

Introducción Práctica a la Inteligencia Artificial y Visión Computacional para Aplicaciones Industriales

PERUMEC 2026

Dr. Edison Moreno-Cardenas

B.Sc. Geyson E. Guerrero-Laurent



UNSAAC
Universidad Nacional de
San Antonio Abad del Cusco



EM Research
& Tech



PERUMEC

Índice

- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV
- 4 Modelos Modernos de Detección de Objetos
- 5 Bibliografía

Sobre el ponente

Edison Moreno-Cardenas

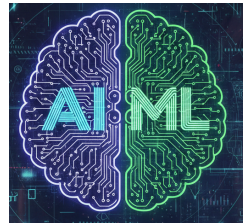
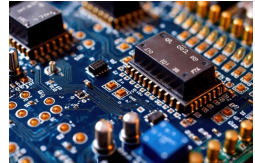
- Doctor en Ingeniería de las Telecomunicaciones
Universitat Politècnica de València (España)
- Máster en Dirección de Empresas (MBA)
Universitat de València (España)
- **Experiencia internacional**
 - España (5 años)
 - Francia (3 meses)
- Investigador en proyectos europeos de I+D sobre redes 5G y 6G
 - **ATLAS** (Dynamic Network Slicing in 5G Radio Access)
 - Radio Access Network Evolution Towards 6G
- Colaborador en el desarrollo del workshop



Sobre el asistente

Geyson E. Guerrero-Laurente

- Estudiante de Ingeniería Electrónica
Universidad Nacional de San Antonio Abad del Cusco (Perú)
- Investigador asociado en [EM Research & Tech](#)
- Áreas de investigación:
 - Sistemas de control automático
 - Inteligencia Artificial y Machine Learning
 - Procesamiento digital de imágenes
 - Sistemas electrónicos
- Apoyo en el desarrollo práctico del workshop



- 1 **Introducción**
 - La Inteligencia Artificial en la Industria
 - ¿Qué es la IA?
 - Evolución de la IA
 - Aplicaciones
 - Objetivos del Workshop

2 Fundamentos de IA

3 Fundamentos de CV

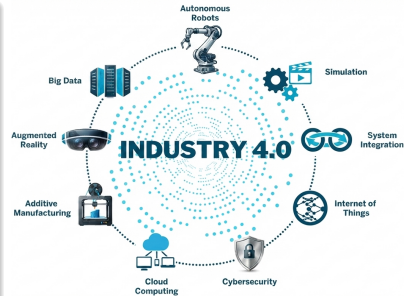
4 Modelos Modernos de Detección de Objetos

5 Bibliografía

La Inteligencia Artificial (IA) transforma la industria

¿Qué está ocurriendo?

- La **Inteligencia Artificial (IA)** está revolucionando la forma en que las industrias producen, inspeccionan y toman decisiones.
- Gracias al Aprendizaje Automático (**Machine Learning, ML**) y a la Visión Computacional (**Computer Vision, CV**), hoy es posible automatizar tareas que antes requerían intervención humana.
- Estas tecnologías constituyen uno de los pilares de la Industria 4.0.



Idea principal

La IA permite que las máquinas perciban su entorno, aprendan de los datos y apoyen la toma de decisiones en tiempo real.

1 Introducción

- La Inteligencia Artificial en la Industria
- ¿Qué es la IA?
- Evolución de la IA
- Aplicaciones
- Objetivos del Workshop

2 Fundamentos de IA

3 Fundamentos de CV

4 Modelos Modernos de Detección de Objetos

5 Bibliografía

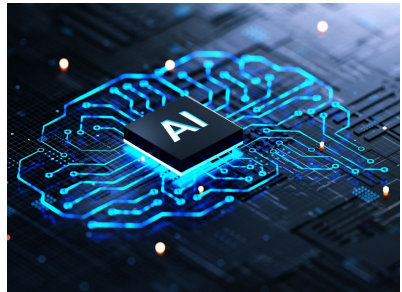
¿Qué es la IA?

Definición

La IA es una rama de la informática que desarrolla sistemas capaces de realizar tareas que normalmente requieren inteligencia humana.

Estas tareas incluyen:

- Aprendizaje.
- Reconocimiento de patrones.
- Toma de decisiones.
- Resolución de problemas.



