TEXAS
The University of Texas at Austin

You Only Look Once: Unified, Real-Time Object Detection

Presenter: Daniel Adebi

8/29/2023

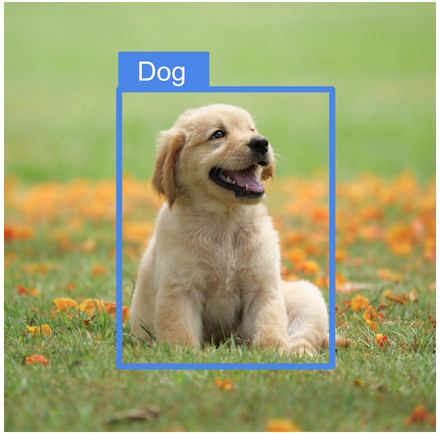
What is Object Detection?

- Combination of **Classification** (what object is) and **Localization** (where it is)

Input Image

Object Detector

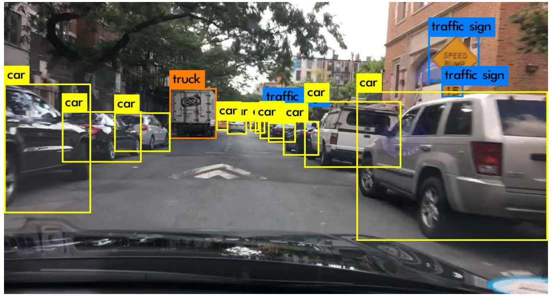
Output



Class: Dog
Bbox: (x, y, w, h)

Why is Object Detection Important?

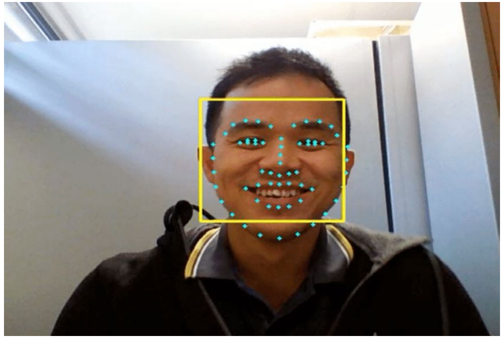
Self-Driving Cars



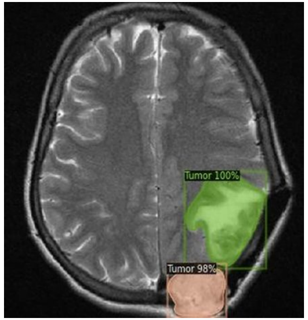
Robotics

Surveillance

Face Recognition



Healthcare

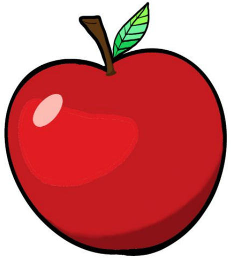


What Makes Object Detection Challenging?

- Making object detection both *fast* and *accurate* is difficult.
- Past methods performed the classification and localization in different phases, making optimization harder.
- Creating generalizable representations of images.

Apple

Apple?



Prior Work

- Deformable Parts Models (DPM)
  - Main Idea: Uses sliding window to cover whole image
  - Problem: Slow
- R-CNN (Regions with CNN features)
  - Main Idea: Generate bounding box proposals, classify proposals, post-process to refine boxes.
  - Problem: Slow and hard to optimize

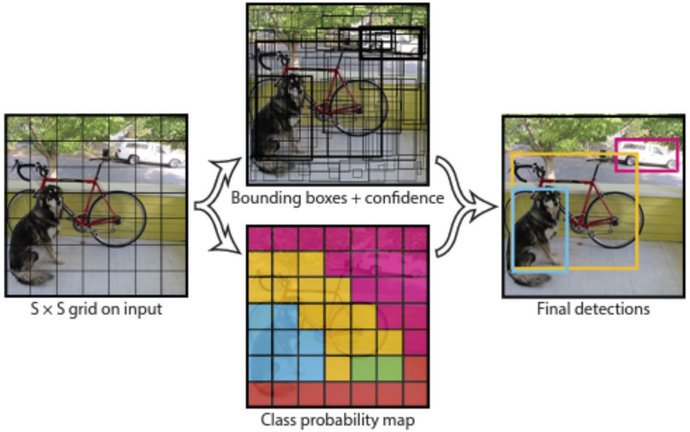
YOLO: Unified Approach

See Figure 2 for more details

Proposal:
An end-to-end unified approach where a single neural network is used to BOTH ***classify*** and ***localize*** the object in the image.

