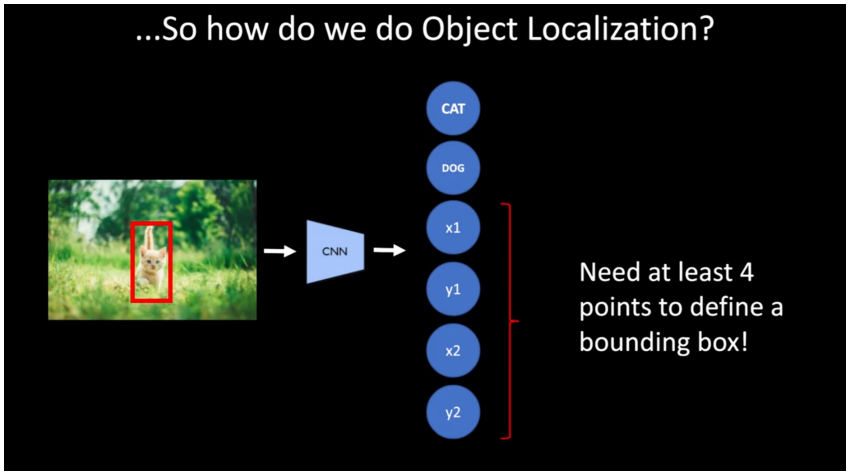
OBJECT DETECTION

Object localization: finding what and where a single object is in an image (simple case of object detection)
Object detection: finding what and where multiple objects are in an image

How is object localization done?



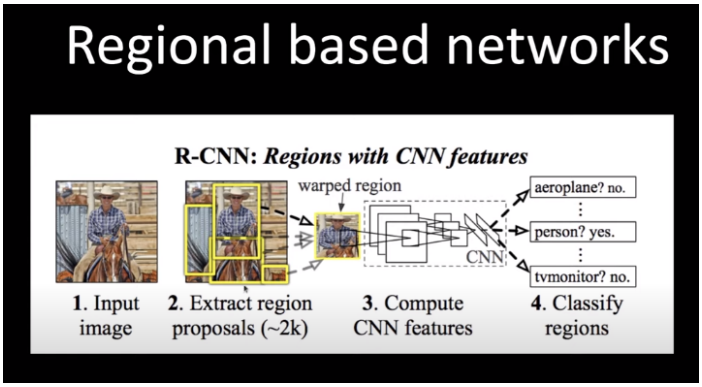
Let's consider the cat vs dog classification problem using CNNs. Normal CNN with 2 output nodes for cat and dog. But we add 4 additional nodes to define a bounding box for the object. You can imagine that for classification, you can have a cross entropy loss for the prediction between cat and dog. But for the bounding box, you can have some L2 loss or MSE on those particular nodes.

This broadly consists of two steps :

1. **Object Localization**: Here, a bounding box or enclosing region is determined in the tightest possible manner in order to locate the exact position of the object in the image.
2. **Image Classification**: The localized object is then fed to a classifier which labels the object.

Some approaches

- Sliding window approach:
  - Define a bounding box beforehand. The box slides and we see if there's a cat or dog in the crop.
  - Problems:
    - Lot of computation needed (each crop is sent to CNN)
    - You need to try multiple times for different sized bounding boxes
    - You get many positive predictions for different bounding boxes. Which one to pick?
- Regional based networks (RCNN)
  - Solved the computation problem from SWA.



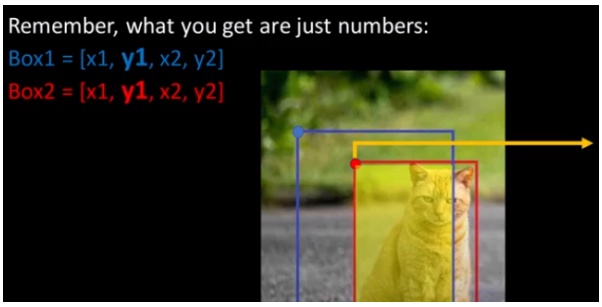
  - You have an image. A deterministic algorithm extracts region proposals, i.e potential bounding boxes. The region is resized to a fixed size which is passed through a CNN.
  - What this solved is:
    - We have a fixed number of region proposals (2000)
    - We don't have to worry about determining the bounding box size.
  - Fast RCNN and Faster RCNN were released after. They region proposal algorithm also became a CNN
  - Problems: still slow and theres 2 step process

