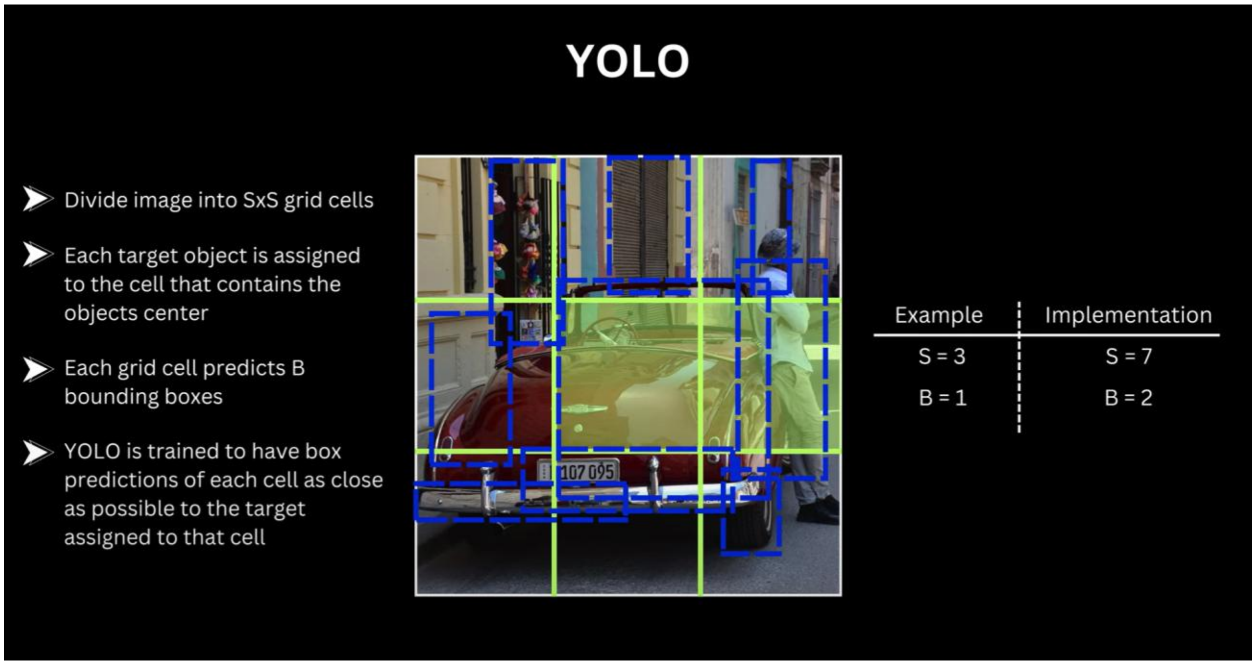
YOLOv1



YOLO

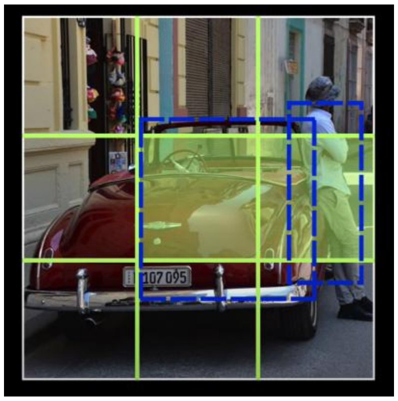
- Divide image into SxS grid cells
- Each target object is assigned to the cell that contains the objects center
- Each grid cell predicts B bounding boxes
- YOLO is trained to have box predictions of each cell as close as possible to the target assigned to that cell

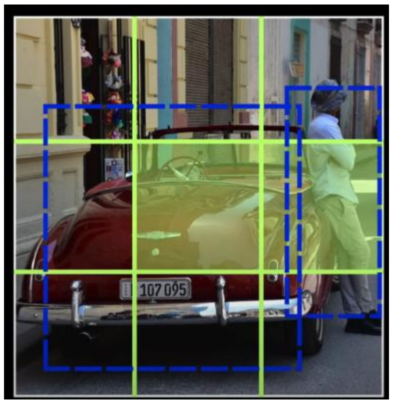
Example	Implementation
S = 3	S = 7
B = 1	B = 2

During training, each target object is assigned to the cell that contains the object’s center.

