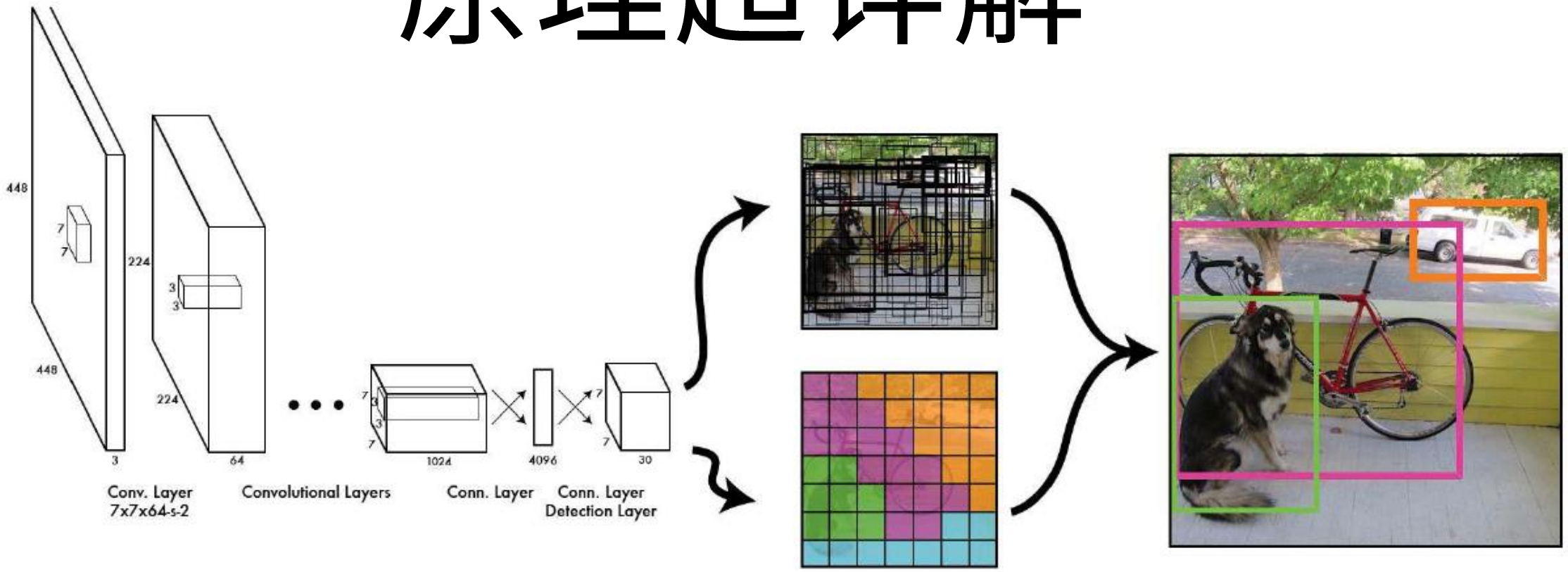
目标检测 YOLO V1

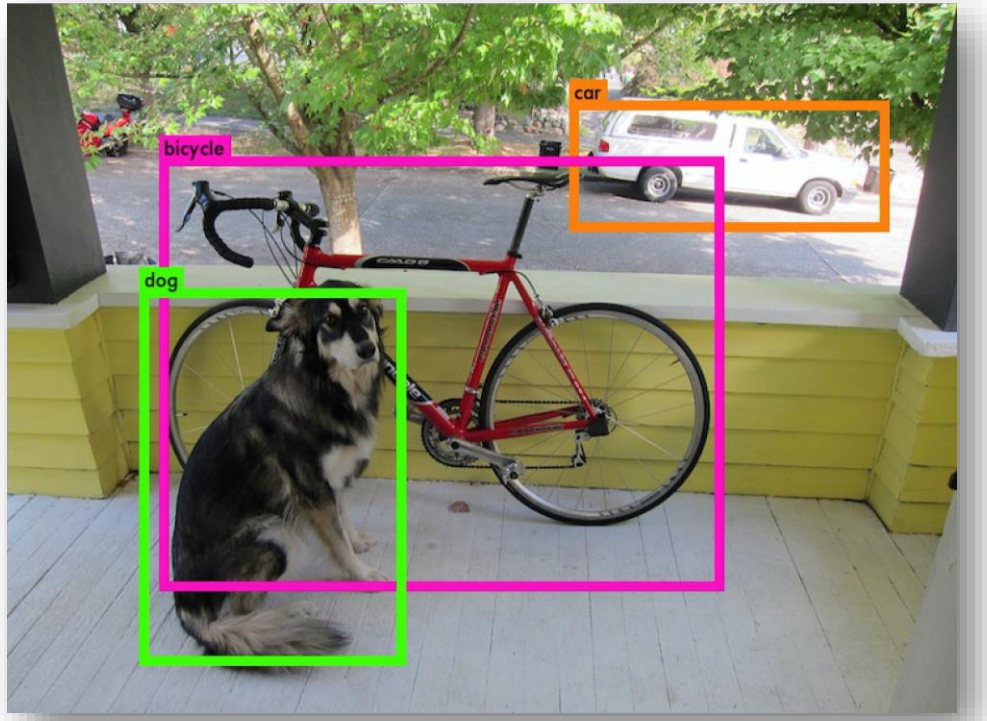
原理超详解



Object Detection

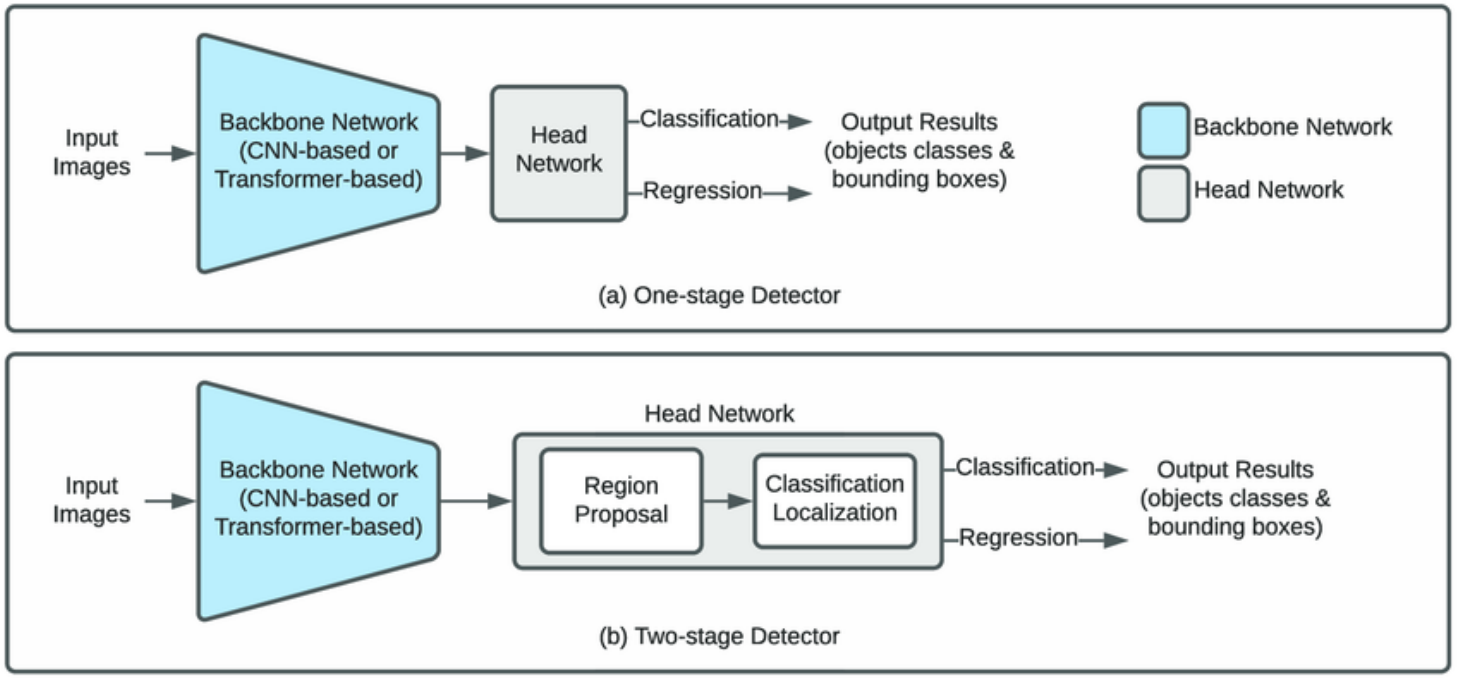
- **Object Detection** is a computer vision task in which the goal is to detect and locate objects of interest in an image or video.
  - Identify the position and boundaries
  - Classify the objects into different categories.

- The state-of-the-art methods can be categorized into two main types:
  - One-stage methods
