Computer Vision

DATA.ML.300, 5 study credits

Esa Rahtu
Unit of Computing Sciences, Tampere University

Object category detection

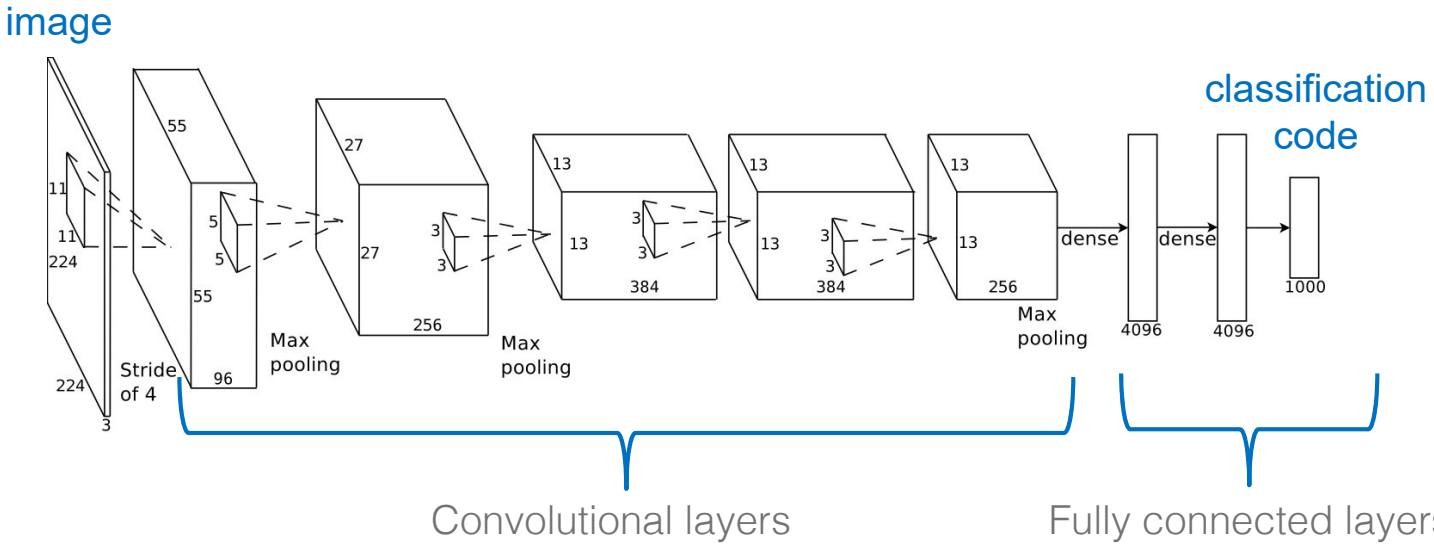
Outline for this week

- Lecture 1: Principles of sliding window detectors
  - Training a sliding window detector
  - Speeding up inferences
- **Lecture 2: Deep networks for object category detection**
  - Two-stage and one-stage networks
  - State-of-the-art

Background

Reminder: Classification CNNs

AlexNet (Krizhevsky et al. 2012)

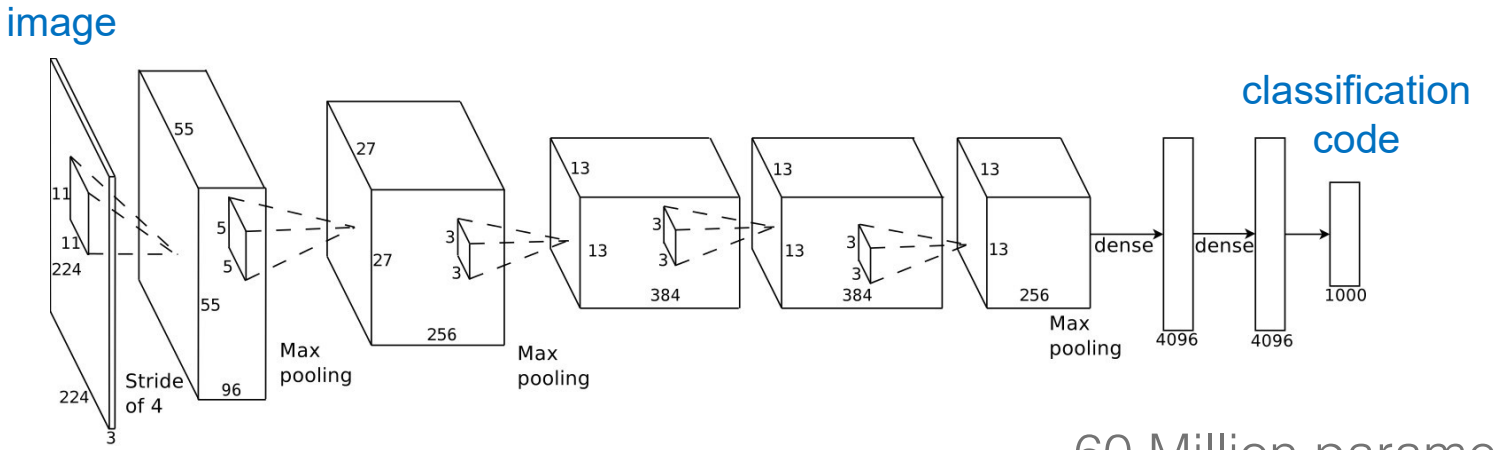


60 Million parameters

ImageNet classification challenge

- 1000 categories
- 1000 images from each category for training (approx. 1M images)
- 100k images for testing

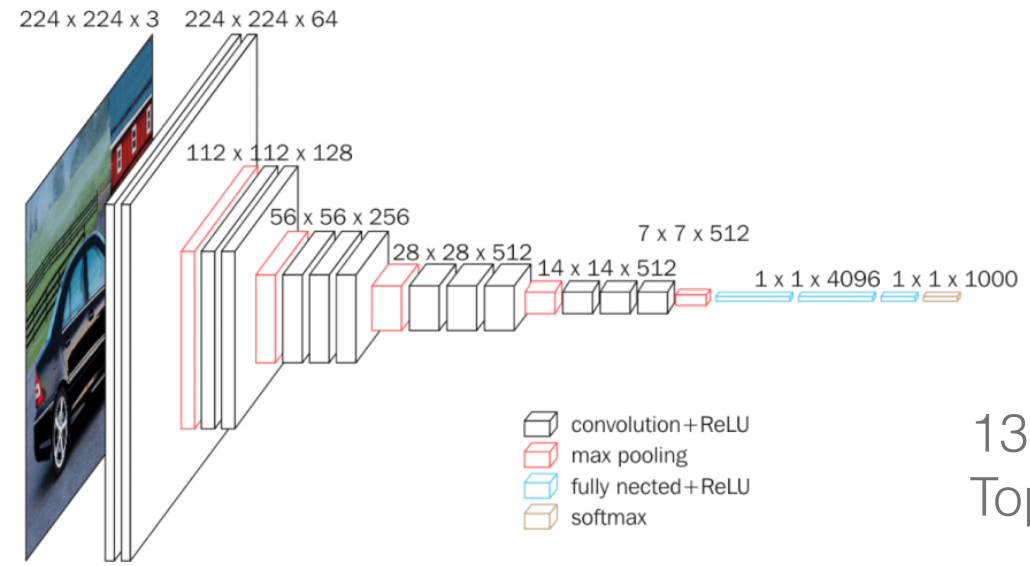
AlexNet (Krizhevsky et al. 2012)



60 Million parameters
Top-5 error 16%

ImageNet classification with deep convolutional neural networks, Krizhevsky et al. NIPS 2012

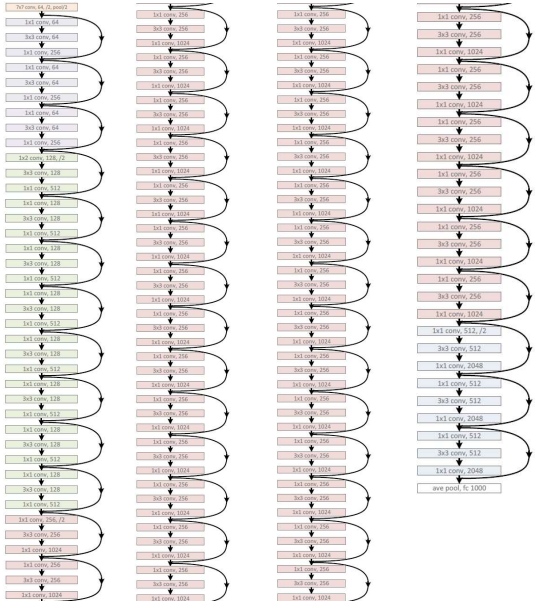
VGG-16 (Simonyan & Zisserman 2014)



138 Million parameters
Top-5 error 7%

Very deep convolutional networks for large-scale image recognition, Simonyan et al. arXiv 2014

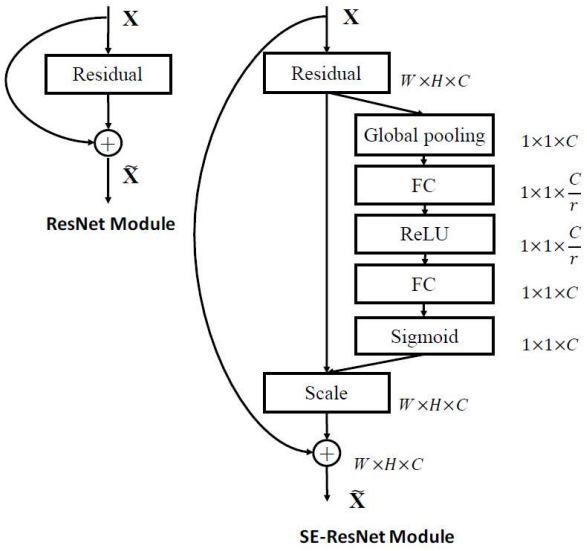
ResNet (He et al. 2015)

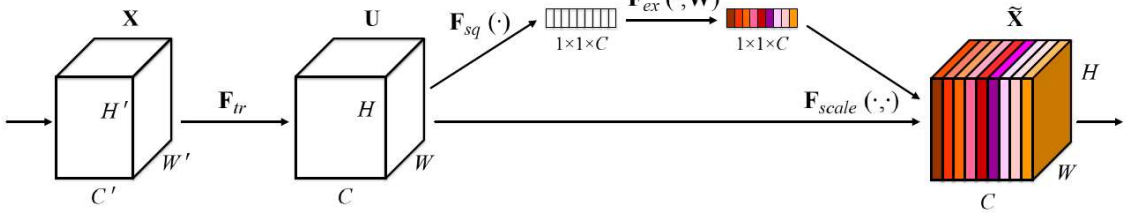


152 layers (60 Million parameters)
Top-5 error 4%

Deep residual learning for image recognition, He et al. CVPR 2016

Squeeze & Excitation (Hu et al. 2017)

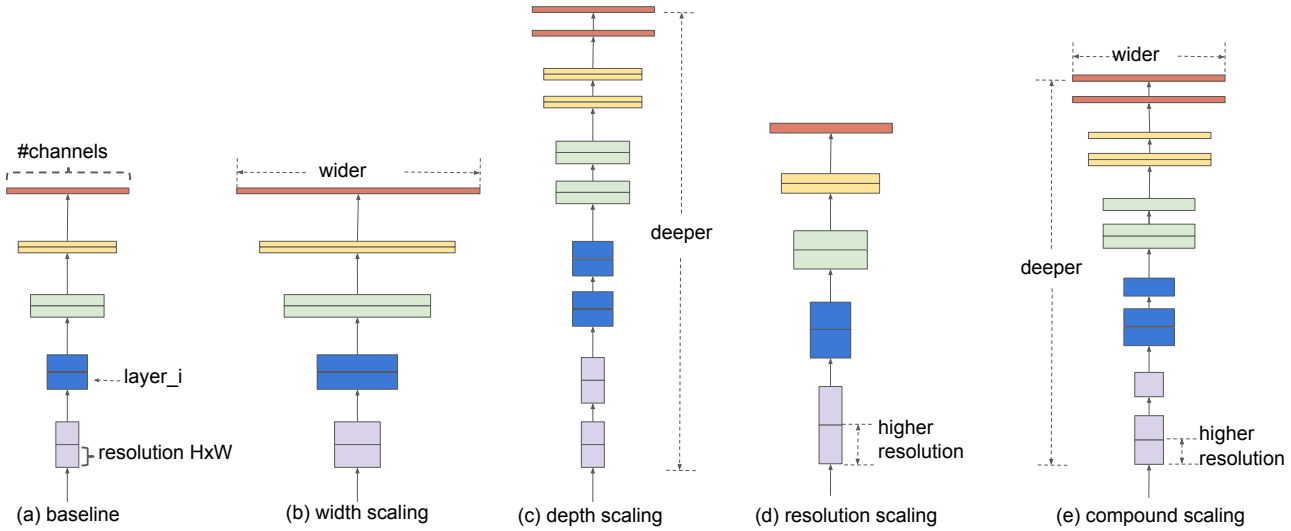




S&E ResNext 152
Top-5 error 2.3%

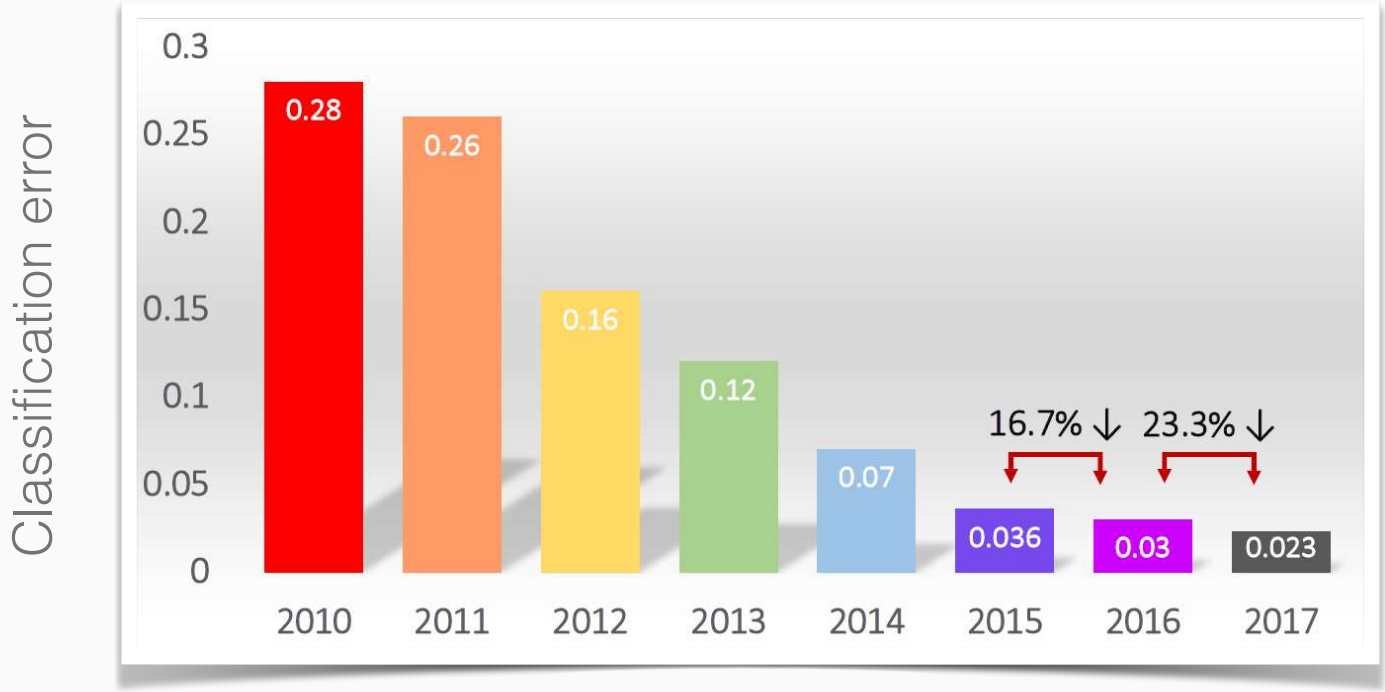
Squeeze-and-excitation networks, Hu et al. arXiv 2017

EfficientNet (Tan and Le 2019)



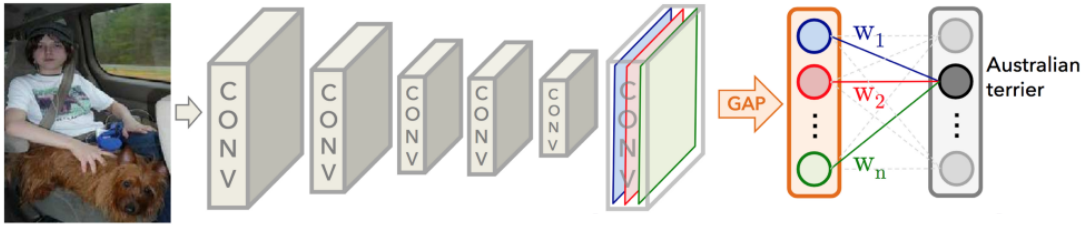
EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, Tan and Le arXiv 2019

ImageNet classification results (CLS)

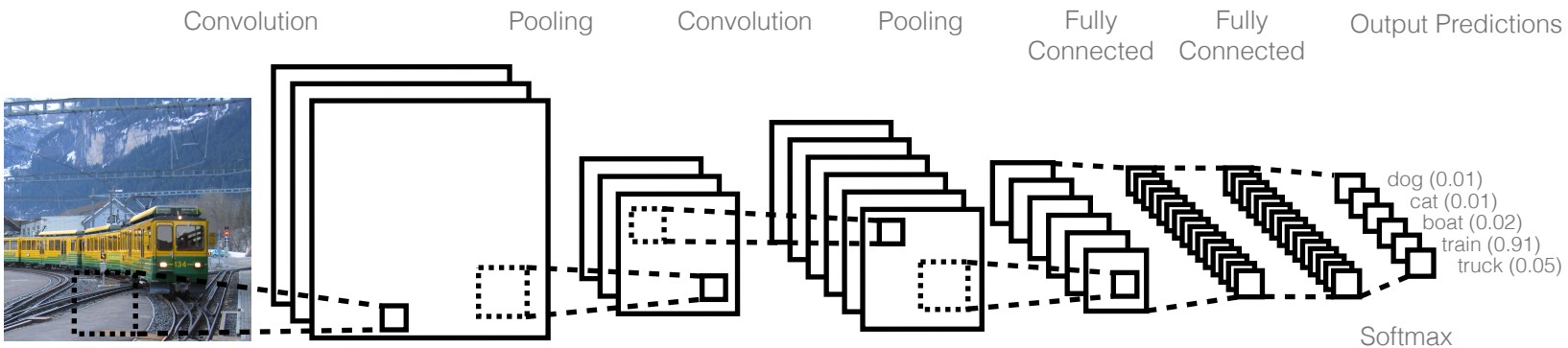


CNNs for detection - intuition I

- Modern classification architectures, such as ResNet or Inception, use convolutional layers throughout
  - No fully connected layers
  - Less parameters
  - Feature vector by spatial pooling

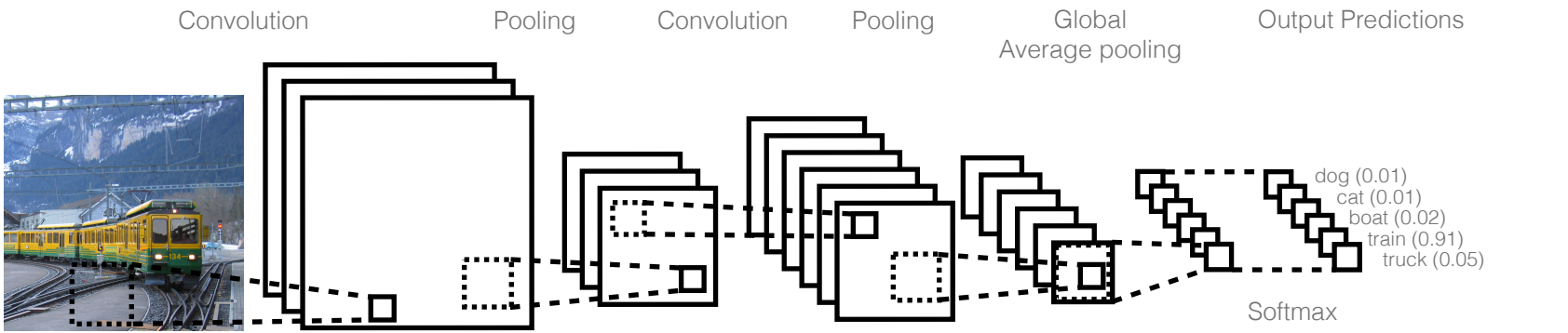


CNNs for detection - intuition I



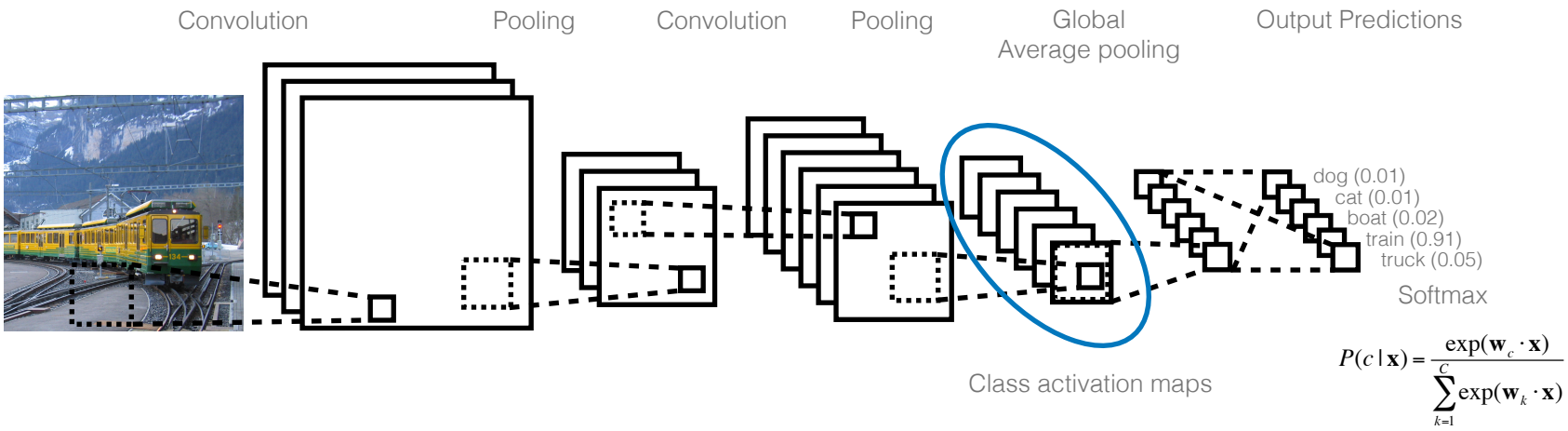
$$P(c \mid \mathbf{x}) = \frac{\exp(\mathbf{w}_c \cdot \mathbf{x})}{\sum_{k=1}^{C} \exp(\mathbf{w}_k \cdot \mathbf{x})}$$

