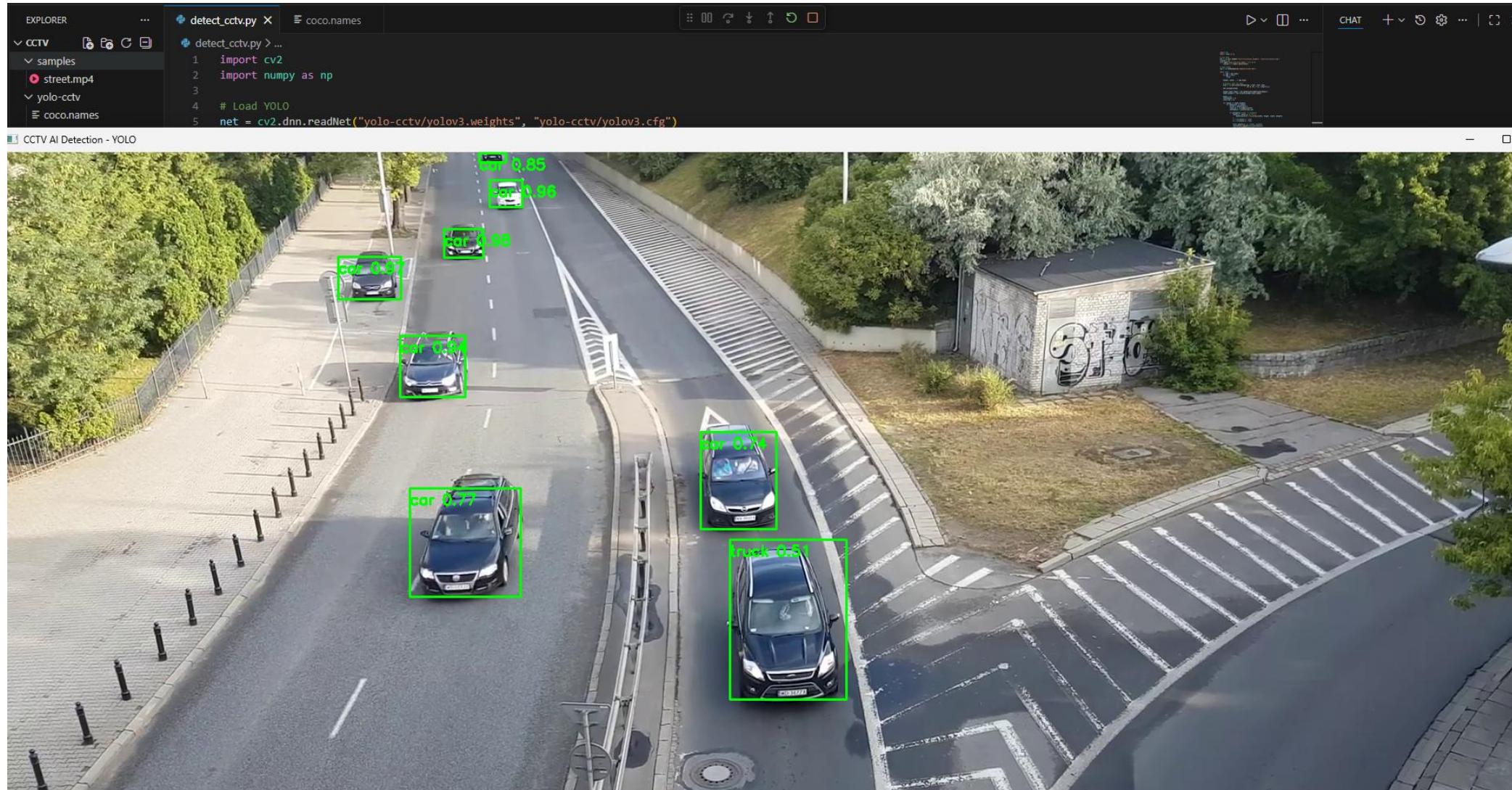


YOLO : You Only Look Once - Real Time Object Detection

Krijimi i programit për identifikim të objekteve që përdorë modelin e trajnuar me YOLO



Komponentët kryesorë të programit

The screenshot shows a code editor with a dark theme. On the left, there is a file tree for a project named 'cctv'. The tree includes a 'samples' folder containing 'street.mp4', a 'yolo-cctv' folder containing 'coco.names', 'yolov3.cfg', and 'yolov3.weights', and the main Python script 'detect_cctv.py' which is selected.

```
    < cctv
      < samples
        street.mp4
      < yolo-cctv
        coco.names
        yolov3.cfg
        yolov3.weights
      < detect_cctv.py
```

