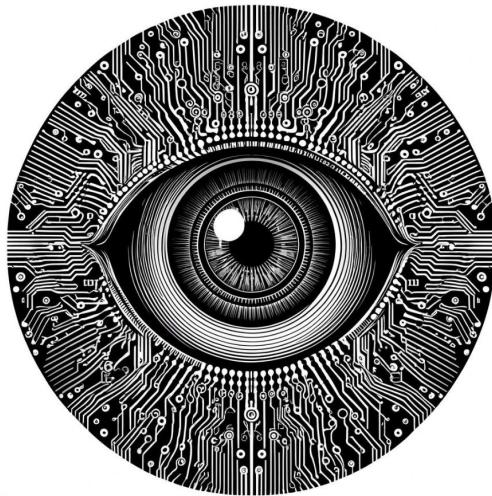


Workshop 11 - Deep Learning Approaches to Object Detection and Image Segmentation



Antonio Rueda-Toicen

About me

- AI Researcher at [Hasso Plattner Institute](#), AI Engineer & DevRel for [Voxel51](#)
- Organizer of the [Berlin Computer Vision Group](#)
- Instructor at [Nvidia's Deep Learning Institute](#) and Berlin's [Data Science Retreat](#)
- Made a [MOOC for OpenHPI](#)



[LinkedIn](#)

How to use our Discord channel during the workshop

- Our channel is **#practical-computer-vision-workshops**. Please ask all questions there instead of the Zoom chat. Through Discord we can have better and more detailed discussions.
- **Step 1 - Use the Discord invite on the Voxel51 website**
<https://discord.com/invite/fiftyone-community>
- **Step 2 - Access channel** (use the direct link below or search)
<http://bit.ly/3YmvPXG>

Agenda

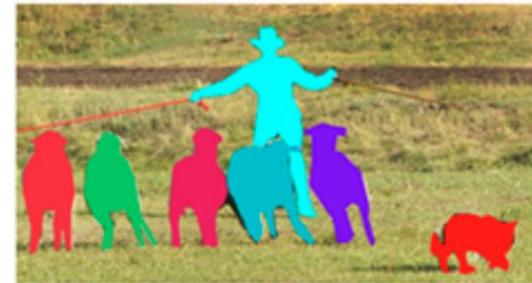
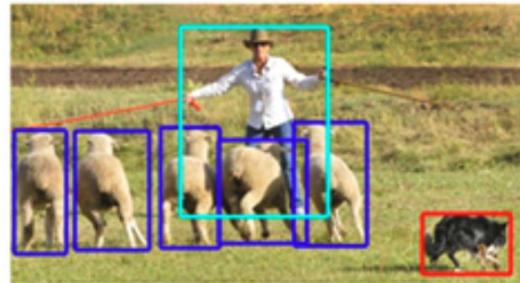
- Approaches to Object Detection
- Approaches to Image Segmentation

Notebooks

- Open-set Detection with Grounding DINO
- Object Detection with YOLOv11 on COCO's validation set
- Bike Lane Segmentation with Mask2Former
- U-net for semantic image segmentation (review)
- Instance Segmentation with YOLOE on COCO's validation set

Discriminative computer vision

- “Which label or value can I assign to this image?”



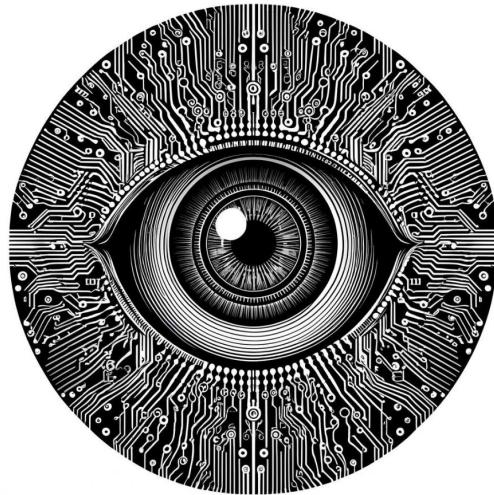
Classification

Object Detection

Instance Segmentation

Image from “Learning to Segment” by Piotr Dollar, 2017

Approaches to Object Detection



Learning goals

- Recognize object detection as a regression and classification problem
- Describe the use of anchor boxes on single shot detectors (RetinaNet, YOLOv1-v5) and two-stage detectors (Faster R-CNN)
- Gain familiarity with anchor-box-free object detection approaches (YOLOv6+, DETR, Grounding DINO)

