

Sistema de detección en tiempo real de publicidad en la vía pública

del Mazo, Federico

Pastine, Casimiro

Tutor: Ing. Martín Buchwald





Motivaciones Personales

- Trabajar sobre un **campo desconocido** para nosotros, diferente a nuestras carreras profesionales
- Hacer un proyecto aplicando lo **aprendido en la carrera**
- Lograr un **balance** entre un **proyecto práctico** y una **exploración teórica** con su debida investigación
- Elaborar un trabajo **modularizable** y fácilmente **extensible**

Inspiración



BART

blocking ads in real time



bart: blocking ads in real time ^④



Configuration

Model

s2oeb dataset - 8155 images

Transformation

Detect

Detection score

0 65 100

Enable profiling

A screenshot of a software interface for real-time advertising detection. The interface has a dark theme with white text. At the top, it says "Configuration" and "Model" with a dropdown menu set to "s2oeb dataset - 8155 images". Below that is a "Transformation" section with a dropdown menu set to "Detect". Under "Detection score", there is a horizontal slider with a red track and a blue dot at the 65 mark, with numerical labels "0" and "100" at the ends. At the bottom, there is a checkbox labeled "Enable profiling" with a small circular icon next to it.

BART

blocking ads in real time



BART

blocking ads in real time



Objetivos

Detectar anuncios

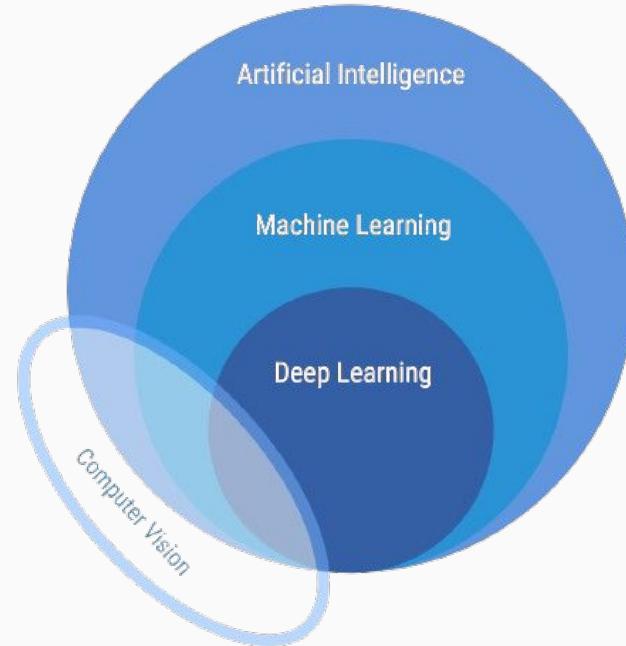
- ¿Qué es un anuncio?
- ¿Existen conjuntos de datos públicos de anuncios?
- ¿Es posible detectar en videos tan fácilmente como en imágenes?

Tiempo real

- ¿Cómo detectamos objetos plano por plano?
- ¿Es posible detectar más rápido de lo que se filma?
- ¿Cómo se puede disponibilizar?
- ¿Se puede hacer desde un celular?

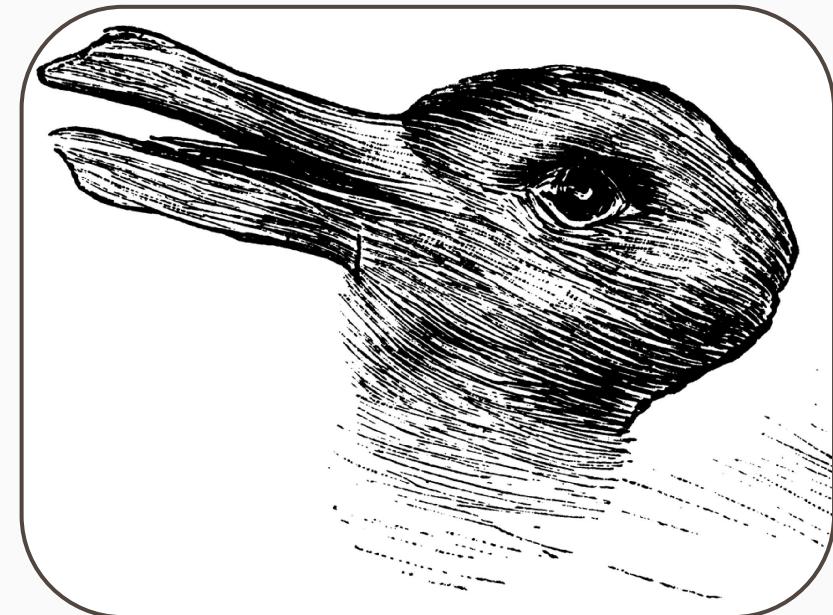
Computer Vision

Lograr que un sistema
entienda el contenido de una
imagen digital



Computer Vision

Entender el contenido de una
imagen digital no es una
tarea simple



Tareas de Computer Vision

Image Classification



La clasificación **categoriza**

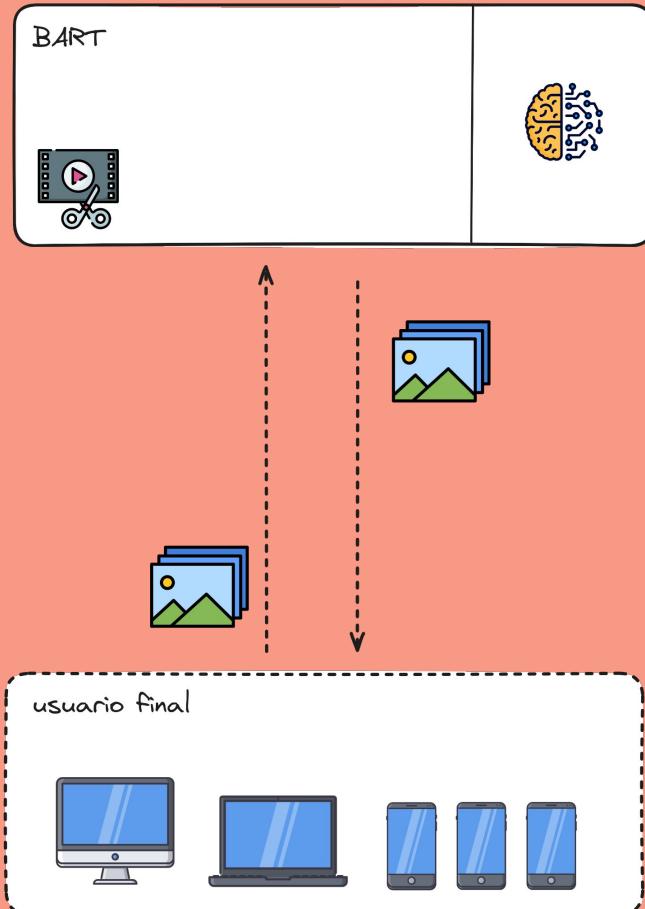
Object Detection



La detección **localiza**

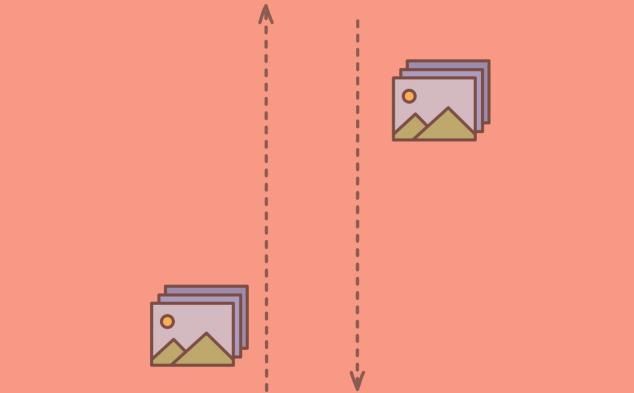
Objetivos

1. Construir un sistema que detecte **carteles publicitarios** dada una imagen
2. **Recibir y entregar** los videos del usuario en **tiempo real**
3. **Presentar** al usuario **las detecciones** del modelo