  - Two stage-methods

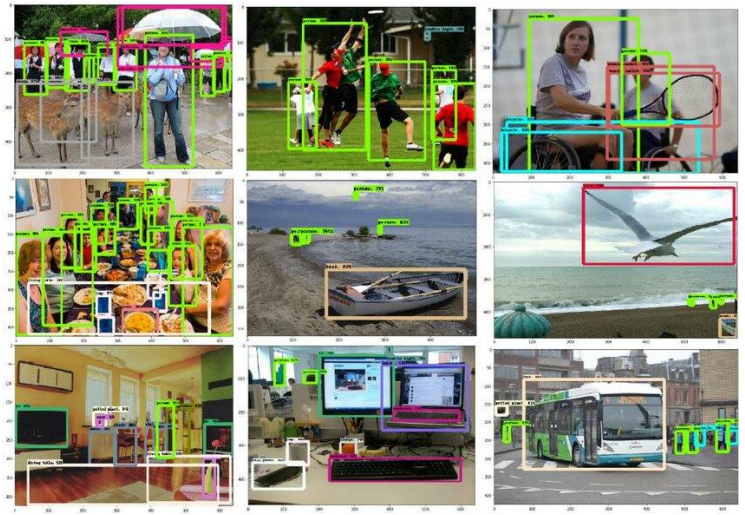


One-stage Methods & Two-stage Methods

- **One-stage methods** prioritize inference speed, and example models include YOLO, SSD and RetinaNet.
- **Two-stage methods** prioritize detection accuracy, and example models include Faster R-CNN, Mask R-CNN and Cascade R-CNN.

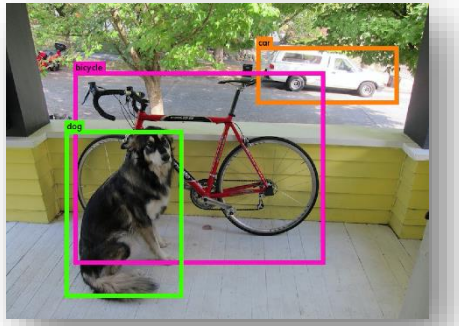


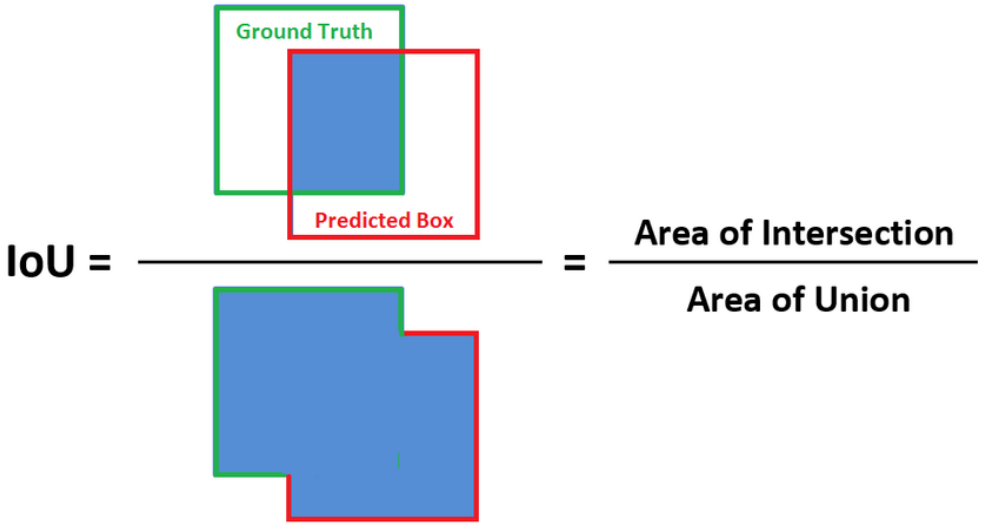
- The most popular benchmark is the MSCOCO dataset.