The content of 'detect_cctv.py' is as follows:

```
#!/usr/bin/env python3
# detect_cctv.py > ...
# Import OpenCV (cv2) and NumPy (np)
import cv2
import numpy as np

# Load YOLO
net = cv2.dnn.readNet("yolo-cctv/yolov3.weights", "yolo-cctv/yolov3.cfg")
classes = []
with open("yolo-cctv/coco.names", "r") as f:
    classes = f.readlines()

# Input Video
cap = cv2.VideoCapture("samples/street.mp4")

while True:
    _, img = cap.read()
    if img is None:
        break

    height, width, _ = img.shape

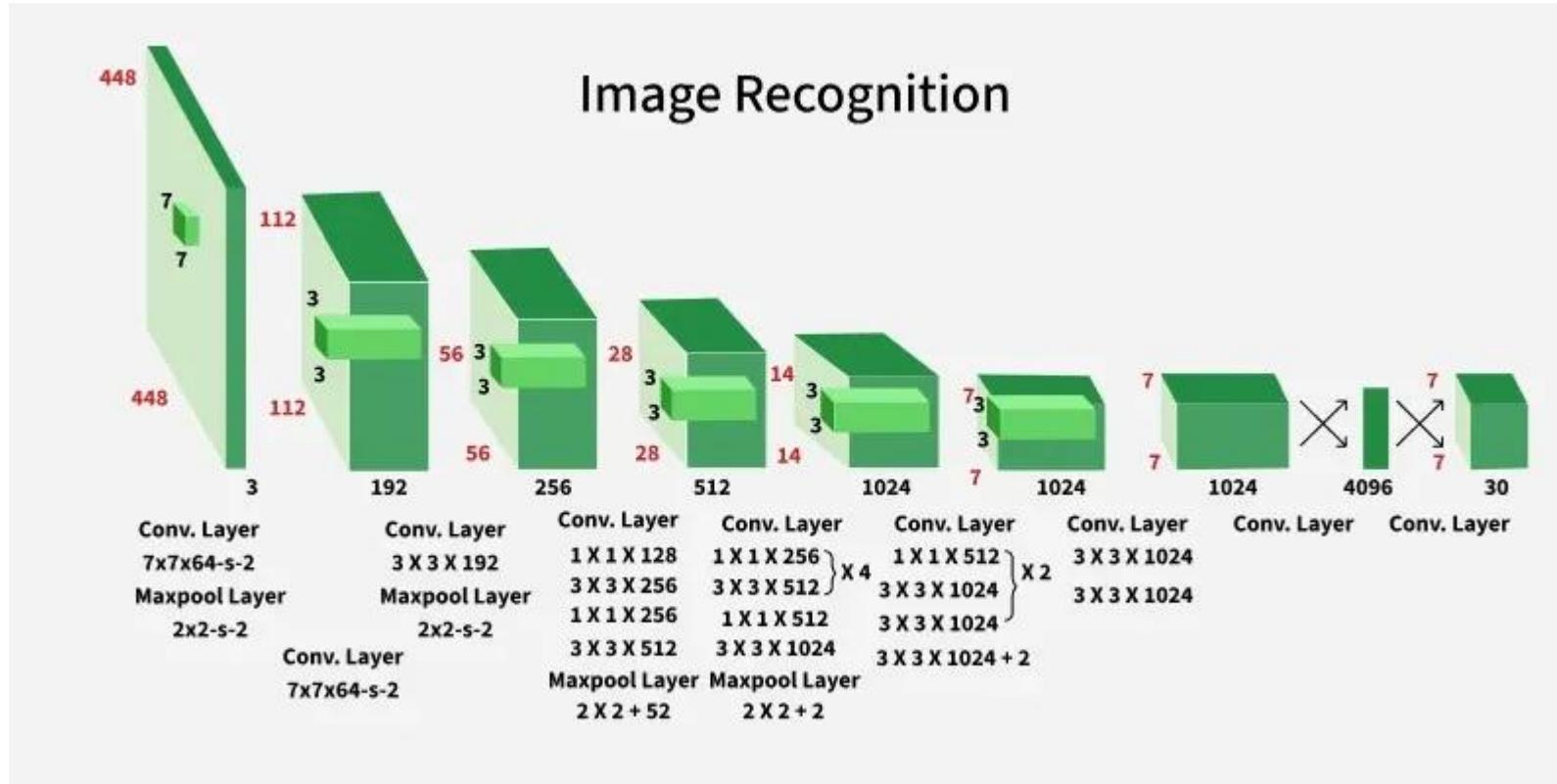
    # Prepare input for YOLO
    blob = cv2.dnn.blobFromImage(img, 1/255, (416, 416),
                                 (0, 0, 0), True, crop=False)
    net.setInput(blob)

    output_layer_names = net.getUnconnectedOutLayersNames()
    layer_outputs = net.forward(output_layer_names)

    boxes = []
    confidences = []
    class_ids = []
```

- **yolov3.cfg** = si ndërtohet rrjeti
- **yolov3.weights** = rezultati i trajnimit
- **coco.names** = emrat e objekteve

Arkitektura e YOLO – Çka ndodh brenda rrjetit?



YOLO përdor një Convolutional Neural Network si bazë të tij.

Imazhi i hyrjes, 448×448 , kalon nëpër disa shtresa konvolucionale dhe pooling.

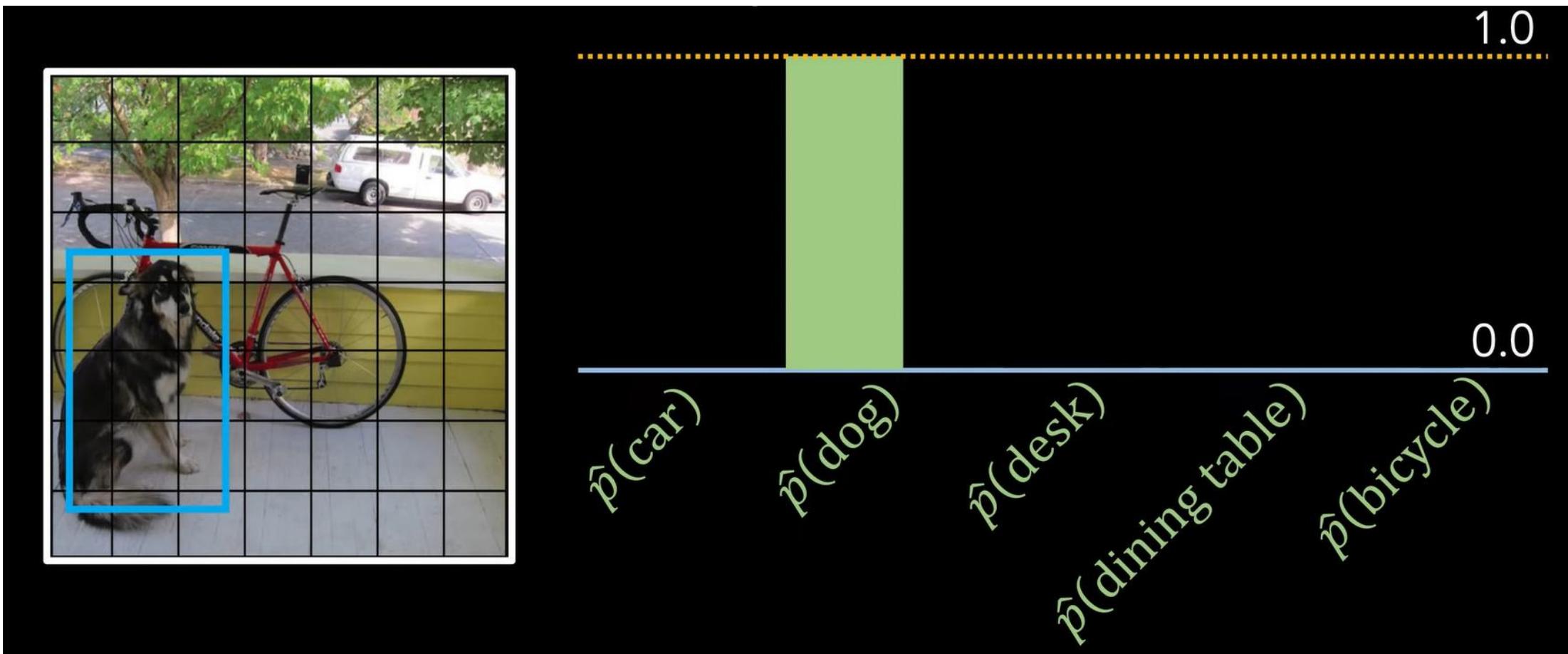
Gjatë këtij procesi, dimensionet zvogëlohen:

$448 \times 448 \rightarrow 224 \times 224 \rightarrow 112 \times 112 \rightarrow 56 \times 56 \rightarrow 28 \times 28 \rightarrow 14 \times 14 \rightarrow 7 \times 7$

Në fund, rrjeti krijon një hartë 7×7 , dhe për secilën qelizë parashikon:

- 2 bounding boxes
- 1 confidence score
- probabilitetet e klasave për objektin

Trajnimi i modelit : Ground truth(class probabilities)



Ground Truth është vlera e vërtetë dhe e saktë e objektit në imazh.

Bounding box-i i Ground Truth konsiderohet 100% i saktë dhe shërben si standard për të matur sa afër janë parashikimet e modelit gjatë trajnimit

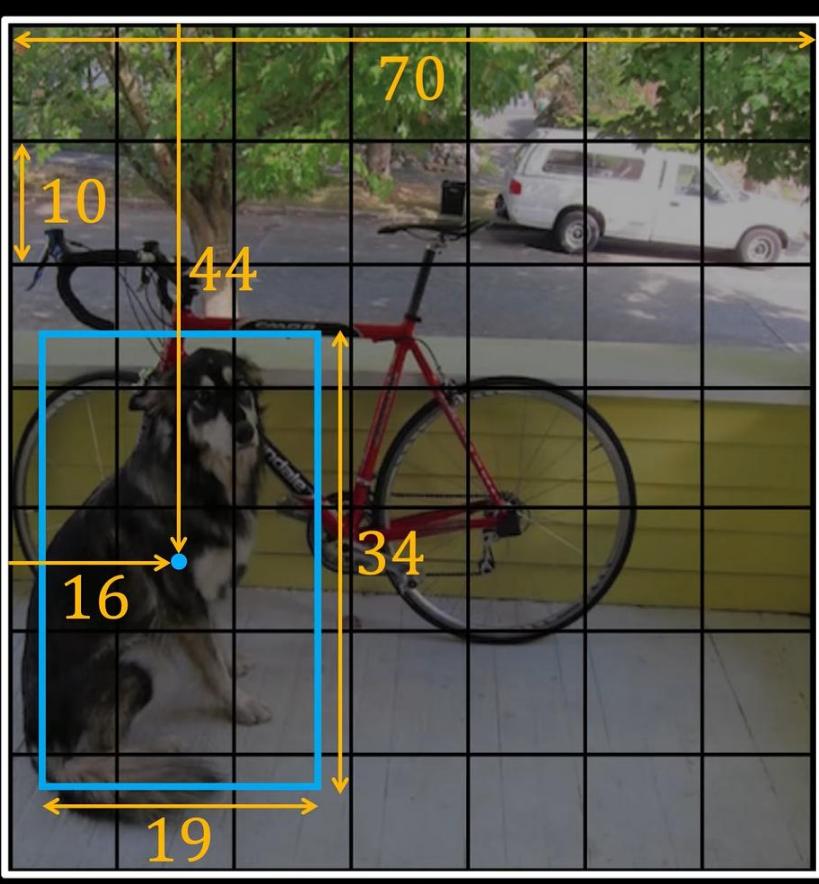
Trajnimi i modelit : Ground truth(box coordinates)

$$x = \frac{16 \% 10}{10} = 0.60$$

$$y = \frac{44 \% 10}{10} = 0.40$$

$$w = \frac{19}{70} = 0.27$$

$$h = \frac{34}{70} = 0.49$$



Box Coordinates janë vlerat që përshkruajnë vendndodhjen dhe madhësinë e një objekti në imazh. Ato përdoren nga YOLO për të vizuatuar bounding boxes dhe për të llogaritur saktësinë e parashikimeve kundrejt Ground Truth.

Pozita dhe madhësia e objekteve i lejojnë YOLO-s jo vetëm të identifikojë çfarë objekti është në imazh, por gjithashtu edhe ku ndodhet dhe sa i madh është. Kjo e bën YOLO-n shumë të vlefshëm për aplikime reale si CCTV, monitorim trafiku dhe robotikë.

Trajnimi i modelit : Ground truth(confidence)

- Does an object appear in this grid cell or not?

$$\text{Red square} = 0$$

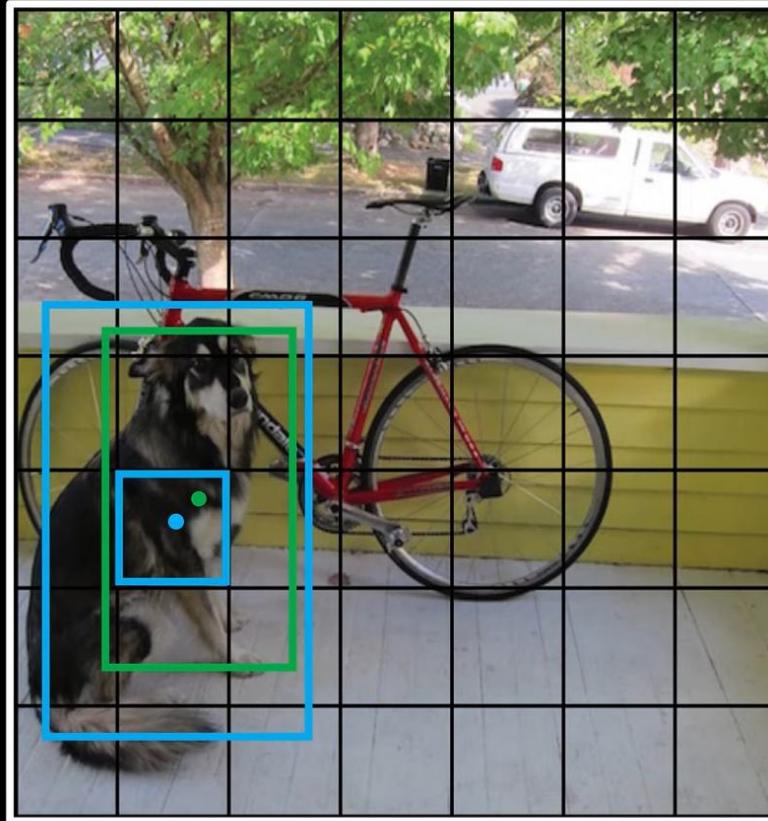
$$\text{Green square} = \text{IoU}(\text{pred}, \text{true})$$



Ground Truth Confidence është vlera binare nga dataset-i që tregon nëse një objekt ekziston në një qelizë specifike të grid-it. Përdoret për të llogaritur gabimin e konfidencës gjatë trajnimit — nëse modeli thotë se ka objekt aty ku nuk ka, ai penalizohet.

Trajnimi i modelit : Ground truth(confidence)

$$\hat{C} = \text{IoU}(\text{pred}, \text{true})$$



Gjatë trajnimit, IoU mat sa përputhen kutitë e parashikuara nga modeli me Ground Truth. Sa më i lartë IoU, aq më i saktë modeli. IoU përdoret për të llogaritur humbjen dhe për të udhëhequr përmirësimin e modelit gjatë mësimit.