During training, each grid cell predicts B bounding boxes. And then YOLO learns to predict the bounding boxes as close as possible to the bounding boxes assigned to that cell—if any.

So for those cells that had targets assigned we had these predictions:





During training, YOLO learns to make these predictions as close to the ground truth boxes

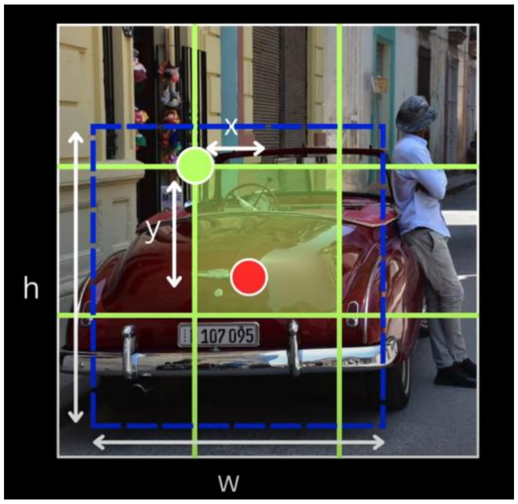
YOLO predicts 5 parameters for each bounding box predicted.

It predicts

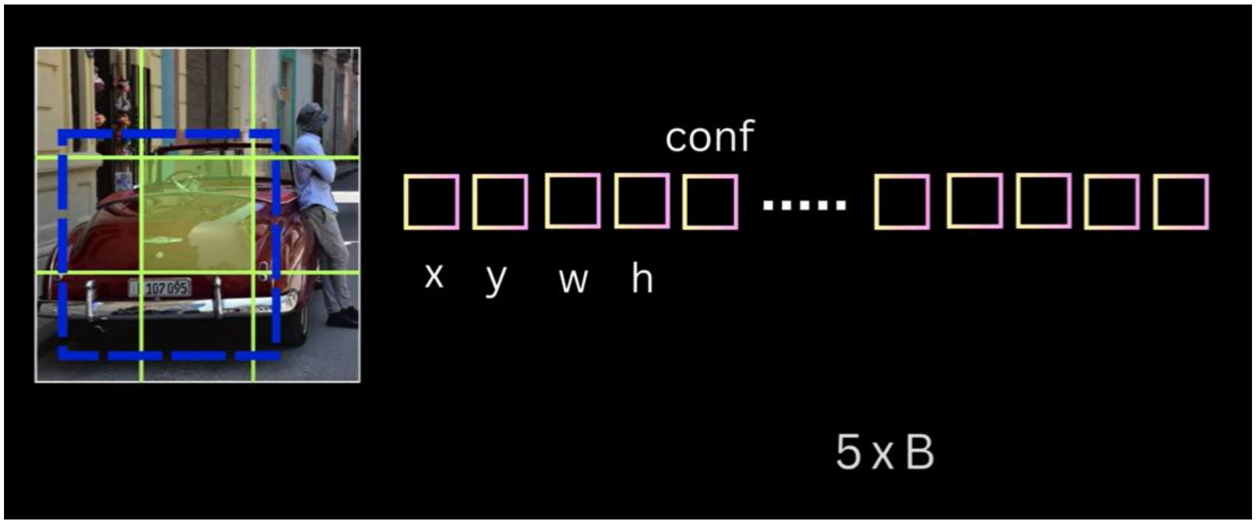
1. Width and Height of the boxes (relative to the image width and height normalized to [0, 1]. So width of 1 means that the width is the same as the width of the image)
2. Center of the box as the offset from the top left corner of the grid cell that the bounding box belongs to. The offsets are also normalized between [0, 1] relative to the width and height of the grid cell.
3. A confidence score. This attempts to capture two aspects:
    a. Confidence that the model thinks there indeed is an object within this box
    b. How accurate or a good fit that the model thinks that the predicted box is for the object contained in this box?

