YOLO - You only look once

Shiva Krishna, Sasi Kiran

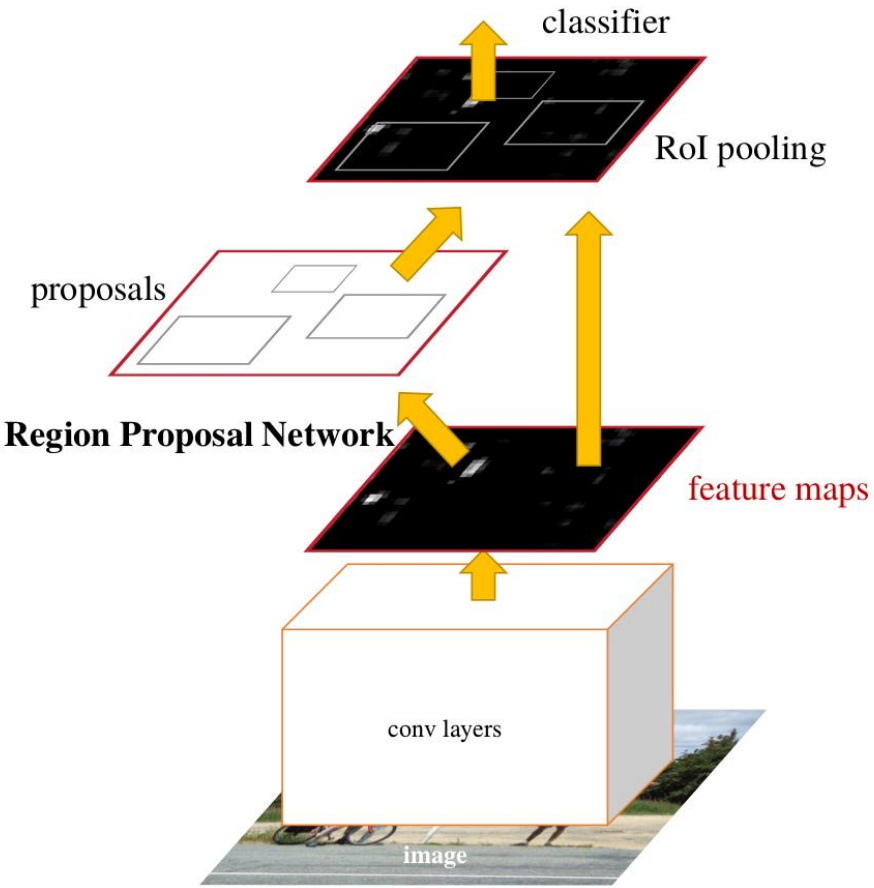
IIT-Madras

April 10, 2017

Overview

Motivation

State of the art models like Faster RCNN are slow and hard to optimize. Needs heavy parallelization for efficient implementation in real-life applications



Motivation

Consider the real-life application of Object Detection task in Self-driving car. Let's say the car is moving at $60 kmph$

Motivation

Algorithm	Pascal 2007 mAP	Frame Rate	Speed
DPM v5	33.7	0.07FPS	14 s/img
RCNN	66.0	0.05FPS	20 s/img

Table: Comparison of Object Detection Algorithms

If we use RCNN, the car moves $\frac{1}{3}km$ before halting on seeing an object.

Motivation

Algorithm	**Pascal 2007 mAP**	**Frame Rate**	**Speed**
DPM v5	33.7	0.07FPS	14 s/img
RCNN	66.0	0.05FPS	20 s/img
Fast RCNN	70.0	0.5FPS	2 s/img

Table: Comparison of Object Detection Algorithms

If we use Fast RCNN, the car moves $33m$ before halting on seeing an object.