Bounding Boxes

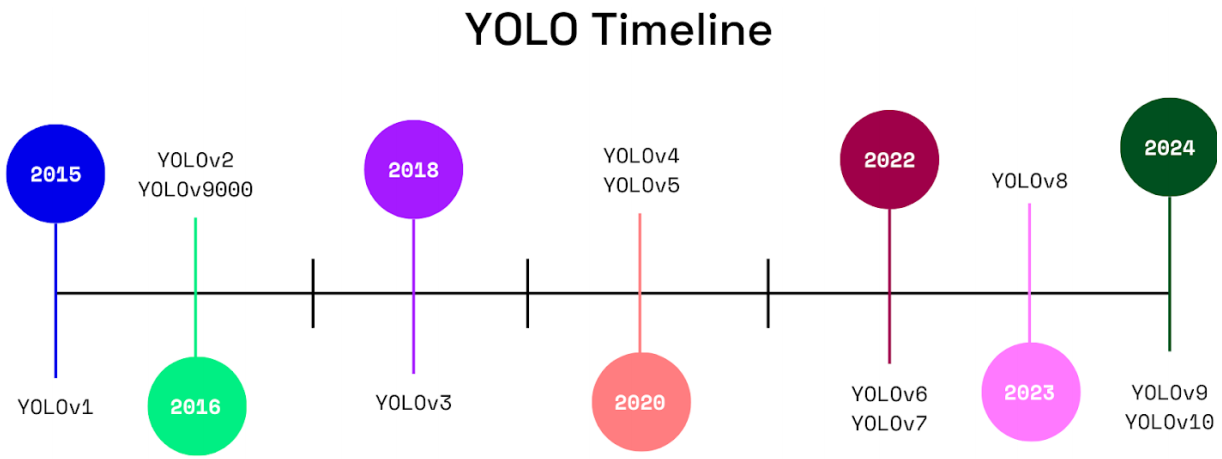
- **Bounding boxes** are rectangular region labels used for computer vision (CV) tasks.
- In supervised machine learning (ML), an object detection model uses bounding box labels to learn about the contents of an image.





- For each bounding box, we measure overlap between the predicted bounding box and the ground truth bounding box. This is measured by **intersection over union (IoU).**

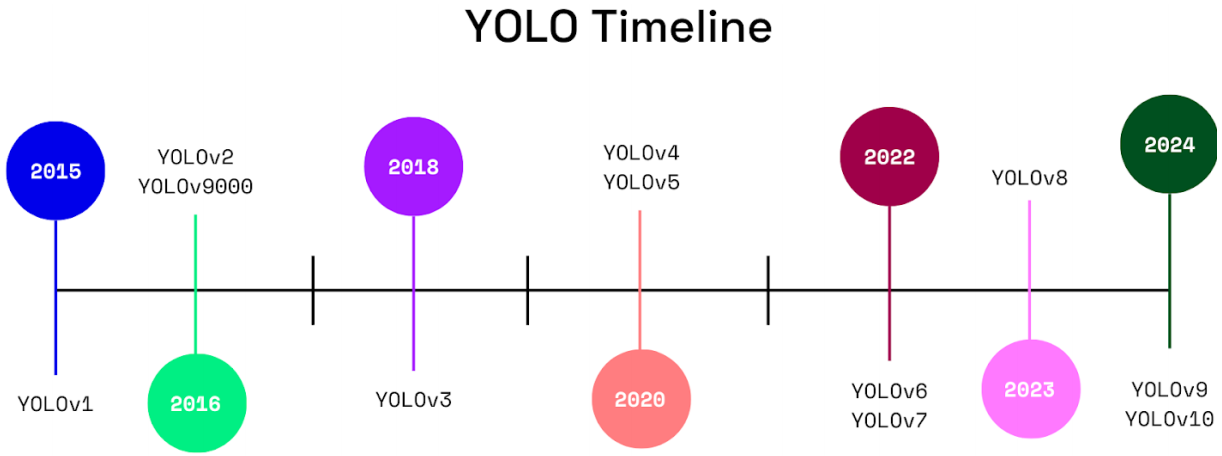
What is YOLO?



- YOLO (You Only Look Once) is a state-of-the-art real-time object detection method. Unlike traditional methods, YOLO applies detection to the whole image at once, making it faster and more efficient.

- Advantages:
  - High-speed performance, ideal for real-time applications.
  - Improved accuracy over time with each version.
- Limitations: Less accurate for small objects or objects that are close together due to the grid system.

What is YOLO?

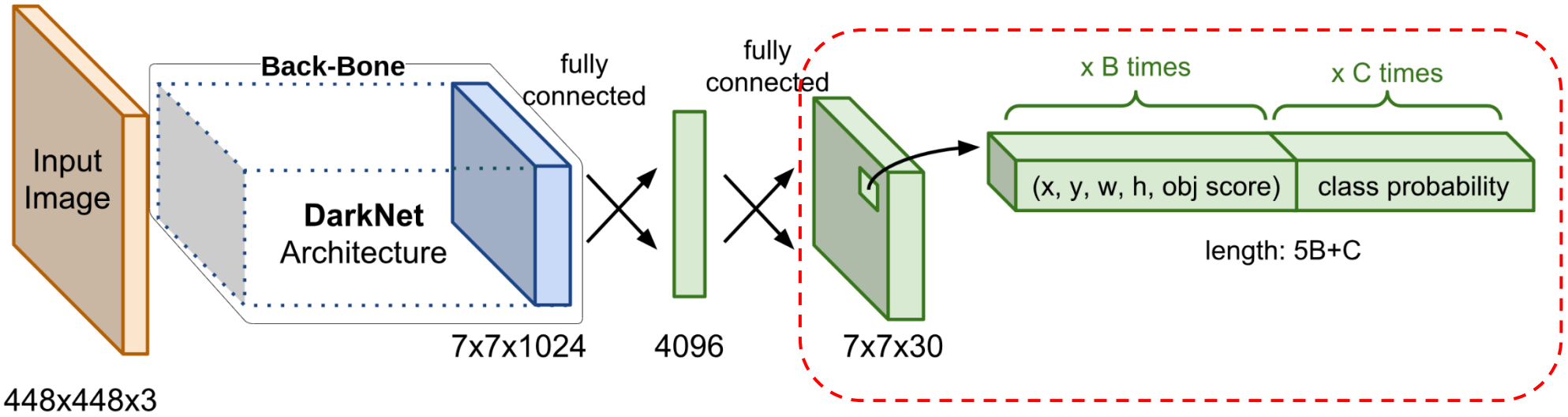


Joseph Redmon

Alexey Bochkovskiy

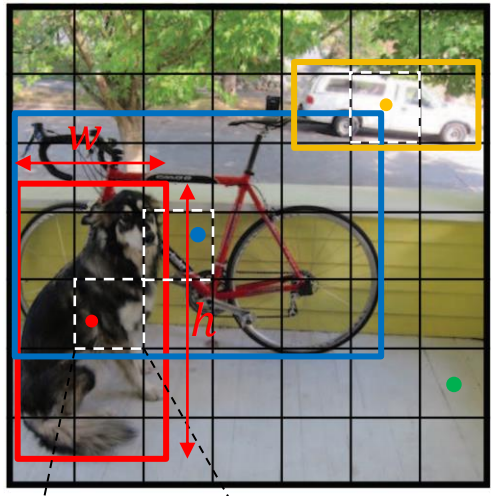
- Joseph Redmon: The creator of YOLO, Joseph Redmon, first introduced the YOLO framework in 2015. He worked alongside Ali Farhadi to develop YOLOv2 and YOLOv3. However, in 2020, Redmon decided to stop his computer vision research due to ethical concerns related to military applications of AI.
- Alexey Bochkovskiy: In 2020, Alexey Bochkovskiy and his team took over Redmon's work and published YOLOv4, emphasizing optimizing the network hyperparameters and an IOU-based loss function.

