Object Detection with CNN part 2

Roberto Paredes

Centro de Investigación
Pattern Recognition and Human Language Technologies
Universidad Politécnica de Valencia

Index

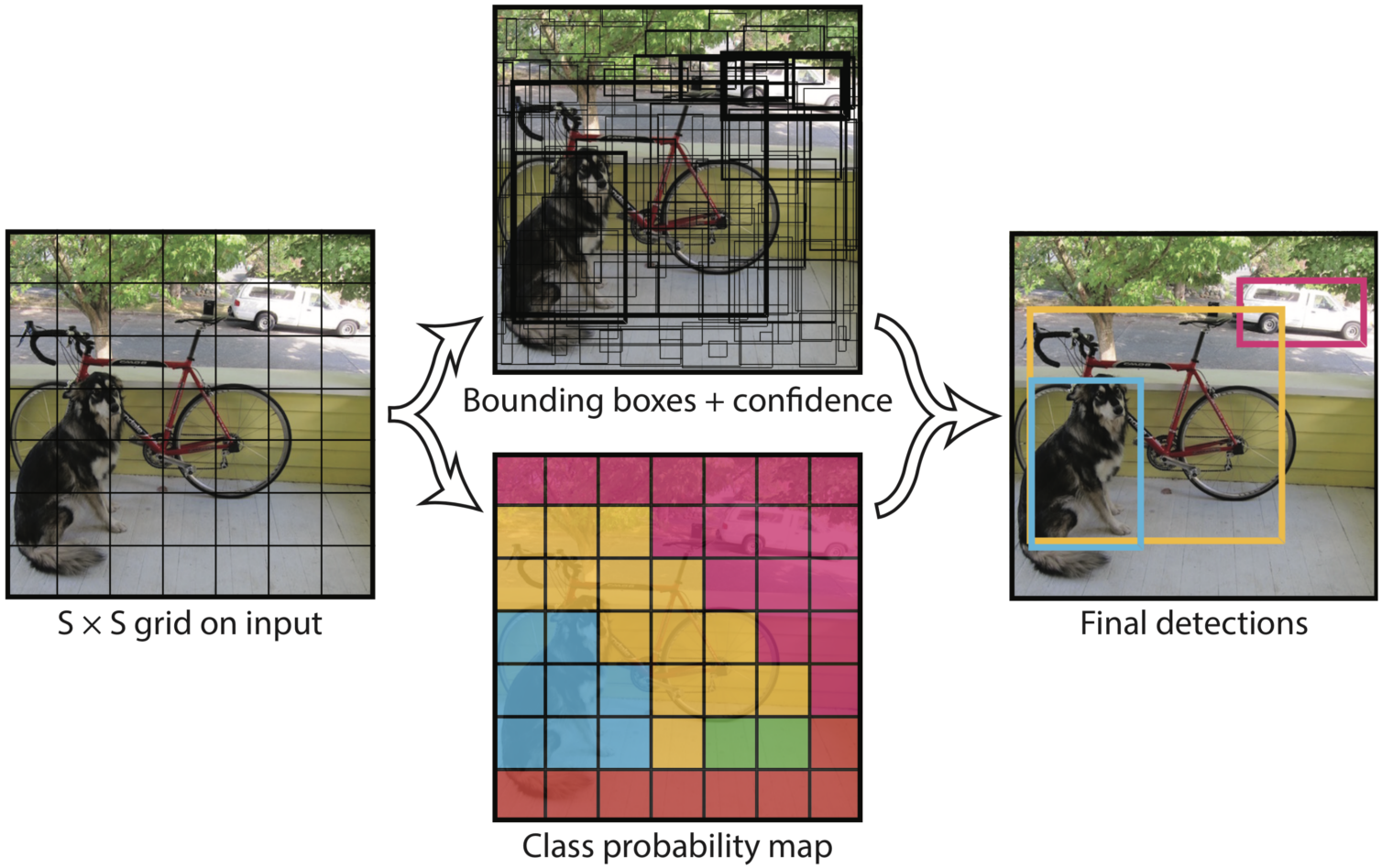
- Yolo
- SSD
- Yolo v2

You Only Look Once (YOLO)

- An unified, real-time object detection
- object detection as a regression problem
- you only look once at an image to predict what objects are present and where they are
- A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes
- Straightforward ent-to-end training
- the entire image is processed during training and test time, implicitly encode:
  - contextual information about classes
  - their appearance

YOLO

- Unified Detection



YOLO

- YOLO divides the input 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
- Each grid cell predicts B bounding boxes and confidence scores for those boxes
- Each bounding box consists of 5 predictions: x, y, w, h, and confidence:
  - The (x, y) coordinates represent the center of the box relative to the bounds of the grid cell
  - The width and height are predicted relative to the whole image
  - The confidence prediction represents the IOU between the predicted box and any ground truth box: $Pr(Object) * IOU_{pred}^{trhth}$