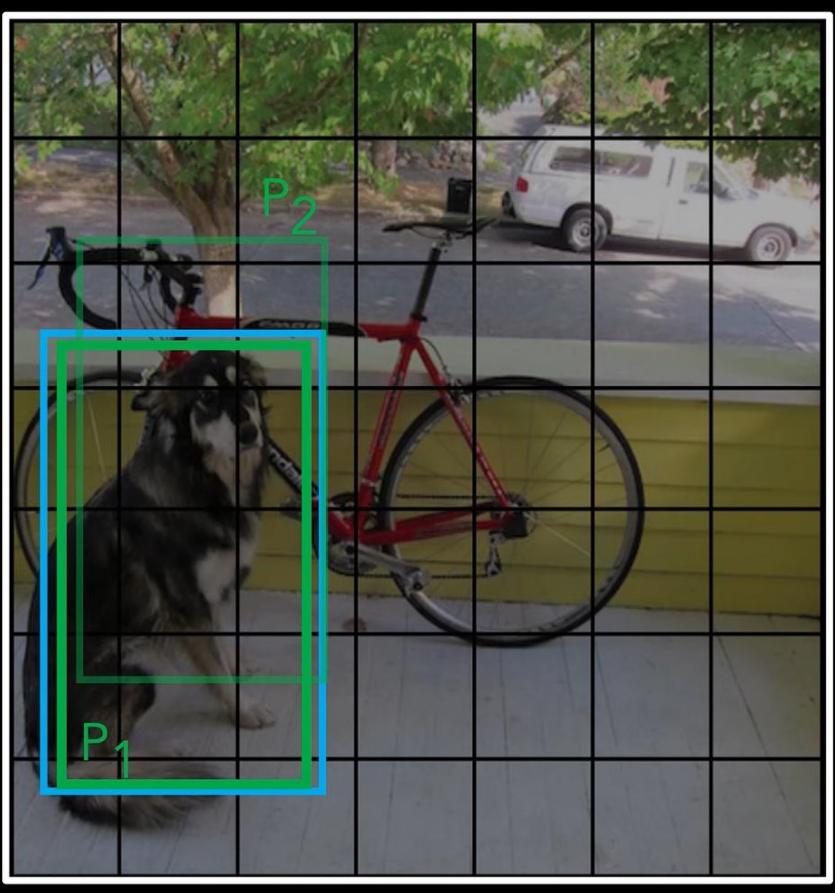
Gjatë trajnimit, modeli parashikon kutitë fillestare. Ne e krahasojmë secilën me Ground Truth duke përdorur IoU. Sa më shumë të përputhen kutitë, humbja është më e ulët. Modeli vazhdimesht përditëson parametrat derisa të arrijë konvergjencë — pra të mësojë të detektojë objektet me saktësi të lartë.

Trajnim i modelit : Box selection(inference)

If $\text{IoU}(P_1, P_2) > \text{Threshold}$:

$$P = \text{argmax}(\text{C}(P_1), \text{C}(P_2))$$

(Non max suppression)



Non-Max Suppression është një teknikë që heq kutitë e tepërta të detektuara për të njëjtin objekt, duke zgjedhur vetëm kutinë me probabilitetin më të lartë dhe përputhjen më të mirë

Trajnimi i modelit : Loss function

$$L = \sum_{i=0}^{S^2} \mathbb{I}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{ij}^{\text{obj}} (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{ij}^{\text{obj}} (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B [\mathbb{I}_{ij}^{\text{noobj}} (C_i - \hat{C}_i)^2] + \sum_{i=0}^{S^2} \sum_{j=0}^B [\mathbb{I}_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2]$$

