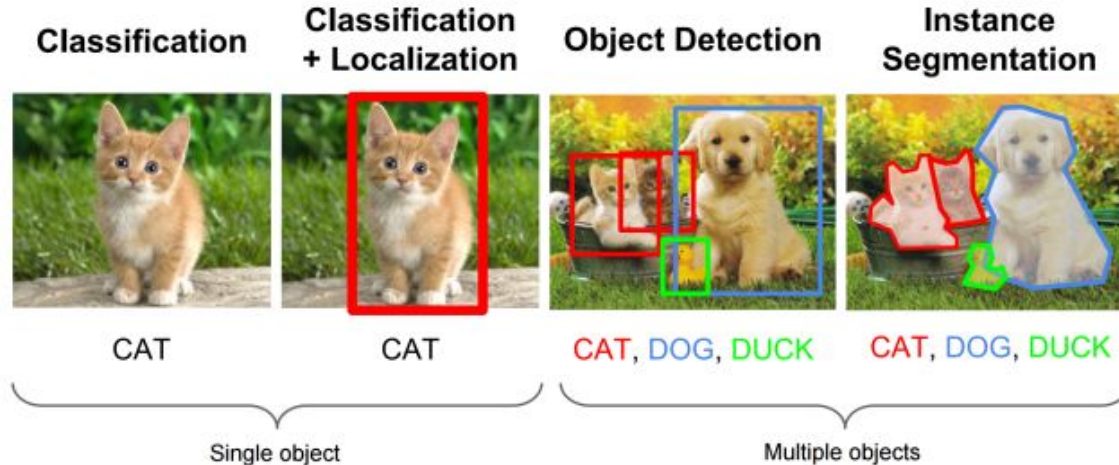


YOLO Algorithm for Image Segmentation

What is YOLO?

- You Only Look Once- Real time object detection- 45fps
- Real time object detection algorithm.
- Object detection consists of: Image classification; Object localization; Object Detection; Image/ Instance segmentation.



