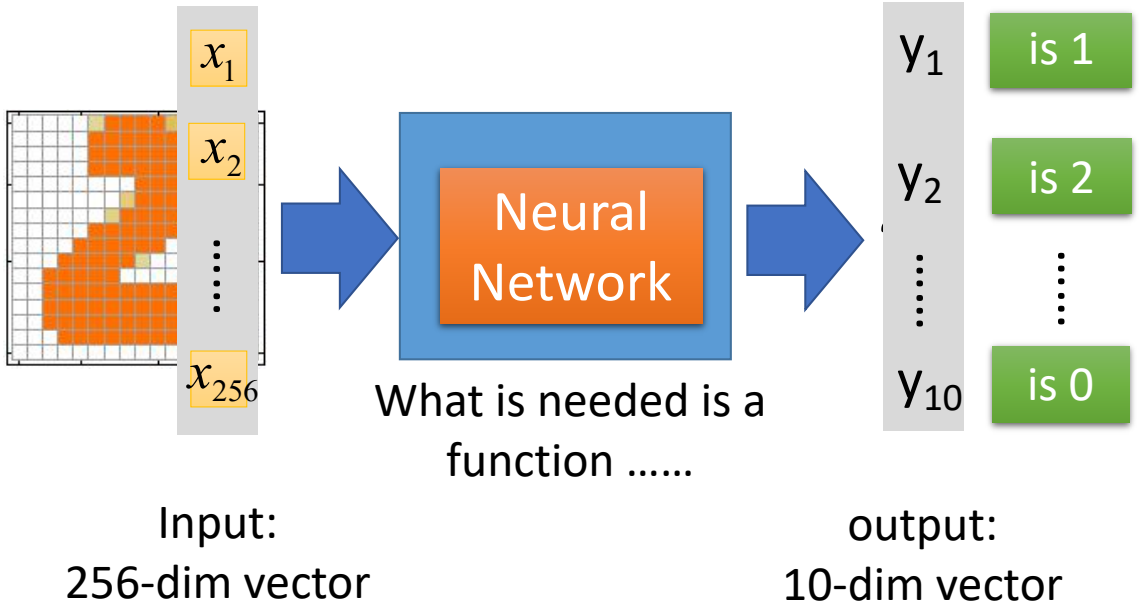
Object Detection with Deep Learning

Some Knowledge before Deep Learning

Classification and Regression

- Classification
  - Softmax
  - Cross Entropy



Classification and Regression

- Classification
  - **Softmax**
  - Cross Entropy

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \ldots, K.$$

EX.

1.0	0.0236405
2.0	0.0642617
3.0	0.174681
4.0	0.474833
1.0	0.0236405
2.0	0.0642617
3.0	0.174681

Classification and Regression

- Classification
  - Softmax
  - **Cross Entropy**

$$L = -\sum_{j=1}^{T} y_j log s_j$$

Ground truth label: 0 or 1

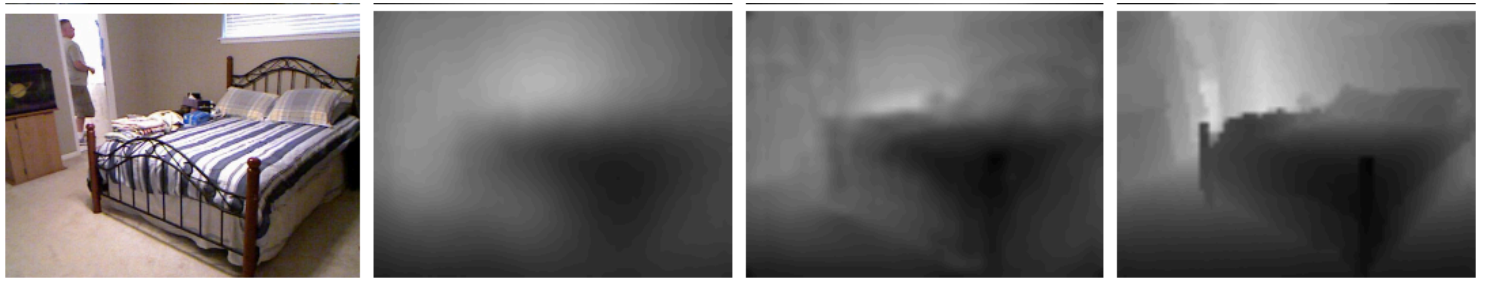
Softmax predict result: 0 ~ 1

Classification and Regression

- Regression

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \widehat{y}_i\right)^2 \qquad MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \widehat{y}_i\right|$$