CNNs for detection - intuition I

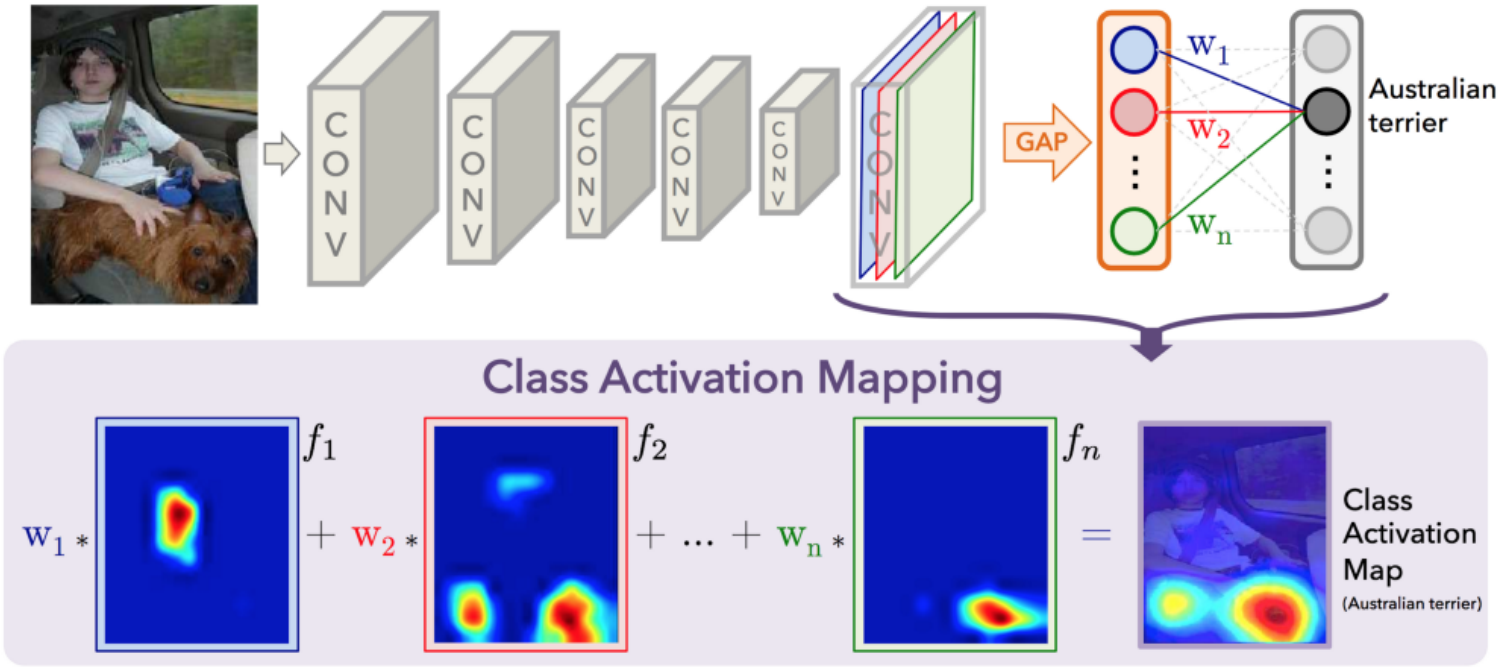


$$P(c \mid \mathbf{x}) = \frac{\exp(\mathbf{w}_c \cdot \mathbf{x})}{\sum_{k=1}^{C} \exp(\mathbf{w}_k \cdot \mathbf{x})}$$

CNNs for detection - intuition I

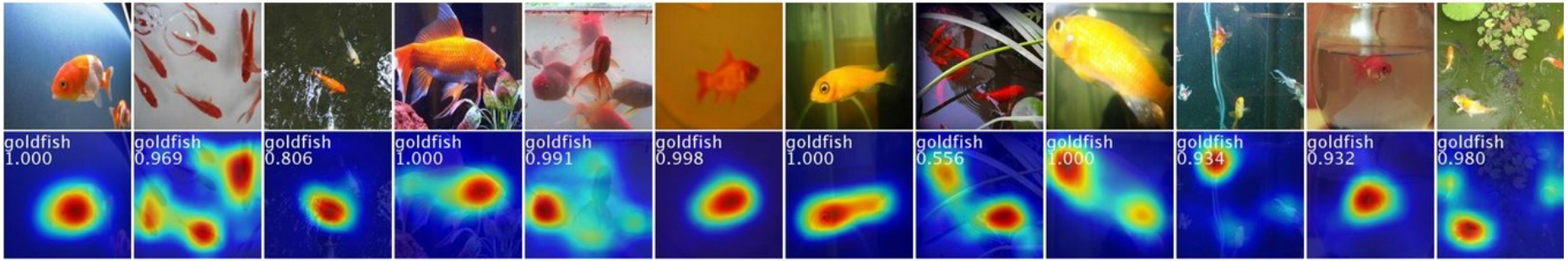


CNNs for detection - intuition II



Is object localisation for free?-weakly-supervised learning with convolutional neural networks, Oquab et al. CVPR 2015
Learning deep features for discriminative localisation, Zhou et al. CVPR 2016

CNNs for detection - intuition II



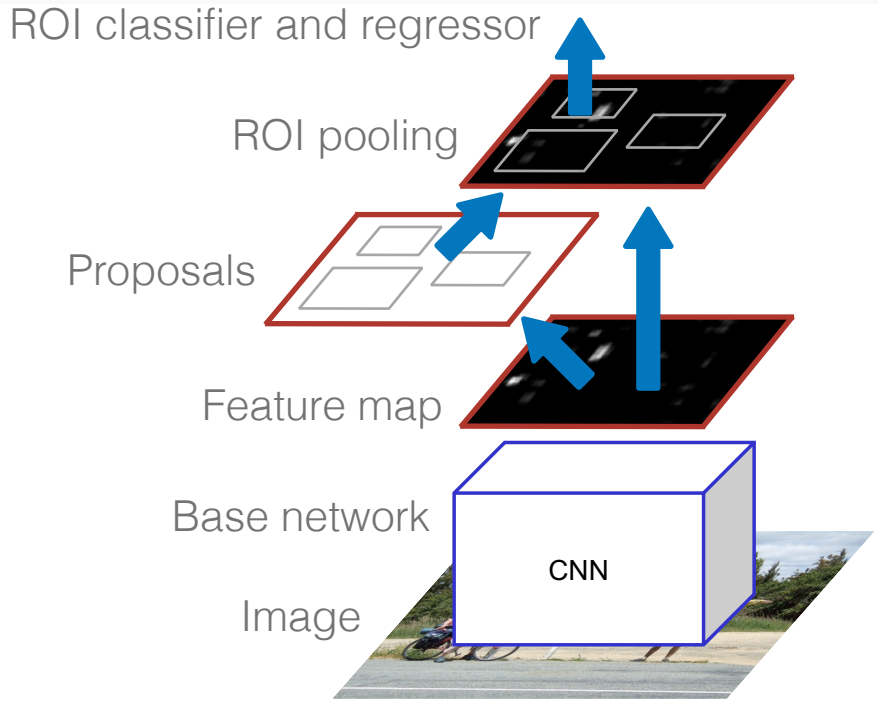
Object detection networks

Classical object detectors

- Two stage procedure:
  1. Propose class agnostic regions in the image (sliding window or proposals)
  2. Classify regions into object classes or background
- Can this be captured in a deep network?

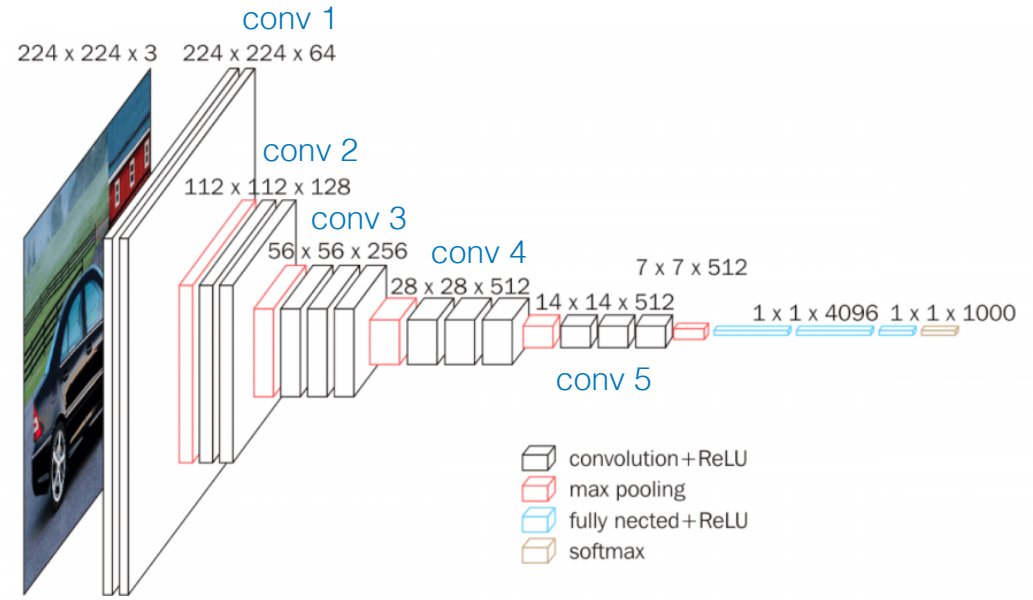
Faster R-CNN

- Two stage system:
  - Region proposal network (RPN)
  - Classification/regression network
- Base network VGG16



Faster R-CNN: Towards real-time object detection with region proposals, Ren et al. NIPS 2015

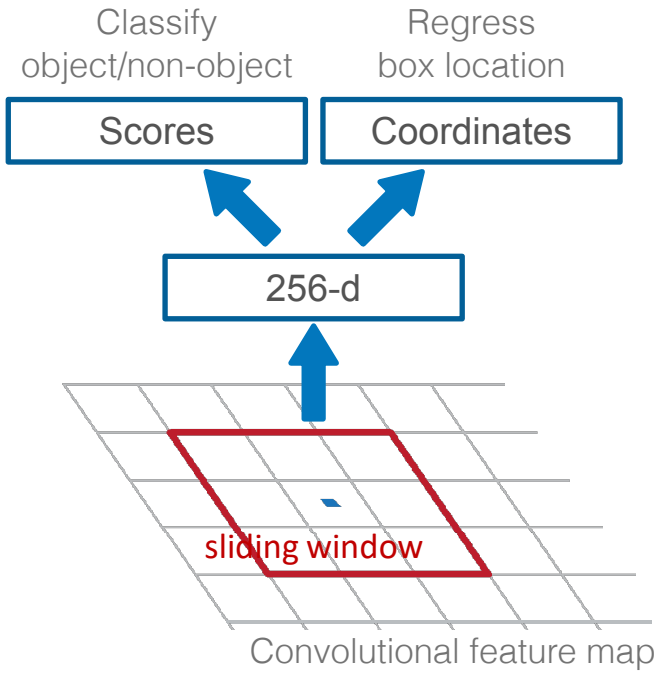
Reminder VGG-16



Very deep convolutional networks for large-scale image recognition, Simonyan et al. arXiv 2014

Region proposal network (RPN)

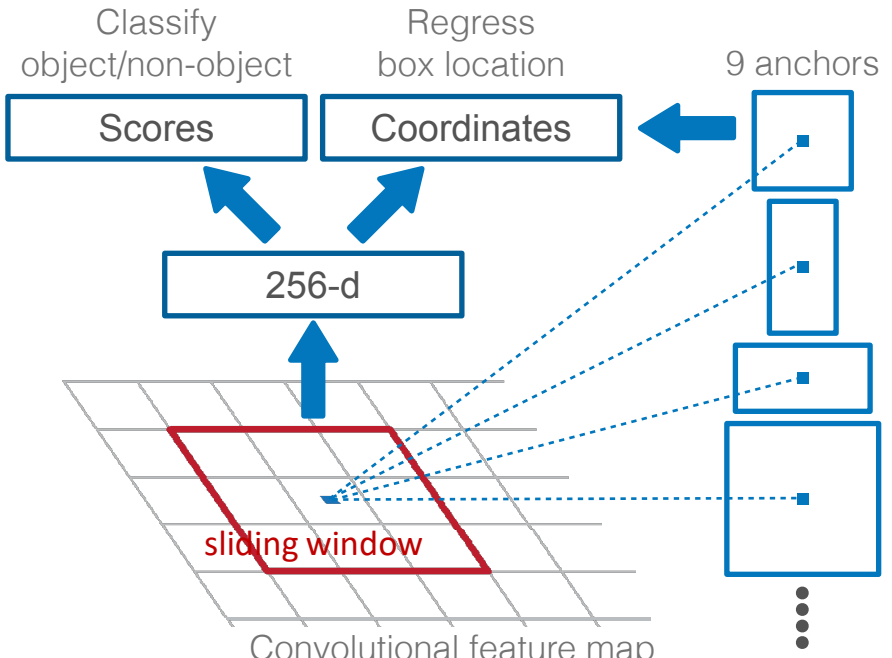
- Slide a small window on feature map
- Window position provides localisation **with reference to the image**
- Box regression provides finer localisation **with reference to window**



Faster R-CNN: Towards real-time object detection with region proposals, Ren et al. NIPS 2015

“Anchors”: predefined candidate regions

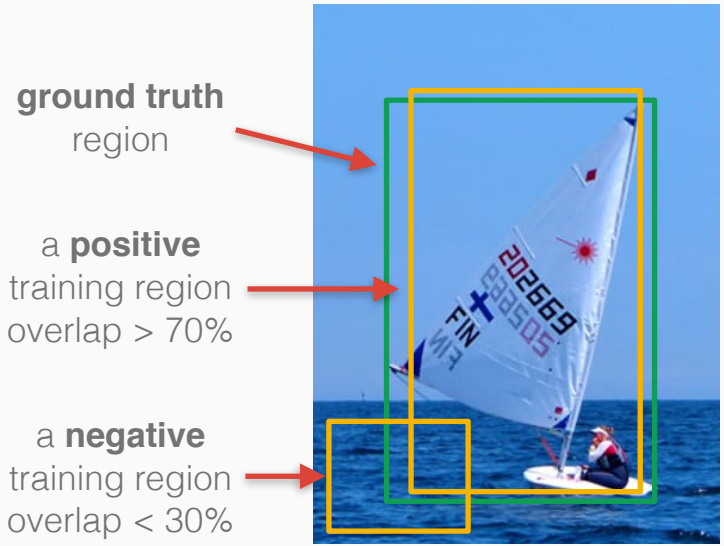
- Multi-scale/size anchors are used at each position: 3 scales x 3 aspect ratios yields 9 anchors
- Each anchor has its own prediction function
- **Single-scale** features, multi-scale predictions



Faster R-CNN: Towards real-time object detection with region proposals, Ren et al. NIPS 2015

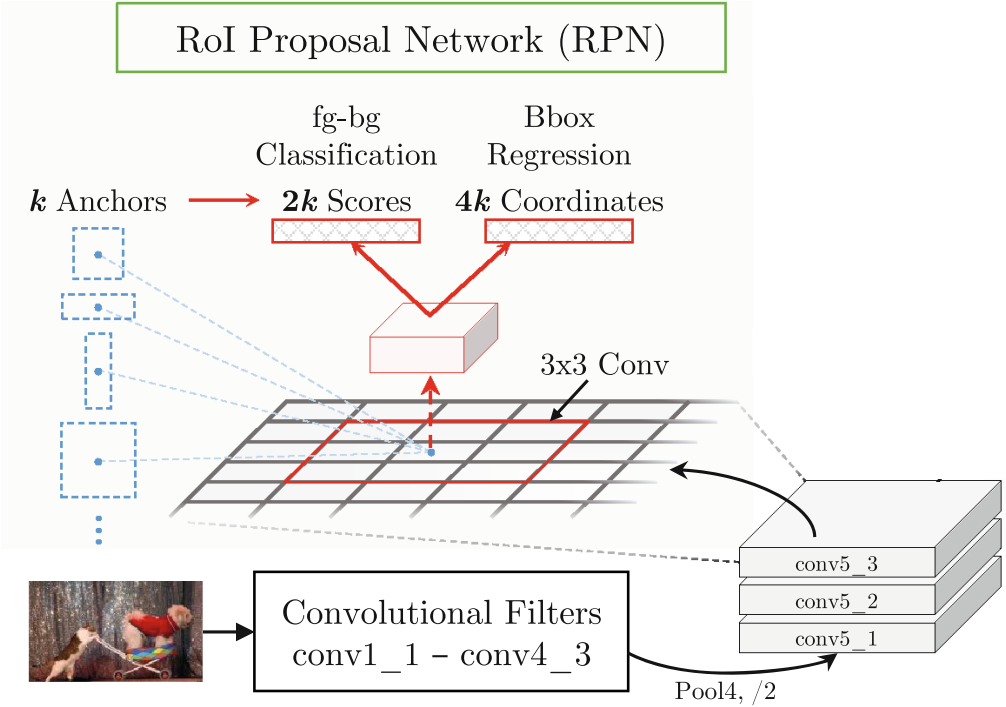
Training data: positive and negative boxes

- Label training boxes based on overlap with ground truth box
- Pre-train VGG16 CNN on ImageNet classification task



Faster R-CNN: Towards real-time object detection with region proposals, Ren et al. NIPS 2015

Faster R-CNN



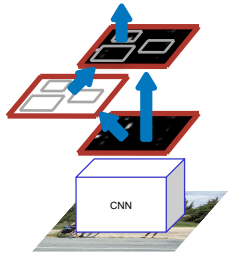


Figure from: Contextual priming and feedback for Faster R-CNN, Srivastava et al., ECCV 2016

Faster R-CNN

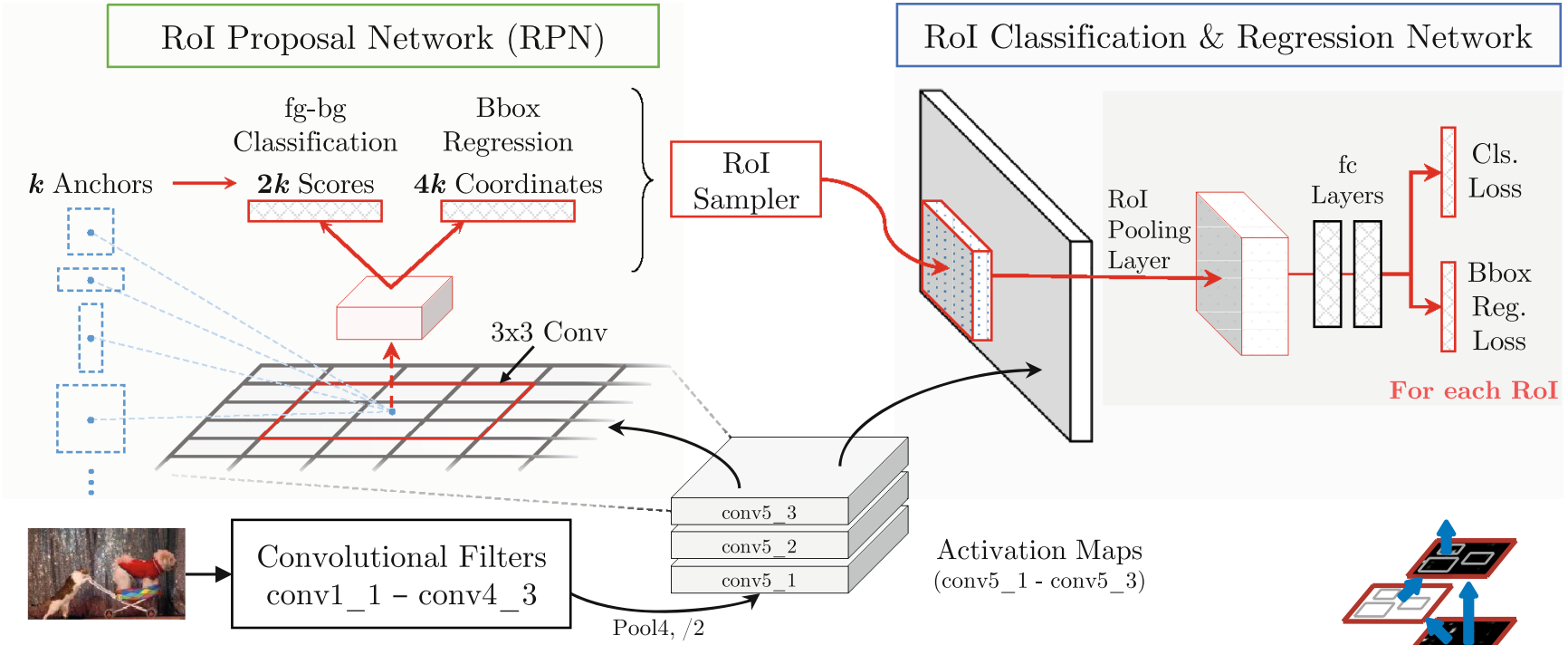
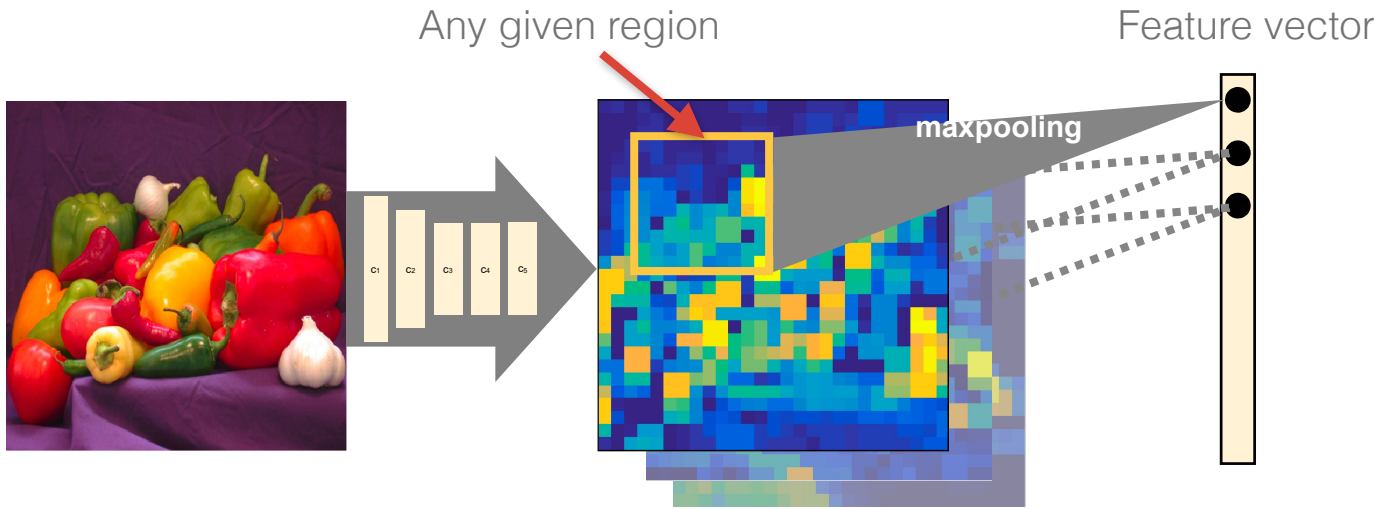


Figure from: Contextual priming and feedback for Faster R-CNN, Srivastava et al., ECCV 2016

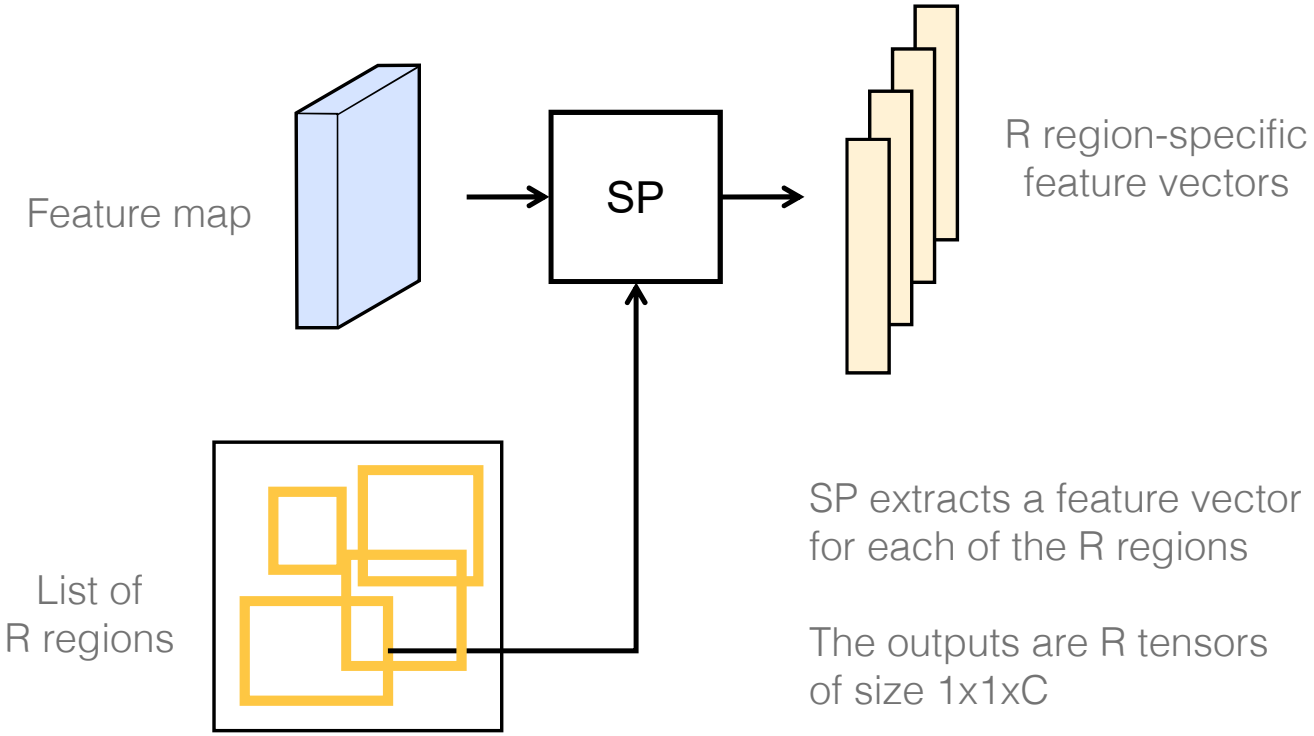
The Spatial Poolin (SP) layer

- Performs max-pooling for the feature responses in a given region
- Can be used to extract many region-specific feature vectors using same convolutional feature output



Spatial pyramid pooling (SPP) in deep convolutional networks for visual recognition, He et al. ECCV 2014

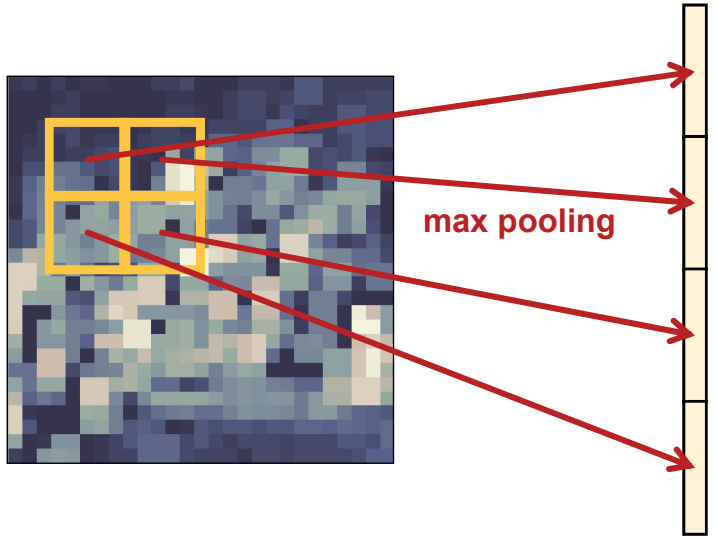
The Spatial Poolin (SP) layer as a building block



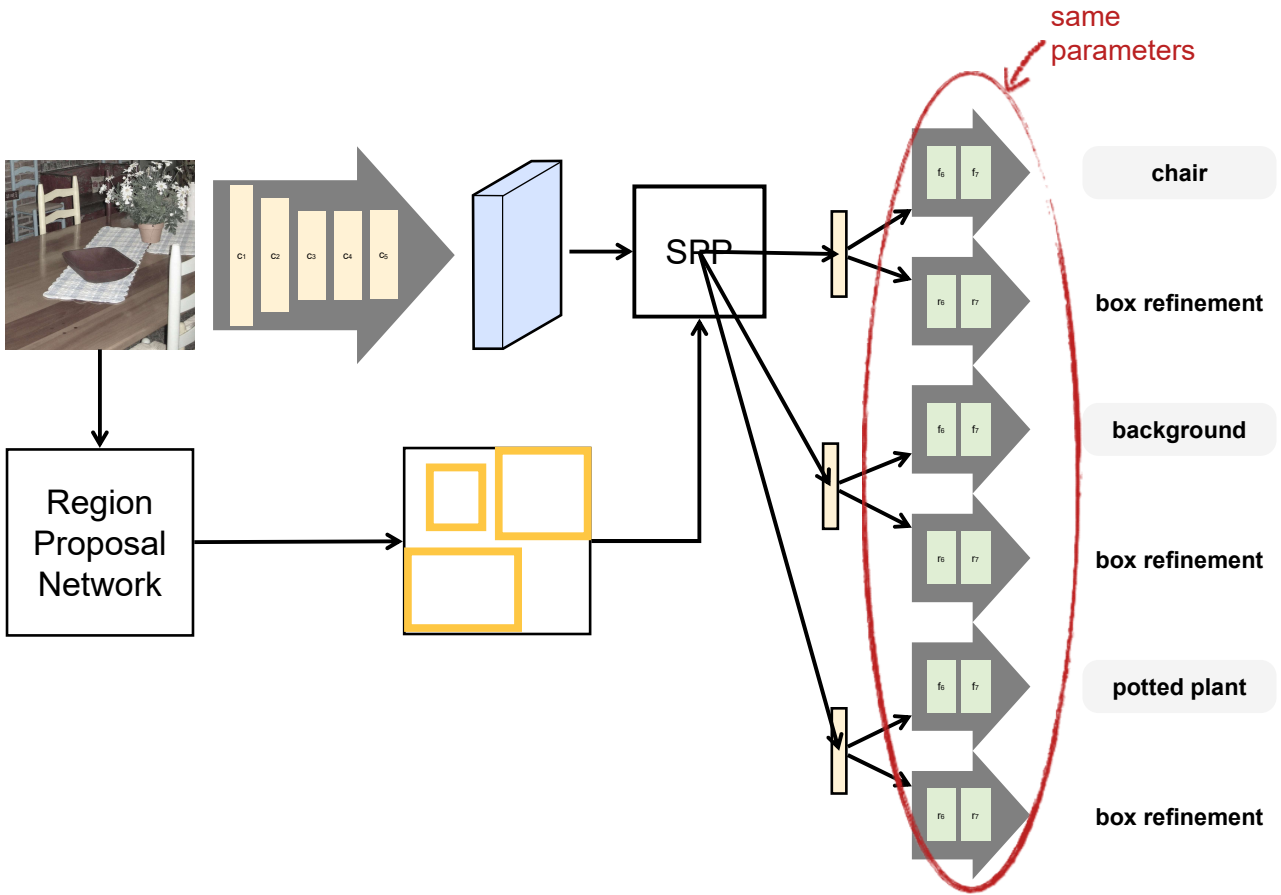
Spatial pyramid pooling (SPP) in deep convolutional networks for visual recognition, He et al. ECCV 2014

The Spatial Pyramid Pooling (SPP) layer

- Similar to SP, but pools features in tiles of a grid-like subdivision of the region (SP with multiple subdivisions)
- Feature vector **captures the spatial layout** of the original region
- Converts the region to a **fixed size vector**

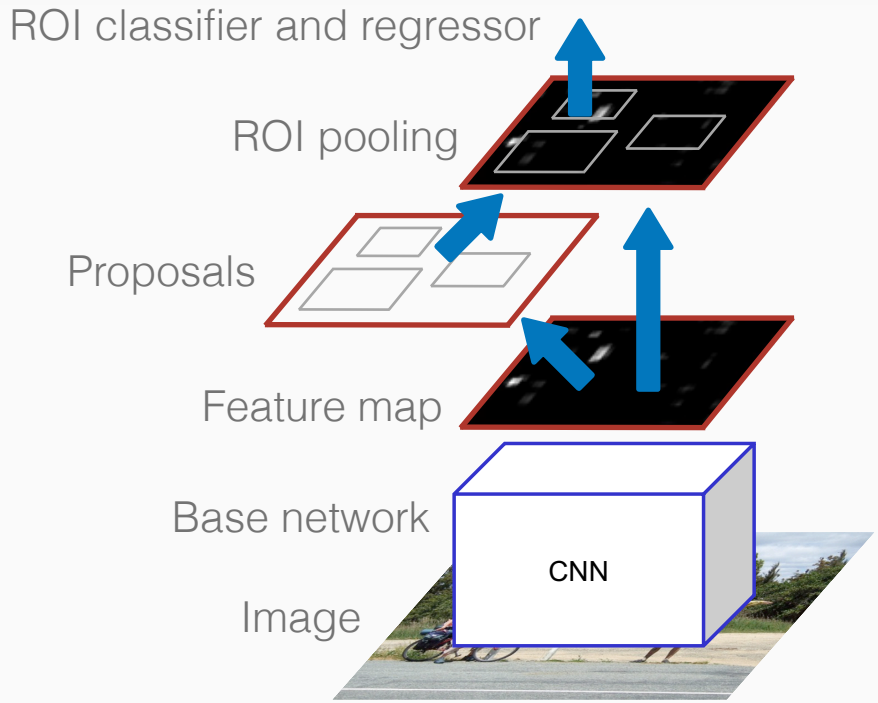


The Spatial Pyramid Pooling (SPP) layer



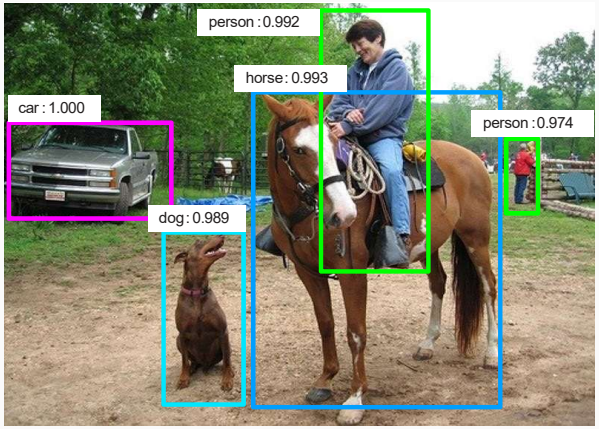
Faster R-CNN

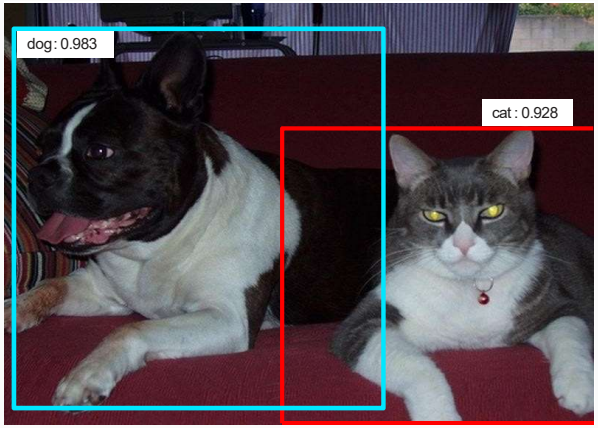
- Same CNN conv5 features used for:
  - The region proposal network
  - Classifying/regressing the regions
- Thus CNN runs only once on image
- Trained end-to-end
- Base network VGG 16

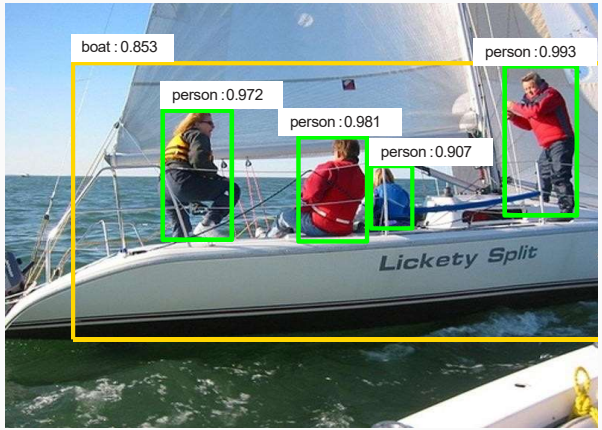


Faster R-CNN: Towards real-time object detection with region proposals, Ren et al. NIPS 2015

Example detections

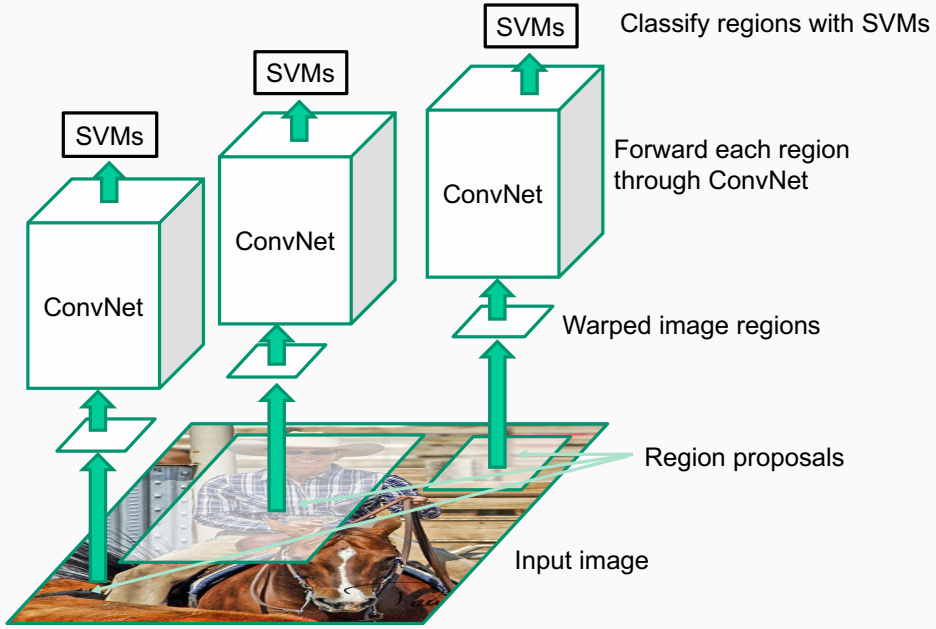






Why “Faster R-CNN”?

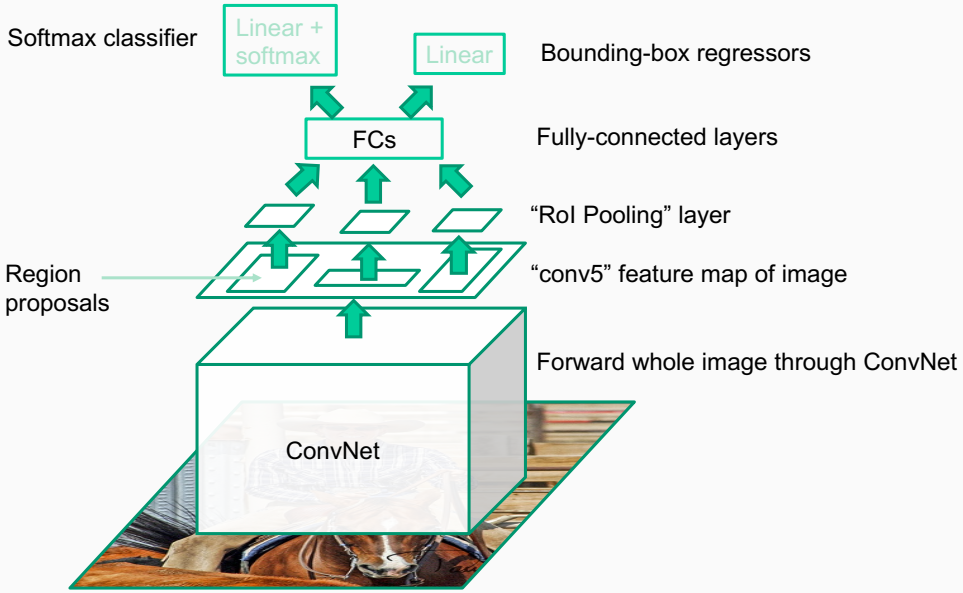
- First: R-CNN
- Inference time approx.
  50s per image



Rich feature hierarchies for accurate object detection and semantic segmentation, Girshick et al., CVPR 2014

Why “Faster R-CNN”?