Object detection as bounding box localization

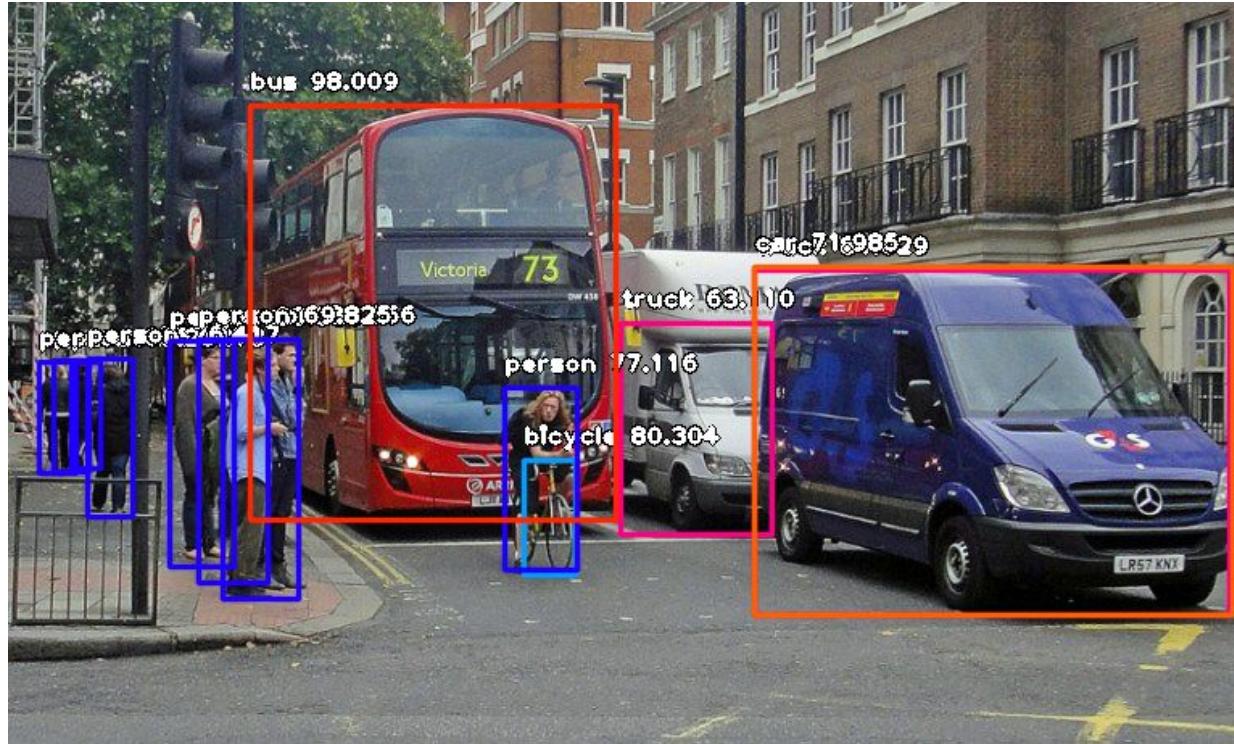
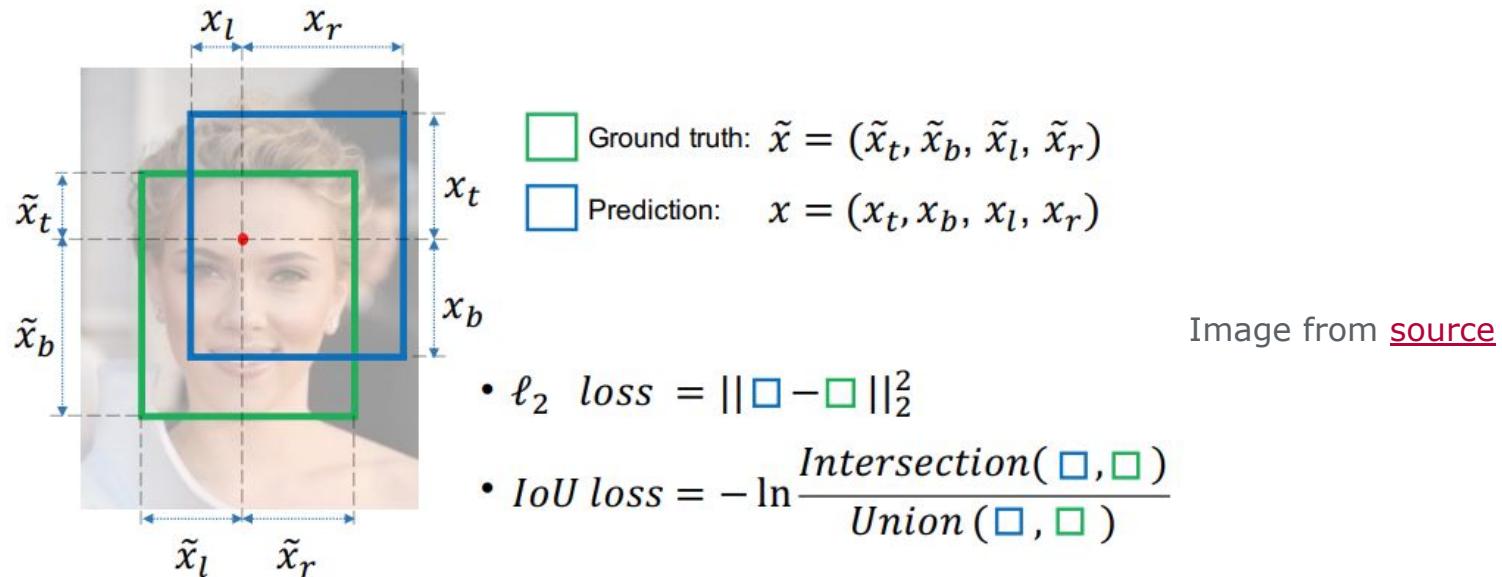


Image [source](#)

Object detection as bounding box regression and classification



'Regression' = predicting a continuous value (bounding box coordinates)

Loss functions combine classification and regression error

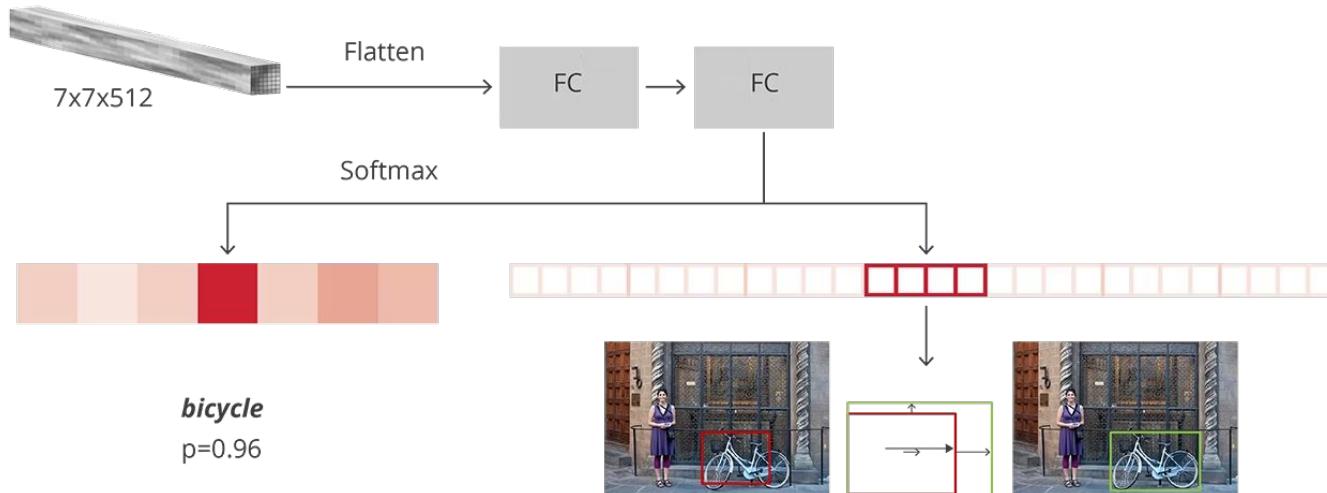
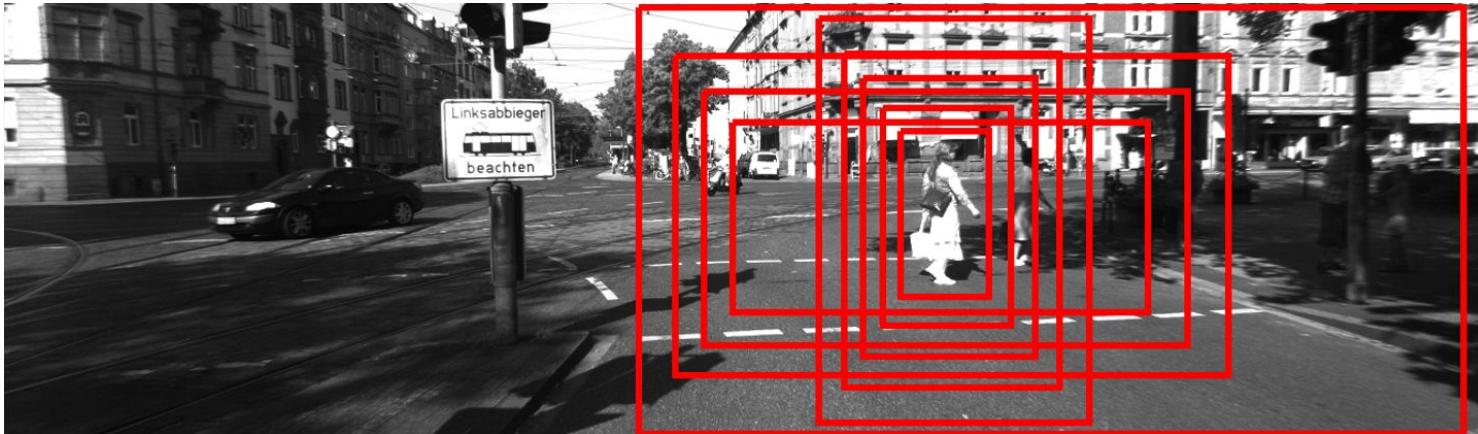