An example:
Depth estimation



course network

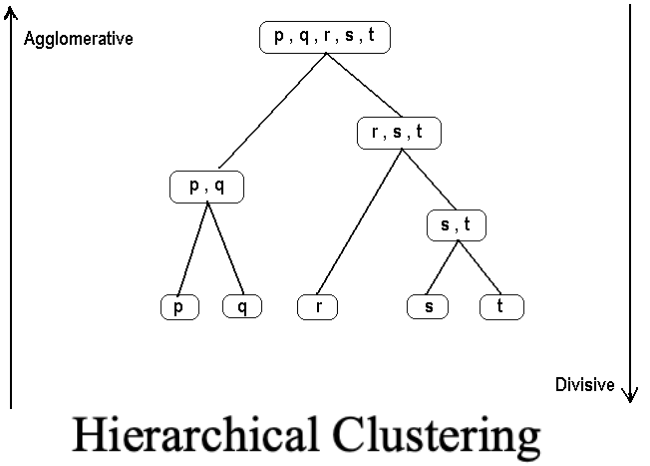
refined with
fine network

ground truth

Object Detection with Deep Learning

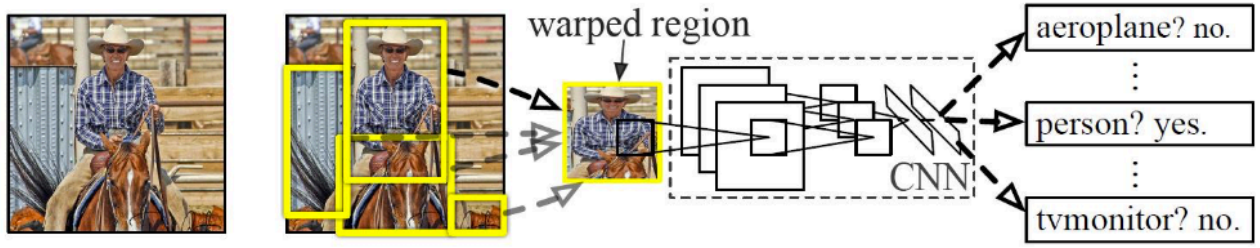
R-CNN: Regions with CNN Features

by UC Berkerley on CVPR 2014



Hierarchical Clustering

- **Region Proposal Extraction**: Selective Search
  - hierarchical clustering by color, texture, size & shape similarity
- **Feature Extraction**: CNN on every region proposal (with size normalization)
- **Classification**: SVM
- **Bounding Box Regression**: refine the position & size of bounding box
- **mAP**: 53.7% vs 35.1% (state-of-the-art)
- Running Time:
  - 53 s/image on CPU
  - 13 s/image on GPU

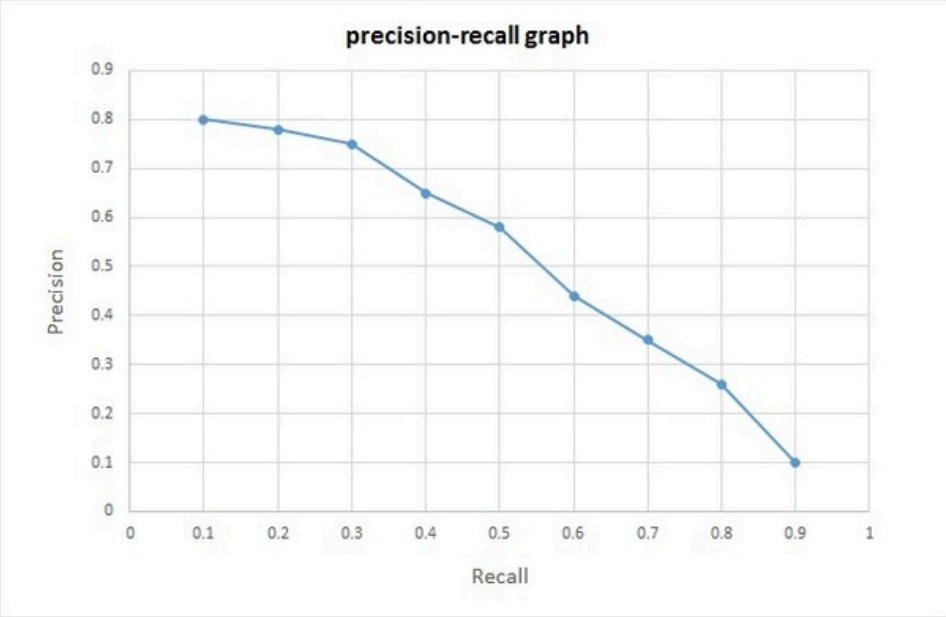


Ross Girshick, et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," CVPR 2014.

mAP: Mean Average Precision

- **AP**:
  - The average of the maximum precisions at different recall values.

- **mAP**:
  - The mean of APs of all classes.



Precision: measures how accurate is your predictions. i.e. the percentage of your positive predictions are correct.

$$\textbf{Precision} = \frac{\text{TP}}{\text{TP+FP}}$$

Recall: measures how good you find all the positives. For example, we can find 80% of the possible positive.

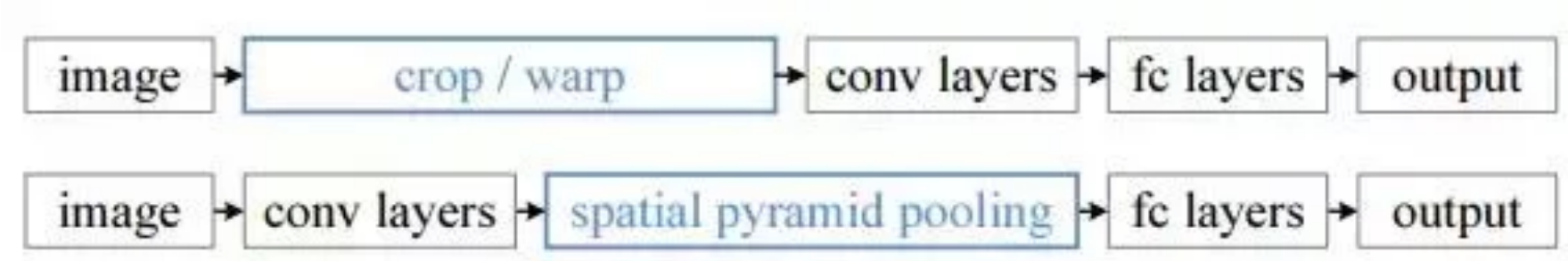
$$\textbf{Recall} = \frac{\text{TP}}{\text{TP+FN}}$$