Motivation

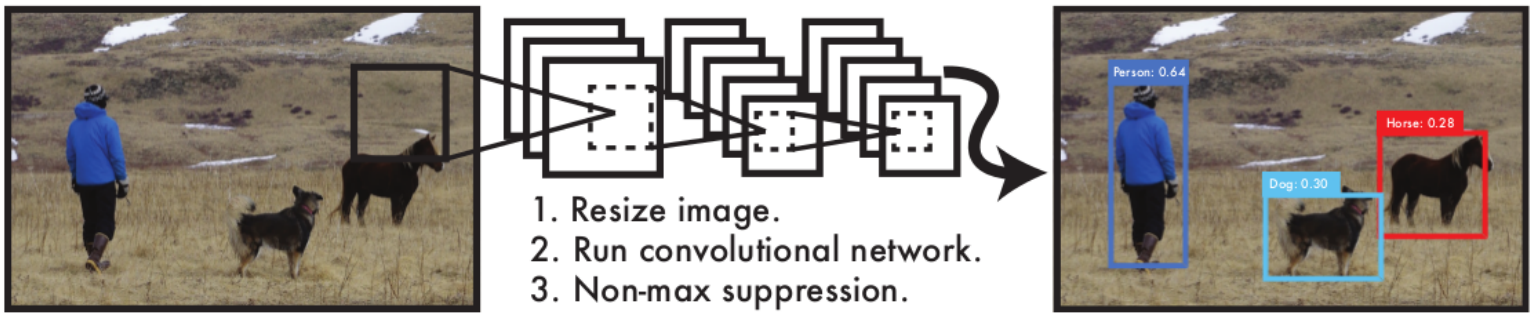
Algorithm	Pascal 2007 mAP	Frame Rate	Speed
DPM v5	33.7	0.07FPS	14 s/img
RCNN	66.0	0.05FPS	20 s/img
Fast RCNN	70.0	0.5FPS	2 s/img
Faster RCNN	73.2	7FPS	140 ms/img

Table: Comparison of Object Detection Algorithms

If we use Faster RCNN, the car moves $2.33m$ before halting on seeing an object. Thus arises a need for real-time object detection system.

Introduction

Here comes YOLO - You Only Look Once.



YOLO is refreshingly simple. A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes.

Introduction

This unified model has several benefits over traditional methods of object detection.

- YOLO is extremely fast since it avoids a complex pipeline. We can simply run the forward pass of convolutional network at test time to detect predictions.
- YOLO reasons globally about the image when making predictions. This is because YOLO sees the entire image during training and test time.
- YOLO generalizes well. The reason is lesser number of parameters (less complex model). Hence, YOLO can be applied to new domains.

Concept of Unified Model

Start with an input Image

Concept of Unified Model

Divide the image into an $S \times S$ grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.

Concept of Unified Model

Each cell predicts B bounding boxes and confidence score for each box. These confidence scores indicate how confident the model is that the box contains an object and how accurate it thinks the box is that it predicts.

Concept of Unified Model

This confidence score can be modeled mathematically as
$ConfidenceScore = Pr(Object) * IOU_{pred}^{truth}$

Concept of Unified Model

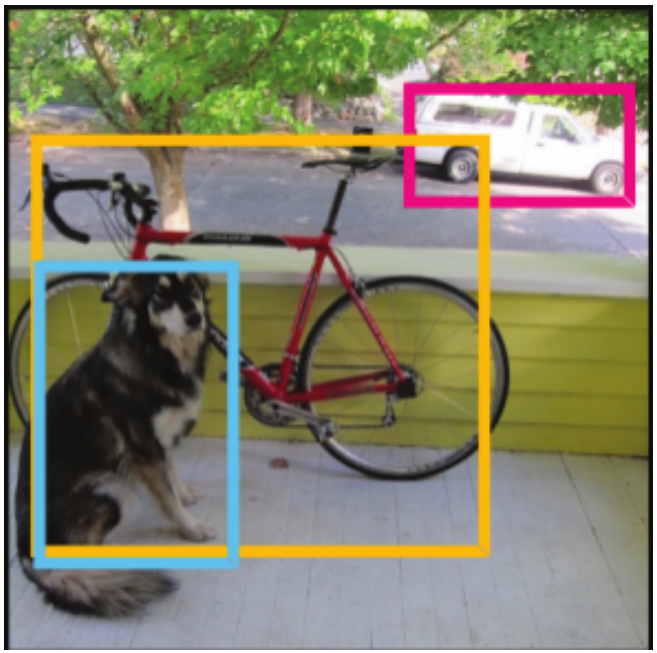
Each bounding box consists of 5 predictions: x, y, w, h, and confidence. The (x, y) coordinate represents the center of the box relative to the grid-cell. The width w and height h are predicted relative to the whole image.

Concept of Unified Model

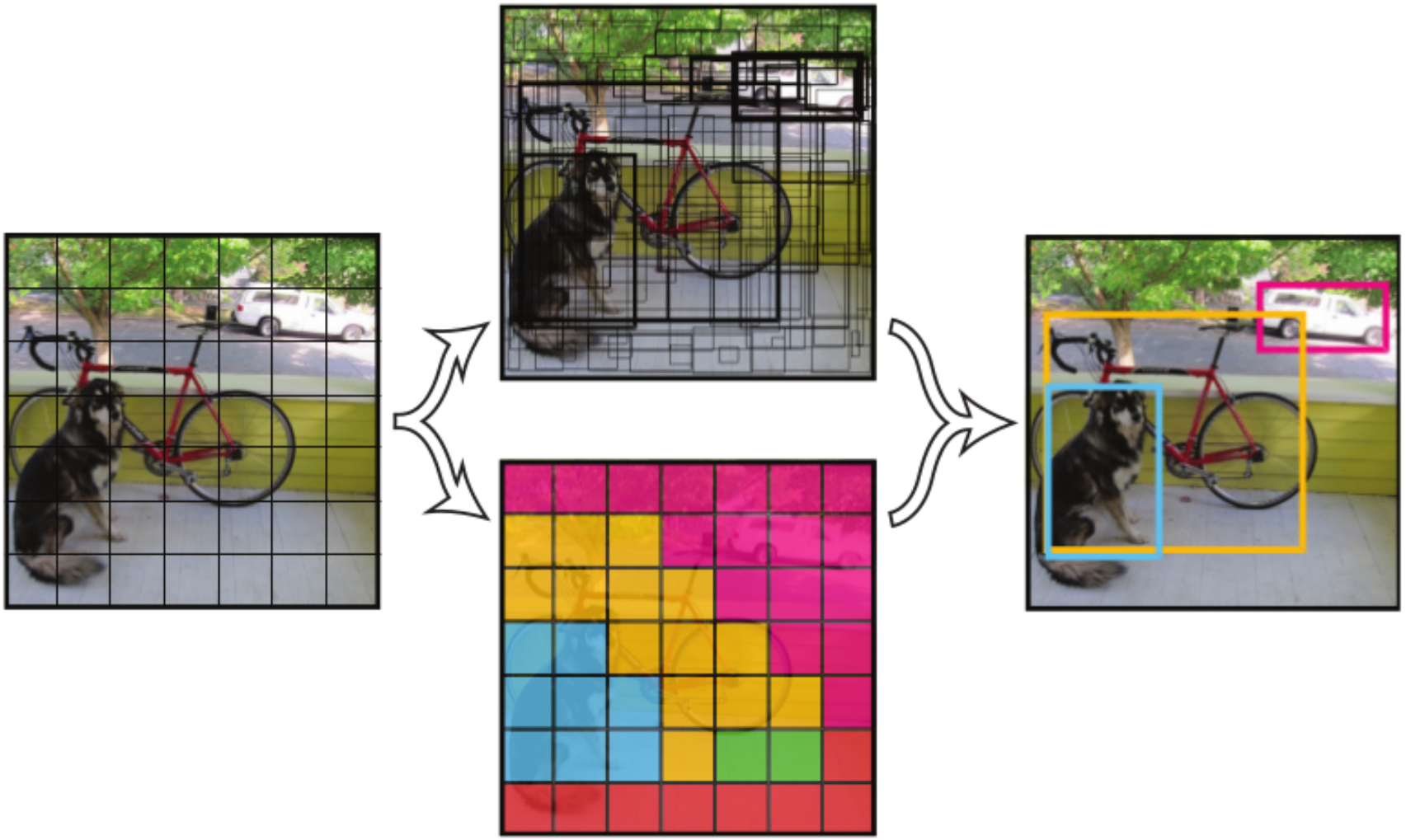
Each cell also predicts C conditional class probabilities. $Pr(Class_i|Object)$. These probabilities are conditioned on the grid cell containing an object. Only one set of class probabilities are predicted regardless of number of boxes B.