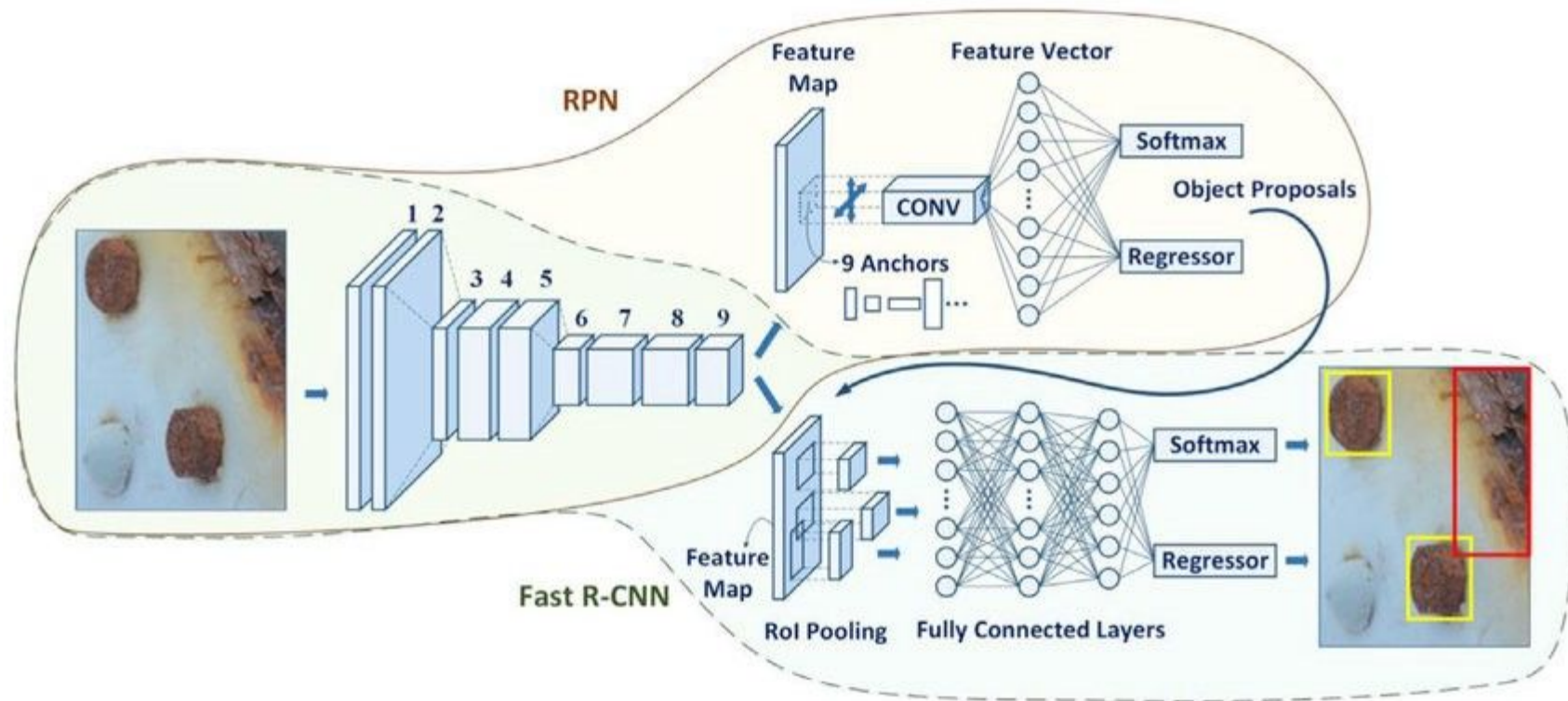
You Only
Live Once

You Only
Look Once

Traditional Object Detection Methods

- Used classifiers and repurposed them to perform object detection.
 - Deformable Parts Models (DPMs) used a sliding window approach.
 - R-CNN family (R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN)-
Region proposal methods where potential bounding boxes were created on an image and then a classifier was run on these bounding boxes.
- After classification, post-processing is used to refine the bounding boxes, eliminate duplicate detections, and rescore the boxes based on other objects in the scene.
 - This is slow and hard to optimize as the various sub components have to be trained separately.