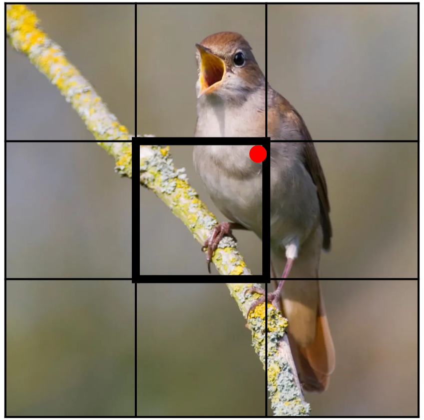
Input: Image divided into grid.

Outputs: For each cell in grid:
- *B* bounding boxes and confidence scores
- *C* conditional class probabilities



YOLO: Understanding Inputs and Outputs

Input: S x S grid. If object's center lies in a grid cell, that grid cell needs to detect that object.



Output: S x S x (5*B + C) tensor

- *B* bounding boxes are produced for each grid cell. (x, y, h, w, c)
    - c = confidence score. Measures IoU (Intersection over Union)
- *C* conditional class probabilities
    - Pr[Class_i | object]

YOLO: Network Design

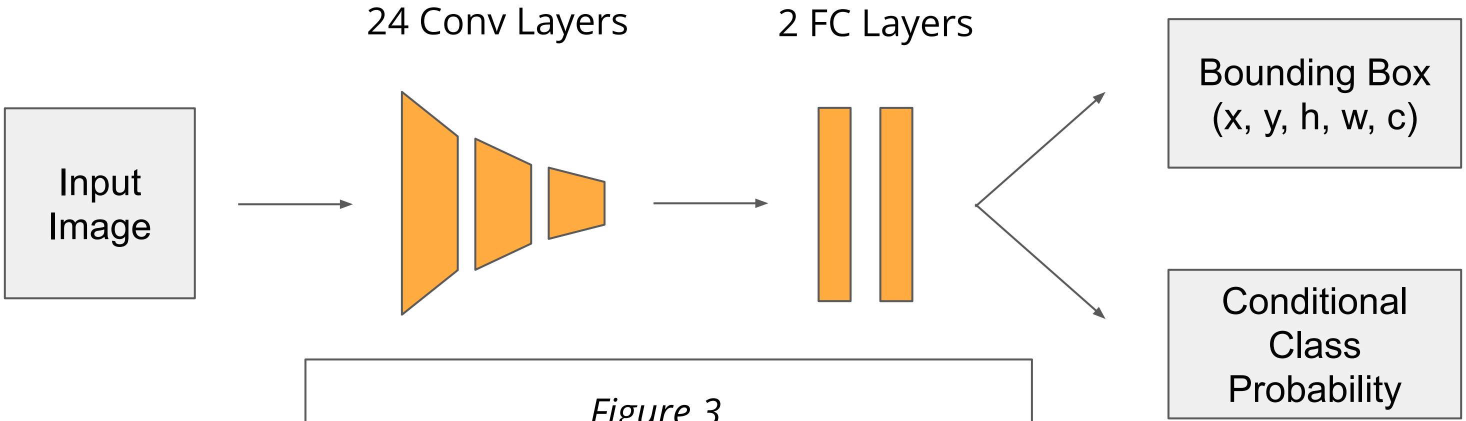
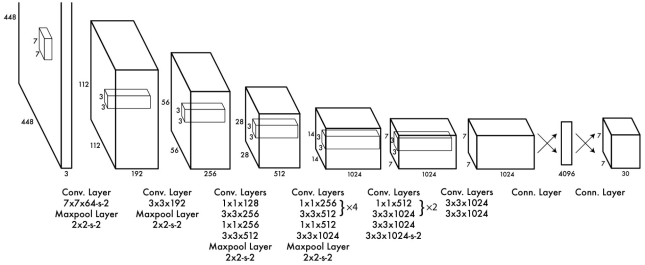


Figure 3



YOLO: Training Framework

Pretrain first 20 convolutional layers on ImageNet 1000-class for classification.

Connect pretrained layers to 4 remaining conv layers + 2 FC layers for detection.

Normalize bounding box dimensions, and optimize model using sum-squared error.