Concept of Unified Model

At test time, we multiply the conditional class probabilities and the individual box confidence predictions. Mathematically,
$Pr(Class_i|Object) * Pr(Object) * IOU_{pred}^{truth} = Pr(Class_i) * IOU_{pred}^{truth}$



Concept of Unified Model



Concept of Unified Model

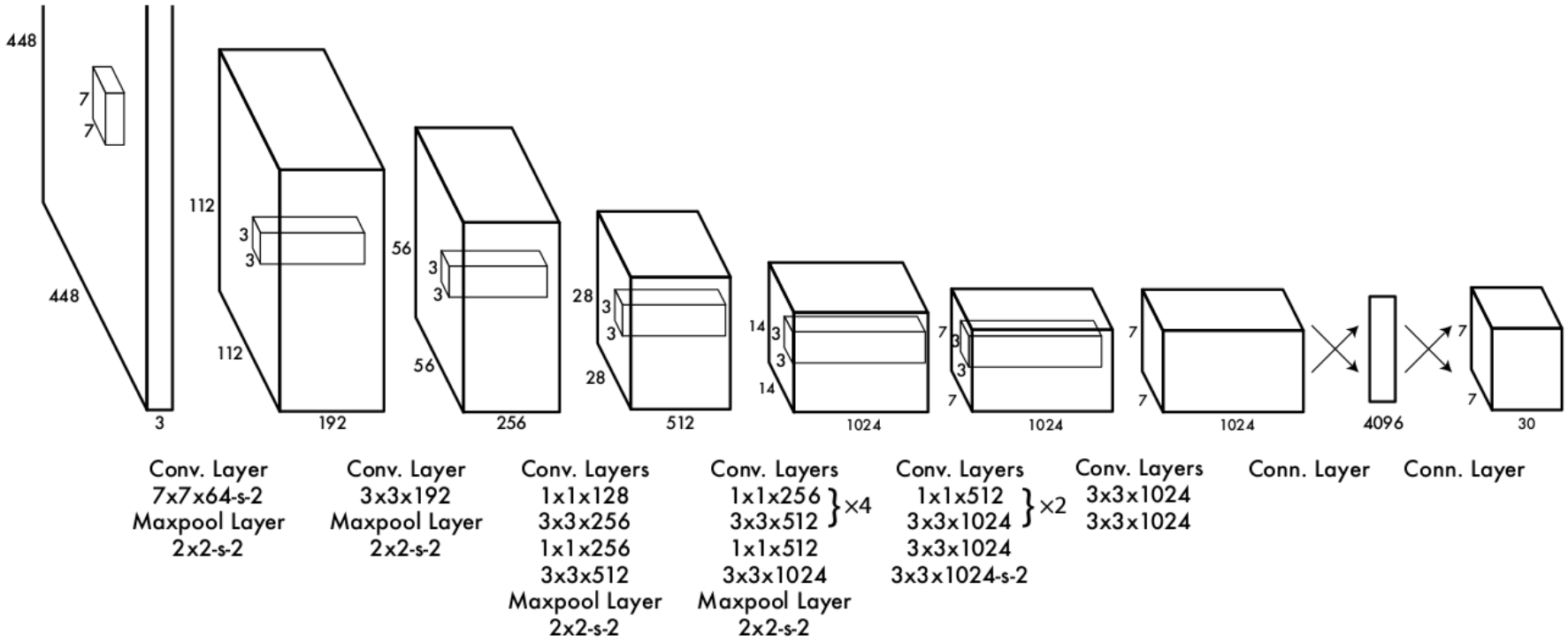
In a short summary, divides the image into an even grid and simultaneously predicts bounding boxes, confidence in those boxes, and class probabilities.These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.
For evaluating YOLO on PASCAL VOC, The values used are $S = 7, B = 2$. PASCAL VOC has 20 classes. So final prediction vector is of size $7 \times 7 \times 30$

Design

The network designed is similar to GoogLeNet model for image classification.