1 Introducción

- La Inteligencia Artificial en la Industria
- ¿Qué es la IA?
- **Evolución de la IA**
- Aplicaciones
- Objetivos del Workshop

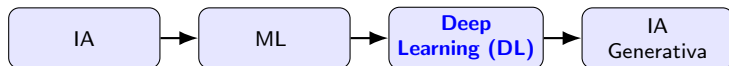
2 Fundamentos de IA

3 Fundamentos de CV

4 Modelos Modernos de Detección de Objetos

5 Bibliografía

Evolución de la IA



Idea clave

- La IA es el campo general.
- ML permite que los modelos aprendan a partir de datos.
- Aprendizaje Profundo (**Deep Learning, DL**) utiliza redes neuronales profundas.
- La IA Generativa crea contenido nuevo como texto, imágenes y código.

1 Introducción

- La Inteligencia Artificial en la Industria
- ¿Qué es la IA?
- Evolución de la IA
- **Aplicaciones**
- Objetivos del Workshop

2 Fundamentos de IA

3 Fundamentos de CV

4 Modelos Modernos de Detección de Objetos

5 Bibliografía

Aplicaciones de la IA en la Industria 4.0

Sectores

- Manufactura
- Minería
- Agricultura
- Transporte y Energía
- Telecomunicaciones
- Salud



Ejemplos

- Inspección automática.
- Detección de defectos.
- Conteo de objetos.
- Vehículos autónomos.
- Robots colaborativos.

1 Introducción

- La Inteligencia Artificial en la Industria
- ¿Qué es la IA?
- Evolución de la IA
- Aplicaciones
- **Objetivos del Workshop**

2 Fundamentos de IA

3 Fundamentos de CV

4 Modelos Modernos de Detección de Objetos

5 Bibliografía

Objetivos del Workshop

Objetivo General

Introducir a los participantes en el uso práctico de herramientas modernas de IA y CV para desarrollar aplicaciones básicas de detección de objetos basadas en modelos de DL.

Objetivos Específicos

1

Comprender los conceptos fundamentales de IA y CV.

2

Conocer el flujo de trabajo de un sistema de detección de objetos.

3

Implementar un ejemplo práctico utilizando Google Colab y YOLO.

4

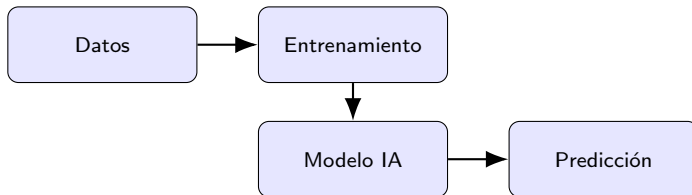
Identificar aplicaciones reales en entornos industriales.

- 1 Introducción
- 2 Fundamentos de IA
 - ¿Cómo aprende una computadora?
 - Programación Tradicional vs ML
 - Tipos de ML
 - DL
 - ¿Por qué DL revolucionó la CV?
- 3 Fundamentos de CV
- 4 Modelos Modernos de Detección de Objetos
- 5 Bibliografía

¿Cómo aprende una computadora?

Aprender significa encontrar patrones

A diferencia de la programación tradicional, en la que el desarrollador define explícitamente todas las reglas, en ML los algoritmos aprenden automáticamente dichas reglas a partir de ejemplos.



Idea clave

Los datos son el elemento más importante del ML.

1 Introducción

2 Fundamentos de IA

- ¿Cómo aprende una computadora?
- Programación Tradicional vs ML
- Tipos de ML
- DL
- ¿Por qué DL revolucionó la CV?

3 Fundamentos de CV

4 Modelos Modernos de Detección de Objetos

5 Bibliografía

Programación Tradicional vs ML

Programación Tradicional

```
key_press(self, symbol, modifiers):
    Delegate key_press()
    self.context_index == -1:
    if symbol == key.UP and not self.active_index == 8:
        self.menu_labels[self.active_index].color = [255, 255, 255]
        self.mags.dt = self.get_act_color_mag()
    elif symbol == key.DOWN and not self.active_index == 3:
        self.menu_labels[self.active_index].color = [255, 255, 255]
        self.mags.dt = self.get_act_color_mag()
    elif symbol == key.ENTER:
        if self.active_index == 3:
            pygame.app.exit()
        else:
            self.context_index = self.active_index
    elif symbol == key.ESCAPE:
        if self.context_index == -1:
            pygame.app.exit()
        else:
            self.context_index = -1
    if self.context_index == 1:
        if symbol == key.ESCAPE:
            self.context_index = -1
        else:
            self.cur_game.on_key_press(symbol, modifiers)
    se:
        if symbol == key.ESCAPE:
            self.context_index = -1
```

Datos + Reglas $\Rightarrow$ Resultado

ML



Datos + Resultados $\Rightarrow$ Modelo

Conclusión

En ML, el modelo aprende automáticamente las reglas a partir de los datos.