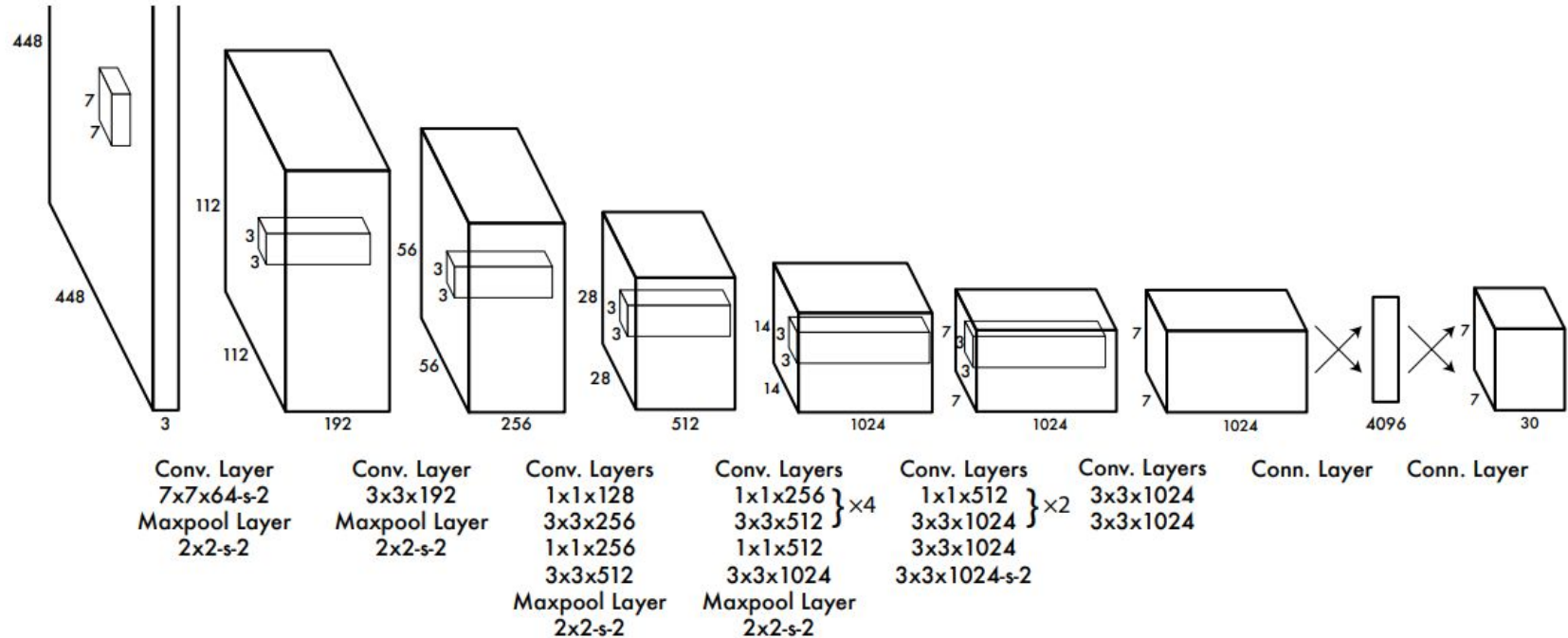
Faster R-CNN Architecture



What does YOLO do?

- Reframes object detection as a single regression problem.
- So instead of having several neural networks interconnected with each other, there is just one model which does all the work- classification, localization, detection, and segmentation.
- The model has just 24 convolutional layers followed by 2 FC layers.
- The single CNN predicts bounding boxes and class probabilities for those boxes.

YOLOv1 Architecture

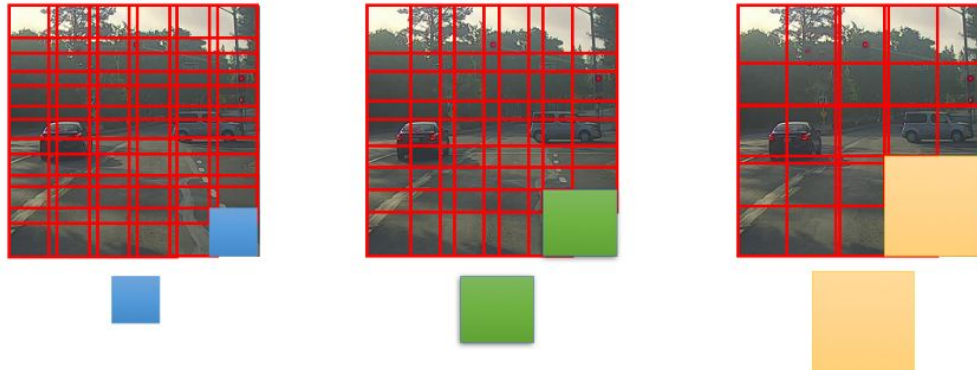


What makes YOLO good?

- Convolutional implementation of sliding window approach
- Intersection over Union
- Non-Max Suppression
- Anchor boxes