- Network has 24 convolutional layers followed by 2 fully connected layers
- However, the inception module is replaced by simple 1×1 reduction layers followed by 3×3 convolutional layers.
- The final output of the network is the $7 \times 7 \times 30$ tensor of predictions.

Design


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

Training

- The first 20 convolutional layers are pre-trained on ImageNet by adding an average pooling and FC layer for predictions.
- Then four convolutional layers and two FC layers are additionally added and weights are randomly initialized.
- Since Object Detection requires fine-grained visual information, so input dimensions are increased from 224×224 to 448×448

Training

- The bounding box width and height are normalized by image width and height to make them in range $[0, 1]$.
- Similarly, x and y are offsets of grid-cell and hence also lie in $[0, 1]$
- Leaky ReLU activation function is used. $\phi(x) = \begin{cases} x & if\ x > 0 \\ 0.1x & o.w. \end{cases}$

Training

Sum of squares loss function can be used. But it has the following problems

- Weighs Localization error equally with classification error which may not be ideal.
- Also many grid cells do not generally contain objects and hence heavily contribute to the loss function compared to grid cells containing objects.
- Additionally, weighs localization errors in small boxes and large boxes equally.

Training

The following are the modifications to be done on the squared error loss function.

- Increase the loss from bounding box co-ordinates prediction.
- Decrease the loss from confidence predictions of boxes that don't contain objects.
- Use square root of width and height while computing loss function.

Training

$$
\begin{aligned}
Loss \quad = \quad & \lambda_{coods} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{i,j}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
& + \lambda_{coods} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{i,j}^{obj} \left[(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \\
& + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{i,j}^{obj} (c_i - \hat{c}_i)^2 \\
& + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{i,j}^{noobj} (c_i - \hat{c}_i)^2 \\
& + \sum_{i=0}^{S^2} \mathbb{I}_{i}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2
\end{aligned}
$$

Inference

- Compute the product of class conditionals and confidence score for each bounding box and use thresholding to get the object bounding boxes.
- Can use non-maximal suppression to fix the same object being detected by multiple bounding boxes (duplicate detections).

Results

Algorithm	Pascal 2007 mAP	Frame Rate	Speed
DPM v5	33.7	0.07FPS	14 s/img
RCNN	66.0	0.05FPS	20 s/img
Fast RCNN	70.0	0.5FPS	2 s/img
Faster RCNN	73.2	7FPS	140 ms/img
YOLO	63.4	45FPS	22 ms/img

Table: Comparison of Object Detection Algorithms

If we use YOLO, the car moves $0.366m$ before halting on seeing an object.

Results

Algorithm	**Pascal 2007 mAP**	**Frame Rate**	**Speed**
DPM v5	33.7	0.07FPS	14 s/img
RCNN	66.0	0.05FPS	20 s/img
Fast RCNN	70.0	0.5FPS	2 s/img
Faster RCNN	73.2	7FPS	140 ms/img
YOLO	63.4	45FPS	22 ms/img
YOLO + Fast RCNN	75.0	0.5FPS	2 s/img