Evaluating bounding box prediction

We use the metric called Intersection over Union (IoU)

Steps:

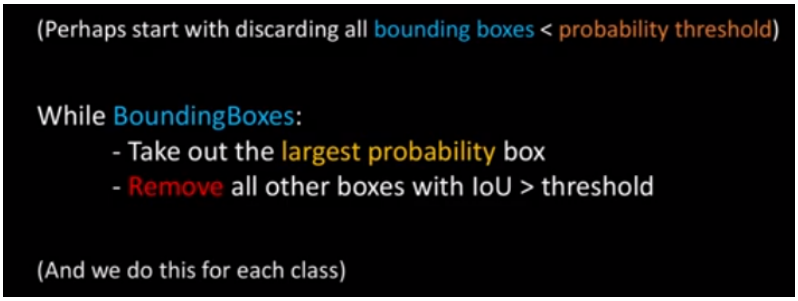
- Calculate intersection of the bounding boxes
- Calculate union of the bounding boxes
- IoU = intersection/union
    - Returns a value between 0 and 1 that indicates how good a prediction is
    - 0.5, 0.7, 0.9 = decent, pretty good, almost perfect



    - X1 = max(box1[0],box2[0]) AND y1 = max(box1[1],box2[1])
    - X2 = max(box1[2],box2[2]) AND y2 = max(box1[3],box2[3])

Non max suppression

- Method for cleaning up bounding box predictions: model will produce multiple bounding boxes for an object; pick the right one.



- The bounding box belongs to is also something that our model is going to learn to predict

Mean Average Precision

1. Get all bounding box predictions on test set
2. You get a true positive if predicted bounding box has IoU with actual bounding box of over 0.5
3. Precision: TP/TP+FP: of all the predictions, what fraction was actually correct
4. Recall: TP/TP + FN: of all target bounding box, what fraction was correct
5. Calculate precision and recall and for each bounding box

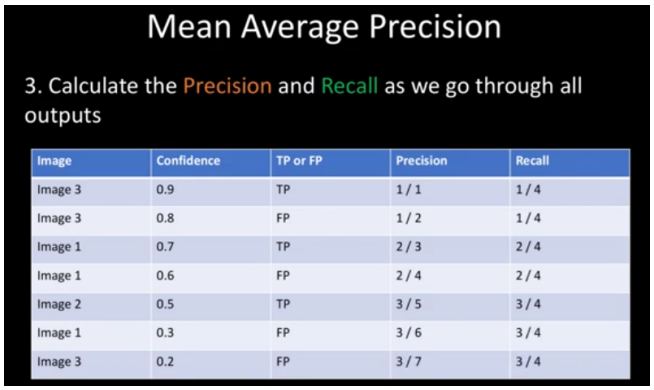
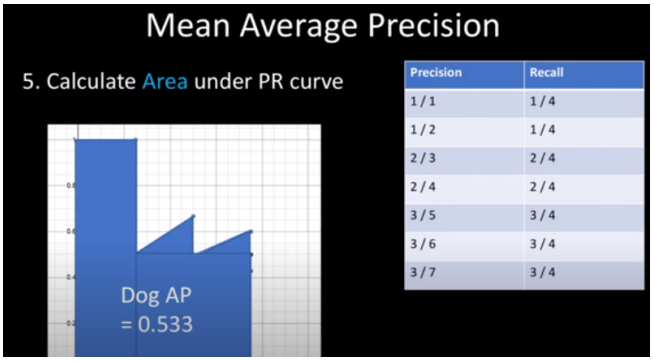


