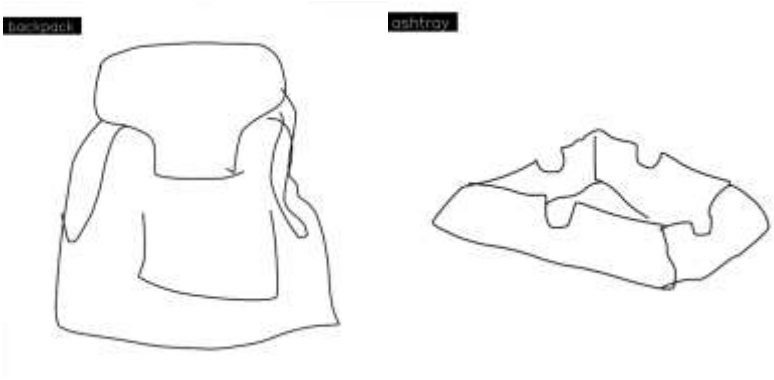


Imagen 1: Resultados de la función de visualización.

RESULTADOS

Se obtiene el archivo de pesos correspondiente al entrenamiento realizado. Dicho archivo será lo que se utilice para la clasificación en el producto final. La matriz de confusión (**Imagen 2**) permitió evaluar la precisión y el desempeño general del modelo, identificando posibles errores y clases más desafiantes.

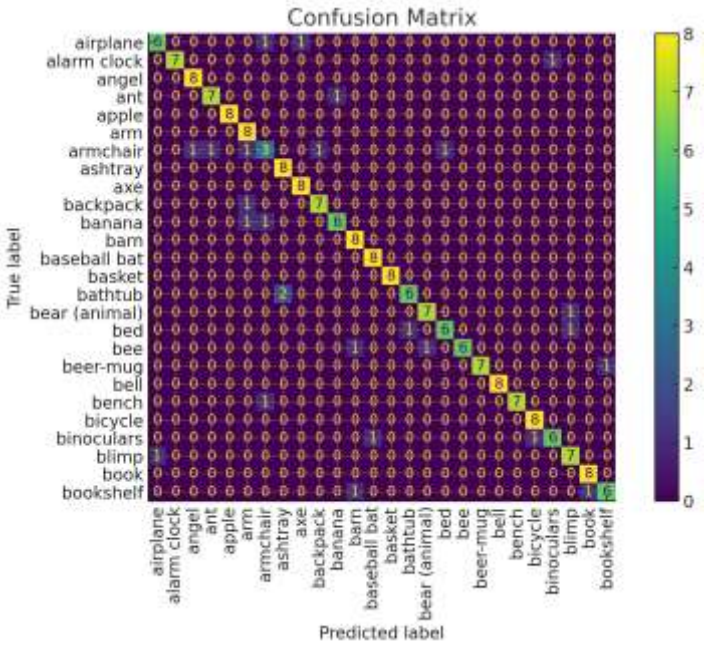


Imagen 2. Matriz de Confusión del modelo especializado para nuestra tarea de clasificación.

El modelo clasificó correctamente todos los ejemplos de las clases *basket*, *apple y bell*, alcanzando un *Precision*, *Recall* y *F1-Score* perfectos, lo que sugiere que estas categorías poseen características muy distintivas que el modelo ha podido aprender.

Para el resto de las clases, también se mostró un buen desempeño, con métricas elevadas, aunque cometió pequeños errores de clasificación. Finalmente, la clase *armchair* destacó por su peor rendimiento, por lo que podemos inferir que el modelo tiene más dificultades para clasificarla comparado con el resto. Puede que se deba a la falta de características visuales distintivas o a su alto parecido con otras categorías.

En conclusión, en general, el sistema de clasificación basado YOLOv11 ha demostrado ser efectivo para categorizar bocetos de objetos cotidianos, alcanzando valores altos de *Precision*, *Recall* y *F1-Score* en la mayoría de las clases (ver **Tabla 1**). Sin embargo, su rendimiento varió significativamente entre categorías, con un desempeño óptimo en clases con características visuales distintivas y dificultades en aquellas más abstractas o con similitudes visuales entre clases. Esto resalta la importancia de un conjunto de datos equilibrado y representativo.

Tabla 1. Métricas del modelo para cada una de las clases. Los valores de Precision, Recall y F1-Score se infirieron a partir de los datos de la matriz de confusión.

	Precision	*Recall*	*F1-Score*
basket	1.000000	1.000	1.000000
apple	1.000000	1.000	1.000000
bell	1.000000	1.000	1.000000
book	0.888889	1.000	0.941176
axe	0.888889	1.000	0.941176
bicycle	0.888889	1.000	0.941176
angel	0.888889	1.000	0.941176
baseball bat	0.888889	1.000	0.941176
beer-mug	1.000000	0.875	0.933333
bench	1.000000	0.875	0.933333
alarm clock	1.000000	0.875	0.933333
barn	0.800000	1.000	0.888889
ashtray	0.800000	1.000	0.888889
bear (animal)	0.875000	0.875	0.875000
ant	0.875000	0.875	0.875000
backpack	0.875000	0.875	0.875000
bee	1.000000	0.750	0.857143
arm	0.727273	1.000	0.842105
blimp	0.777778	0.875	0.823529
binoculars	0.857143	0.750	0.800000
airplane	0.857143	0.750	0.800000
bed	0.857143	0.750	0.800000
bathtub	0.857143	0.750	0.800000
banana	0.857143	0.750	0.800000
bookshelf	0.857143	0.750	0.800000
armchair	0.500000	0.375	0.428571