YOLO

- Also for each cell C conditional class probabilities are predicted:

$$Pr(Class_i \mid Object)$$

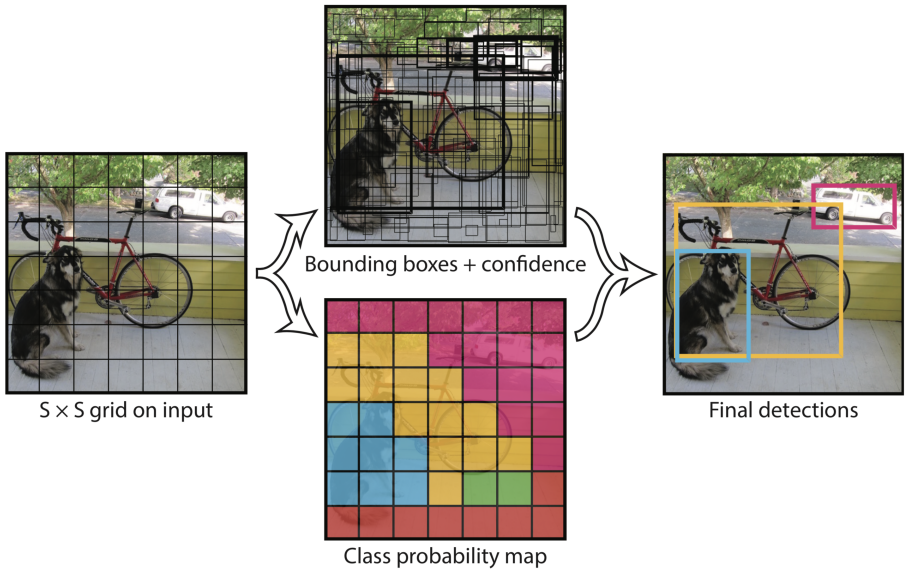
- At test time:

$$Pr(Class_i \mid Object) * Pr(Object) * IOU_{pred}^{trhth}$$

- These scores encode both:
  - the probability of that class appearing in the box
  - how well the predicted box fits the object

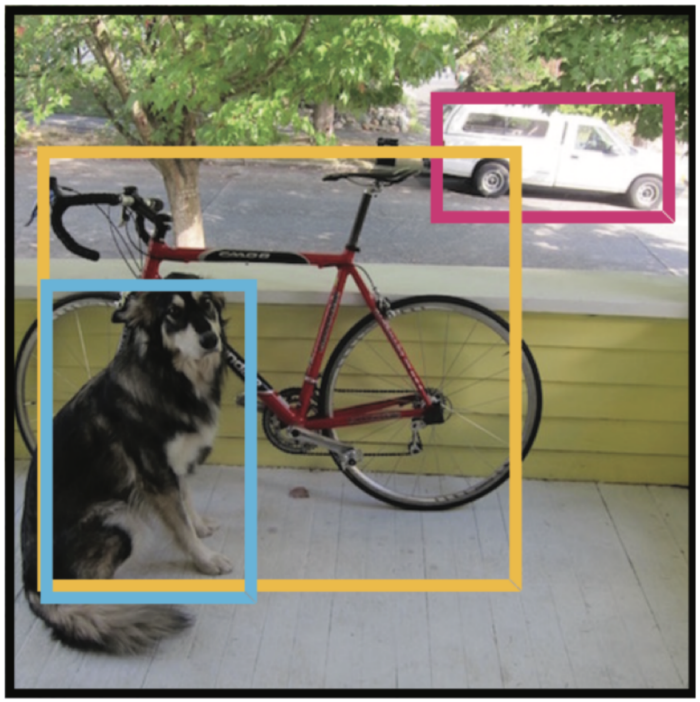
YOLO Summary

- divides the image into an $S \times S$ grid
- for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities
- These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.
- Example, on PASCAL VOC, $S = 7$, $B = 2$, $C = 20$, therefore a $7 \times 7 \times 30$ tensor



YOLO Training

IN TRAINING TIME



7x7x30 Tensor

[0, 0, 0, 0, ... 0.7,0.7,...0,0,0,]

YOLO Training

- If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.
- YOLO predicts B bounding boxes per grid cell. However, at training time we only want one bounding box predictor to be responsible for each object
- The one which prediction has the highest current IOU with the ground truth
- This leads to specialization between the bounding box predictors

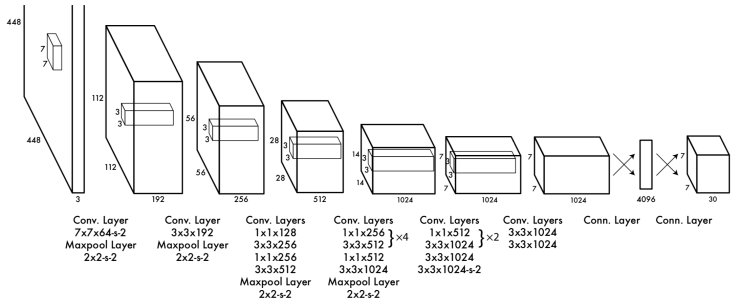
YOLO Training

- Finally, a multi-loss is minimized:

$$
\begin{aligned}
&\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2
\end{aligned}
$$