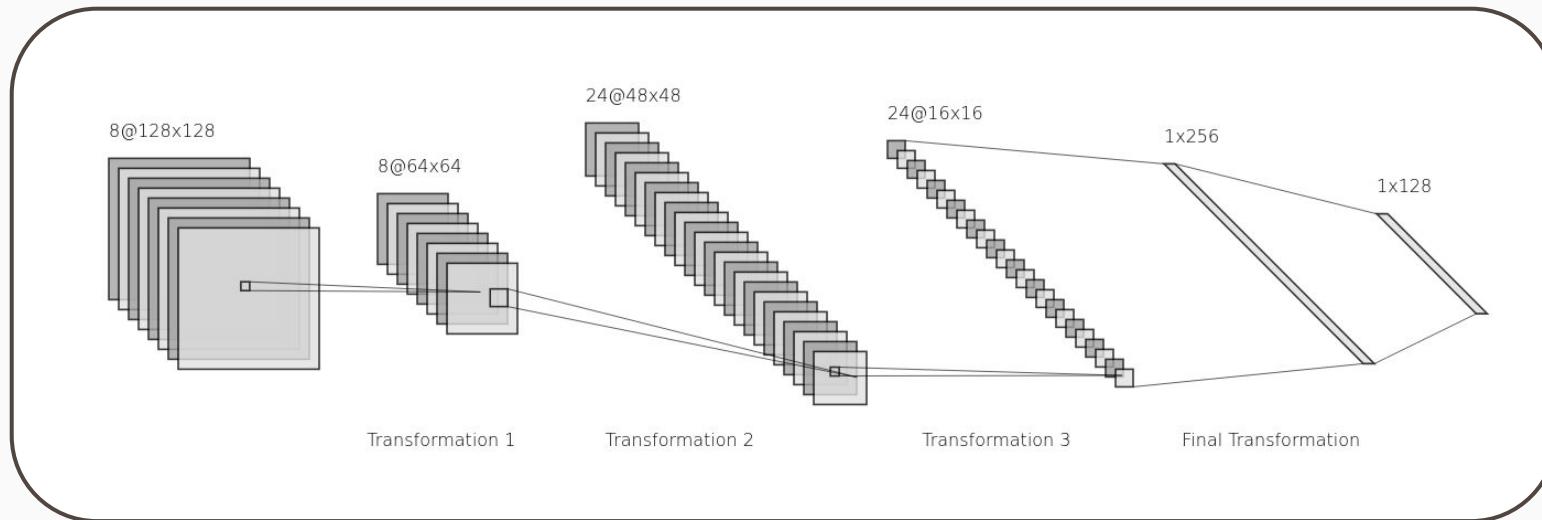
Objetivos

1. Construir un sistema que detecte **carteles publicitarios** dada una imagen



Redes Neuronales Convolucionales

Sistema de capas sucesivas encadenadas que reducen los datos de entrada a sus características principales



