Informe Técnico: Detección de Grafitis "Street Scan"

Samuel Fernandez Ortiz
Samuel Botero Rivera
Universidad Nacional De Colombia Sede Medellin

7 de noviembre de 2025

Resumen

Este documento detalla el proceso técnico y los resultados del proyecto "Street Scan", una aplicación diseñada para la detección y geolocalización de grafitis en la ciudad de Medellín. Se describe la metodología de adquisición de datos mediante Mapillary, el entrenamiento de un modelo de detección de objetos (YOLO) y la implementación de una aplicación web para la visualización de resultados. Finalmente, se analizan las métricas del modelo y se presenta una conclusión crítica sobre la presencia de fuga de datos (data leakage) en el conjunto de entrenamiento, lo que conduce a resultados excesivamente optimistas.

1. Introducción

La gestión de elementos urbanos, como los grafitis, presenta un desafío logístico significativo para las administraciones de la ciudad. Cuantificar, clasificar y localizar estos elementos es un proceso costoso en tiempo y recursos. El proyecto "Street Scan"busca solucionar esta problemática mediante el uso de visión por computador y aprendizaje profundo.

El objetivo principal fue desarrollar una herramienta capaz de procesar imágenes a nivel de calle, predecir la presencia de grafitis (clasificándolos como artísticos o vandálicos) y mostrar su ubicación geográfica en un mapa interactivo.

2. Metodología y Proceso

El desarrollo del proyecto se dividió en cuatro fases principales: adquisición de datos, preparación y aumento de datos, entrenamiento del modelo y desarrollo de la aplicación de visualización.

2.1. Adquisición de Datos

La fuente principal de imágenes fue la plataforma Mapillary, que provee imágenes a nivel de calle con datos de geolocalización. Debido al gran volumen de datos y las limitaciones de procesamiento, el área de estudio (Medellín) se segmentó en 120 áreas rectangulares (BBoxes).

Se generó un archivo de texto (`lista_bboxes.txt`) con las coordenadas de estas áreas, que se utilizó para consultar la API de Mapillary y obtener los enlaces de descarga de las imágenes disponibles, resultando en un conjunto de datos inicial de aproximadamente 300,000 imágenes potenciales.

2.2. Preparación de Datos y Aumento

De las imágenes disponibles, se descargó un subconjunto (aprox. 10,000 imágenes) para el entrenamiento y evaluación. Estas imágenes fueron anotadas manualmente para crear las etiquetas (bounding boxes) de las clases "Graffiti Artísticoz "Graffiti Vandálico".

Para robustecer el modelo, se utilizó la plataforma Roboflow para aplicar técnicas de aumento de datos (*data augmentation*). Estas técnicas incluyeron:

- Cambios en la iluminación (brillo, contraste).
- Desenfoque (blur).
- Rotaciones y volteos leves.

Es crucial notar que este proceso de aumento se realizó sobre el conjunto de datos completo *antes* de dividirlo en los conjuntos de entrenamiento, validación y prueba.

2.3. Entrenamiento del Modelo

Se empleó un modelo de la familia YOLO (You Only Look Once), conocido por su alta eficiencia y precisión en tareas de detección de objetos en tiempo real. El entrenamiento se llevó a cabo utilizando la infraestructura de GPU de un notebook de Kaggle (ver `entrenamiento_del_modelo.ipynb`). El modelo fue entrenado para localizar y clasificar los grafitis según las etiquetas generadas. El modelo final (`best.pt`) se guardó para su uso en la aplicación.

2.4. Aplicación de Visualización

Se desarrolló una aplicación web (`app.py`) que permite a un usuario visualizar los resultados. La aplicación carga el modelo entrenado y procesa las imágenes del conjunto de prueba. Los resultados se presentan en un mapa interactivo (usando Folium) que muestra marcadores en las ubicaciones geográficas de las imágenes donde se detectaron grafitis, indicando la clase predicha y el nivel de confianza de la detección.

3. Resultados del Modelo

Las métricas de rendimiento del modelo se evaluaron en el conjunto de prueba. A continuación, se presentan las gráficas generadas durante el entrenamiento y la evaluación.