Image	Confidence	TP or FP	Precision	Recall
Image 3	0.9	TP	1 / 1	1 / 4
Image 3	0.8	FP	1 / 2	1 / 4
Image 1	0.7	TP	2 / 3	2 / 4
Image 1	0.6	FP	2 / 4	2 / 4
Image 2	0.5	TP	3 / 5	3 / 4
Image 1	0.3	FP	3 / 6	3 / 4
Image 3	0.2	FP	3 / 7	3 / 4

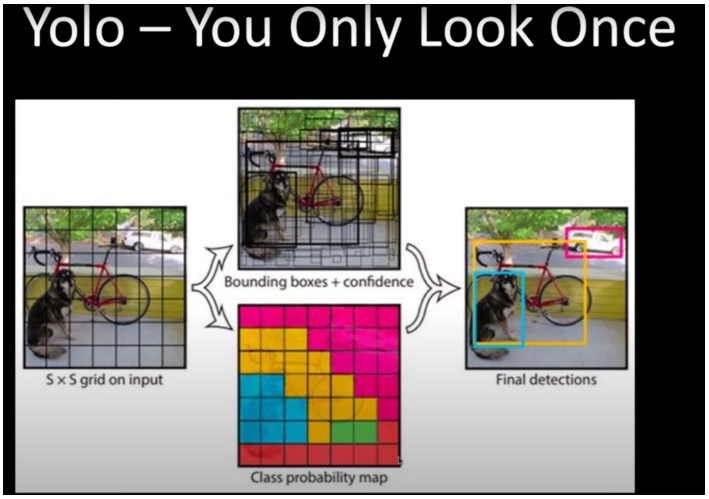
6. Plot precision vs recall on graph

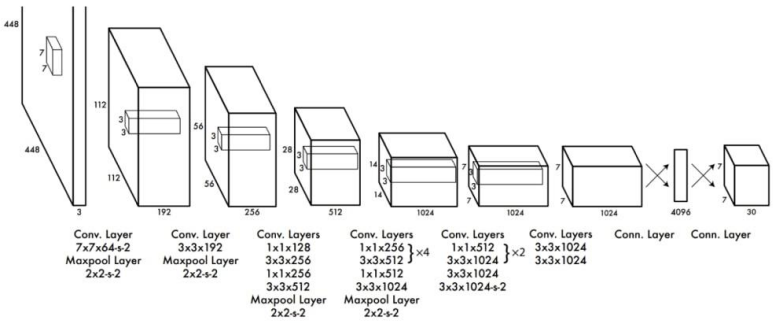


7. Do the above for all classes
   mAP = sum of average precision of each class divided by total number of classes
8. The threshold specified in step 2 is the 0.5 in mAP:0.5. We need to redo the computations for other IoUs
9. Averaging mAPs for all IoUs is the final IoU.

YOLO

- Split the image into an SxS by grid
- Each cell is responsible for predicting:
  - If there's a bounding box in that cell
  - What the class probability is for that particular cell
- A cell is responsible for outputting the bounding box if it's the center point of the object.
  - Difficult for a cell to know if it's the center of the bounding box. So you get many bounding boxes





The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

- The first 20 convolution layers of the model are pre-trained using ImageNet by plugging in a temporary average pooling and fully connected layer.
- Then, this pre-trained model is converted to perform detection. YOLO's final fully connected layer predicts both class probabilities and bounding box coordinates.
- YOLO doesn't use only convolutions. As you can see in the architecture diagram, there are two fully-connected layers between the main convolutional part and the final convolutional output. These fully-connected layers allow it to essentially do regression on the bounding box center coordinates as well as the size and width which can range over the whole image.

IMAGE SEGMENTATION

Image segmentation: is a method of dividing a digital image into subgroups called image segments, reducing the complexity of the image and enabling further processing or analysis of each image segment.