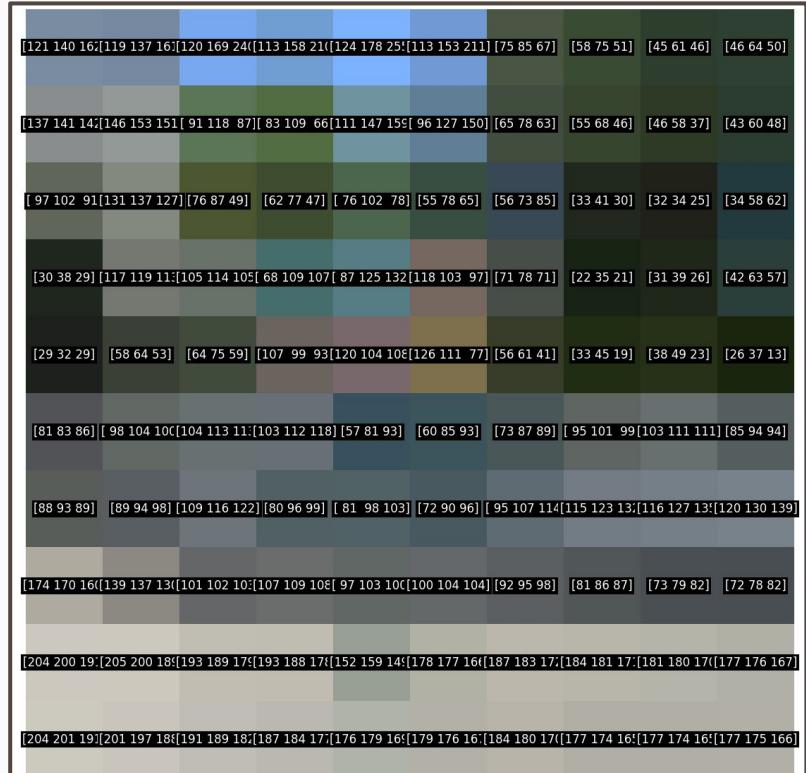
Redes Neuronales Convolucionales

Los datos de entrada son imágenes



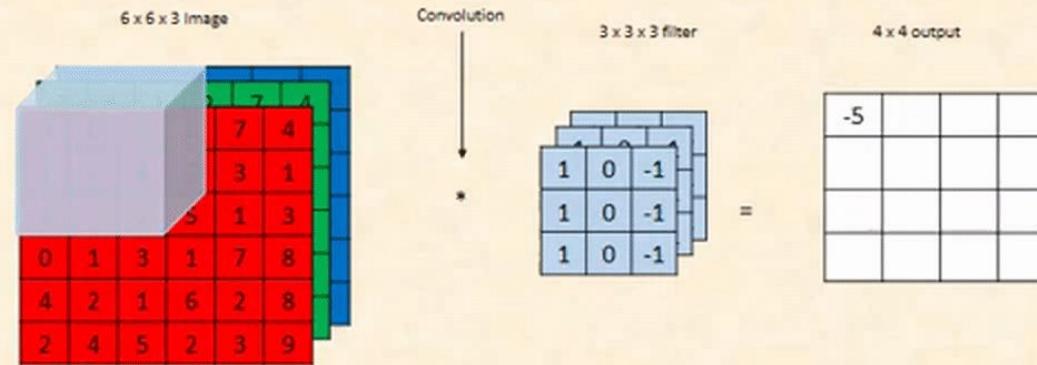
Redes Neuronales Convolucionales

Los datos de entrada son imágenes expresadas como matrices



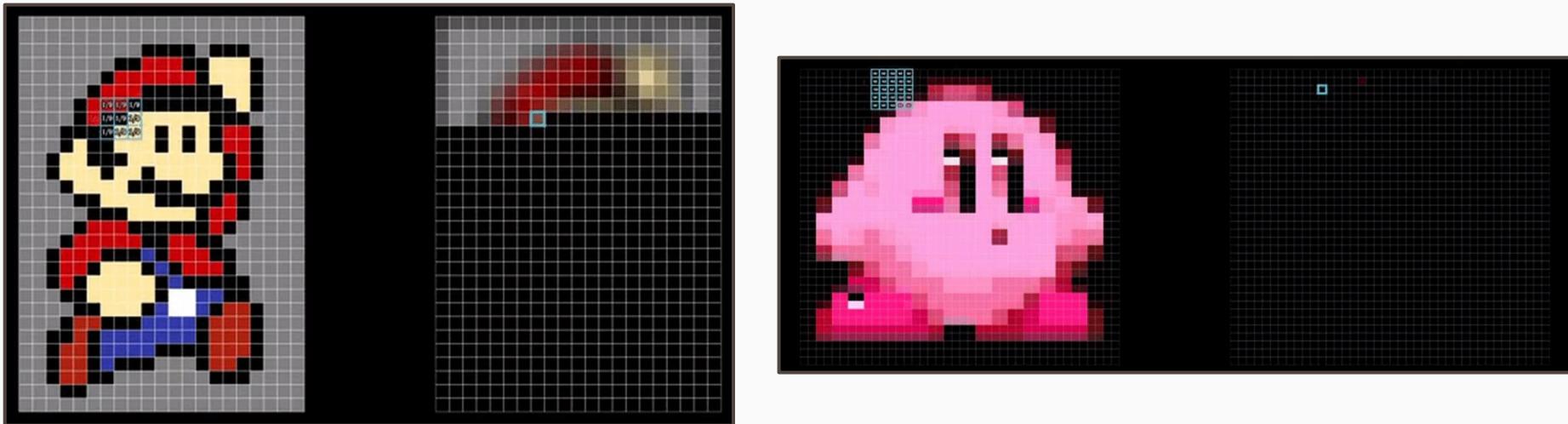
Redes Neuronales Convolucionales

La operación fundamental es la **convolución**



Convoluciones

Multiplicación de matrices para transformar los datos de entrada y encontrar las características principales



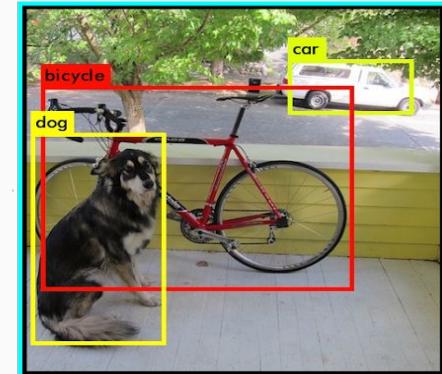
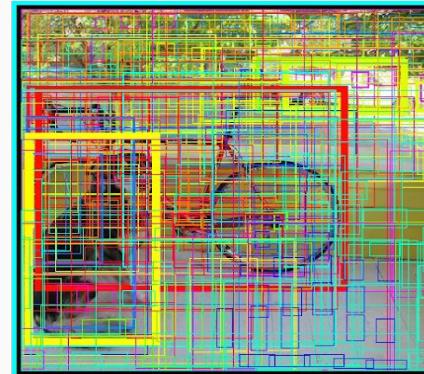
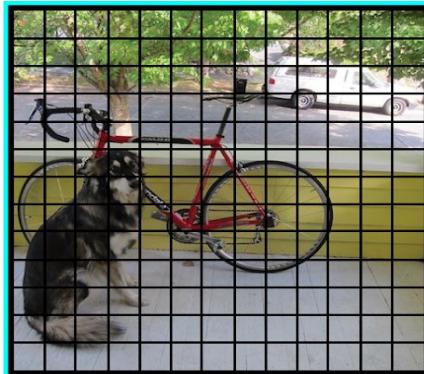
YOLO: You Only Look Once

Algoritmos de detección de objetos

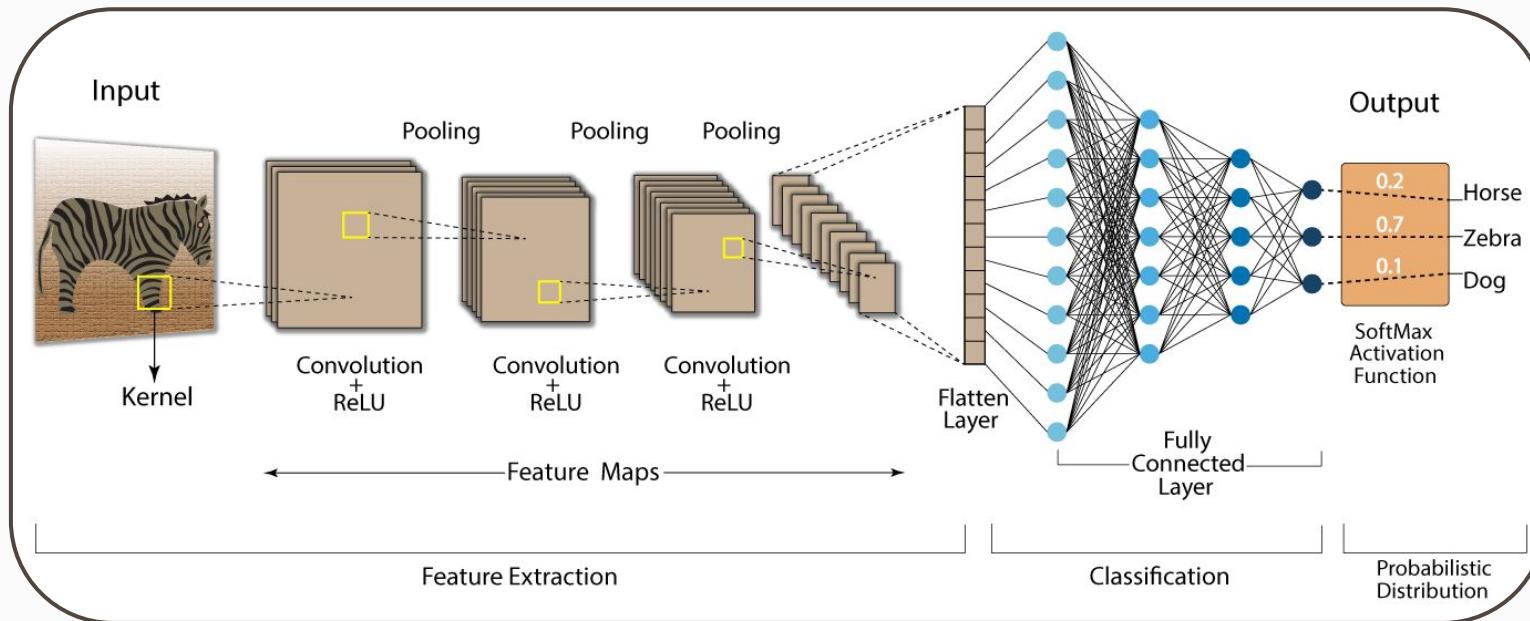
Un método **moderno** de una **única etapa** que balancea la **velocidad y precisión** de las detecciones

Divide la imagen en regiones, **predice** los objetos presentes y **descarta** las predicciones redundantes