Image from [Faster R-CNN: Down the rabbit hole of modern object detection](#)

$$\mathcal{L} = - \sum_x P(x) \log(Q(x)) + \lambda \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

Partitioning an image into regions



Possible approaches:

- Anchor-boxes
- Keypoint identification

Anchor boxes - defining potential detections



Figure 14.7: **Left:** Creating anchors starts with the process of sampling the coordinates of an image every r pixels ($r = 16$ in the original Faster R-CNN implementation). **Right:** We create a total of nine anchors centered around *each* sampled (x, y) -coordinate. In this visualization, $x = 300, y = 200$ (center blue coordinate). The nine total anchors come from every combination of scale: 64×64 (red), 128×128 (green), 256×256 (blue); and aspect ratio: $1 : 1, 2 : 1, 1 : 2$.

Image from [Faster R-CNNs - PyImageSearch](#)

Anchor boxes - defining sizes and aspect ratios

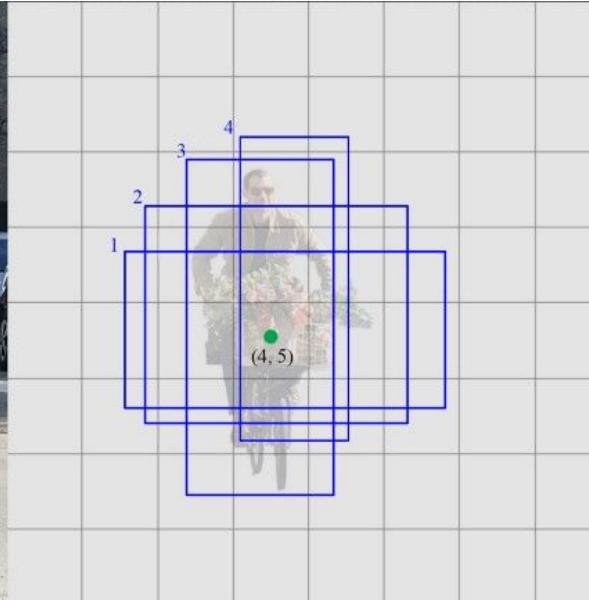


Image from [source](#)

Anchor boxes - the challenge of filtering candidates



Image from [source](#)

Single shot detectors vs two stage detectors

YOLO (original)

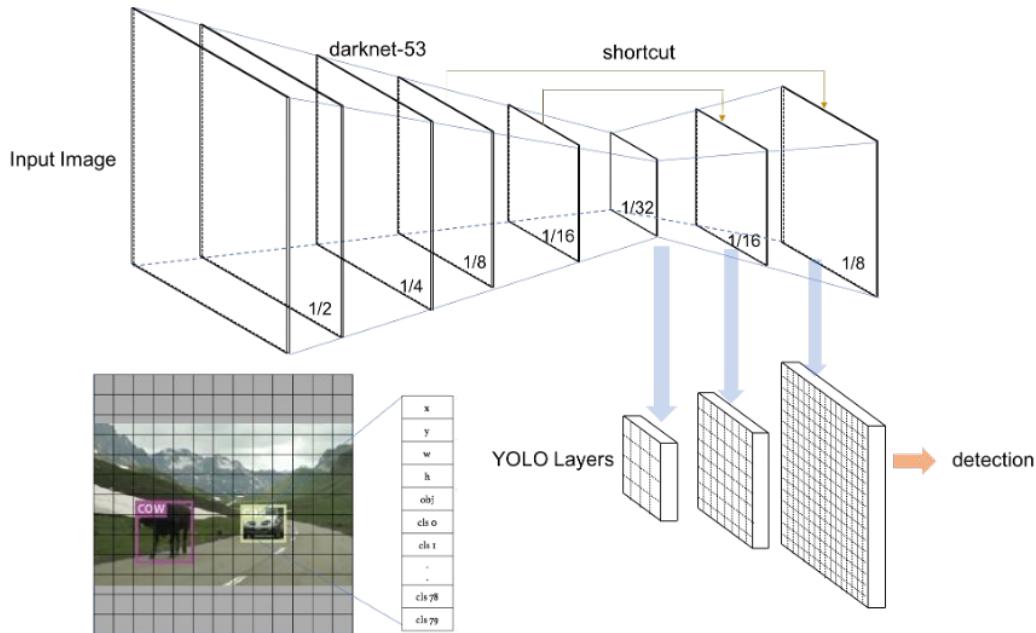


Image from [YOLO: Real-Time Object Detection](#)

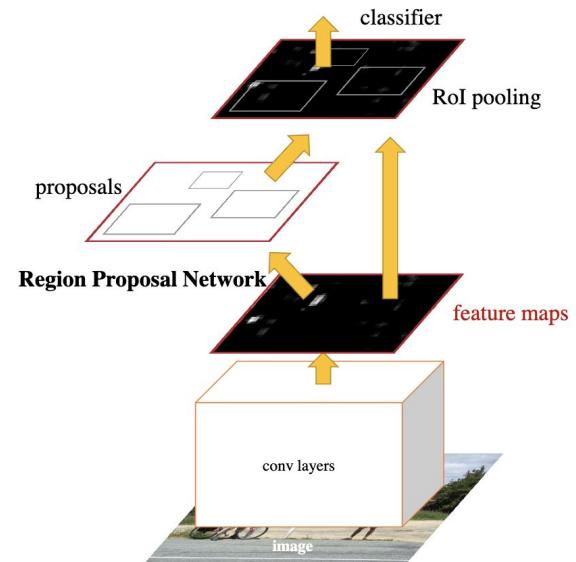
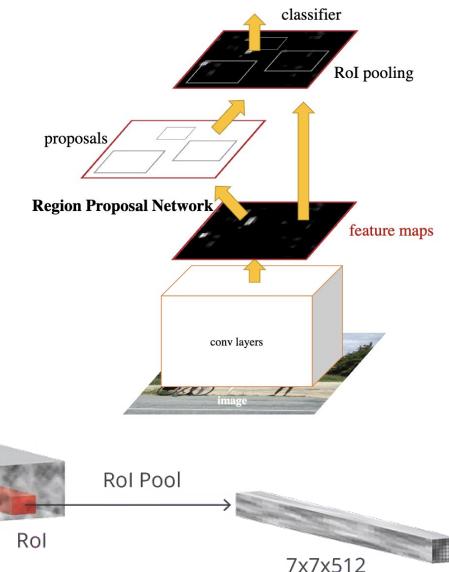
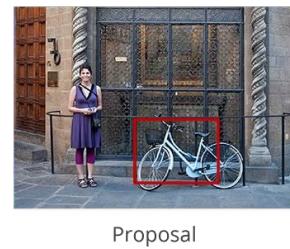
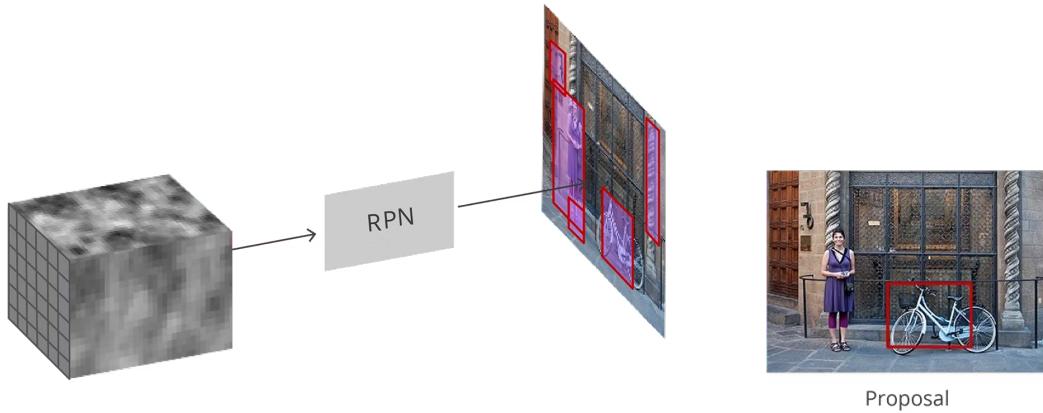