[YOLO V1] Network Design



- The architecture of YOLO v1 is a simple convolutional network with Maxpool layers and LeakyReLU activation functions followed by a linear layer and the prediction tensor.
- **Training Phase**: How to establish a connection between the 7x7x30 output and the labels of the input image to compute the loss?
- **Inference Phase**: How to obtain bounding boxes and class probabilities from the 7x7x30 output?

[YOLO V1] Labeling the data



S × S grid on input

x	0.25	0.75	0.50	-
y	0.60	0.30	0.50	-
w	0.34	0.78	0.41	-
h	0.57	0.50	0.17	-
Objectness	1	1	1	0
Dog	1	0	0	0
Bicycle	0	1	0	0
Car	0	0	1	0
Desk	0	0	0	0
…				

- YOLO divides the input image into a 7x7 grid.
- **The label vector includes:**

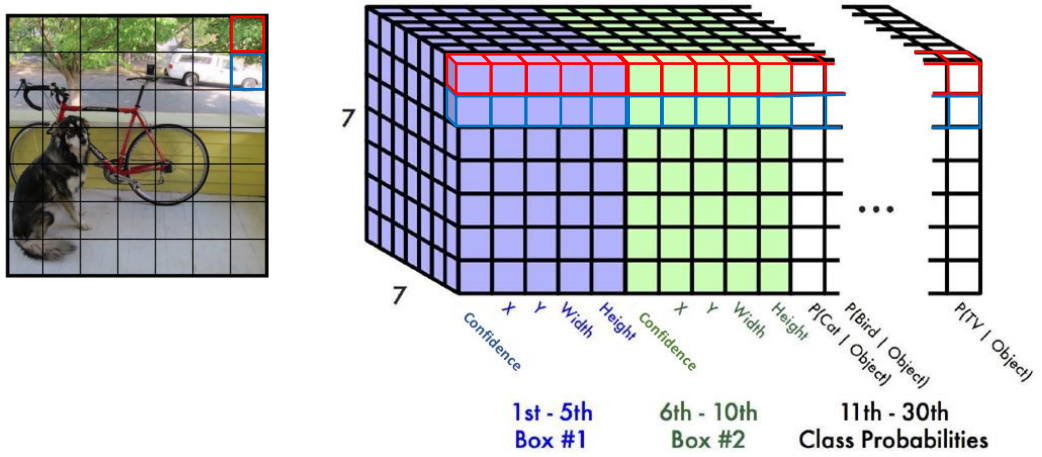
  $[x, y, w, h, \text{Objectness}, \text{Class1}, \text{Class2}, \dots]$

  - Objectness: Label = 1 if the grid cell contains a center. Label = 0 otherwise.
  - x, y: Center coordinates relative to the grid cell.
  - w, h: Width and height relative to the full image.
  - Class probabilities: For each class.
- If multiple objects are in the same cell, only one class is assigned for training.

[YOLO V1] The Prediction Tensor

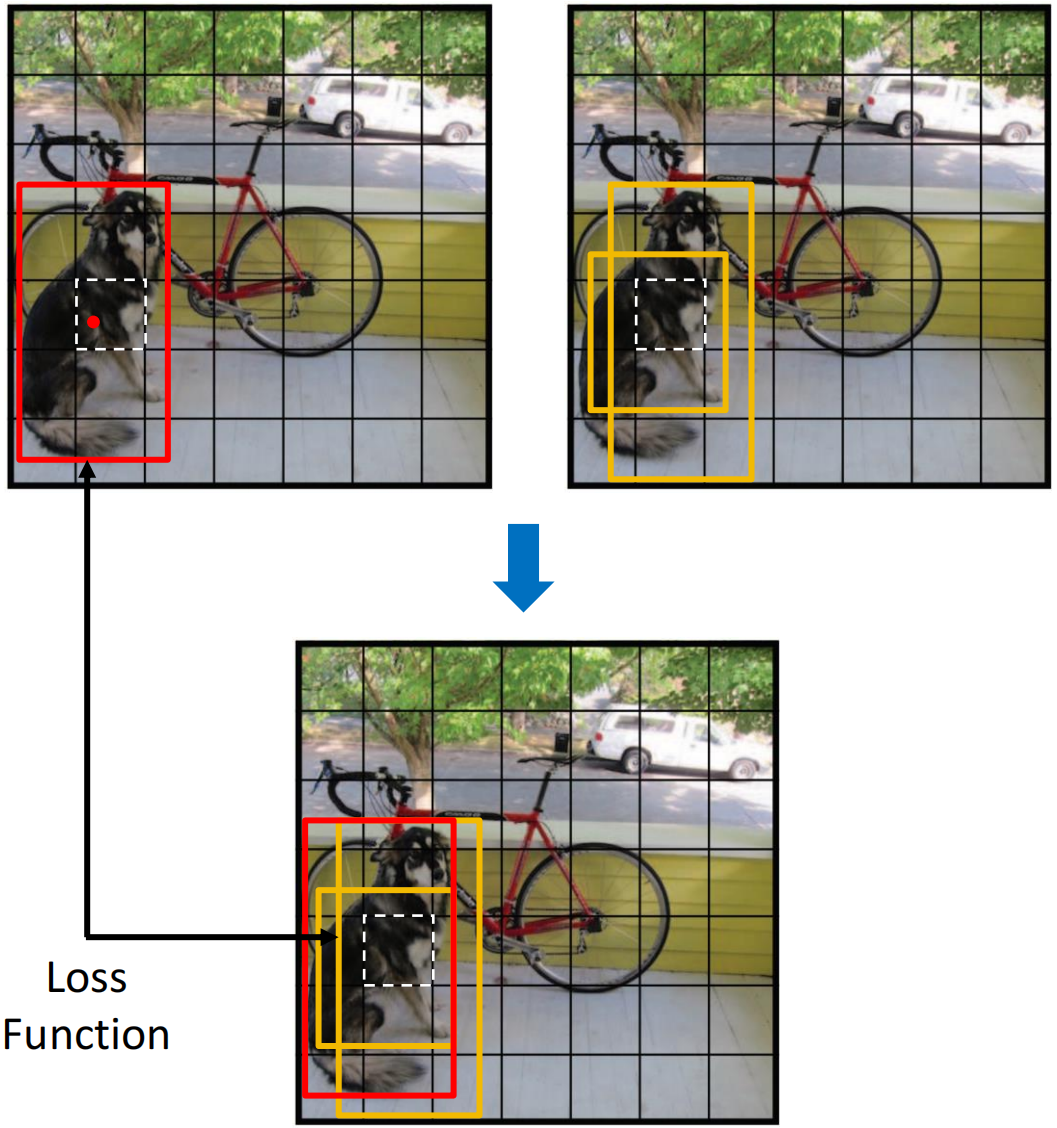
- YOLO V1 formulates object detection as a regression problem. The image is divided into an $S \times S$ grid, where each grid cell predicts bounding boxes and class probabilities.
- Each grid cell predicts B bounding boxes(for V1, we have $B = 2$), and each bounding box includes:
  - **Position Parameters**: x, y, w, h, representing the center coordinates and dimensions (width and height) of the bounding box.
  - **Confidence Score**: $\Pr(\text{Object}) \times \text{IOU}|_{\text{pred}}^{\text{truth}}$