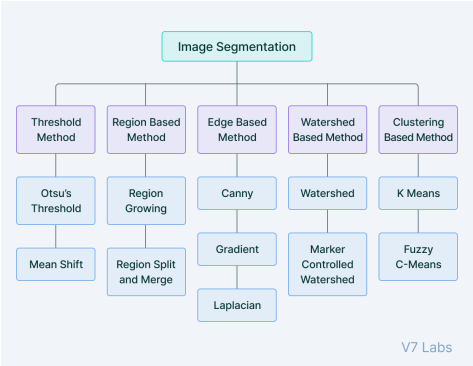
Types

- Instance
- Semantic
- Panoptic

Traditional Methods

Traditional image segmentation techniques based on such heuristics can be fast and simple, but they often require significant fine-tuning to support specific use cases with manually designed heuristics. They are not always sufficiently accurate to use for complex images.

- Edge based segmentation
- Threshold based segmentation
- Region based segmentation
- Cluster based segmentation
- Watershed Segmentation



Machine Learning methods

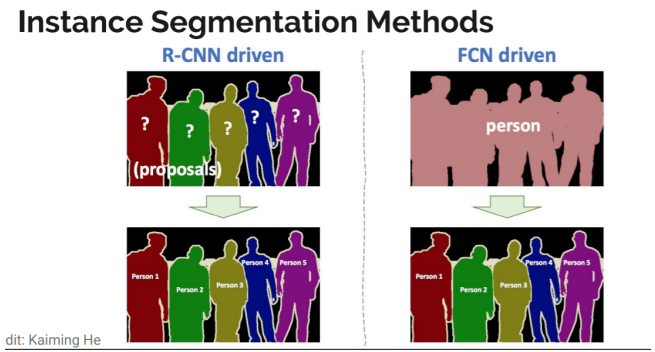
Machine learning-based image segmentation approaches use model training to improve the program's ability to identify important features. Deep neural network technology is especially effective for image segmentation tasks.

- An **encoder**—a series of layers that extract image features using progressively deeper, narrower filters. The encoder might be pre-trained on a similar task (e.g., image recognition), allowing it to leverage its existing knowledge to perform segmentation tasks.
- A **decoder**—a series of layers that gradually convert the encoder's output into a segmentation mask corresponding with the input image's pixel resolution.
- **Skip connections**—multiple long-range neural network connections allowing the model to identify features at different scales to enhance model accuracy.