Image from [Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks](#)

Faster R-CNN's Region Proposal Network (RPN)

The RPN takes the convolutional feature map and generates unlabeled proposals over the image



Region of Interest (ROI) pooling uses the downsampled original features cropped on the proposal area to feed the classifier

Images from [Faster R-CNN: Down the rabbit hole of modern object detection](#)

Grid-based anchor-free detection (YOLOv11)

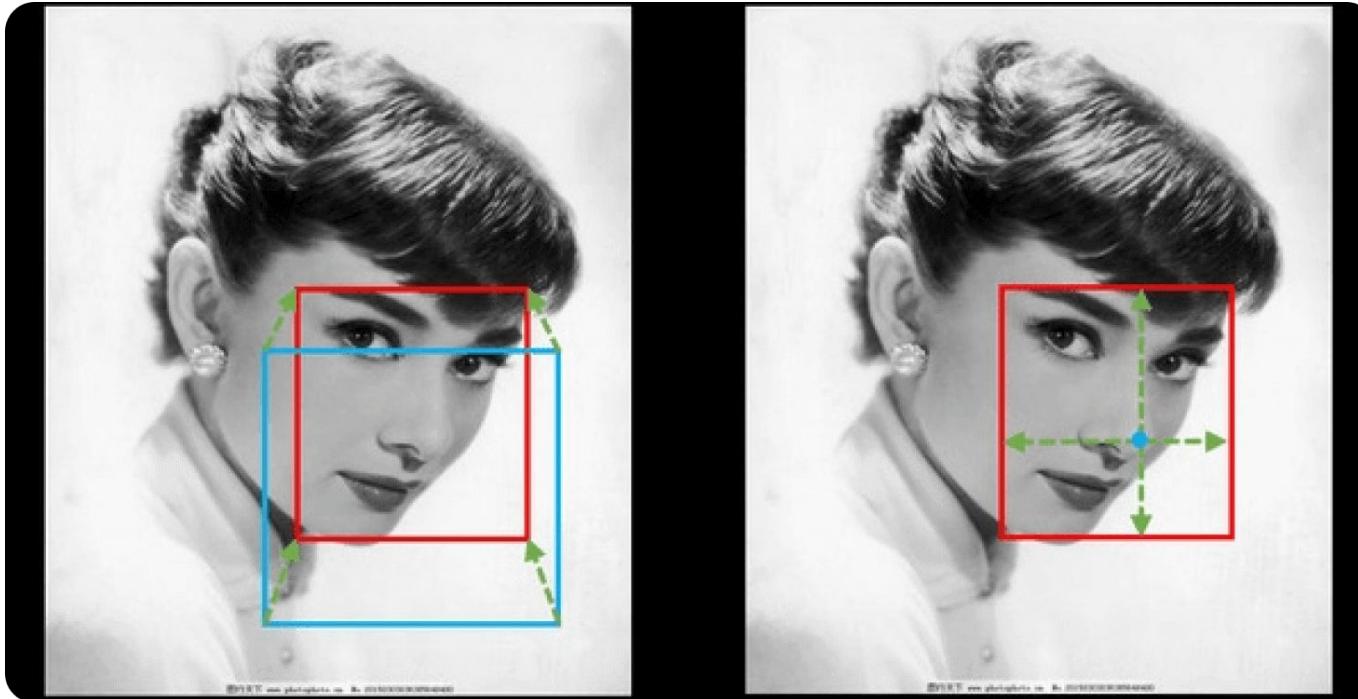


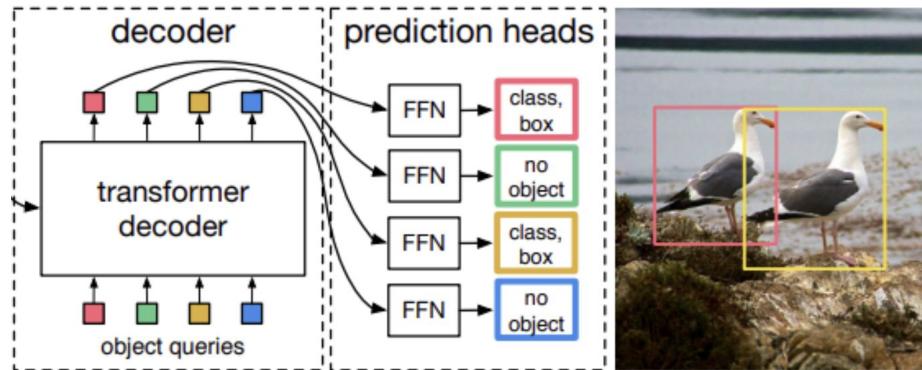
Image from the [Ultralytics blog](#)

Anchor-based vs anchor-free detection



- Require prior knowledge or aspect ratios and sizes of potential anchor boxes
 - More settings to tune
 - Remain competitive in different object scales and with partially occluded objects (Faster R-CNN excels at this)
- Fewer hyperparameters (less tuning)
 - Difficulty handling partially occluded objects

DETR's object queries replace anchor boxes



- Unlike anchor boxes: no geometric prior
- Learned embeddings with same dimension as all other embedded components of DETR
- Each object query embedding specializes on a region
- 100 by default, max number of detections
- Output class and bounding box after FFN

```
predictions = [ {"query": 1, "class": "bird", "box": [0.2, 0.3, 0.1, 0.1]},  
                {"query": 2, "class": "bird", "box": [0.5, 0.6, 0.2, 0.2]} ]
```

