Computer Vision: Object Detection, Tracking, Segmentation,

Terminology Recap: Learning

- We assume that there is some kind of **model** we are training (usually, a **neural network**)
- **Training**
  - We train on the **training data** - usually very large set of data, thousands, millions, billions of items
  - We validate on separate **validation data** - usually about 10% of the training data
  - We say that we have **overfitting** if the model works well on the training data, but not on validation. We say that it does not **generalize well**.

Terminology Recap: Learning

- **Training from scratch**: we start with an untrained, random model which we start training on your dataset
- **Transfer learning**: we start with a model that was trained on something else, and adapt it
  - We did this with vgg19: it was trained on ImageNet, we adapted it to our cat-dog-monkey dataset
  - The term is usually used when we move from one domain from another
- **Finetuning**: further refine an already trained model
  - The term is usually used when the finetuning brings in new requirements
  - Most famous example: finetune an already trained LLM to follow human instructions

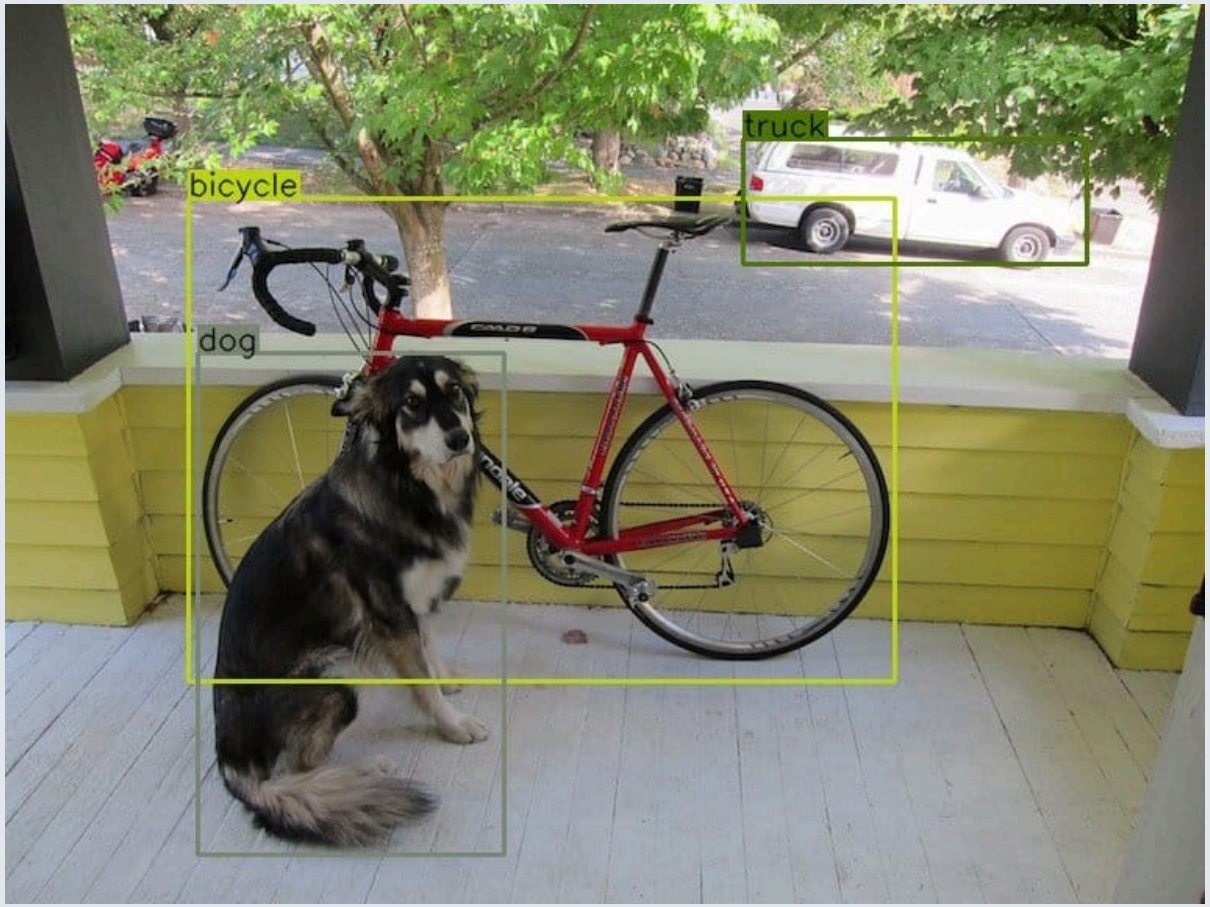
Terminology Recap: Learning

- **Few-shot learning**: learn from a small number of training examples (eg. 5 dog pictures)
- **One-shot learning**: self-explanatory
- **Zero-shot learning**: learn without an example
    - Usually, instead of an example, we have a **description**: eg. a "a green panda"

