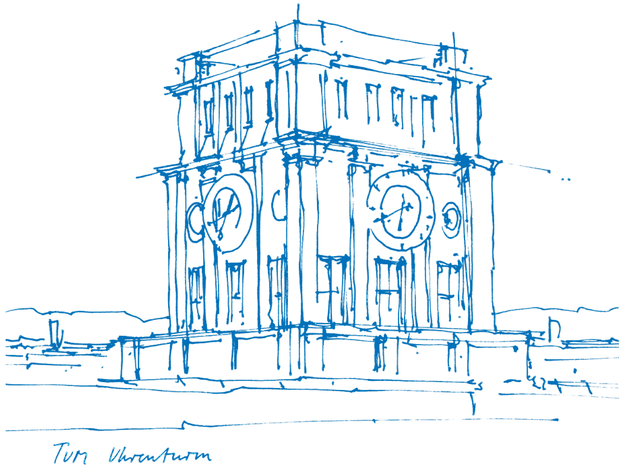
Computer Vision III:

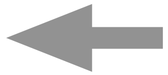
Two-stage object detectors

Dr. Nikita Araslanov
31.10.2023

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

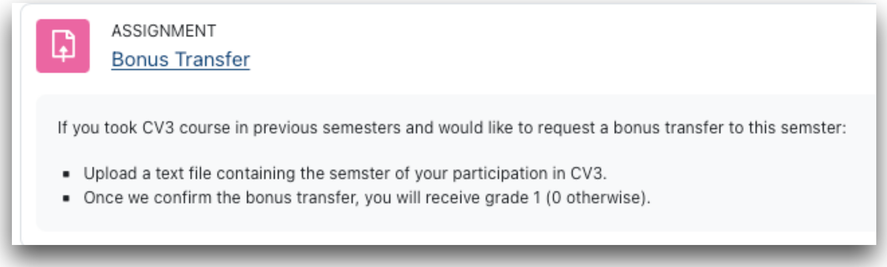


Our progress so far

1. Introduction. Object detection with sliding window
2. Two-stage object detection
3. **Single-stage object detection**  We are here