Language-guided detection with Grounding DINO

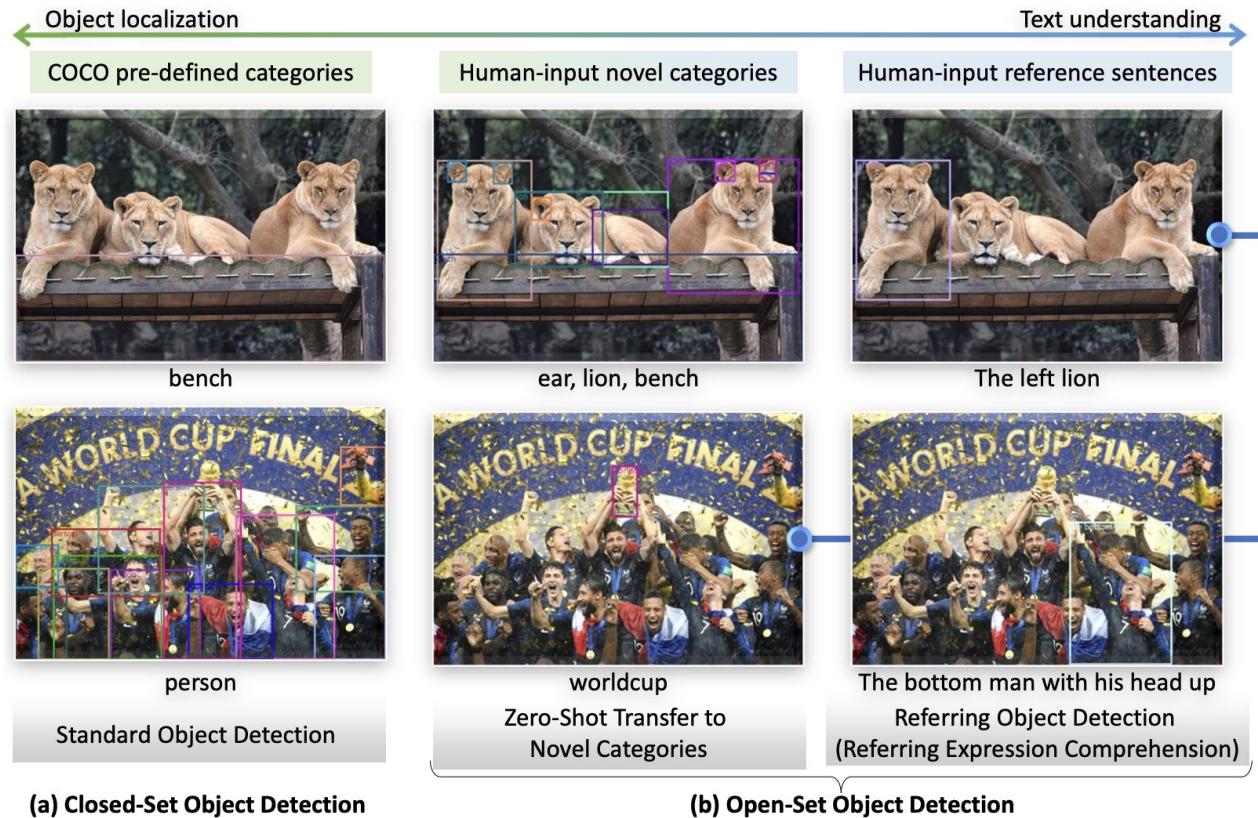
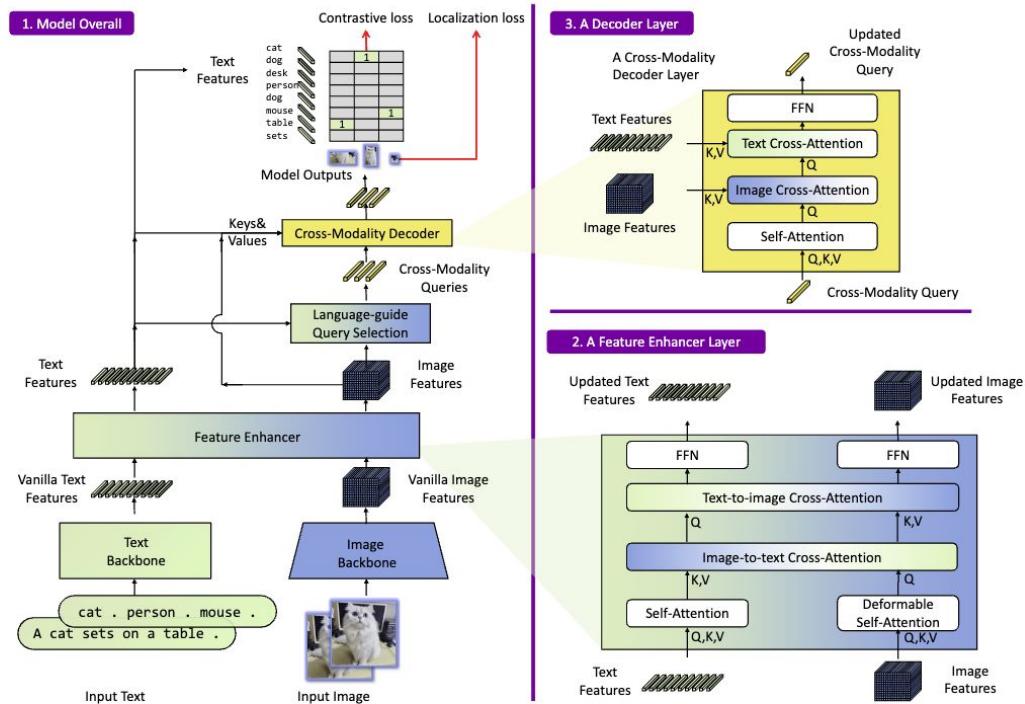


Image from [Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection](#)

Grounding DINO's architecture



- 0.8M bounding box annotations
- 4M text-image pairs from Object365 and COCO
- 3M text-image pairs from Google's CC3M dataset

Figure 3. The framework of Grounding DINO. We present the overall framework, a feature enhancer layer, and a decoder layer in block 1, block 2, and block 3, respectively.