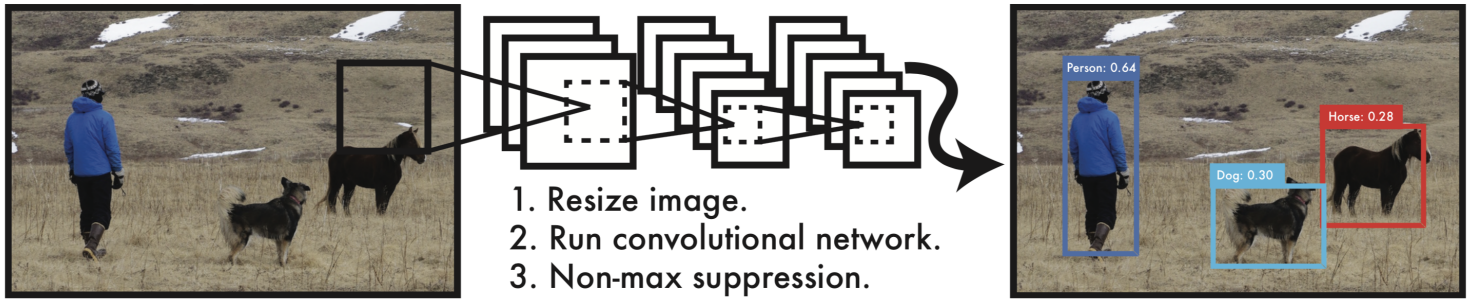
YOLO Training

- CCN inspired in GoogleNet without inception and $1x1$ convs (24 Conv Layers)



- Fast-Yolo (9 Conv Layers)
- Imagenet pre-training (with only 20 Conv)
- Add two conv and two FC Layers for detection and 448×448 input

YOLO Inference



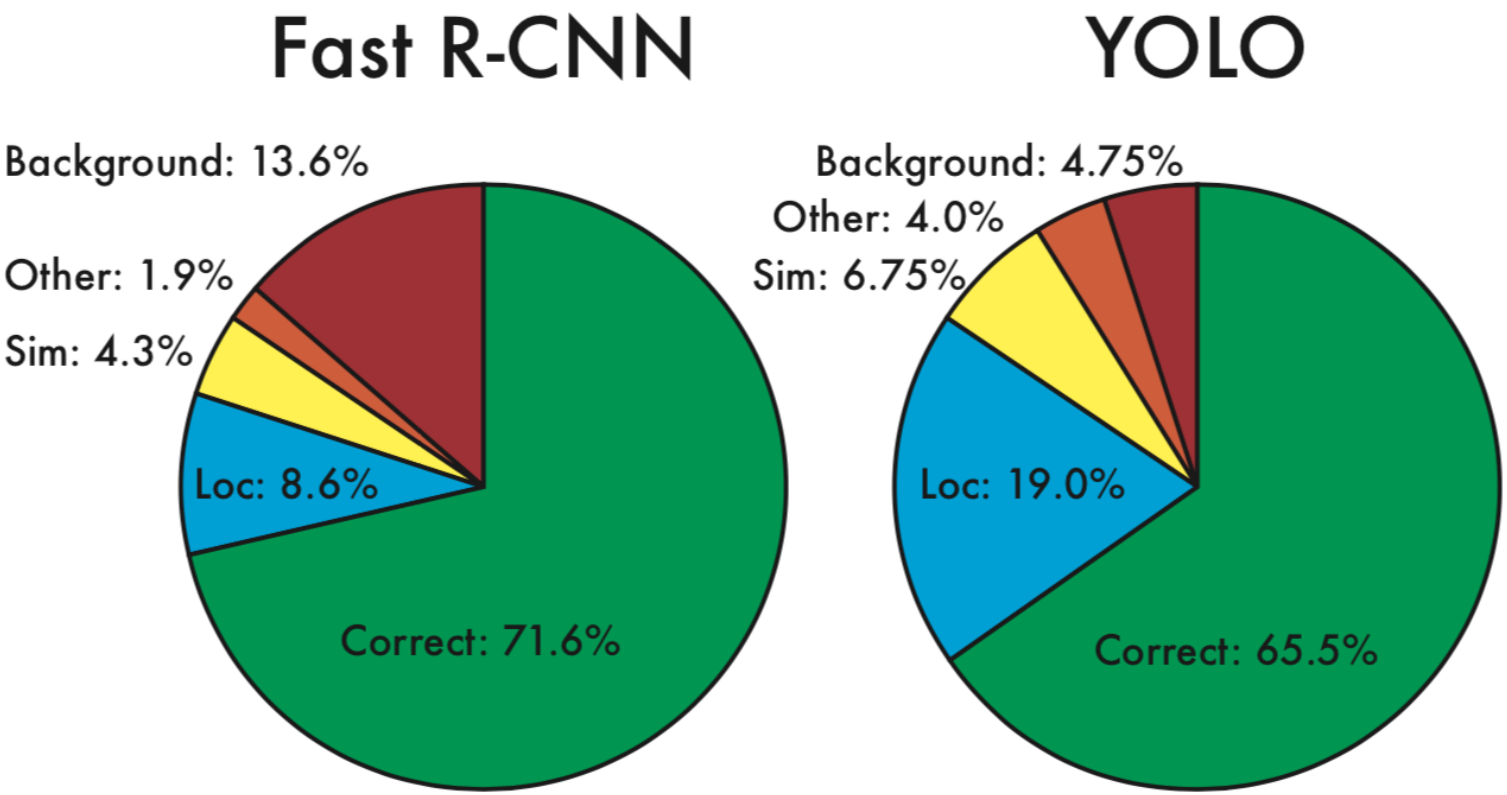
YOLO Inference Details

- Predicting detections for a test image only requires one network evaluation
- Non-maximal suppression is used to fix these multiple detections
- loss function treats errors the same in small bounding boxes versus large bounding boxes
- main source of error is incorrect localizations