Nosotros utilizamos la **octava generación** del algoritmo, provista por



Redes Neuronales Convolucionales





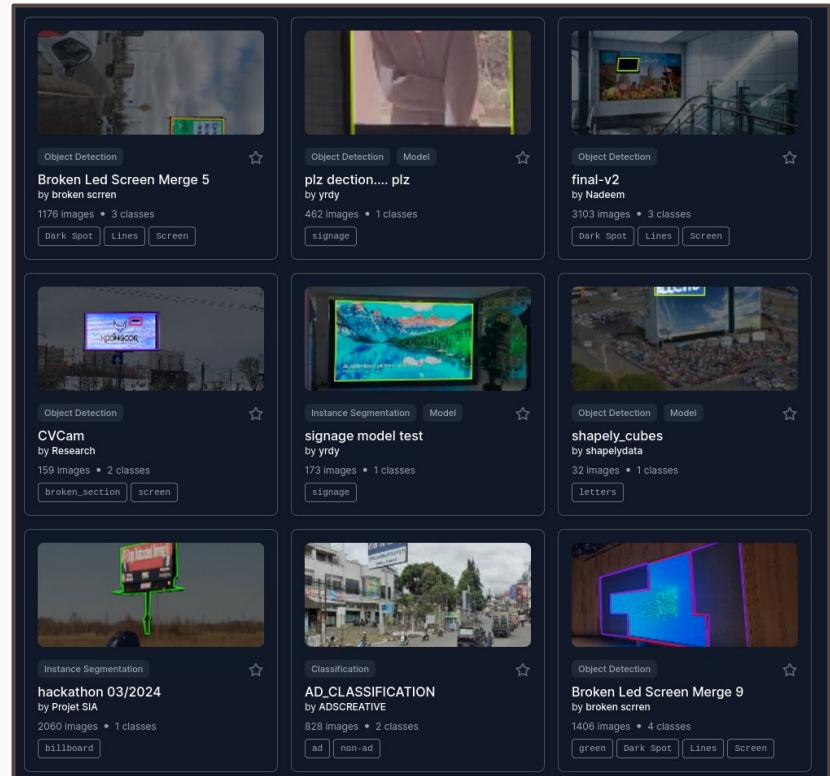
Entrenamiento Supervisado

- Requiere imágenes para las cuales **ya conocemos el resultado deseado**
- **Múltiples iteraciones** de entrenamiento por la red neuronal
- Al finalizar cada una, se calculan los errores en las predicciones y se **actualizan los parámetros** internos de la red
- Al finalizar el entrenamiento, se **evalúa el desempeño final** con el subconjunto de pruebas

Construcción de Datos de Entrenamiento

Encontramos varios conjuntos de datos de carteles publicitarios en **bases de datos públicas** como  **roboflow**

Combinamos lo encontrado para generar nuevos conjuntos más diversos para mitigar posibles sesgos



Construcción de Datos de Entrenamiento

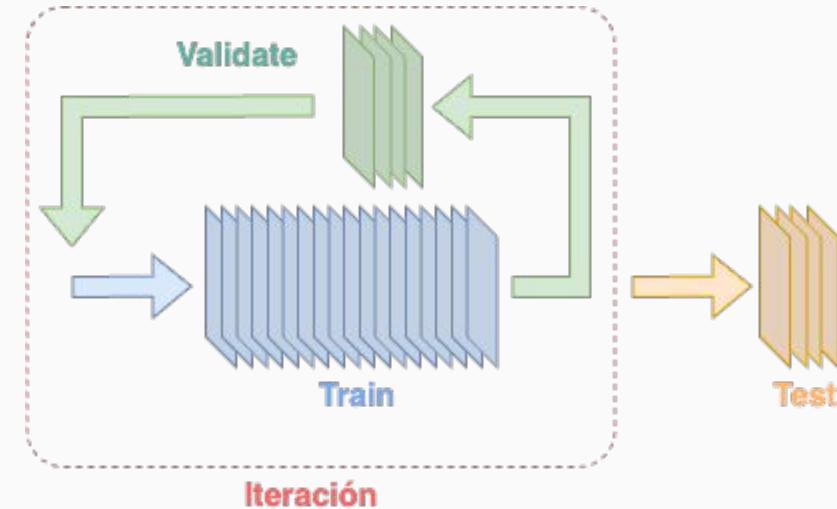


Construcción de Datos de Entrenamiento

La mayor parte de los datos se utilizan para el **proceso de aprendizaje**

Una porción no significativa se utiliza para **re-calibrar el entrenamiento** luego de cada iteración

Una pequeño porcentaje se separa para **probar el modelo sobre datos desconocidos** y evaluar el rendimiento final



Matriz de Confusión

La matriz de confusión nos permite **distribuir las predicciones del modelo en 4 categorías** y entender dónde nos falta mejorar

Debemos **minimizar los errores y sesgos** que traigan los conjuntos de datos



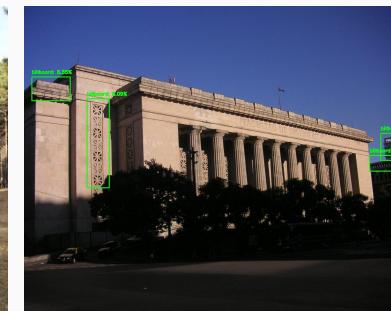
falso negativo ✗



verdadero negativo ✓



verdadero positivo ✓



falso positivo ✗

Técnicas de Aumento de Datos

Negative Images

- Busca **introducir imágenes “vacías”** para que el modelo entienda que **no siempre va a haber un cartel** en la foto
- Utilizamos el dataset público de COCO con **imágenes de la vía pública sin carteles publicitarios**
- **Aumenta** el conjunto entre un **10%** y un **15%** del tamaño original

Data Augmentation

- Busca **mejorar la tolerancia del modelo** a imágenes de peor calidad, imperfectas o incluso cortadas
- **Transformamos las imágenes del conjunto original** para aumentar el volumen
- Las imágenes se pueden **rotar, re-escalar, invertir, recortar**, etc.



Nuestros Datos

11 conjuntos de datos distintos

Más de **52.000** fotos de carteles publicitarios

300+ horas de entrenamiento



¿Cómo Hacer un Buen Modelo?

- Debemos **balancear la precisión y velocidad** del modelo eligiendo correctamente la **cantidad de capas** que este va a tener
- Es **crucial** elegir la **arquitectura de red** más adecuada para el problema que queremos resolver
 - Cuantas **más capas** de convolución tengamos, **mejores** van a ser las **predicciones** del modelo
 - Cuantas **menos capas** de convolución tengamos, **más rápidas** van a ser las **predicciones** del modelo



¿Cómo Hacer un Buen Modelo?

Mejorar la **velocidad** del modelo muchas veces **implica bajar su precisión**

Probamos **5 arquitecturas distintas** para llegar a un **balance adecuado** de velocidad y precisión que nos permita lograr una **detección en tiempo real de buena calidad**

Velocidad

La definimos como la cantidad de segundos que se toma el modelo para realizar una predicción sobre una imagen

Precisión

Hicimos análisis cualitativos y cuantitativos para determinar si un modelo contaba con una precisión aceptable

Precisión de Manera Cuantitativa

El resultado de una predicción de detección de objetos es el rectángulo minimal que contiene al objeto detectado



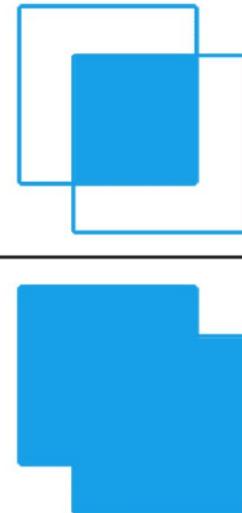
Precisión de Manera Cuantitativa

IoU (Intersection over Union) es la métrica que usamos para medir la precisión de nuestros modelos

Un valor de IoU de **1** indica una predicción **perfecta** que coincide con la realidad

Un valor de IoU de **0** indica que no hay intersección entre la realidad y lo predicho

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$



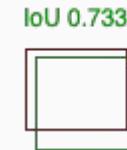
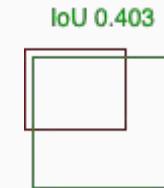
Precisión de Manera Cuantitativa

IoU (Intersection over Union) es la métrica que usamos para medir la precisión de nuestros modelos

Un valor de IoU de **1** indica una predicción **perfecta** que coincide con la realidad

Un valor de IoU de **0** indica que no hay intersección entre la realidad y lo predicho

Predicción Mala



Predicción Aceptable



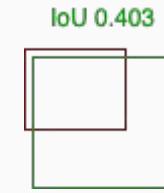
Predicción Buena

Precisión de Manera Cuantitativa

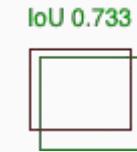
IoU nos sirve para **medir aciertos y errores** en **base a un umbral** que fijamos