The problem: Object Detection

- Let us say that I show you a picture
  - and I give you a certain type of object: "human", "car" etc.
- I want a tool that:
  - identifies objects of that type in the picture
  - gives me information of where are those (eg. in the form of a box drawn around them)
  - possibly also gives me information of how confident it is that it is an object of that kind

Example of object detection



What makes this an object detection problem:

- **Inputs**:
  - An image
  - A list of classes ("human", "car")
- **Outputs**: a list of bounding boxes
  - X, Y, width, height
  - label: "human", "car"
  - confidence: 0.8, 0.5, etc.

How do I know that it works?

- We often use the term **ground truth** for what is really there
- **False negatives** - objects that are in the picture, the right type, but not detected
- **False positives**
  - Objects that are detected, but they are not there in the ground truth
  - Or the box is in the wrong place
  - Or multiple detections are made when in reality we only have one object
  - Or we detect the wrong label (mistake a car for a human)

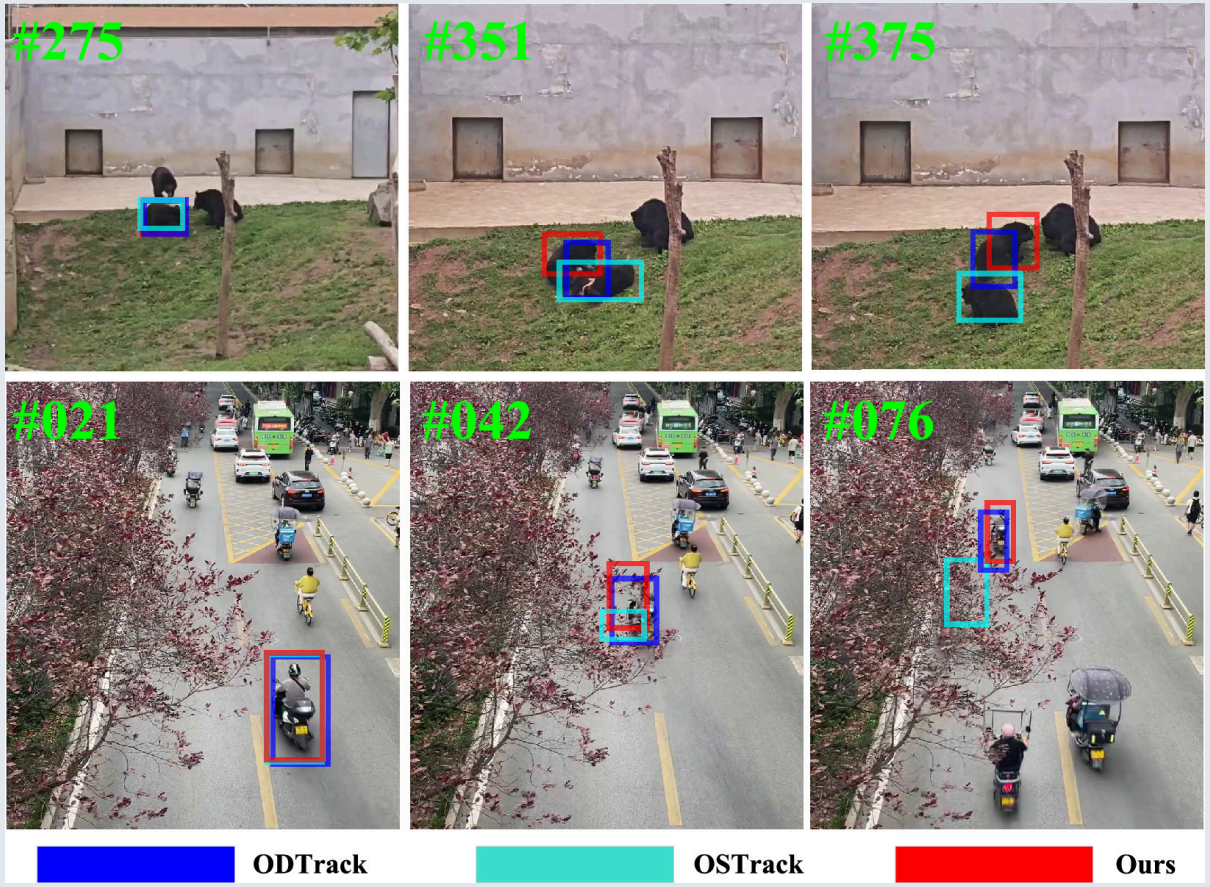
How do I know that it works? (cont'd)

