Clase 132 — Localización, detección, segmentación (+ DETR, Segment Anything, YOLOv11)

Parte: 2 — Deep Learning · Fuente: Géron, cap. 14 § Object Detection y § Semantic Segmentation + papers DETR, SAM, YOLOv11. Duración estimada: 90 min.

Clase 132 — Localización, detección, segmentación (+ DETR, Segment Anything, YOLOv11)

> *Parte: 2 — Deep Learning · Fuente: Géron, cap. 14 § Object Detection y § Semantic Segmentation + papers DETR, SAM, YOLOv11. Duración estimada: 90 min.*

Objetivo

Saber detectar y segmentar objetos en imágenes — la tarea de visión más compleja y más comercial. Conocer la evolución: Faster R-CNN (two-stage, lento + preciso) → YOLO (one-stage, rápido) → DETR (Transformer, end-to-end) → YOLOv11 (estado del arte 2024) → Segment Anything (SAM/SAM 2) (foundation model para segmentación).

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Distinguir las 4 tareas: clasificación, localización (1 objeto), detección (N objetos + cajas), segmentación (pixel-wise mask).
- Usar un modelo YOLOv11 preentrenado con ultralytics: model = YOLO('yolo11n.pt'); results = model('imagen.jpg').
- Aplicar Segment Anything (SAM 2) para segmentación promptable con puntos o cajas como input.
- Reconocer cuándo elegir DETR (mejor formal, lento) vs YOLO (rápido, default industrial) vs SAM (pre-entrenado universal).
- Métricas: IoU, mAP, COCO AP@[.5:.05:.95].

Temas

- Localización vs detección vs instance segmentation vs semantic segmentation.
- IoU, mAP, COCO benchmark.
- Anchors vs anchor-free.
- One-stage (YOLO, SSD) vs two-stage (Faster R-CNN).
- Complemento moderno: DETR (Carion et al. 2020), SAM/SAM 2 (Meta 2023/2024), YOLOv11 (Ultralytics 2024).
- Pre-trained pipelines: ultralytics, transformers (DETR), segment_anything.

Versión profundizada — 2026

El tema moderno que antes vivía como complemento dentro de esta clase ahora tiene su(s) clase(s) propia(s) con patrón completo, ejercicios y homework:

- Clase 117b — Segment Anything (SAM / SAM 2)
- Clase 117c — YOLOv11 práctico: detección, segmentación, pose, tracking

Definiciones y características

- Bounding box: (x_min, y_min, x_max, y_max) o (cx, cy, w, h).
- IoU (Intersection over Union): area(intersección) / area(unión). Match si IoU > 0.5.
- mAP (mean Average Precision): promedio del AP sobre clases y umbrales IoU. COCO usa IoU 0.5 → 0.95 en pasos de 0.05.
- NMS (Non-Maximum Suppression): post-procesado que elimina cajas superpuestas dejando la de mejor score. DETR no la necesita.
- Anchor-based: pre-define cajas en cada posición y aprende offsets.
- Anchor-free: predice directamente, sin pre-definir cajas (FCOS, DETR, YOLOX+).
- Semantic vs instance segmentation: semantic = mismo color para todas las instancias de "persona"; instance = cada persona con su color.
- Promptable segmentation (SAM): dado un punto/caja/texto, devolver la máscara correspondiente.

Dataset / recursos

- COCO (detección/seg estándar): tfds.load('coco/2017').
- Pascal VOC: más chico, bueno para experimentos.
- Custom: anotaciones en YOLO format (txt por imagen) o COCO format (JSON).
- Librerías: ultralytics (pip install ultralytics), transformers, segment-anything.

Ejercicios

1. YOLO inference: cargar yolo11n.pt y detectar sobre 3 imágenes propias. Visualizar boxes + labels.
2. DETR inference: usar facebook/detr-resnet-50 desde HF. Comparar resultados con YOLO sobre las mismas imágenes.
3. SAM segmentation: dar un punto sobre un objeto en una imagen, obtener máscara. Probar con cajas.
4. mAP a mano: dado un set de predicciones y GT, implementar IoU y calcular AP@0.5 manualmente.
5. YOLO fine-tune: dataset propio (≥ 100 imágenes anotadas), model.train(data='dataset.yaml', epochs=50). Reportar mAP.

Homework verificable

Detector custom de 2-3 clases con YOLOv11:

1. Etiquetar ~100 imágenes con CVAT, LabelImg o Roboflow → YOLO format.
2. dataset.yaml con paths + clases.
3. model.train(data='dataset.yaml', epochs=100, imgsz=640).
4. Reportar mAP@0.5 en val + 3 imágenes con bounding boxes superpuestos.

Criterio de aceptación: mAP@0.5 ≥ 0.7 sobre val set; el modelo detecta correctamente las clases en las 3 imágenes de inspección visual.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
YOLO devuelve cajas raras tras fine-tuning	Pocos datos o etiquetado inconsistente. Fi
imgsz=320 para mejor velocidad pierde much	Trade-off speed/accuracy. Fix: 640 default
SAM lento sin GPU	El modelo ViT-H tiene 636M params. Fix: vi
DETR lento en training	DETR converge muy lento (500 epochs en COC

NMS no aplicado → muchas cajas superpuesta	YOLO lo aplica internamente (conf=0.25, io

Preguntas frecuentes

¿YOLO o DETR para producción?

YOLO es default industrial (rápido, bien soportado, fácil deploy). DETR es preferido en investigación.

¿SAM puede reemplazar a un segmentador clásico?

Para tareas donde quieras "segmentar lo que el usuario apunta", sí. Para clases específicas sin intervención humana, fine-tunear un segmentador es más eficiente.

¿Open-vocabulary detection?

Detectores que aceptan clases nuevas vía texto sin reentrenar (CLIP-based). OWL-ViT, GroundingDINO. Útil cuando las clases cambian.

¿Anotación manual cuántas imágenes?

Para 2-3 clases simples con YOLO: 100-500 ya da resultados decentes. Para escenarios complejos: miles. Roboflow / CVAT facilitan el workflow.

¿Tracking de objetos en video?

YOLO + tracker (ByteTrack, BoT-SORT) integrados en ultralytics. Para video más complejo, SAM 2.

Referencias

- Géron, cap. 14 — Object Detection y Semantic Segmentation.
- Carion et al. (2020), End-to-End Object Detection with Transformers (DETR), ECCV.
- Kirillov et al. (2023), Segment Anything, ICCV.
- Ravi et al. (2024), SAM 2: Segment Anything in Images and Videos, Meta.
- Ultralytics YOLOv11 docs.
- Segment Anything project.

Siguiente clase

Clase 133 — Segment Anything (SAM / SAM 2): foundation model para segmentación

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb