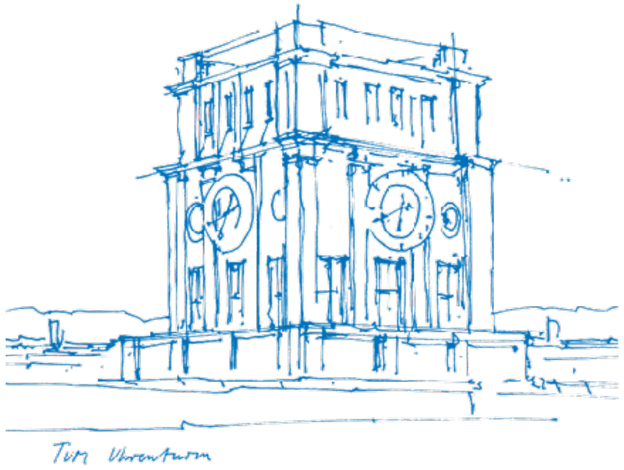
Computer Vision III:

Two-stage object detectors

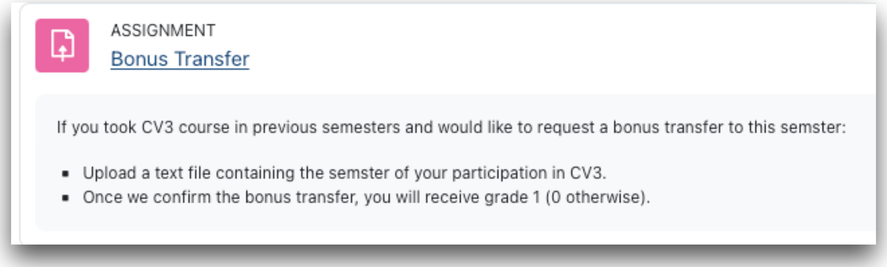
Nikita Araslanov
15.11.2022

Content credit:
Prof. Laura Leal-Taixé
https://dvl.in.tum.de



Announcement

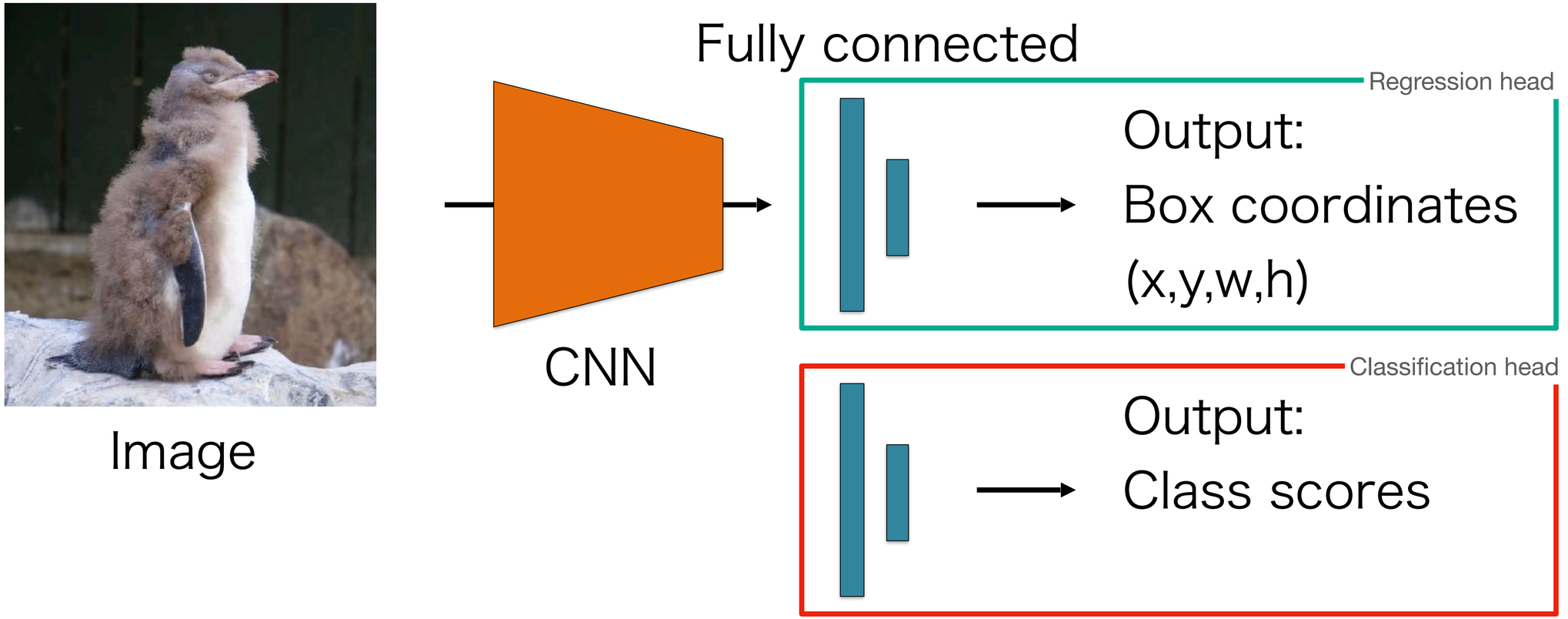
- Bonus transfer for assignments
- If you earned the CV3 bonus in previous semesters:
  - Go to Moodle
  - Find “Bonus Transfer”
  - Upload a text file (e.g. with the semester of participation)
  - We will double-check and will provide a feedback
  - Deadline: November 30



Lecture 02 recap

Localisation and classification

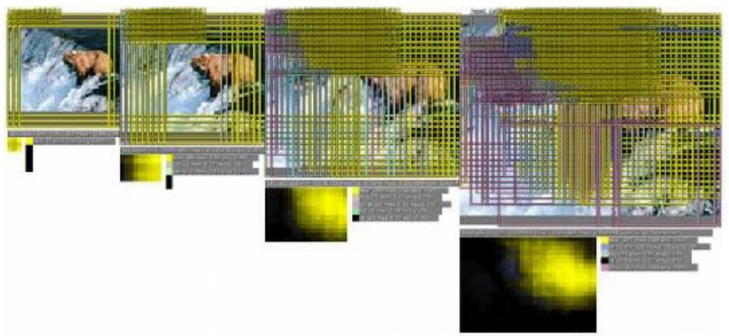
- Bounding box regression and classification



Overfeat

- In practice: use many sliding window locations and multiple scales

Window positions + score maps

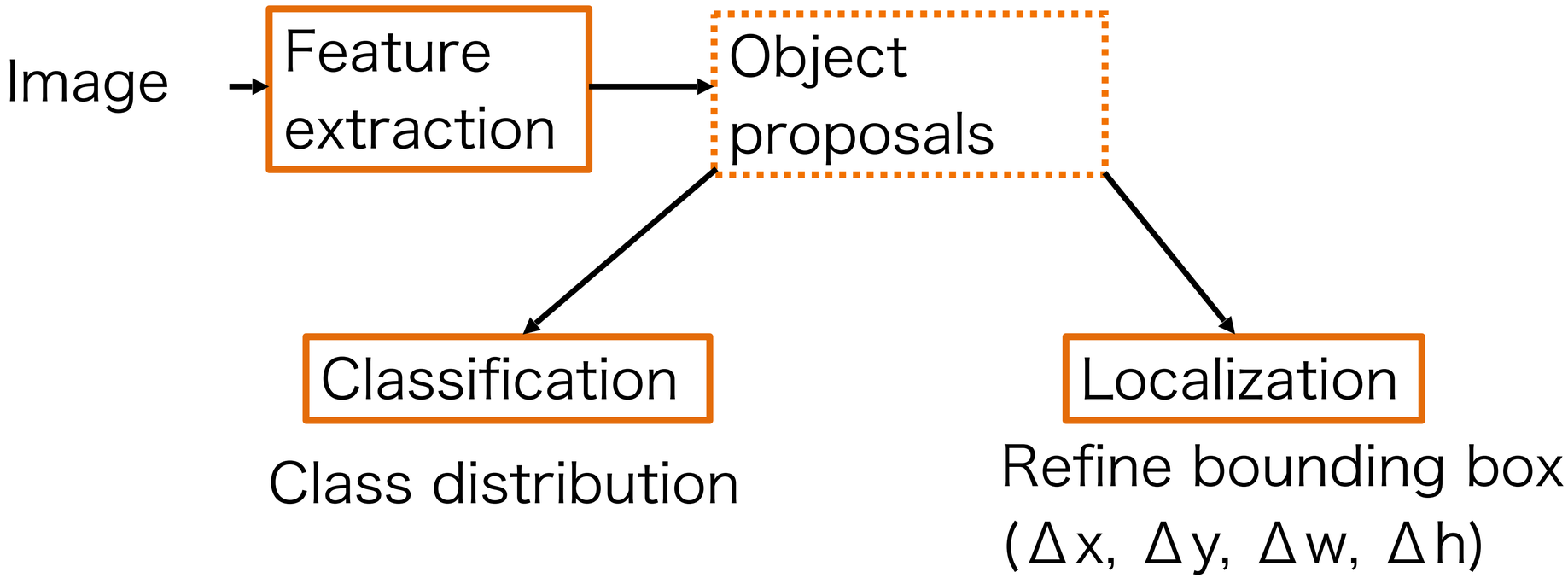


Box regression outputs

Final Prediction

Sermanet et al, “Overfeat: Integrated Recognition, Localization and Detection using Convolutional Networks”, ICLR 2014

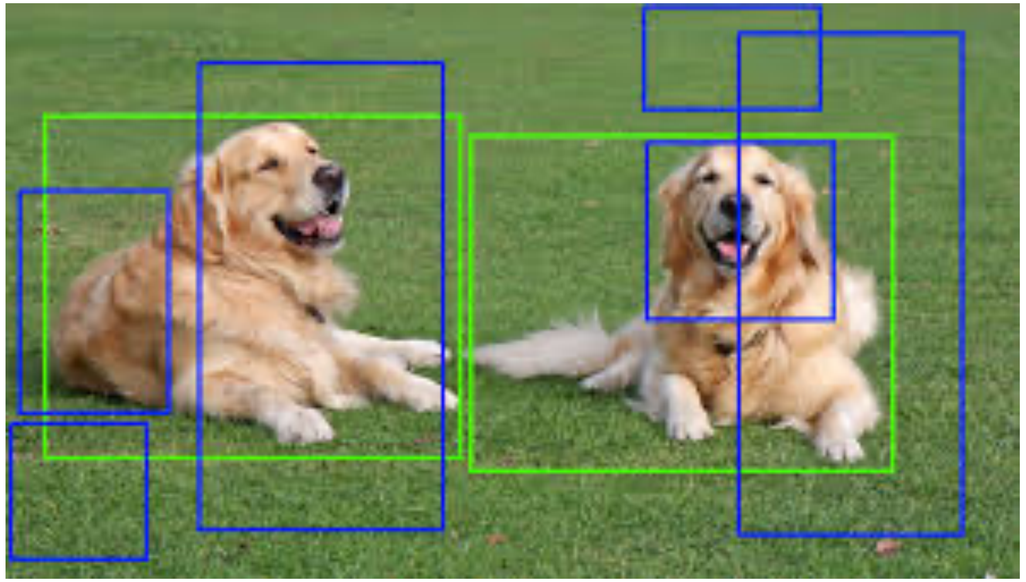
Two-stage detectors



Region proposals

- We heuristic-based methods that gives us “interesting” regions in an image

1. Obtain region proposals.
2. Classify & refine them.

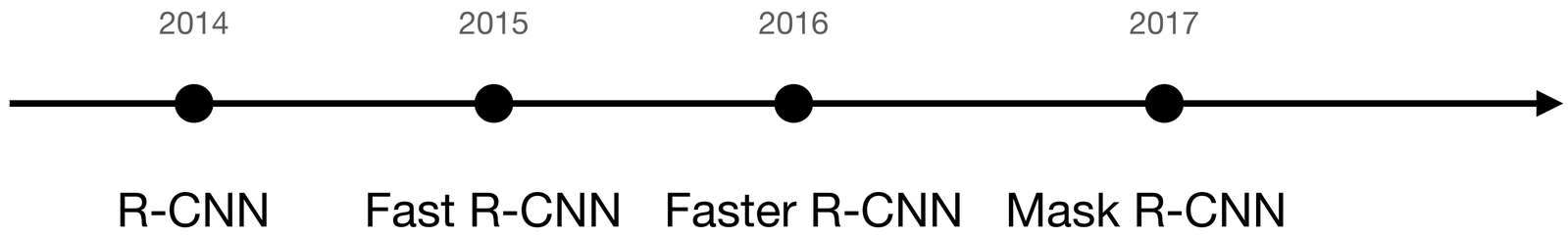


Uijlings et al. Selective Search for Object Recognition. IJCV 2013

R-CNN family

R-CNN family

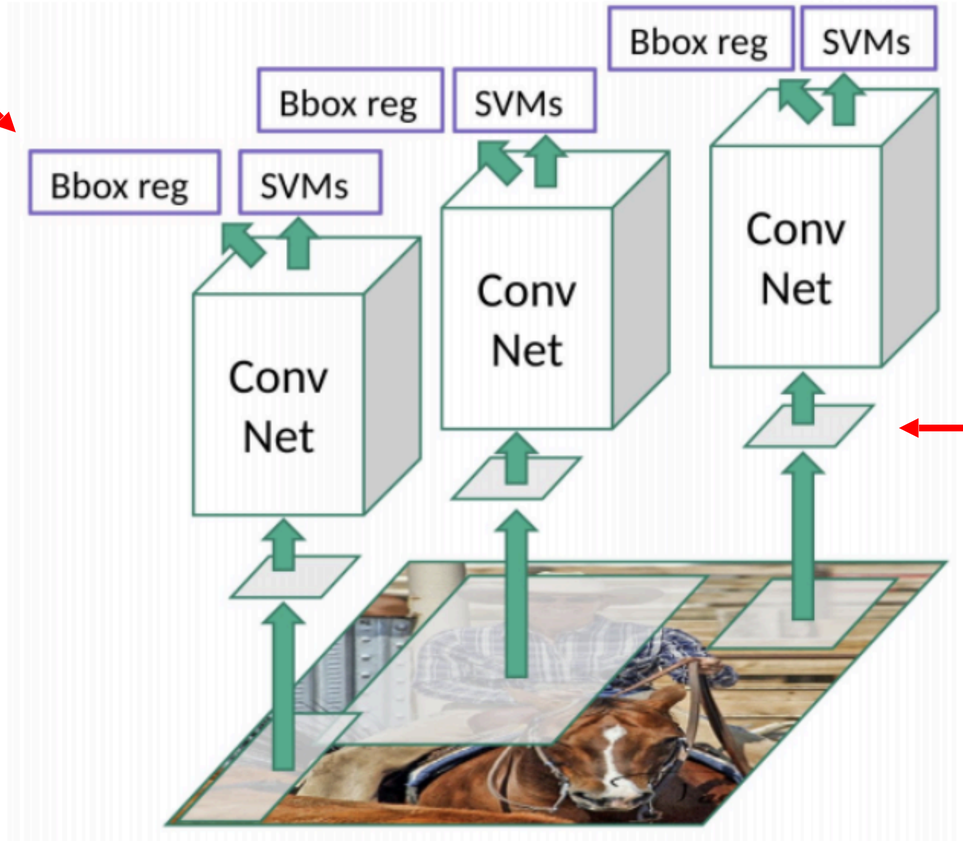
- “Regions with CNN features”
- One of the most impactful lines of work on multi-object detection and segmentation
- … and in CV
  - >120K total citations: R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN



R-CNN

Regression head to refine the bounding box location

Classification head



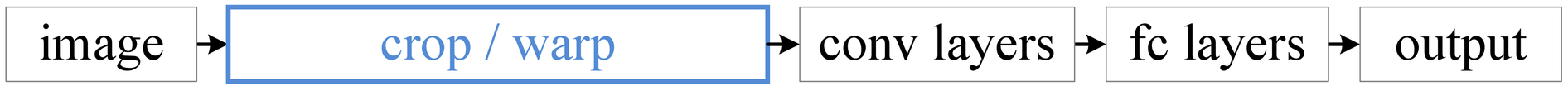
Extract features

Warping to a fix size 227 x 227

SPP-Net: Spatial Pyramid Pooling

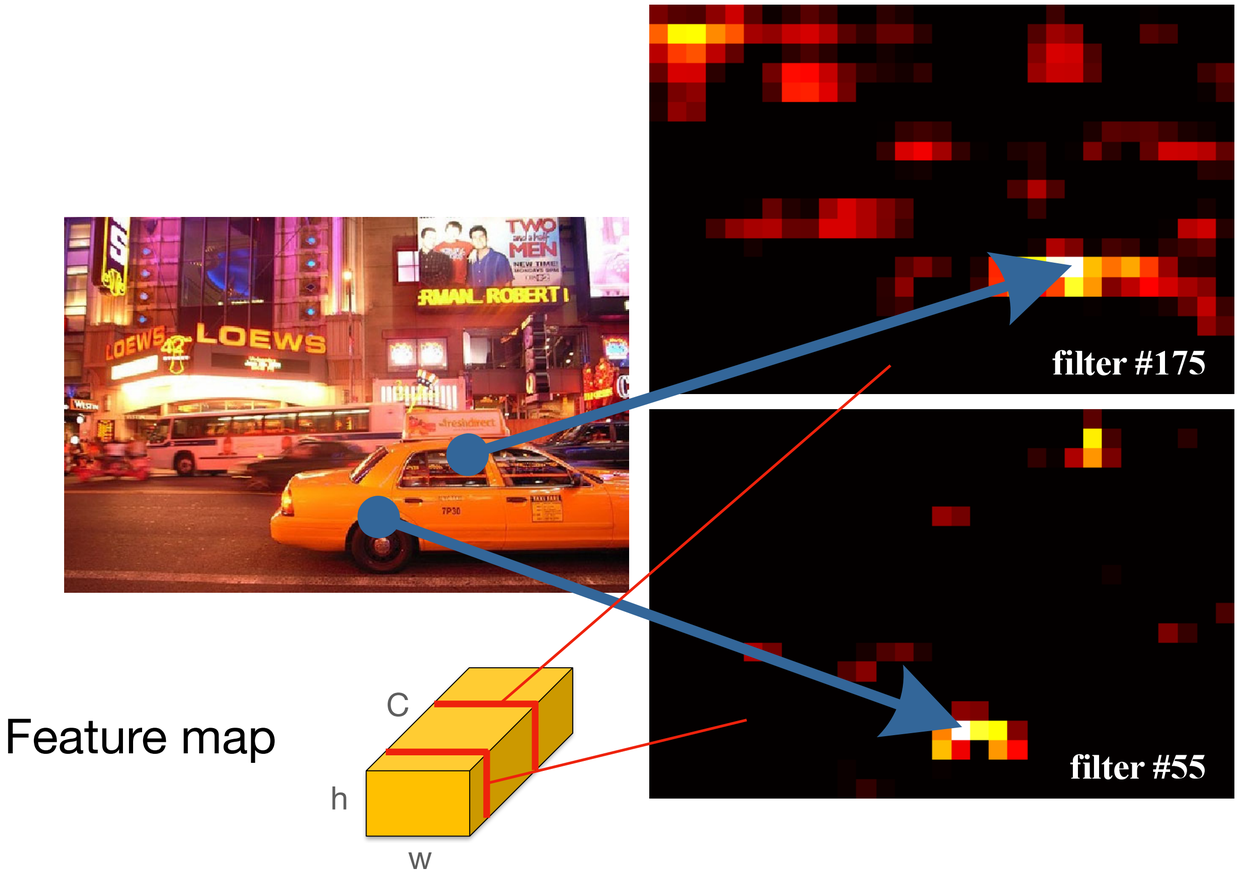
crop

warp



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

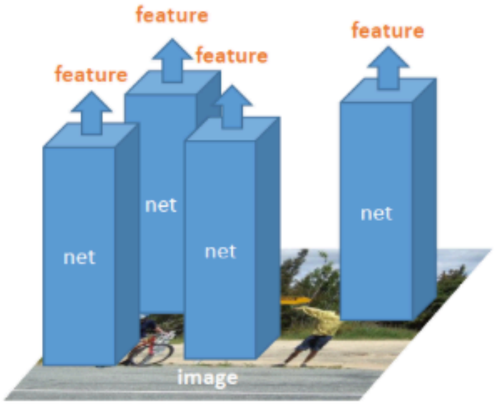
Object detection with deep nets



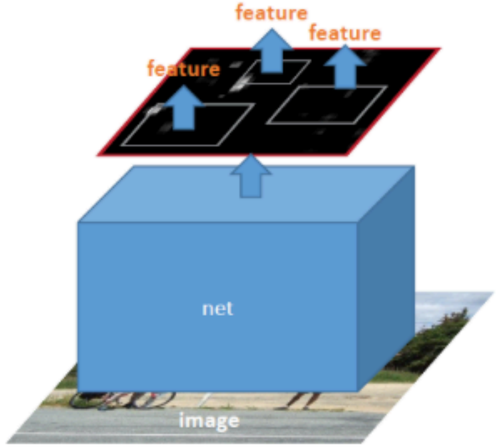
He et al. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. ECCV 2014.

