INFORME DE IMPLEMENTACIÓN DE UN DETECTOR DE TELÉFONOS USANDO YOLOV5

Este informe describe el proceso de desarrollo, entrenamiento y optimización de un sistema de detección de teléfonos móviles utilizando YOLOv5, un modelo de detección de objetos basado en redes neuronales convolucionales (CNN). El objetivo es crear un detector eficiente y adaptable que pueda identificar en una imagen la presencia de un teléfono móvil que ha sido dejado en el suelo, trabajando con un conjunto de datos etiquetados y proporcionando coordenadas precisas de detección. Además, se exploran las posibles mejoras en el sistema, tanto a nivel de optimización del modelo como de mejora de los datos de entrada.

Descripción del Proceso de Trabajo

1. **Estructura de Datos y Preparación del Conjunto de Entrenamiento:** Para entrenar el modelo, se utilizó un conjunto de datos proporcionado por el cliente, que incluye imágenes y un archivo labels.txt que contiene las coordenadas de los teléfonos móviles en dichas imágenes. El proceso de preparación de los datos involucró los siguientes pasos:

   - Organización de Imágenes y Etiquetas: Se generó una estructura de carpetas compatible con YOLOv5, creando directorios separados para las imágenes (./images) y las etiquetas (./labels). Cada imagen tenía un archivo de etiqueta correspondiente en formato YOLO (con coordenadas del centro y dimensiones normalizadas).
   - Conversión de Etiquetas: A partir del archivo labels.txt, se convirtieron las coordenadas proporcionadas en el formato adecuado para YOLOv5, es decir, las etiquetas incluyeron el identificador de clase (teléfono móvil = 0) seguido por las coordenadas del centro de la caja delimitadora, su ancho y alto, todo normalizado entre 0 y 1.

2. **Arquitectura del Modelo:** El modelo utilizado fue YOLOv5s, una versión ligera y eficiente del conjunto de modelos YOLOv5. Este modelo fue seleccionado por su balance entre velocidad y precisión, lo que lo hace ideal para aplicaciones en tiempo real, como la detección de teléfonos móviles.

   El archivo de configuración de la arquitectura del modelo (yolov5s.yaml) fue utilizado para definir la estructura de la red, que consta de una serie de capas convolucionales intercaladas con capas de normalización y activación para detectar objetos en múltiples escalas.

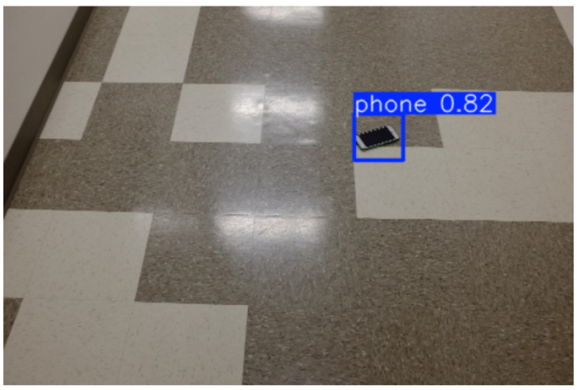
3. **Entrenamiento del Modelo:** El entrenamiento se realizó utilizando el script train.py de YOLOv5, modificado para aceptar las imágenes y etiquetas preparadas. Los principales parámetros utilizados fueron:

   - Tamaño de las imágenes: 640x640 píxeles.
   - Número de épocas: 50 épocas para obtener resultados preliminares.
   - Tamaño del batch: 16 imágenes por iteración.
   - Pesos pre-entrenados: Se utilizaron los pesos pre-entrenados de yolov5s como punto de partida, lo que permitió aplicar la técnica de fine-tuning.

El modelo fue entrenado utilizando las imágenes etiquetadas del cliente, y los resultados fueron almacenados para su análisis posterior.

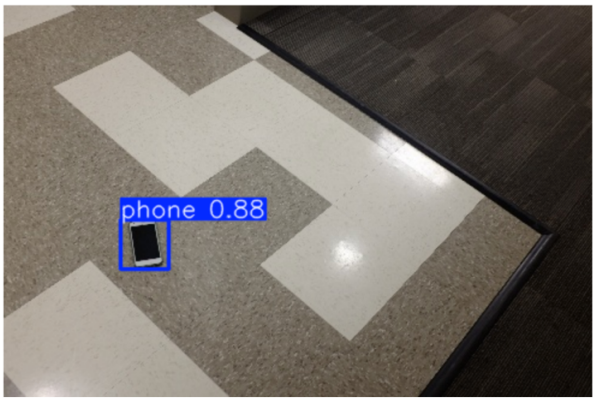
A continuación, algunos resultados de las imágenes con el teléfono detectado y sus respectivas coordenadas reales y de predicción:

Coordenadas Predichas: (x: 0.6578, y: 0.3449)
Coordenadas Reales: (x: 0.6551, y: 0.3466)
Imagen: 8.jpg, Distancia: 0.0031