1 Introducción

2 Fundamentos de IA

- ¿Cómo aprende una computadora?
- Programación Tradicional vs ML
- **Tipos de ML**
- DL
- ¿Por qué DL revolucionó la CV?

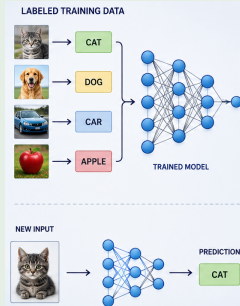
3 Fundamentos de CV

4 Modelos Modernos de Detección de Objetos

5 Bibliografía

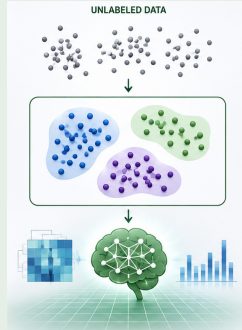
Principales tipos de ML

Supervisado



Aprende utilizando datos etiquetados.

No Supervisado



Descubre patrones sin etiquetas.

Aprendizaje por Refuerzo



Aprende mediante recompensas.

1 Introducción

2 Fundamentos de IA

- ¿Cómo aprende una computadora?
- Programación Tradicional vs ML
- Tipos de ML
- **DL**
- ¿Por qué DL revolucionó la CV?

3 Fundamentos de CV

4 Modelos Modernos de Detección de Objetos

5 Bibliografía

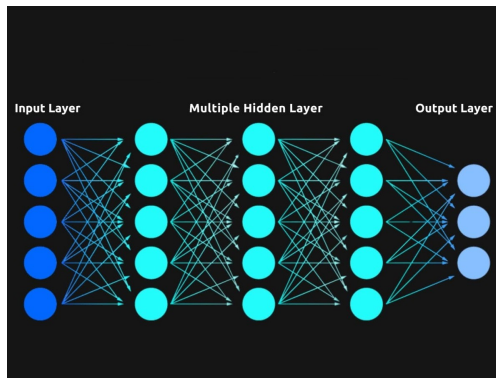
¿Qué es DL?

DL

DL es una rama del ML basada en Redes Neuronales Profundas (**Deep Neural Networks, DNN**) capaces de aprender automáticamente características complejas a partir de grandes volúmenes de datos.

Aplicaciones:

- CV
- Reconocimiento de voz
- Traducción automática
- IA Generativa



1 Introducción

2 Fundamentos de IA

- ¿Cómo aprende una computadora?
- Programación Tradicional vs ML
- Tipos de ML
- DL
- ¿Por qué DL revolucionó la CV?

3 Fundamentos de CV

4 Modelos Modernos de Detección de Objetos

5 Bibliografía

¿Por qué DL revolucionó la CV?

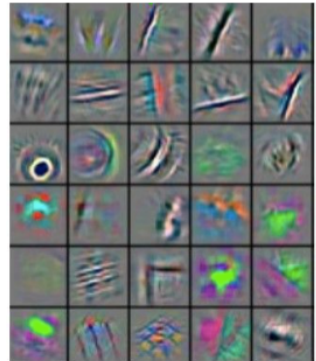
Antes

Los sistemas tradicionales requerían diseñar manualmente las características de las imágenes:

- Bordes
- Esquinas
- Texturas
- Colores

Ahora

Las redes neuronales aprenden automáticamente estas características directamente desde los datos.



Idea principal

Gracias al DL fue posible desarrollar detectores modernos como YOLO, capaces de trabajar en tiempo real.