SPP-Net: Overview

How do we “pool” these features into a common size



R-CNN
2000 nets on image regions



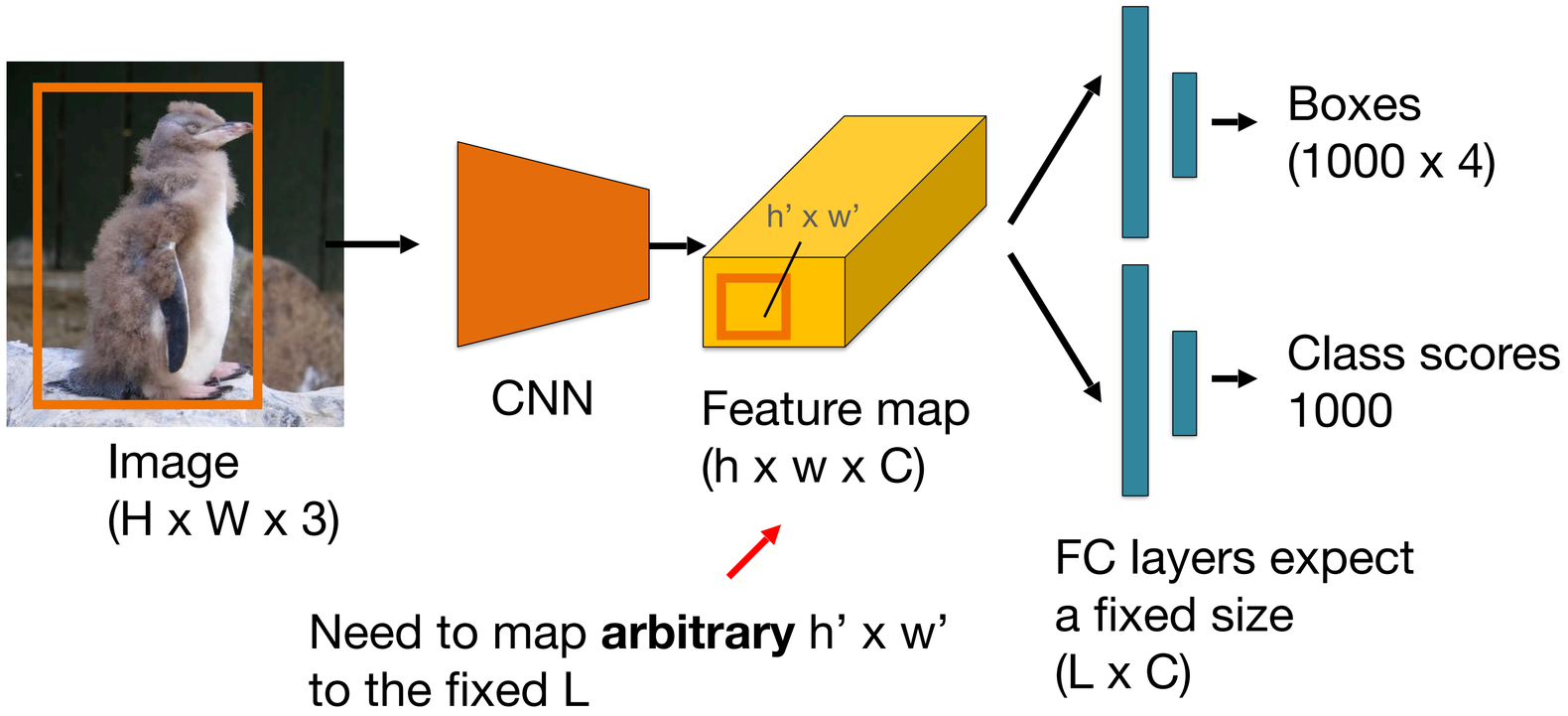
SPP-net
1 net on full image

Frozen

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

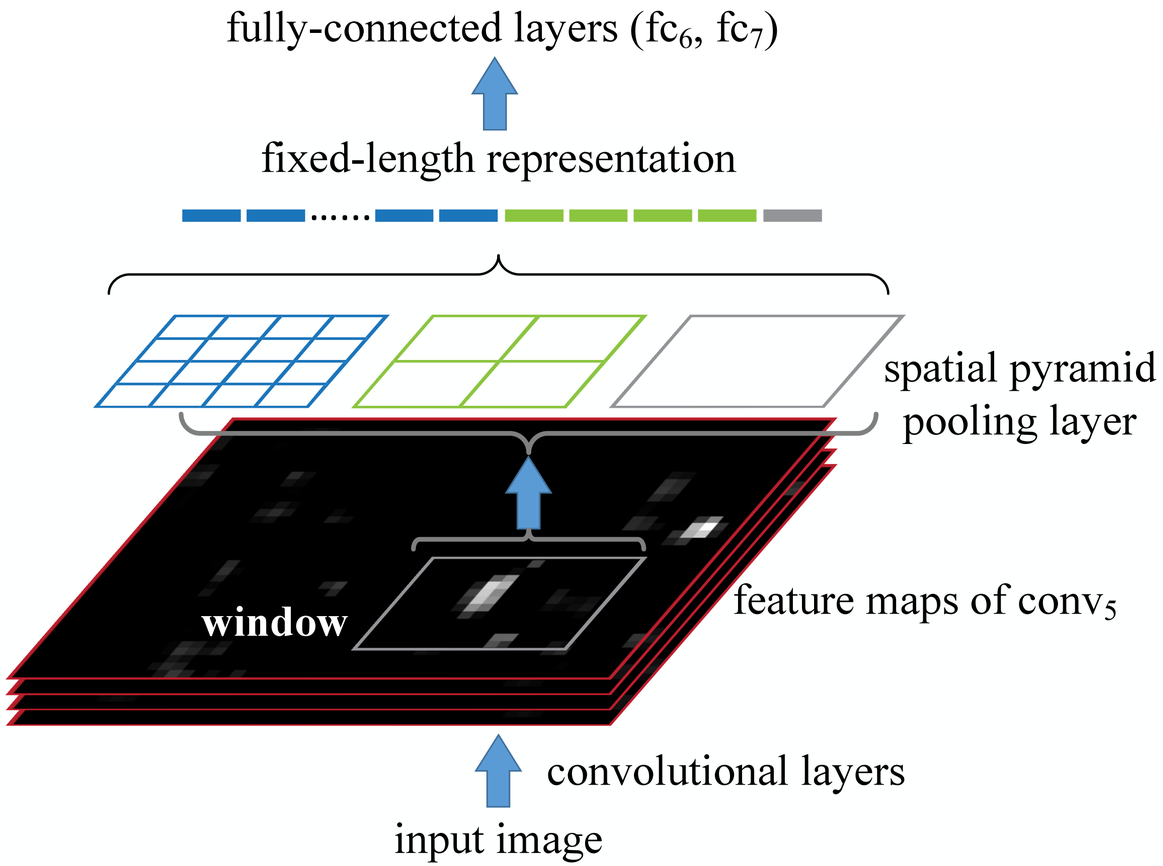
Fast R-CNN: RoI pooling

- Region-of-Interest pooling



Sermanet et al, “Integrated Recognition, Localization and Detection using Convolutional Networks”, ICLR 2014

SPP-Net: Spatial Pyramid Pooling



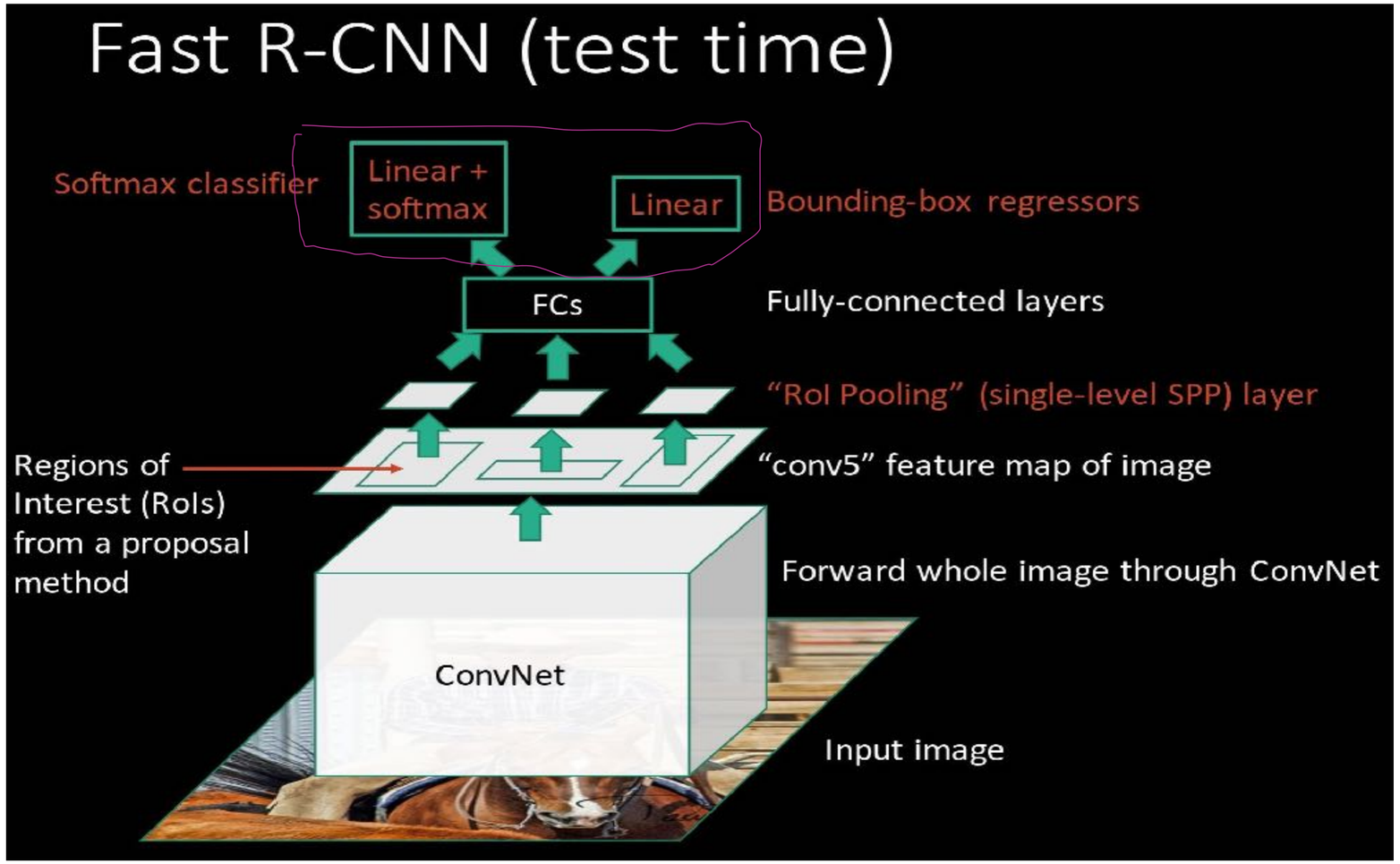
He et al. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. ECCV 2014.

SPP-Net

- Faster training and testing than R-CNN
- Training scheme is still complex
- Still no end-to-end training (fixed convolutional layers)

- Integrate Spatial Pooling into R-CNN → Fast R-CNN

Fast R-CNN

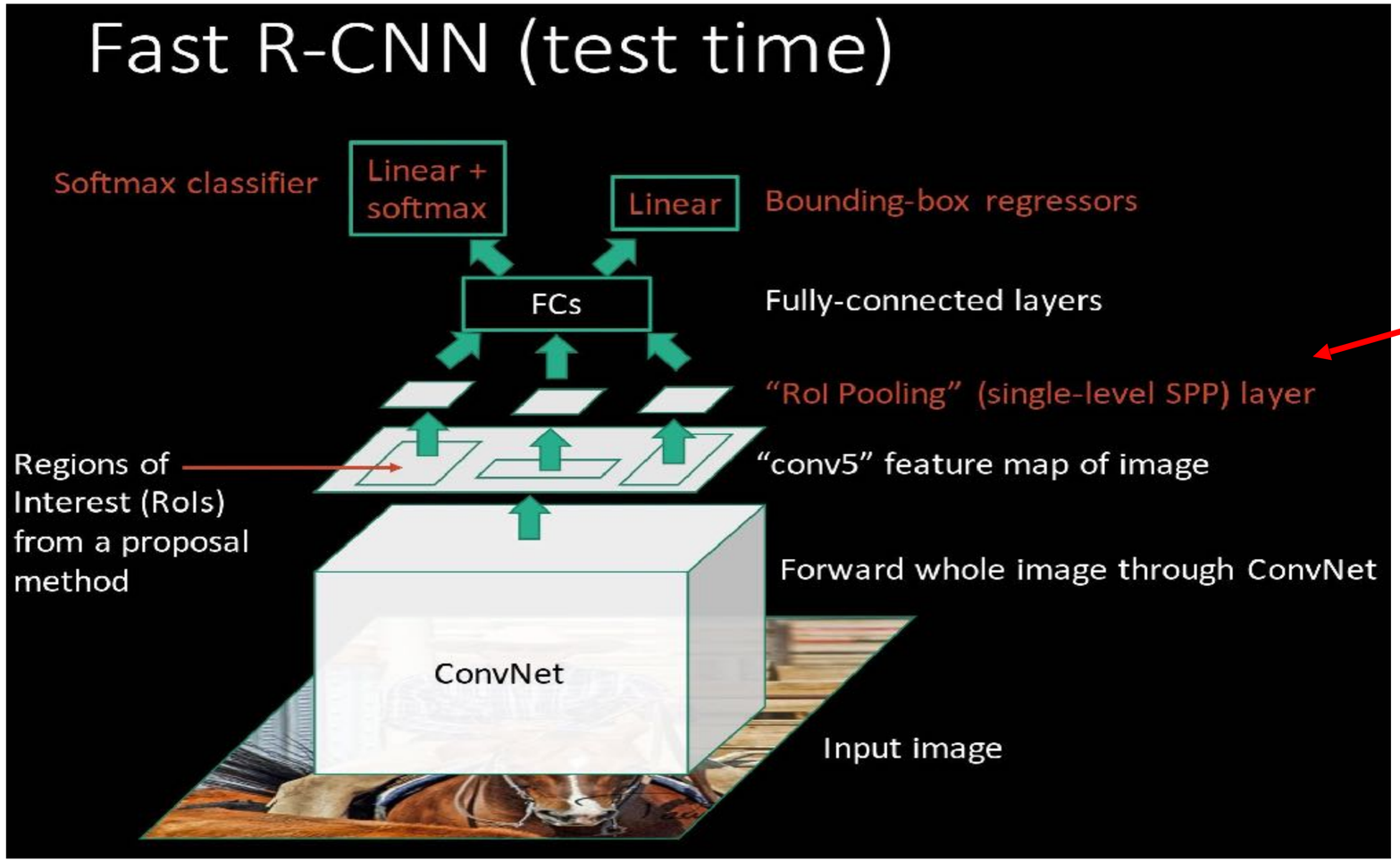


Shared computation at test time (like SPP)

Girschick, "Fast R-CNN", ICCV 2015

Slide credit: Ross Girschick

Fast R-CNN



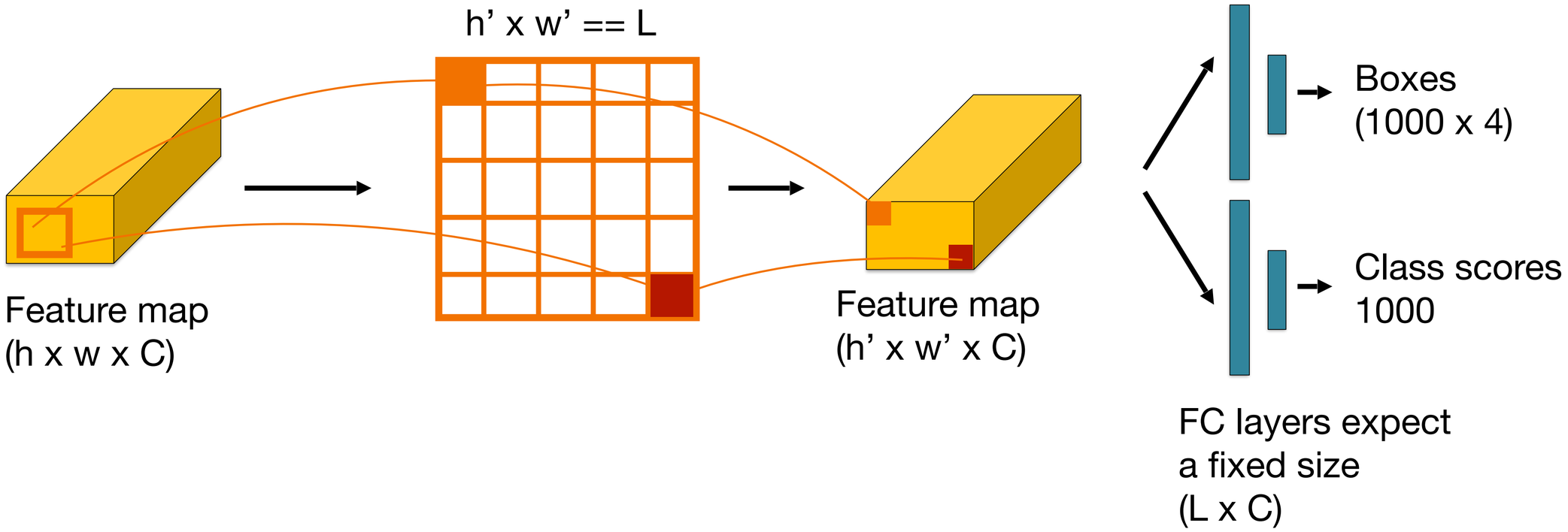
Region-of-Interest pooling

Girschick, "Fast R-CNN", ICCV 2015

Slide credit: Ross Girschick

Fast R-CNN: RoI pooling

- Region-of-Interest pooling



Sermanet et al, “Integrated Recognition, Localization and Detection using Convolutional Networks”, ICLR 2014