YOLO: Loss Function

Bounding box position

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

Bounding box dimension

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

Confidence

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

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

Class Probability

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

YOLO: Loss Function (Cont)

Only penalizes predictor responsible for ground truth box

Bounding box position

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

Bounding box dimension

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

Confidence

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

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

Confidence conditioned for both cases w/ and w/o object

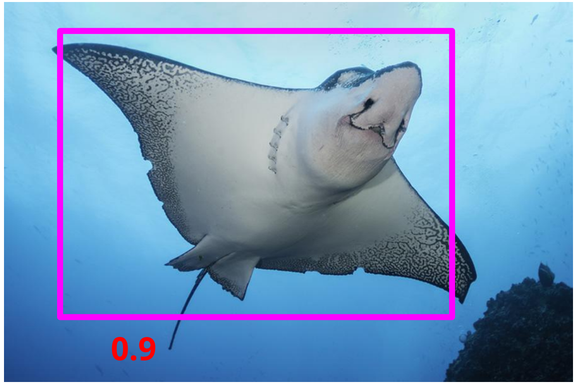
Class Probability

Only penalize if object is in cell *i*

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

YOLO: Inference

- Usually, it’s clear which cell is responsible for which object.
- But sometimes, you can have two (or more) high quality bounding boxes, especially for larger objects.
- Solution? Non-maximal suppression. Idea: remove bounding boxes with high overlap (IoU)



Experiment Structure

Datasets

- Pascal VOC 2007 and VOC 2012
- Picasso Artwork and People-Art Datasets

Metrics:

- mAP and FPS

YOLO Model Variants

- YOLO
- Fast YOLO (less conv layers)
- YOLO VGG-16 (slow but more accurate)

Comparison Models:

- DPM
- R-CNN (and variants)

Results: Comparison to Real-Time Systems

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

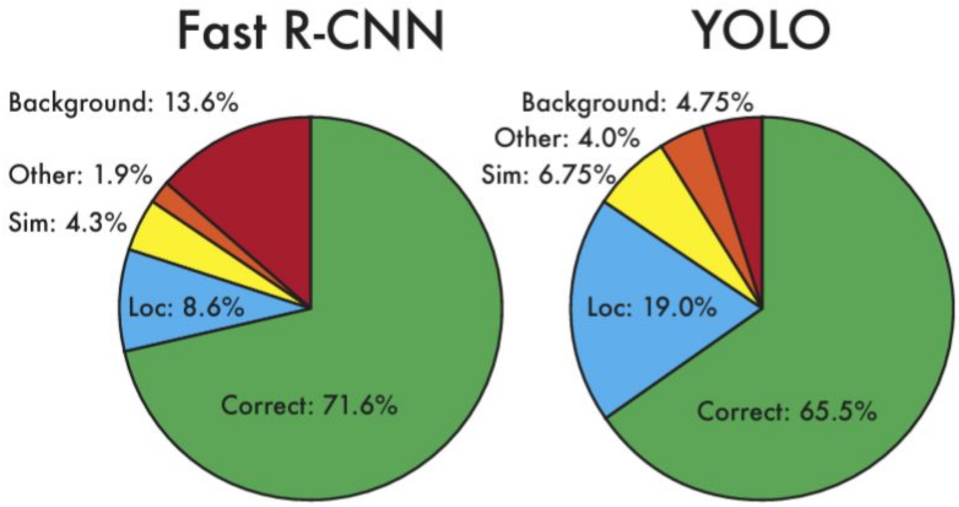
YOLO is both faster and more accurate as real-time system

Results: YOLO vs R-CNN

Error Analysis (Figure 4)

Fast R-CNN

- Localization error is low
- Background error is high



YOLO

- Localization error is high
- Background error is low

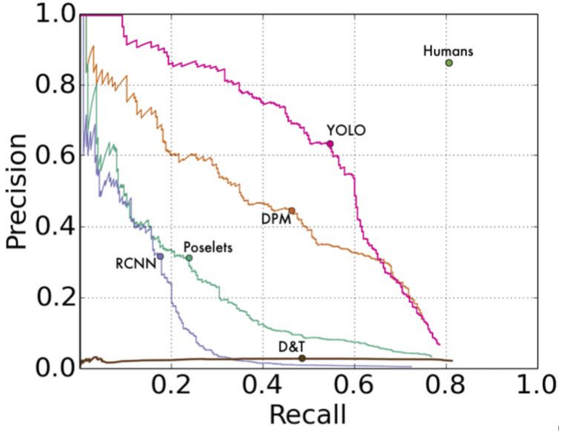
Results: YOLO with R-CNN

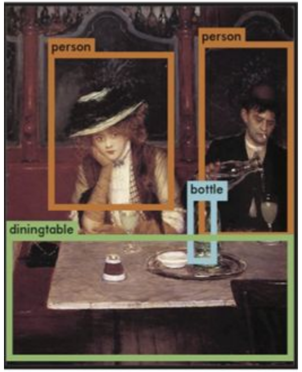
- YOLO can be combined with R-CNN to improve performance.
- Logic: Both methods have opposing strengths and weaknesses and can work better together.

	mAP	Combined	Gain
Fast R-CNN	71.8	-	-
Fast R-CNN (2007 data)	**66.9**	72.4	.6
Fast R-CNN (VGG-M)	59.2	72.4	.6
Fast R-CNN (CaffeNet)	57.1	72.1	.3
YOLO	63.4	**75.0**	**3.2**

Results: Generalizability

- Trained on VOC and Tested on two artwork datasets.
- YOLO generalizes significantly better than prior works.