Table: Comparison of Object Detection Algorithms

Results

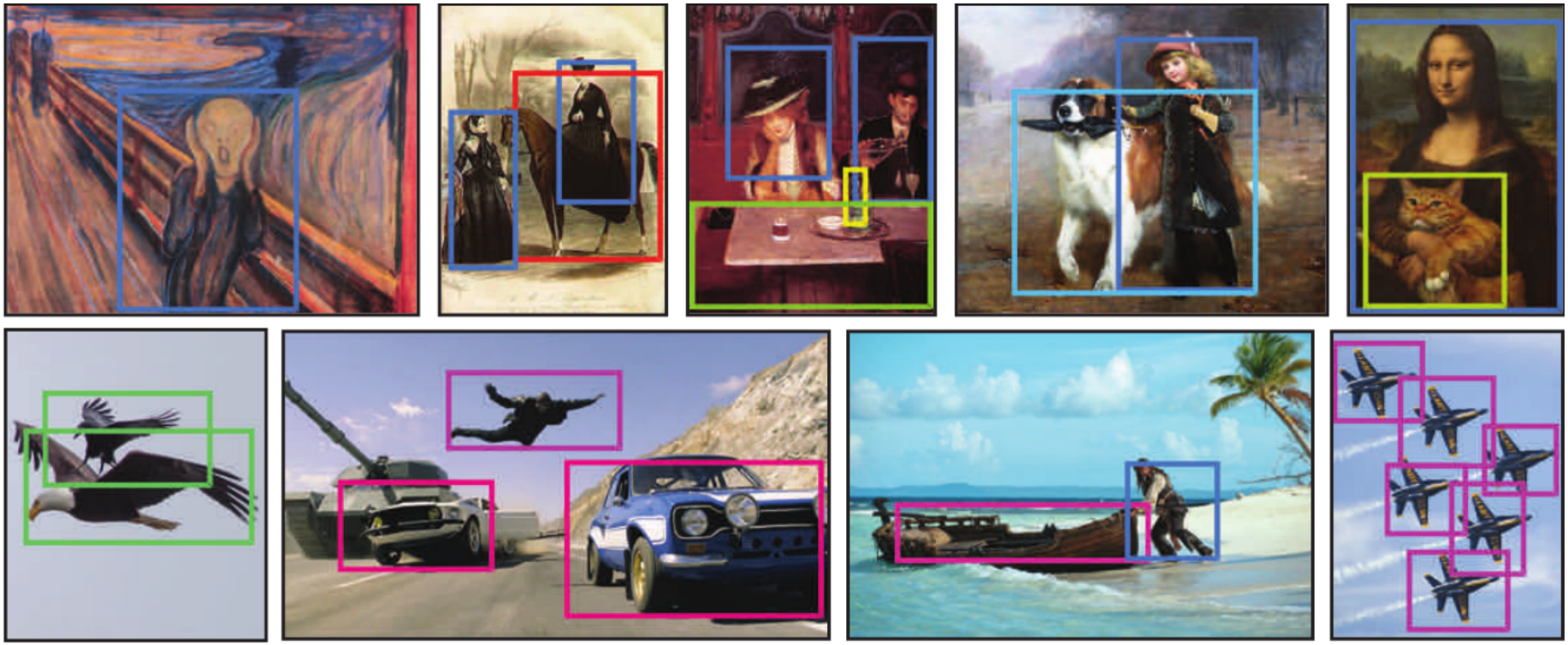
Algorithm	Pascal 2007 mAP	Frame Rate	Speed
Fast RCNN	70.0	0.5FPS	2 s/img
YOLO	63.4	45FPS	22 ms/img
YOLO + Fast RCNN	75.0	0.5FPS	2 s/img

Table: Comparison of Object Detection Algorithms

Combining YOLO and Fast RCNN gives a significant boost in performance of the model while taking the same time as that of Fast RCNN. For every bounding box that R-CNN predicts we check to see if YOLO predicts a similar box and boost its score if it does. This happens because YOLO makes far fewer background mistakes compared to Fast RCNN. (attributed to taking the whole image each time)

Generalizability

YOLO applied on Picasso and People art



Limitations

- Struggles with small objects
- Struggles to generalize for new aspect ratios of objects.
- Number of bounding boxes B restricts the number of nearby objects that can be predicted.
- Not the best loss function.

References

Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi

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

The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788.

The End