Fast R-CNN

by Microsoft Research on ICCV 2015

- **Region Proposal Extraction**: Selective Search
- **Feature Extraction**: CNN on whole image
- **ROI Projection**
- **ROI Pooling**: inspired by SPP Net
- Classification: CNN
- Bounding Box Regression: CNN

Main difference with R-CNN:

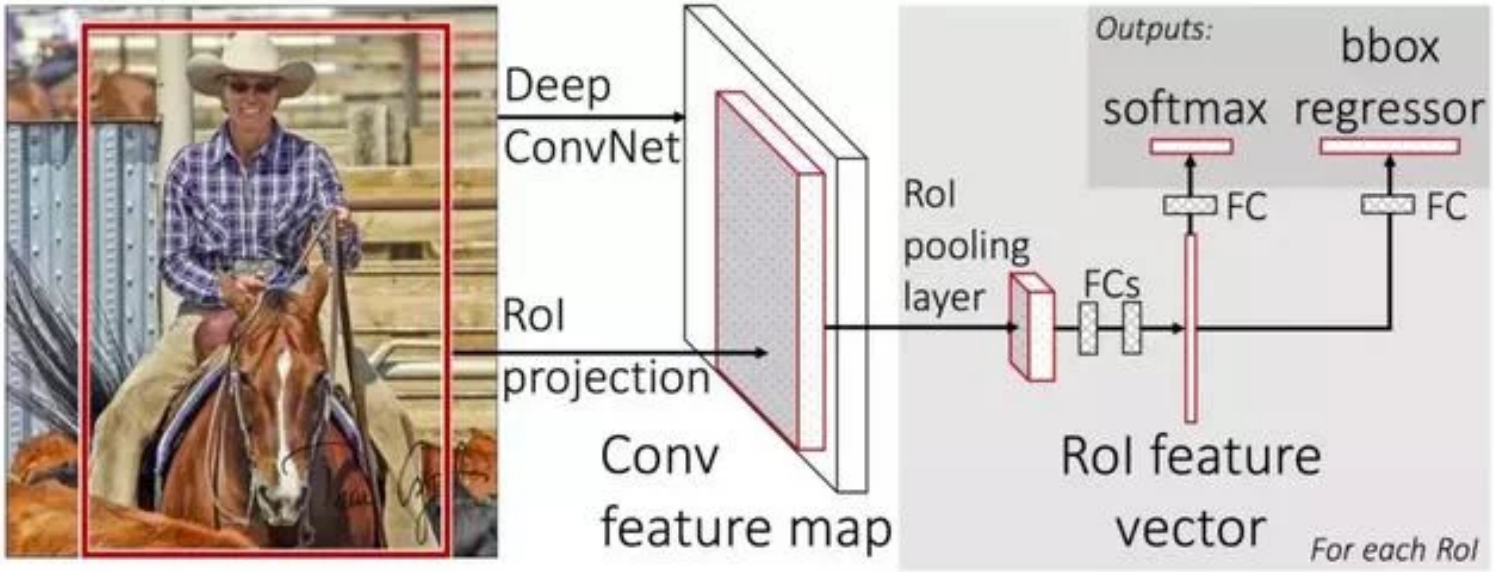


Ross Girshick, "Fast R-CNN," ICCV 2015.

Fast R-CNN

by Microsoft Research on ICCV 2015

- **Region Proposal Extraction**: Selective Search
- **Feature Extraction**: CNN on whole image
- **ROI Projection**
- **ROI Pooling**: inspired by SPP Net
- **Classification**: CNN
- **Bounding Box Regression**: CNN



Ross Girshick, "Fast R-CNN," ICCV 2015.

SPP Net vs ROI Pooling in Fast R-CNN

- **SPP Net**: Spatial Pyramid Pooling Network
  - Add a **spatial pyramid pooling layer** between the last convolution layer and the fully connected layer of a CNN.
- **Spatial Pyramid Pooling Layer**
  - Divide the input feature map into different number bins (e.g. 16, 4, 1), do the max pooling, then concatenate the result of all bins.
  - The input feature map can be arbitrary size.
- **ROI Pooling in Fast R-CNN**
  - Do spatial pyramid pooling only for 7x7 bins.

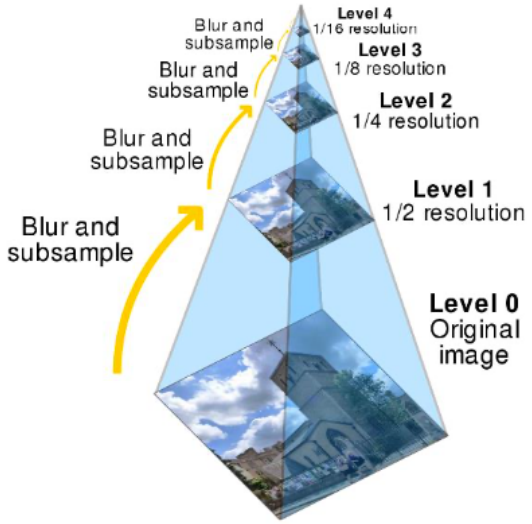
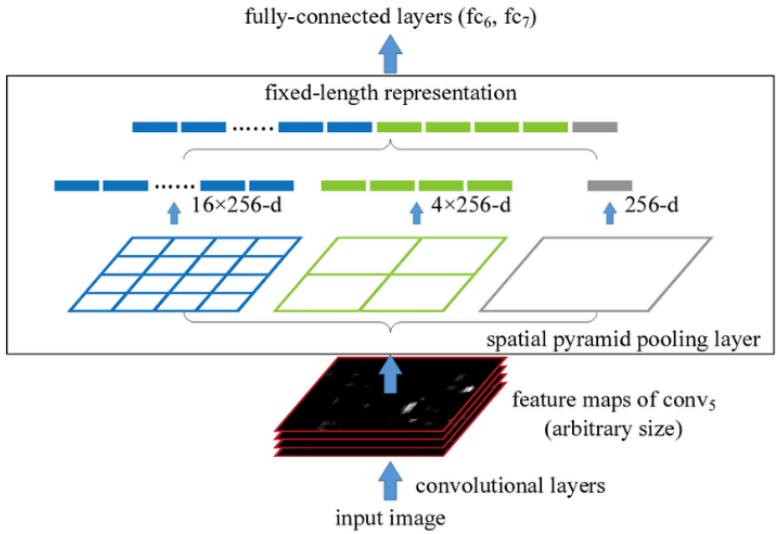


Image Pyramid



Kaiming He, et al., "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," ECCV 2014.

Fast R-CNN

- **mAP**: 68.4% vs 62.4% (R-CNN)
- **Running Time**:

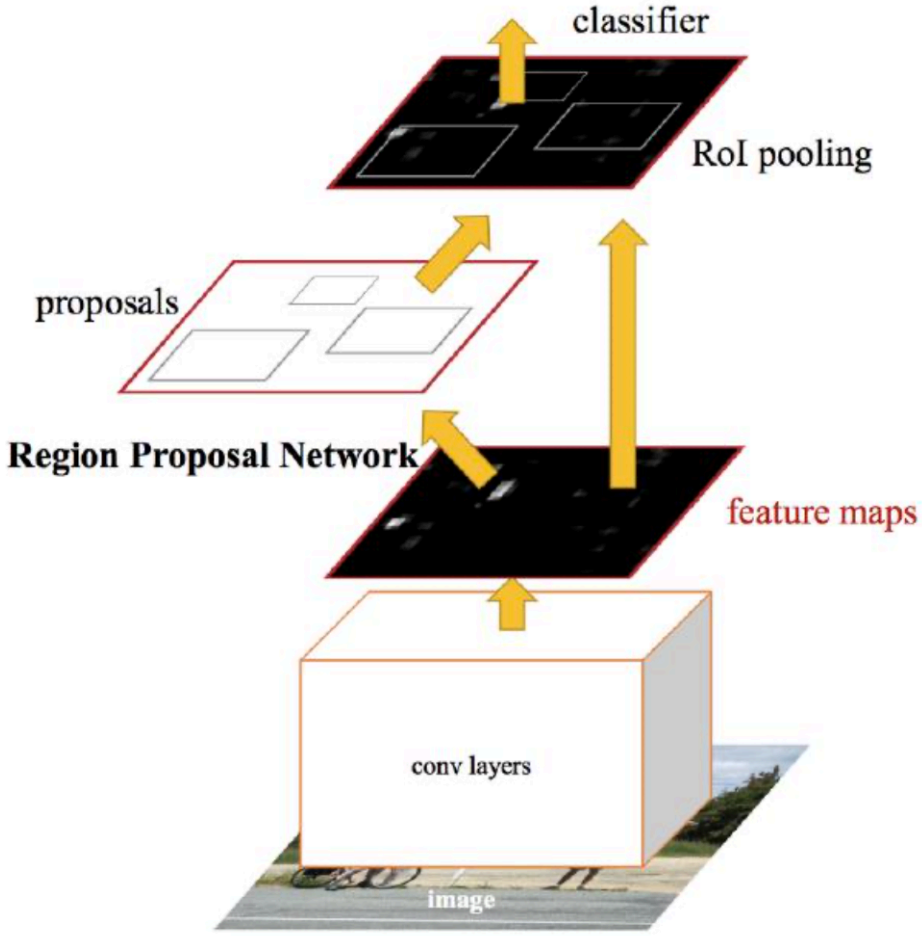
	Fast R-CNN			R-CNN		
	S	**M**	**L**	**S**	**M**	**L**
train time (h)	**1.2**	2.0	9.5	22	28	84
train speedup	**18.3**×	14.0×	8.8×	1×	1×	1×
test rate (s/im)	0.10	0.15	0.32	9.8	12.1	47.0
test speedup	98×	80×	146×	1×	1×	1×

S, M, L: Pre-trained ImageNet models
S: CaffeNet, **M**: VGG_CNN_M_1024, **L**: VGG16

Faster R-CNN

by Microsoft Research on NIPS 2015 and PAMI 2017

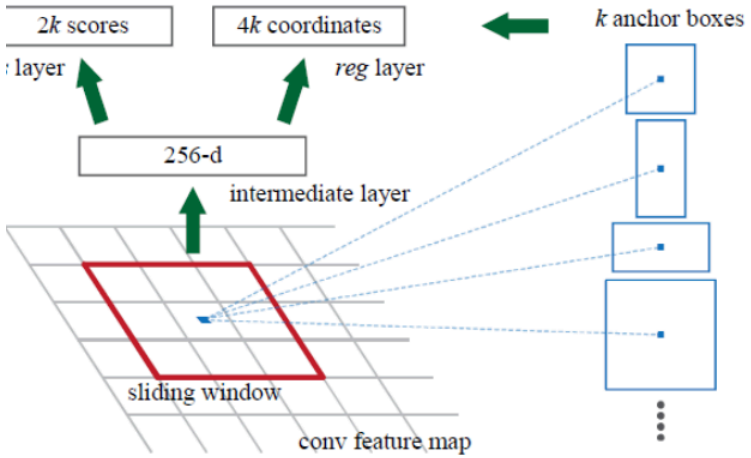
- **Region Proposal Extraction**: Region Proposal Network (RPN)
- **Feature Extraction**: CNN on whole image
- **ROI Projection**
- **ROI Pooling**:
- **Classification**: CNN
- **Bounding Box Regression**: CNN

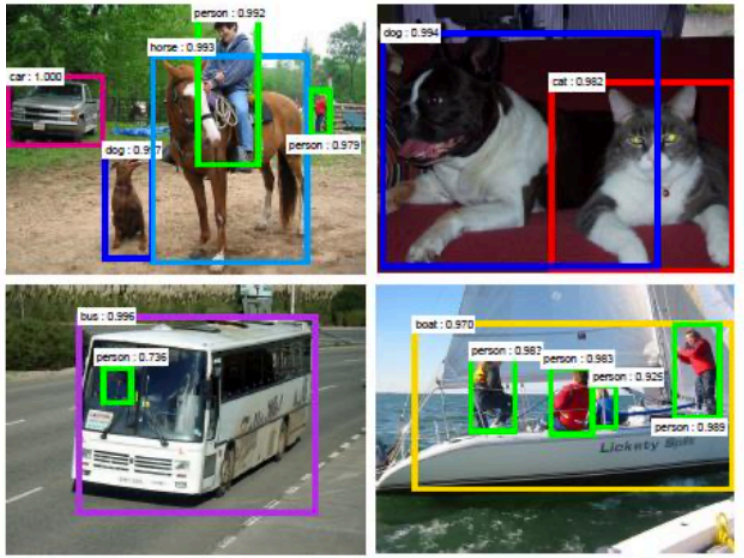


Shaoqing Ren, et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," NIPS 2015.

Region Proposal Network (RPN)

- Input: image/feature map
- Each W x H image/feature map has W x H sliding windows.
- Each sliding window has 9 anchors (initial region proposal).
  - 3 different scales (128^2, 256^2, 512^2)
  - 3 different aspect ratios (1:1, 1:2, 2:1)
- For each region proposal of a sliding window, predict if it has an object and predict the best position & size.
- Output: region proposal
  - 2 scores: the probability of object / not-object
  - 4 coordinates: x and y of center, width and height of the box





Shaoqing Ren, et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," NIPS 2015.

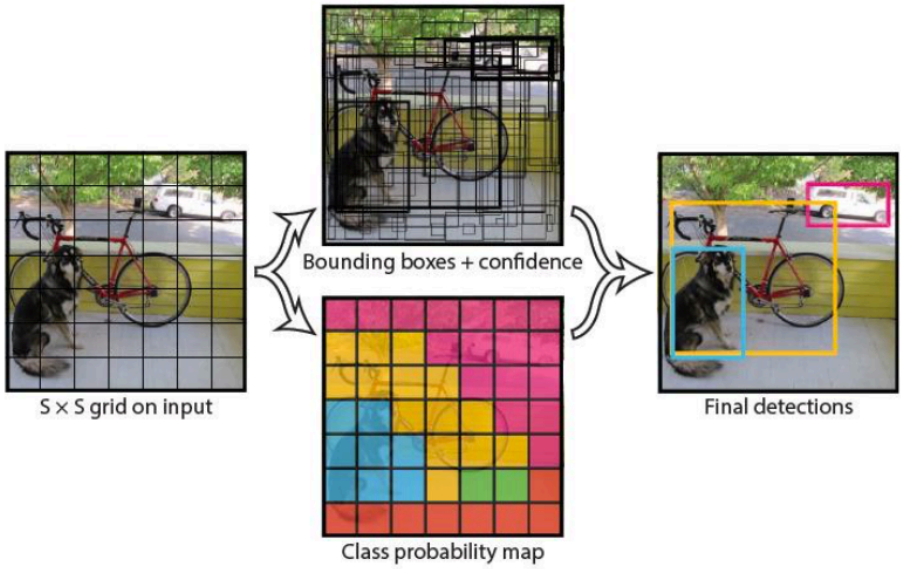
R-CNN vs Fast R-CNN vs Faster R-CNN

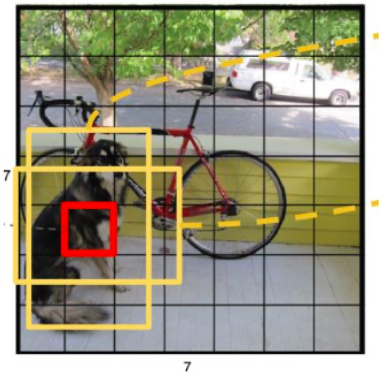
	R-CNN	**Fast R-CNN**	**Faster R-CNN**
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
(Speedup)	1x	25x	**250x**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

YOLO: You Only Look Once

by University of Washington & Facebook AI Research on CVPR 2016

- Divide an image into S x S grids.
- Each grid has B bounding boxes.
- For each bounding box of a grid, predict if it has an object and the best position & size.
  - 5 information: x, y of the center, w, h, confidence
- For each grid, predict which object it has (C class probabilities) (only one object).
- Each image has S x S x (B x 5 + C) dimensions.
  - in YOLO, S = 7, B = 2, C = 20
- Network is modified from GoogLeNet.
- Doesn't do region proposal extraction.





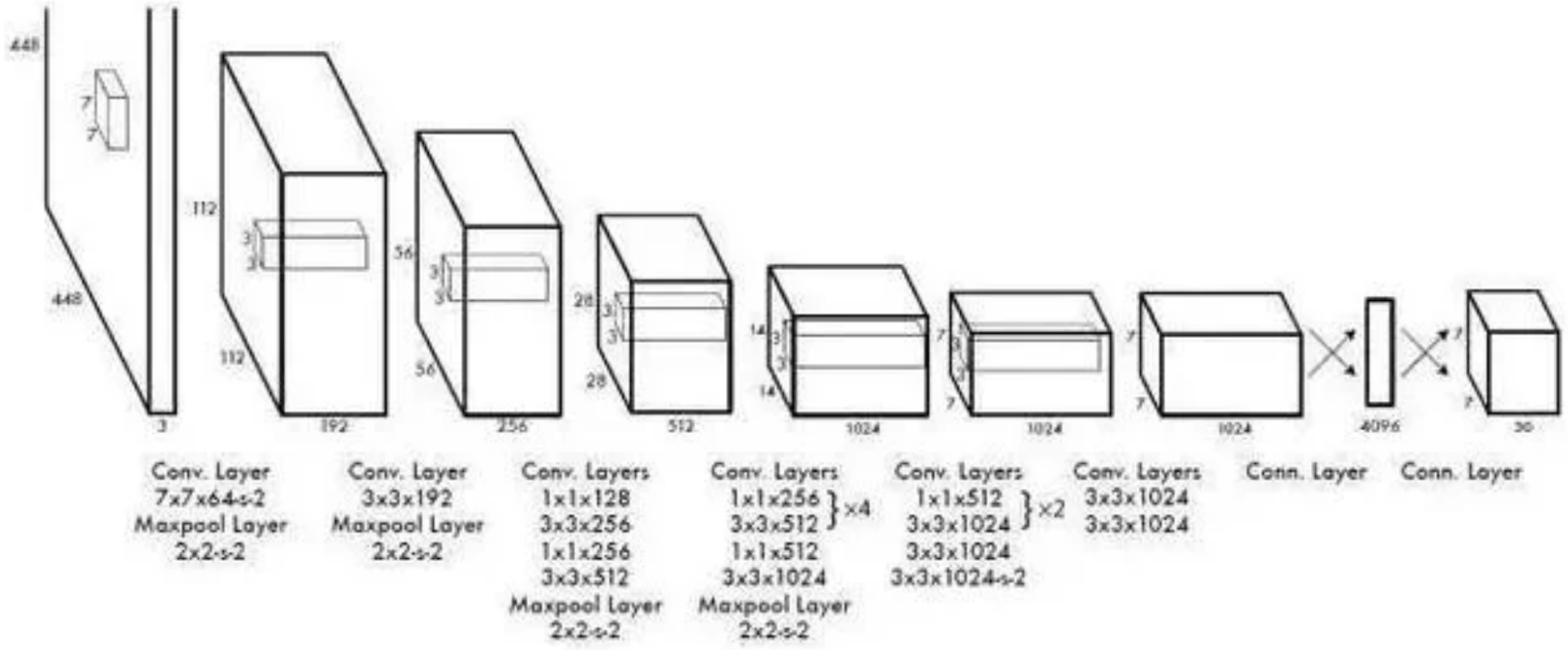
Two kinds of bounding box

Joseph Redmon, et al., "You Only Look Once: Unified, Real-Time Object Detection," CVPR 2016.

YOLO: You Only Look Once

by University of Washington & Facebook AI Research on CVPR 2016

Network architecture of YOLO:



Joseph Redmon, et al., "You Only Look Once: Unified, Real-Time Object Detection," CVPR 2016.