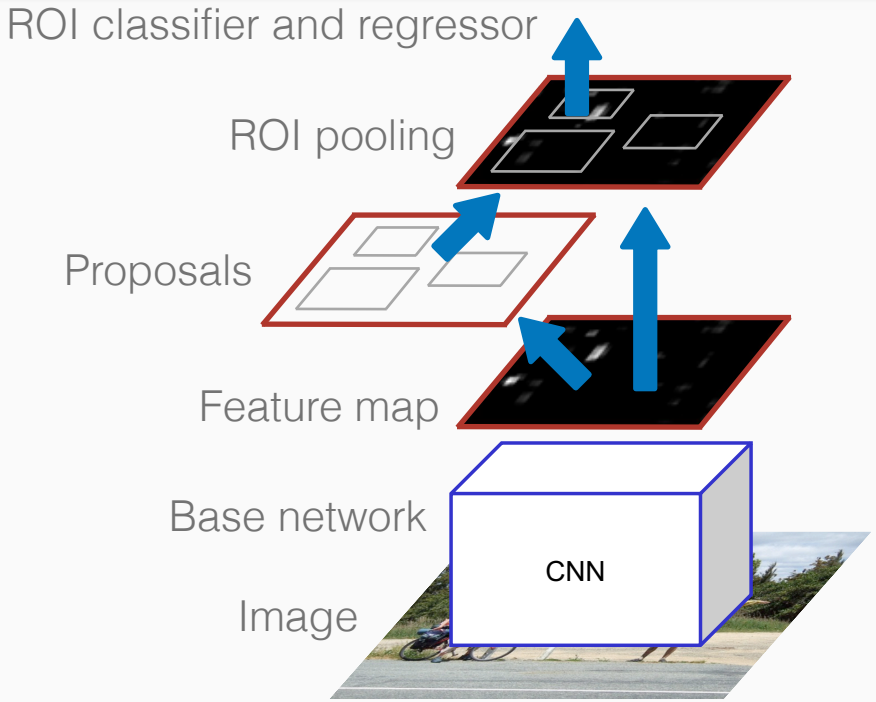
- Second: Fast R-CNN
- Inference time approx. 2s per image



Fast R-CNN, Girshick., ICCV 2015

Why “Faster R-CNN”?

- Third: Faster R-CNN
- Inference time approx. 198ms per image



Faster R-CNN: Towards real-time object detection with region proposals, Ren et al. NIPS 2015

Evaluating object detectors

Evaluating object detectors

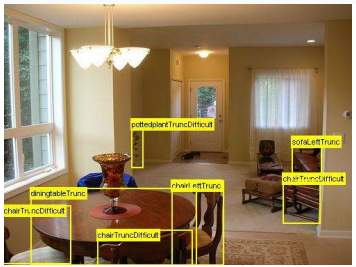
- Classical benchmark:

The PASCAL Visual Object Classes (VOC) dataset and Challenge 2007-2012

Mark Everingham, Luc Van Gool, Chris Williams, John Winn, Andrew Zisserman

PASCAL VOC dataset content

- Objects from 20 classes:
  aeroplane, bicycle, boat, bottle, bus, car, cat, chair, cow, dining table, dog, horse, motorbike, person, potted plant, sheep, train, TV
- Real world images downloaded from Flickr (not filtered for “quality”)
- Complex scenes, multiple scales, lighting, occlusions,….

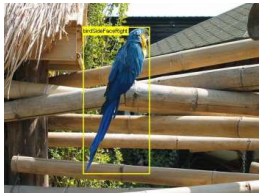


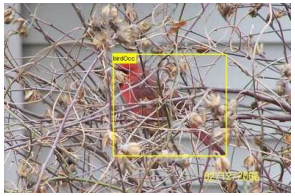
Examples

Aeroplane

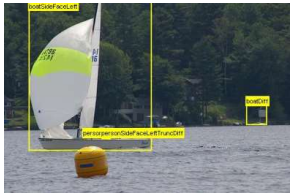
Bicycle

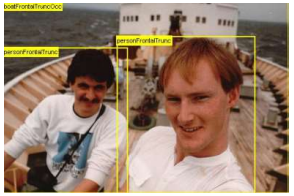
Bird





Boat





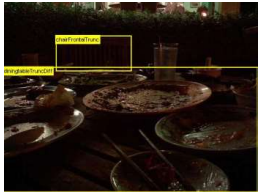
Bottle

Bus

Car

Cat

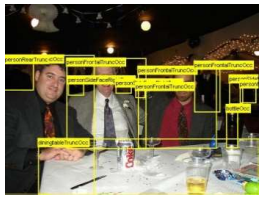
Chair



Cow

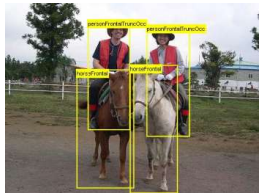
Examples

Dining Table

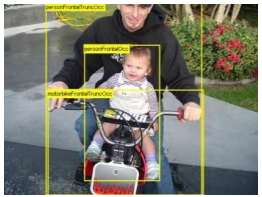


Dog

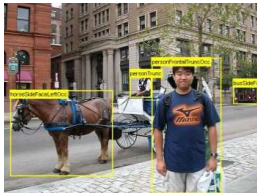
Horse

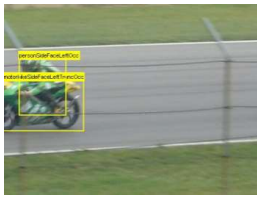


Motorbike

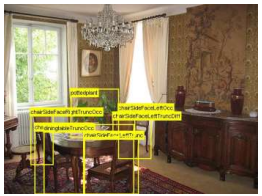


Person

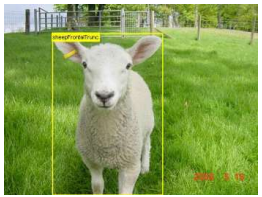


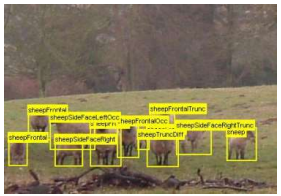


Potted Plant

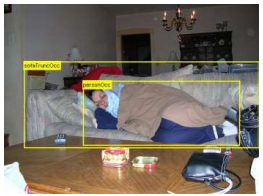


Sheep



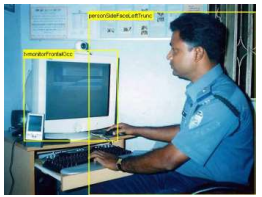


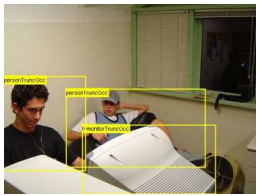
Sofa



Train

TV/Monitor



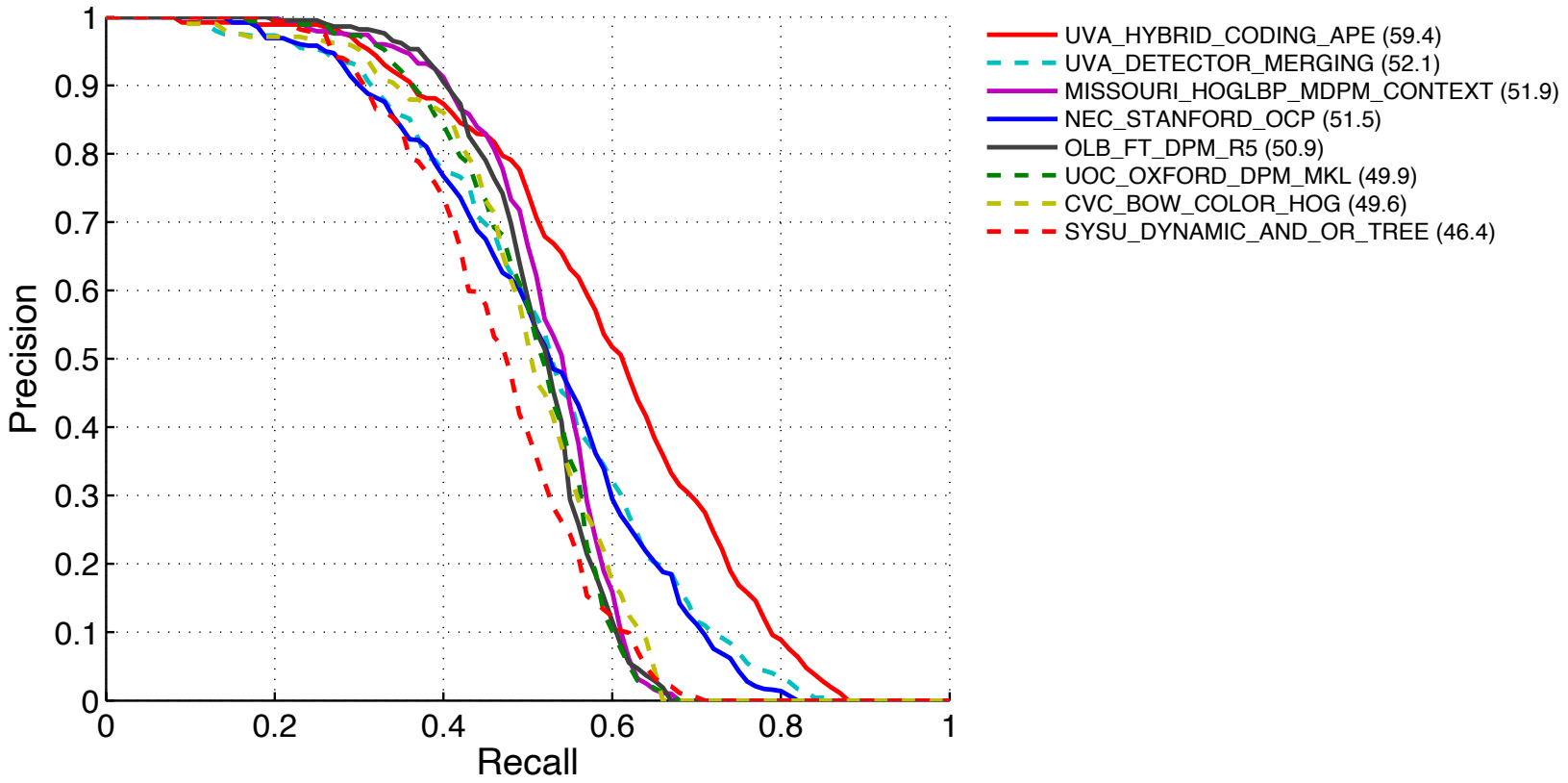


PASCAL VOC statistics

- Minimum 600 training objects per category
- Approx. 2000 cars, 1500 dogs, 8500 people
- Approximately similar distribution across training and test sets

	Training	Testing
Images	11,540	10,994
Objects	27,450	27,078

Precision-recall for Motorbikes VOC 2012



PASCAL VOC Leaderboard Nov 2014

Detection challenge comp3: **train on VOC 2012 data**

Average Precision (AP %)

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
HybridCodingApe [?]	**40.9**	61.8	52.0	24.6	**24.8**	20.2	57.1	44.5	53.6	17.4	33.0	**38.3**	**42.8**	48.8	**59.4**	35.7	22.8	40.3	**39.5**	51.1	**49.5**	23-Sep-2012
Data Decomposition and Distinctive Context [?]	**40.9**	55.0	**58.1**	22.5	18.8	33.9	**57.6**	**54.5**	42.6	20.2	**40.3**	29.3	37.1	**54.6**	58.3	**51.6**	14.7	**44.8**	32.1	51.7	41.0	13-Oct-2011
segDPM [?]	40.7	59.1	54.3	**28.2**	24.4	**34.5**	53.4	48.1	51.3	18.1	37.8	29.9	40.4	48.9	52.9	46.4	16.1	39.5	35.4	50.8	44.9	24-Feb-2014
Fisher with FLAIR [?]	40.6	61.7	52.0	27.9	24.0	18.9	56.5	45.3	53.4	15.5	34.6	36.3	42.3	48.4	57.9	36.6	**24.3**	40.6	38.0	49.8	49.0	17-Jun-2014
NYU-UCLA_Hierarchy [?]	40.6	56.3	55.9	23.4	20.3	27.2	56.6	48.1	**53.8**	**23.3**	32.9	33.4	39.2	53.0	56.9	43.6	14.3	37.9	39.4	**52.6**	43.7	13-Oct-2011
DPM-MKL [?]	39.1	59.6	54.5	21.9	21.6	32.1	52.5	49.3	40.8	19.1	35.2	28.9	37.2	50.9	49.9	46.1	15.6	39.3	35.6	48.9	42.8	23-Sep-2012
DPM-MK [?]	38.3	56.0	53.3	19.2	17.3	25.8	53.1	45.4	44.5	20.1	32.1	28.1	37.2	52.3	56.6	43.3	12.1	34.3	37.6	51.8	45.2	13-Oct-2011
NEC_STANFORD_OCP [?]	36.7	**65.1**	46.8	25.0	24.6	16.0	51.0	44.9	51.5	13.0	26.6	31.0	40.2	39.7	51.5	32.8	12.6	35.7	33.5	48.0	44.8	23-Sep-2012
Detector-Merging [?]	36.5	47.2	50.2	18.3	21.4	25.2	53.3	46.3	46.3	17.5	27.8	30.3	35.0	41.6	52.1	43.2	18.0	35.2	31.1	45.4	44.4	23-Sep-2012
MISSOURI_HOGLBP_MDPM_CONTEXT [?]	36.4	51.4	53.7	18.3	15.6	31.6	56.5	47.1	38.6	19.5	32.0	22.1	25.0	50.3	51.9	44.9	11.9	37.7	30.6	50.9	39.3	23-Sep-2012

PASCAL VOC Leaderboard Nov 2014

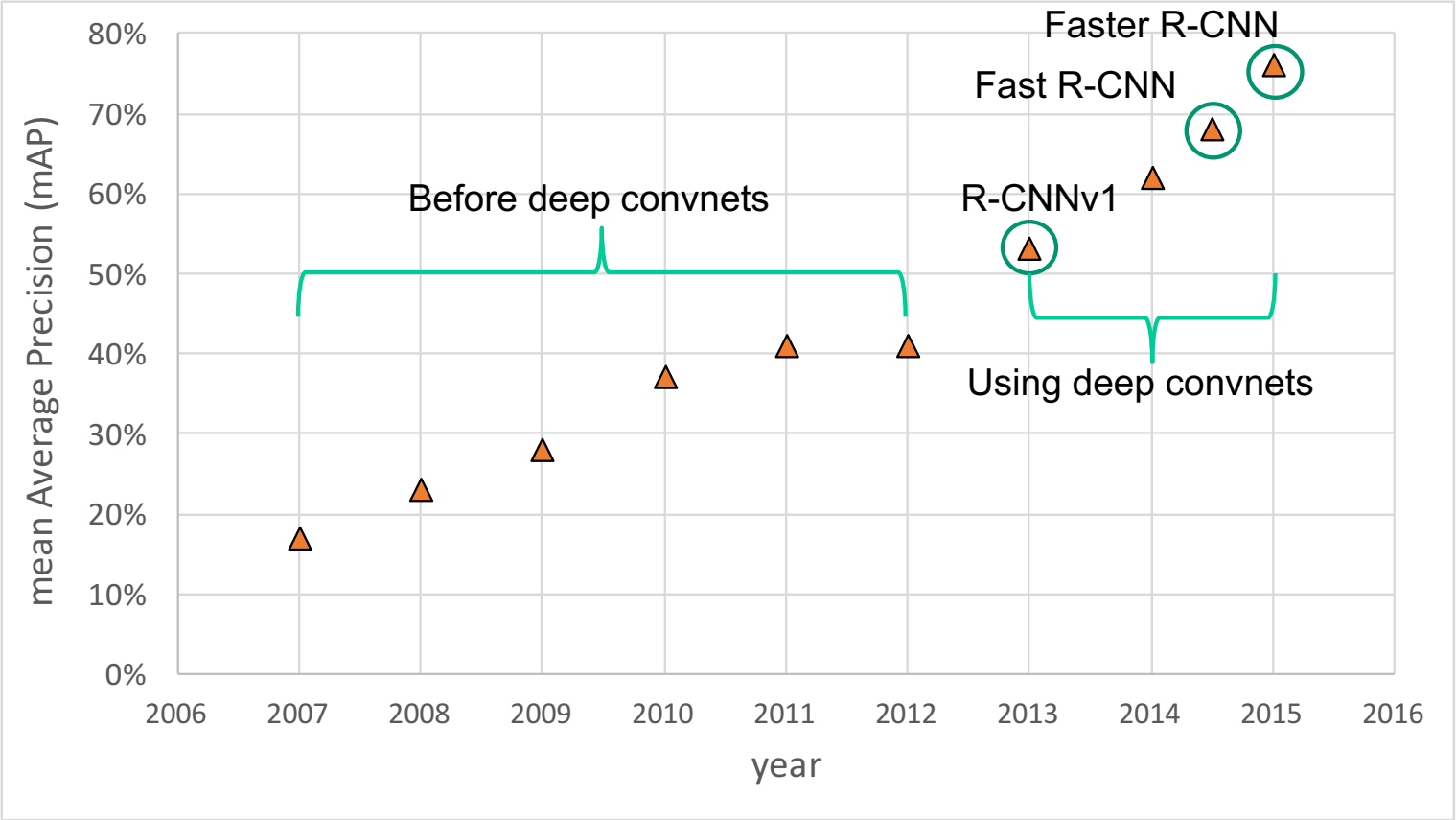
Detection challenge comp4: **train on own data**

Average Precision (AP %)

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
NUS_NIN_c2000 [?]	**63.8**	**80.2**	73.8	61.9	43.7	43.0	**70.3**	**67.6**	80.7	**41.9**	69.7	**51.7**	78.2	**75.2**	76.9	65.1	**38.6**	**68.3**	**58.0**	**68.7**	**63.3**	30-Oct-2014
BabyLearning [?]	63.2	78.0	**74.2**	61.3	**45.7**	42.7	68.2	66.8	80.2	40.6	**70.0**	49.8	79.0	74.5	**77.9**	64.0	35.3	67.9	55.7	**68.7**	62.6	12-Nov-2014
R-CNN (bbox reg) [?]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	**84.6**	38.5	67.2	46.7	**82.0**	74.8	76.0	**65.2**	35.6	65.4	54.2	67.4	60.3	26-Oct-2014
NUS_NIN [?]	62.4	77.9	73.1	**62.6**	39.5	**43.3**	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7	30-Oct-2014
R-CNN [?]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7	25-Oct-2014
Feature Edit [?]	56.3	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8	06-Sep-2014
R-CNN (bbox reg) [?]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1	13-Mar-2014
SDS [?]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7	21-Jul-2014
R-CNN [?]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6	30-Jan-2014

First deep network object detectors

Progress in object detection



PASCAL VOC Leaderboard Dec 2015

Detection challenge comp4: train on own data

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
Faster RCNN, ResNet (VOC+COCO) [?]	**83.8**	**92.1**	**88.4**	**84.8**	**75.9**	**71.4**	**86.3**	**87.8**	**94.2**	**66.8**	**89.4**	**69.2**	**93.9**	**91.9**	**90.9**	**89.6**	**67.9**	**88.2**	**76.8**	**90.3**	**80.0**	10-Dec-2015
ION [?]	76.4	87.5	84.7	76.8	63.8	58.3	82.6	79.0	90.9	57.8	82.0	64.7	88.9	86.5	84.7	82.3	51.4	78.2	69.2	85.2	73.5	23-Nov-2015
MNC baseline [?]	75.9	86.4	81.1	76.4	64.3	57.8	81.1	80.3	92.0	55.2	82.6	61.0	89.9	86.4	84.6	85.4	53.1	79.8	66.1	84.7	69.9	15-Dec-2015
Faster RCNN baseline (VOC+COCO) [?]	75.9	87.4	83.6	76.8	62.9	59.6	81.9	82.0	91.3	54.9	82.6	59.0	89.0	85.5	84.7	84.1	52.2	78.9	65.5	85.4	70.2	24-Nov-2015
LocNet [?]	74.8	86.3	83.0	76.1	60.8	54.6	79.9	79.0	90.6	54.3	81.6	62.0	89.0	85.7	85.5	82.8	49.7	76.6	67.5	83.2	67.4	06-Nov-2015
** HRCNN ** [?]	74.6	85.9	83.9	75.5	60.9	54.5	81.4	79.1	90.6	53.3	79.7	61.6	89.9	86.2	85.8	78.2	49.1	75.1	68.6	86.1	67.7	13-Nov-2015
MR_CNN_S_CNN_MORE_DATA [?]	73.9	85.5	82.9	76.6	57.8	62.7	79.4	77.2	86.6	55.0	79.1	62.2	87.0	83.4	84.7	78.9	45.3	73.4	65.8	80.3	74.0	06-Jun-2015
HyperNet_VGG [?]	71.4	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.9	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7	12-Oct-2015
HyperNet_SP [?]	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6	28-Oct-2015

Mean average precision 83.8

Since then: More data - MS COCO

What is COCO?