    The authors formulated confidence as:

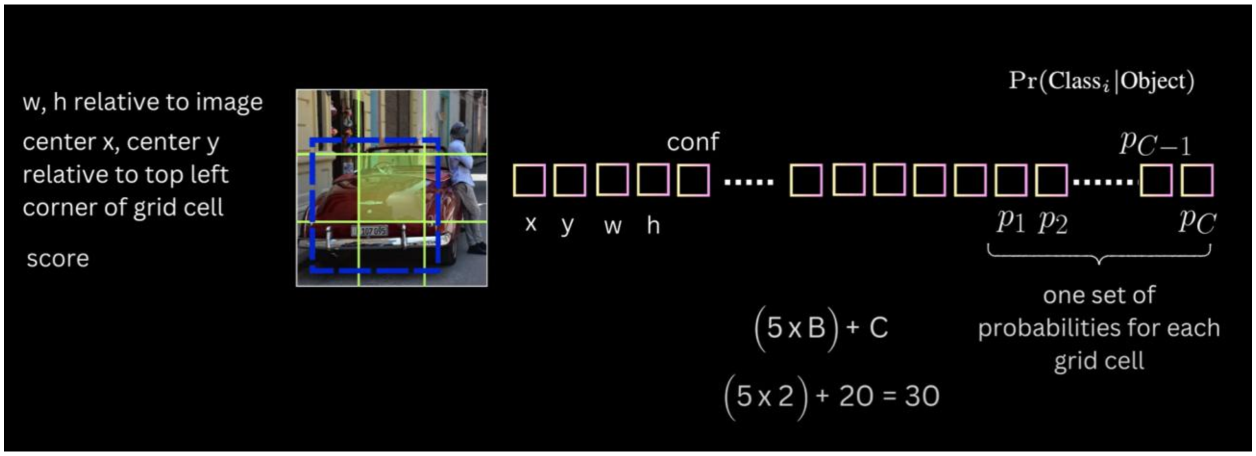
$$\Pr(\text{Object}) \times \text{IOU}_{\text{pred}}^{\text{truth}}$$



So if each grid cell predicts B bounding boxes, then each grid cell will amount to 5*B values:

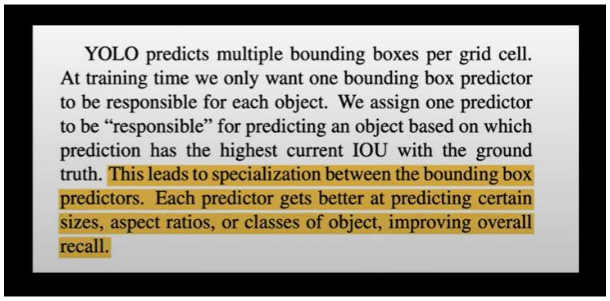


In addition to that, for each grid cell, YOLO predicts ONE set of class conditional probabilities. Specifically, these are the probabilities that the class exists in that grid cell given that there is an object in that grid cell.

