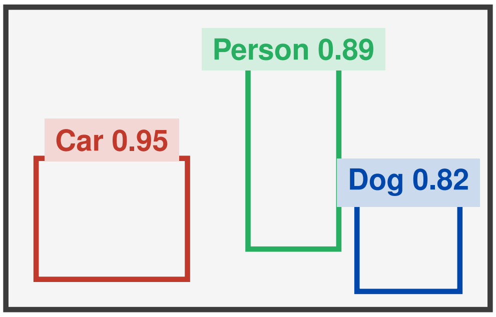
Deep Learning for Perception

Lecture 06: Object Detection & YOLO



Object Detection: Localize + Classify

Topics Covered in This Lecture:

- Object Detection vs Classification
- Semantic Segmentation
- IoU (Intersection over Union)
- Traditional Detection Pipeline
- YOLO Architecture
- YOLO Grid System
- Bounding Box Predictions
- Non-Max Suppression (NMS)
- YOLO Loss Function
- YOLO Versions (v1, v2, v3+)

Instructor: Aqsa Younas

Department of Computer Science
FAST-NU, CFD Campus

Contents

1 Computer Vision Tasks Overview 2

1.1 Classification vs Detection vs Segmentation 2

1.2 Bounding Box Representation 3

2 Intersection over Union (IoU) 4

3 Traditional Object Detection Pipeline 6

4 YOLO: You Only Look Once 7

4.1 YOLO Key Insight 7

4.2 YOLO Grid System 7

4.3 YOLO Output Tensor 8

4.4 Bounding Box Predictions 9

5 Non-Max Suppression (NMS) 11

6 YOLO Loss Function 13

7 YOLO Architecture and Versions 15

7.1 YOLO v1 Architecture 15

7.2 YOLO Versions Comparison 15

7.3 YOLO Advantages and Limitations 15

8 Summary 17

9 Glossary 18

Advance Organizer — What You'll Learn

Learning Objectives: By the end of this lecture, you will be able to:

1. **Differentiate** between classification, detection, and segmentation
2. **Calculate** Intersection over Union (IoU) for bounding boxes
3. **Explain** the YOLO architecture and its advantages
4. **Compute** YOLO output tensor dimensions
5. **Apply** Non-Max Suppression to filter detections
6. **Calculate** YOLO loss function components
7. **Solve** numerical problems on object detection

Prior Knowledge Required:

- Convolutional Neural Networks
- Classification networks
- Loss functions

1 Computer Vision Tasks Overview

Why It Matters

Understanding the different computer vision tasks helps you choose the right approach for your problem. Object detection is crucial for autonomous vehicles, robotics, surveillance, and many real-world applications.

1.1 Classification vs Detection vs Segmentation

Definition

Image Classification: Assign a single label to the entire image.

- Input: Image
- Output: Class label (e.g., "cat")

Object Detection: Locate AND classify multiple objects in an image.

- Input: Image
- Output: Bounding boxes + class labels + confidence scores

Semantic Segmentation: Classify every pixel in the image.

- Input: Image
- Output: Per-pixel class labels

Instance Segmentation: Segment individual object instances.

- Input: Image
- Output: Per-pixel labels distinguishing different instances

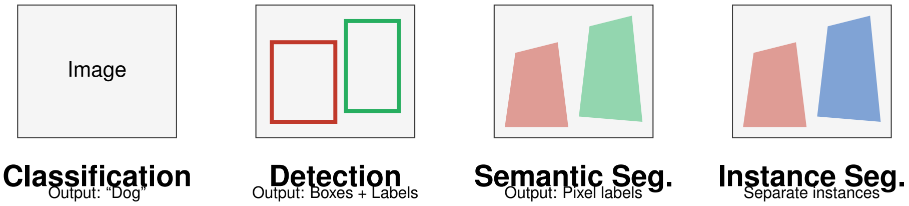


Figure 1: Comparison of computer vision tasks

1.2 Bounding Box Representation

Definition

A **bounding box** is a rectangle that tightly encloses an object.
Two common formats:
1. Corner Format: (x_1, y_1, x_2, y_2)

- (x_1, y_1) = Top-left corner
- (x_2, y_2) = Bottom-right corner

2. Center Format: (c_x, c_y, w, h)

- (c_x, c_y) = Center coordinates
- w, h = Width and height

Key Formula

Conversion Formulas:
Corner → Center:

$$c_x = \frac{x_1 + x_2}{2}, \quad c_y = \frac{y_1 + y_2}{2}, \quad w = x_2 - x_1, \quad h = y_2 - y_1$$

Center → Corner:

$$x_1 = c_x - \frac{w}{2}, \quad y_1 = c_y - \frac{h}{2}, \quad x_2 = c_x + \frac{w}{2}, \quad y_2 = c_y + \frac{h}{2}$$

2 Intersection over Union (IoU)

Why It Matters

IoU is the standard metric for evaluating how well a predicted bounding box matches the ground truth. It's used in training (loss functions), evaluation (mAP calculation), and inference (NMS).

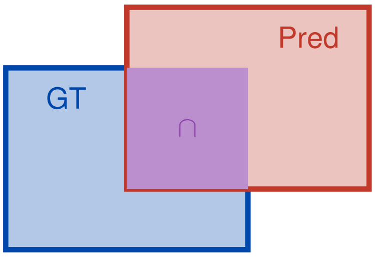
Definition

Intersection over Union (IoU), also called Jaccard Index, measures the overlap between two bounding boxes:

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

Range: $0 \leq \text{IoU} \leq 1$

- IoU = 0: No overlap
- IoU = 1: Perfect overlap
- IoU $\geq$ 0.5: Typically considered a "good" detection



$$\text{IoU} = \frac{\text{Purple (Intersection)}}{\text{Blue + Red - Purple (Union)}}$$

Figure 2: IoU: Intersection (purple) divided by Union (total colored area)

Key Formula

IoU Calculation Steps:
Given two boxes: $B_1 = [x_1^{(1)}, y_1^{(1)}, x_2^{(1)}, y_2^{(1)}]$ and $B_2 = [x_1^{(2)}, y_1^{(2)}, x_2^{(2)}, y_2^{(2)}]$
Step 1: Find intersection coordinates

$$x_1^{(\cap)} = \max(x_1^{(1)}, x_1^{(2)}) \qquad x_2^{(\cap)} = \min(x_2^{(1)}, x_2^{(2)})$$
$$y_1^{(\cap)} = \max(y_1^{(1)}, y_1^{(2)}) \qquad y_2^{(\cap)} = \min(y_2^{(1)}, y_2^{(2)})$$

Step 2: Calculate intersection area

$$A_\cap = \max(0, x_2^{(\cap)} - x_1^{(\cap)}) \times \max(0, y_2^{(\cap)} - y_1^{(\cap)})$$

Step 3: Calculate union area

$$A_\cup = A_1 + A_2 - A_\cap$$

Step 4: Calculate IoU

$$\boxed{\text{IoU} = \frac{A_\cap}{A_\cup}}$$

Solved Example 1: IoU Calculation

Given Data:

- Predicted box: $[100, 100, 200, 200]$ (corner format)
- Ground truth box: $[120, 120, 220, 220]$

Task: Calculate the IoU between these boxes.

Solution:

Step 1: Find intersection coordinates

$$x_1^{(\cap)} = \max(100, 120) = 120 \qquad x_2^{(\cap)} = \min(200, 220) = 200$$
$$y_1^{(\cap)} = \max(100, 120) = 120 \qquad y_2^{(\cap)} = \min(200, 220) = 200$$

Step 2: Calculate intersection area

$$A_\cap = (200 - 120) \times (200 - 120) = 80 \times 80 = 6400$$

Step 3: Calculate individual box areas

$$A_1 = (200 - 100) \times (200 - 100) = 100 \times 100 = 10000$$
$$A_2 = (220 - 120) \times (220 - 120) = 100 \times 100 = 10000$$

Step 4: Calculate union area

$$A_\cup = 10000 + 10000 - 6400 = 13600$$

Step 5: Calculate IoU

$$\text{IoU} = \frac{6400}{13600} = 0.4706$$

Answer: IoU = **0.4706** (or 47.06%)
Since IoU $<$ 0.5, this would typically be considered a marginal detection. Many systems use IoU $\geq$ 0.5 as the threshold for a correct detection.

3 Traditional Object Detection Pipeline

Why It Matters

Understanding the traditional approach helps appreciate why YOLO was revolutionary. The old pipeline was slow and complex.

Definition

Traditional Detection Pipeline (R-CNN family):
Step 1: Region Proposal — Generate candidate bounding boxes (1000-2000 regions)
Step 2: Feature Extraction — Run CNN on each region
Step 3: Classification — Classify each region
Step 4: Post-processing — Filter redundant boxes using NMS

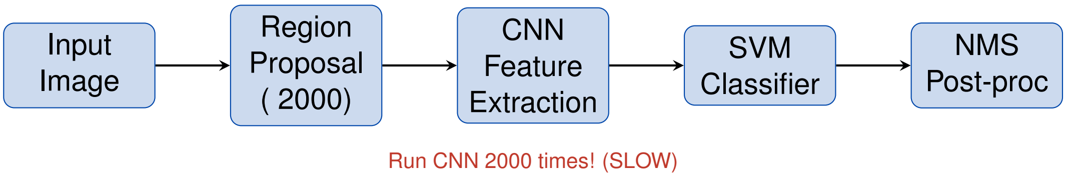


Figure 3: Traditional R-CNN pipeline: Multiple stages, multiple CNN passes

Problems with Traditional Approach

1. **Slow:** Must run CNN on each proposed region (2000+ times)
2. **Complex:** Separate models for each stage
3. **Not end-to-end:** Cannot train the entire pipeline together
4. **Local context only:** Each region processed independently
5. **Many false positives:** Limited context leads to errors

4 YOLO: You Only Look Once

Why It Matters

YOLO revolutionized object detection by framing it as a single regression problem. One neural network, one forward pass, real-time detection!

4.1 YOLO Key Insight

Definition

YOLO (You Only Look Once) performs detection as a **single regression problem**:

- Input: Entire image
- Output: All bounding boxes and class probabilities in one pass
- Single neural network does both localization AND classification

Analogy — Think of It Like This

Traditional vs YOLO:
Traditional: Like searching for your keys by checking every drawer one by one, then deciding if keys are there.
YOLO: Like glancing at the entire room once and immediately knowing where everything is.

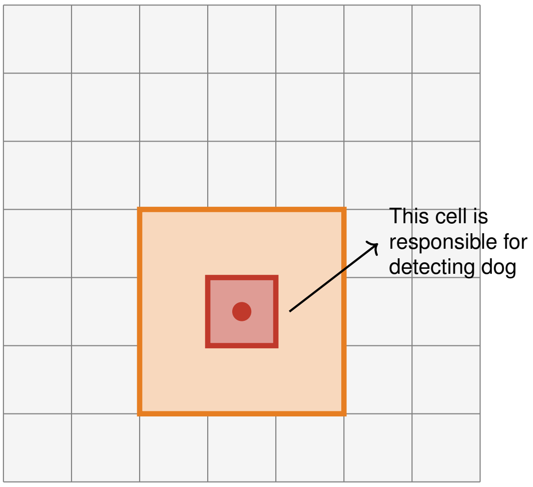
4.2 YOLO Grid System

Definition

YOLO Grid System:

1. Divide image into $S \times S$ grid cells
2. Each cell predicts B bounding boxes
3. Each bounding box has 5 values: $(x, y, w, h, \text{confidence})$
4. Each cell also predicts C class probabilities

YOLO v1 Parameters: $S = 7$, $B = 2$, $C = 20$ (PASCAL VOC)



7×7 Grid (YOLO v1)

Figure 4: YOLO grid: The cell containing the object's center is responsible for detecting it

4.3 YOLO Output Tensor

Key Formula

YOLO Output Tensor:

$$\boxed{\text{Output shape} = S \times S \times (B \cdot 5 + C)}$$

For YOLO v1:

$$S = 7 \text{ (grid size)}$$
$$B = 2 \text{ (bounding boxes per cell)}$$
$$C = 20 \text{ (classes in PASCAL VOC)}$$

$$\text{Output} = 7 \times 7 \times (2 \cdot 5 + 20) = 7 \times 7 \times 30 = \mathbf{1470} \text{ values}$$

Per-cell output (30 values):

- Box 1: x_1, y_1, w_1, h_1, c_1 (5 values)
- Box 2: x_2, y_2, w_2, h_2, c_2 (5 values)
- Class probabilities: $P(C_1), P(C_2), \ldots, P(C_{20})$ (20 values)

Solved Example 2: YOLO Output Tensor Size

Given Data:
Design a YOLO network for the COCO dataset:

- Grid size: $S = 13$
- Bounding boxes per cell: $B = 5$
- Number of classes: $C = 80$

Task: Calculate the output tensor dimensions.

Solution:
Step 1: Calculate per-cell output size

$$\text{Per cell} = B \times 5 + C = 5 \times 5 + 80 = 25 + 80 = 105$$

Step 2: Calculate total output tensor

$$\text{Output} = S \times S \times (B \times 5 + C) = 13 \times 13 \times 105 = 169 \times 105 = 17,745$$

Answer:

- Output tensor shape: $\mathbf{13 \times 13 \times 105}$
- Total values: $\mathbf{17,745}$

4.4 Bounding Box Predictions

Definition

YOLO Bounding Box Parameters:
For each bounding box, YOLO predicts 5 values:

1. x, y**:** Center of box **relative to grid cell** (0 to 1)
2. w, h**:** Width and height **relative to image** (0 to 1)
3. **Confidence:** $P(\text{Object}) \times \text{IoU}_{\text{pred}}^{\text{truth}}$

The confidence score represents both the probability that an object exists AND how accurate the box is.

Solved Example 3: Grid Cell Assignment

Given Data:

- Image size: 448×448 pixels
- Grid: $S = 7$ (so 7×7 cells)
- Object center location: $(200, 150)$ pixels

Task: Which grid cell is responsible? What are the relative coordinates?

Solution:
Step 1: Calculate cell size

$$\text{Cell size} = \frac{448}{7} = 64 \text{ pixels}$$

Step 2: Find grid cell indices

$$\text{Cell}_x = \left\lfloor \frac{200}{64} \right\rfloor = \lfloor 3.125 \rfloor = 3$$

$$\text{Cell}_y = \left\lfloor \frac{150}{64} \right\rfloor = \lfloor 2.34 \rfloor = 2$$

Step 3: Calculate relative position within cell

$$x_{\text{rel}} = \frac{200 - 3 \times 64}{64} = \frac{200 - 192}{64} = \frac{8}{64} = 0.125$$

$$y_{\text{rel}} = \frac{150 - 2 \times 64}{64} = \frac{150 - 128}{64} = \frac{22}{64} = 0.344$$

Answer:

- Responsible grid cell: $(\mathbf{3}, \mathbf{2})$ (0-indexed)
- Relative coordinates: $x = \mathbf{0.125}$, $y = \mathbf{0.344}$

These relative coordinates are what YOLO would predict (values between 0 and 1).

5 Non-Max Suppression (NMS)

Why It Matters

Multiple grid cells might detect the same object. NMS filters redundant detections, keeping only the best one for each object.

Definition

Non-Max Suppression (NMS) removes redundant overlapping boxes:

1. Sort boxes by confidence score (descending)
2. Select the box with highest confidence
3. Remove all boxes with IoU $>$ threshold with selected box
4. Repeat until no boxes remain

Non-Max Suppression Algorithm

Input: List of boxes B, confidence scores S, IoU threshold t
Output: Filtered list of boxes

```
1. Sort boxes by confidence (highest first)
2. Initialize empty list $D$ (final detections)
3. **While** boxes remain:
    a. Take box with highest confidence, add to $D$
    b. Remove this box from candidates
    c. For each remaining box:
        • If IoU with selected box $>$ threshold $t$: discard it
4. **Return** $D$
```

Solved Example 4: Non-Max Suppression

Given Data:
Four detected bounding boxes:

Box	Coordinates	Confidence
0	[100, 100, 200, 200]	0.90
1	[110, 110, 210, 210]	0.75
2	[105, 105, 205, 205]	0.80
3	[300, 300, 400, 400]	0.85

IoU threshold: 0.5
Task: Apply NMS to filter the boxes.

Solution:

Step 1: Sort by confidence

Order: Box 0 (0.90) $>$ Box 3 (0.85) $>$ Box 2 (0.80) $>$ Box 1 (0.75)

Step 2: Select Box 0 (confidence 0.90)

- Add Box 0 to final detections
- Calculate IoU with remaining boxes:
- IoU(0, 1) $\approx$ 0.68 $>$ 0.5 $\rightarrow$ **Discard Box 1**
- IoU(0, 2) $\approx$ 0.72 $>$ 0.5 $\rightarrow$ **Discard Box 2**
- IoU(0, 3) = 0 $<$ 0.5 $\rightarrow$ Keep Box 3

Step 3: Select Box 3 (confidence 0.85)

- Add Box 3 to final detections
- No more boxes to compare

Final Detections after NMS:

Box	Coordinates	Confidence
0	[100, 100, 200, 200]	0.90
3	[300, 300, 400, 400]	0.85

Boxes 1 and 2 were suppressed because they overlapped significantly with Box 0.

6 YOLO Loss Function

Why It Matters

The YOLO loss function is carefully designed to balance localization accuracy, confidence prediction, and classification performance.

Key Formula

YOLO Loss Function:

$$\mathcal{L} = \mathcal{L}_{\text{coord}} + \mathcal{L}_{\text{conf}} + \mathcal{L}_{\text{class}}$$

Coordinate Loss:

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

Confidence Loss:

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} \nVdash_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \nVdash_{ij}^{\text{noobj}} (C_i - \hat{C}_i)^2$$

Classification Loss:

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

Where:

- $\lambda_{\text{coord}} = 5$ (weight for coordinate loss)
- $\lambda_{\text{noobj}} = 0.5$ (weight for no-object confidence)
- $\nVdash_{ij}^{\text{obj}}$ = 1 if object in cell i, box j responsible
- Square root of w, h reduces sensitivity to large boxes

Solved Example 5: YOLO Loss Calculation

Given Data:
For one grid cell with object present:

- Ground truth: $x = 0.5, y = 0.5, w = 0.3, h = 0.4$, confidence=1.0
- Prediction: $x = 0.48, y = 0.52, w = 0.28, h = 0.38$, confidence=0.85
- Class (one-hot): GT = $[0, 0, 1, 0, 0]$, Pred = $[0.1, 0.1, 0.7, 0.05, 0.05]$
- $\lambda_{\text{coord}} = 5$

Solution:
Coordinate Loss:

$$\mathcal{L}_{xy} = 5 \times [(0.5 - 0.48)^2 + (0.5 - 0.52)^2] = 5 \times [0.0004 + 0.0004] = 0.004$$

Size Loss (using square roots):

$$\begin{aligned}\mathcal{L}_{wh} &= 5 \times [(\sqrt{0.3} - \sqrt{0.28})^2 + (\sqrt{0.4} - \sqrt{0.38})^2] \\ &= 5 \times [(0.548 - 0.529)^2 + (0.632 - 0.616)^2] \\ &= 5 \times [0.00036 + 0.00026] = 0.003\end{aligned}$$

Confidence Loss:

$$\mathcal{L}_{\text{conf}} = (1.0 - 0.85)^2 = 0.0225$$

Classification Loss:

$$\begin{aligned}\mathcal{L}_{\text{class}} &= (0 - 0.1)^2 + (0 - 0.1)^2 + (1 - 0.7)^2 + (0 - 0.05)^2 + (0 - 0.05)^2 \\ &= 0.01 + 0.01 + 0.09 + 0.0025 + 0.0025 = 0.115\end{aligned}$$

Total Loss:

$$\mathcal{L} = 0.004 + 0.003 + 0.0225 + 0.115 = 0.1445$$

Loss Components:

Component	Value
Coordinate loss (x, y)	0.004
Size loss (w, h)	0.003
Confidence loss	0.0225
Classification loss	0.115
Total Loss	**0.1445**

The classification loss dominates here because the predicted class probability (0.7) differs significantly from ground truth (1.0).

7 YOLO Architecture and Versions

7.1 YOLO v1 Architecture

Definition

YOLO v1 Network:

- Input: $448 \times 448 \times 3$
- 24 Convolutional layers
- 2 Fully connected layers
- Output: $7 \times 7 \times 30$
- Inspired by GoogLeNet architecture

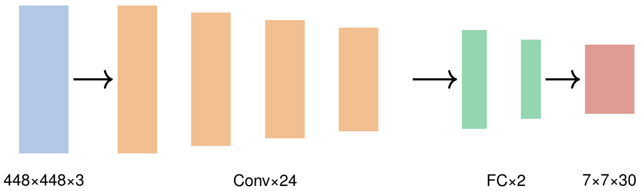


Figure 5: Simplified YOLO v1 architecture

7.2 YOLO Versions Comparison

Version	YOLO v1 (2016)	YOLO v2 (2017)	YOLO v3 (2018)
Classes	20 (VOC)	9000+	80 (COCO)
Speed	45 FPS	67 FPS	30 FPS
Key Features	Single network	Batch norm, anchor boxes	Multi-scale detection
Small objects	Poor	Better	Good
Backbone	Custom	Darknet-19	Darknet-53

Table 1: Comparison of YOLO versions

7.3 YOLO Advantages and Limitations

YOLO Pros and Cons

Advantages:

- **Speed:** Real-time detection (45+ FPS)
- **Global context:** Sees entire image, fewer false positives
- **Generalizable:** Works well on artwork, new domains
- **End-to-end:** Single network, easy to train

Limitations:

- **Small objects:** Struggles with small or grouped objects

- **Localization:** More localization errors than R-CNN
- **Spatial constraint:** Limited boxes per cell
- **Aspect ratios:** Fixed anchor boxes

8 Summary

Key Takeaways

1. Object Detection Tasks

- Classification: One label per image
- Detection: Bounding boxes + labels
- Segmentation: Per-pixel labels

2. IoU (Intersection over Union)

$$\boxed{\text{IoU} = \frac{\text{Intersection Area}}{\text{Union Area}}}$$

- IoU ≥ 0.5 typically considered good detection

3. YOLO Key Ideas

- Single neural network for detection
- Divide image into $S \times S$ grid
- Each cell predicts B boxes with $(x, y, w, h, \text{conf})$
- Output: $S \times S \times (B \cdot 5 + C)$

4. Non-Max Suppression

- Sort by confidence, keep best, remove overlapping

5. YOLO Loss

$$\mathcal{L} = \lambda_{\text{coord}}\mathcal{L}_{xy,wh} + \mathcal{L}_{\text{conf}} + \mathcal{L}_{\text{class}}$$

Self-Test — Check Your Understanding

Quick Quiz:

1. What is the IoU if intersection = 400 and union = 800?
2. For YOLO v1 (S=7, B=2, C=20), what is the output tensor size?
3. In NMS with threshold 0.5, if two boxes have IoU=0.6, what happens?

Answers:

1. IoU = 400/800 = 0.5
2. $7 \times 7 \times (2 \times 5 + 20) = 7 \times 7 \times 30 = 1470$
3. The box with lower confidence is suppressed (removed)

9 Glossary

Term	Definition
Object Detection	Localizing and classifying multiple objects in an image
Bounding Box	Rectangle enclosing a detected object
IoU	Intersection over Union, measures overlap between boxes
YOLO	You Only Look Once, real-time detection algorithm
NMS	Non-Max Suppression, filters redundant detections
Confidence	Probability that box contains object × IoU accuracy
Anchor Box	Pre-defined box shapes to improve detection
mAP	Mean Average Precision, detection evaluation metric
FPS	Frames Per Second, speed metric