Image from [Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection](#)

Summary

Object detection combines classification and bounding box regression

- Regression loss functions (e.g. L1, L2) quantify the positioning error of bounding boxes, cross-entropy quantifies the classification error

Anchor box-based detectors

- Anchor boxes define potential object locations, scales, and aspect ratios
- Two stage detectors use region proposal networks to filter candidate anchor boxes, and tend to have higher accuracy at higher computation cost
- The number and variety of anchor boxes influences the performance of the model

Anchor-free and grounded detection

- DETR removes anchor boxes by using object queries and the attention mechanism
- Grounding DINO enables language-guided detection, with an accuracy tradeoff vs fully supervised models

References and further reading

Focal Loss for Dense Object Detection

- <https://arxiv.org/abs/1708.02002>

End-to-End Object Detection with Transformers

- <https://arxiv.org/abs/2005.12872>
- <https://www.youtube.com/watch?v=utxbUlo9CyY>

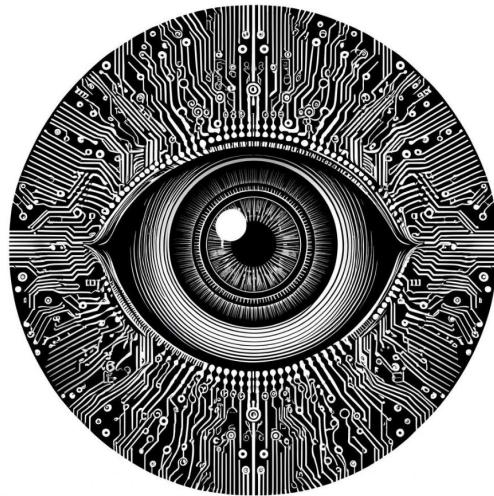
Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

- <https://arxiv.org/abs/1506.01497>
- <https://pyimagesearch.com/2023/11/13/faster-r-cnns/>

Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection

- <https://arxiv.org/abs/2303.05499>

Deep Learning Approaches to Image Segmentation



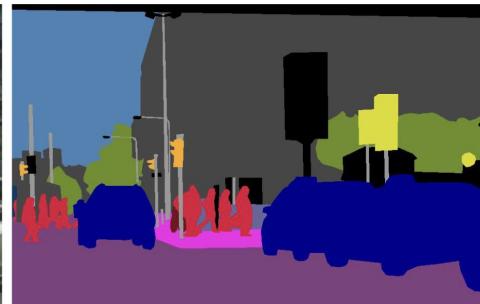
Learning goals

- Understand the deep learning solutions for labeled image segmentation: semantic, instance, panoptic
- Describe class-agnostic and zero-shot segmentation with Segment Anything (SAM)

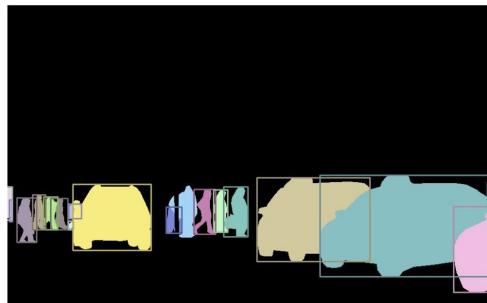
Semantic, Instance, and Panoptic Segmentation



(a) image



(b) semantic segmentation



(c) instance segmentation



(d) panoptic segmentation

Image from [Panoptic Segmentation](#)

Example of
instance
segmentation
with YOLO11

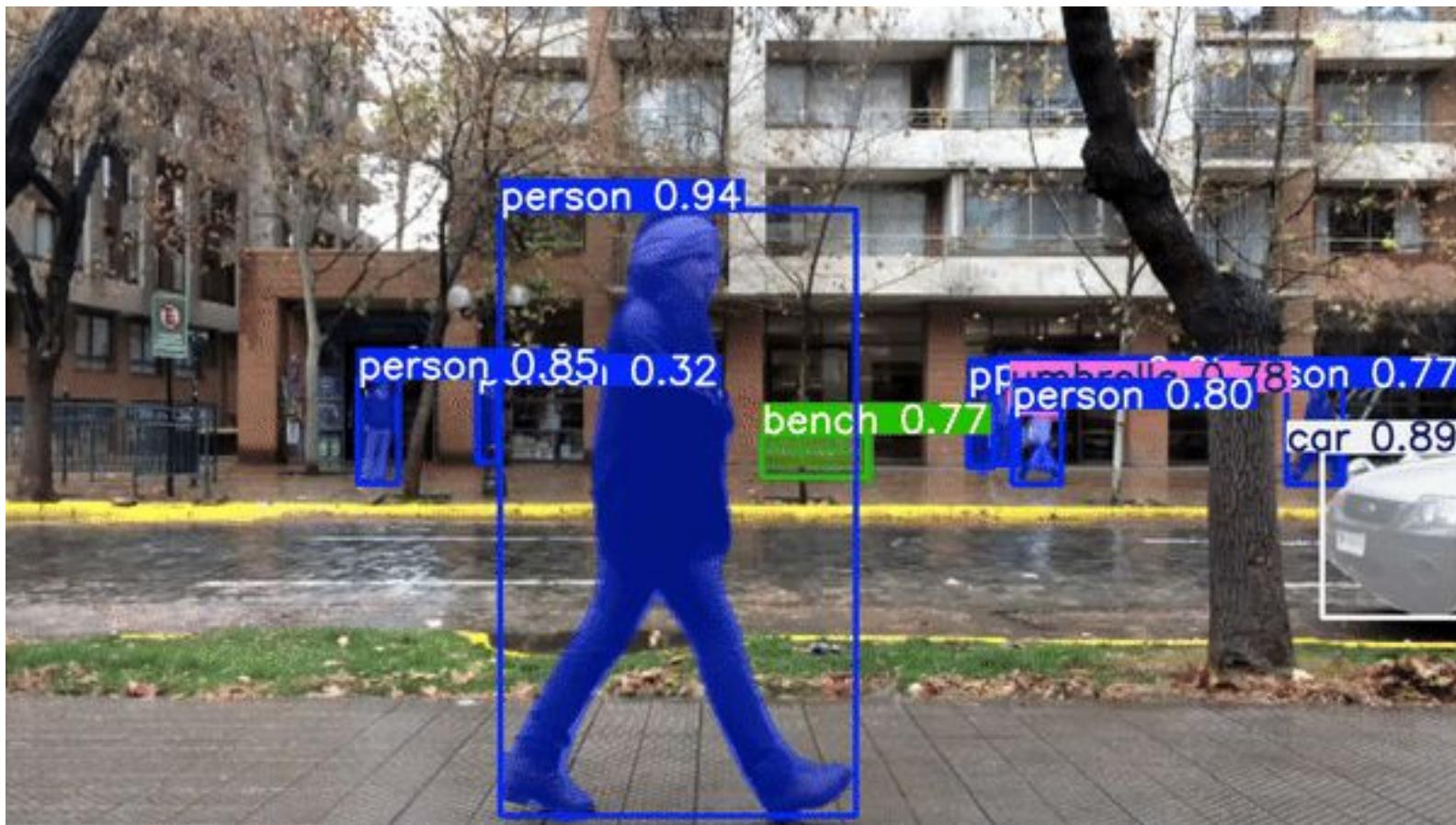
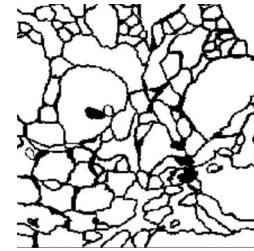
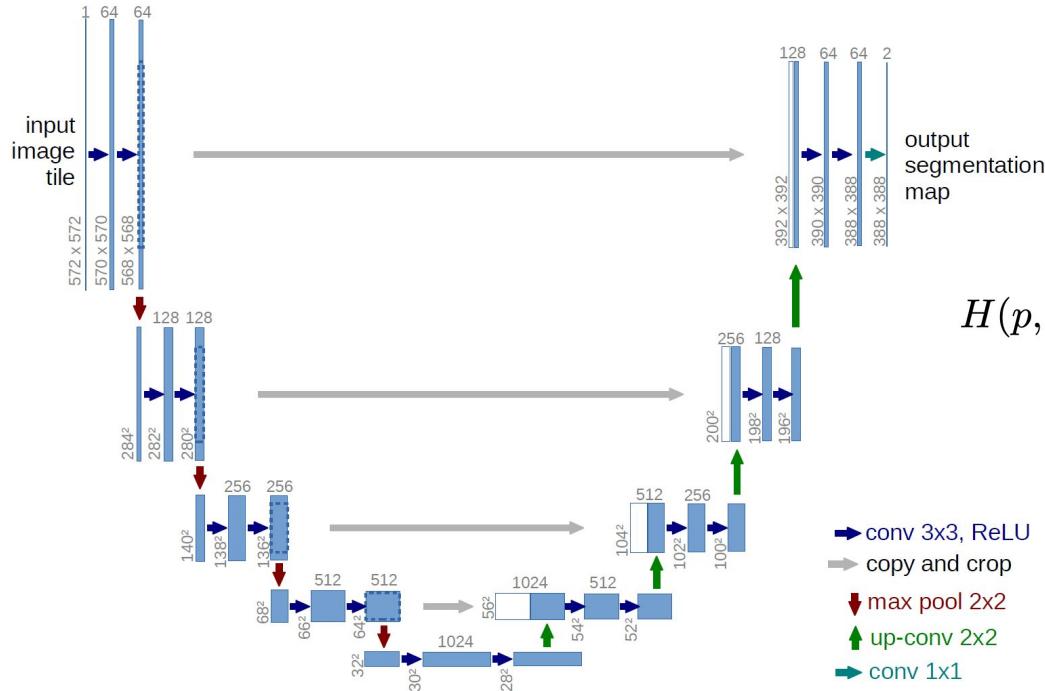
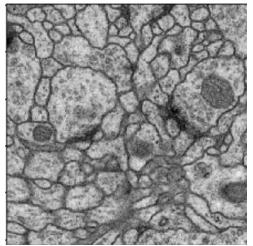
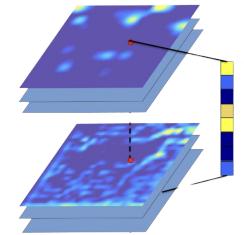