Sliding Window approach to Object Detection

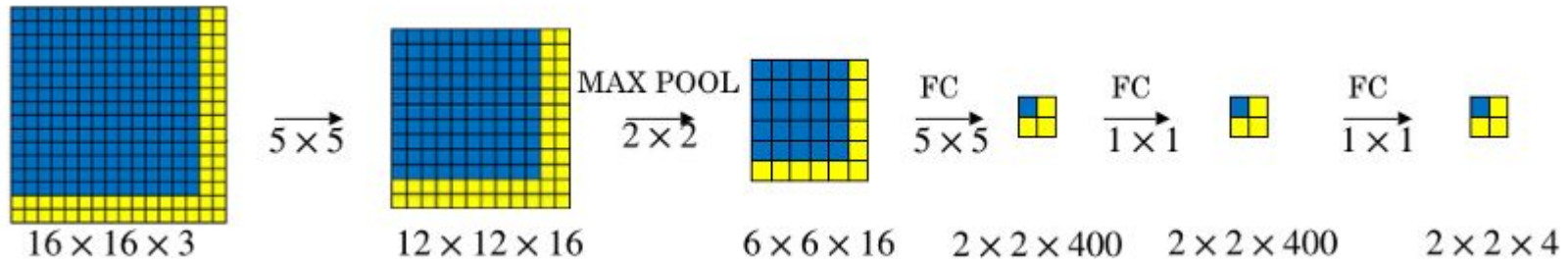
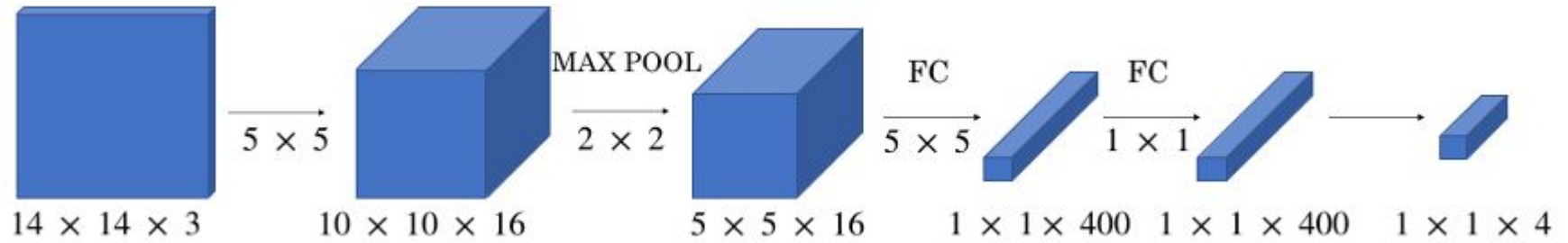
- In this algorithm, a grid cell of a specific size is chosen. Let us take the size of 2×2 .
- We pass the above grid cell through the image and convolute the part of the image in the grid cell and predict the output- basically run an image classification algorithm on that piece of the image.
- Then we slide the grid cell through stride-2 and then convolute the next part of the image.
- In this way, we go cover the whole image.
- We repeat the same procedure with different grid cells size.



- Note that the same image has to be passed through the NN several times- the NN looks at the image multiple times.
- What does the output look like?
- Pc- Whether or not there is an object.
- bx, by- centre of the image
- bh, bw- height and width of the image
- c1, c2, c3 etc.- which class the image belongs to.

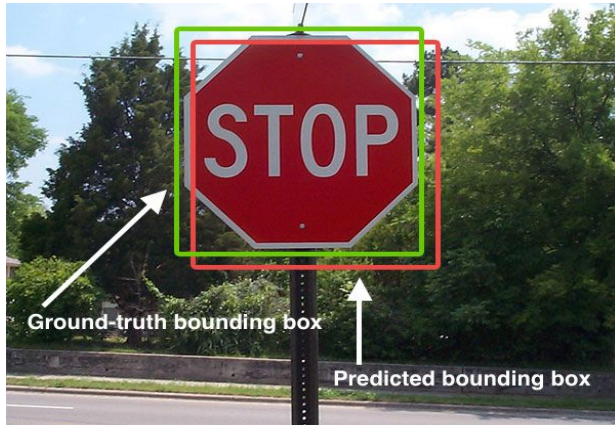
$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

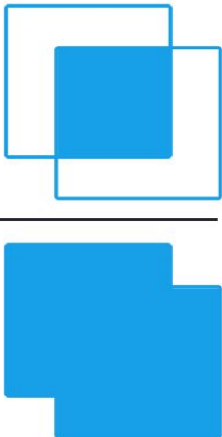
Convolutional approach to sliding window