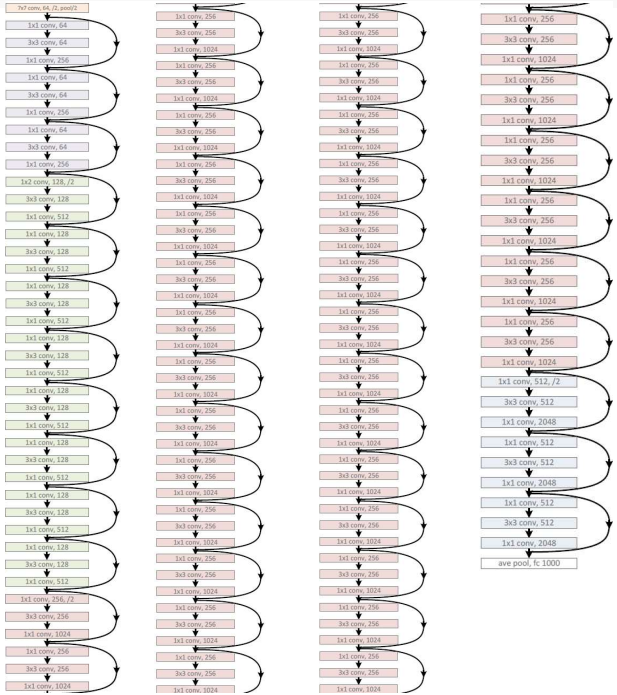
COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

- **Object segmentation**
- **Recognition in context**
- **Superpixel stuff segmentation**
- **330K images (>200K labeled)**
- **1.5 million object instances**
- **80 object categories**
- **91 stuff categories**
- **5 captions per image**
- **250,000 people with keypoints**

COCO
Common Objects in Context

Since then: Deeper backbone networks

- ResNet (He et al. 2015)
- 152 layers



Deep residual learning for image recognition,
He et al., CVPR 2016

Application: Faster R-CNN face detector

- VGG16 pre-trained on ImageNet
- Detector trained on the WIDER dataset (12k images 160k faces)

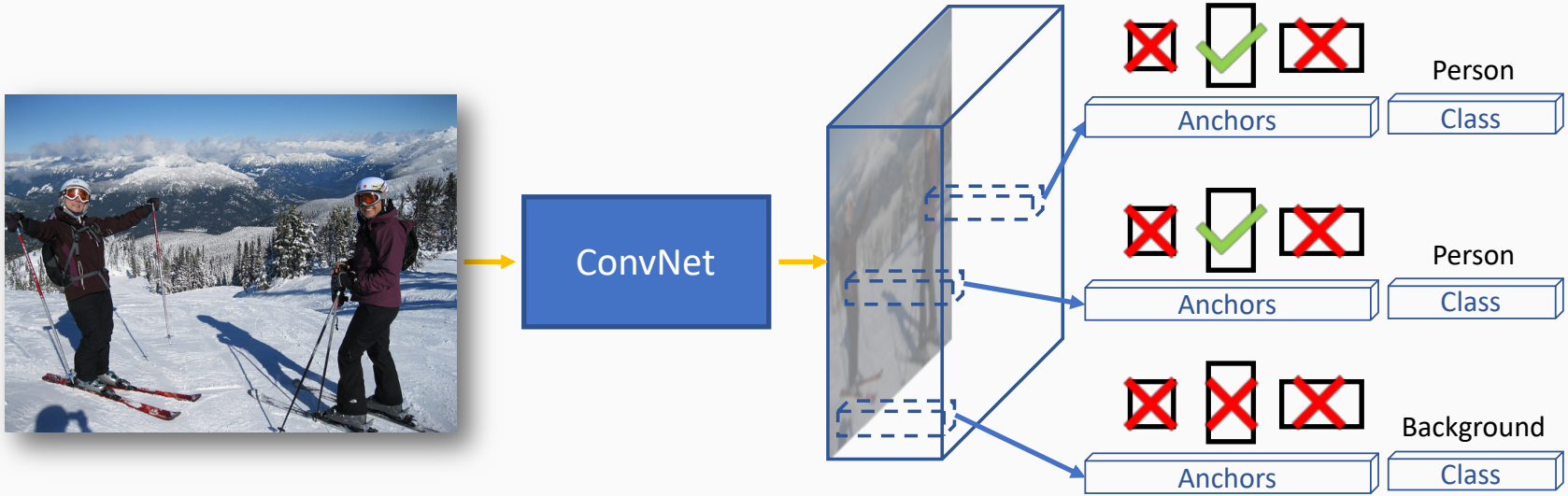
Credits: Sam Albanie, Qiong Cao

Single stage detectors

Two strands of detection architectures

- Detectors using region proposal networks (RPN)
  - Two stages: 1) RPN, followed by 2) features from regions for classification and regression of box
  - Possibly slow due to two steps
  - Examples: Faster RCNN, R-FCN
- Detector using unified framework (no explicit RPN)
  - Regions are build into the architecture (convolutional layers) -> possibly fast
  - Examples: YOLO, SSD, TinyFaces

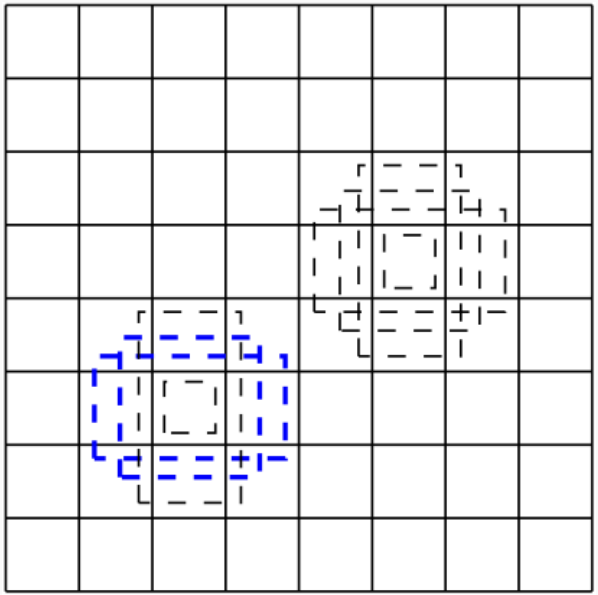
One-stage detectors



Redmond et al. CVPR 2017, Shen et al. ICCV 2017, Liu et al. ECCV 2016,
Fu et al. arXiv 2017, Lin et al. ICCV 2017, Zhang et al. CVPR 2018

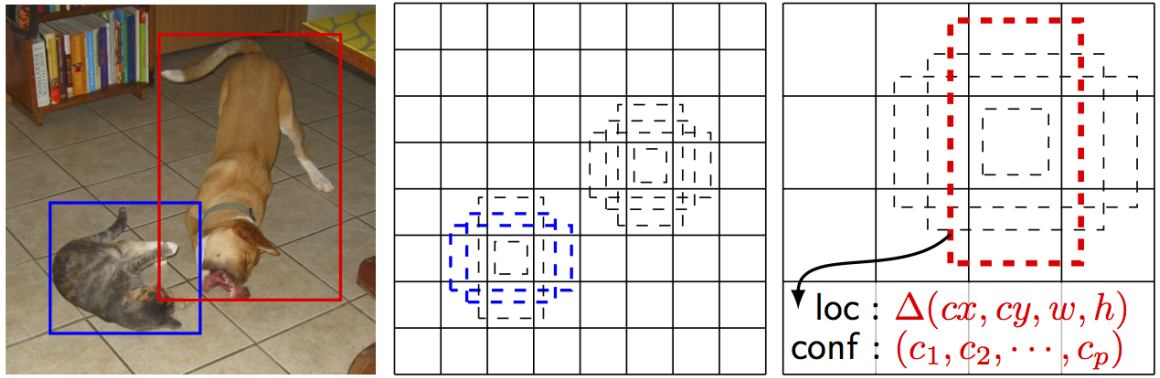
Single Shot MultiBox Detector (SSD)

- Fully convolutional detector (no RPN)
- Pre-defines regions:
  - Predict categories and box offsets
  - Multiple aspect ratios per cell
  - Similar to Faster R-CNN anchor boxes

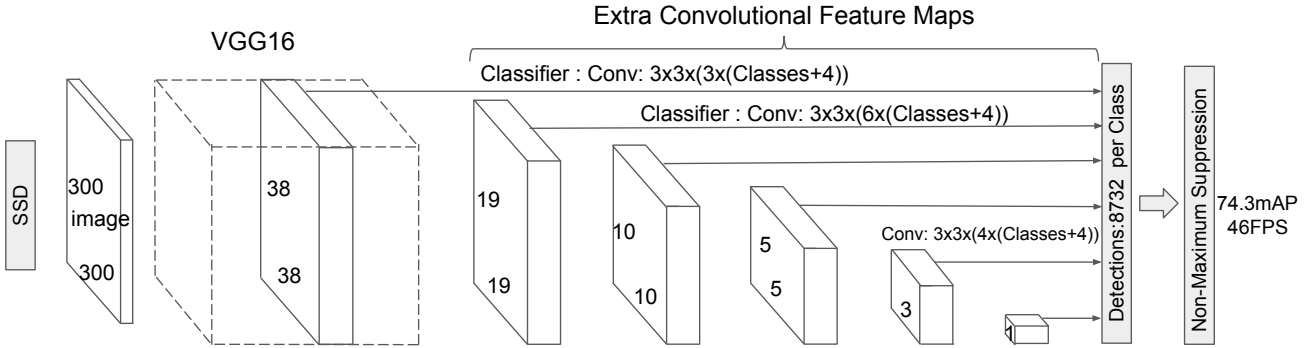


SSD: Single Shot MultiBox Detector, Liu et al., ECCV 2016

Single Shot MultiBox Detector (SSD)



SSD runs at 59 fps
cf. 7 fps for Faster R-CNN



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

PASCAL VOC Leaderboard Dec 2015

Detection challenge comp4: train on own data

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
Faster RCNN, ResNet (VOC+COCO) [?]	**83.8**	**92.1**	**88.4**	**84.8**	**75.9**	**71.4**	**86.3**	**87.8**	**94.2**	**66.8**	**89.4**	**69.2**	**93.9**	**91.9**	**90.9**	**89.6**	**67.9**	**88.2**	**76.8**	**90.3**	**80.0**	10-Dec-2015
ION [?]	76.4	87.5	84.7	76.8	63.8	58.3	82.6	79.0	90.9	57.8	82.0	64.7	88.9	86.5	84.7	82.3	51.4	78.2	69.2	85.2	73.5	23-Nov-2015
MNC baseline [?]	75.9	86.4	81.1	76.4	64.3	57.8	81.1	80.3	92.0	55.2	82.6	61.0	89.9	86.4	84.6	85.4	53.1	79.8	66.1	84.7	69.9	15-Dec-2015
Faster RCNN baseline (VOC+COCO) [?]	75.9	87.4	83.6	76.8	62.9	59.6	81.9	82.0	91.3	54.9	82.6	59.0	89.0	85.5	84.7	84.1	52.2	78.9	65.5	85.4	70.2	24-Nov-2015
LocNet [?]	74.8	86.3	83.0	76.1	60.8	54.6	79.9	79.0	90.6	54.3	81.6	62.0	89.0	85.7	85.5	82.8	49.7	76.6	67.5	83.2	67.4	06-Nov-2015
** HRCNN ** [?]	74.6	85.9	83.9	75.5	60.9	54.5	81.4	79.1	90.6	53.3	79.7	61.6	89.9	86.2	85.8	78.2	49.1	75.1	68.6	86.1	67.7	13-Nov-2015
MR_CNN_S_CNN_MORE_DATA [?]	73.9	85.5	82.9	76.6	57.8	62.7	79.4	77.2	86.6	55.0	79.1	62.2	87.0	83.4	84.7	78.9	45.3	73.4	65.8	80.3	74.0	06-Jun-2015
HyperNet_VGG [?]	71.4	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.9	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7	12-Oct-2015
HyperNet_SP [?]	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6	28-Oct-2015

Mean average precision 83.8

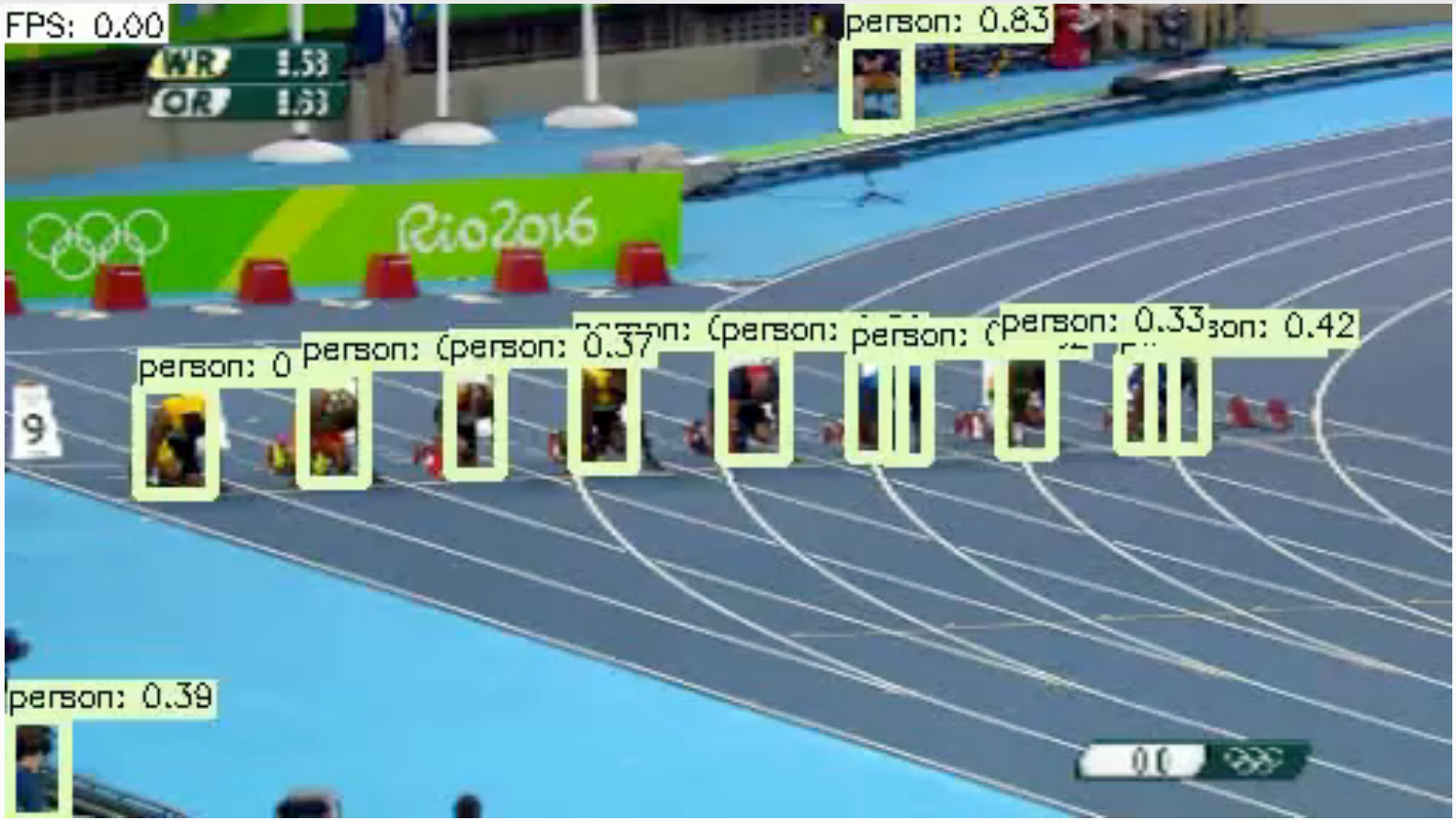
PASCAL VOC Leaderboard Dec 2018

Detection challenge comp4: train on own data

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
** Sogou_MM_GCFE_RCNN(ensemble model) ** [?]	**91.1**	95.9	94.6	**93.3**	**86.2**	87.1	**93.2**	**95.1**	97.1	**81.1**	**94.4**	**77.1**	96.5	96.6	**95.8**	95.4	77.9	**95.4**	**84.1**	95.0	**89.5**	25-Sep-2018
** Sogou_MM_GCFE_RCNN(single model) ** [?]	91.0	95.9	94.1	**93.3**	**86.2**	87.0	93.1	**95.1**	97.1	**81.1**	**94.4**	**77.1**	96.5	96.6	**95.8**	95.4	77.9	**95.4**	83.4	94.9	**89.5**	25-Sep-2018
ATLDET [?]	90.7	96.0	94.9	91.8	85.2	**87.6**	93.0	94.5	97.5	80.7	93.8	75.6	96.6	96.2	**95.8**	**95.5**	78.3	95.2	82.5	94.8	89.2	13-Aug-2018
FXRCNN (single model) [?]	90.7	**96.4**	**95.1**	92.0	84.3	87.1	92.8	94.4	97.4	80.7	93.5	76.0	96.7	**96.7**	95.6	**95.5**	78.3	94.6	83.3	**95.4**	88.0	13-Jul-2018
** tencent_retail_ft:DET ** [?]	90.0	95.6	94.0	91.8	84.2	86.4	92.8	94.4	96.8	78.7	93.3	74.4	95.9	95.7	95.2	95.2	**78.8**	94.0	80.1	94.0	88.4	02-Jan-2019
Ali_DCN_SSD_ENSEMBLE [?]	89.2	95.4	93.7	91.8	82.8	81.7	92.4	93.4	97.6	75.7	94.1	74.2	96.4	95.1	94.2	93.3	72.5	94.1	82.8	94.6	87.7	28-May-2018
VIM_SSD(COCO+07++12, single model, one-stage) [?]	89.0	96.0	93.0	90.3	83.4	80.6	91.9	94.4	96.2	77.5	93.3	75.1	95.2	95.1	94.2	93.6	72.0	93.6	82.7	94.5	86.6	27-Jun-2018
FOCAL_DRFCN(VOC+COCO, single model) [?]	88.8	95.0	93.3	91.8	82.9	81.9	91.6	93.0	97.1	76.7	92.5	71.7	96.2	94.9	94.2	93.7	75.3	93.3	80.0	94.7	85.4	01-Mar-2018
R4D_faster_rcnn [?]	88.6	94.6	92.3	91.3	82.3	79.4	91.8	91.8	97.4	76.6	93.6	75.3	97.0	94.6	93.5	92.6	75.1	92.0	80.9	94.4	86.5	20-Nov-2016
FF_CSSD(VOC+COCO, one-stage, single model) [?]	88.4	95.4	93.5	90.8	82.8	78.4	90.4	91.8	96.9	75.1	92.7	74.2	95.7	95.1	94.2	93.0	71.6	93.9	81.9	94.1	86.7	28-May-2018

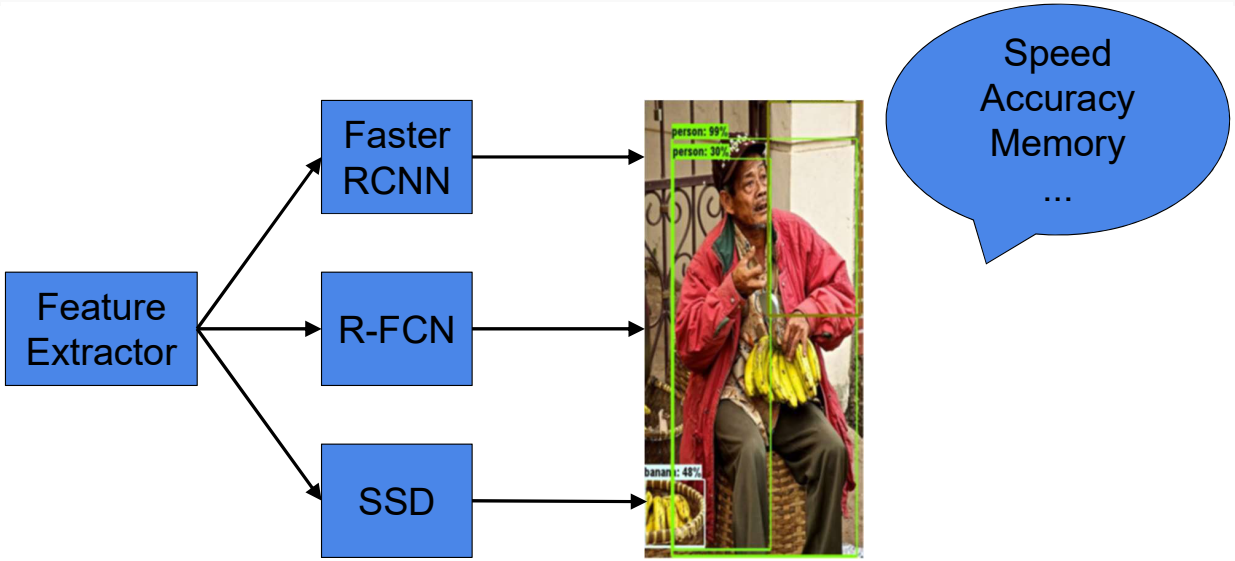
Mean average precision 91.1

Single Shot MultiBox Detector - video example



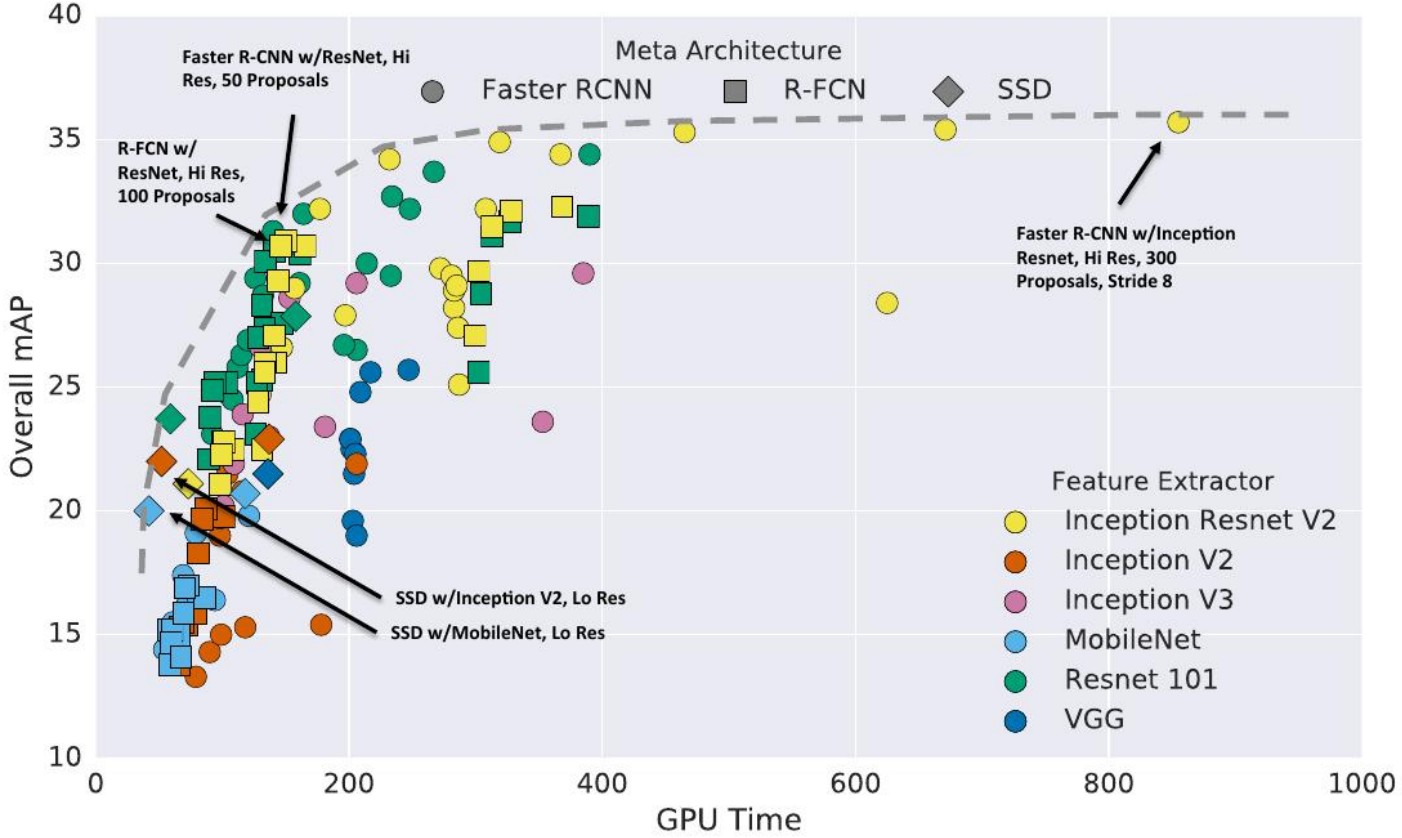
SSD: Single Shot MultiBox Detector, Liu et al., ECCV 2016