  If no object exists in that cell, the confidence scores should be zero. Otherwise we want the confidence score to equal the IOU between the predicted box and the ground truth.
- Each grid cell also predicts a set of class probabilities $\Pr(\text{Class}_i \mid \text{Object})$, shared across all B bounding boxes.



$$S \times S \times (5 \times B + C) = 7 \times 7 \times (5 \times 2 + 20) = 1470$$

[YOLO V1] Training Phase



- The input image is divided into an S×S grid.
- Each object's center point determines which grid cell is responsible for predicting the object's bounding box.
- Each grid cell predicts two bounding boxes, but only the one with the highest IoU with the Ground Truth is selected.
- If a grid cell does not contain the center of an object, it is treated as a background grid and contributes to confidence loss but not localization or classification loss.

[YOLO V1] Loss Function

Bounding box coordinate regression

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

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

Bounding box score prediction

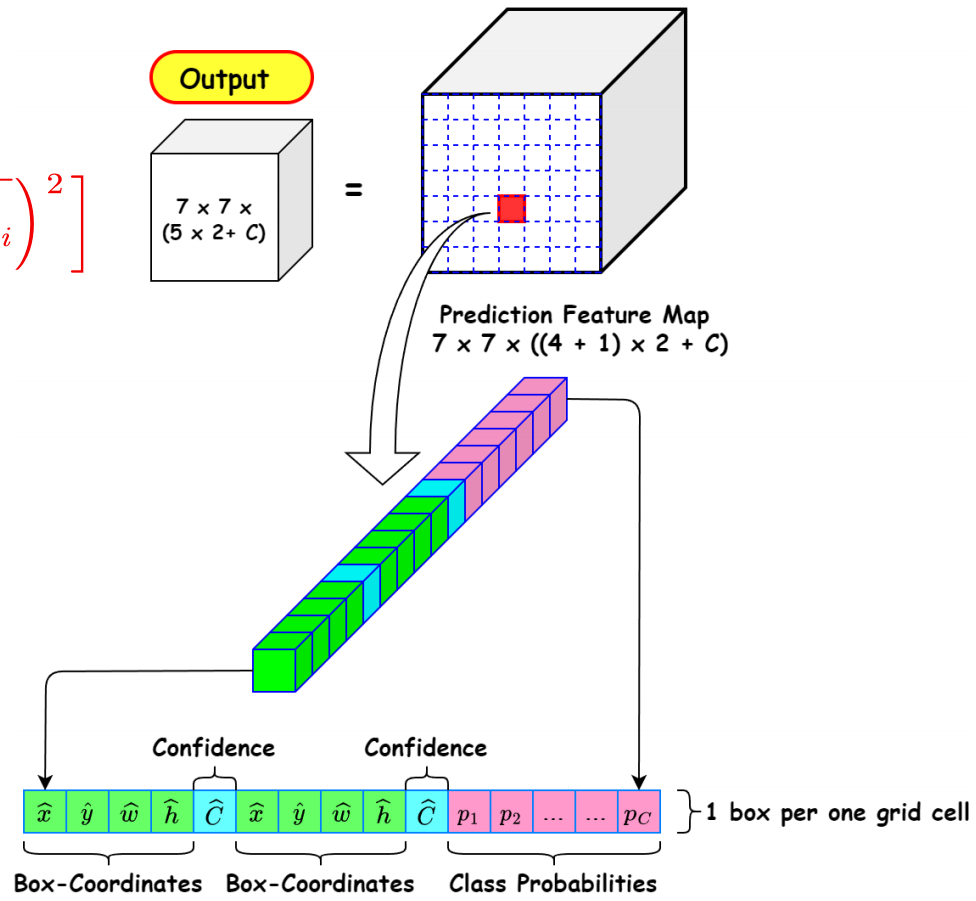
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i\right)^2$$

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

Classscore prediction

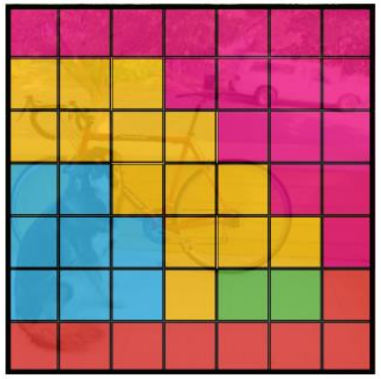
$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{ classes}} \left(p_i(c) - \hat{p}_i(c)\right)^2$$

where $\mathbb{1}_i^{\text{obj}}$ denotes if object appears in cell i and $\mathbb{1}_{ij}^{\text{obj}}$ denotes that the jth bounding box predictor in cell i is "responsible" for that prediction.