Intersection Over Union

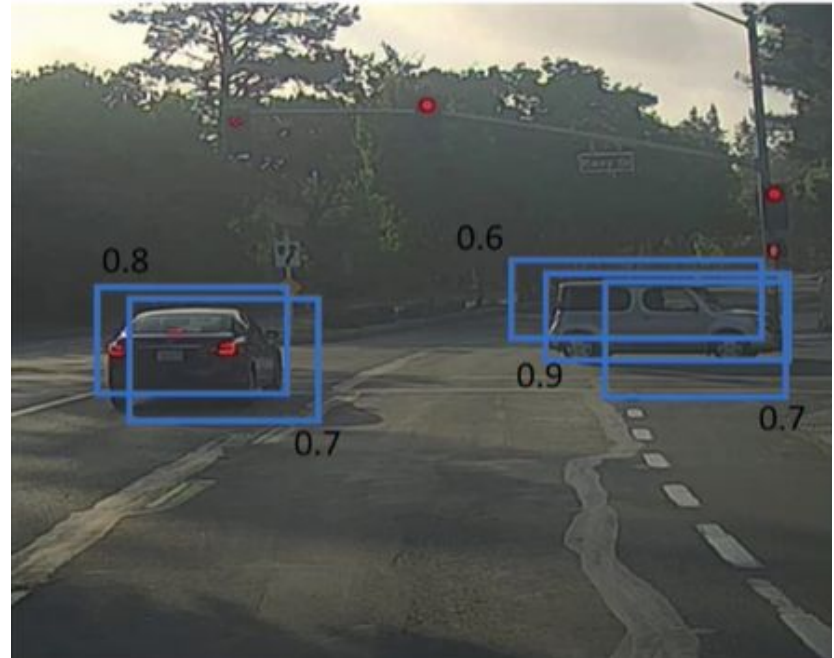
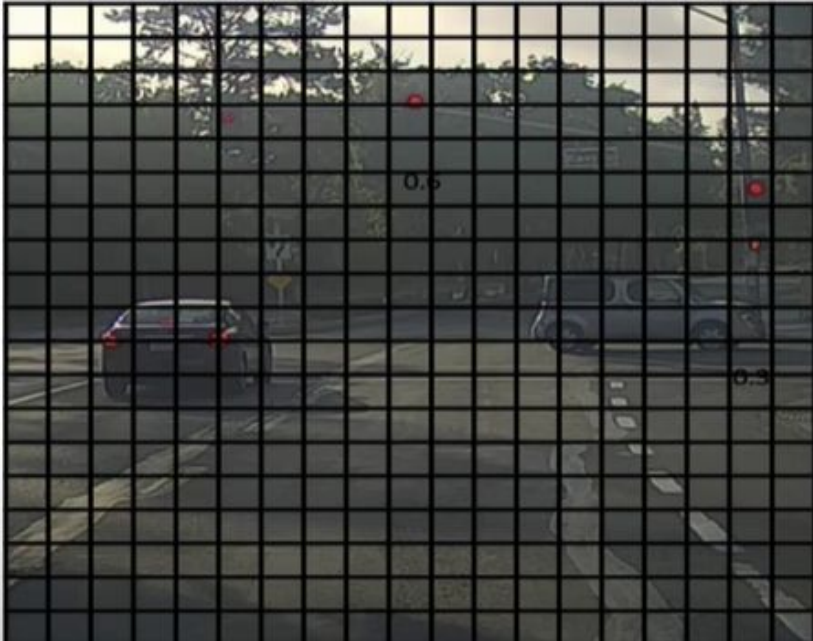
- An evaluation metric used to detect accuracy of object detector.
- Finds the intersection of ground truth and predicted bounding boxes, and divides it by the union of ground truth and predicted bounding boxes.
- Your object detection model might predict several bounding boxes- IoU is used to choose the best bounding box.



$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


Non Max Suppression

- What if there are multiple instances of the same object?

