INFORME DE TRABAJO
RENDIMIENTO DE ETIQUETAS

David Urrego Serna
Miguel Angel Fonseca Aldana

Sábado 19 de Julio

Santiago Hernández Torres
Medellín, Colombia
2025-1

Introducción:
Primero, cabe recordar la pregunta que plantea el problema, ¿qué modelo es mejor en la detección precisa de vehículos en escenarios complejos, y cómo varía esta efectividad según el tamaño del vehículo?.
Con esto en mente, los problemas en la detección de vehículos recaen no sólo en la cantidad de objetos en la imagen, sino también en la contaminación del ambiente (SMOG), la calidad de la foto o hasta la lejanía de los objetos.
Por lo anterior, la detección y clasificación de vehículos es complejo, en ocasiones hasta para el ojo humano, por lo mismo, se entrenaron tres modelos, YOLOv5, Faster R-CNN y RetinaNet, de los cuales se obtuvieron diferentes resultados.

Metodología:
El dataset utilizado para el entrenamiento de los modelos consta miles de imágenes aéreas de diferentes lugares, ángulos y provenientes de diferentes dispositivos (para más info, https://www.kaggle.com/datasets/dronevision/vsaiv1).
Dicho dataset, contiene, no solo las imágenes ya divididas en train, test y val, sino también información adicional sobre cada imagen, información que fue usada para el entrenamiento de los modelos.
Cada modelo fue entrenado obteniendo valores de lecturas tanto para vehículos pequeños(ap 0.50) como vehículos grandes(ar 0.50:0.95); además es importante aclarar que, para el reconocimiento de estas imágenes, se usaron librerías nativas de python, particularmente la referencia de detracción de pytorch, o ambientes de ML además de un aporte importante de documentación de github(https://github.com/pytorch/vision/tree/main/references/detection).
el proyecto se estructuro de a siguiente manera:

```
Proyecto-Final-ML/
│
├── data/
│   └── VSAlv1/
│       └── split_ss_444_lsv/
│           ├── train/
│           │   ├── (carpeta imágenes y carpetas de anotaciones)...
│           ├── test/
│           │   ├── (carpeta imágenes y carpetas de anotaciones)...
│           └── val/
│               ├── (carpeta imágenes y carpetas de anotaciones)...
│
├── model_train.py
└── model_evaluation.py
│
├── requirements.txt    # Lista de dependencias
└── README.md           # Este archivo
```

esta estructura modular facilita la gestión del código y la replicabilidad de los experimentos, los scripts model_train.py y model_evaluation.py encapsulan la lógica de entrenamiento y evaluación.

Entrenamiento y resultados de los modelos:

El entrenamiento de los modelos, aunque tardado, fue bastante fructífero, en cuanto a los puntajes de Average Precision(AP) y un Average Recall(AR) es necesario recalcar que, para ambos,la siguiente interpretación de puntajes es válida:

- <2 es un puntaje malo
- [2-4) es aceptable
- [4-5] es bueno
- >5 es muy bueno

Con esto en cuenta, continuamos con la evaluación de cada modelo

- **RetinaNet:** fue el modelo con peores resultados, por lo mismo no se prosiguió con el análisis de las imágenes

```
IoU metric: bbox
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.068
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.165
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.044
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.012
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.112
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.172
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.027
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.108
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.171
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.038
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.218
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.418
```

(metrics_retinanet_20250719_195333.txt)

- **Faster R-CNN:** fue el segundo mejor rendimiento, demarcaba claramente los rectángulos con los vehículos, con más dificultan los pequeños que los grandes y teniendo algunas confusiones elementos de la imagen

```
IoU metric: bbox
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.347
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.558
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.372
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.159
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.396
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.598
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.107
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.280
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.427
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.221
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.481
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.689
```