Fast R-CNN results

VGG-16 CNN on Pascal VOC 2007 dataset

	R-CNN	Fast R-CNN
Training Time:	84 hours	9.5 hours
Speed-up	1x	8.8x

Fast R-CNN results

VGG-16 CNN on Pascal VOC 2007 dataset

	R-CNN	Fast R-CNN
Training Time:	84 hours	9.5 hours
Speed-up	1x	8.8x
Test time per image	47 seconds	0.32 seconds
Speed-up	1x	146x

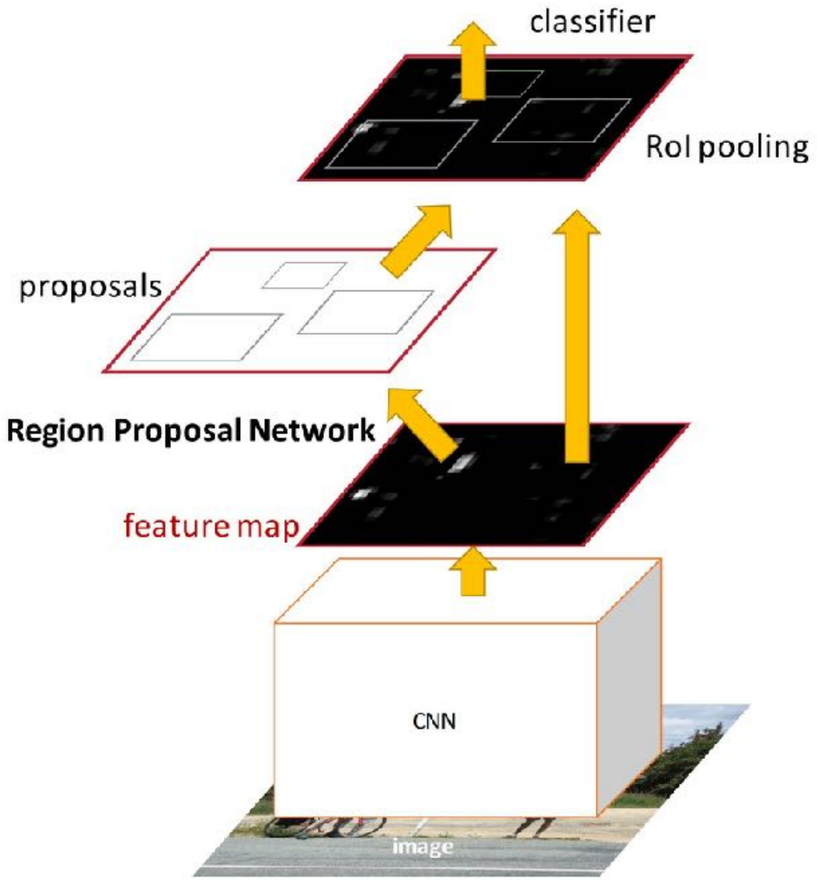
Making Fast R-CNN faster

- We still rely on a standalone object proposals

- Faster R-CNN: Integrated proposal generation with the rest of the pipeline
  - Region Proposal Network (RPN)

- Other than RPN, everything is like Fast R-CNN

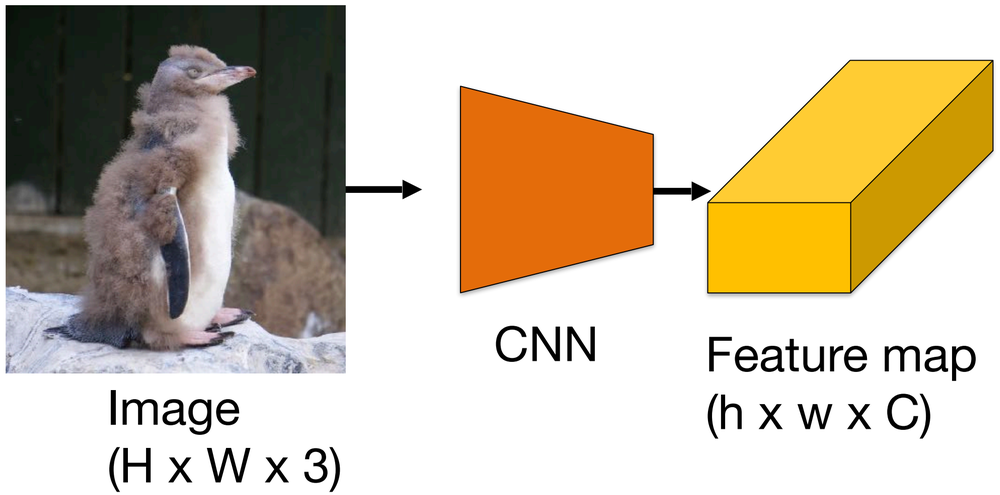
Faster R-CNN



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

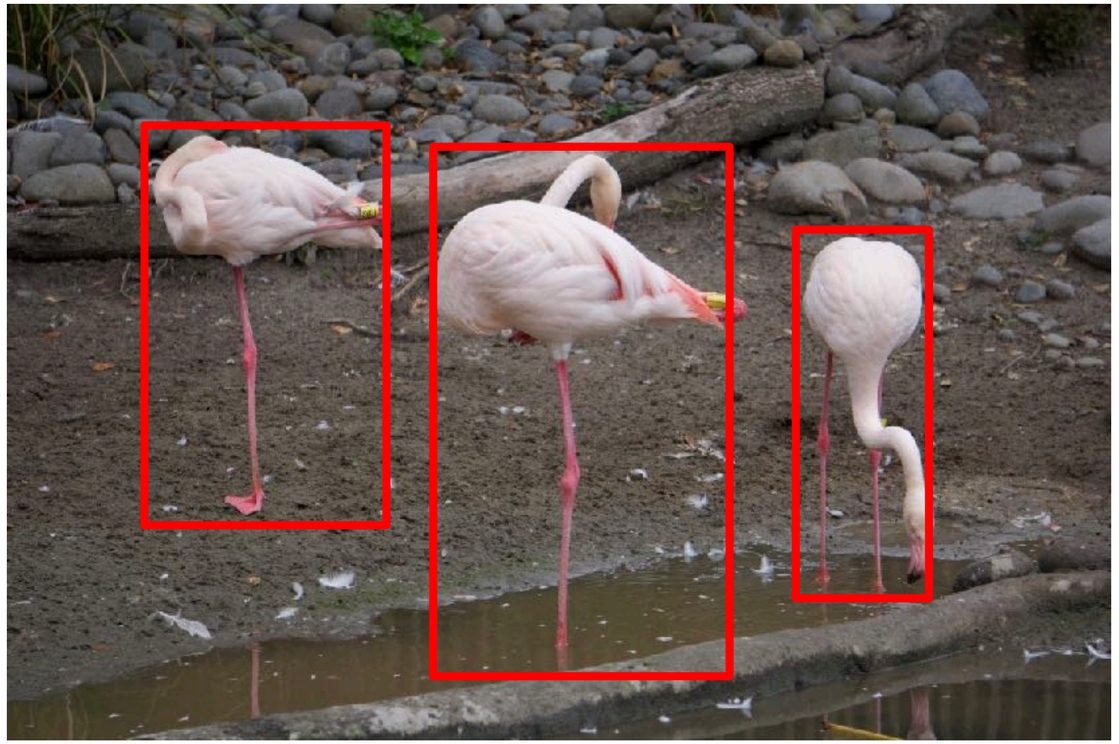
Slide credit: Ross Girschick

Region Proposal Network



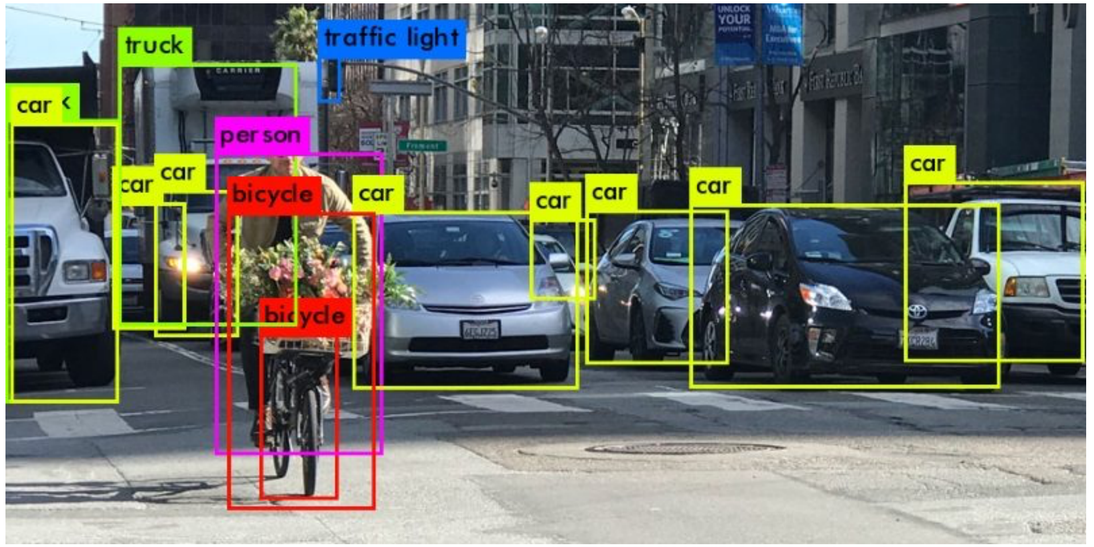
- How many proposals?
- How are they placed?

Multi-object detection



3 objects means having an output of 12 numbers (3 x 4)

Multi-object detection

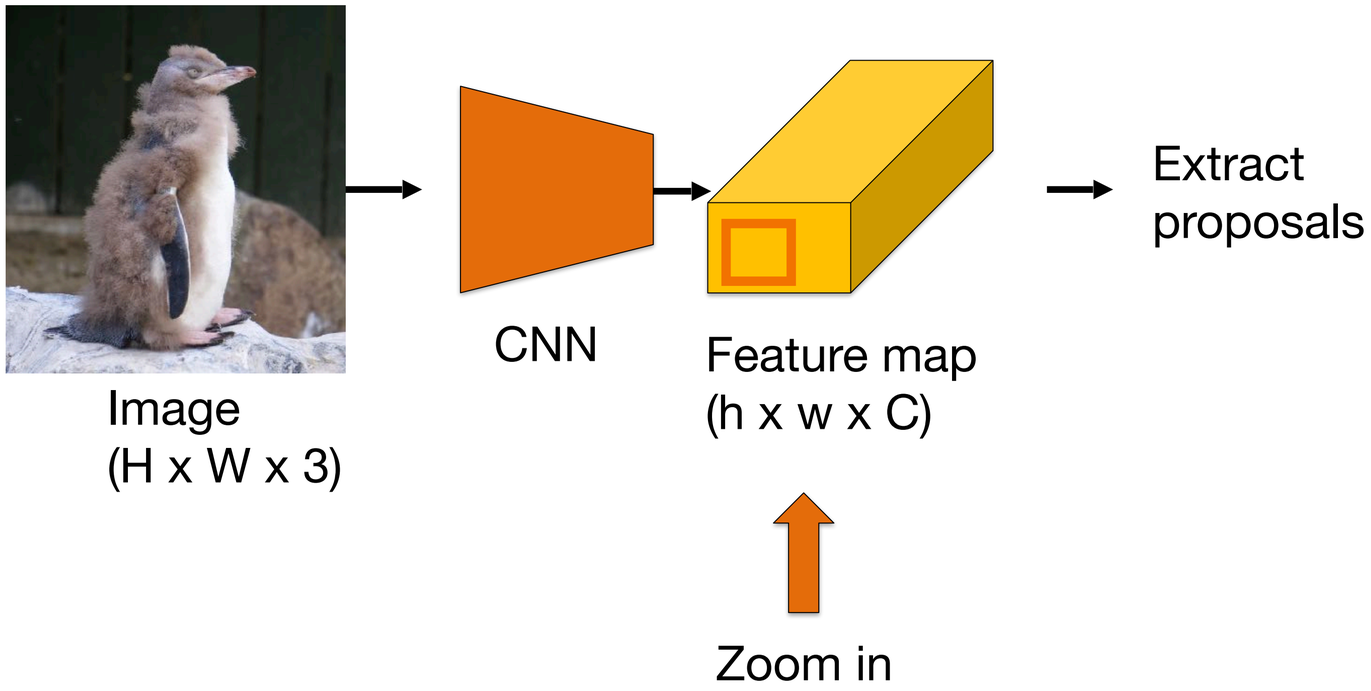


14 objects means having an output of 56 numbers (14 x 4)

Multi-object detection

- Dealing with variable-sized output is challenging
  - There are a couple of workarounds:
    - RNN: (Stewart et al., 2016)
    - Predict the # of objects: (e.g., Rezatofighi, 2018)
- RPN: Place multiple proposals uniformly densely
  - Learn to predict confidence for each proposal

Region Proposal Network

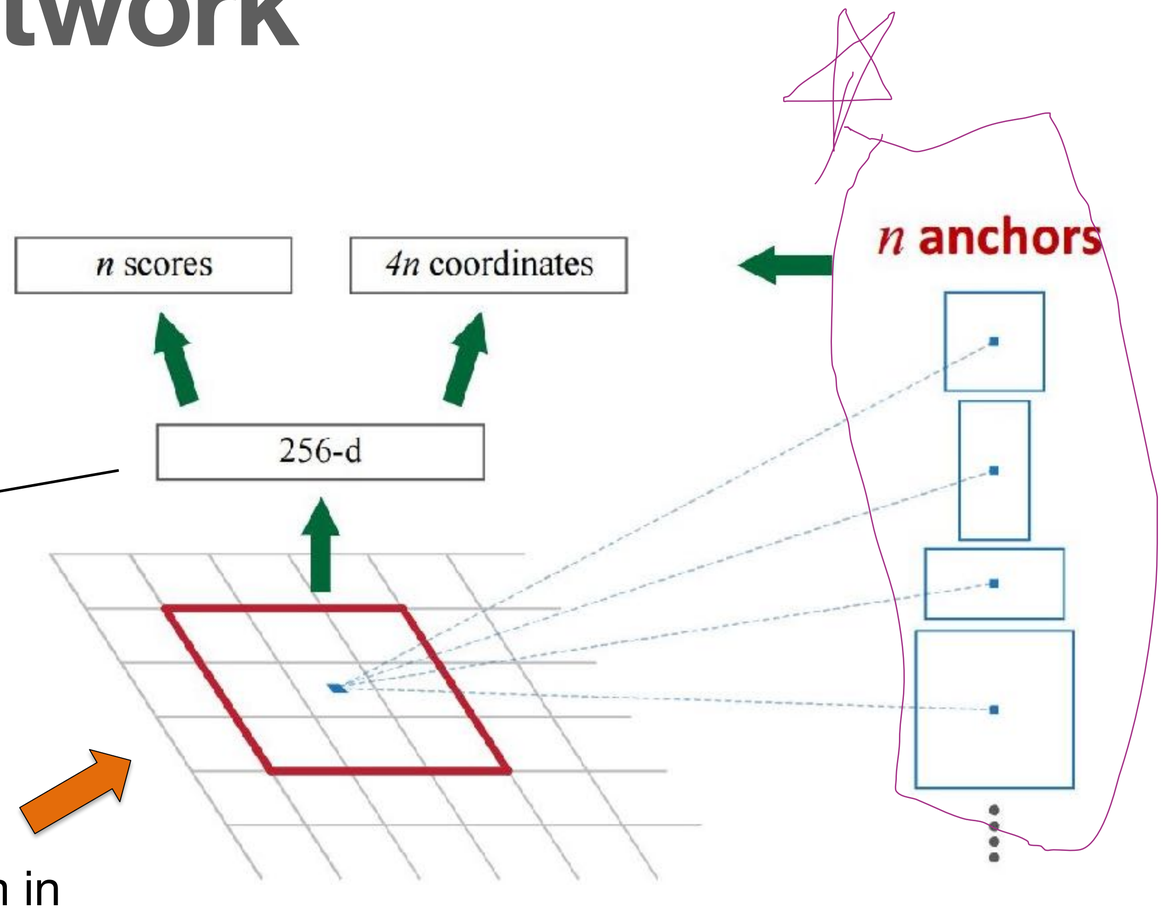


Region Proposal Network

We fix the number of proposals
by using a set of $n = 9$ anchors per location.