- **Localization accuracy**
  - IoU (Intersection over Union)
    - Overlap between predicted and true bounding boxes.
    - Gives 1 if the boxes overlap perfectly, 0 if they are disjoint
  - Center error / position error
    - Distance between predicted and true object centers (often in pixels).

The problem: Object Tracking

- Sister problem of object detection
- We can try to do object detection in every frame of a video, but this is not enough
- We want to get the **trajectory** of objects, while maintaining identity accross frames
    - Even if the objects might not be visible in some frames!

Example of object tracking



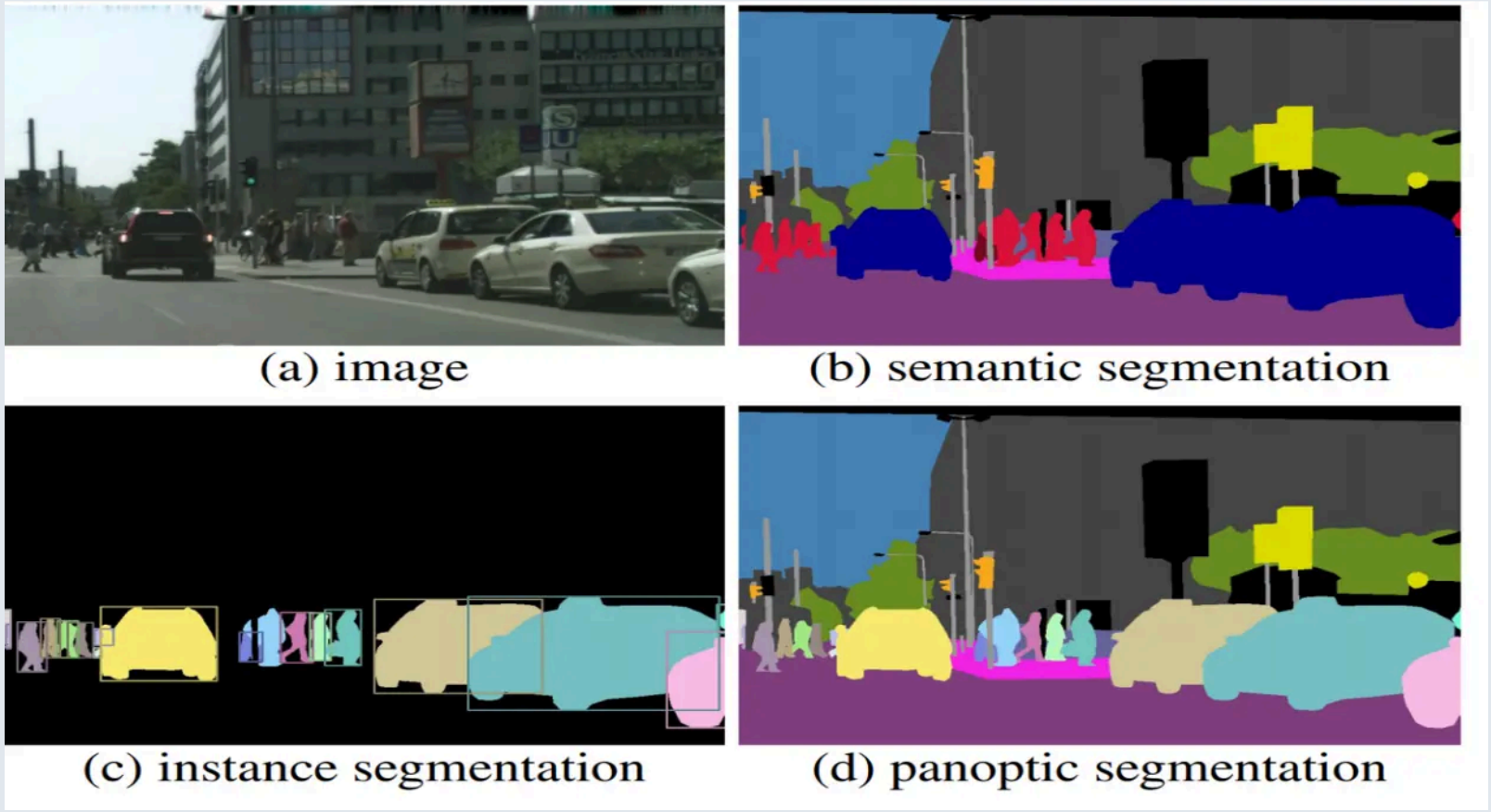
How do I know that it works?