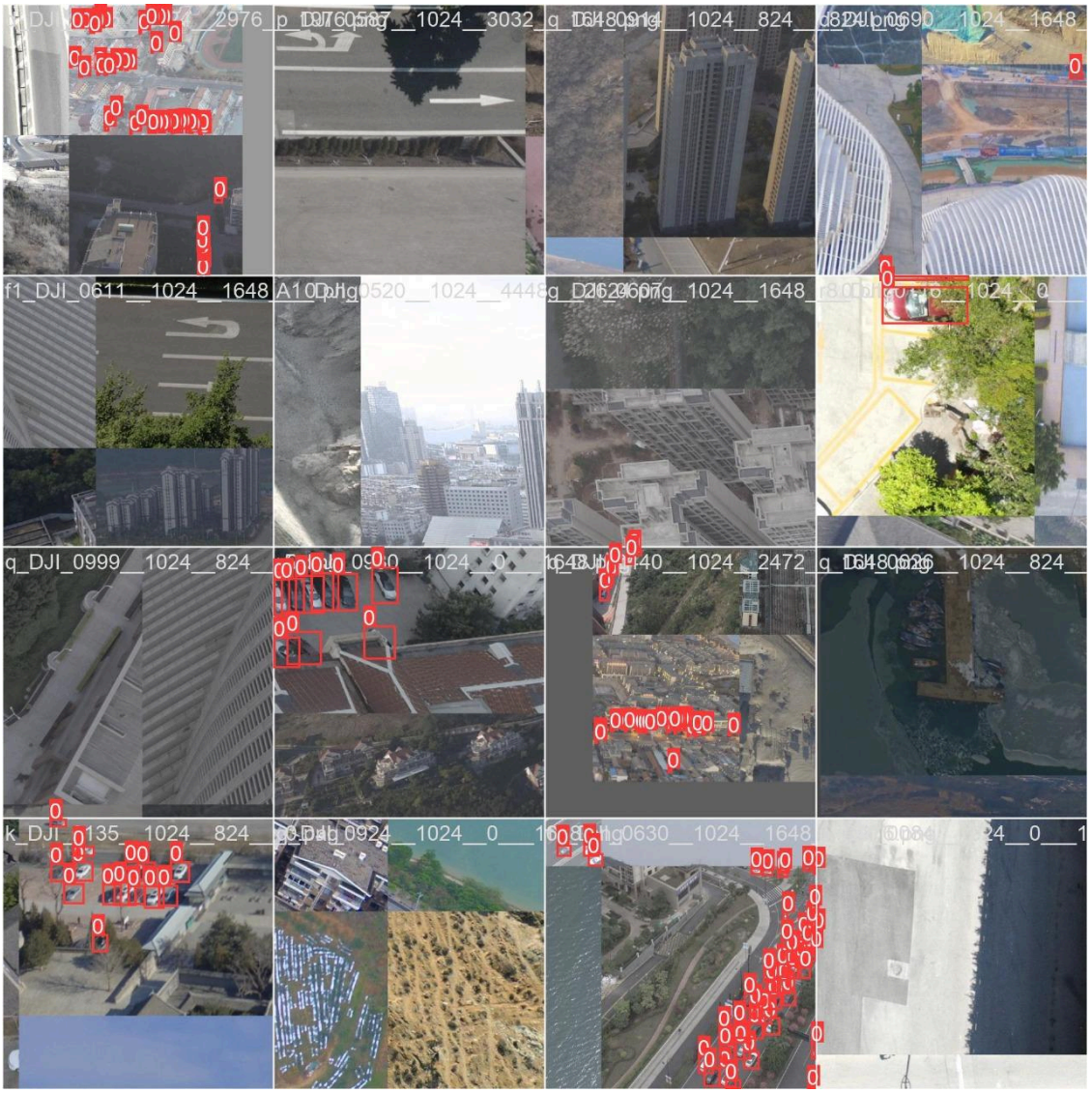
(metrics_faster_r-cnn_20250719_195032.txt)

en la imagen anterior se puede ver que en general tiene un buen rendimientoa la hora del reconocimiento del vehículo, como se puede ver a continuación:

se pueden notar algunos errores, como el reconocimiento de objetos aleatorios como vehículos o confundiendo vehículos grandes con pequeños, pero de manera general,el reconocimiento de vehículos es muy bueno

- **YOLOv5:** fue el modelo con mejores resultados tanto en AP como en AR, respectivamente y con diferencia

```
0.60779,          0.69175,
0.58912,           0.6708,
0.61167,          0.69444,
0.60416,          0.69094,
0.60686,          0.68234,
0.61193,          0.68919,
0.61021,          0.68278,
0.63845,          0.69463,
0.61374,          0.68628,
0.62315,          0.68413,
 0.6149,          0.68499,
  0.622,          0.69142,
0.62588,          0.69625,
```



toda la info mencionada anteriormente se encuentra en la carpeta yolov5 (se crea despues de ejecución)

Análisis de componentes Principales (PCA):

Para comprender mejor la variabilidad en los datos y la relación entre las características de los vehículos y los niveles de oclusión, se realizó un Análisis de Componentes Principales (PCA). Los resultados se presentan en dos gráficos principales:

Gráfico 1: PCA por Clase de Vehículo

- Ejes: PC1 y PC2 son las dos primeras componentes principales. PC1 explica el 33.91% de la varianza, mientras que PC2 explica el 33.20% de la varianza. En conjunto, estas dos componentes capturan una parte significativa de la variabilidad total de los datos.