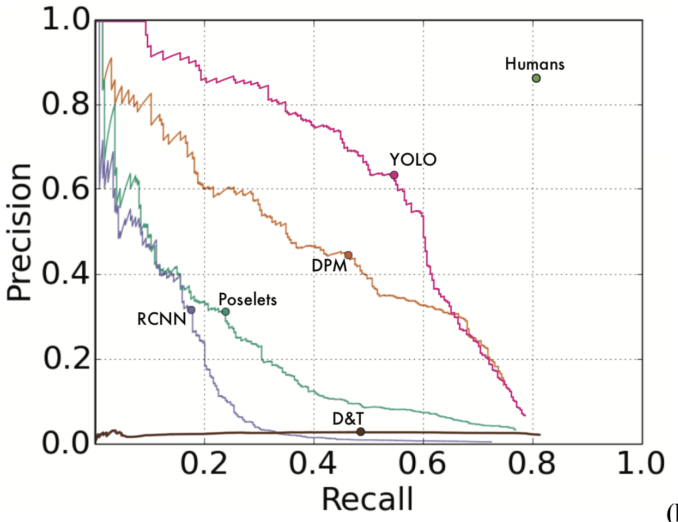
YOLO Results

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 Comparison



YOLO Generalization

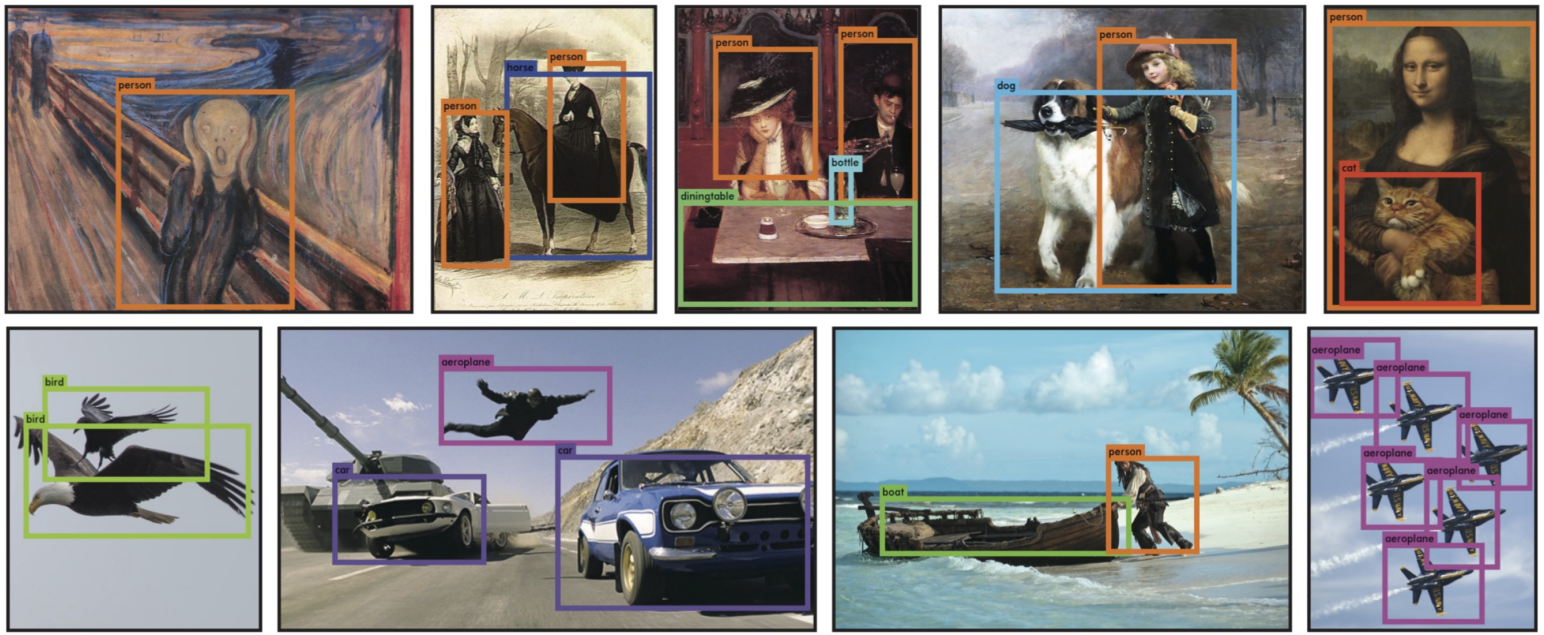


(a) Picasso Dataset precision-recall curves.

	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	

(b) Quantitative results on the VOC 2007, Picasso, and People-Art Datasets. The Picasso Dataset evaluates on both AP and best F_1 score.