Summary and comparison



Speed/accuracy trade-offs for modern convolutional object detectors, Huang et al. CVPR 2017
Unified tensor flow architecture for comparing speed, accuracy, and memory usage

Accuracy vs speed (COCO)



Speed/accuracy trade-offs for modern convolutional object detectors, Huang et al. CVPR 2017

Recent trends and highlights

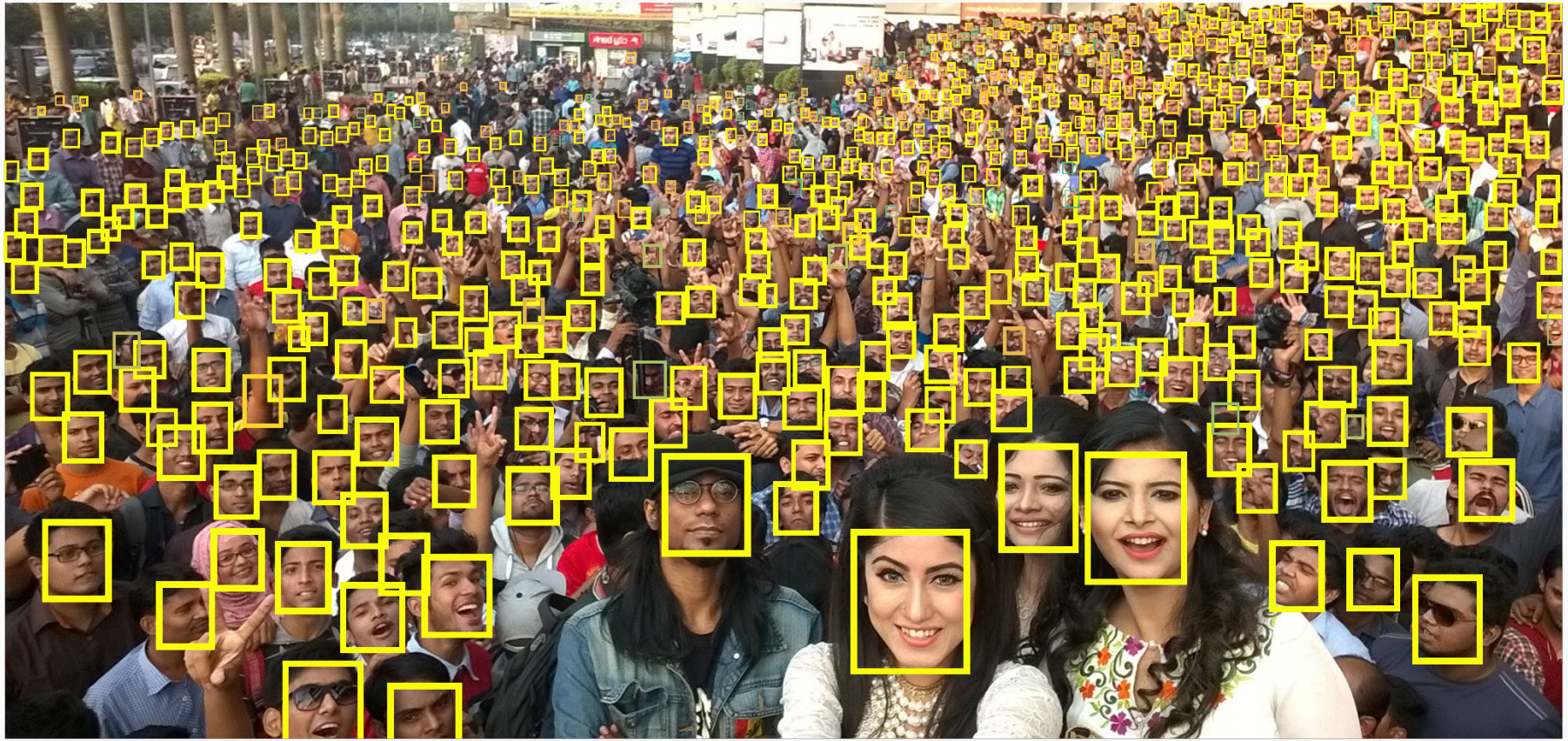
Recent trends 2016->

- Multiple scales
- More context
- Larger images
- Faster R-CNN improvements
  - Detection + X, e.g. instance segmentation or pose, Mask R-CNN
  - Reduce size fo fully connected layers
- Highly optimised architectures (EfficientDet and YOLOv4)

Highlights

- Finding Tiny Faces
- CornerNet
- Mask R-CNN
- EfficientDet
- YOLOv4

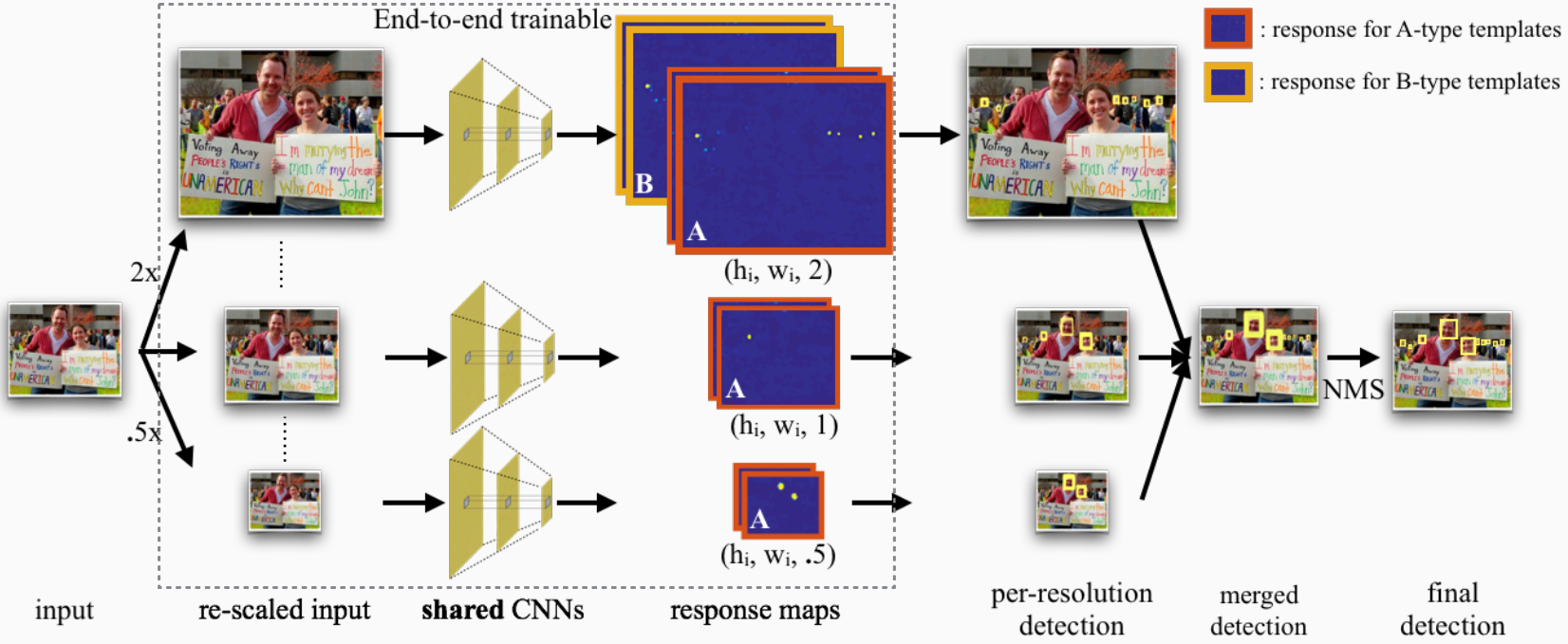
Finding Tiny Faces



Finding tiny faces, Hu et al. CVPR 2017

Finding Tiny Faces

Fully convolutional detection networks



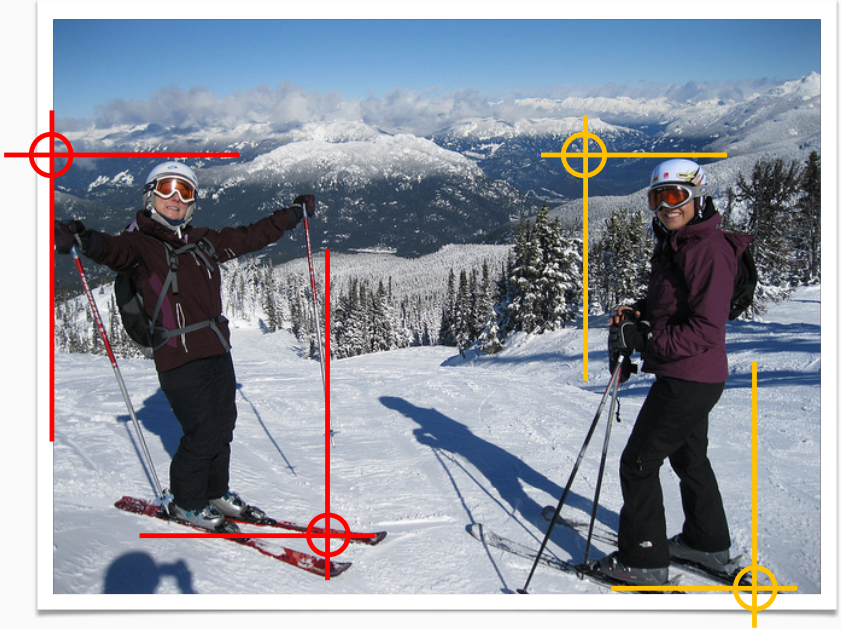
Finding tiny faces, Hu et al. CVPR 2017

CornerNet

- Drawbacks of anchor boxes
  - Need a large number of anchors
    -> Tiny fraction are positive examples
    -> Slower training [Lin et al. ICCV 2017]
  - Extra hyperparameters -
    sizes, aspect ratios

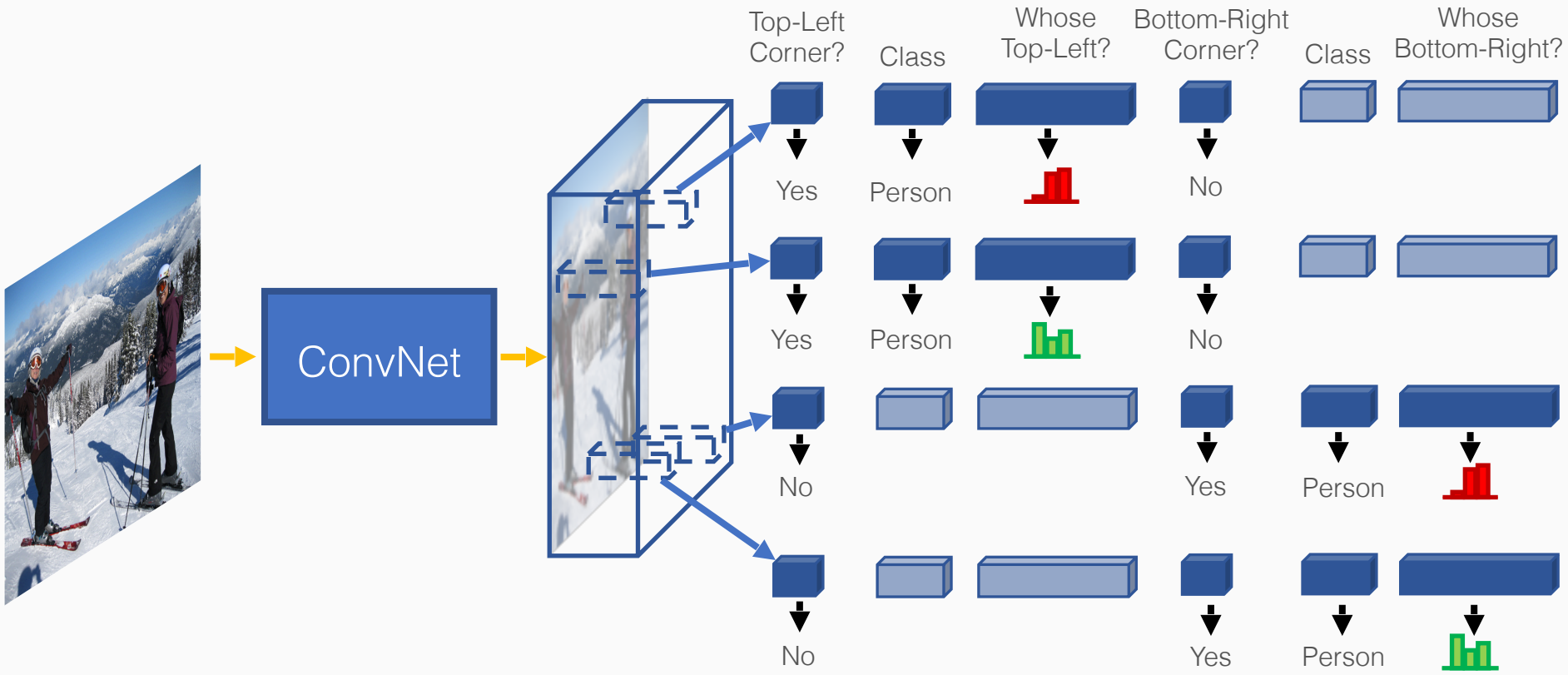
CornerNet: Detecting objects as paired key points, Law et al., ECCV 2018

CornerNet: Detecting objects as paired keypoints



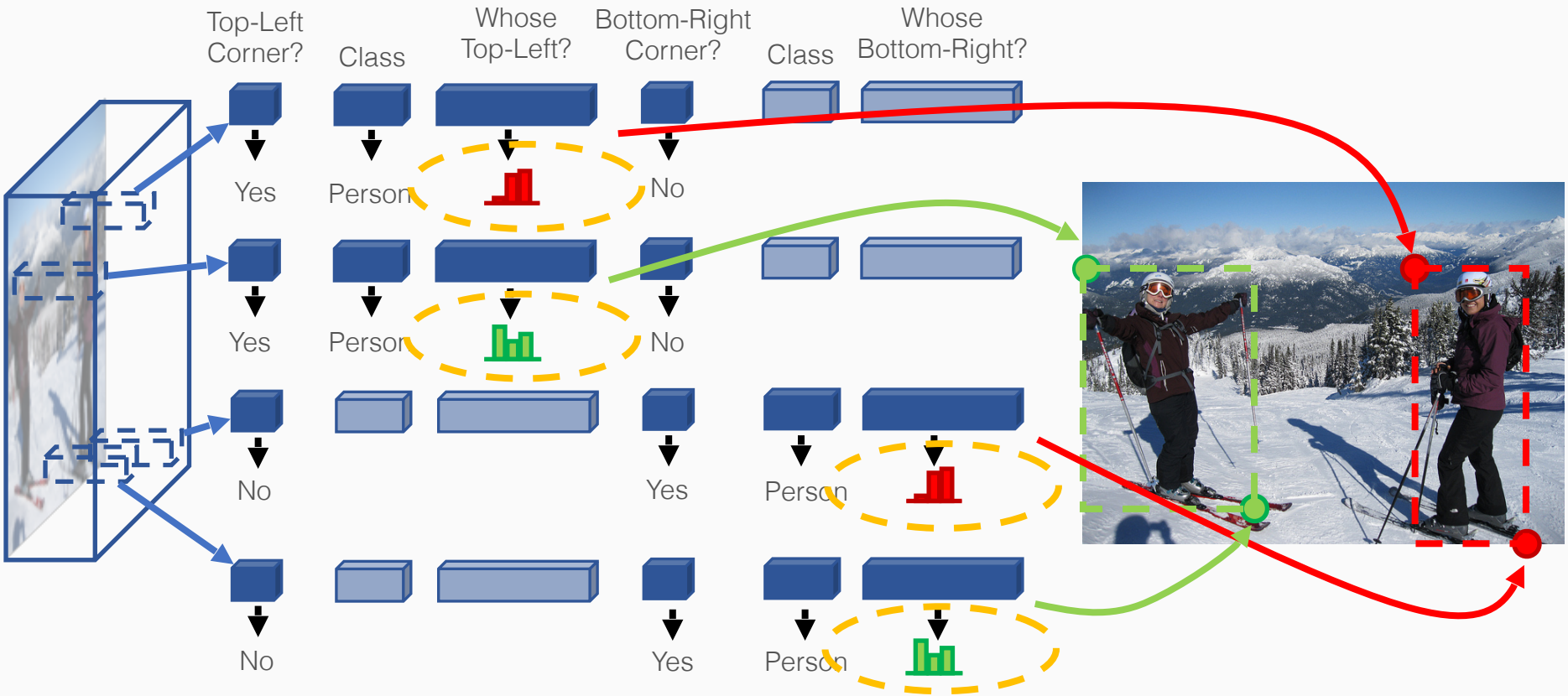
CornerNet: Detecting objects as paired key points, Law et al., ECCV 2018

CornerNet - Architecture



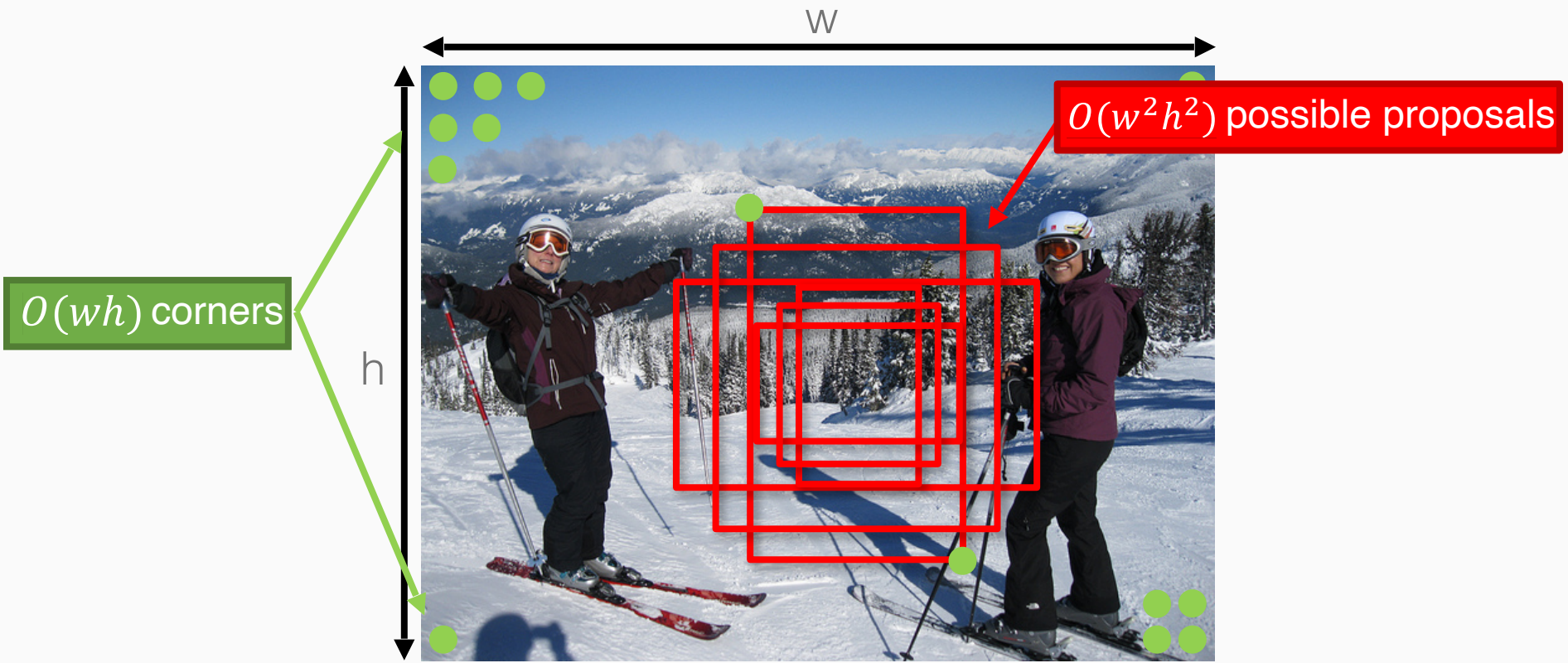
CornerNet: Detecting objects as paired key points, Law et al., ECCV 2018