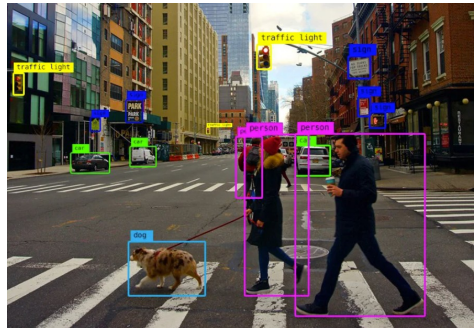
- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV**
 - **¿Qué es CV?**
 - Principales Tareas
 - Clasificación
 - Detección de Objetos
 - Segmentación de Imágenes y de objetos
- 4 Modelos Modernos de Detección de Objetos
- 5 Bibliografía

¿Qué es CV?

Definición

CV es una rama de la IA que permite a las computadoras interpretar y comprender información visual proveniente de imágenes o videos.

El objetivo de CV es imitar la capacidad humana para observar, reconocer y analizar objetos dentro de una escena.



Idea principal

Las cámaras capturan imágenes; la IA les da significado.

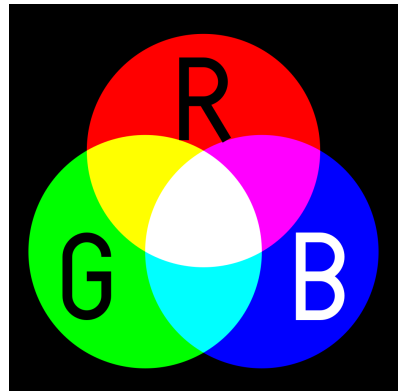
¿Cómo representa una computadora una imagen?

Imagen Digital

Una imagen está formada por millones de pequeños elementos llamados **pixeles**. Cada pixel almacena información sobre el color mediante tres canales:

- Rojo (R)
- Verde (G)
- Azul (B)

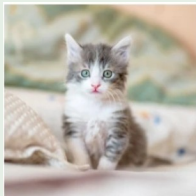
La computadora interpreta una imagen como una matriz numérica.



- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV**
 - ¿Qué es CV?
 - **Principales Tareas**
 - Clasificación
 - Detección de Objetos
 - Segmentación de Imágenes y de objetos
- 4 Modelos Modernos de Detección de Objetos
- 5 Bibliografía

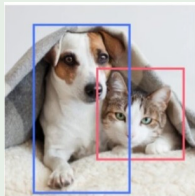
Principales tareas de CV

Clasificación



Una imagen
↓
Una etiqueta

Detección



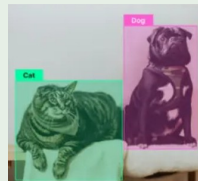
Objeto
+
Ubicación

Segmentación



Cada pixel
↓
Una clase

Tracking



Seguimiento
en video

- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV**
 - ¿Qué es CV?
 - Principales Tareas
 - Clasificación**
 - Detección de Objetos
 - Segmentación de Imágenes y de objetos
- 4 Modelos Modernos de Detección de Objetos
- 5 Bibliografía

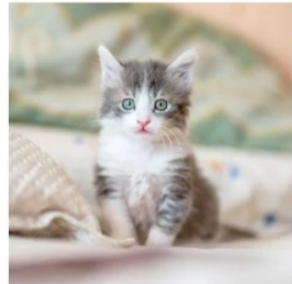
Clasificación de Imágenes

¿Qué hace?

El modelo analiza una imagen completa y asigna una única categoría.

Ejemplos:

- Perro
- Gato
- Automóvil
- Persona



Salida

Gato (99.4%)

- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV**
 - ¿Qué es CV?
 - Principales Tareas
 - Clasificación
 - Detección de Objetos**
 - Segmentación de Imágenes y de objetos
- 4 Modelos Modernos de Detección de Objetos
- 5 Bibliografía

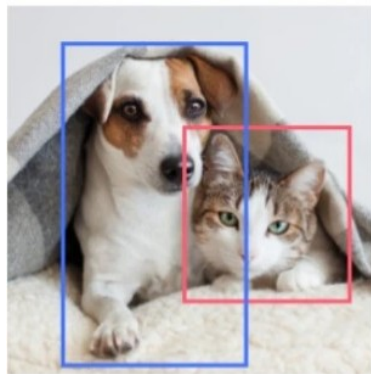
Detección de Objetos

¿Qué hace un detector?

Además de identificar el objeto, estima su ubicación dentro de la imagen mediante un rectángulo delimitador (Bounding Box).