YOLO Generalization

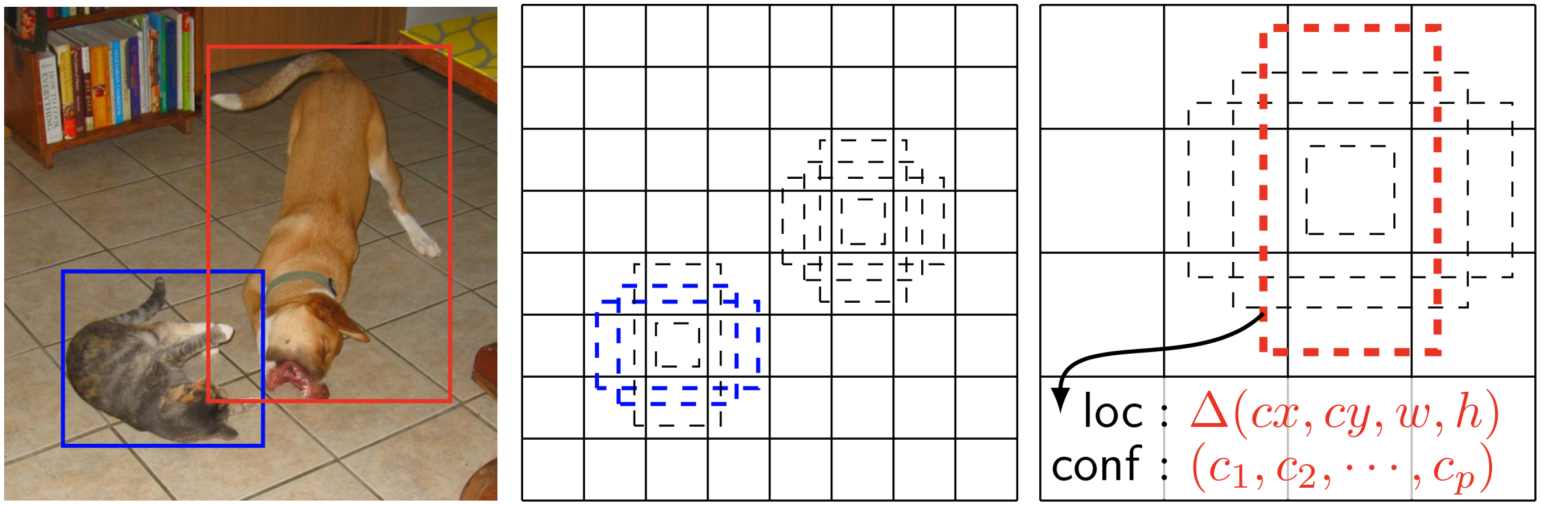


SSD: Single Shot MultiBox Detector

- Similar end-to-end procedure
- A small convolutional filter to predict object categories and offsets in bounding box locations
- Separate predictors (filters) for different aspect ratio detections
- Using multiple layers for prediction at different scales
- Improves accuracy on real-time detection for PASCAL VOC from 63.4% mAP for YOLO to 74.3%

SSD: Single Shot MultiBox Detector

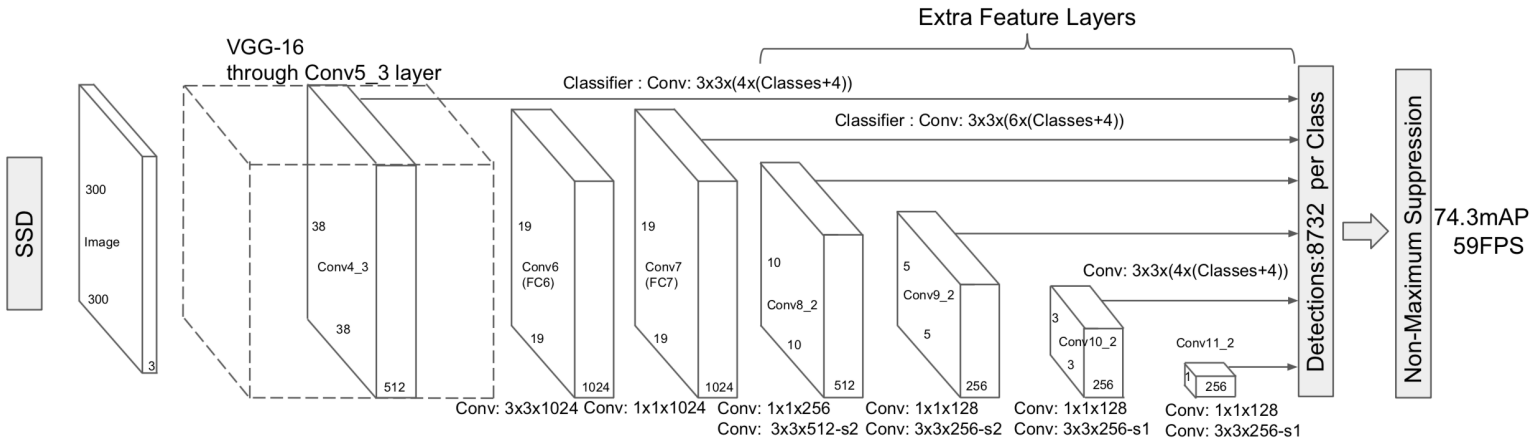
- In training time default boxes K are matched to the ground truth boxes in **different** scales
- The default boxes are *positives* or *negative*



- The model loss is a weighted sum:
  - localization loss (regression)
  - confidence loss (softmax)