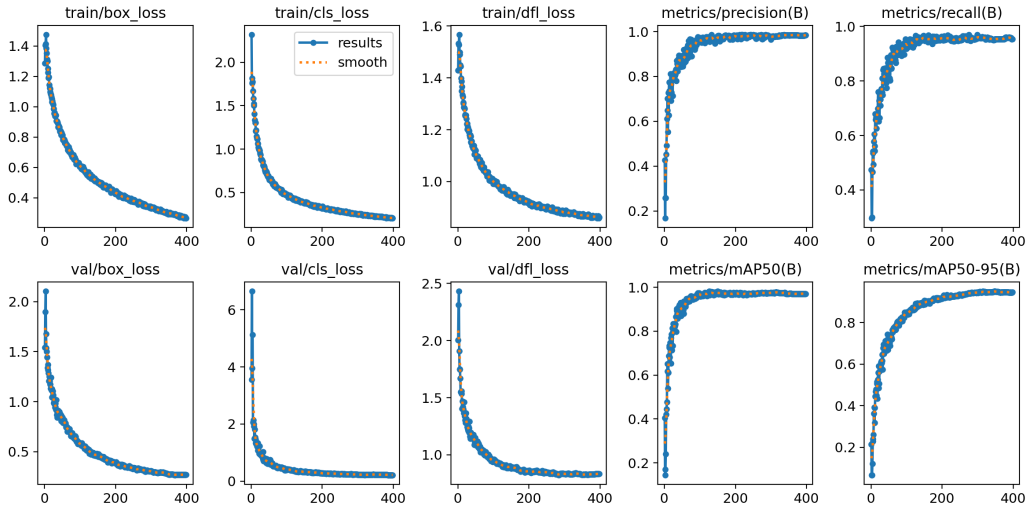

Figura 1: Curvas de entrenamiento del modelo. Muestra la evolución de las métricas de pérdida (loss) y precisión (mAP) a lo largo de las épocas.

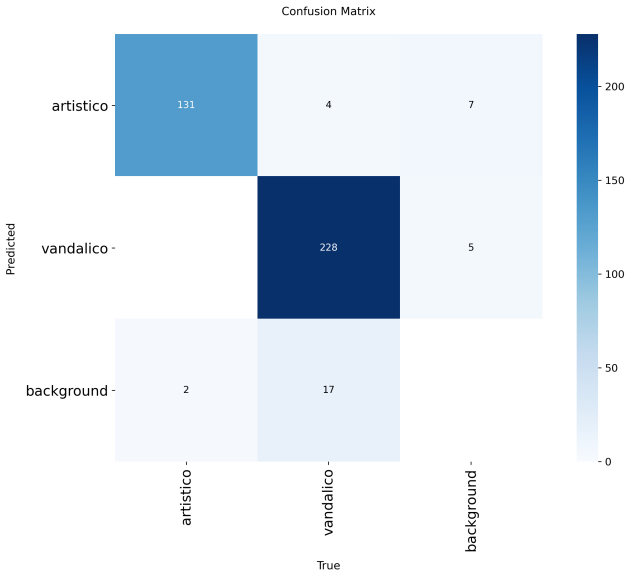

Figura 2: Matriz de confusión generada sobre el conjunto de prueba. Muestra las predicciones correctas e incorrectas por clase.

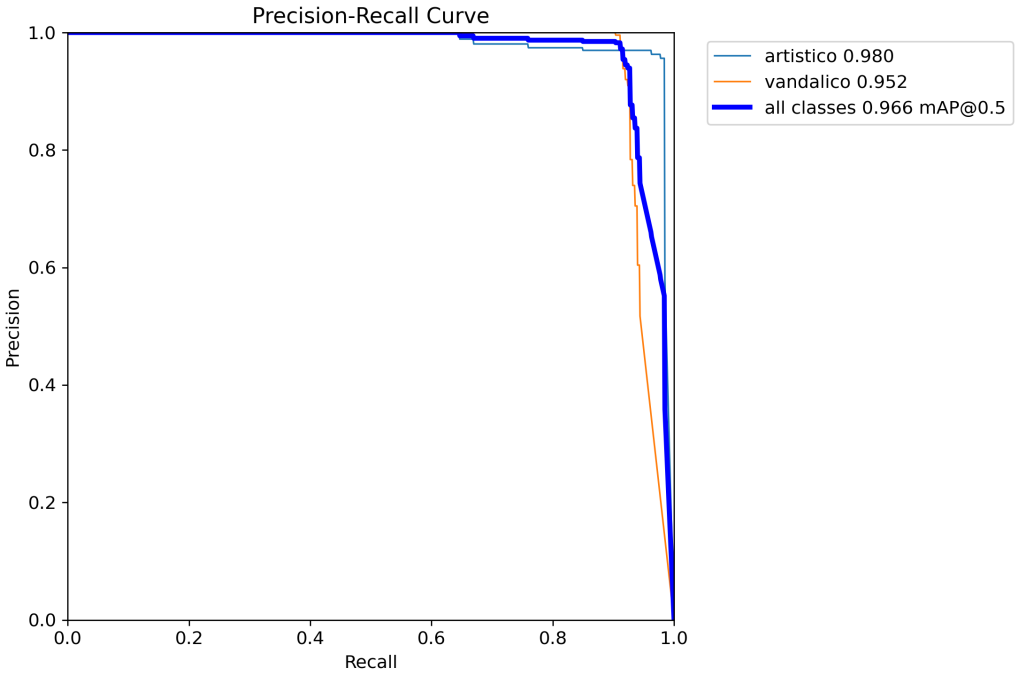

Figura 3: Curva de Precisión-Recall (PR) para la detección (Box).