El resultado incluye:

- Clase
- Coordenadas
- Nivel de confianza



- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV**
 - ¿Qué es CV?
 - Principales Tareas
 - Clasificación
 - Detección de Objetos
 - **Segmentación de Imágenes y de objetos**
- 4 Modelos Modernos de Detección de Objetos
- 5 Bibliografía

Segmentación de Imágenes

¿Qué hace?

La segmentación, en lugar de dibujar únicamente un rectángulo, identifica cada píxel perteneciente al objeto.

La segmentación se utiliza en:

- Medicina
- Agricultura
- Vehículos autónomos
- Inspección industrial

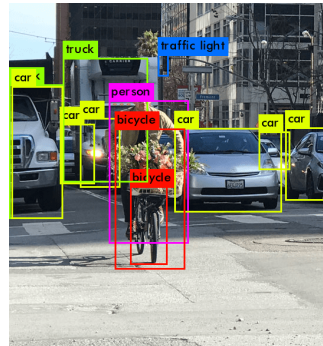


Seguimiento de Objetos (Tracking)

Tracking

El seguimiento de objetos consiste en mantener la identidad de cada objeto a lo largo del tiempo mientras aparece en una secuencia de video. Aplicaciones:

- Tráfico
- Videovigilancia
- Robots móviles
- Drones
- Logística



Importante

YOLO detecta objetos; para realizar seguimiento normalmente se combina con algoritmos como ByteTrack o DeepSORT.

Seguimiento de Objetos (Tracking)

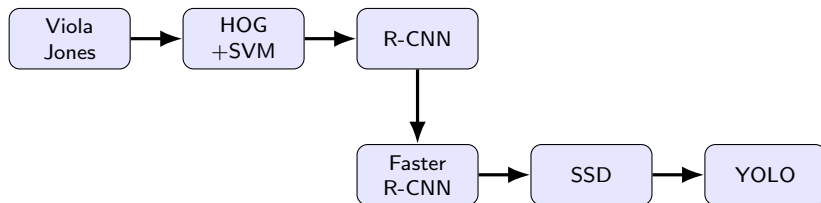
En este punto haría una pequeña demostración de 2–3 minutos antes de explicar YOLO. En lugar de pasar directamente a la teoría de los detectores, mostraría un video corto (30–45 segundos) donde se vea un sistema detectando personas, vehículos y cascos de seguridad en tiempo real. Eso crea un momento de "¡ahora entiendo para qué sirve!" y hace que los asistentes lleguen mucho más motivados a la sección siguiente, donde explicarás YOLO, OpenCV, Google Colab y el laboratorio práctico. Esa transición suele funcionar muy bien en talleres de este tipo.

Importante

Crear video

- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV
- 4 Modelos Modernos de Detección de Objetos**
 - Evolución
 - Ventajas
 - Aplicaciones Industriales
 - Herramientas
 - Caso Práctico
- 5 Bibliografía

Evolución de los Detectores de Objetos



Idea principal

Durante los últimos años se han desarrollado numerosos modelos para la detección de objetos. Entre ellos, YOLO destaca por su velocidad, precisión y capacidad para realizar inferencia en tiempo real.

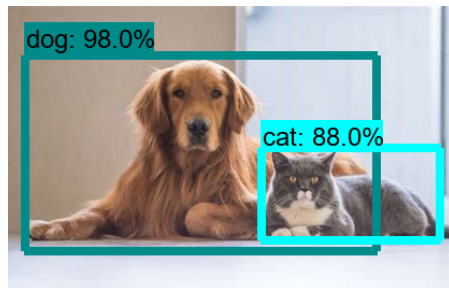
¿Qué es YOLO?

YOLO

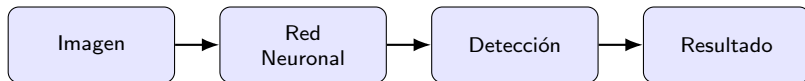
YOLO (*You Only Look Once*) es una familia de modelos de DL diseñada para detectar múltiples objetos en una imagen mediante una única pasada por la red neuronal.

Características:

- Alta velocidad.
- Alta precisión.
- Tiempo real.
- Código abierto.