Usando **IoU 0.5** nos permite admitir una predicción con un **IoU > 0.5 como acertada y** una predicción con un **IoU < 0.5 como un error**

IoU < 0.5 es error



IoU > 0.5 es acierto

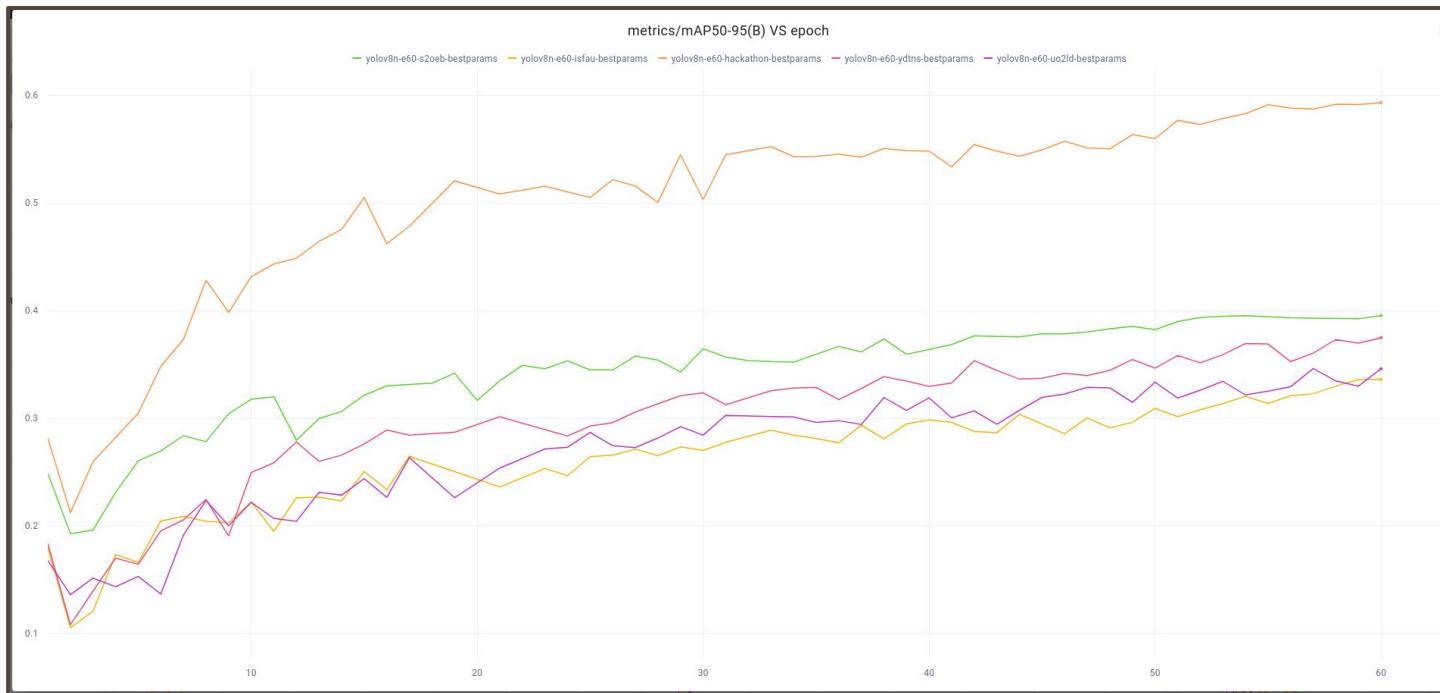


IoU > 0.5 es acierto





Resultados



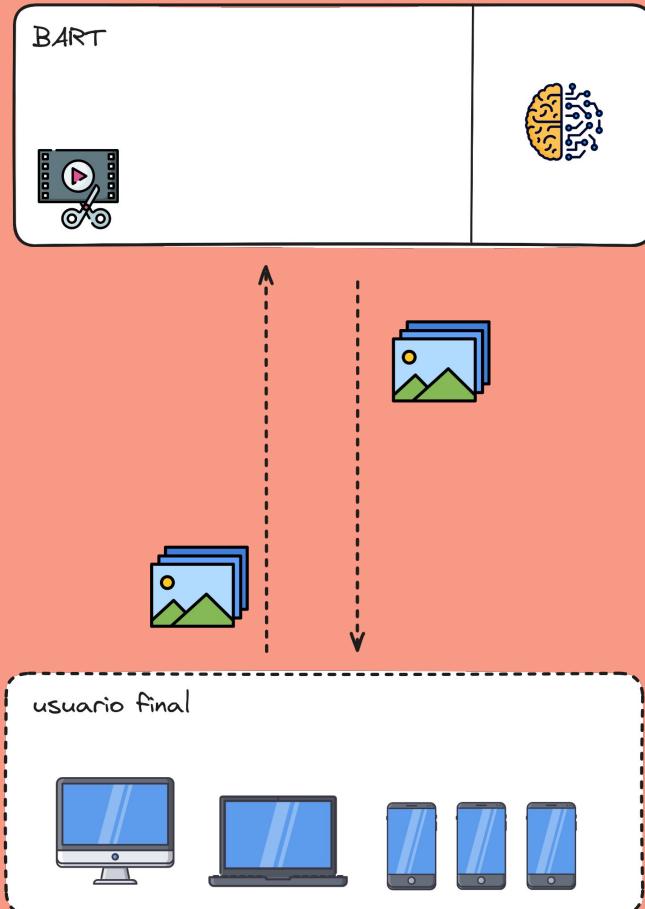


Resultados

Model Name	# Training Images	mAP50 (training)	mAP50 (validation)
multi	2514	0.862	0.540
hackathon	4503	0.807	0.496
isfau	7137	0.658	0.602
uo2ld	1903	0.663	0.597
ydt�s	2719	0.687	0.626
s2oeb	8155	0.680	0.636

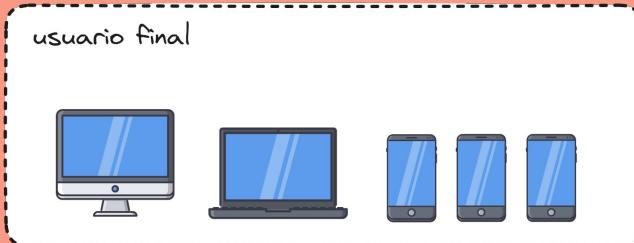
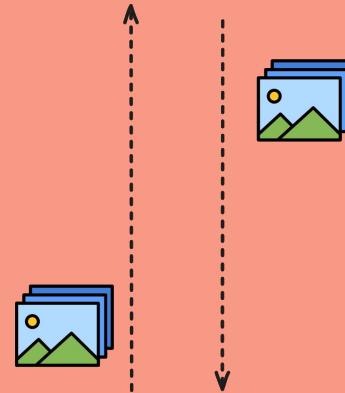
Objetivos

1. Construir un sistema que detecte **carteles publicitarios** dada una imagen
2. **Recibir y entregar** los videos del usuario en **tiempo real**
3. **Presentar** al usuario **las detecciones** del modelo



Objetivos

2. Recibir y entregar los videos del usuario en tiempo real





Despliegue y Acceso

- No queremos entregar un script por línea de comando
- Priorizamos que se pueda acceder **desde celulares que no están conectados a WiFi**
- Precisamos de un entorno que pueda **procesar la entrada del usuario** fácilmente

Despliegue y Acceso

- No queremos entregar un script por línea de comando
- Priorizamos que se pueda acceder **desde celulares que no están conectados a WiFi**
- Precisamos de un entorno que pueda **procesar la entrada del usuario** fácilmente



Posibilidades

Cliente

- ¿Puede un celular ejecutar nuestro modelo eficientemente?
- ¿Hay forma de tener un buen rendimiento, sea en el dispositivo que sea?