	VOC 2007	Picasso		People-Art
	AP	AP	Best F_1	AP
YOLO	**59.2**	**53.3**	**0.590**	**45**
R-CNN	54.2	10.4	0.226	26
DPM	43.2	37.8	0.458	32
Poselets [2]	36.5	17.8	0.271	
D&T [4]	-	1.9	0.051	

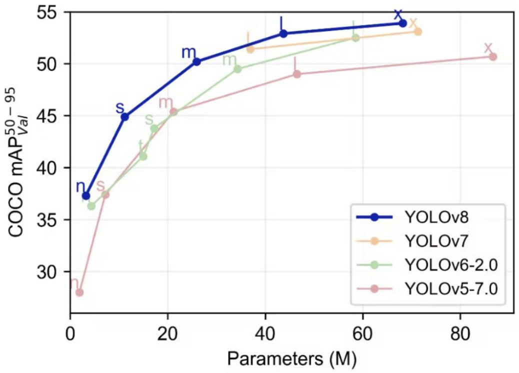
Limitations of YOLO

- Strong spatial constraints on bbox prediction limits number of nearby objects predicted
- Struggles to predict smaller objects
- Struggles to generalize to boxes with different aspect ratios
- Treats small and large bbox errors similarly.

Potential Research Extensions and Applications

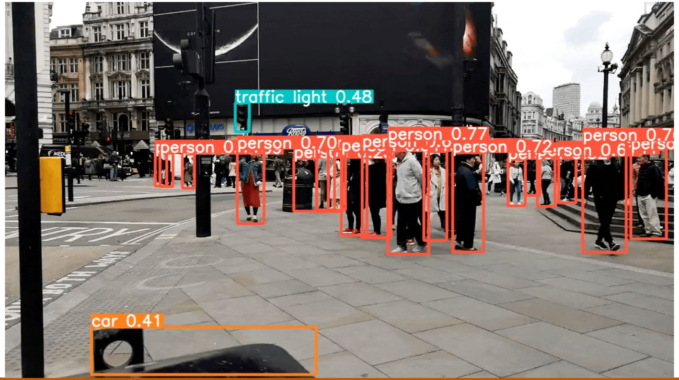
Improvements Made

- More accurate versions of YOLO have been developed (YOLOv2, YOLOv3, YOLOv4, … YOLOv8)
- Employ techniques like using anchor boxes, data augmentations, architecture changes, etc.



Open Problems

- Adapting 2D object detection models to various hardware (e.g. robots, IoT devices)
- Adapt YOLO models to work in multi-modal settings (e.g. vision, language, audio)



More Research Extensions

3D Object Detection ([YOLO3D])

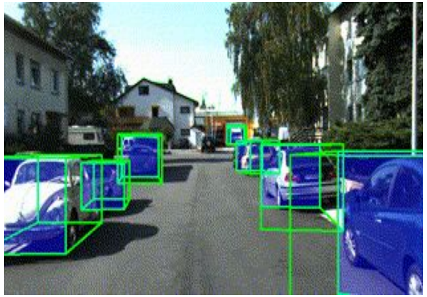
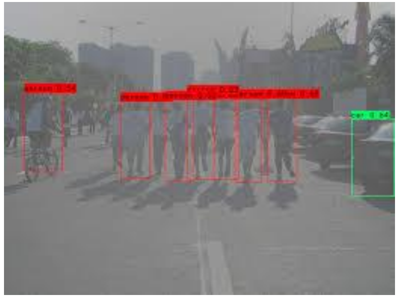


Image Restoration ([TogetherNet])



Instance Segmentation ([YOLACT])

Robotics ([YOLO-GD])

Recap

Problem

- Real-time 2D object detection is *slow* and *difficult*.
- Past methods separated localization and classification (difficult to optimize).

Solution:

- YOLO is *unified* detector that makes object detection *fast* while still maintaining accuracy.
- Generalizes well to new domains.

Thank you!