Flujo General de un Detector de Objetos



Salida

El modelo devuelve para cada objeto:

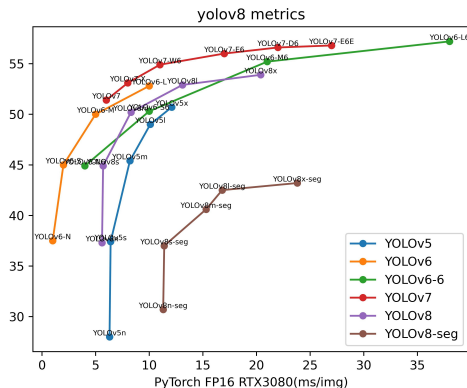
- Clase.
- Bounding Box.
- Nivel de confianza.

- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV
- 4 Modelos Modernos de Detección de Objetos**
 - Evolución
 - Ventajas**
 - Aplicaciones Industriales
 - Herramientas
 - Caso Práctico
- 5 Bibliografía

¿Por qué utilizaremos YOLO?

Ventajas

- Inferencia en tiempo real.
- Fácil de utilizar.
- Excelente precisión.
- Compatible con Python.
- Amplia documentación.
- Comunidad muy activa.



- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV
- 4 Modelos Modernos de Detección de Objetos**
 - Evolución
 - Ventajas
 - Aplicaciones Industriales**
 - Herramientas
 - Caso Práctico
- 5 Bibliografía

Aplicaciones Industriales

Casos de uso

- Control de calidad.
- Detección de defectos.
- Seguridad industrial.
- Conteo automático.
- Inventario.
- Agricultura inteligente.
- Minería.
- Drones.

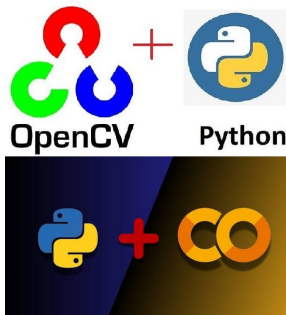


- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV
- 4 Modelos Modernos de Detección de Objetos**
 - Evolución
 - Ventajas
 - Aplicaciones Industriales
 - Herramientas**
 - Caso Práctico
- 5 Bibliografía

Herramientas que utilizaremos

Software

- Python
- Google Colab
- Ultralytics
- OpenCV
- YOLO11



Siguiente paso

A continuación construiremos paso a paso nuestro primer sistema de detección automática de objetos utilizando Google Colab y YOLO.

- 1 Introducción
- 2 Fundamentos de IA
- 3 Fundamentos de CV
- 4 Modelos Modernos de Detección de Objetos**
 - Evolución
 - Ventajas
 - Aplicaciones Industriales
 - Herramientas
 - **Caso Práctico**
- 5 Bibliografía

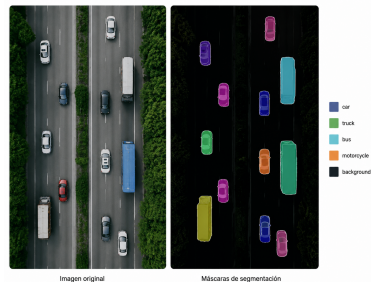
Caso Práctico: Segmentación con YOLO11

Objetivo

Entrenar un modelo de segmentación YOLO11 sobre un dataset propio, etiquetado con Label Studio, para resolver un problema real de clasificación por regiones (pixel a pixel).

Dataset

- Imágenes propias, anotadas en Label Studio
- Formato: YOLO segmentación (polígonos)
- División: **train** / **val** / **test**



Ejemplo de segmentación con máscaras

Fuentes recomendadas

- **Roboflow Universe:**
datasets públicos ya anotados
- **Kaggle:** datasets y competencias
- **Google Images:**
búsqueda + descarga masiva controlada
- **Videos propios**
(YouTube o celular):
extracción de frames
- **Cámara/planta:**
captura directa del proceso real

datos para entrenar tu modelo?

Extraer frames de un video (Python)

```
import cv2, os

video = "video.mp4"
salida = "frames"
os.makedirs(salida, exist_ok=True)

fps_deseado = 2
cap = cv2.VideoCapture(video)
fps_video = cap.get(cv2.CAP_PROP_FPS)
paso = int(fps_video / fps_deseado)
```

Idea clave

Variedad de ángulos, iluminación y fondos $\Rightarrow$ mejor generalización del modelo.

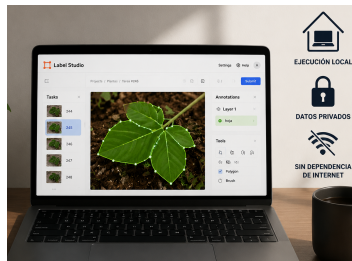
¿Por qué Label Studio?