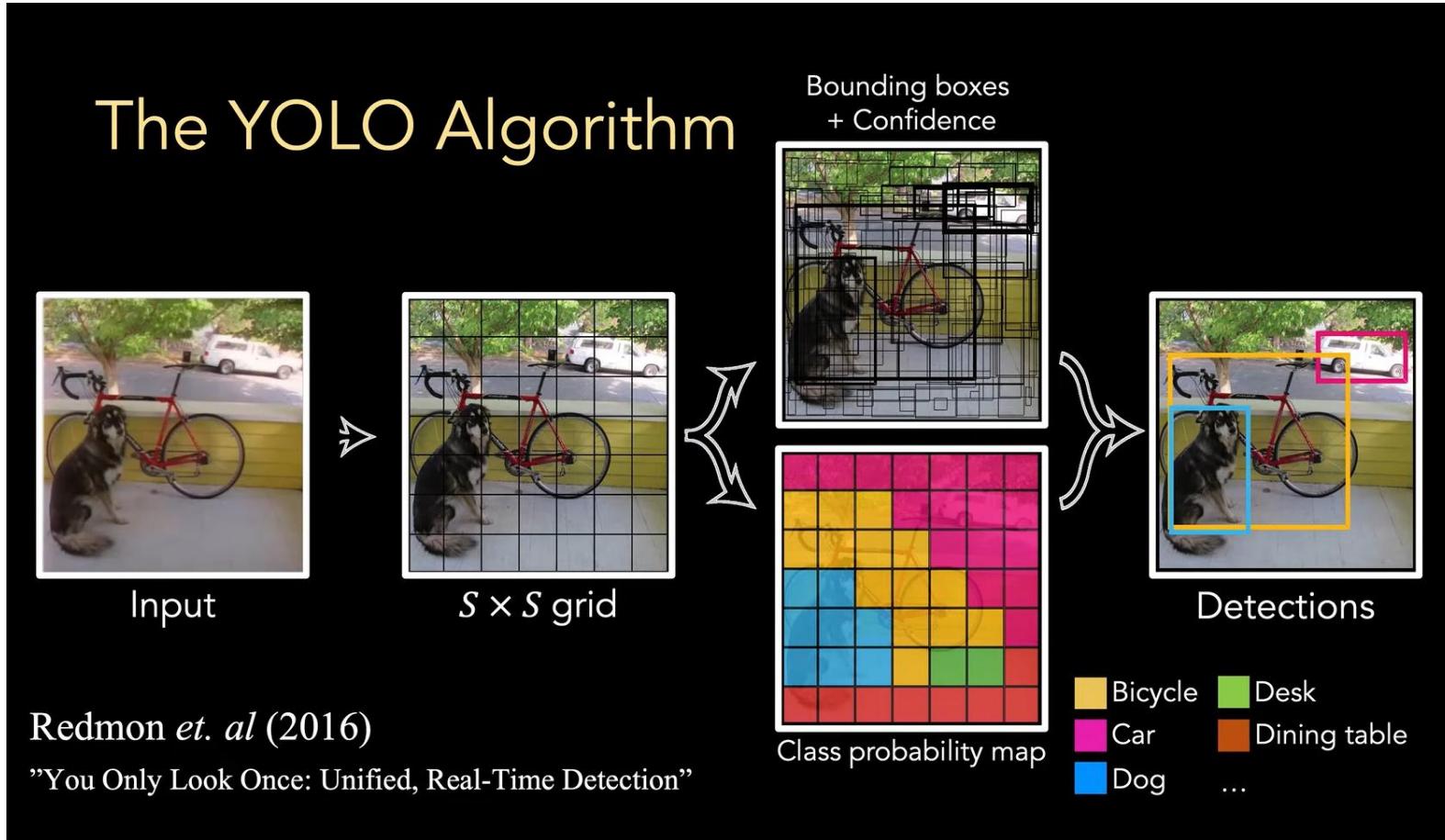
Loss function
(Object detection as regression)

Loss Function në YOLO përbëhet nga tre komponentë kryesorë:

- Localization Loss për pozicionin dhe përmasat e kutive,
- Confidence Loss për vlerësimin e pranisë së objektit dhe
- Classification Loss për të identifikuar saktë klasën e objektit.

Gjatë trajnimit, modeli përditëson parametrat në mënyrë që të minimizohet vlera totale e humbjes dhe të përmirësojë saktësinë e detektimit.

Inference

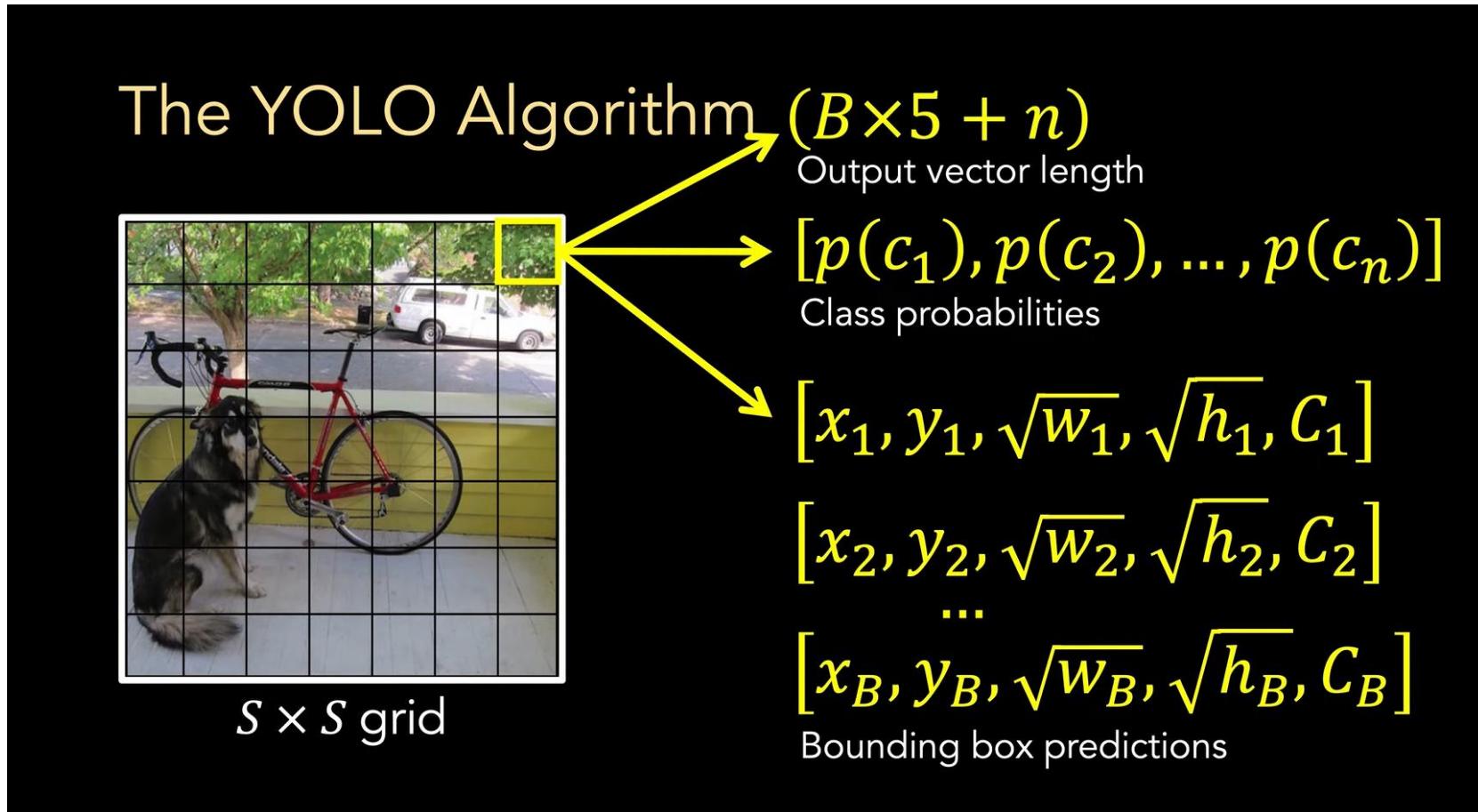


Imazhi ndahet në 7×7 grid që çdo qelizë të jetë përgjegjëse për detektimin e objektit në zonën e saj. Kjo e thjeshton detektimin, e ul kompleksitetin dhe e bën YOLO-n jashtëzakonisht të shpejtë për aplikime në kohë reale si CCTV.

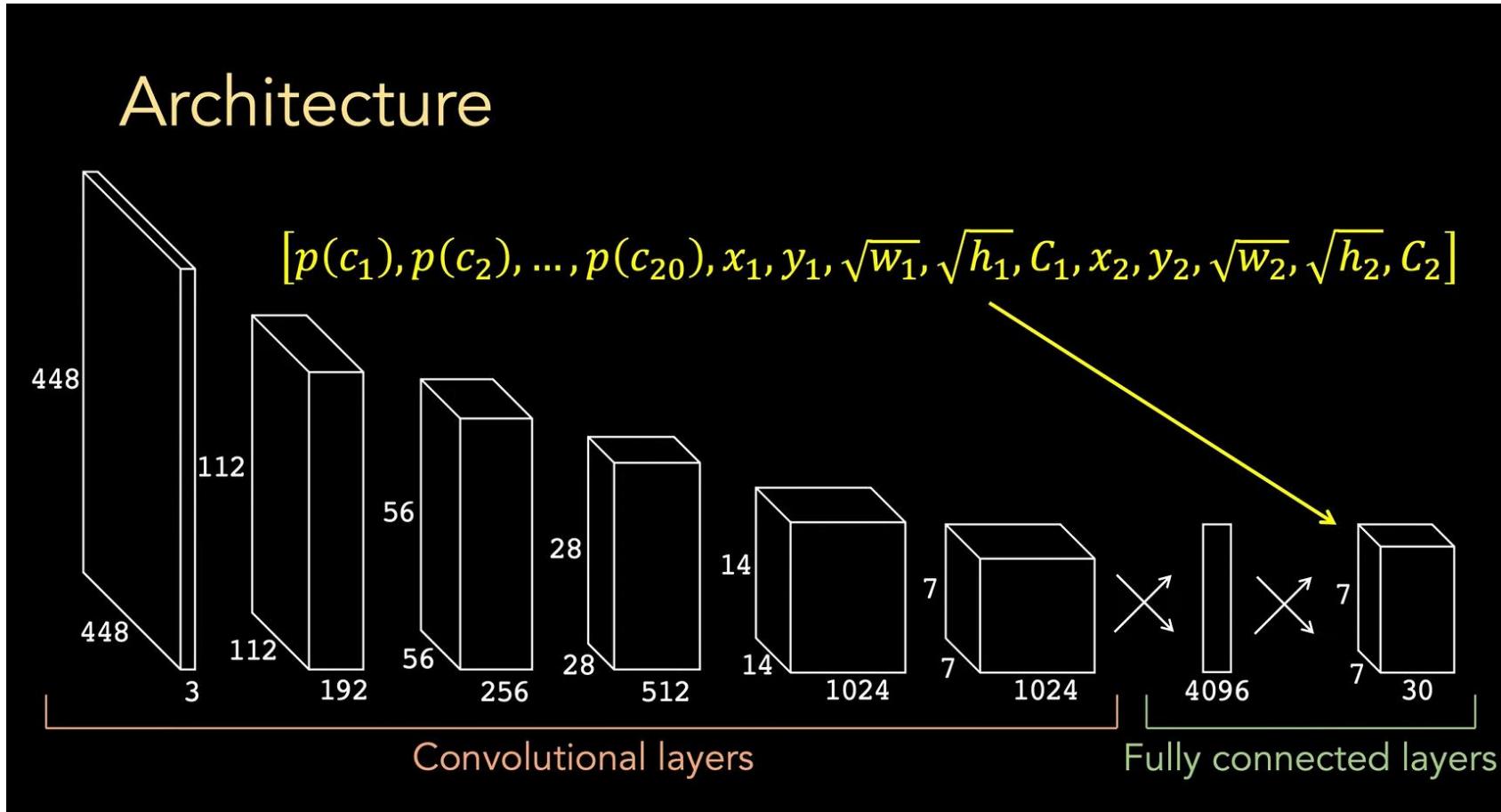
Bounding boxes tregojnë vendndodhjen e objektit, ndërsa confidence score tregon sa i sigurt është modeli në këtë parashikim.

Class Probability Map është grapi i probabiliteteve për secilën klasë objekti në çdo qelizë të grid-it. Ai i tregon modelit se çfarë objekti është

Inference - vektorizimi



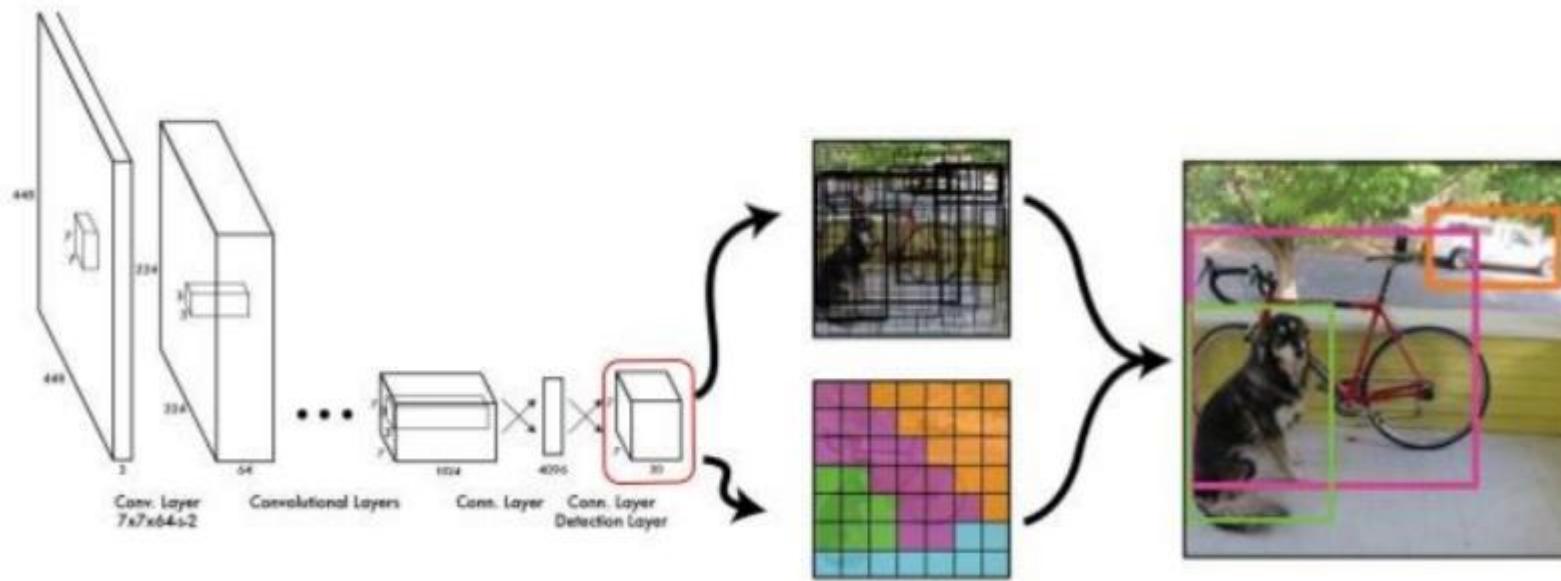
Inference - CNN



YOLO transformon imazhin fillestar përmes shtresave konvolucionale për të nxjerrë tipare, dhe më pas përmes fully connected layers prodhon vektorët e parashikimit të objekteve për secilën nga 49 qelizat e grid-it (7×7), duke përfshirë klasa, pozicion dhe madhësi të bounding box-eve.

Inference - CNN

YOLO: You Only Look Once



Imazhi përpunohet nga CNN për të nxjerrë tipare vizuale.

Për secilën qelizë të grid-it, modeli parashikon coordinates e bounding box-it, confidence score dhe probabilitetet e klasave.

Pas filtrimit të kutive të tepërtë, YOLO identifikon objektet finale dhe pozicionin e tyre në imazh në kohë reale.