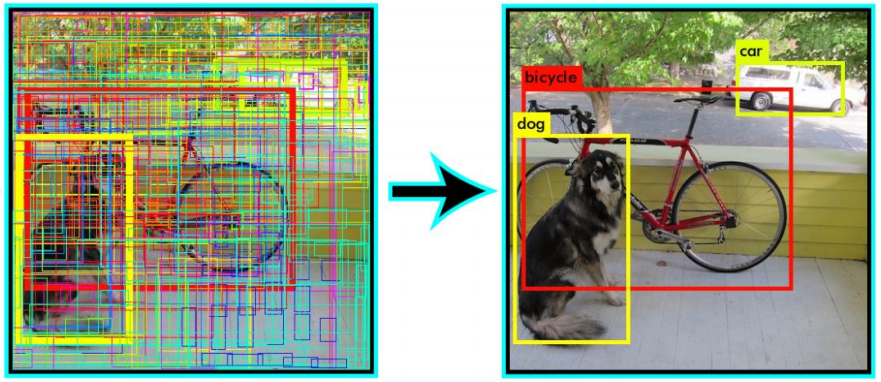


[YOLO V1] Inference Phase

Bounding boxes confidence



Class probability map



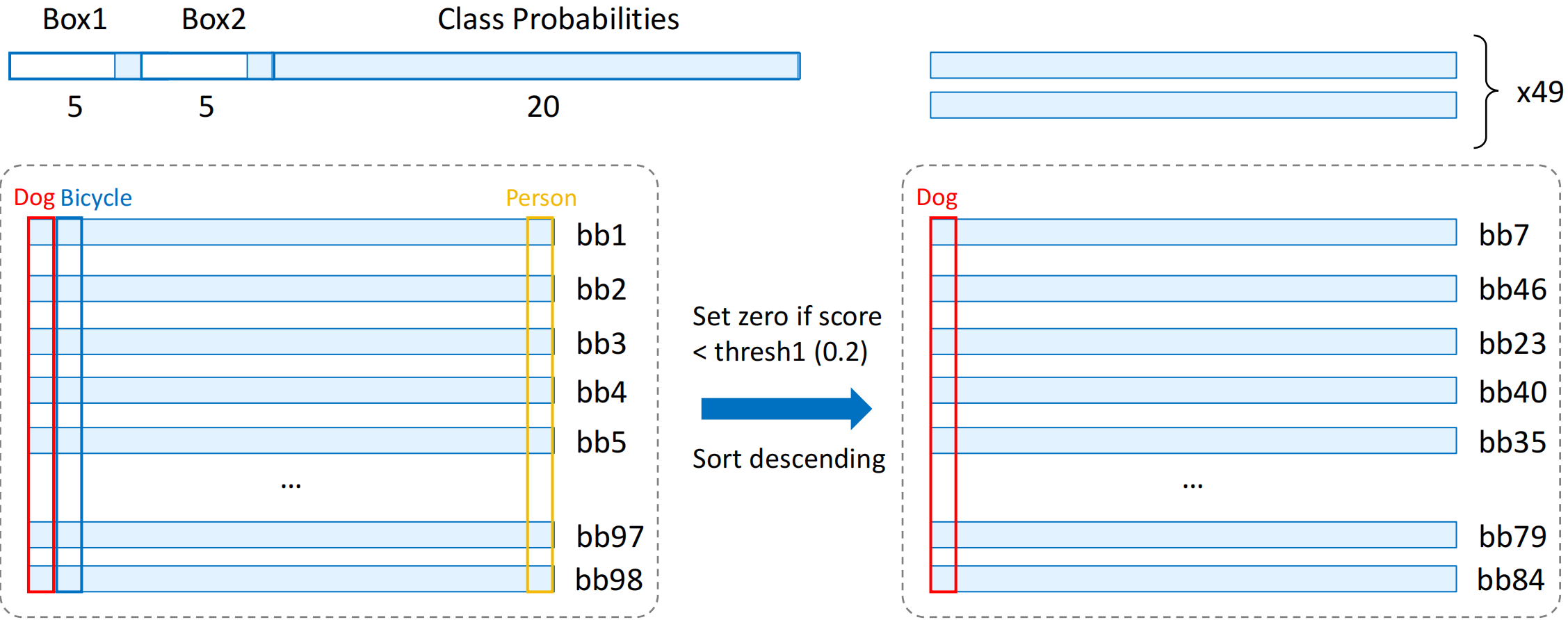
- At test time we multiply the conditional class probabilities and the individual box confidence predictions,

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

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

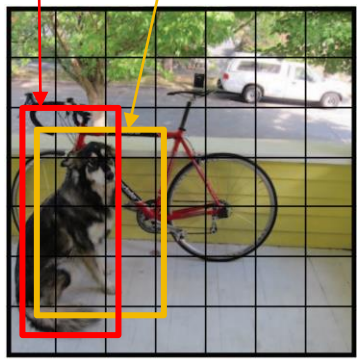
which gives us class-specific confidence scores for each box. These scores encode both the probability of that class appearing in the box and how well the predicted box fits the object.

[YOLO V1] Non-Maximum Suppression (NMS)



[YOLO V1] Non-Maximum Suppression (NMS)

	bb7	bb46	bb40	bb23	bb35	bb79	...	bb84
Dog	**0.5**	0.3	0.2	0.1			...	0

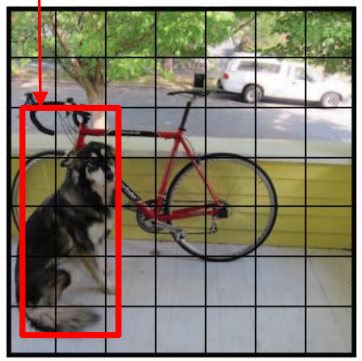


Get bbox with max score.

If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

[YOLO V1] Non-Maximum Suppression (NMS)

	bb7	bb46	bb40	bb23	bb35	bb79		bb84
Dog	**0.5**	0	0.2	0.1			...	0

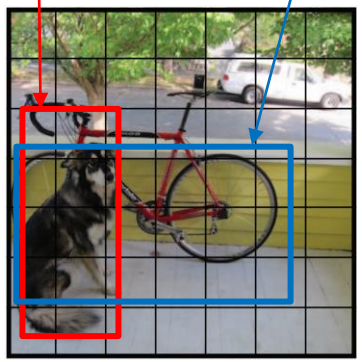


Get bbox with max score.

If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

[YOLO V1] Non-Maximum Suppression (NMS)