Ejecución 100% local

- Se instala y corre en tu propia máquina (no depende de la nube)
- Tus imágenes **nunca salen** de tu equipo
- Ideal cuando el dataset es sensible o confidencial (procesos industriales, datos de planta, propiedad intelectual)

Además

- Código abierto (versión Community, gratuita)
- No depende de conexión a internet estable
- Exporta directo a formato YOLO



Comparación

Herramientas de anotación en la nube → tus datos pasan por servidores de terceros.
Label Studio local → el dataset no compromete su confidencialidad.

Label Studio: Más Allá de la Visión Computacional

Categorías de proyecto

- Computer Vision
- Natural Language Processing
- Audio/Speech Processing
- Conversational AI / Chat
- Ranking & Scoring
- Structured Data Parsing
- Time Series Analysis
- Videos
- Generative AI

Plantillas en Computer Vision

- **Semantic Segmentation with Polygons** ← la usaremos hoy
- Semantic Segmentation with Masks
- Object Detection with Bounding Boxes
- Image Classification
- Image Captioning
- Keypoint Labeling (Enterprise)
- OCR Labeling for PDFs (Enterprise)

Idea clave

Es una sola herramienta para casi cualquier tipo de dato: imagen, texto, audio, video o series de tiempo.

Estructura de Carpetas y dataset.yaml

Organización

```
dataset/  
  images/  
    train/  
    val/  
  labels/  
    train/  
    val/
```

dataset.yaml

```
path: /ruta/a/dataset  
train: images/train  
val: images/val  
  
nc: 4  
names:  
  - objeto_conforme  
  - objeto_defecto_leve  
  - objeto_defecto_grave  
  - fondo
```

Nota

nc y names se generan automáticamente con `yaml.dump()` en Python. Las clases mostradas son de ejemplo — se ajustan según el caso que trabajemos en vivo.

Verificación del Dataset

Código de verificación

```
for split in ['train', 'val']:
    imgs = [f for f in os.listdir(f'images/{split}')
            if f.lower().endswith(('.jpg', '.png'))]
    lbls = [f for f in os.listdir(f'labels/{split}')
           if f.endswith('.txt')]
    print(f'{split}: {len(imgs)} imagenes, {len(lbls)} labels')
```

Qué revisa

Imágenes sin label asociado

Labels huérfanos (sin imagen)

Archivos corruptos o nombres inconsistentes

En el taller

Corremos esta celda en vivo sobre el dataset real y vemos el resultado juntos.

Entrenamiento: Parámetros Clave

Configuración base

```
model = YOLO("yolo11m-seg.pt")

model.train(
    data=yaml_path,
    epochs=180, imgsz=800,
    batch=4, patience=50,
    optimizer="AdamW",
    lr0=0.0008, lrf=0.01,
    warmup_epochs=5,
    mosaic=0.5, flipplr=0.5,
    degrees=5, scale=0.3,
    device=0, workers=4,
)
```

¿Qué significa cada uno?

- `imgsz`: resolución de entrada
- `batch`: imágenes por lote (limitado por VRAM)
- `patience`: épocas sin mejora antes de parar
- `lr0/lrf`: velocidad de aprendizaje
- `mosaic/flipplr/degrees/scale`: aumento de datos

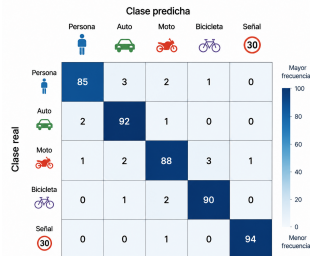
Cómo Leer la Matriz de Confusión

¿Qué muestra?

- Filas = clase real
- Columnas = clase predicha
- Diagonal = aciertos del modelo
- Fuera de la diagonal = confusiones entre clases

En el taller

Vamos a generar y leer la matriz de confusión de nuestro propio entrenamiento, en vivo.