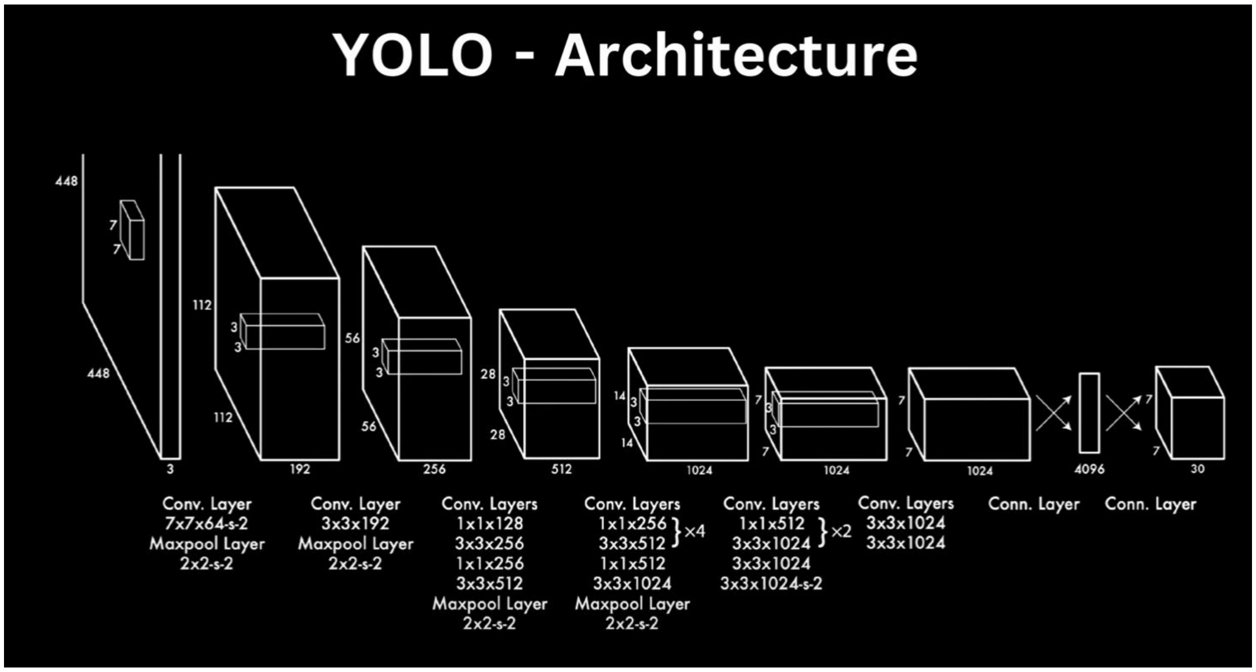

w, h relative to image
center x, center y relative to top left corner of grid cell
score
conf
x
y
w
h
$\Pr(\text{Class}_i|\text{Object})$
p_{C-1}
p_1 p_2
p_C
one set of probabilities for each grid cell
$(5 \times B) + C$
$(5 \times 2) + 20 = 30$

So if we had 20 classes (like VOC) and we predict 2 bounding boxes for each grid cell, YOLO would predict 30 values for each grid cell.

Interesting points regarding YOLO predictions:

1. Eventhough YOLO predicts multiple boxes per grid cell, it only predicts a single set of class probabilities regardless of the number of boxes predicted per cell. So each grid cell will only predict one type of object.
2. So if YOLO predicts multiple boxes per grid cell, given a single target associated with that grid cell, do we train both boxes? And if not, how to choose? Indeed the authors choose the bounding box that has the greatest IOU with the target box, this box is known as the responsible predictor box. This allows different predictors to specialize: 

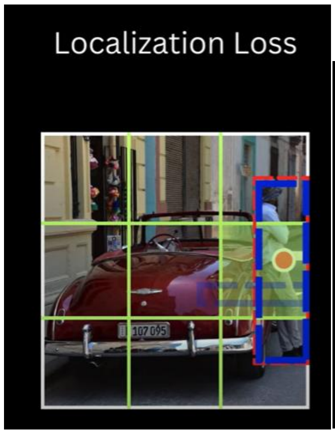
YOLO predicts multiple bounding boxes per grid cell. At training time we only want one bounding box predictor to be responsible for each object. We assign one predictor to be "responsible" for predicting an object based on which prediction has the highest current IOU with the ground truth. This leads to specialization between the bounding box predictors. Each predictor gets better at predicting certain sizes, aspect ratios, or classes of object, improving overall recall.