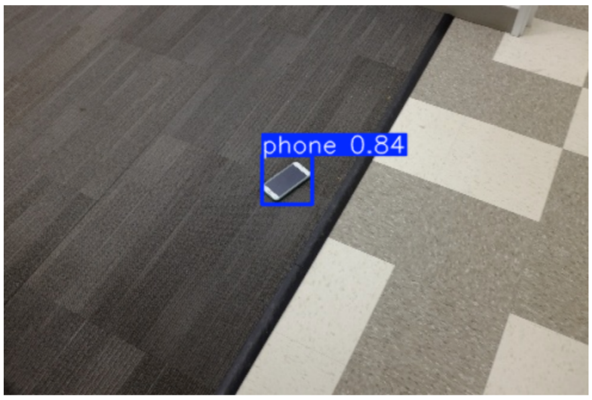


Precisión de predicción: 100.00%

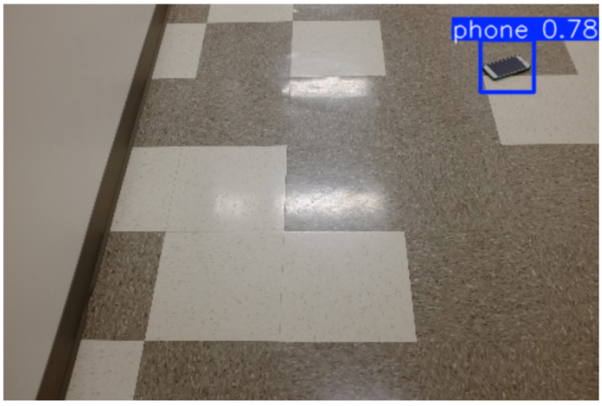
Coordenadas Predichas: (x: 0.2404, y: 0.6099)
Coordenadas Reales: (x: 0.2388, y: 0.6012)
Imagen: 51.jpg, Distancia: 0.0088



Coordenadas Predichas: (x: 0.4849, y: 0.4468)
Coordenadas Reales: (x: 0.4837, y: 0.4509)
Imagen: 119.jpg, Distancia: 0.0043



Coordenadas Predichas: (x: 0.8463, y: 0.1566)
Coordenadas Reales: (x: 0.8449, y: 0.1564)
Imagen: 58.jpg, Distancia: 0.0014



Soluciones No Tomadas

Durante el desarrollo del proyecto, se consideraron varias alternativas que finalmente no fueron implementadas por motivos de tiempo o complejidad:

- Uso de Redes Alternativas a YOLOv5: Se evaluaron otras arquitecturas de detección de objetos, como Faster R-CNN y EfficientDet. Aunque ofrecen buena precisión, se decidió continuar con YOLOv5 por su velocidad y facilidad de uso en producción.
- Aumento Extensivo de Datos (Data Augmentation): El uso de técnicas avanzadas de aumento de datos como rotación, escala, cambios de color, y deformaciones no fue implementado en esta fase. Estas técnicas podrían mejorar la capacidad del modelo para generalizar en situaciones más variadas, como ángulos de cámara inusuales o condiciones de iluminación adversas.
- Mejoras en la Red Utilizando Transfer Learning con Más Datos: Aunque se utilizó fine-tuning sobre un modelo pre-entrenado, no se realizaron experimentos con otros grandes conjuntos de datos específicos de detección de objetos pequeños, lo que podría mejorar la precisión del modelo en objetos pequeños como teléfonos móviles.

Próximos Pasos para Mejorar el Detector

1. **Optimización de Hiperparámetros:** Se recomienda realizar una búsqueda de hiperparámetros para encontrar los valores óptimos para el número de épocas, tamaño del batch, tasa de aprendizaje y otros parámetros del modelo. Esto podría mejorar el rendimiento del detector al ajustarlo mejor a las características del conjunto de datos proporcionado.

2. **Aumento de la Cantidad de Datos:** El detector podría beneficiarse significativamente de un mayor volumen de datos de entrenamiento, especialmente si se incluyen ejemplos en condiciones diversas (diferentes ángulos, iluminación, resolución, etc.). Técnicas como Data Augmentation pueden ayudar a incrementar el tamaño efectivo del conjunto de datos sin necesidad de obtener nuevas imágenes.

3. **Uso de Modelos más Grandes:** Experimentar con versiones más grandes de YOLO como YOLOv8, que podrían proporcionar una mayor precisión a cambio de un tiempo de inferencia mayor.

Conclusiones

El trabajo realizado para la detección de teléfonos móviles usando YOLOv5 demuestra que este modelo es una solución eficaz para la tarea de detección de objetos pequeños en imágenes. A través del uso de técnicas de fine-tuning, se logró adaptar un modelo pre-entrenado para obtener resultados razonables en un conjunto de datos específico. Sin embargo, hay un amplio margen para mejorar, especialmente mediante la optimización de hiperparámetros y aumento del conjunto de datos.