Estructura general de una matriz de confusión

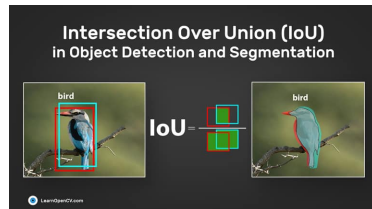
mAP50 vs mAP50-95: ¿Qué Significan?

mAP50

$\text{IoU} = 0.5$ (umbral relajado). Mide si el modelo detecta el objeto, sin exigir un contorno perfecto.

mAP50-95

Promedio de mAP entre IoU 0.5 y 0.95. Exige precisión de localización/segmentación.



Visualización del IoU (Intersection over Union)

Regla práctica

mAP50 alto + mAP50-95 bajo $\Rightarrow$ detecta bien, pero el contorno se puede afinar. Los valores de nuestro modelo los vemos en vivo.

Inferencia: Probar el Modelo Entrenado

Código de inferencia

```
from ultralytics import YOLO

model = YOLO("best.pt")

resultados = model.predict(
    source="imagen.jpg",
    conf=0.5,
    save=True
)

for r in resultados:
    print(r.boxes.cls) # clases
                        detectadas
    print(r.boxes.conf) # confianza
```

¿Qué devuelve?

- Máscara de segmentación por objeto
- Clase predicha
- Nivel de confianza (0 a 1)
- Imagen anotada guardada automáticamente

Errores Comunes al Entrenar (y cómo resolverlos)

Problema

- CUDA out of memory
- `FileNotFoundError` en `dataset.yaml`
- Labels vacíos o con clase inexistente
- mAP se queda muy bajo desde el inicio

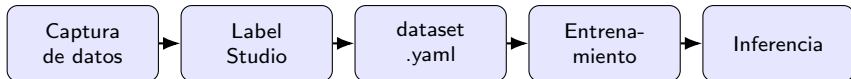
Solución rápida

- Reducir batch o `imgsz`
- Usar ruta **absoluta** en path
- Revisar que no coincida con el número real de clases
- Verificar balance de clases en el dataset

Idea clave

La mayoría de errores en YOLO son de **rutas o de formato del dataset**, no del modelo en sí.

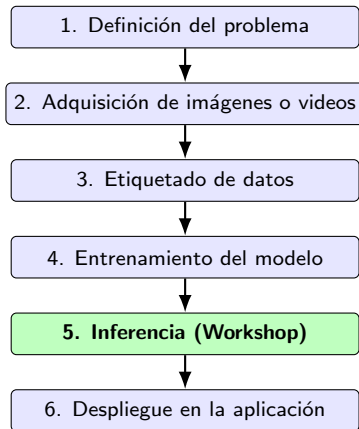
Resumen: De la Idea al Modelo Funcionando



Lo que hicimos hoy

Recorrimos las 5 etapas con nuestro propio dataset — este mismo flujo aplica a cualquier problema de segmentación industrial.

Probar el Modelo Entrenado



Objetivo del Workshop

Durante esta sesión trabajaremos únicamente en la etapa de **inferencia**, utilizando un modelo YOLO previamente entrenado para detectar objetos en imágenes y videos mediante Google Colab.

Contacto

**Edison
Moreno-Cardenas**



edison.moreno@unsaac.edu.pe

**Geyson
Guerrero-Laurent**



201249@unsaac.edu.pe

EM Research & Tech

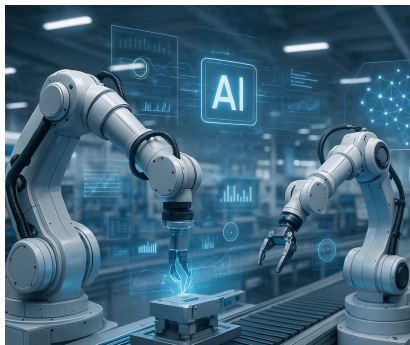


www.emresearchtech.com

¡Gracias por su atención!

Bibliografía

- Shinde, P. P., & Shah, S. (2018). A review of machine learning and deep learning applications. In 2018 Fourth international conference on computing communication control and automation (ICCUBEA) (pp. 1-6). IEEE.
- Horvitz, E. J., Breese, J. S., & Henrion, M. (1988). Decision theory in expert systems and artificial intelligence. International journal of approximate reasoning, 2(3), 247-302.
- Lee, J., Davari, H., Singh, J., & Pandhare, V. (2018). Industrial Artificial Intelligence for industry 4.0-based manufacturing systems. Manufacturing letters, 18, 20-23.



¿Preguntas?
¿Sugerencias?