Image from <https://learnopencv.com/yolo11/>

Semantic segmentation with U-Net



$$H(p, q) = - \sum_i p(i) \log q(i)$$



Instance segmentation with Mask R-CNN

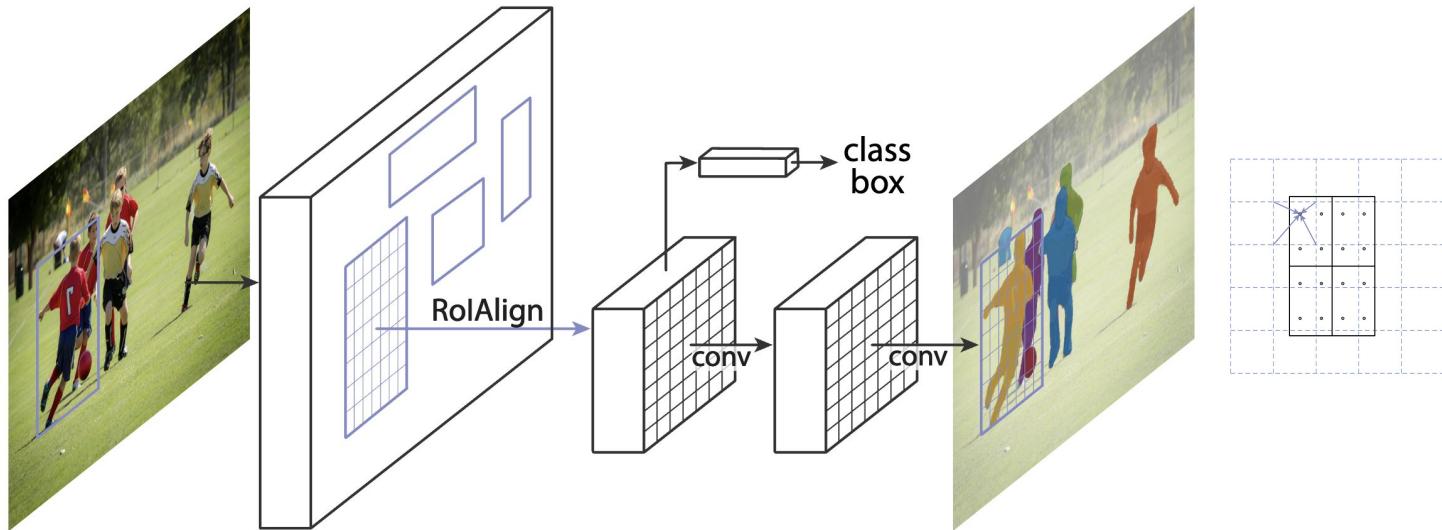
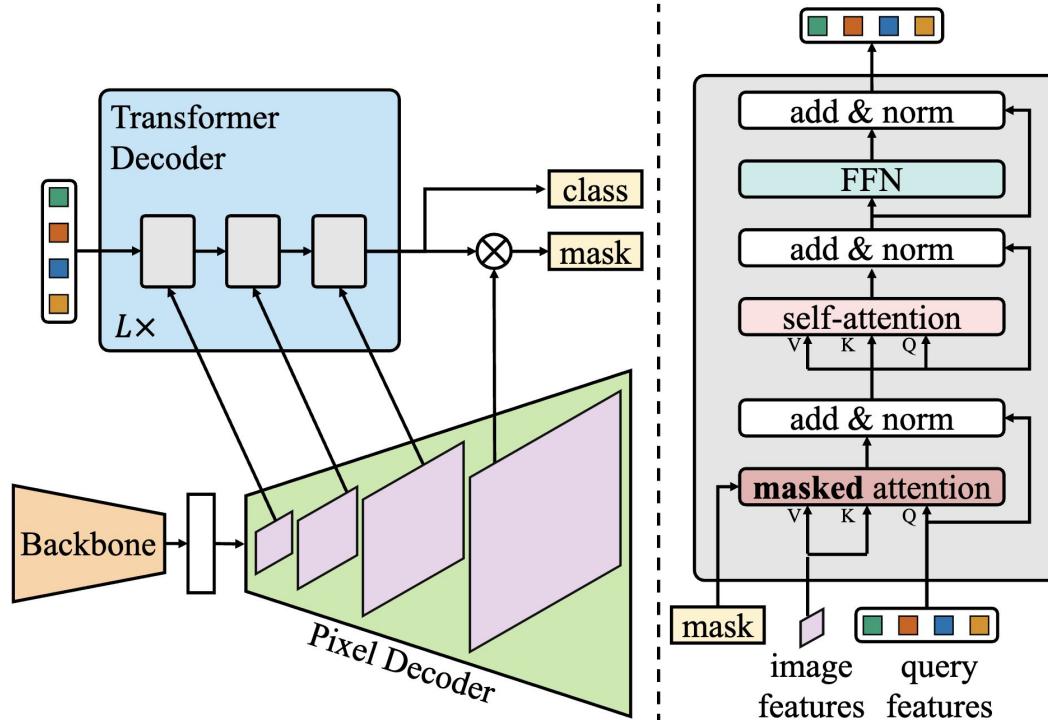


Figure 3. **RoIAlign:** The dashed grid represents a feature map, the solid lines (with 2×2 bins in this example), and the 4 sampling points in each bin. RoIAlign computes the value of each sampling point by bilinear interpolation from the nearest four feature map pixels. No quantization is performed on any coordinates involved in the RoI, its bins, or the sampling points.