CornerNet - Architecture



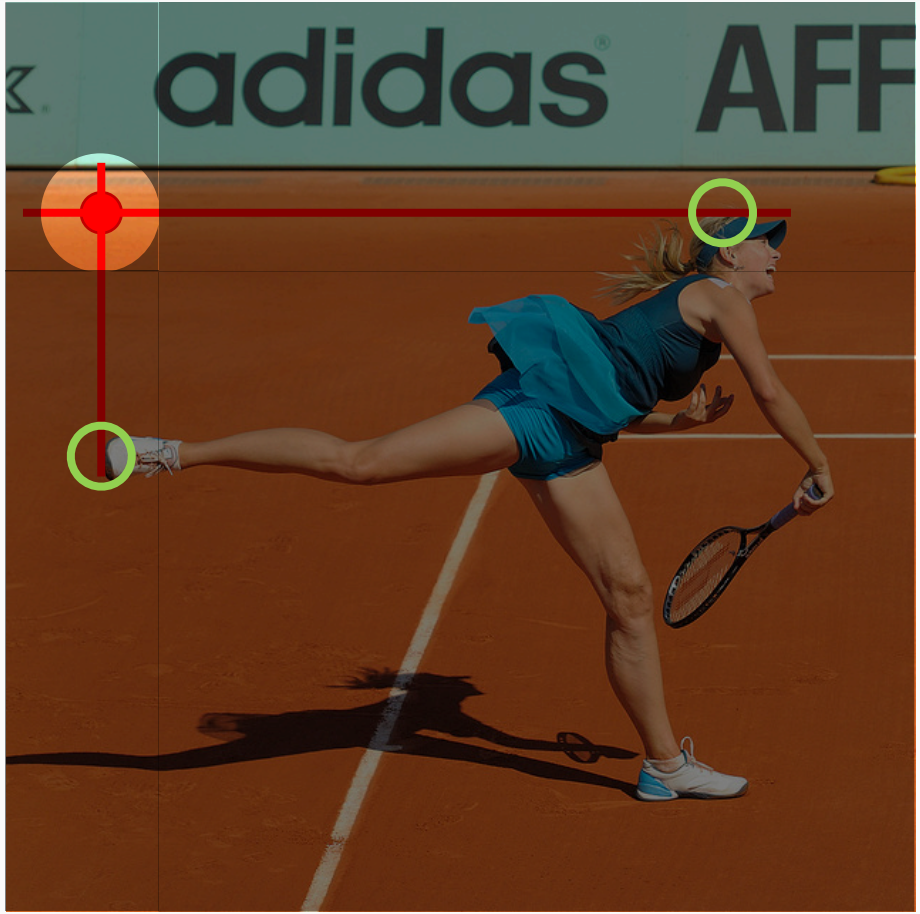
CornerNet: Detecting objects as paired key points, Law et al., ECCV 2018

CornerNet - Advantages of detecting corners



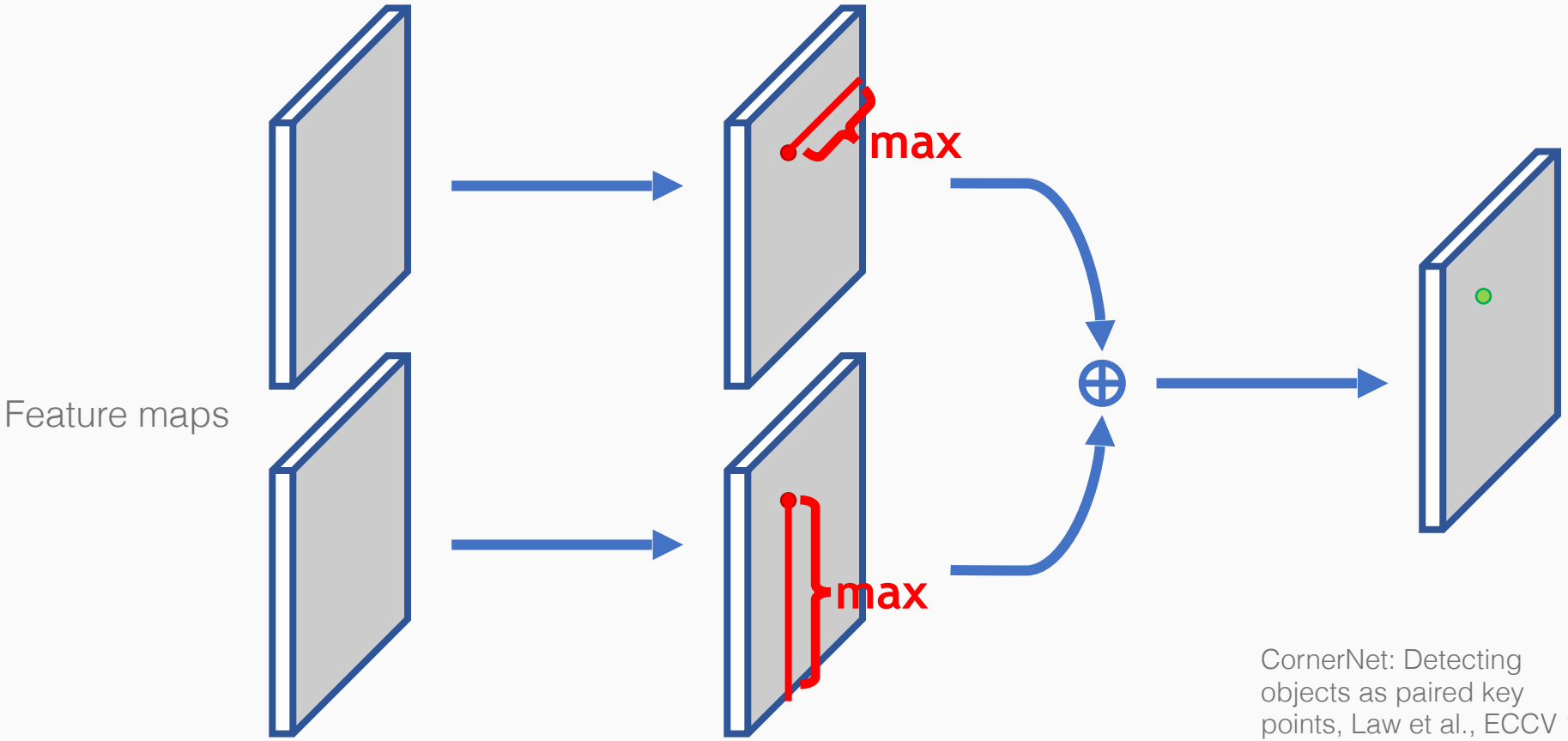
Represent $O(w^2h^2)$ possible proposals using only $O(wh)$ corners

CornerNet - Corner pooling



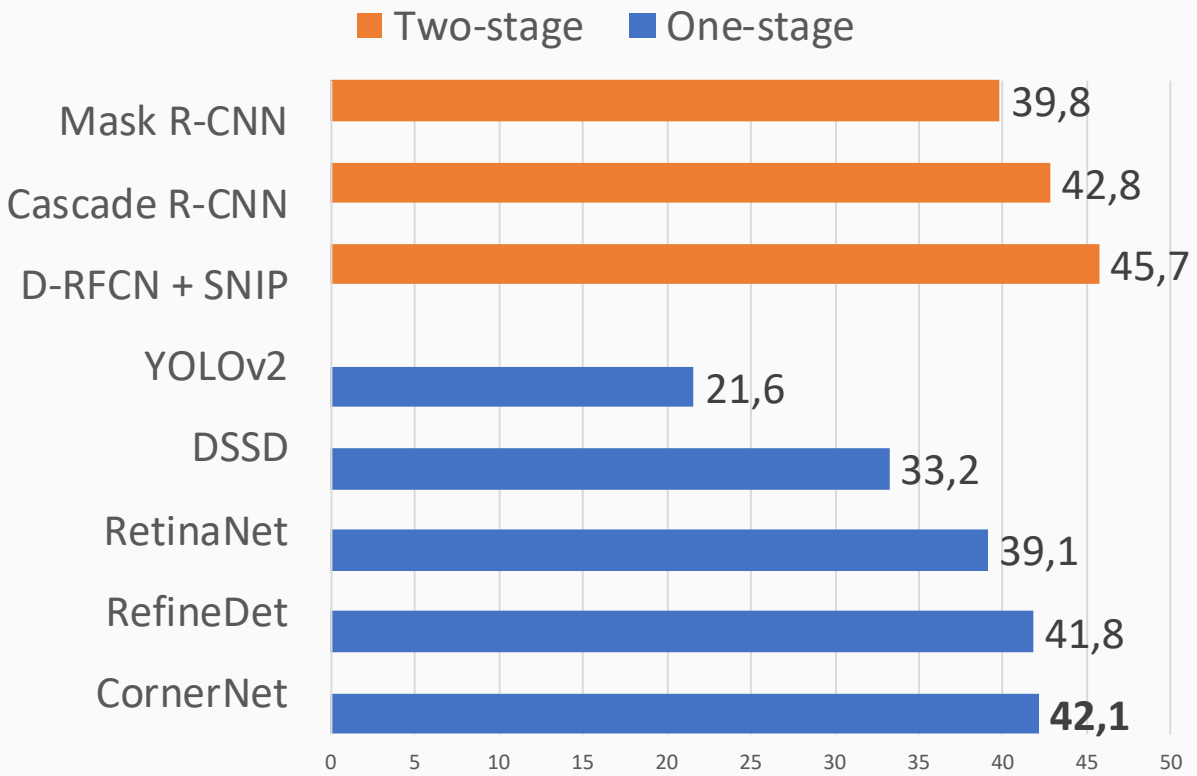
CornerNet: Detecting objects as paired key points, Law et al., ECCV 18

CornerNet - Corner pooling



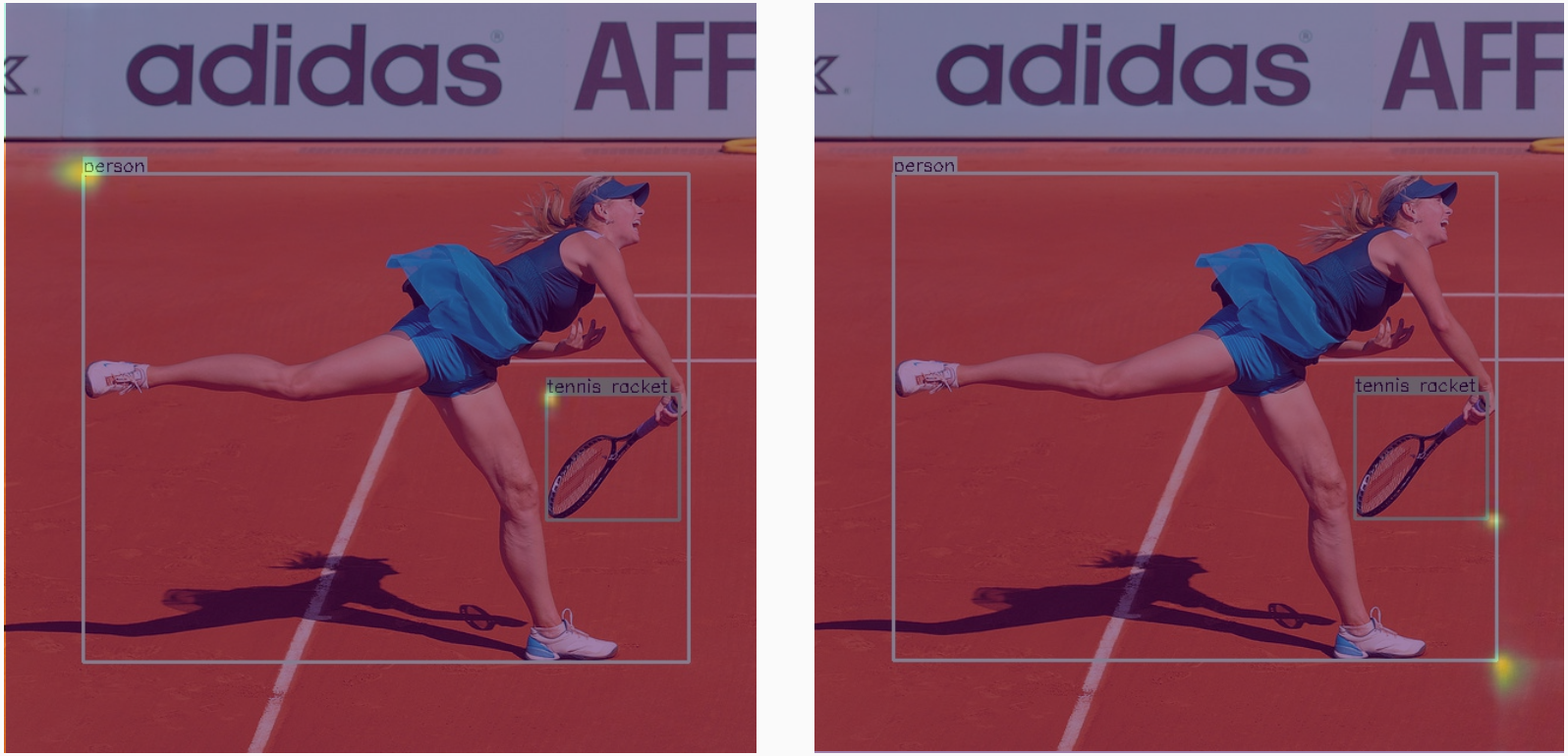
CornerNet: Detecting objects as paired key points, Law et al., ECCV 18

CornerNet - Comparison with others (COCO)



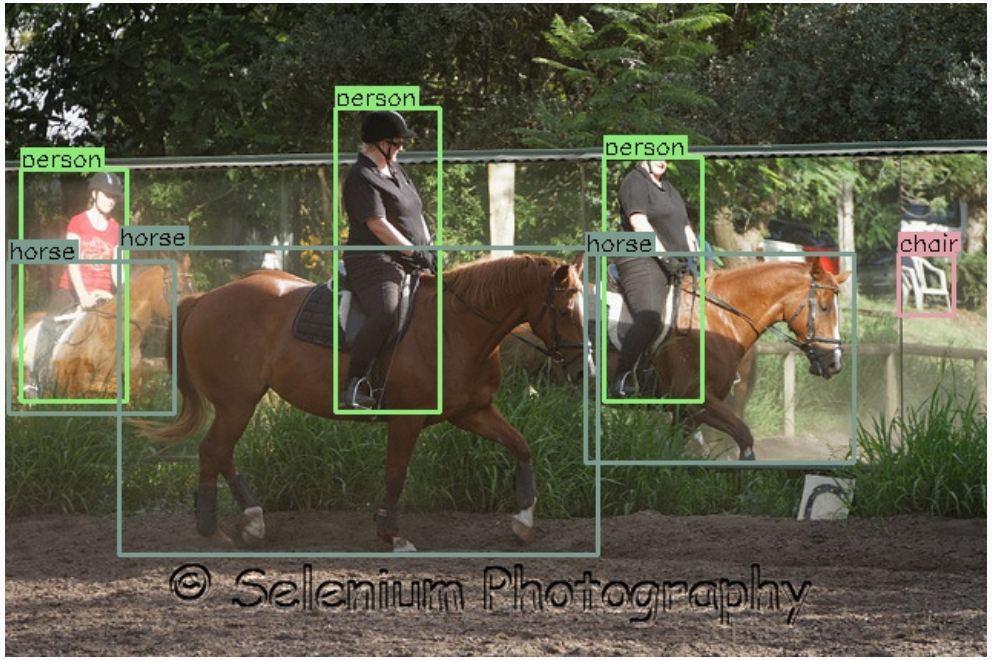
CornerNet: Detecting objects as paired key points, Law et al., ECCV 2018

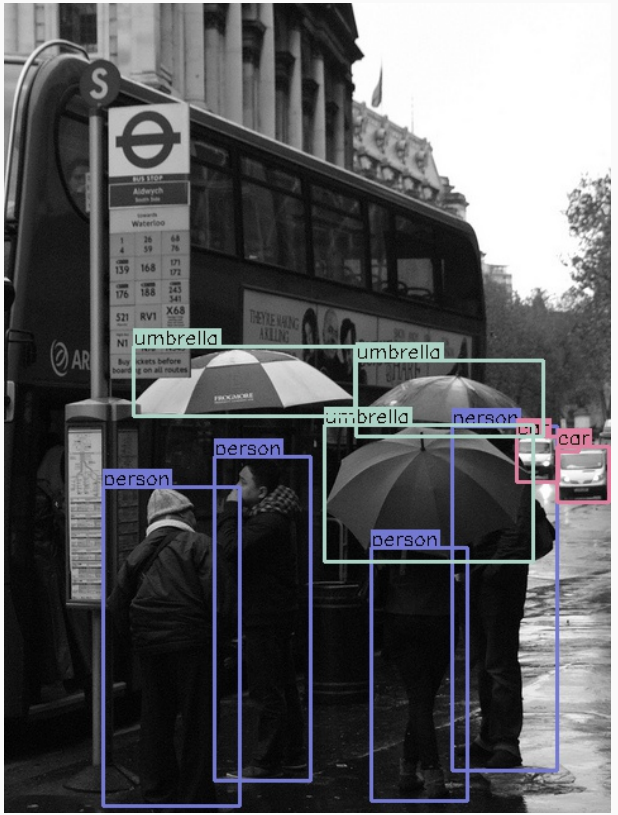
CornerNet - Examples



CornerNet: Detecting objects as paired key points, Law et al., ECCV 2018

CornerNet - Examples

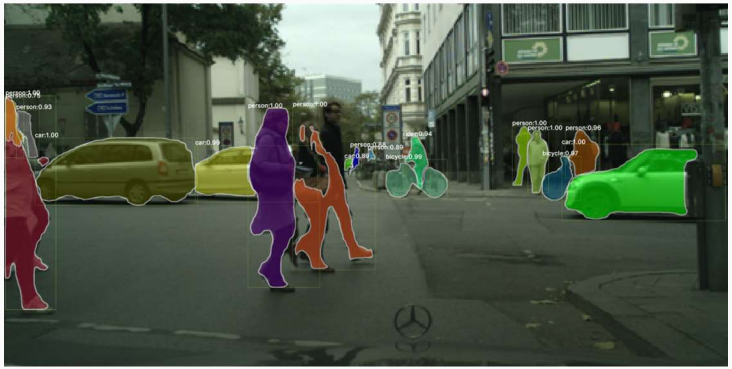


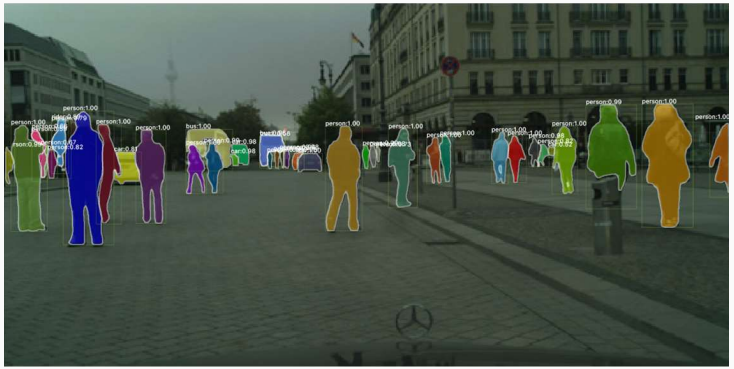


CornerNet: Detecting objects as paired key points, Law et al., ECCV 2018

Instance segmentation

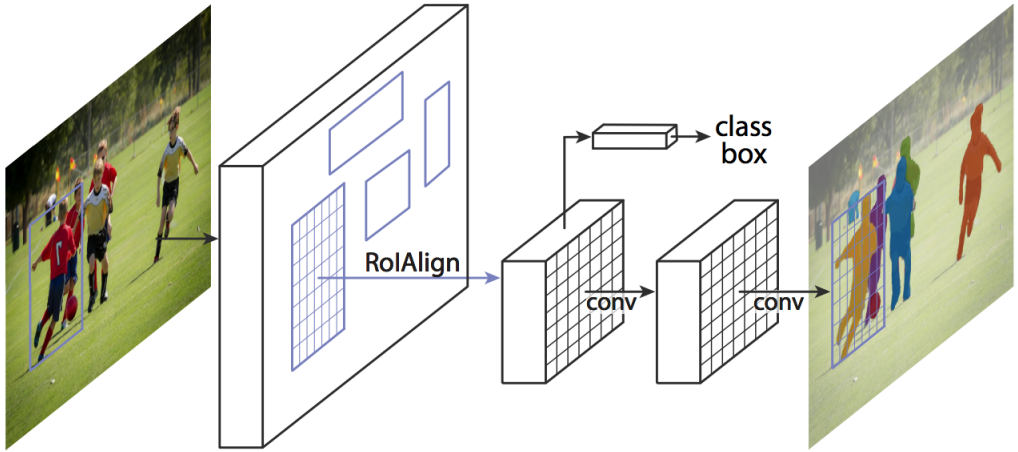
- Given an image produce instance-level segmentation
  - Which class does each pixel belong to?
  - Which instance does each pixel belong to?