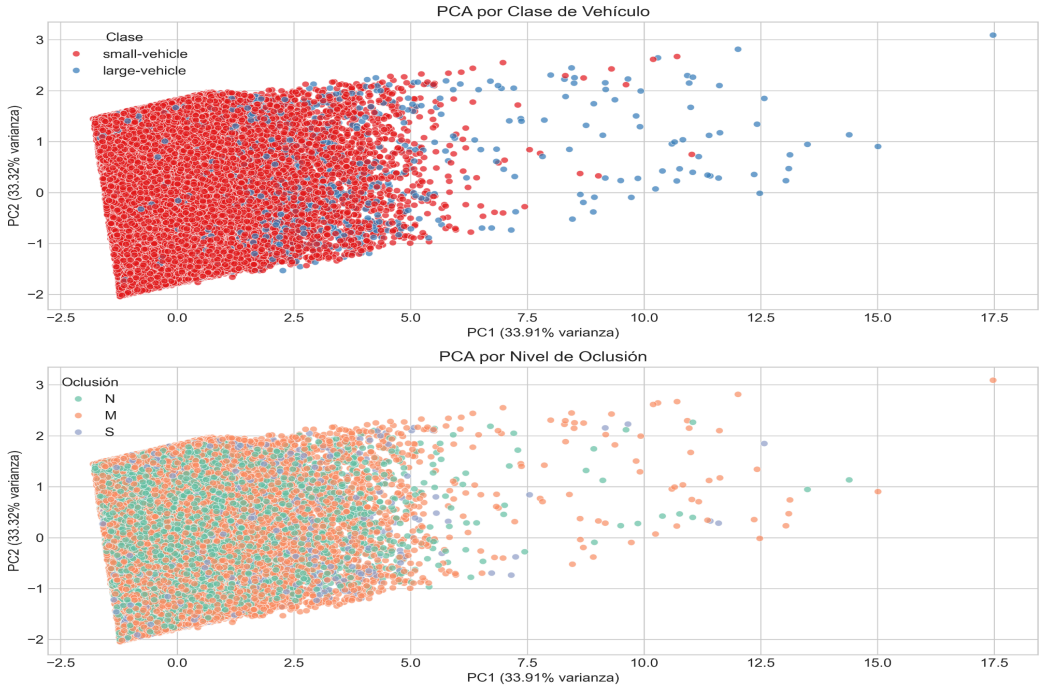
- Colores: Los puntos rojos representan small-vehicle (vehículos pequeños) y los puntos azules representan large-vehicle (vehículos grandes).

- Interpretación: Se observa que la mayoría de los puntos rojos (vehículos pequeños) se agrupan hacia a la izquierda del gráfico. Por otro lado, los puntos azules (vehículos grandes) tienden a dispersarse más hacia la derecha, especialmente a partir de un valor de PC1 mayor a aproximadamente 5. Esto sugiere que la Componente Principal 1 (PC1) logra separar parcialmente las clases de vehículos (pequeños vs. grandes), aunque no de manera perfecta, indicando que existen otras dimensiones de variabilidad que influyen en la distinción.

Gráfico 2: PCA por Nivel de Oclusión

- Colores: Verde (N) indica sin oclusión, Naranja (M) indica oclusión media, y Azul (S) indica oclusión severa.

- Interpretación: A diferencia del análisis por clase de vehículo, no se observa una separación clara entre los diferentes niveles de oclusión en las componentes principales. Los tres niveles (sin oclusión, oclusión media y oclusión severa) se encuentran bastante mezclados en el espacio definido por PC1 y PC2. Esto implica que la oclusión no tiene una influencia significativa en la variabilidad capturada por las dos primeras componentes principales, sugiriendo que PC1 y PC2 no son las dimensiones más adecuadas para distinguir entre los niveles de oclusión.



Conclusiones y Recomendaciones:

Basado en el análisis exhaustivo de los modelos de detección y los resultados del Análisis de Componentes Principales, se extraen las siguientes conclusiones y recomendaciones:

1. **Separación de Clases de Vehículos:** La Componente Principal 1 (PC1) es efectiva para diferenciar vehículos grandes de pequeños, aunque esta separación no es perfecta. Esto sugiere que, si bien el tamaño del vehículo es un factor importante en la variabilidad de los datos, no es el único determinante. Para tareas de clasificación que dependen fuertemente del tamaño, PC1 puede ser un preprocesamiento útil, pero se deben considerar otras características para una distinción más precisa.

2. **Influencia de la Oclusión:** La oclusión no parece tener una influencia significativa en las dos primeras componentes principales (PC1 y PC2). Esto indica que estas componentes no son adecuadas para capturar la variabilidad relacionada con los niveles de oclusión. Para analizar el impacto de la oclusión, sería necesario explorar otras técnicas de reducción de dimensionalidad o características específicas que sean más sensibles a este factor.

3. **Utilidad del PCA:** El PCA puede ser una herramienta valiosa para preprocesar datos cuando el objetivo es distinguir entre diferentes tipos de vehículos, especialmente en función de su tamaño. Sin embargo, no es recomendable utilizar PCA para predecir o clasificar directamente el nivel de oclusión, dado que las primeras componentes no muestran una separación clara para esta característica.