Think of this as an additional pass after running Faster R-CNN on the image

Mask2Former: unified approach for labeled segmentation



Promptable masks with Segment Anything (SAM)



Figure 3: Each column shows 3 valid masks generated by SAM from a single ambiguous point prompt (green circle).

Image from [Segment Anything](#)

Class-agnostic segmentation with Segment Anything (SAM)

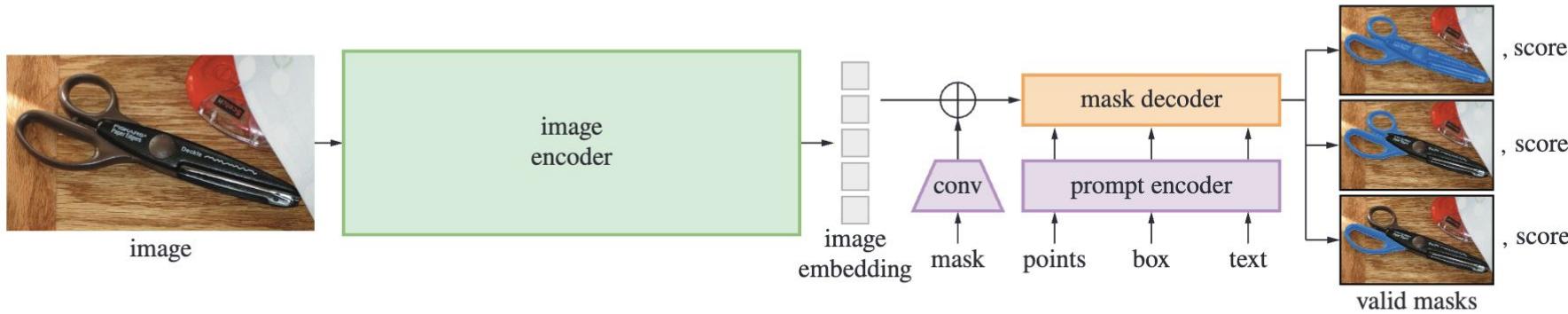


Figure 4: Segment Anything Model (SAM) overview. A heavyweight image encoder outputs an image embedding that can then be efficiently queried by a variety of input prompts to produce object masks at amortized real-time speed. For ambiguous prompts corresponding to more than one object, SAM can output multiple valid masks and associated confidence scores.

Image from [Segment Anything](#)

Creating labeled masks with object detectors and SAM

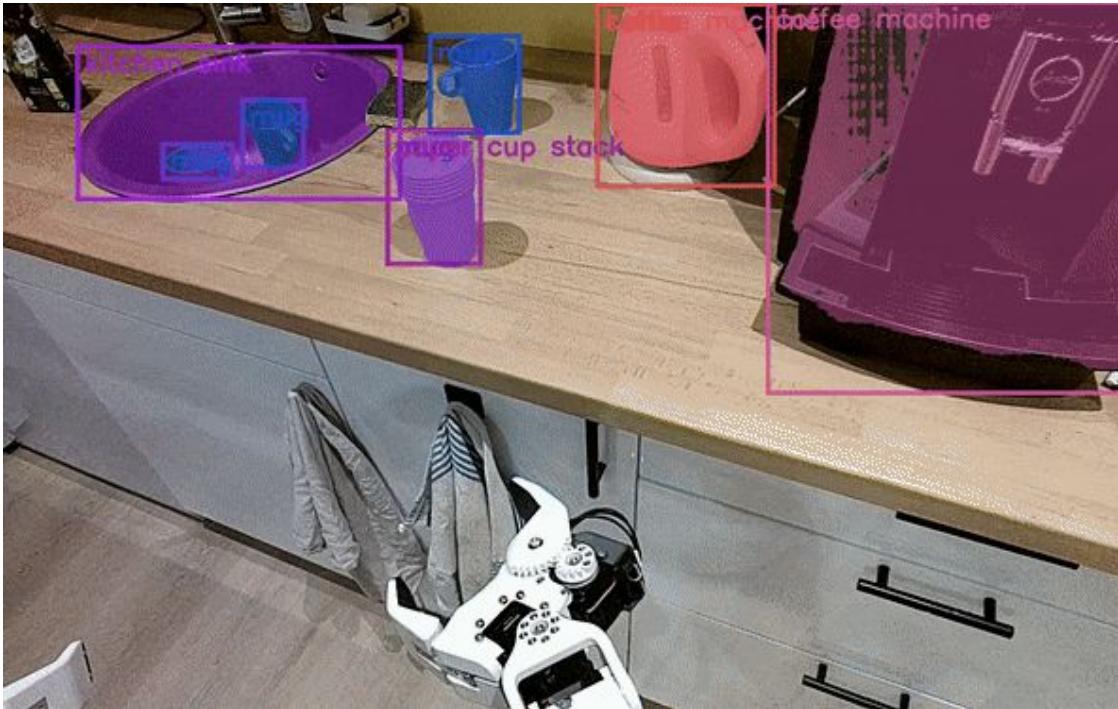


Image from <https://github.com/pollen-robotics/pollen-vision>

Summary

Semantic segmentation assigns class labels to individual pixels

- Loss functions compare predictions with ground truth masks at the pixel level

Instance segmentation separates objects of the same class

- Each detected object receives a unique mask identifier

Panoptic segmentation combines instance and semantic segmentation

- Labels pixels as countable (instance) or uncountable (semantic) classes

Segment anything (SAM) produces zero-shot masks

- Generates class-agnostic masks from prompts
- Integrates with object detectors for class labels

Further reading and references

U-Net: Convolutional Networks for Biomedical Image Segmentation

- <https://arxiv.org/abs/1505.04597>

Mask R-CNN

- <https://arxiv.org/abs/1703.06870>

Mask2Former: Masked-attention Mask Transformer for Universal Image Segmentation

- <https://arxiv.org/abs/2112.01527>

Segment Anything

- <https://arxiv.org/abs/2304.02643>

Resources

- [Github Repository](#)
- [YouTube playlist](#)
- [Discord channel](#)

#practical-computer-vision-workshops

Resources

- [Github Repository](#)
- [YouTube playlist](#)
- [Discord channel](#)

#practical-computer-vision-workshops

Review questions

- Object detection
- Image segmentation