Cliente-Servidor

- ¿Se pueden enviar y recibir todos los fotogramas en tiempo real?
- ¿Cómo impacta la latencia?
- ¿Dónde lo hosteamos?

¿se puede hacer en Python?

Posibilidades

Cliente



Web Assembly

Un estándar moderno con
mucho futuro y gran
adopción...

Posibilidades

Cliente



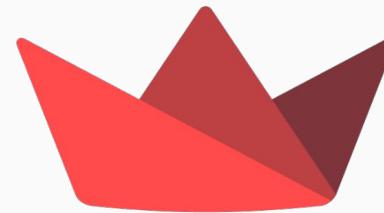
Web Assembly

Un estándar moderno con mucho futuro y gran adopción... al que todavía le falta madurez en Python

Posibilidades

Cliente-Servidor

Un *framework* para
disponibilizar herramientas de
Python fácilmente



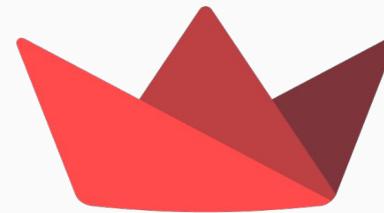
Streamlit

Posibilidades

Cliente-Servidor

Un *framework* para
disponibilizar herramientas de
Python fácilmente

- Python ✓
- Especializado para ciencia de datos ✓
- Velocidad ✓
- Acceso a cámara web ✓

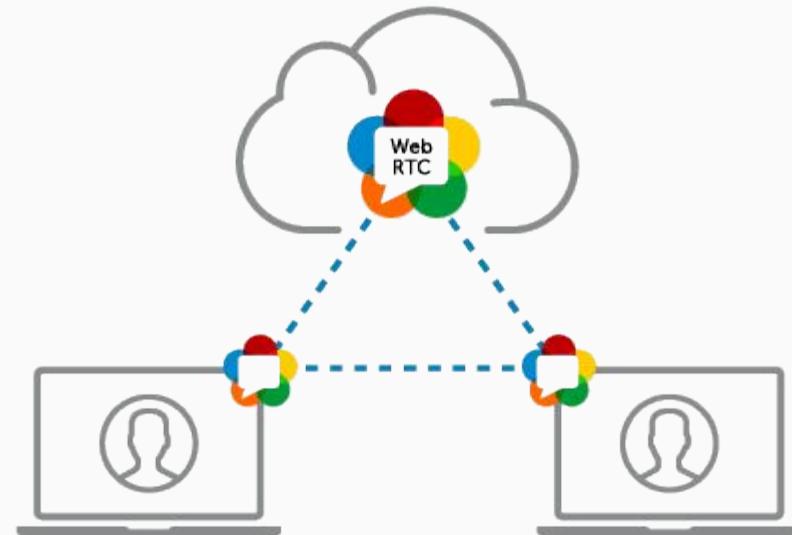


Streamlit

Comunicación

WebRTC

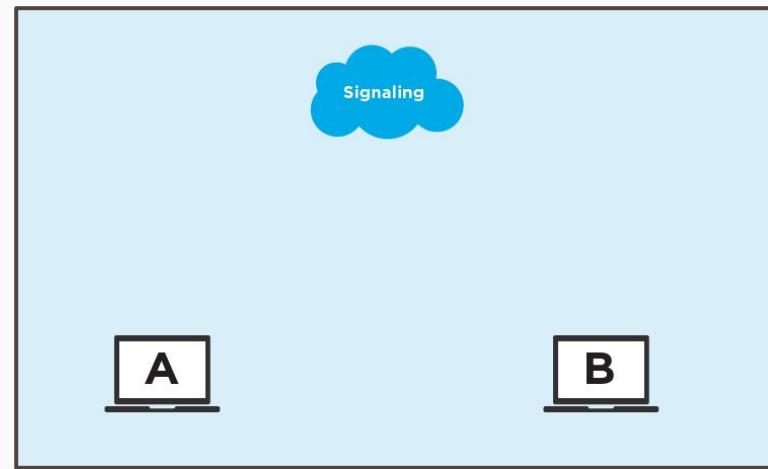
- Es un protocolo moderno que permite **comunicaciones en tiempo real**
- Se utiliza para **videollamadas, streaming de videos, chat**
- Coordina y establece **conexiones peer-to-peer** entre los clientes



Comunicación

WebRTC

- Utiliza un **servidor de señalización** llamado **STUN** para establecer la conexión entre pares
- Una vez establecida la conexión los **datos viajan de un cliente al otro** sin pasar por el servidor STUN



Optimizaciones

Cantidad de cuadros

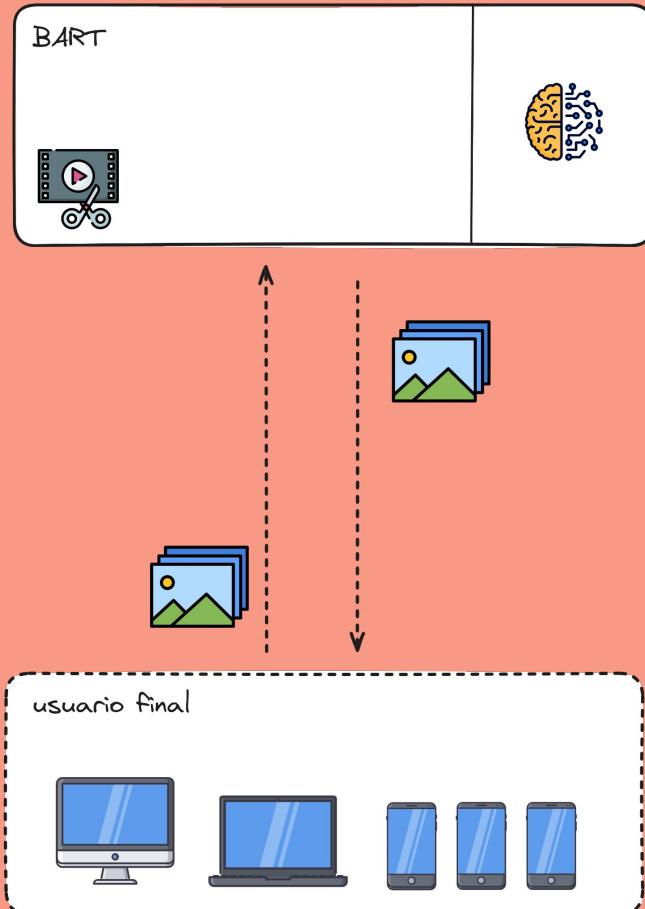
Al enviar menos cuadros podemos mejorar la velocidad total de procesamiento, pero hay que cuidar la experiencia final del usuario

Resolución de cada cuadro

Imagenes de menor resolución se procesan más rápido (aunque sus detecciones son menos precisas)

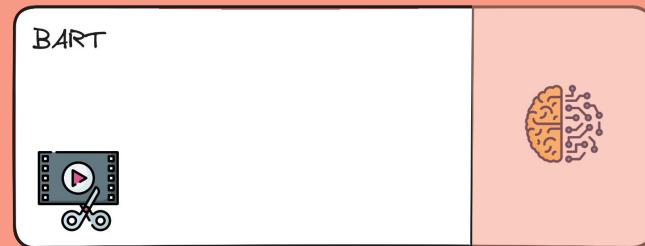
Objetivos

1. Construir un sistema que detecte **carteles publicitarios** dada una imagen
2. **Recibir y entregar** los videos del usuario en **tiempo real**
3. **Presentar** al usuario **las detecciones** del modelo