YOLO: You Only Look Once

by University of Washington & Facebook AI Research on CVPR 2016

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [30]	2007	16.0	100
30Hz DPM [30]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [37]	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[27]	2007+2012	73.2	7
Faster R-CNN ZF [27]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

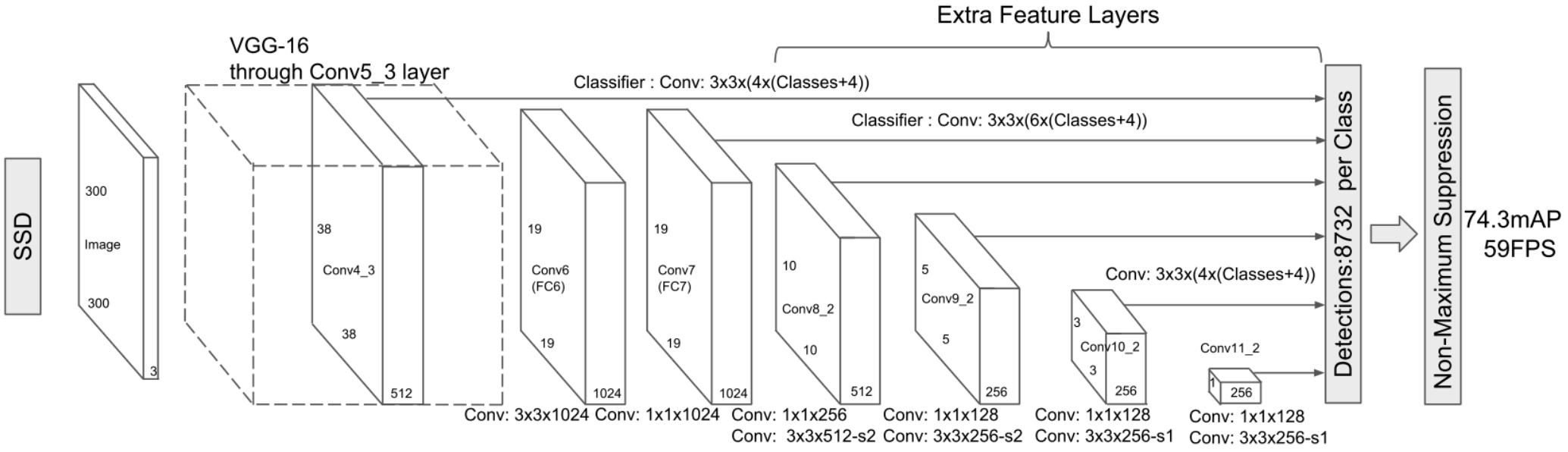
Fast YOLO: reduce the network layers of YOLO from 24 to 9.
R-CNN Minus R: R-CNN with static bounding boxes

VGG16: K. Simonyan, et al., "Very Deep Convolutional Networks for Large-Scale Image Recognition," NIPS 2015.

SSD: Single Shot MultiBox Detector

by UNC Chapel Hill & Google on ECCV 2016

- Add several convolutional layers after VGG-16
- Every W x H feature map of a convolutional layer has W x H default boxes.
- Each add layer has different size & aspect ratio default boxes.
- For each added convolutional layer, predicts the shape offset & probabilities of all classes.
  - shape offset: $\Delta(cx, cy, w, h)$



Wei Liu, et al., "SSD: Single Shot MultiBox Detector," ECCV 2016.

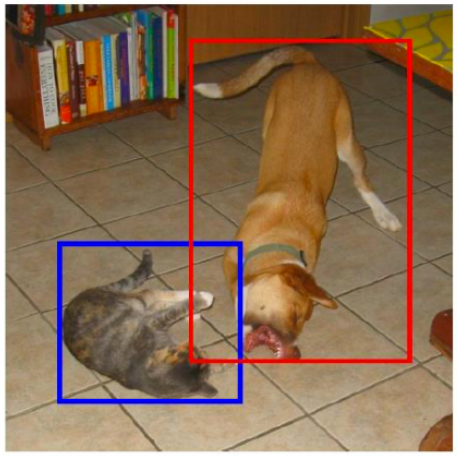
SSD: Single Shot MultiBox Detector

by UNC Chapel Hill & Google on ECCV 2016

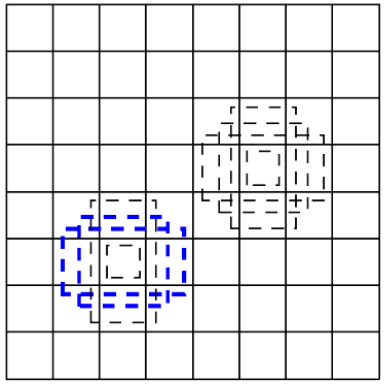
- Size & aspect ratio of default box
  - For m added convolutional layer
  - The scale of the kth added convolutional layer is

    $$s_k = s_{min} + \frac{s_{max} - s_{min}}{m - 1}(k - 1), k \in [1, m]$$

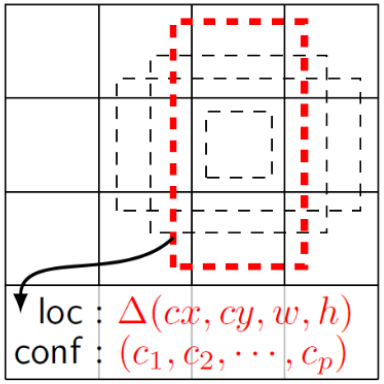
    where $s_{min} = 0.2, s_{max} = 0.9$
  - The aspect ratio is $a_r \in \left\{1, 2, 3, \frac{1}{2}, \frac{1}{3}\right\}$
  - $w_k^a = s_k\sqrt{a_r}; h_k^a = s_k/\sqrt{a_r}$
  - For aspect ratio 1, add a default box with scale $s_k' = \sqrt{s_k s_{k+1}}$



(a) Image with GT boxes



(b) 8×8 feature map



(c) 4×4 feature map

Wei Liu, et al., "SSD: Single Shot MultiBox Detector," ECCV 2016.