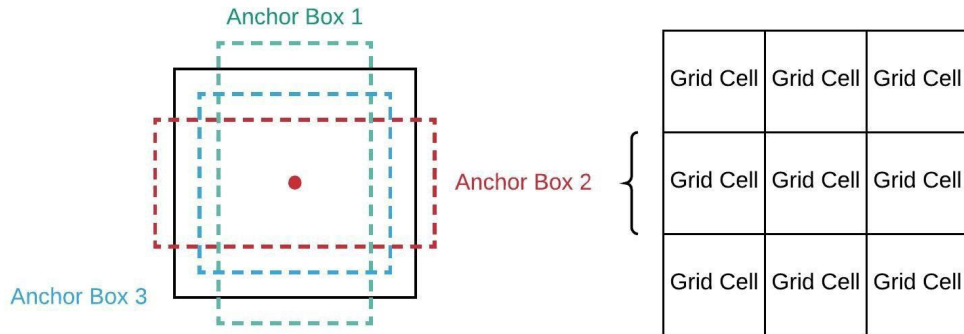


Algorithm of non max suppression:

- For each class of object:
 - Discard all bounding boxes with confidence score < 0.6
 - While there are any remaining boxes:
 - Pick box with largest confidence, output that as prediction.
 - Discard any remaining boxes with IoU > 0.5 with the output box.
- Non- max suppression- any bounding box that does not have maximum confidence score for a particular instance of an object is suppressed.

Anchor Boxes

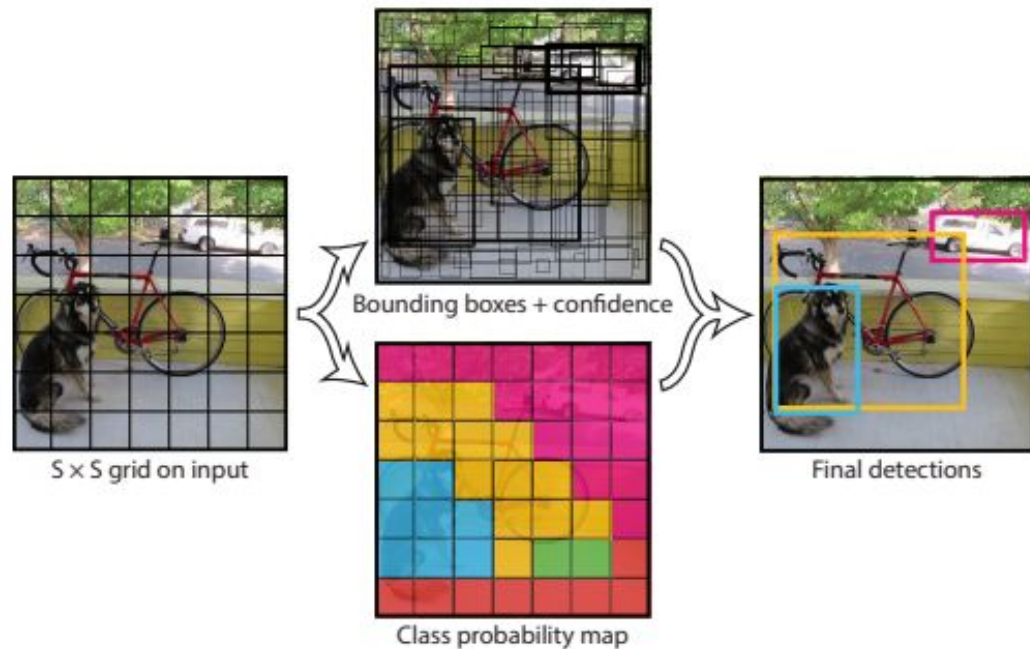
- Traditionally, one object is assigned to one grid cell- if the centre of an object falls in a particular grid cell, then that grid cell is responsible for detecting it.
- However, what if many objects have centre in the same grid cell?
- Several anchor boxes of different sizes and shapes.
- Each object is assigned to the anchor box with highest IoU with the anchor box.



$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

YOLO

- It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.
- Later versions of YOLO (v2 and v3) use Darknet architecture to extract features.



YOLO- Outputs