YOLO - Architecture
448
7
7
448
3
112
3
3
112
192
56
3
3
56
256
28
3
3
28
512
14
3
3
14
1024
7
3
3
7
1024
7
7
1024
4096
7
7
30
Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 }×4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 }×2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer

YOLO Loss

YOLO loss has 3 components.

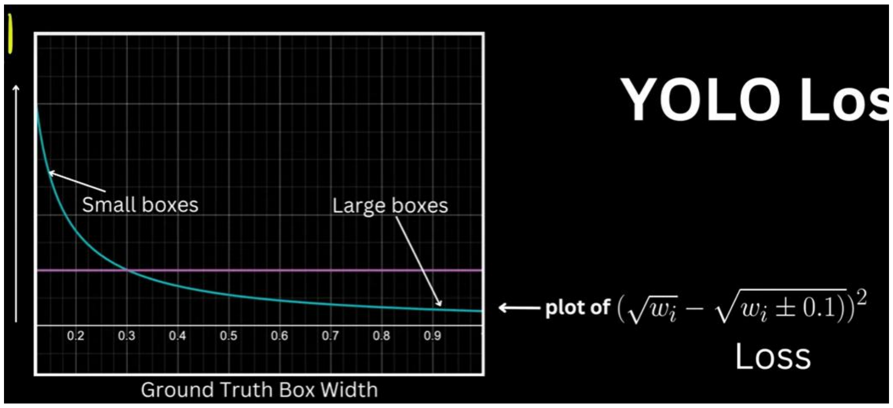
1. Localization loss – to ensure that the model predicts w,h,x_offset, y_offset of the responsible predictor box as close as possible to the target assigned to that cell.



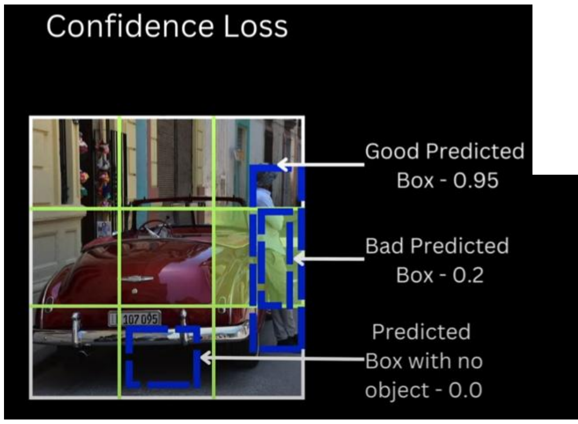
$$\left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2\right]$$

$$\left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2\right]$$

Note that we take the sqrt when regressing the width and height because you want to penalize a 0.1 error more for smaller boxes than for larger boxes.

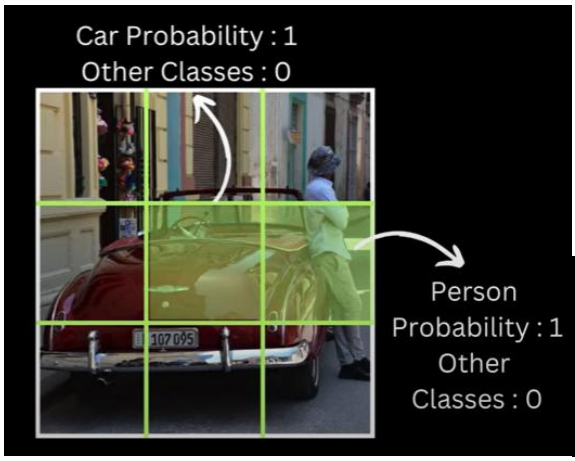


2. Confidence Loss – meant to train the model to predict correct confidence. This captures two aspects: presence of an object and how good a fit this bounding box is. So want the confidence score to be of high value if predicted box has a high overlap with the ground truth target, and even 0 if the predicted box has no target object.



$$\left(C_i - \hat{C}_i\right)^2$$

3. Classification Loss – For the cells that contain an object (because it predicts conditional probabilities), it must predict 1 for the class that is actually present.



$$\sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c)\right)^2$$

For localization loss, we only want to calculate the MSE for boxes that are responsible for some target ground truth and ignore the rest.