9 anchors = 3 scales x 3 aspect ratios

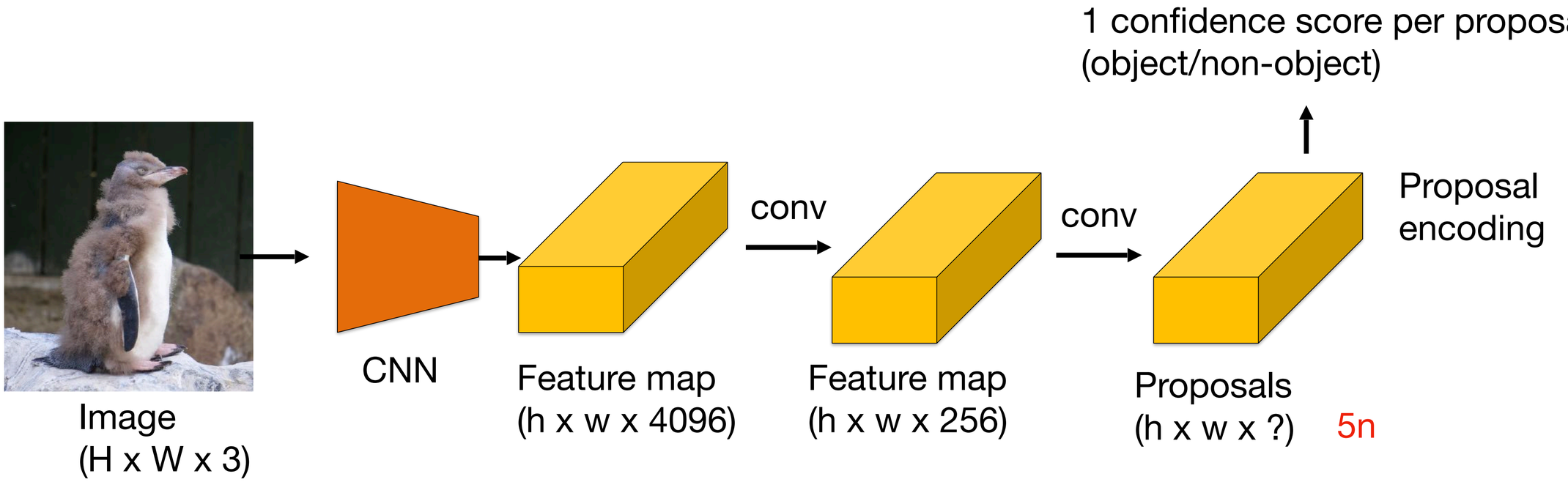
Every location is characterised by
a 256-d descriptor



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

Slide credit: Ross Girschick

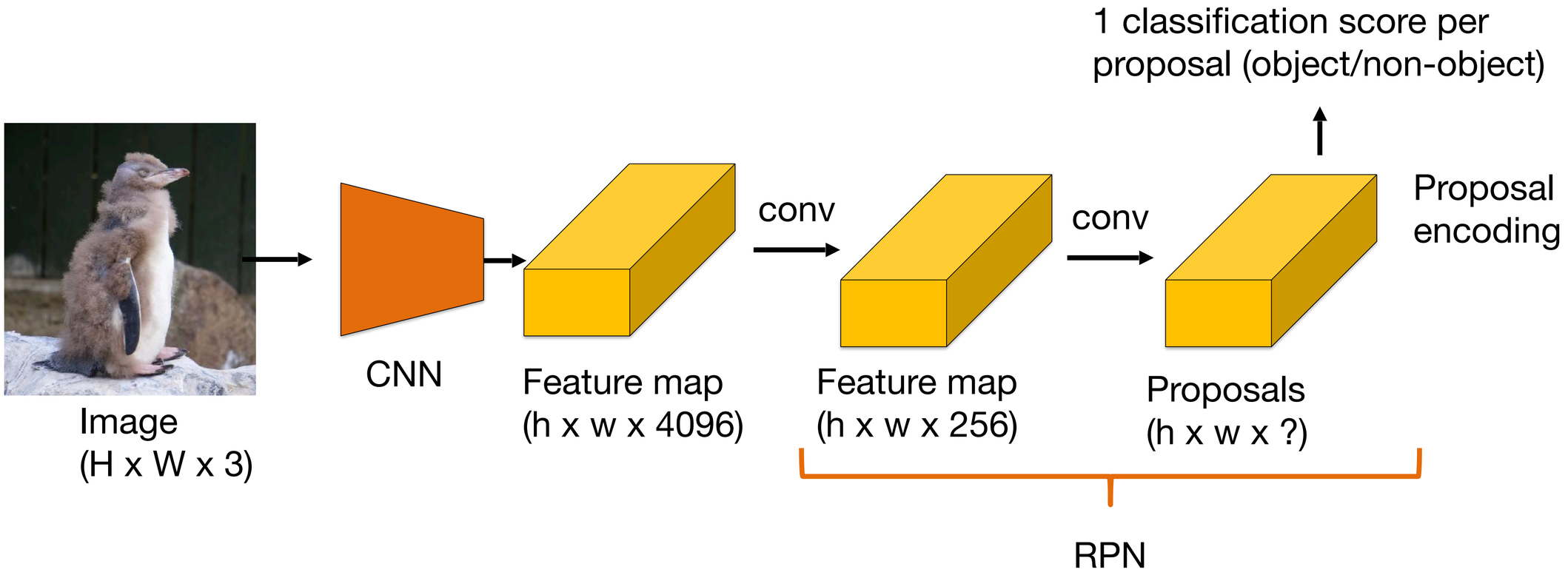
Region Proposal Network



QUIZ: Suppose we have n anchors

We need 4 points in 1 anchor. Also we need a value for if the anchor includes object or not . that is why 4+1 = 5

Region Proposal Network



Per feature map location, we get a set of anchor correction and classification into object/non-object

RPN: Training

- Classification ground truth: We compute $p*$ which indicates how much an anchor overlaps with the ground truth bounding boxes

$$p^* = 1 \quad if \quad \text{IoU} > 0.7$$

$$p^* = 0 \quad if \quad \text{IoU} < 0.3$$

- 1 indicates the anchor represent an object (foreground) and 0 indicates background object. The rest do not contribute to the training.

RPN: Training

- For an image, we randomly sample 256 anchors to form a mini-batch (balanced objects vs. non-objects)
- We learn anchor confidence with the binary cross-entropy loss:

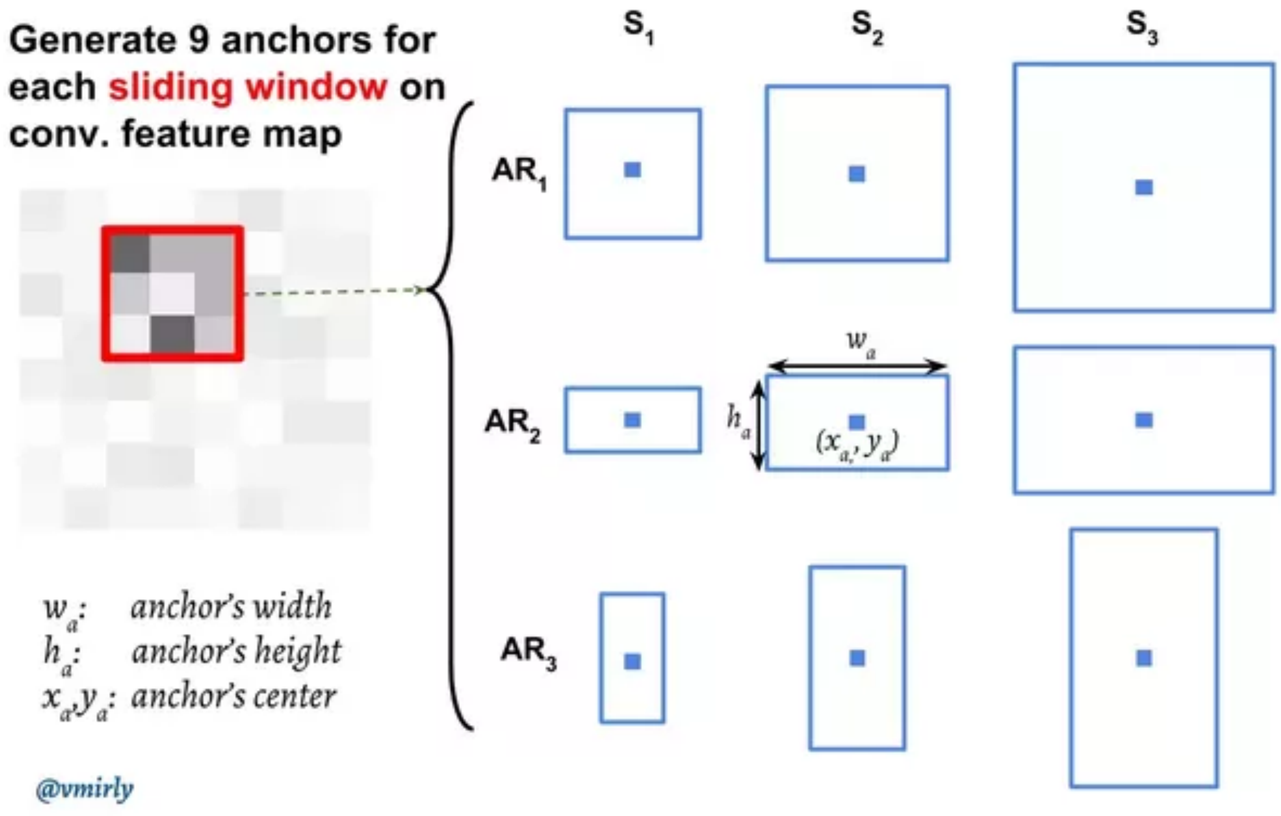
$$\mathcal{L}_{bce} = -c \log p - (1 - c) \log(1 - p)$$

$c \in \{0,1\}$ ground truth; $p \in [0,1]$ model prediction.

- Those anchors that contain an object are used to compute the regression loss

RPN: Training

- Each anchor is described by the center position, width and height



RPN: Training

- Each anchor is described by the center position, width and height
- What the network actually predicts are

$$t_x = (x - x_a)/w_a, \qquad t_y = (y - y_a)/h_a,$$

Normalized horizontal shift — Normalized vertical shift

$$t_w = log(w/w_a), \qquad t_h = log(h/h_a),$$

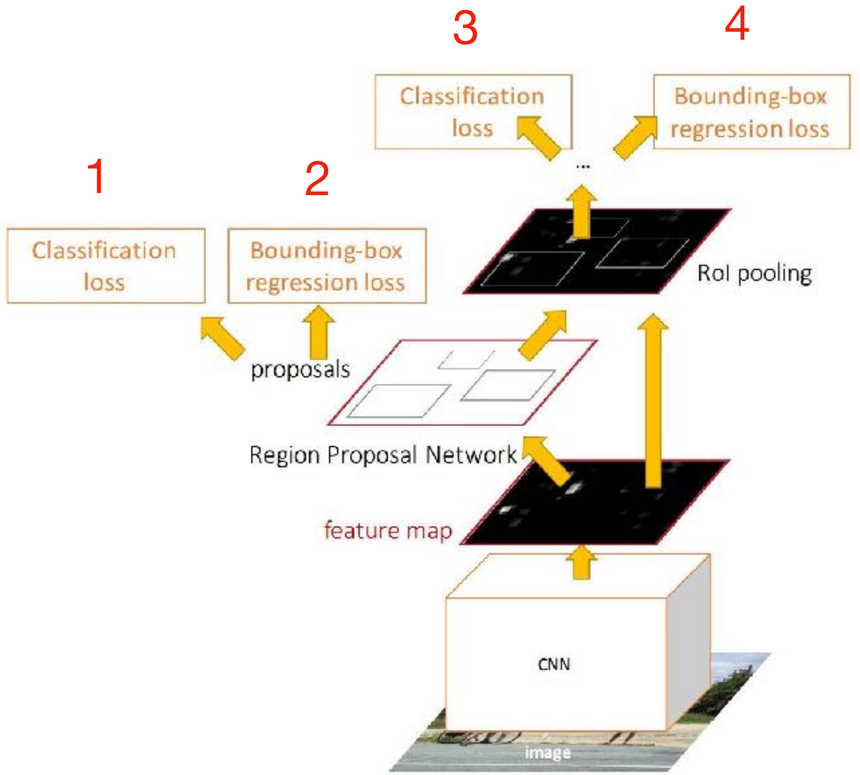
Normalized width — Normalized height

- Smooth L1 loss on regression targets

Faster R-CNN: Training

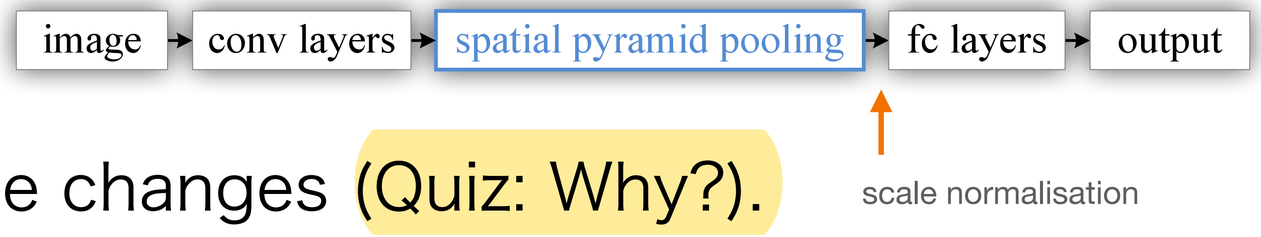
- First implementation, training of RPN separate from the rest.
- Now we can train jointly!

- Four losses:
  1. RPN classification (object/non-object)
  2. RPN regression (anchor → proposal)
  3. Fast R-CNN classification (type of object)
  4. Fast R-CNN regression (proposal → box)



Faster R-CNN vs Fast R-CNN

- 10x faster at test time w.r.t. Fast R-CNN
- Trained end-to-end including feature extraction, region proposals, classifier and regressor
- RPN is fully convolutional
- More accurate:
  - the proposals are learned;
  - classification is robust to scale changes (Quiz: Why?).



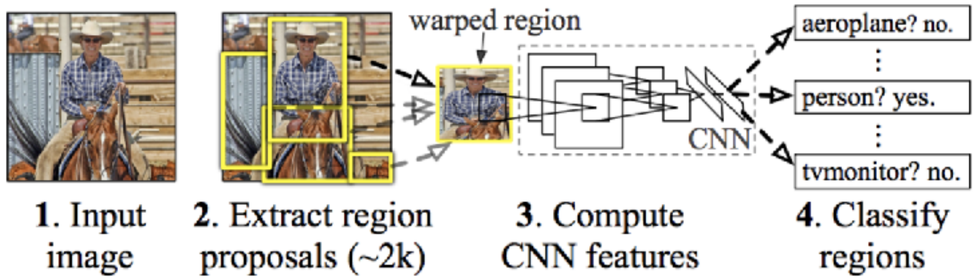
he said that we can define arbitary sizes for object proposals

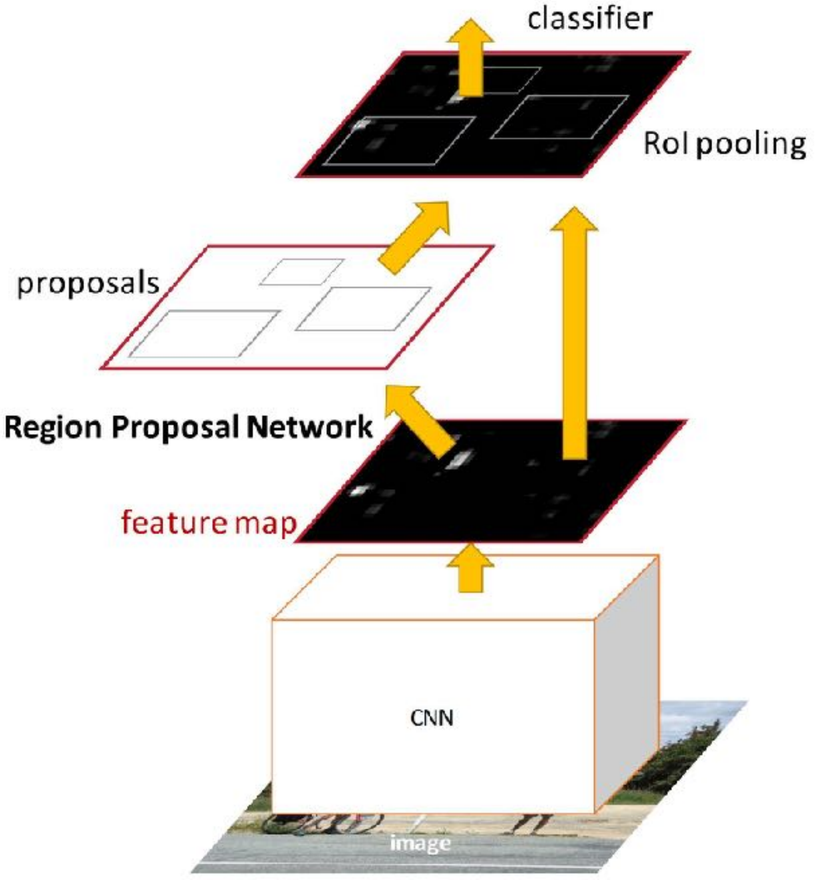
Faster R-CNN: Results

VGG-16 CNN on Pascal VOC 2007 dataset

	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image (w/ proposals, s)	50	2	0.2
Speed-up	1x	8.8x	250x
mAP	66.0	66.9	69.9

From R-CNN to Faster R-CNN





- More efficient and more accurate.
- Incremental but powerful improvements.
- Ent-to-end training: more scalable, i.e. larger benefits from more data

Addressing scale variance

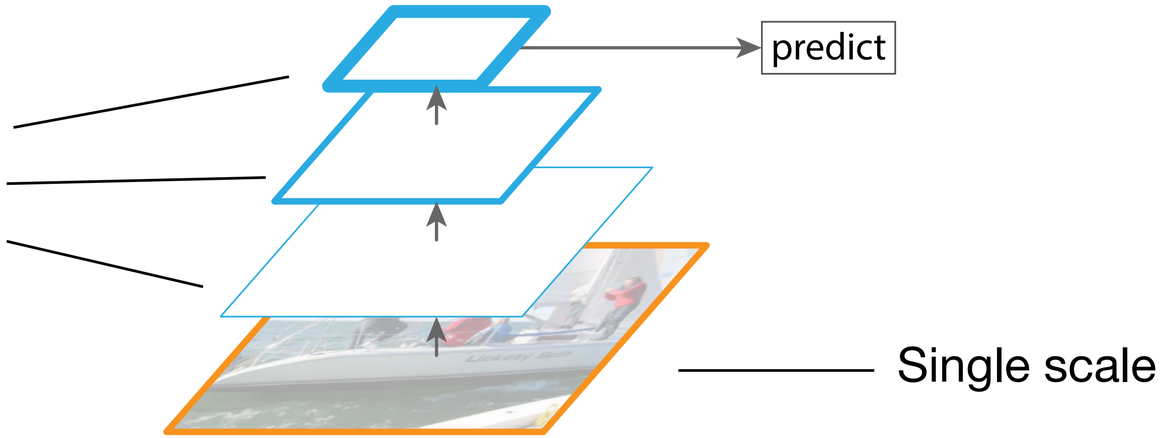
- CNNs are not scale-invariant
- Problem: Object scale can vary drastically

Credit: MS-COCO

Addressing scale variance

- CNNs are not scale-invariant
- Our approach so far:

Features are progressively downsampled in CNNs

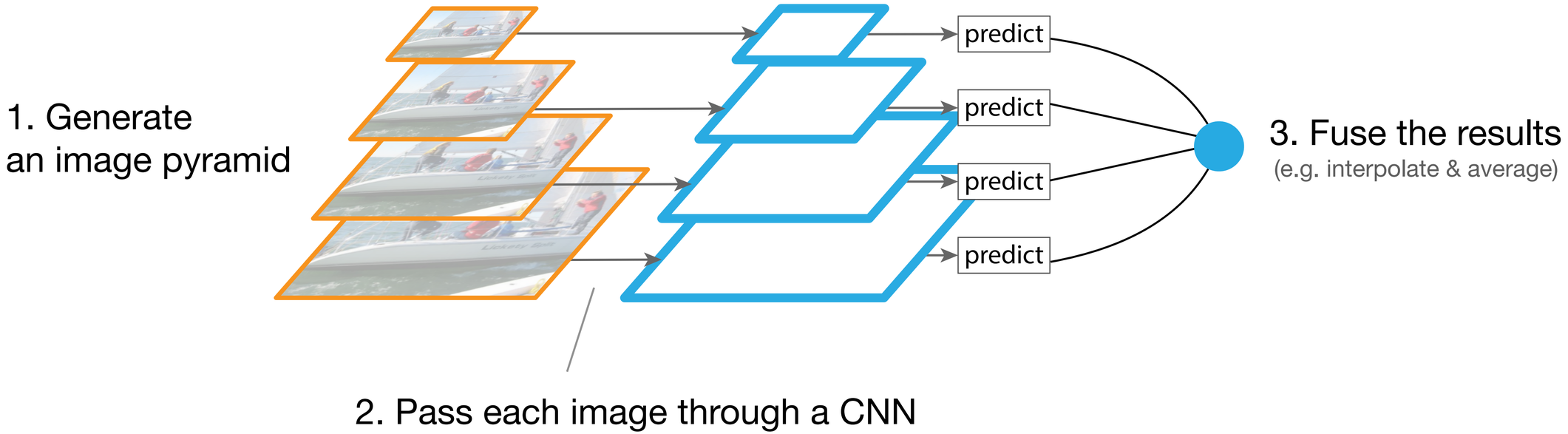


➡ Make CNNs for object detection more robust to scale changes

Lin et al., “Feature Pyramid Networks for Object Detection”. CVPR 2017

Addressing scale variance

- Idea A: featurised image hierarchy

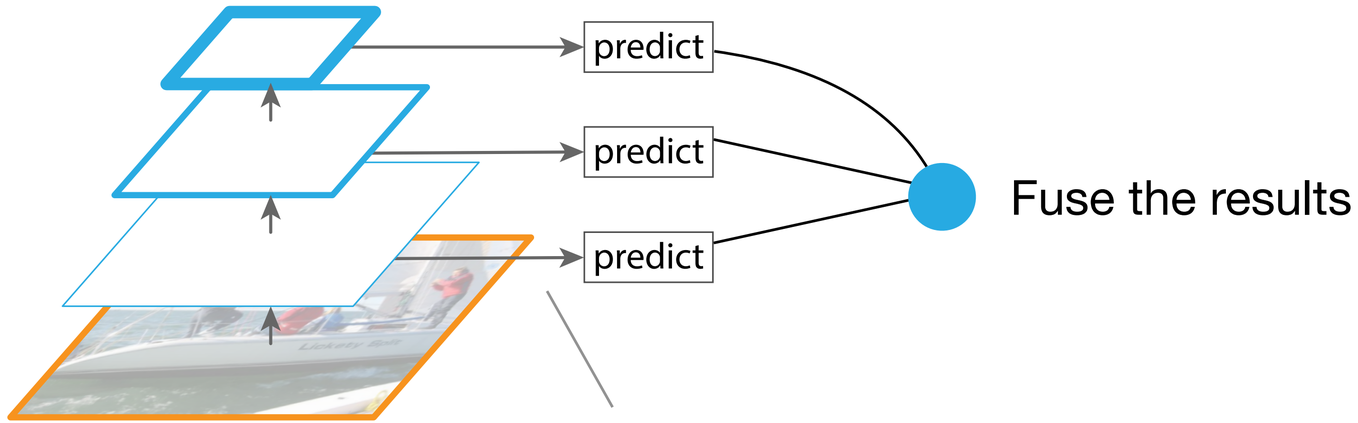


- Pros: Typically boosts accuracy (esp. at test time)
- Cons: Computationally inefficient

Lin et al., “Feature Pyramid Networks for Object Detection”. CVPR 2017

Addressing scale variance

- Idea B: pyramidal feature hierarchy

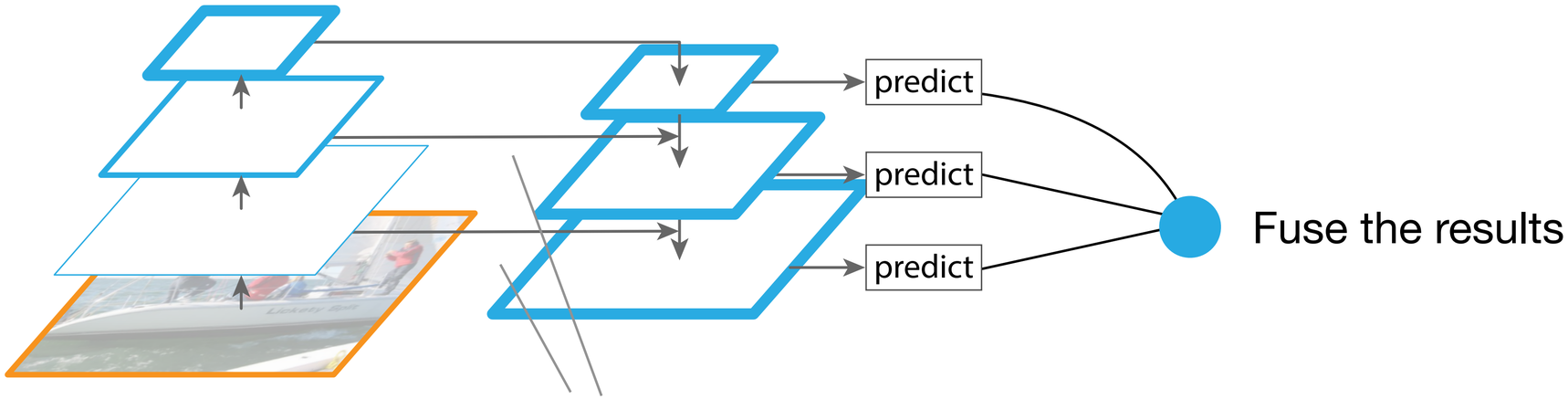


- More efficient than Idea A, but
- limited accuracy (inhibits the learning of deep representations)

Lin et al., “Feature Pyramid Networks for Object Detection”. CVPR 2017

Addressing scale variance

- Idea D: feature pyramid network

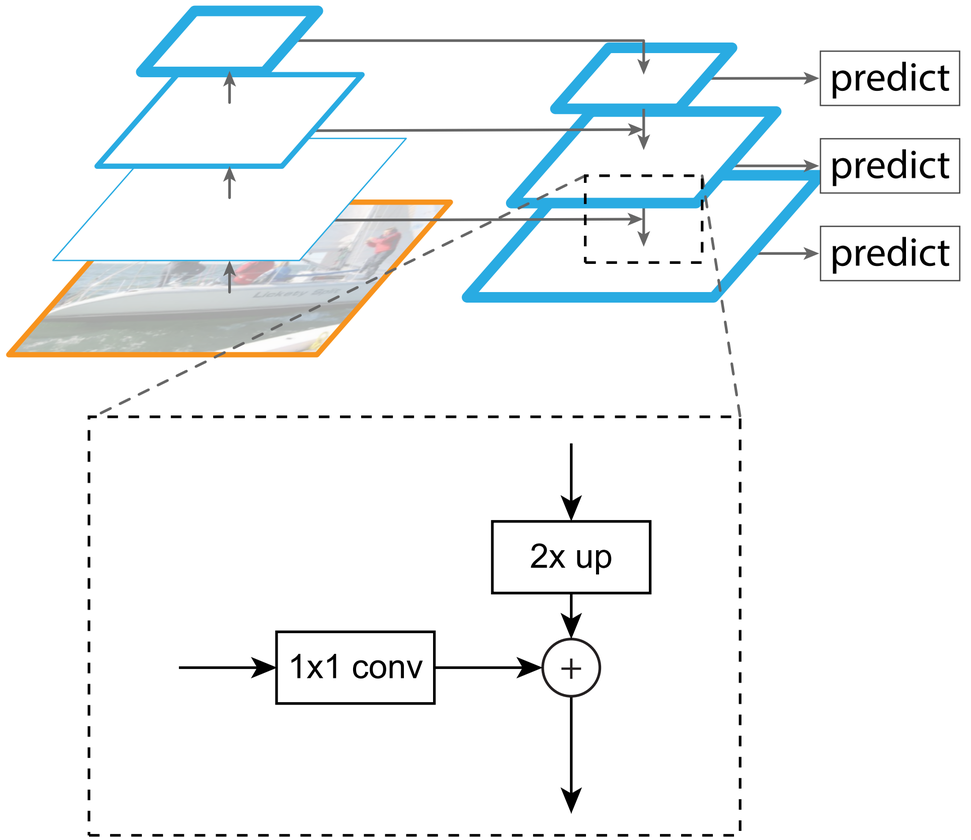


- Higher scales benefit from deeper representation from lower scales
- Efficient and high accuracy

Lin et al., “Feature Pyramid Networks for Object Detection”. CVPR 2017

Feature Pyramid Network (FPN)

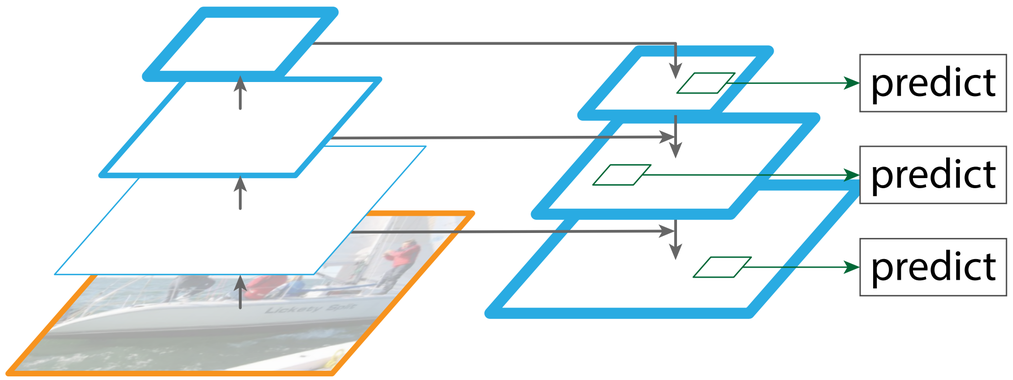
- Straightforward implementation:
  - convolution with 1x1 kernel
  - upsampling (nearest neighbours)
  - element-wise addition



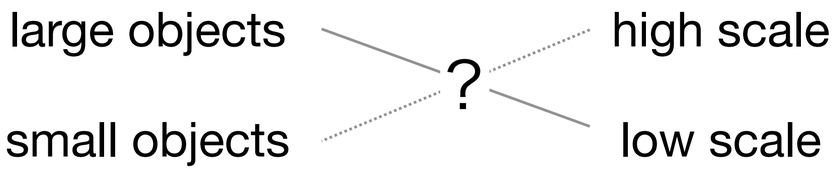
Lin et al., “Feature Pyramid Networks for Object Detection”. CVPR 2017

Feature Pyramid Network (FPN)

- Integrate with RPN for object detection:
  - define RPN on each level of the pyramid;
  - assign ground truth to the pyramid levels;
  - at test time, merge the predictions from all levels.



Quiz: How to assign ground truth to the levels?



Lin et al., “Feature Pyramid Networks for Object Detection”. CVPR 2017

Feature Pyramid Network (FPN)

- Pros:
  - improves recall across all scales (esp. small objects);
  - more accurate (in terms of AP);
  - broadly applicable, also for one-stage detectors (next);
  - still in wide use today.
- Cons:
  - increased model complexity.

Computer Vision III:

One-stage object detectors

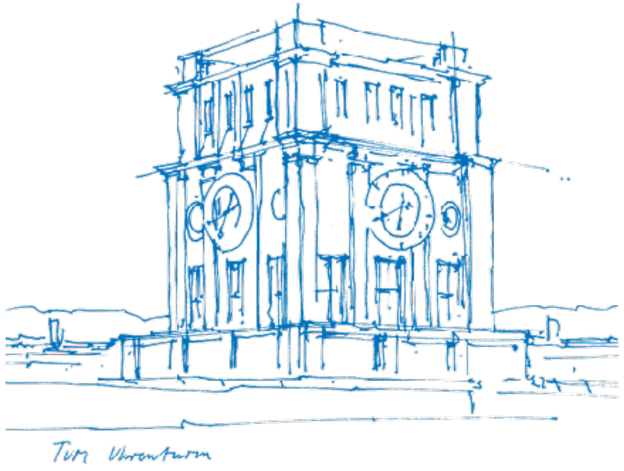
Nikita Araslanov

15.11.2022

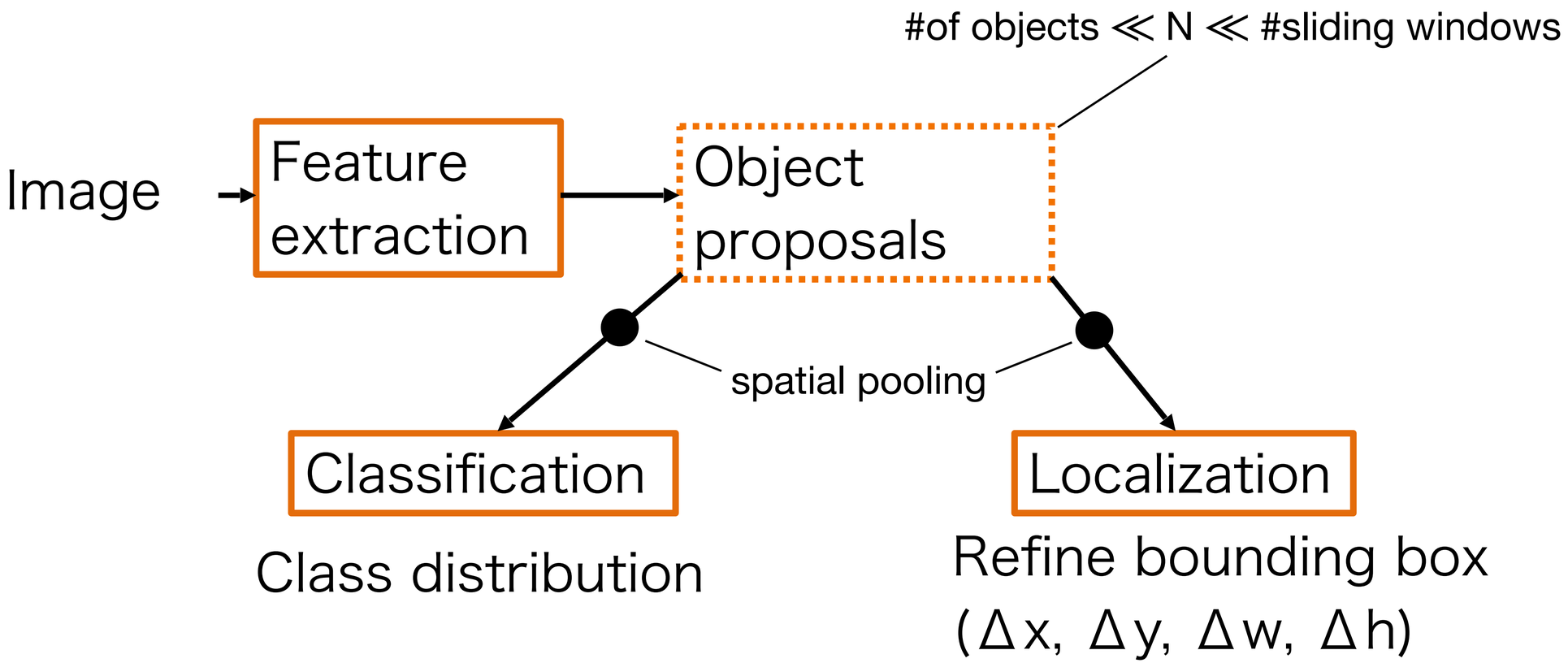
Content credit:

Prof. Laura Leal-Taixé

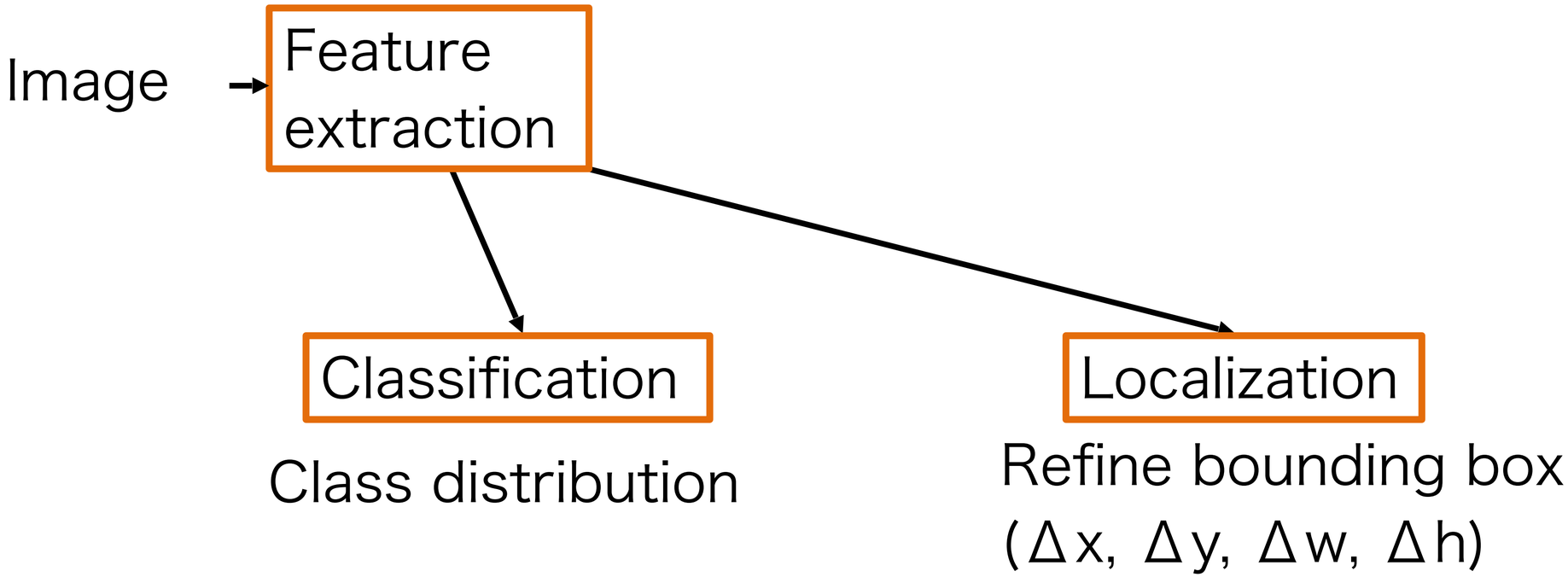
https://dvl.in.tum.de



Two-stage detectors



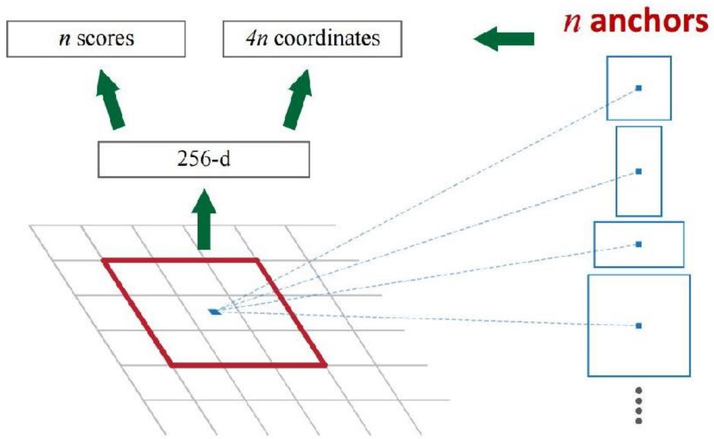
One-stage detectors



Simplifying two-stage detectors

Recall RPN:

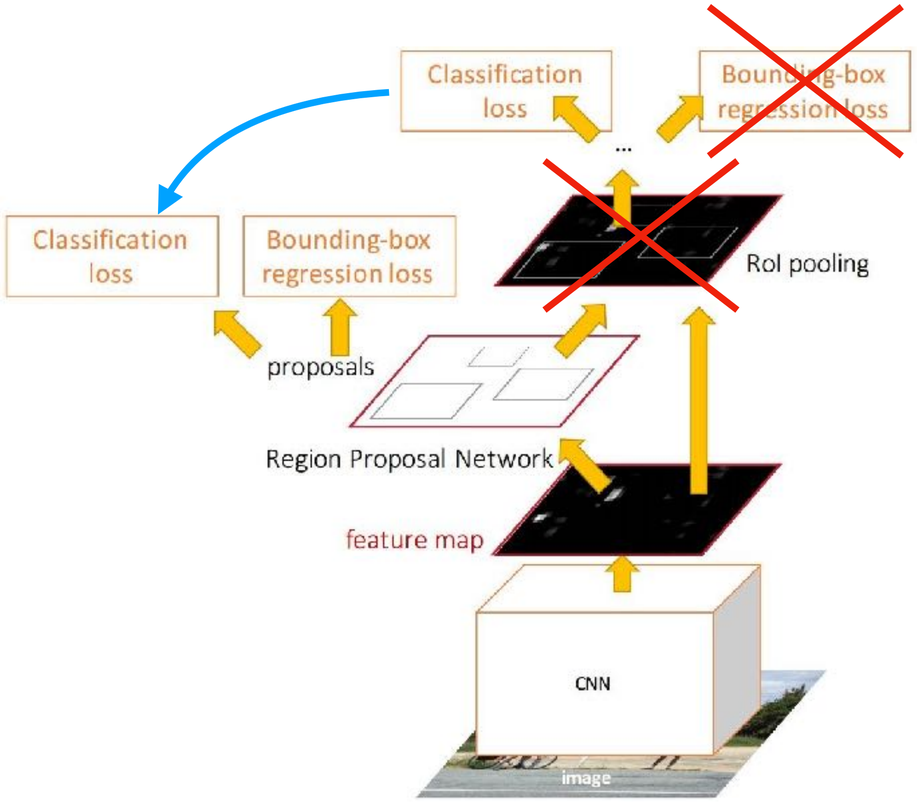
- For each feature cell we define k anchors;
- We refine each anchor with regression;
- We use RoI pooling for classification;

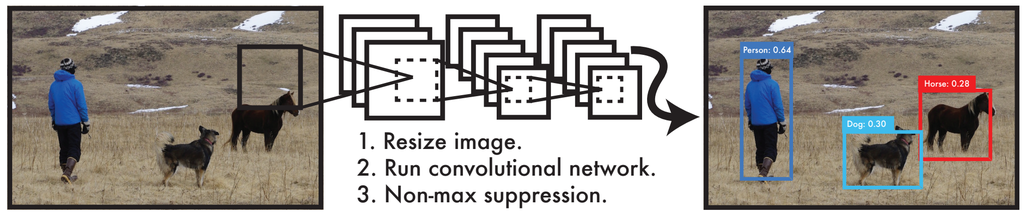


Can we simplify the process to make runtime faster?

One-stage detector: YOLO

What if we remove pooling?

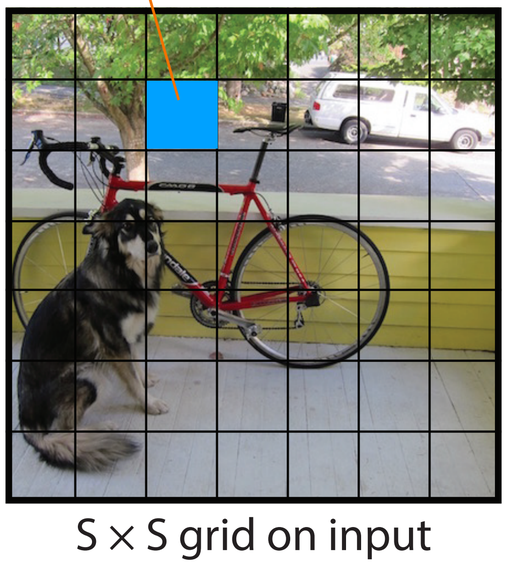




Redmon et al, “You only look once: Unified real-time object detection”, CVPR 2016.

YOLO: You Only Look Once

- Define a coarse grid ($S \times S$);
- Associate B anchors to each cell;
- Each anchor is defined by
  - localisation (x, y, w, h);
  - a confidence value (object / no object);
  - and a class distribution over C classes.

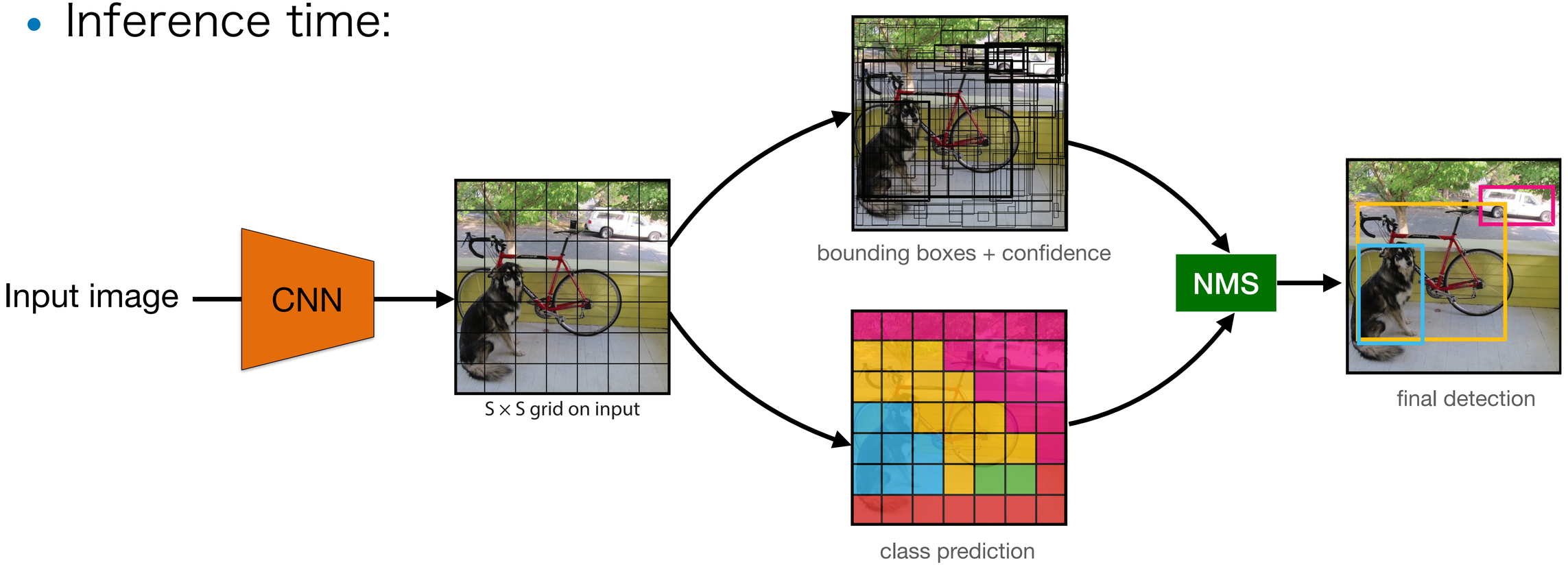


Quiz: What is the dimensionality of the output?

Redmon et al, “You only look once: Unified real-time object detection”, CVPR 2016.

YOLO: You Only Look Once

- Inference time:



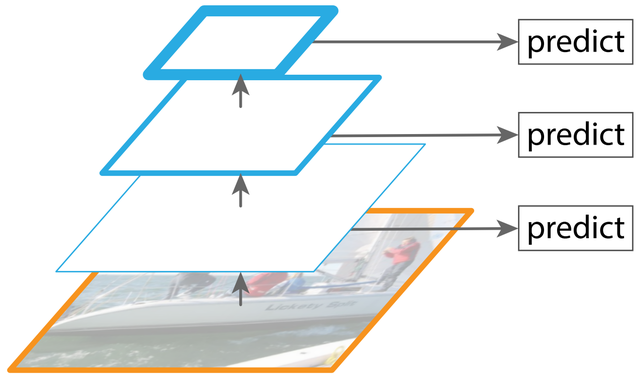
Redmon et al, “You only look once: Unified real-time object detection”, CVPR 2016.

YOLO: You Only Look Once

we sacrificied pooling,

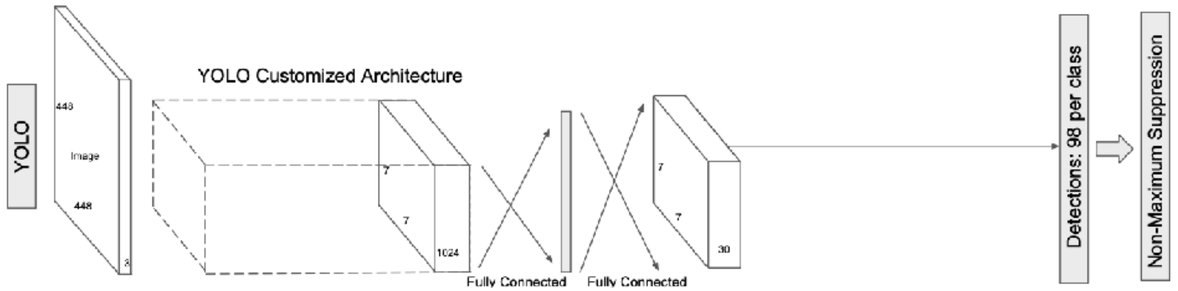
- More efficient (than Faster R-CNN), but less accurate. Why?
  - coarse grid resolution, few anchors per cell – issues with small objects;
  - less robust to scale variation (no spatial pooling).

- How can we improve?
  - Idea 1: recall feature pyramids



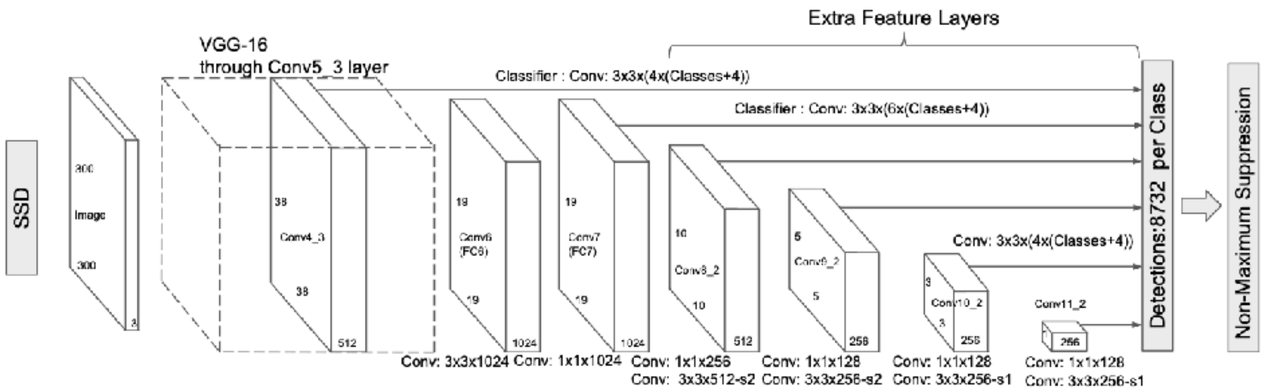
SSD: Single Shot multibox Detector

- YOLO predicts bounding boxes from a single representation



- SSD uses multiple feature scales:

Only a few extra layers



Liu et al. “SSD: Single shot multibox detector”. ECCV 2016

SSD

- Pros:
  - more accurate than YOLO;
  - works well even with lower resolution → improved inference speed.

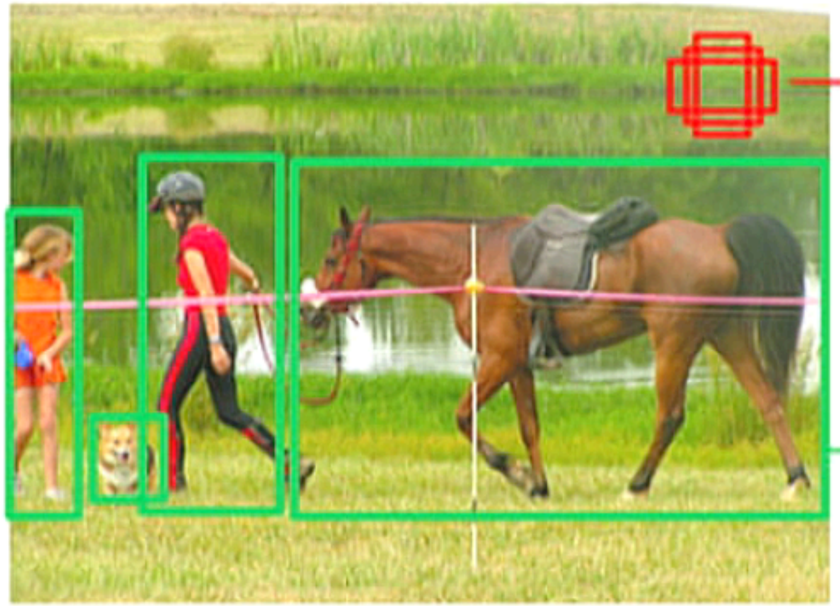
- Cons:
  - still lags behind two-stage detectors;
    - data augmentation is still crucial (esp. random scaling);
  - a bit more complex (due to multi-scale features).

Problem with one-stage detectors?

- Two-stage detectors:
  - Classification only works on “interesting” foreground regions (proposals, ~1-2k). Most background examples are already filtered out.
  - Class balance between foreground and background objects is manageable.
  - Classifier can concentrate on analyzing proposals with rich information content

Problem with one-stage detectors?

- Many locations need to be analyzed (100k) densely covering the image – foreground-background imbalance



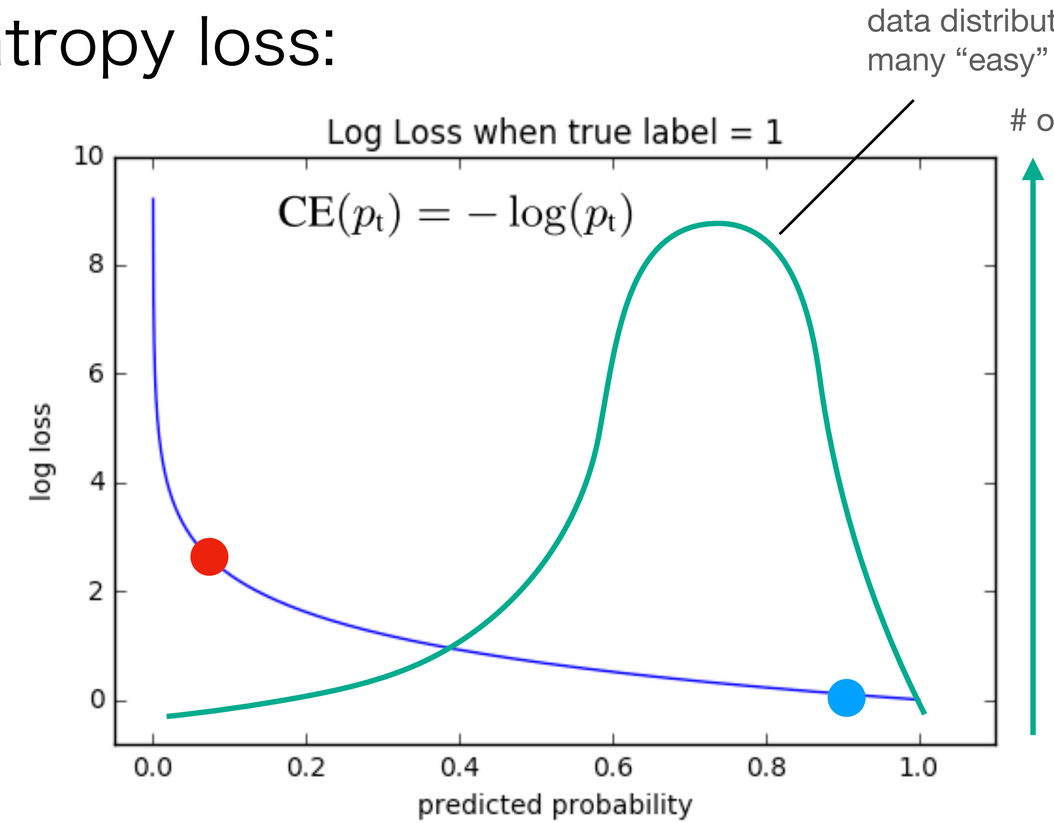
Many negative examples
(every cell on the feature grid)

Few positive examples
(~ actual number of objects)

- Hard negative mining: subsample the negatives with the largest error
  - useful, but not sufficient

Class imbalance

- Idea: balance the positives/negatives in the loss function.
- Recall cross-entropy loss:



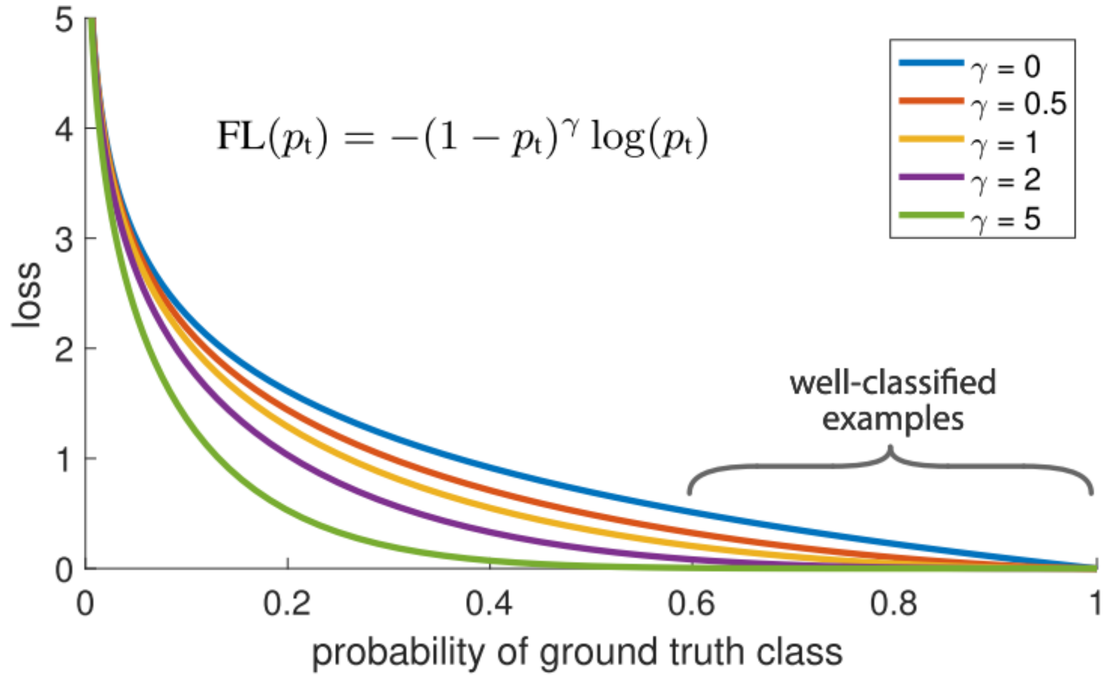
- Very hard example, e.g. $p_t \approx 0.1$, we get loss ≈ 2.3
- An easy example, e.g. $p_t \approx 0.9$, we get loss ≈ 0.1

Back-of-the-envelope calculation:
100 hard examples x 2.3 = 230
10^6 easy examples x 0.1 = 10^5

TY Lin et al. “Focal Loss for Dense Object Detection ”. ICCV 2017

Focal loss

- Replace CE with focal loss (FL):



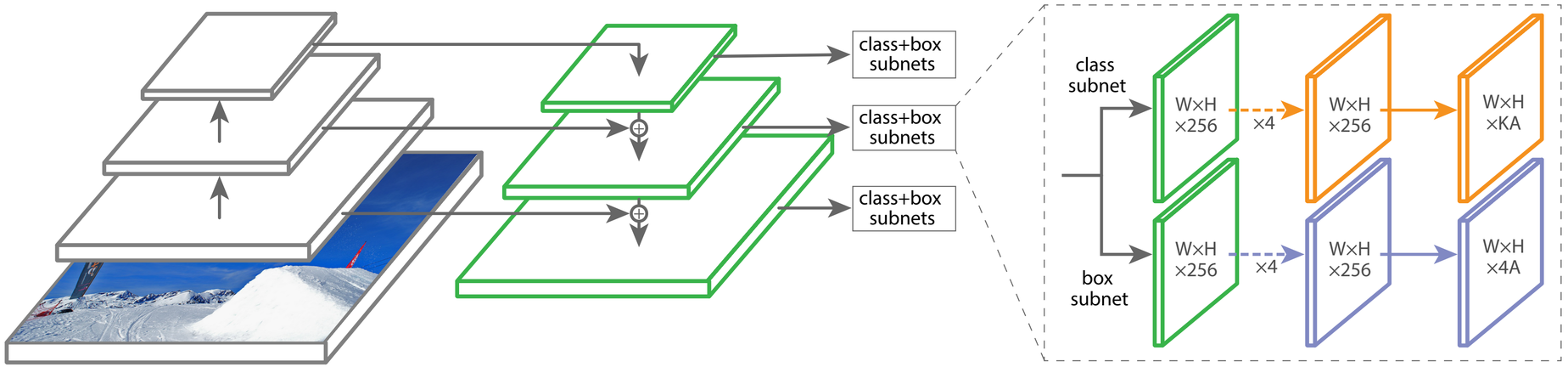
- When $\gamma = 0$, it is equivalent to the cross-entropy loss.
- As γ goes towards 1, the easy examples are down-weighted.
- Example: $\gamma = 2$, if $p_t = 0.9$, FL is $\times 100$ lower than CE.