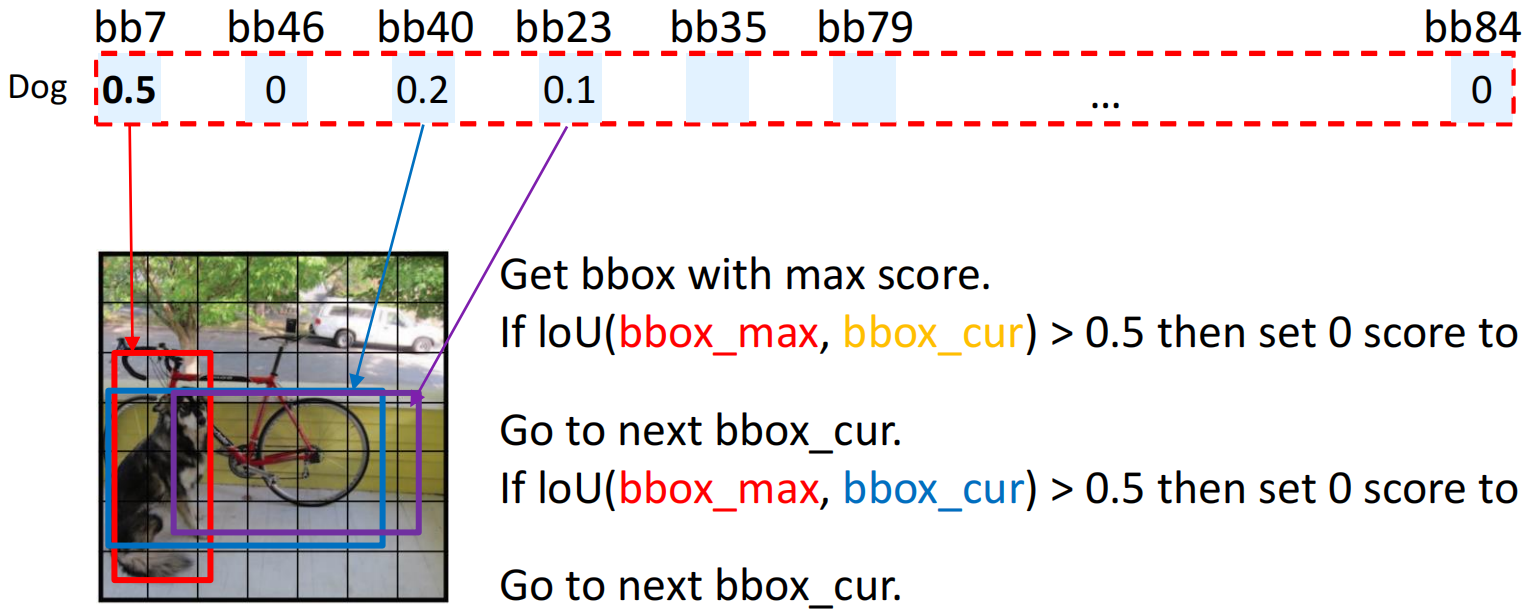
	bb7	bb46	bb40	bb23	bb35	bb79		bb84
Dog	**0.5**	0	0.2	0.1			...	0



Get bbox with max score.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

Go to next bbox_cur.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

[YOLO V1] Non-Maximum Suppression (NMS)



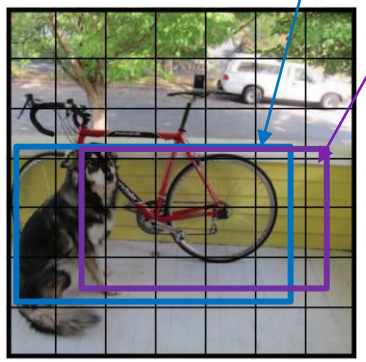
Get bbox with max score.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

Go to next bbox_cur.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

Go to next bbox_cur.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.
Do this procedure for other "bbox_cur".

[YOLO V1] Non-Maximum Suppression (NMS)

	bb7	bb46	bb40	bb23	bb35	bb79		bb84
Dog	0.5	0	**0.2**	0.1			...	0



Get bbox with max score.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

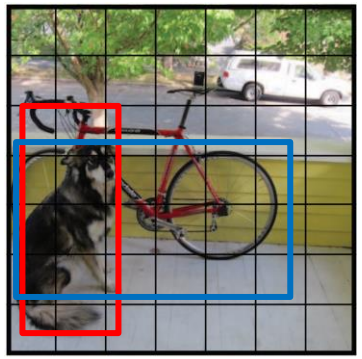
Go to next bbox_cur.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

Go to next bbox_cur.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.
Do this procedure for other "bbox_cur".

Go to next bbox with big score.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

[YOLO V1] Non-Maximum Suppression (NMS)

	bb7	bb46	bb40	bb23	bb35	bb79		bb84
Dog	0.5	0	0.2	0	0	0	...	0



After comparison almost all pairs of bboxes the only two bboxes left with non-zero class score value.

Get bbox with max score.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

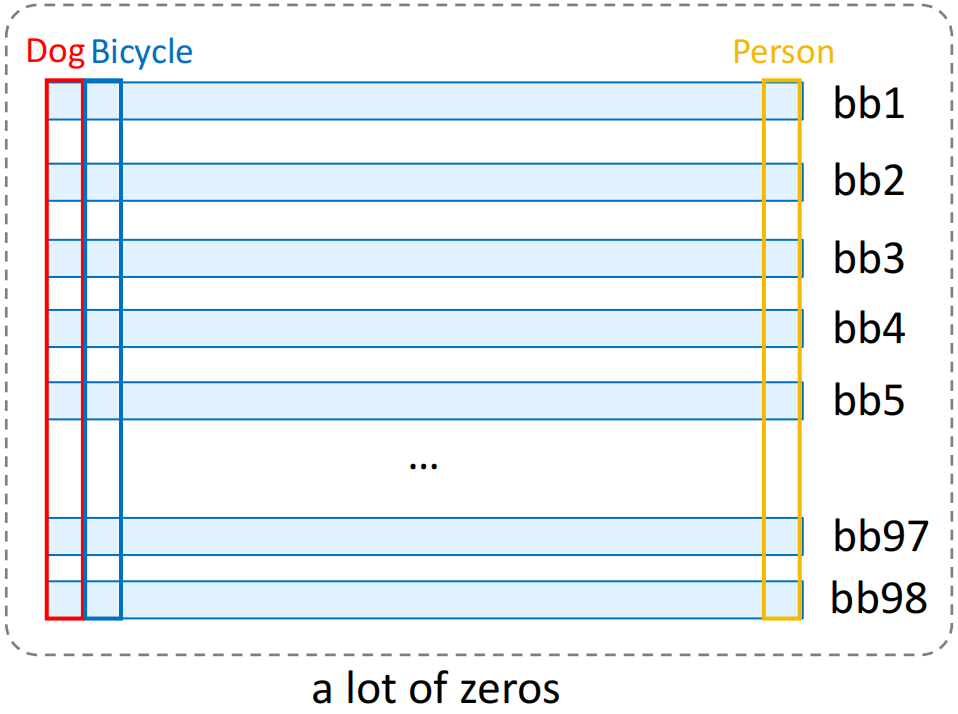
Go to next bbox_cur.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

Go to next bbox_cur.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.
Do this procedure for other "bbox_cur".