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2 \right]$$

The indicator function returns 1 iff the grid cell i has a target assigned and box j is the predictor box.

Similarly for confidence loss. In addition to the usual, we want to train the model to predict the confidence for cells that don't have any ground truth assigned to it as 0.

$$\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{\text{obj}}\left(C_i-\hat{C}_i\right)^2$$

$$\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{\text{noobj}}\left(C_i-\hat{C}_i\right)^2$$

Now for classification loss, this must only be aggregated for grid cells that was assigned some target:

$$\sum_{i=0}^{S^2}\mathbb{1}_{i}^{\text{obj}}\sum_{c\in\text{classes}}\left(p_i(c)-\hat{p}_i(c)\right)^2$$

Final loss:

$$
\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]
$$

$$+$$

$$
\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]
$$

$$+$$

$$
\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2
$$

$$+$$

$$
\lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2
$$

$$+$$

$$
\sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2
$$

$$\lambda_{\text{coord}} \longrightarrow 5$$

$$\lambda_{\text{noobj}} \longrightarrow 0.5$$

But note that the authors included weighting factors to weight localization loss more because otherwise (in a usual image only few grid cells have ground truths assigned to it), the gradients from confidence score of boxes with no objects to overpower those that contain objects.

During testing the authors predict the class specific confidence score by multiplying:

$$\begin{tikzcd}
\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 & & \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \\
\Pr(\text{Object}) \times \text{IOU}_{\text{pred}}^{\text{truth}} \arrow[dr] & & \Pr(\text{Class}_i | \text{Object}) \arrow[dl] \\
& \times &
\end{tikzcd}$$

$$
\Pr(\text{Class}_i) \times \text{IOU}_{\text{pred}}^{\text{truth}}
$$

Which effectively encodes (at test time) the probability of that class present in the box and how well the box fits that object.

YOLOv3 points

The DarkNet doesn't have any pooling, instead uses convolution to downsample without loss of information. Then they use residual connections to keep finegrained features.

YOLOv3 uses 3 anchor boxes for each grid cell (unlike 5 for YOLOv2)

First Question

Firstly, During training, each target object is assigned to the cell that contains the object's center.

During training, each grid cell predicts B bounding boxes. And then YOLO learns to predict the bounding boxes as close as possible to the bounding boxes assigned to that cell—if any.

Unlike YOLOv1, YOLOv3 predicts bounding boxes at three different scales of feature maps (this will be discussed in detail in the next question). At each scale of the feature map, the feature map is divided into SxS many grid cells (S is different for each scale of feature map). During training, each ground-truth target object is assigned to the grid cell that contains the ground truth object's center. Next, for each grid cell at that feature map scale, YOLOv3 has 3 pre-calculated anchor boxes (the details of how the anchor boxes are calculated are explained in the next question). Then, YOLOv3 (unlike YOLOv1 which predicts B many bounding boxes for each grid cell naively), instead predicts parameters to morph a strategically selected anchor box to get as close as possible to the ground truth bounding box.

For YOLOV3, given a 416x416 image, unlike YOLOv1, YOLOv3 predicts bounding boxes at three different scales of feature maps. And at each scale of feature map, the map is divided into SxS many grid cells. For the coarse-grained feature map (when the stride is 32) which is responsible for detecting larger objects, we have 13x13 many grid cells. For the medium grained feature map (when the stride is 16) that is responsible for detecting medium sized objects, we have 26x26 many grid cells. For the fine-grained feature map (when the stride is 8) that is responsible for detecting small objects, we have 52x52 many grid cells.

For each scale of feature map, for each grid cell

For multiscale prediction, it uses two factors hand in hand, one is predictions at different scaled feature map and tailored anchor boxes (talk about how anchor boxes are calculated etc).