- All the performance measures of object detection still apply
- Tracking robustness: how consistently the tracker stays on an object
  - **Success rate**: fraction of frames where the IoU exceeds a threshold
  - **Failure rate**: number of times the tracker loses the object.
  - **Tracking length**: average duration the tracker follows an object without failure.

The problem: Semantic segmentation

- For a picture, label each pixel with the class of the object it belongs to.
  - Grass, sky, person
  - It needs to detect the exact contours of where a person starts and ends, not just to draw a box around them.
- In its default definition, the problem of semantic segmentation only does not separate instances
  - For instance, a crowd of people will not be separated into individual people.
  - Even solving this problem can be useful (e.g.) for identifying driveable surface for a self-driving car
- Instance segmentation: separate individual object
- Panoptic segmentation: semantic + instance segmentation

Example of segmentation types



How it works? Let's invent an object detector!

- We already have code for image classification
- Let us invent a technique for an object detector!
- Sliding windows

Problems with our solution

- We have to go repeatedly to the classifier --> slow
- We are going to have multiple boxes for which we find the object... --> false positives

How it works: convolutional neural network based object detection

- The first really workable approaches for object detection were based on CNN-s, not unlike our idea
- R-CNN (Region-based Convolutional Neural Network) is an early deep-learning approach to object detection.
    - It breaks detection into a sequence of steps instead of doing everything in one pass.
    - Proposes regions that might contain objects (1000s of them...)
    - Classifies each region using a CNN
    - Refines the bounding boxes

Successor algorithms

- To make R-CNN faster, a number of important improvements had been proposed
  - Fast R-CNN
  - Faster R-CNN
  - Mask R-CNN (adds instance segmentation)
  - etc.

YOLO

- YOLO (You Only Look Once) is an object detection approach that treats detection as a single prediction problem, not a multi-step pipeline.
- Instead of proposing regions and classifying them one by one, YOLO:
  - Looks at the entire image once
  - Predicts bounding boxes and class probabilities in one forward pass
- A family of models, Yolo v8 also supports segmentation
- Application note: as of 2026, if what you want is just object detection, this is what you want to use
  - Easy to use, you can run it on your own computer, fast.
  - You can get better performance with other systems, but at a higher computational cost

Try it out: YOLO

Transformer based segmentation

- The current state of the art object detection and segmentation approaches are based on a different technique
- The foundation is a neural network model called a **transformer**
  - Introduced in the 2017 paper called "Attention is all you need"
  - This is what large language models etc. use.
- The general idea is that we can transform the image in some kind of internal representation ("image encoder")
  - And then we query this representation with language.
- We will revisit transformer based models later.

Segment Anything

- The current state of the art in segmentation. A model developed by Meta AI.
  - Current version SAM 3
- Is a computer vision system designed to segment any object in an image with minimal input, even objects it has never seen before.
- Segment Anything separates image understanding from user interaction:
  - First, it builds a general representation of the image
  - Then, it produces object masks based on simple prompts

Try it out: Segment Anything

- https://aidemos.meta.com/segment-anything/

Applications

- List of applications here

Pitfalls and dangers: Overtrust