SSD: Architecture

- Truncated basic CNN, trained on image recognition
- Add conv. layers that decrease in size progressively
- Each added conv. layer can produce a fixed set of detection predictions
- Thus, each added conv. layer is connected to the loss function



SSD: Training

- End-to-end training with the $(C + 4) \times K$ target values for each map position
- Matching ground truth:
  - Best Jaccard index *OR*
  - Jaccard index > 0.5, allowing multiple matchings
- Loss function for N boxes:

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g))$$

- x_{ij}^{p} is an indicator for matching the $i - th$ default box to the $j - th$ ground truth box of category p
- Data Augmentation

SSD: Results

- VGG-16 as basic CNN
- pascal voc 2007 and 2012

Method	data	mAP
Fast [6]	07	66.9
Fast [6]	07+12	70.0
Faster [2]	07	69.9
Faster [2]	07+12	73.2
Faster [2]	07+12+COCO	78.8
SSD300	07	68.0
SSD300	07+12	74.3
SSD300	07+12+COCO	79.6
SSD512	07	71.6
SSD512	07+12	76.8
SSD512	07+12+COCO	**81.6**

Method	data	mAP
Fast[6]	07++12	68.4
Faster[2]	07++12	70.4
Faster[2]	07++12+COCO	75.9
YOLO[5]	07++12	57.9
SSD300	07++12	72.4
SSD300	07++12+COCO	77.5
SSD512	07++12	74.9
SSD512	07++12+COCO	**80.0**

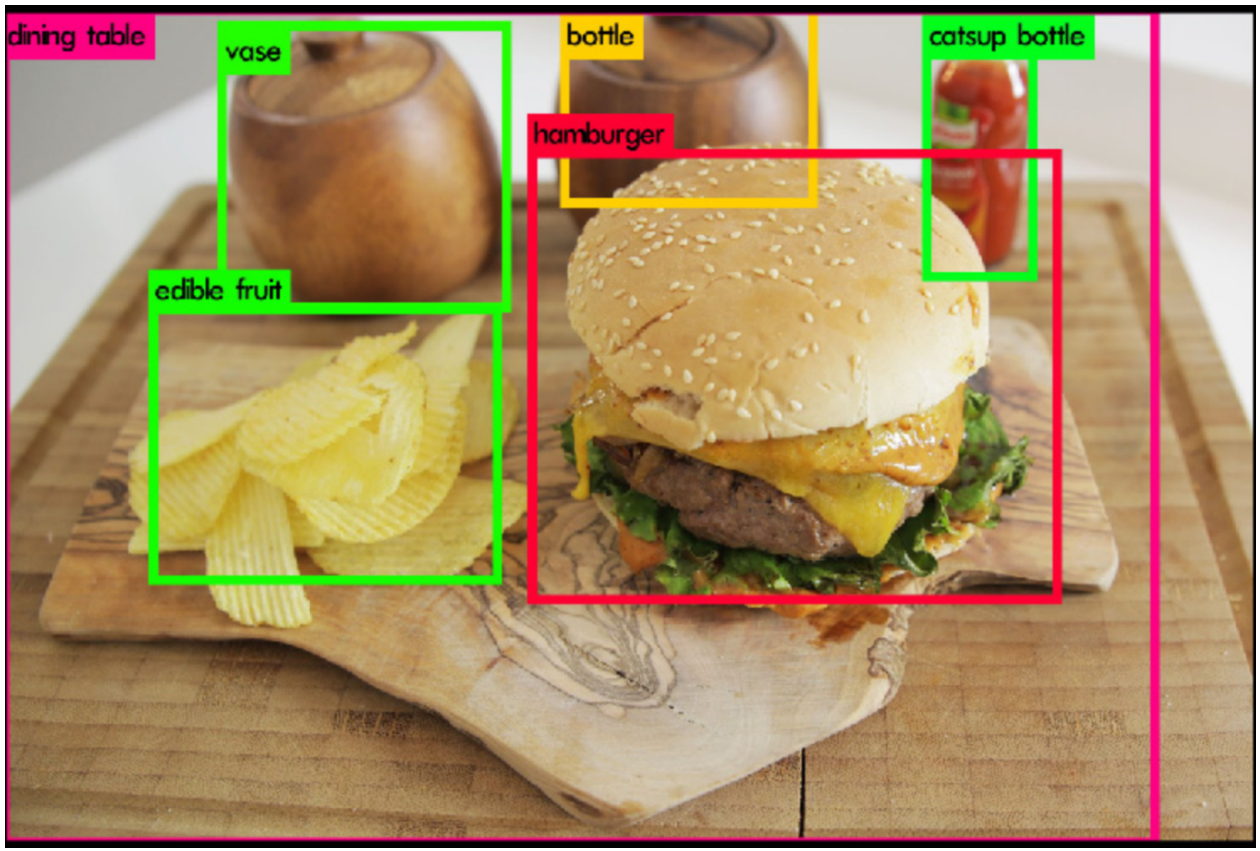
SSD: Results

- mAP and inference time

Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	$\sim 1000 \times 600$
Fast YOLO	52.7	155	1	98	448×448
YOLO (VGG16)	66.4	21	1	98	448×448
SSD300	74.3	46	1	8732	300×300
SSD512	76.8	19	1	24564	512×512
SSD300	74.3	59	8	8732	300×300
SSD512	76.8	22	8	24564	512×512