Objetivos

3. Presentar al usuario las detecciones del modelo



Ejecución del modelo

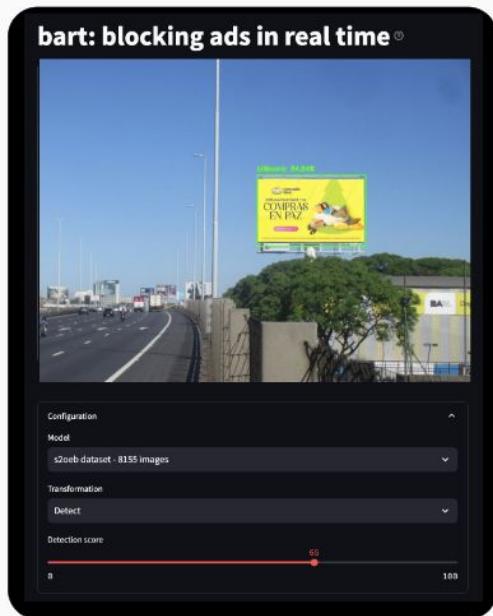
- La **interpretación** de un modelo es un proceso enteramente **independiente de su entrenamiento**
- Consiste en la **inferencia**: ejecutar el modelo sobre información nueva y tomar conclusiones de esta
- Se requiere de **interoperabilidad** entre el proceso de entrenamiento y el ambiente de ejecución



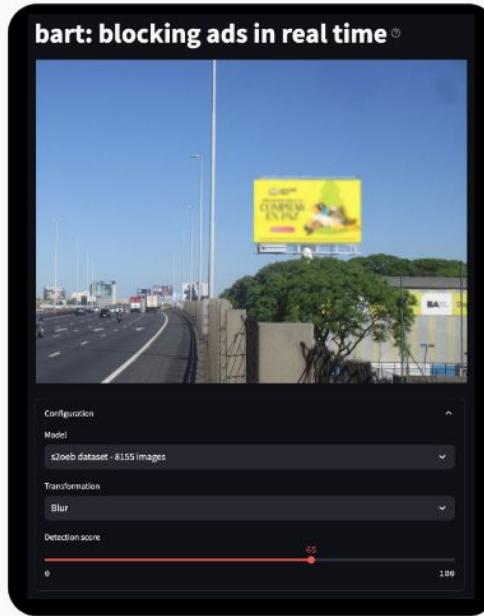


Transformaciones

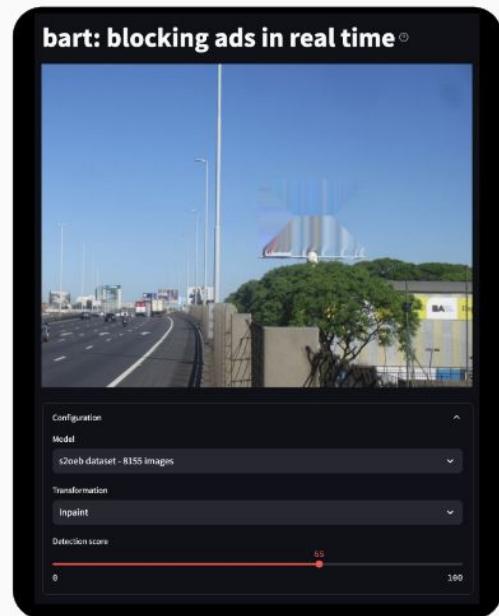
Detección



Difuminado



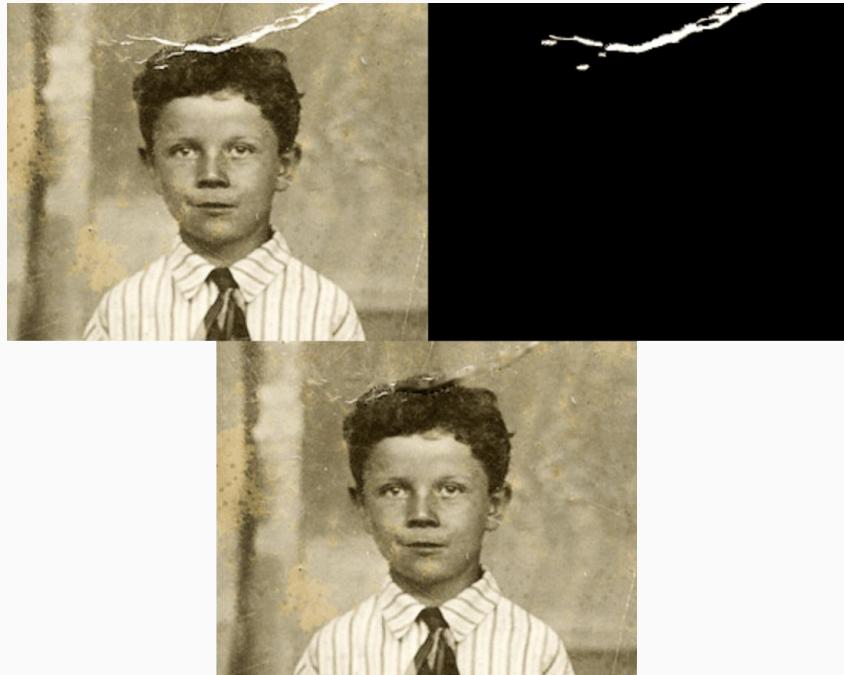
Eliminación



Eliminación

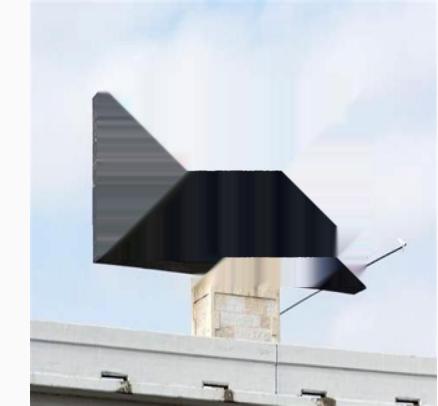
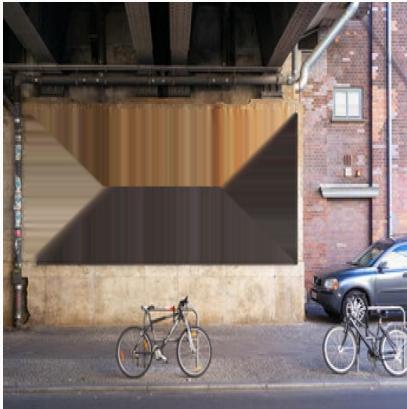
Image Inpainting

- Técnica de **reconstrucción de porciones faltantes o dañadas dentro de una imagen**
- Es utilizada principalmente en la **restauración de imágenes**
- Puede ser utilizada para **quitar objetos de una imagen bajo ciertas condiciones**
- **NO** está pensada para nuestro caso de uso