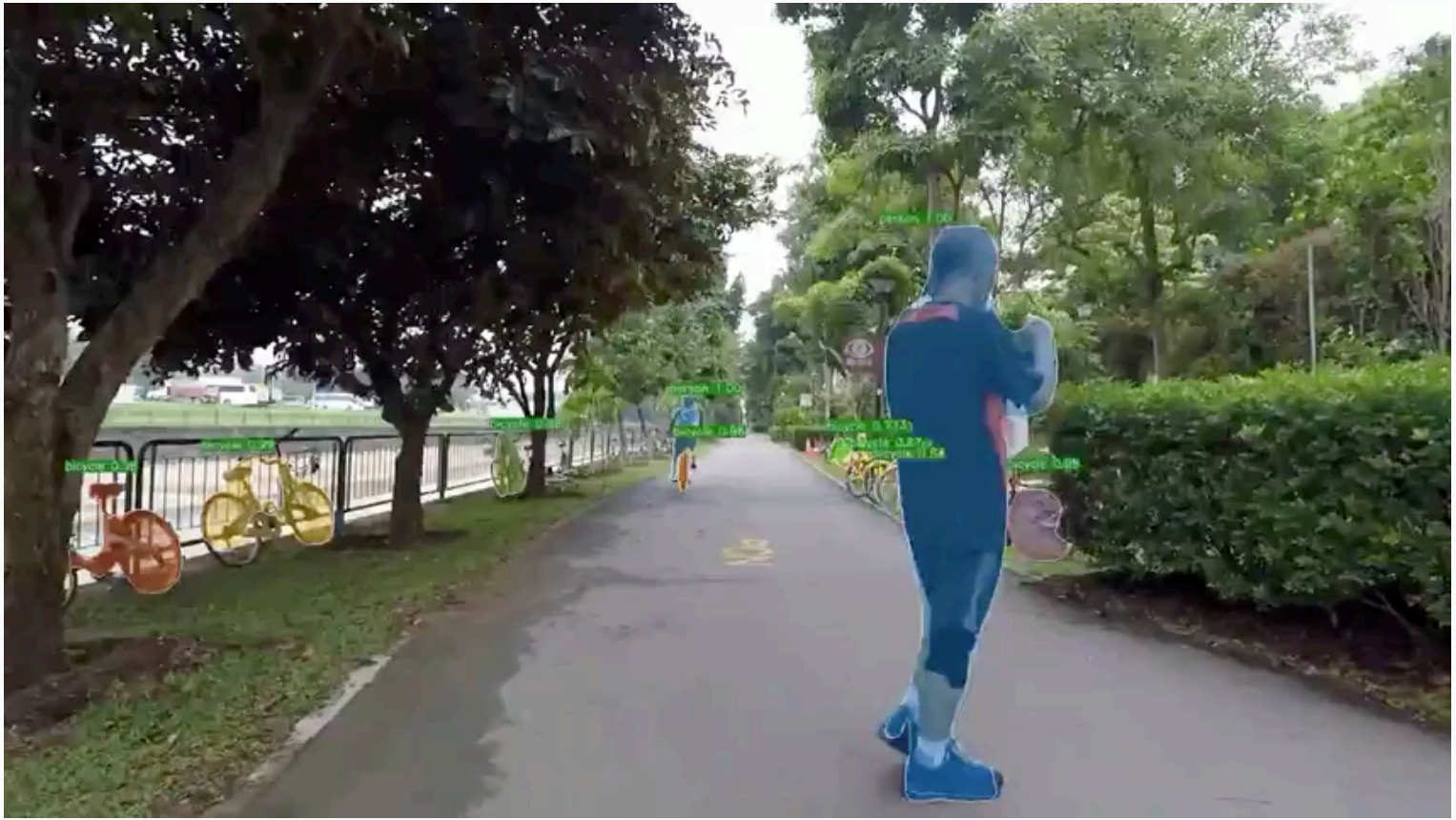
Mask R-CNN

- Extend Faster R-CNN to predict mask as well as a box



Mask R-CNN, He et al., CVPR 2017

Mask R-CNN - video example

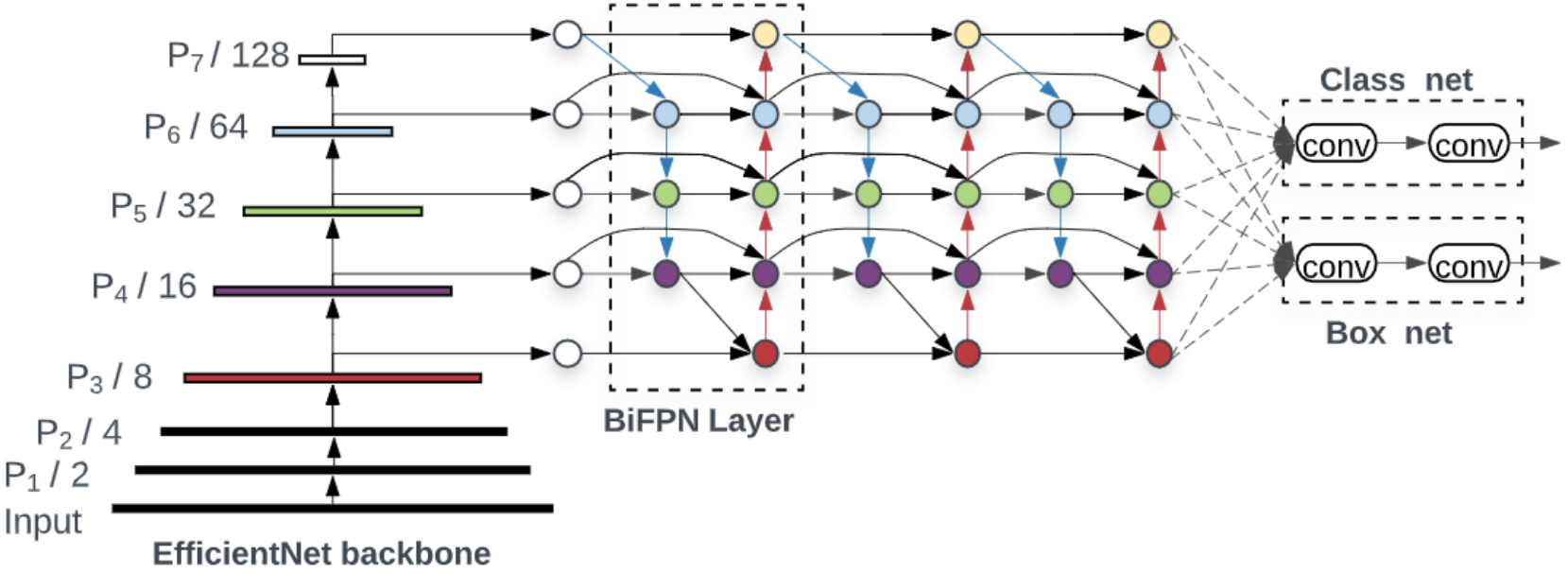


Mask R-CNN, He et al., CVPR 2017

https://www.youtube.com/watch?v=UWtac4cFERM

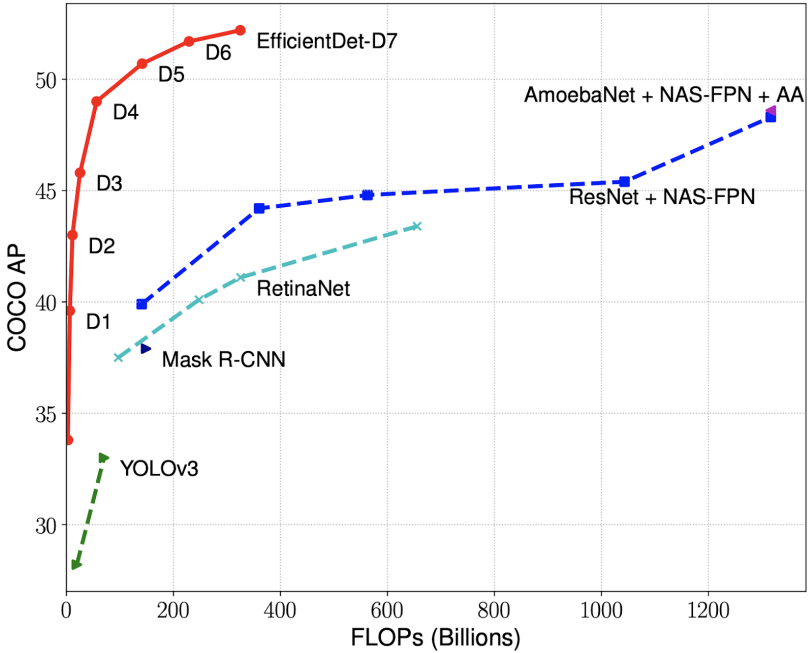
EfficientDet

- Largely follows the one-stage detectors paradigm



EfficientDet: Scalable and efficient object detection, Tan et al., CVPR 2020

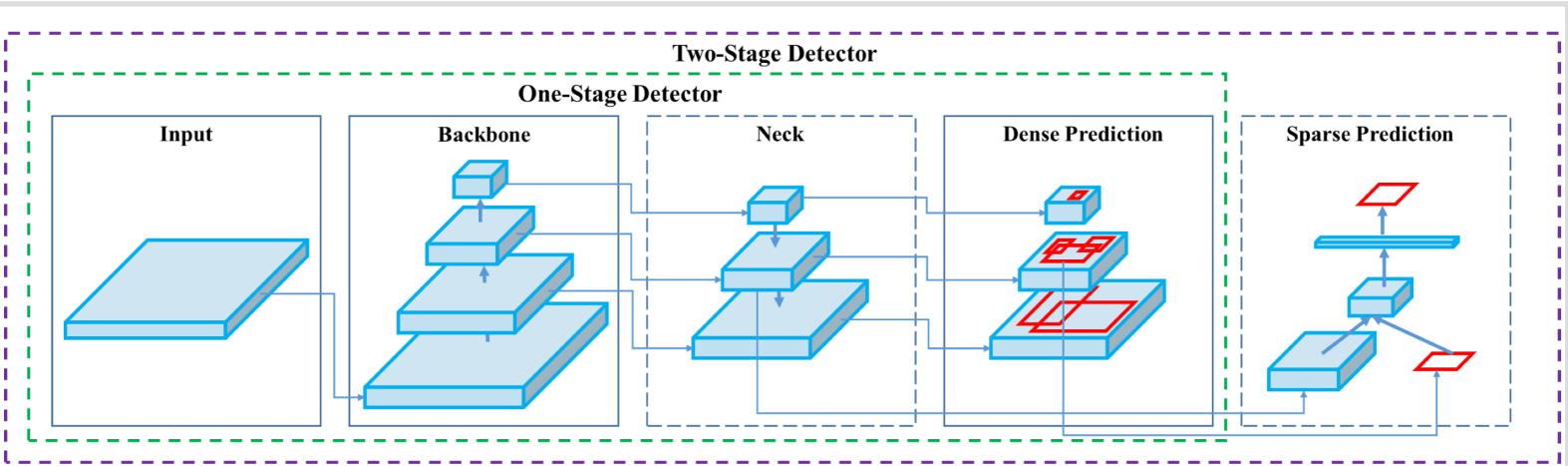
EfficientDet



Model	test-dev AP	test-dev AP_{50}	test-dev AP_{75}	val AP
EfficientDet-D0 (512)	**34.6**	**53.0**	**37.1**	**34.3**
YOLOv3 [34]	33.0	57.9	34.4	-
EfficientDet-D1 (640)	**40.5**	**59.1**	**43.7**	**40.2**
RetinaNet-R50 (640) [24]	39.2	58.0	42.3	39.2
RetinaNet-R101 (640)[24]	39.9	58.5	43.0	39.8
EfficientDet-D2 (768)	**43.9**	**62.7**	**47.6**	**43.5**
Detectron2 Mask R-CNN R101-FPN [1]	-	-	-	42.9
Detectron2 Mask R-CNN X101-FPN [1]	-	-	-	44.3
EfficientDet-D3 (896)	**47.2**	**65.9**	**51.2**	**46.8**
ResNet-50 + NAS-FPN (1024) [10]	44.2	-	-	-
ResNet-50 + NAS-FPN (1280) [10]	44.8	-	-	-
ResNet-50 + NAS-FPN (1280@384)[10]	45.4	-	-	-
EfficientDet-D4 (1024)	**49.7**	**68.4**	**53.9**	**49.3**
AmoebaNet+ NAS-FPN +AA(1280)[45]	-	-	-	48.6
EfficientDet-D5 (1280)	**51.5**	**70.5**	**56.1**	**51.3**
Detectron2 Mask R-CNN X152 [1]	-	-	-	50.2
EfficientDet-D6 (1280)	**52.6**	**71.5**	**57.2**	**52.2**
AmoebaNet+ NAS-FPN +AA(1536)[45]	-	-	-	50.7
EfficientDet-D7 (1536)	**53.7**	**72.4**	**58.4**	**53.4**
EfficientDet-D7x (1536)	**55.1**	**74.3**	**59.9**	**54.4**

EfficientDet: Scalable and efficient object detection, Tan et al., CVPR 2020

YOLO v4



Input: { Image, Patches, Image Pyramid, … }

Backbone: { VGG16 [68], ResNet-50 [26], ResNeXt-101 [86], Darknet53 [63], … }

Neck: { FPN [44], PANet [49], Bi-FPN [77], … }

Head:

Dense Prediction: { RPN [64], YOLO [61, 62, 63], SSD [50], RetinaNet [45], FCOS [78], … }

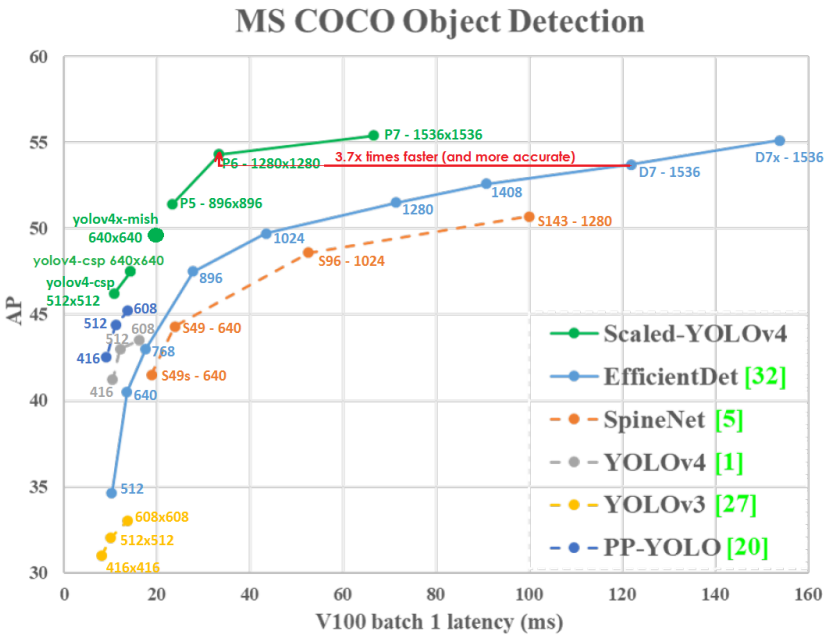
Sparse Prediction: { Faster R-CNN [64], R-FCN [9], … }

YOLOv4 consists of:

- Backbone: CSPDarknet53 [81]
- Neck: SPP [25], PAN [49]
- Head: YOLOv3 [63]

YOLOv4: Optimal speed and accuracy of object detection, Bochkovskiy et al., arXiv 2020

Scaled-YOLO v4

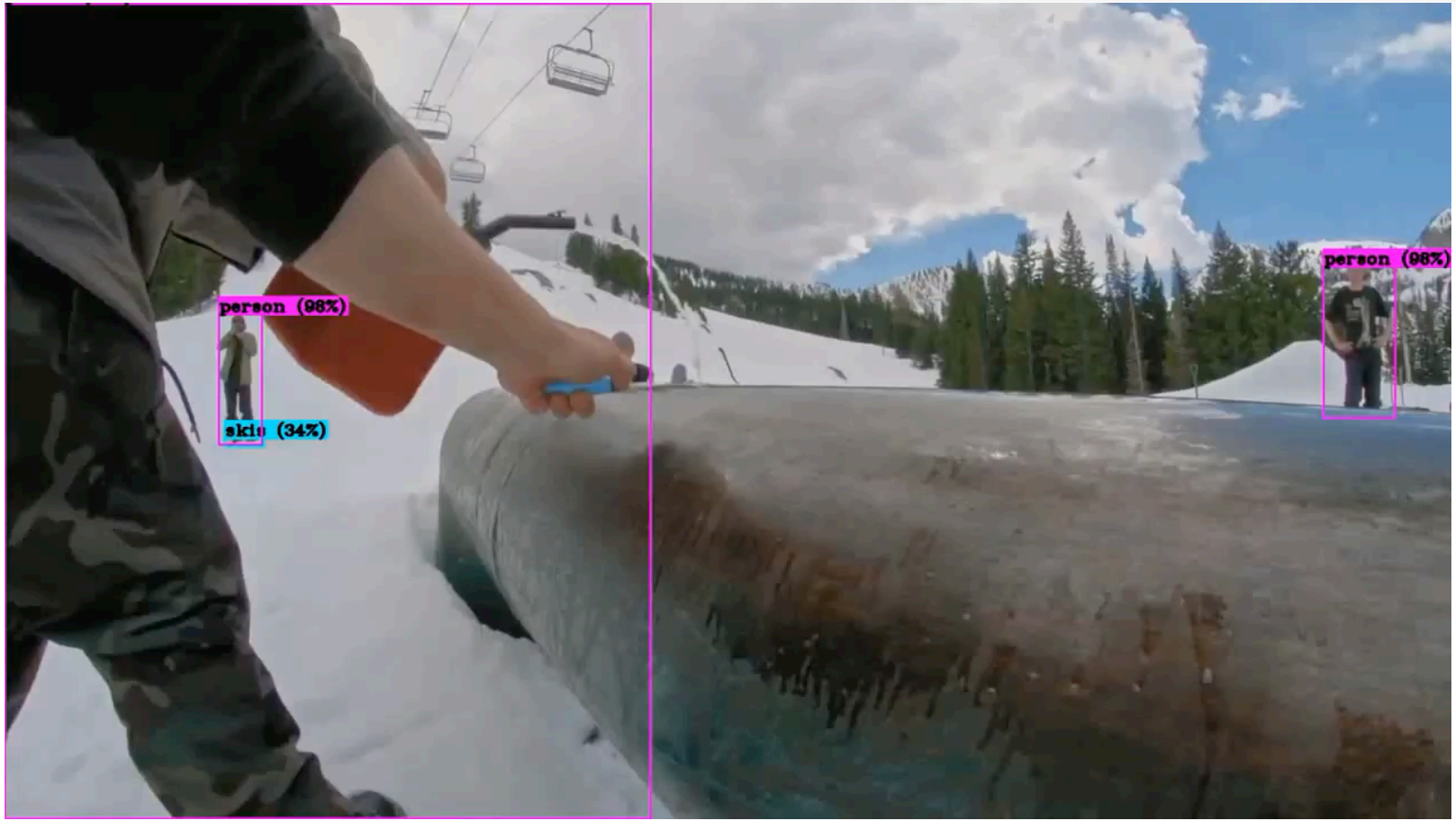


RANK	MODEL	BOX AP	AP50	AP75	APS	APM	APL	EXTRA TRAINING DATA	PAPER	CODE	RESULT	YEAR
1	YOLOv4-P7 (CSP-P7, multi-scale)	55.8	73.2	61.2	38.8	60.1	68.2	×	Scaled-YOLOv4: Scaling Cross Stage Partial Network			2020
2	DetectoRS (ResNeXt-101-64x4d, multi-scale)	55.7	74.2	61.1	37.7	58.4	68.1	×	DetectoRS: Detecting Objects with Recursive Feature Pyramid and Switchable Atrous Convolution			2020
3	YOLOv4-P7 (CSP-P7, single-scale)	55.4	73.3	60.7	38.1	59.5	67.4	×	Scaled-YOLOv4: Scaling Cross Stage Partial Network			2020
4	EfficientDet-D7x (single-scale)	55.1	74.3	59.9	37.2	57.9	68.0	×	EfficientDet: Scalable and Efficient Object Detection			2019
5	CSP-p6 + Mish (multi-scale)	54.9	72.6	60.2	37.4	58.8	66.7	×	Mish: A Self Regularized Non-Monotonic Activation Function			2019
6	DetectoRS (ResNeXt-101-32x4d, multi-scale)	54.7	73.5	60.1	37.4	57.3	66.4	×	DetectoRS: Detecting Objects with Recursive Feature Pyramid and Switchable Atrous Convolution			2020

Accuracy rating of published neural networks: https://paperswithcode.com/sota/object-detection-on-coco

Scaled-YOLOv4: Scaling cross stage partial networks, Wang et al., arXiv 2020

YOLO v4



https://www.youtube.com/watch?v=1_SiUOYUoOI&feature=emb_logo

That’s it folks!