YOLO v2

- Yolo 9000: Better, Faster, Stronger
- Particular improvements to deal with multi-scaling and localization errors
- But also deal with more categories

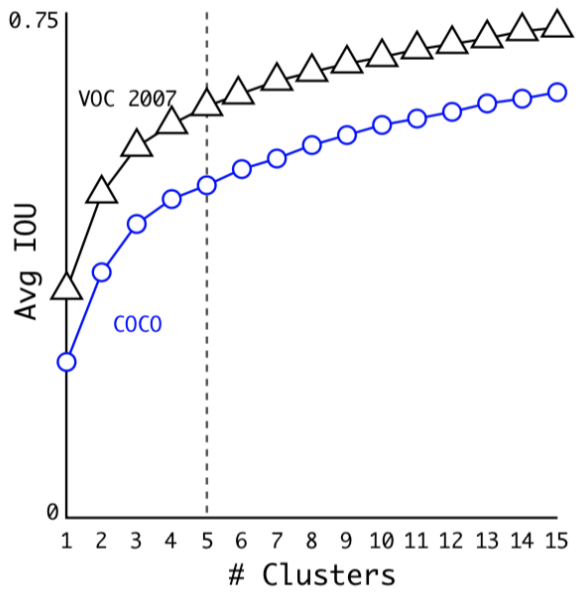


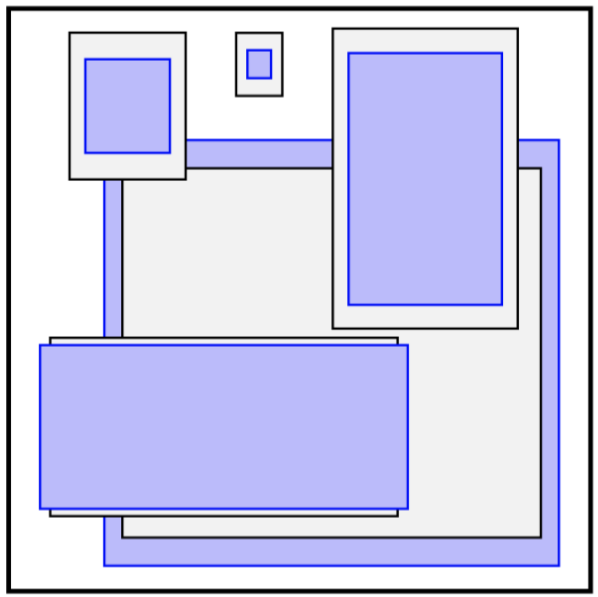
YOLO v2 - improvements

- Yolo shortcomings:
  - significant number of localization errors
  - low recall compared to region proposal-based methods
- New ideas and potential improvements:
  - Batch Normalization, 2% of mAP
  - High Resolution Classifier, from 224×224 to 448×448
  - Convolutional with **anchor boxes**, predict offsets instead of coordinates (better recall)
  - Dimension Clusters

YOLO v2 - Dimension Clusters

- Better box sizes as initial guess
- Instead of fixed sizes, start with better priors
- Run k-mean over training set of bounding boxes
- K-means with appropriate distance to improve IOU