Back-of-the-envelope calculation:
100 hard examples x 2.3 x 0.9^2 = 186.3
10^6 easy examples x 0.1 x 0.1^2 = 1000

TY Lin et al. “Focal Loss for Dense Object Detection ”. ICCV 2017

RetinaNet

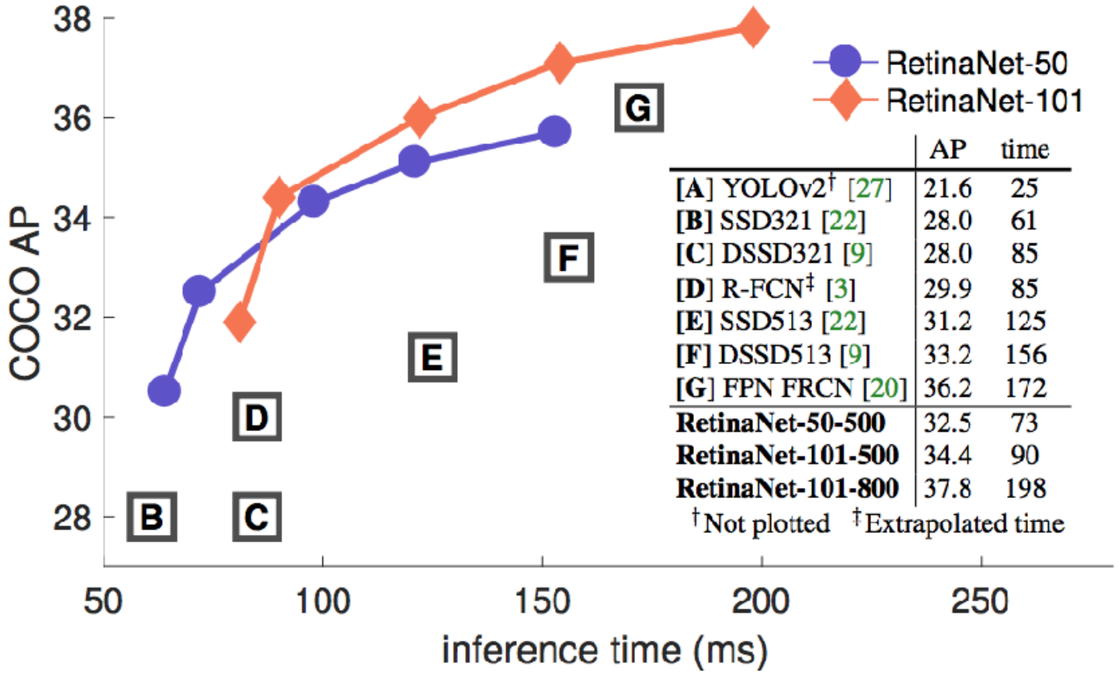
- One-stage (like YOLO and SSD) with Focal Loss
- Feature extraction with ResNet
- Multi-scale prediction – now with FPN



TY Lin et al. “Focal Loss for Dense Object Detection ”. ICCV 2017

RetinaNet

- Exceeds the accuracy of two-stage detectors + more efficient



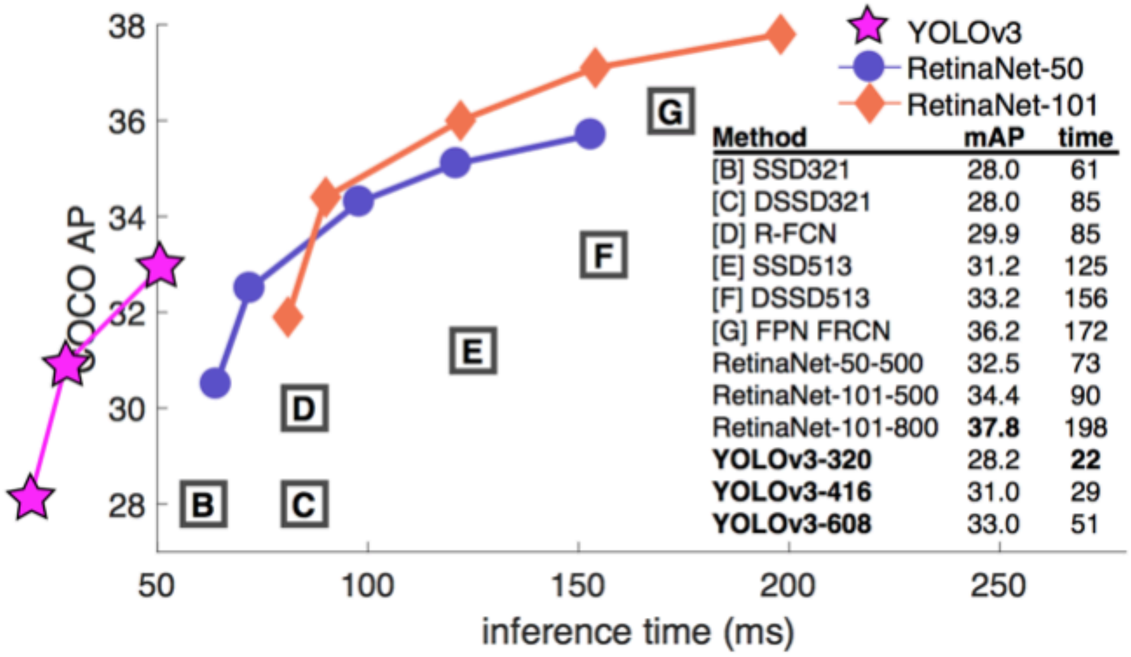
	AP	time
[A] YOLOv2† [27]	21.6	25
[B] SSD321 [22]	28.0	61
[C] DSSD321 [9]	28.0	85
[D] R-FCN‡ [3]	29.9	85
[E] SSD513 [22]	31.2	125
[F] DSSD513 [9]	33.2	156
[G] FPN FRCN [20]	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	37.8	198

†Not plotted ‡Extrapolated time

TY Lin et al. “Focal Loss for Dense Object Detection ”. ICCV 2017

RetinaNet

- The trade-off between the accuracy and efficiency remains.



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

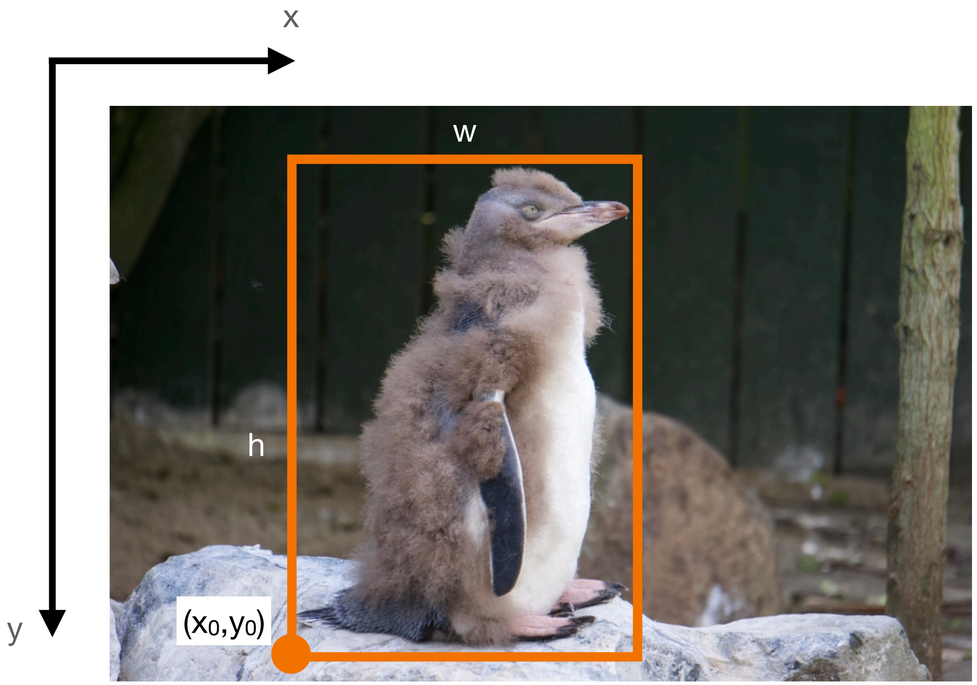
TY Lin et al. “Focal Loss for Dense Object Detection ”. ICCV 2017

Summary: Two-stage vs one-stage

- Two-stage:
  - can be more accurate (robustness to scale variation);
  - more technically involved (due to pooling);
  - slower than one-stage detectors.
- One-stage:
  - easier model design (more versatile in practical use);
  - competitive accuracy (heavy data augmentation);
  - fast.

Box representation

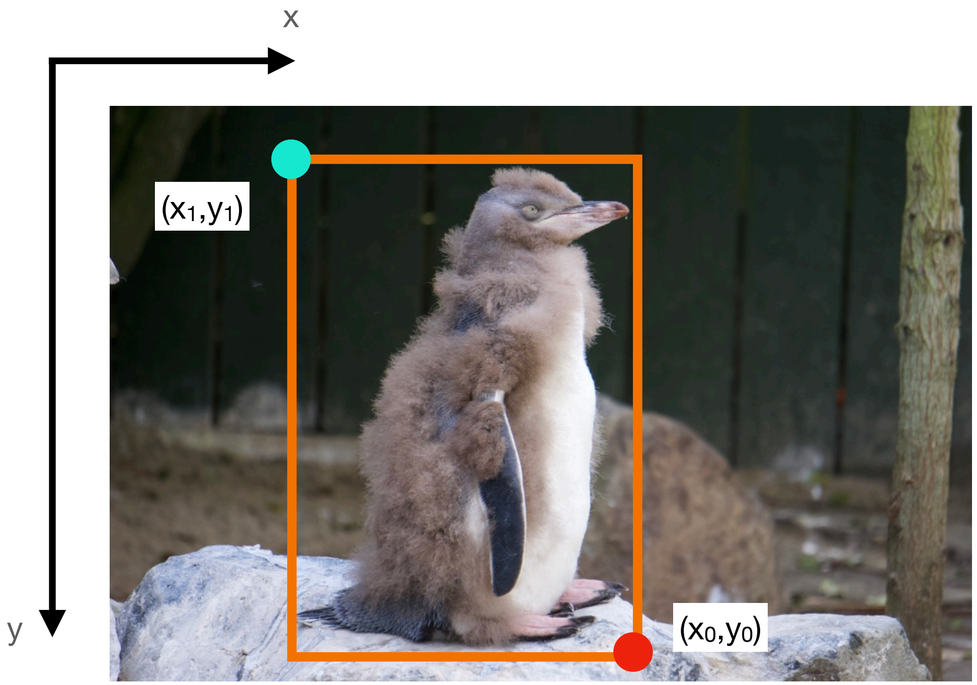
- There are many ways to define a bounding box



- (x_0, y_0, h, w)

Box representation

- There are many ways to define a bounding box



- (x_0, y_0, h, w)
- (x_0, y_0, x_1, y_1)

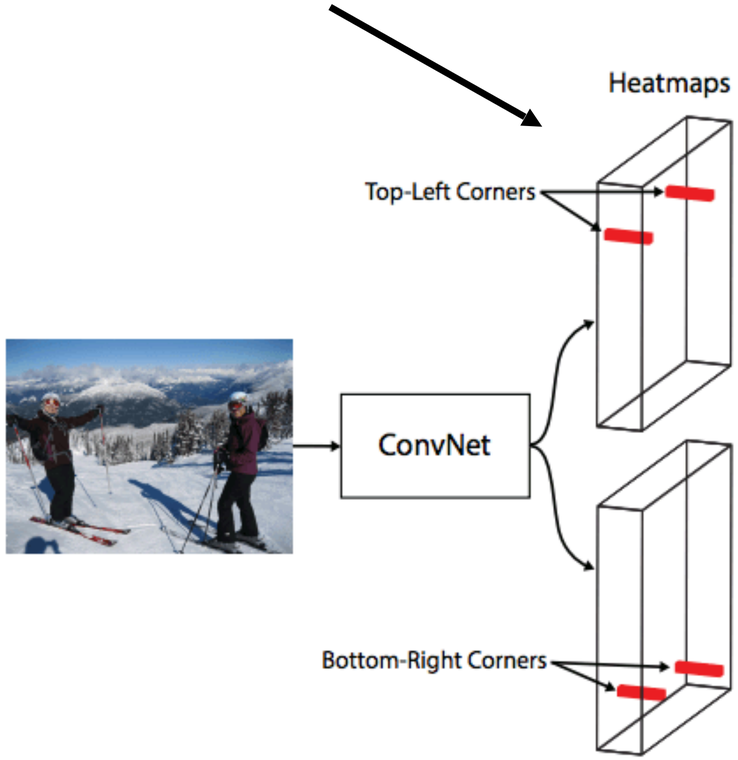
Keypoint-based detection

Heatmap A (top-left corner)

Heatmap B (bottom-right corner)

CornerNet

Probability map for each corner type

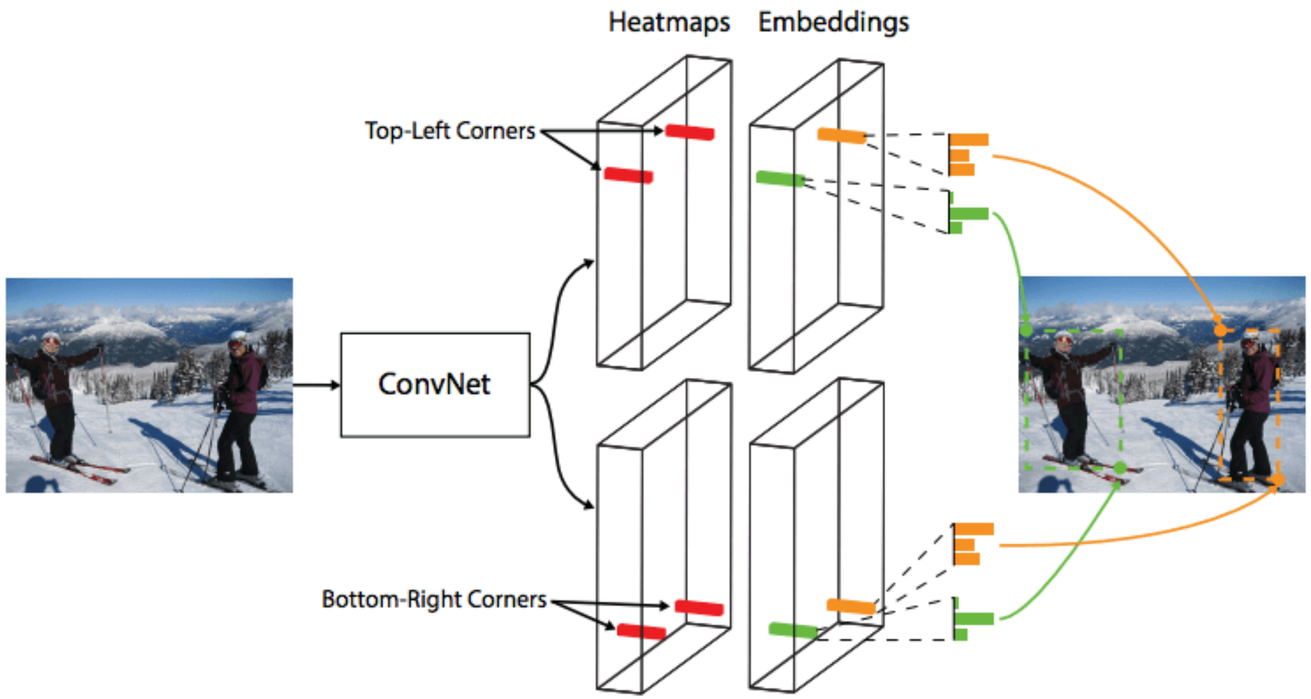


How to match top and bottom keypoints?