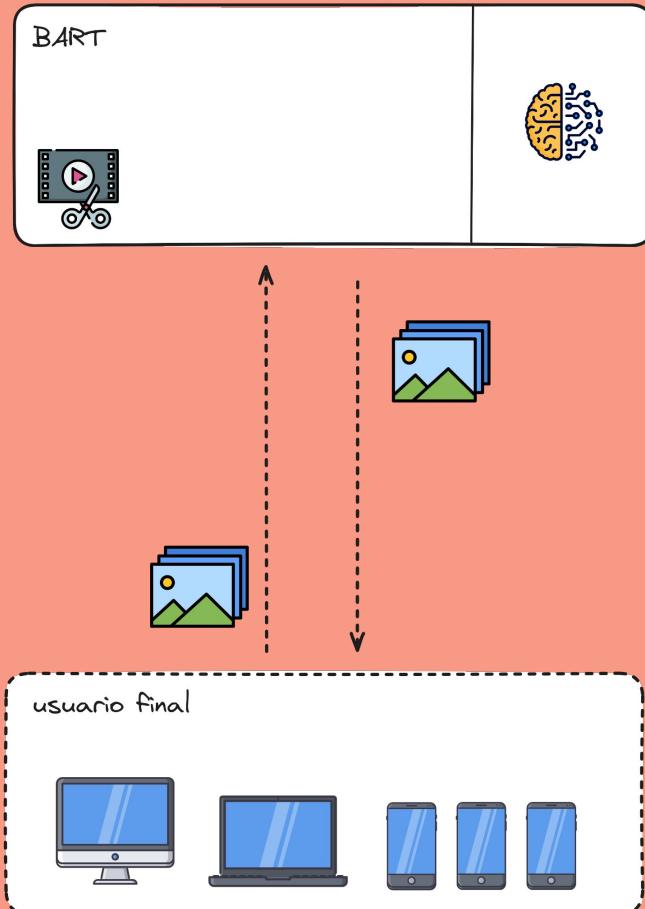
Eliminación

Image Inpainting



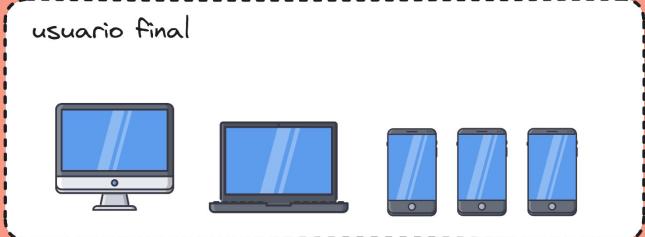
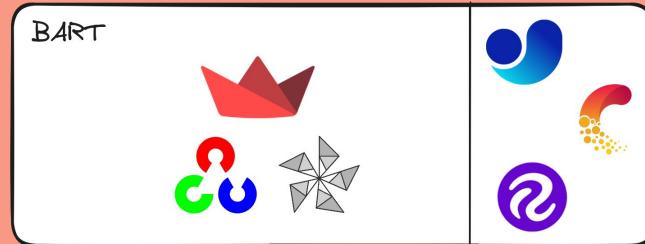
Objetivos

1. Construir un sistema que detecte **carteles publicitarios** dada una imagen
2. **Recibir y entregar** los videos del usuario en **tiempo real**
3. **Presentar** al usuario **las detecciones** del modelo



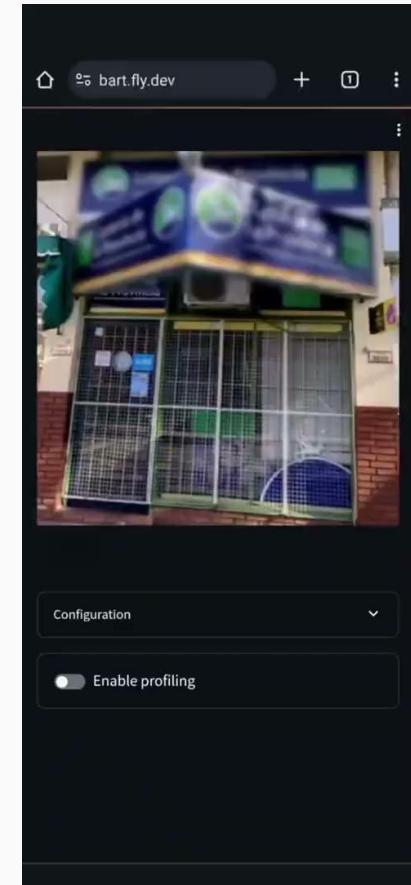
Arquitectura Final

1. Un modelo a base de **YOLO**, entrenado sobre datos públicos
2. Un **framework web** funcionando sobre **WebRTC**
3. La **interpretación** del modelo y post-procesamiento de los resultados





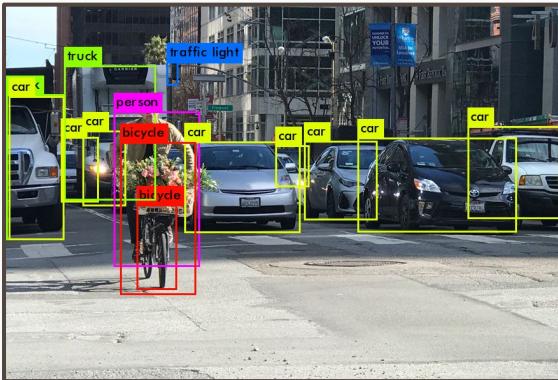
bart.fly.dev



Conclusiones

Posibles extensiones del proyecto

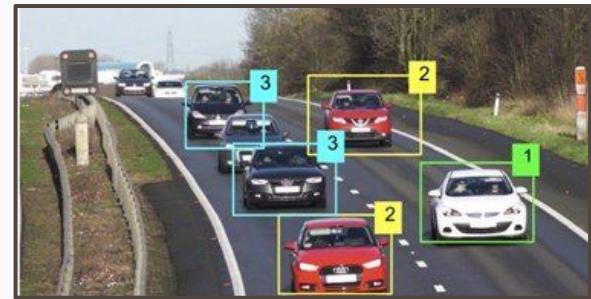
Multi-Class Detection



Segmentation



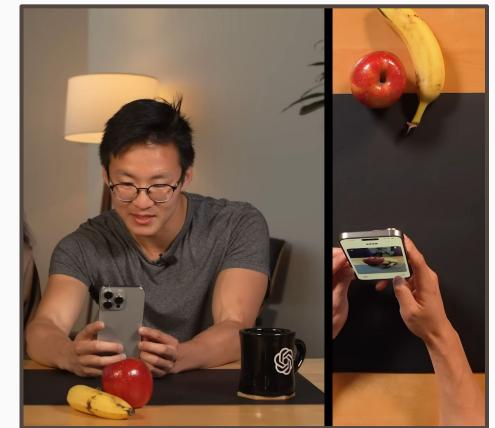
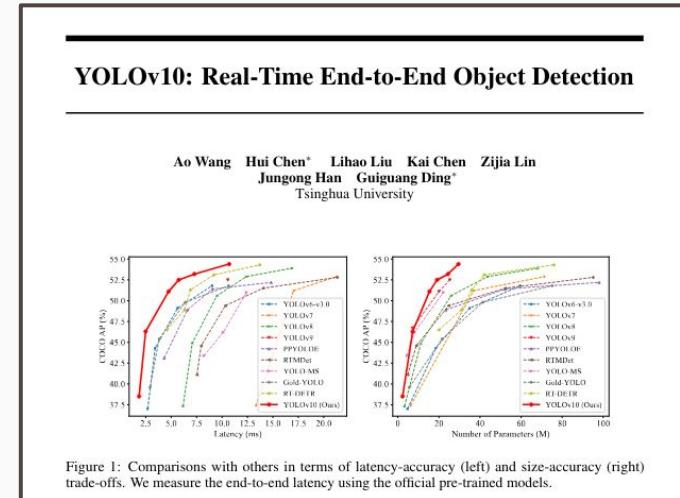
Object Tracking



Conclusiones

Avances tecnológicos

- Cada vez hay **más foco en la entrada visual** y en la reacción instantánea en **tiempo real**
- Hay constantes avances en los **algoritmos de detección de objetos**





Conclusiones

Computer Vision en la Actualidad

- La **experiencia de desarrollo** es de alta calidad
- Hay posibilidades de desarrollo a tan **alto o bajo nivel** como uno quiera
- Se sostiene considerablemente sobre **software libre** y la **investigación continua**
- Para ser competitivo se requiere de **hardware de última generación** y recursos significativos



¿Preguntas?