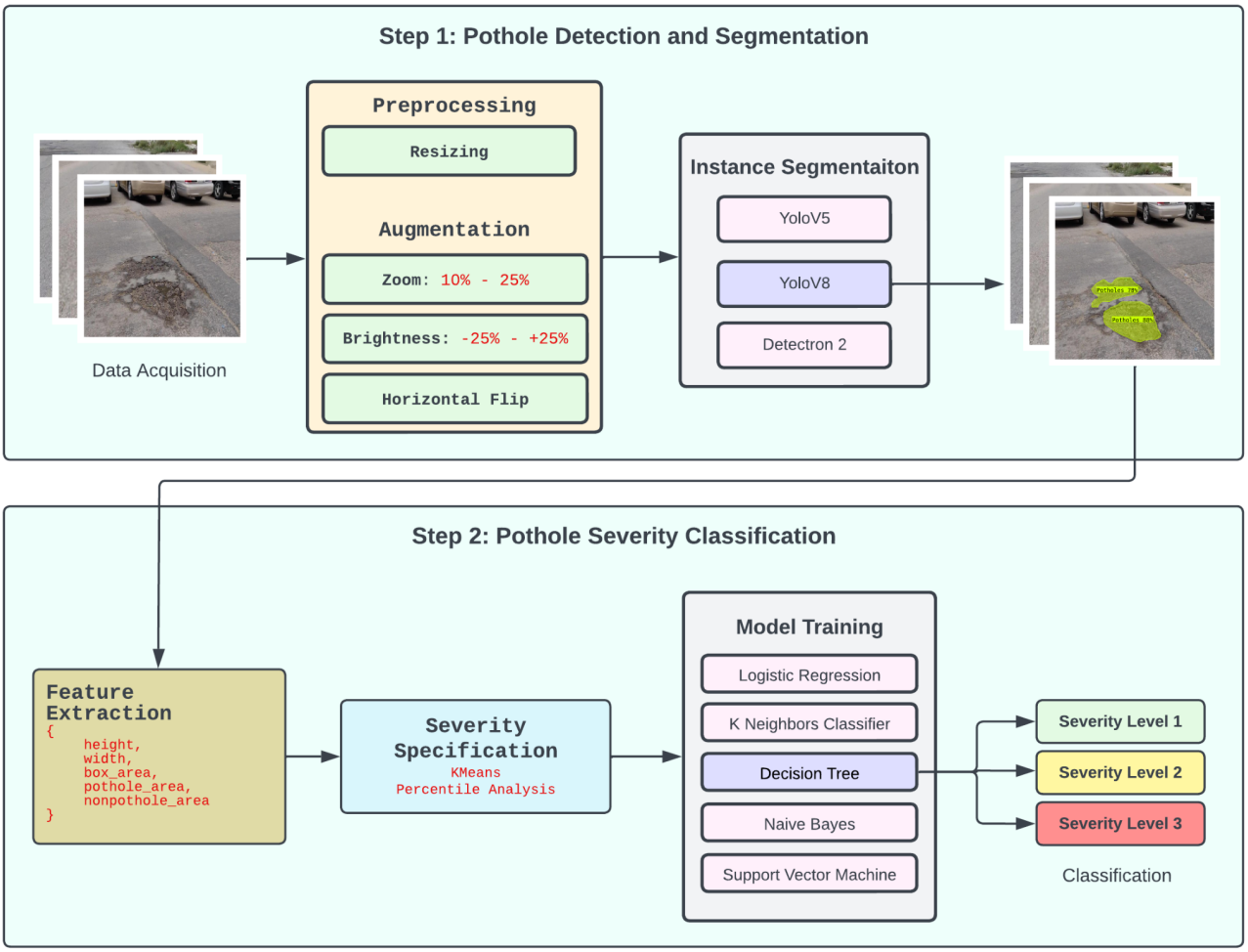
How does Instance Segmentation work?

- Instance Segmentation methods can be both R-CNN driven or FCN driven.
    - FCN (fully convolutional networks): Dense layers are replaced by convolutional layers to simulate the sliding window concept. The shape of the convolutional layers are modified so as to approximate the Dense layer functionality as well.
- Instance segmentation contains 2 major parts:
    - Object Detection
    - Semantic segmentation.

  In other words, it just runs object detection firstly, then uses a semantic segmentation model inside every rectangle



APPROACH



SEGMENTATION RESULTS

YoloV8

	Precision	Recall	mAP:50	mAP:50-95
Box	0.982	0.975	0.987	0.734
Segmentation	0.982	0.975	0.987	0.705

Yolov5

	Precision	Recall	mAP:50	mAP:50-95
Box	0.979	0.976	0.986	0.722
Segmentation	0.979	0.976	0.986	0.648

Detectron 2

	Precision	Recall	mAP:50	mAP:50-95
Box	0.95	0.94	93.731	62.607
Segmentation	0.95	0.94	95.444	62.615

CLASSIFICATION RESULTS

Classification Accuracy Table

Model	Accuracy
Logistic Regression	0.975
K-Nearest Neighbors	0.978
RBF Kernel Support Vector Machine	0.983
Naive Bayes	0.978
Decision Tree	0.989

Stats Table

	height	width	area	pothole_area	nonpothole_area
mean	141.246237	160.008602	24136.586022	16398.926882	7737.659140
std	61.173830	60.393898	16866.241902	11359.076938	6335.443707
min	16.000000	26.000000	416.000000	297.000000	119.000000
25%	93.000000	114.000000	11819.500000	7876.750000	3174.500000
50%	140.500000	152.000000	20167.500000	13939.000000	6038.500000
75%	185.000000	196.000000	33099.000000	22620.000000	10167.500000
max	339.000000	368.000000	106107.00000	87658.000000	50293.000000

Plots