H. Law and J. Deng. „CornerNet: Detecting Objects as Paired Keypoints“. ECCV 2018

CornerNet

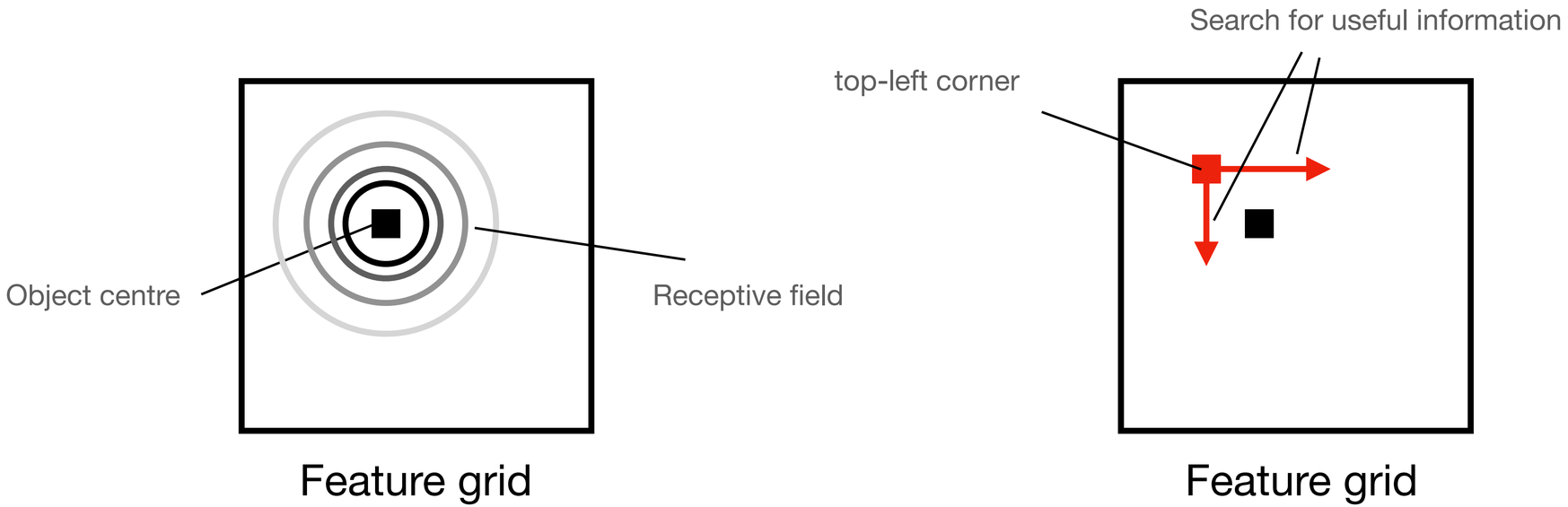
Learn a unique embedding for each object



H. Law and J. Deng. „CornerNet: Detecting Objects as Paired Keypoints“. ECCV 2018

CornerNet

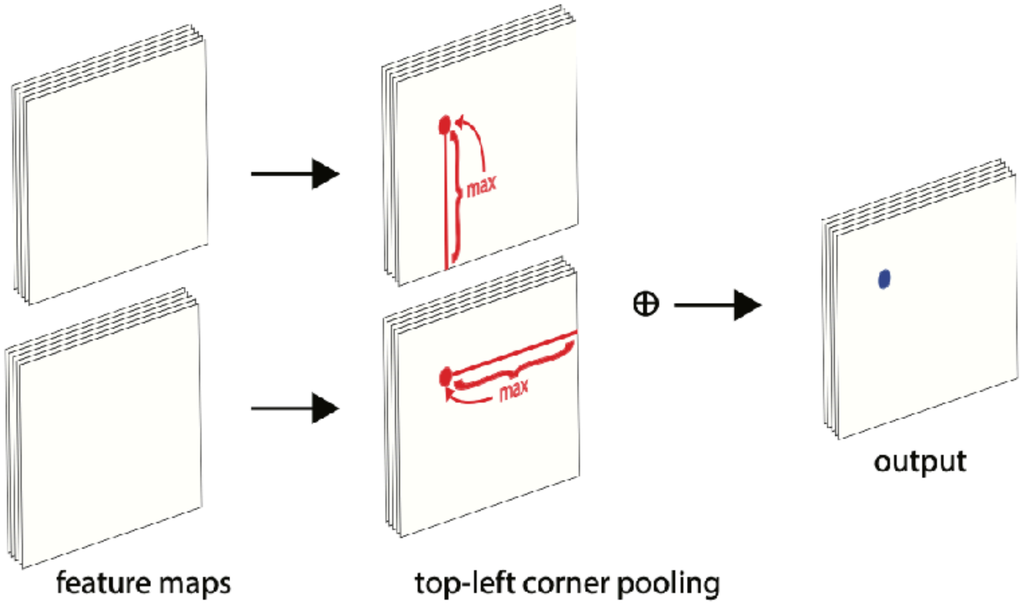
Issue: A displacement in the receptive field



Luo et al., “Understanding the Effective Receptive Field in Deep Convolutional Neural Networks”. NIPS 2016

CornerNet

Aggregate information with corner pooling:

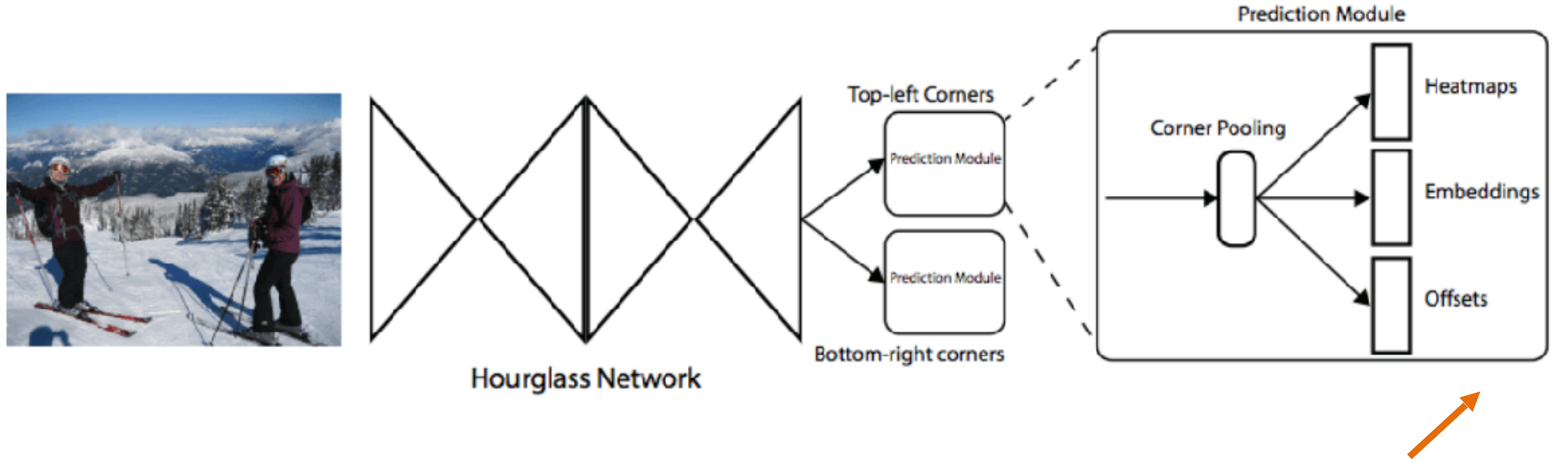


H. Law and J. Deng. „CornerNet: Detecting Objects as Paired Keypoints“. ECCV 2018

CornerNet

Hourglass network for keypoint detection

A. Newell et al. “Stacked hourglass networks for human pose estimation”. 2016



We predict corners at a lower resolution and then regress an offset (bounding box correction as we have seen for all methods)

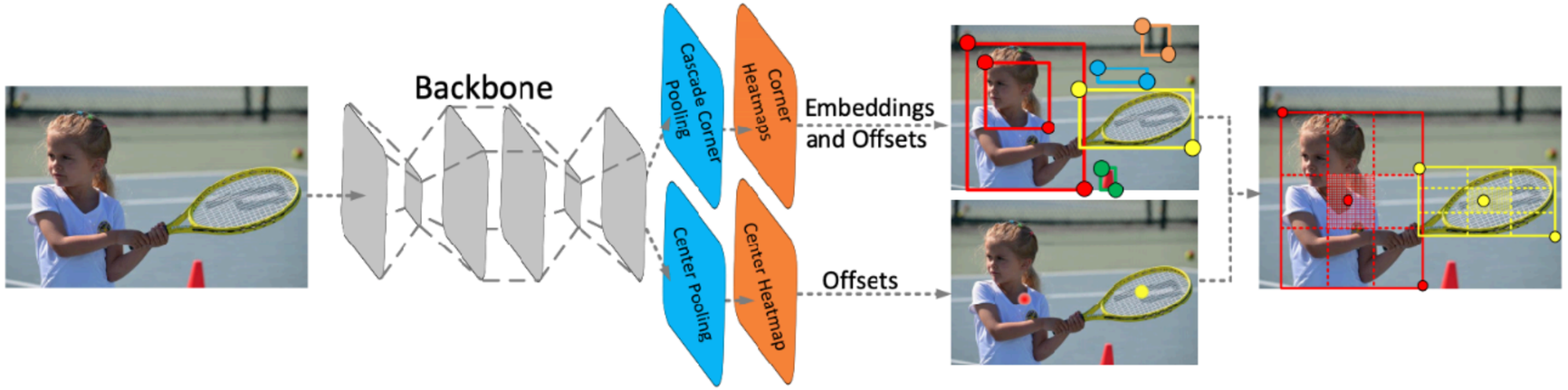
H. Law and J. Deng. „CornerNet: Detecting Objects as Paired Keypoints“. ECCV 2018

CornerNet

- What is the problem with CornetNet?
- Many incorrect bounding boxes (especially small)
  - too many False Positives
- Hypothesis: corner pooling may aggregate irrelevant context

CenterNet

- Idea: focus on the center of the object to infer its class
- Use the corners as proposals, and the center to verify the class of the object and filter out outliers



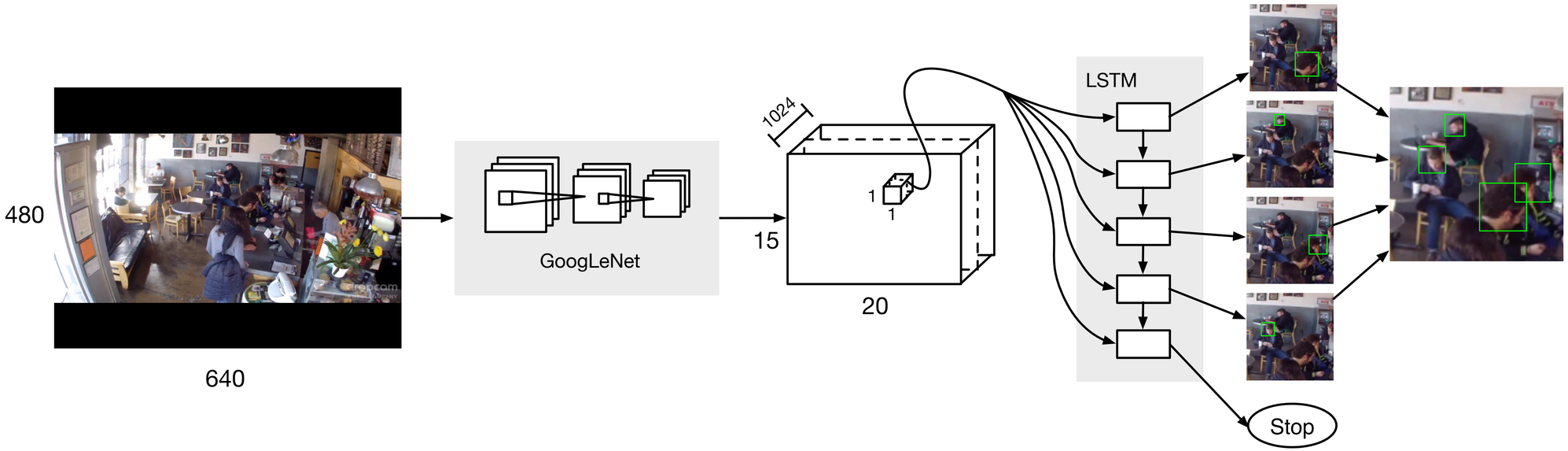
K. Duan et al. „CenterNet: Keypoint Triplets for Object Detection“. ICCV 2019

Sequential detection

- So far:
  - associate anchors with every grid cell;
  - the number of anchors has to be carefully tuned;
  - the total number of anchors may be still excessive.

- Can we detect sequentially instead?
  - Predict one bounding box after another;
  - stop iteration once all objects are detected.

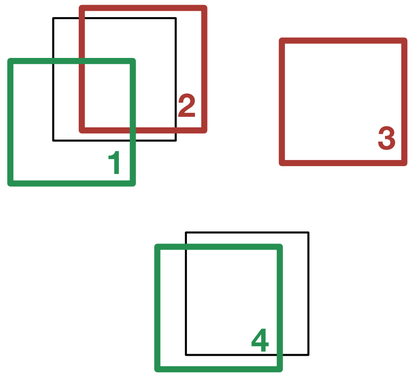
Recurrent object detection



Stewart et al., “End-to-End People Detection in Crowded Scenes”. CVPR 2016

Recurrent object detection

- At training time we iterate N times (N is the number of ground-truth).
- How to associate the predictions with the ground truth?



- We can use Hungarian matching using IoU score

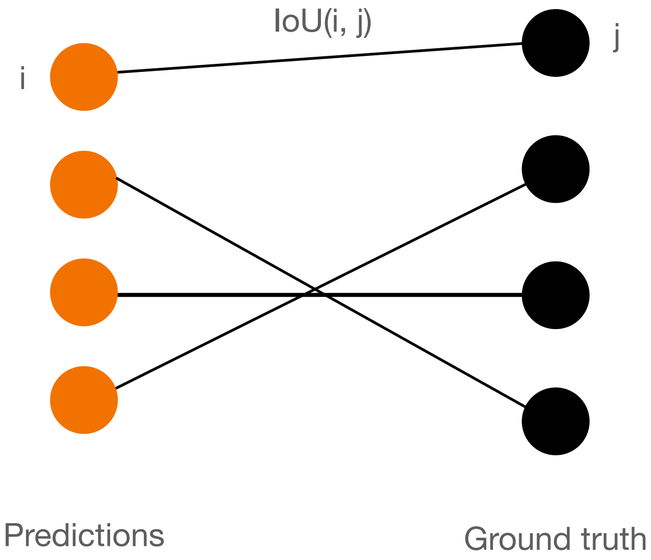
Stewart et al., “End-to-End People Detection in Crowded Scenes”. CVPR 2016

Permutation-invariant training

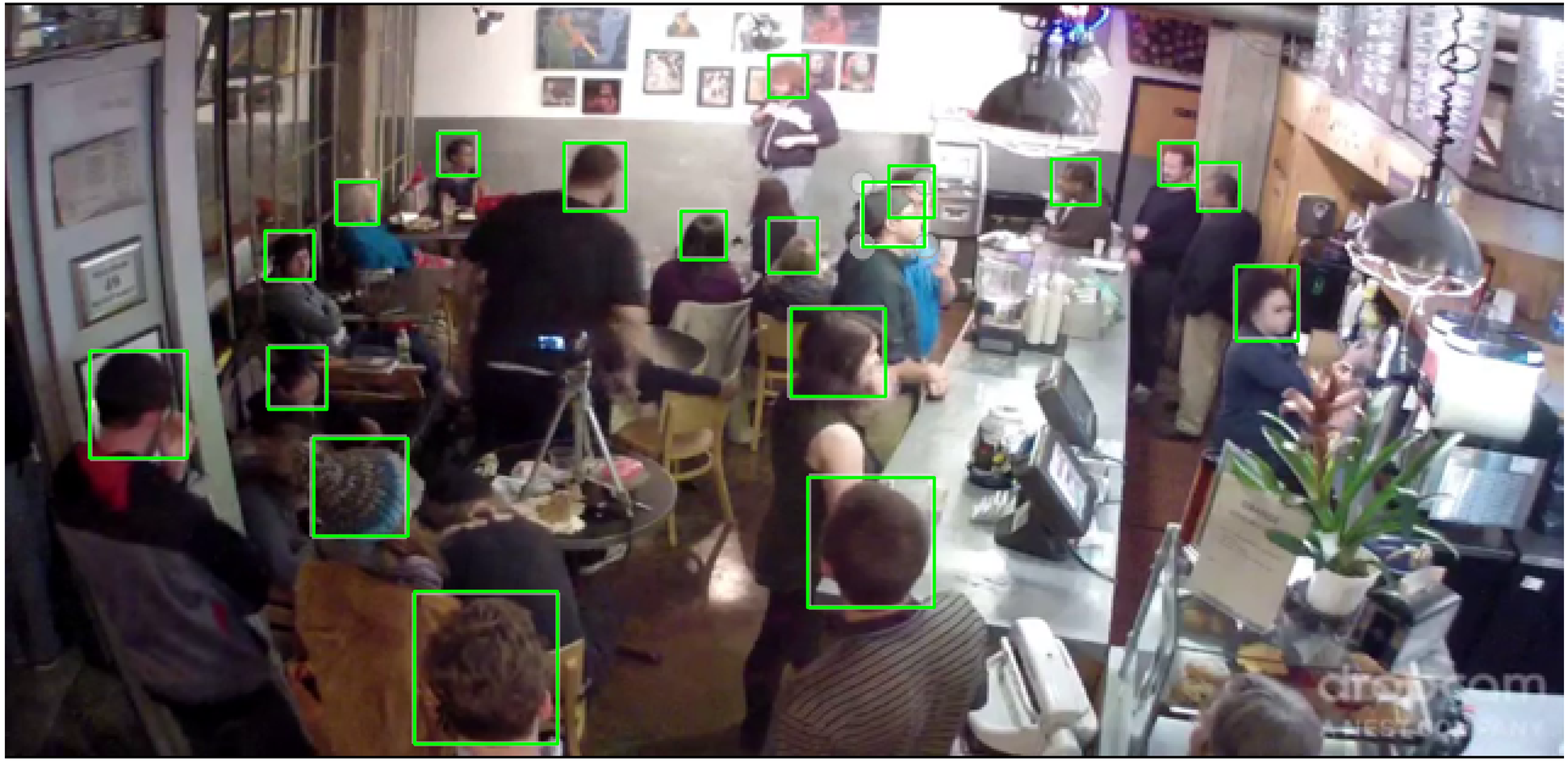
- Bipartite matching

$$\max_{s \in P} \sum_{i} IoU(i, s(i))$$

(P is a permutation)



Recurrent object detection



Stewart et al., “End-to-End People Detection in Crowded Scenes”. CVPR 2016

Recurrent object detection

- No anchors: one iteration – one bounding box.
- Cons:
  - does not scale well to high number of objects;
  - classification, start/stop criteria is challenging to learn well.

Summary

- Advanced two-stage architectures (Fast R-CNN, Faster R-CNN)
- One-stage detectors (YOLO, SSD, RetinaNet);
- Keypoint-based detection (CornerNet);
- Recurrent object detection.

Additional reading

- Shrivastava, Gupta, Girshick. “Training region-based object detectors with online hard example mining”. CVPR 2016.
- Dai, Li, He and Sun. “R-FCN: Object detection via region-based fully convolutional networks”. 2016.
- Dai, Qi, Xiong, Li, Zhang, Hu and Wei. “Deformable convolutional networks”. ICCV 2017.
- Lin, Dollar, Girshick, He, Hariharan and Belongie. “Feature Pyramid Networks for object detection”. CVPR 2017.