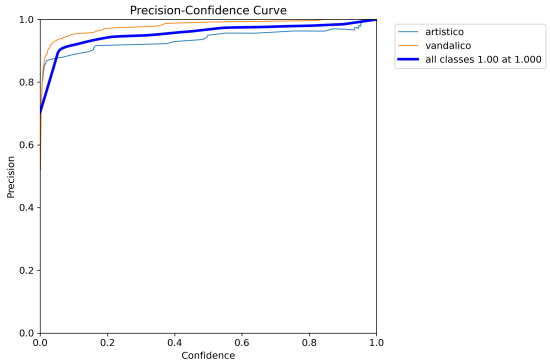

Figura 4: Curva de Precisión vs. Confianza.

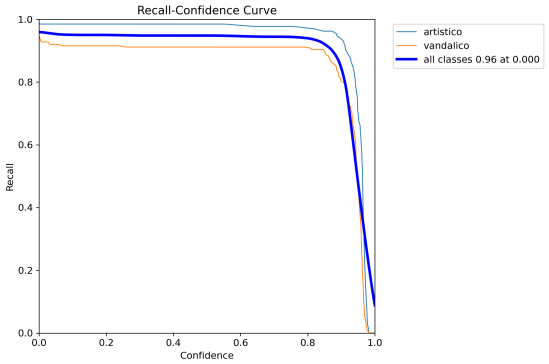

Figura 5: Curva de Recall vs. Confianza.

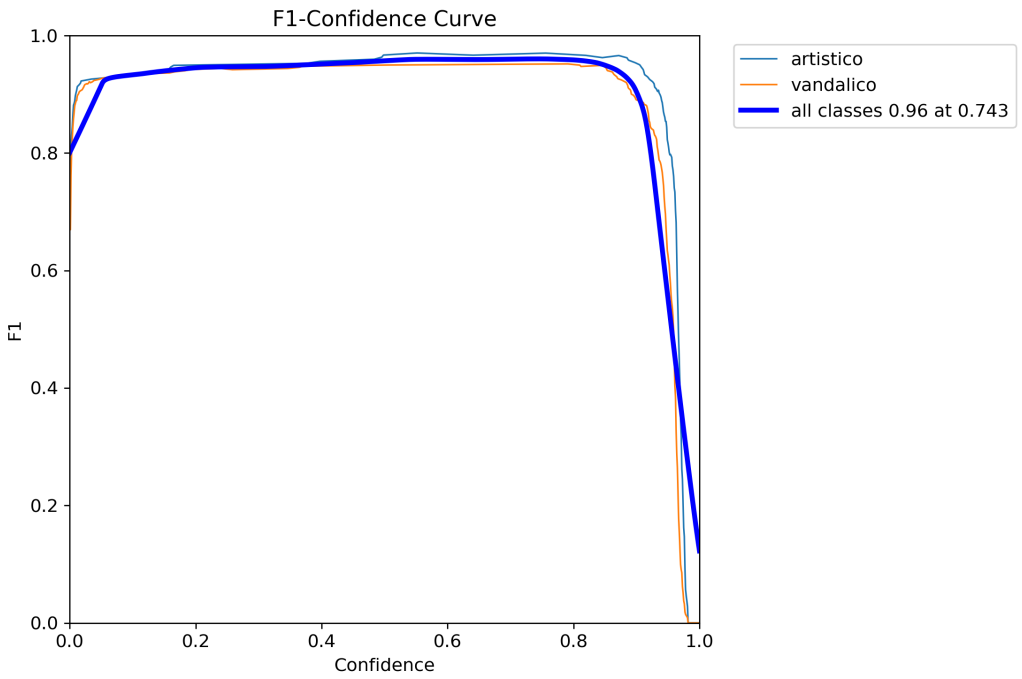

Figura 6: Curva F1 vs. Confianza, mostrando el balance óptimo entre precisión y recall.

4. Observaciones y Conclusiones

A primera vista, los resultados obtenidos (visibles en las curvas mAP y PR) son excepcionalmente buenos, sugiriendo un rendimiento casi perfecto del modelo. Sin embargo, un análisis más profundo de la metodología revela un problema crítico en la preparación de los datos.

Se evidencia una clara presencia de **fuga de datos (data leakage)**. Esto se debe a que la plataforma Roboflow se utilizó para el proceso de aumento de datos, donde se copiaron las mismas imágenes alterándolas solo un poco (aumento de iluminación, desenfoque, etc.). El problema fundamental es que este proceso se realizó *antes* de repartir los datos en los conjuntos de validación, entrenamiento y prueba.

Como resultado, imágenes casi idénticas (la original y sus aumentos) existen en múltiples conjuntos. Por ejemplo, una imagen original puede estar en el conjunto de prueba, mientras que una versión ligeramente desenfocada de la misma imagen está en el conjunto de entrenamiento. El modelo aprende a reconocer estas imágenes específicas en lugar de generalizar.

Por esto, los resultados son tan optimistas: el modelo está siendo evaluado en datos que, en esencia, ya ha "visto"durante el entrenamiento.

Para futuros trabajos, es imperativo corregir esta falla metodológica. El aumento de datos debe aplicarse *exclusivamente* al conjunto de entrenamiento y debe realizarse *después* de que la división de datos (train/validation/test) se haya completado de forma estricta.