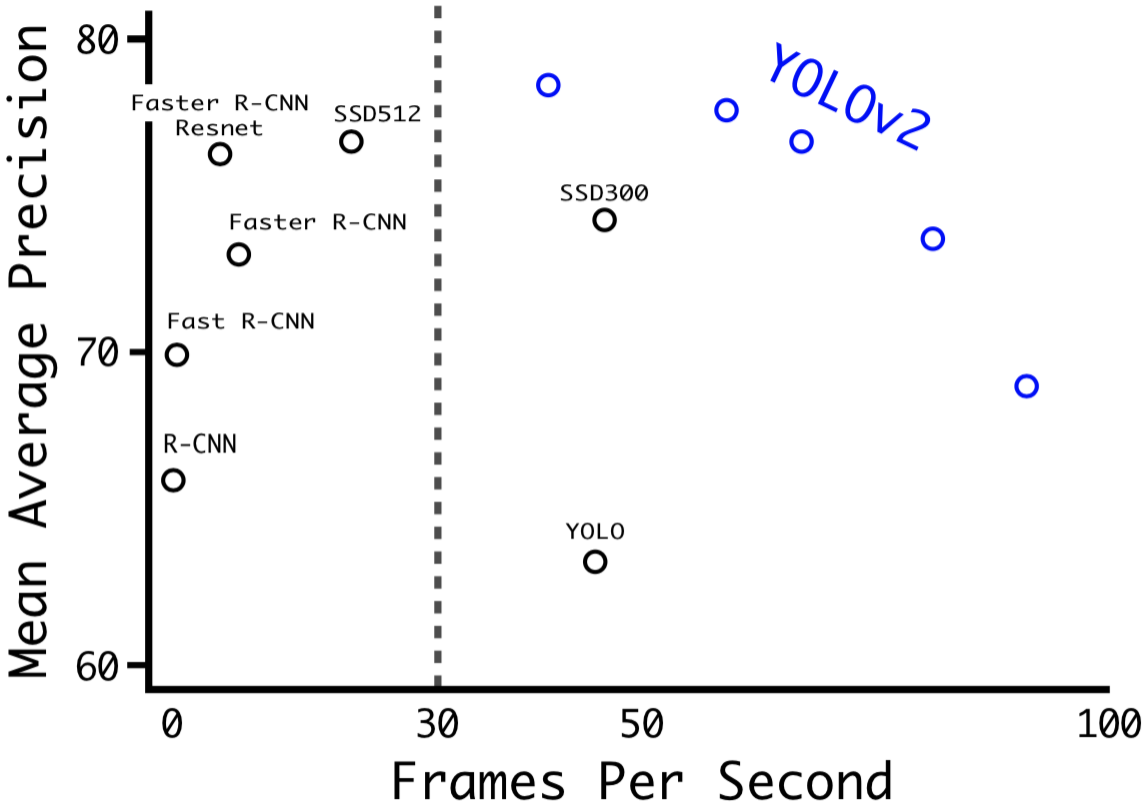


YOLO v2 - more improvements

- Direct location prediction, predict location coordinates relative to the location of the grid cell
- Fine-grained features, resolution, 13×13 **AND** 26×26
- Multi-Scale Training, different input sizes, from 320×320 to 608×608

YOLO v2 - Results

- mAP vs FPS



YOLO v2 - Results

- mAP comparison

Detection Frameworks	Train	mAP	FPS
Fast R-CNN [5]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[15]	2007+2012	73.2	7
Faster R-CNN ResNet[6]	2007+2012	76.4	5
YOLO [14]	2007+2012	63.4	45
SSD300 [11]	2007+2012	74.3	46
SSD500 [11]	2007+2012	76.8	19
YOLOv2 288×288	2007+2012	69.0	91
YOLOv2 352×352	2007+2012	73.7	81
YOLOv2 416×416	2007+2012	76.8	67
YOLOv2 480×480	2007+2012	77.8	59
YOLOv2 544×544	2007+2012	**78.6**	40

YOLO v2 - Faster

- Yolo uses GoogleNet 8.52 billion floating point operations
- instead of typical VGG16 30.69 billion floating point operations
- Darknet-19, 5.58 billion floating point operations

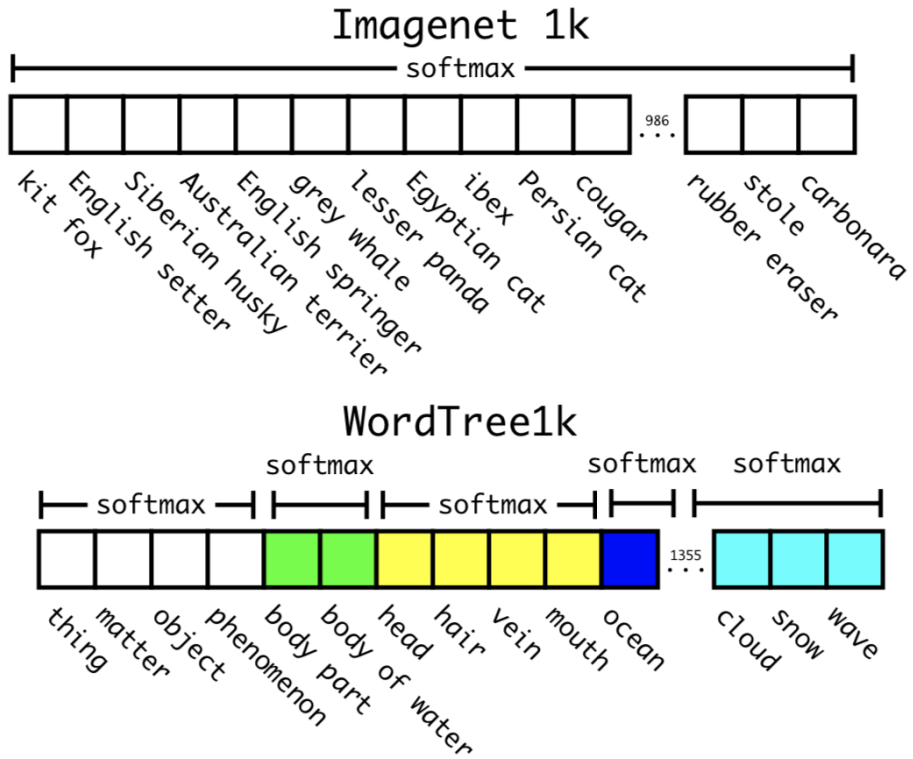
Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

YOLO v2 - Stronger

- Joint training on classification and detection data
- images labelled for detection to learn detection-specific information like bounding box coordinate prediction and objectness
- images with only class labels to expand the number of categories it can detect
- During training Yolov2 mixes images from both detection and classification datasets
- image labelled for detection can backpropagate based on the full YOLOv2 loss function
- classification image we only backpropagate loss from the classification-specific parts of the architecture

YOLO v2 - Stronger

- Problem when classification labels are not mutually exclusive (ImageNet)
- e.g., "Norfolk terrier" (from ImageNet) and "dog" (from COCO)
- ImageNet classes from WordNet
- Solution: Build a hierarchical model of visual concepts,WordTree



YOLO v2 - Stronger

- Combine COCO classes with Imagenet classes