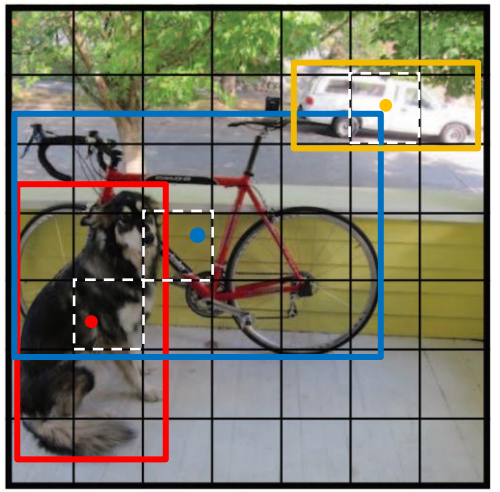
Go to next bbox with big score.
If IoU(bbox_max, bbox_cur) > 0.5 then set 0 score to bbox_cur.

Do this procedure for other “bbox_max” and for other corresponding "bbox_cur".

[YOLO V1] Non-Maximum Suppression (NMS)

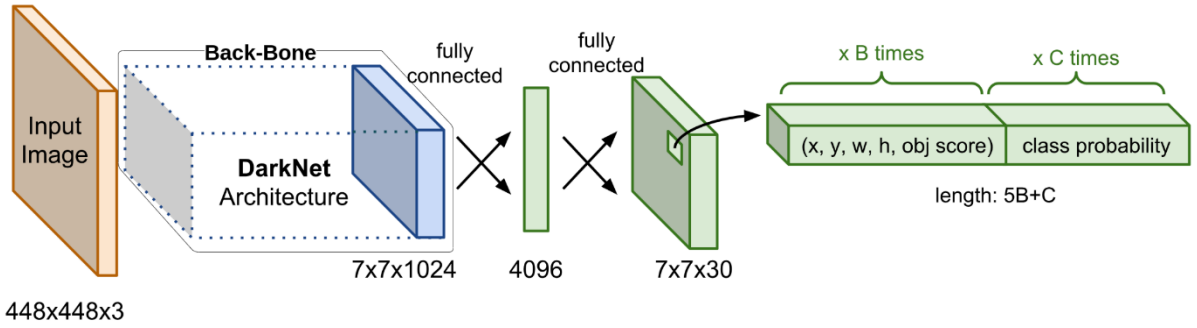


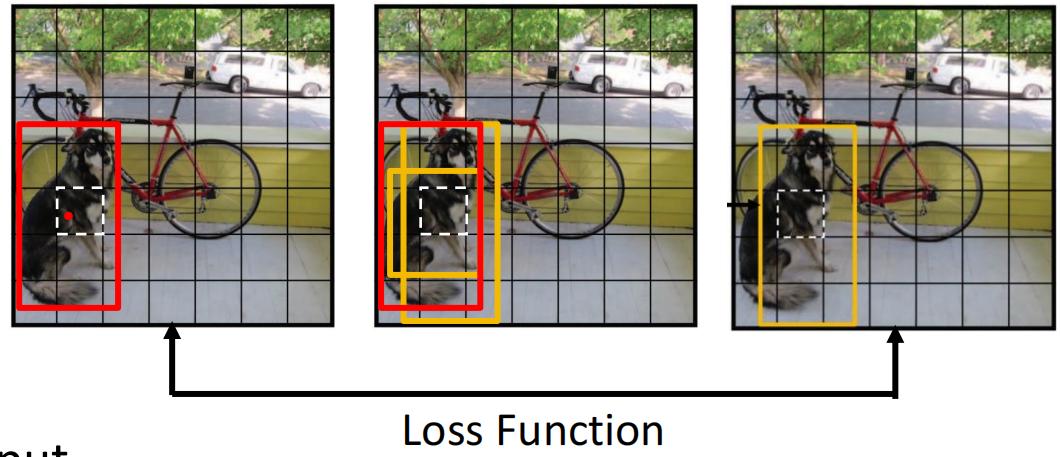
Select bboxes to draw by class score values



S × S grid on input

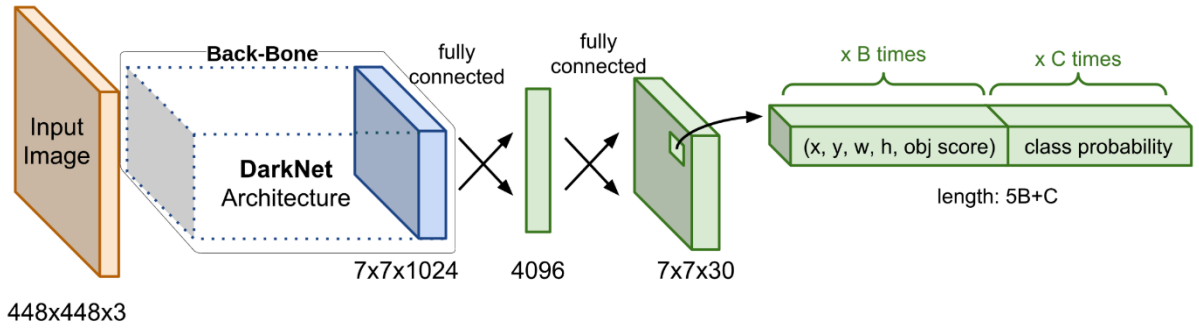
[YOLO V1] Summary

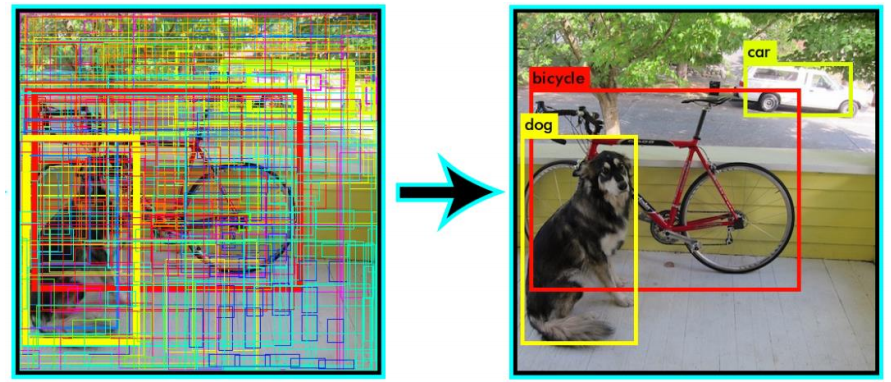




- **Input Image**: 448x448 passed through CNN → 7x7x30 output.
- **Training Phase**：
  - Predict 98 bounding boxes (7x7x2).
  - The ground truth center falls within a grid cell, and the grid cell fits the bounding box.
  - If a grid cell does not contain the object center, it is treated as background and only contributes to confidence loss.

[YOLO V1] Summary





- **Input Image**: 448x448 passed through CNN → 7x7x30 output.
- **Inference Phase**：
  - Predict 98 bounding boxes (7x7x2).
  - Apply NMS to remove redundant bounding boxes based on the Intersection over Union (IoU) threshold.
  - After applying NMS, select the final bounding boxes based on the highest class scores.