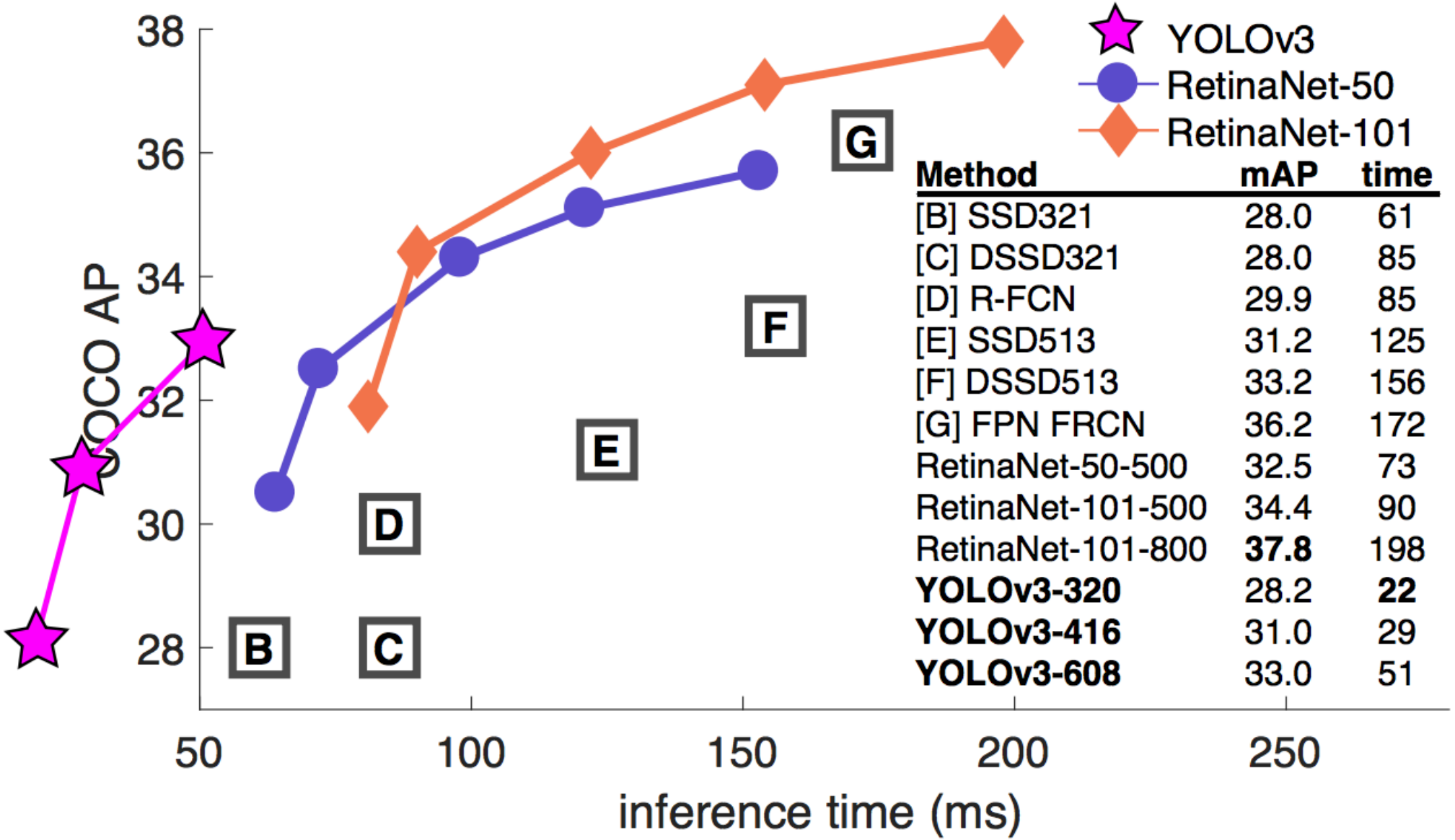
SSD: Single Shot MultiBox Detector

by UNC Chapel Hill & Google on ECCV 2016

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

Wei Liu, et al., “SSD: Single Shot MultiBox Detector,” ECCV 2016.

Performance for YOLOv3



Method	mAP	time
[B] SSD321	28.0	61
[C] DSSD321	28.0	85
[D] R-FCN	29.9	85
[E] SSD513	31.2	125
[F] DSSD513	33.2	156
[G] FPN FRCN	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	**37.8**	198
YOLOv3-320	28.2	**22**
YOLOv3-416	31.0	29
YOLOv3-608	33.0	51

Performance for YOLOv3

Object Detection on COCO test-dev

https://paperswithcode.com/sota/object-detection-on-coco

1	**Cascade Mask R-CNN** (Triple-ResNeXt152, multi-scale)	53.3	71.9	58.5	35.5	55.8	66.7	×	CBNet: A Novel Composite Backbone Network Architecture for Object Detection	2019
36	**Faster R-CNN** (HRNetV2p-W48)	42.4	63.6	46.4	24.9	44.6	53.0	×	Deep High-Resolution Representation Learning for Visual Recognition	2019
46	**RetinaNet** (ResNeXt-101-FPN)	40.8	61.1	44.1	24.1	44.2	51.2	×	Focal Loss for Dense Object Detection	2017
79	**YOLOv3 + Darknet-53**	33.0						×	YOLOv3: An Incremental Improvement	2018

Object Detection with Deep Learning