Announcement

- Bonus transfer for assignments
  - Deadline: November 30

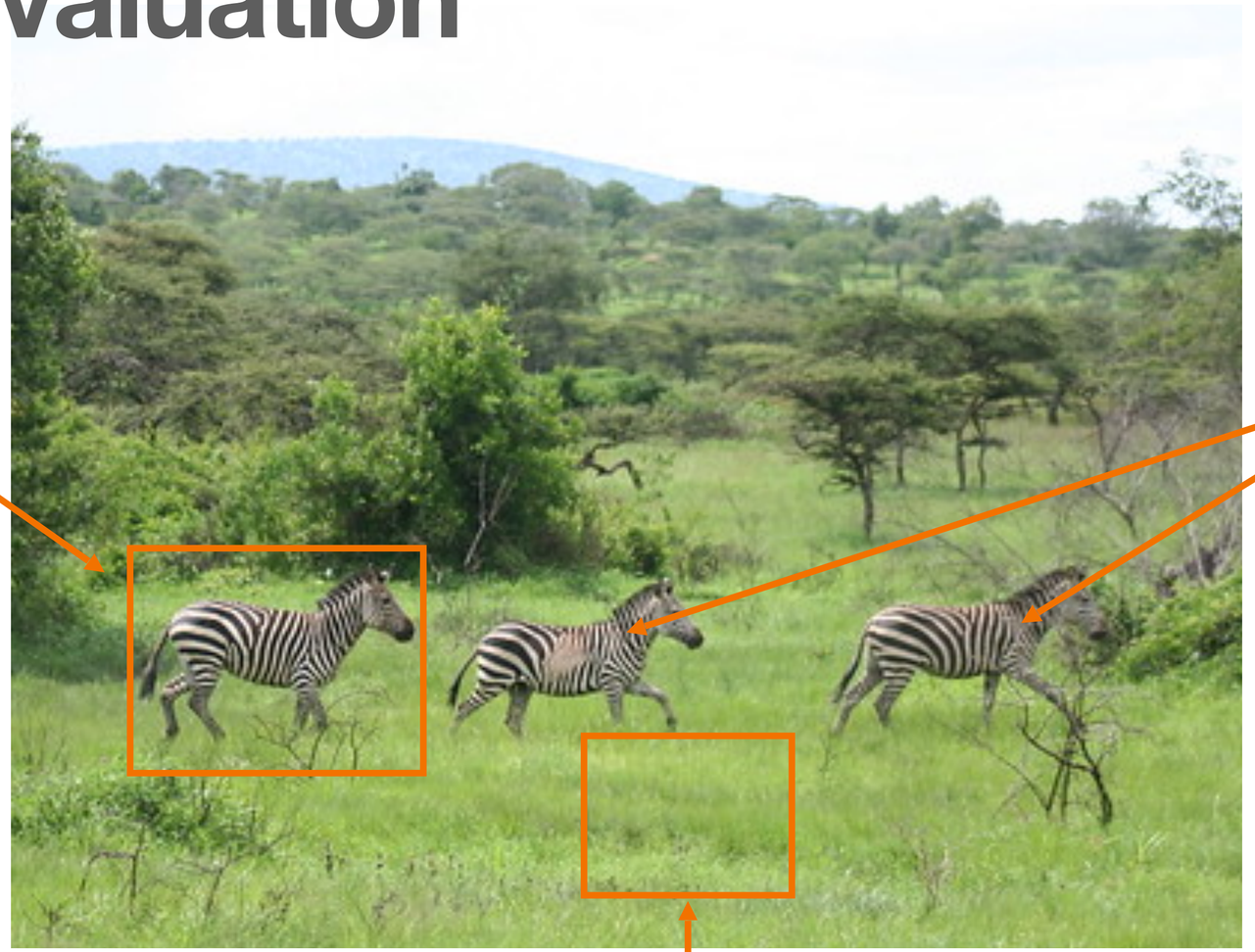


- Lecture 02 – second take:
  - https://live.rbg.tum.de/w/CV3DSuT/40572

Detection evaluation

Detection evaluation

TP = True positive



FN = False negative

FP = False positive

Detection evaluation

- Precision: “How many zebras you found are actually zebras”

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

# of predicted boxes

- Recall: “How many actual zebras in the image/dataset you could find”

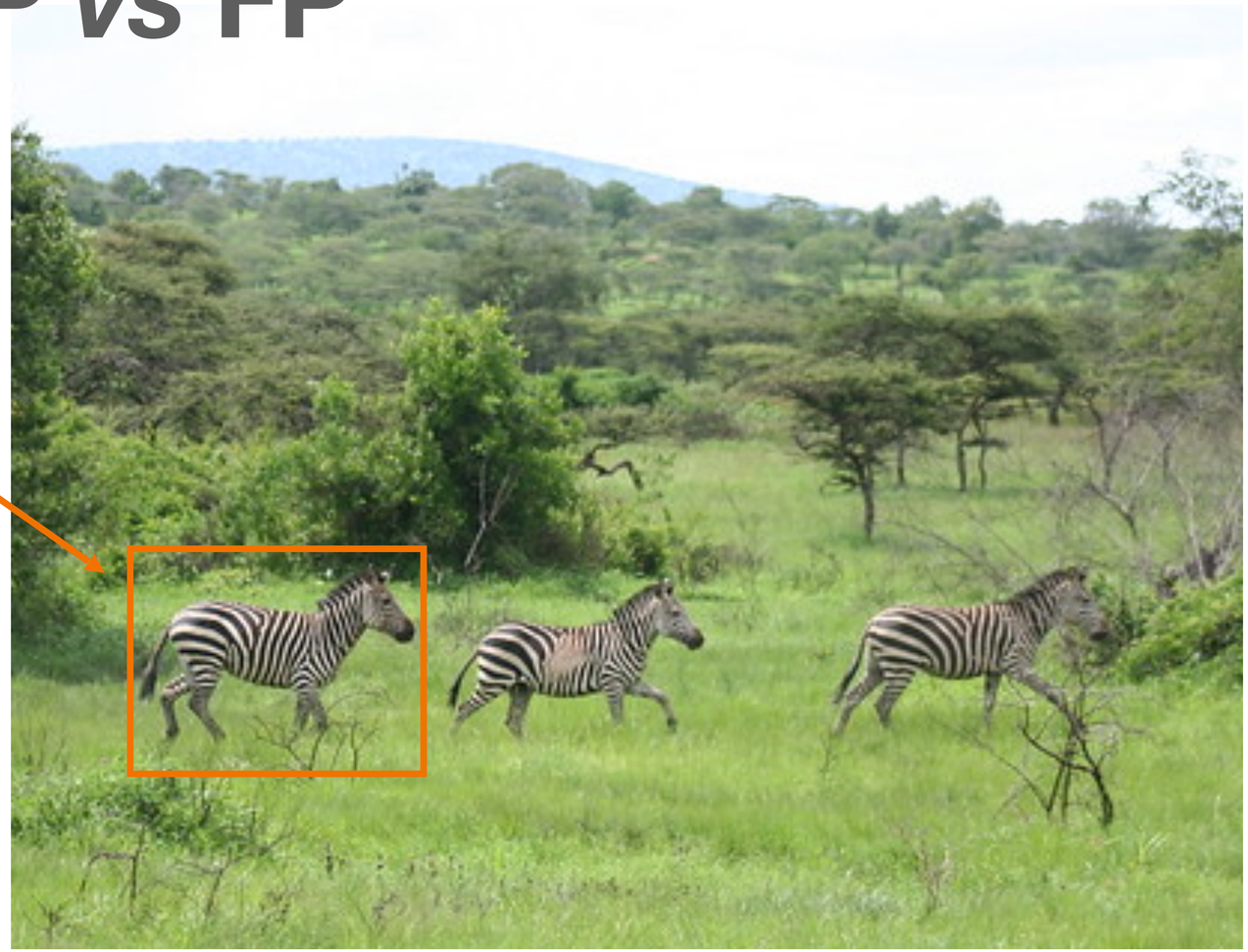
$$Recall = \frac{TP}{TP + FN}$$

# of ground-truth boxes

TP = True positives FP = False positives FN = False negatives

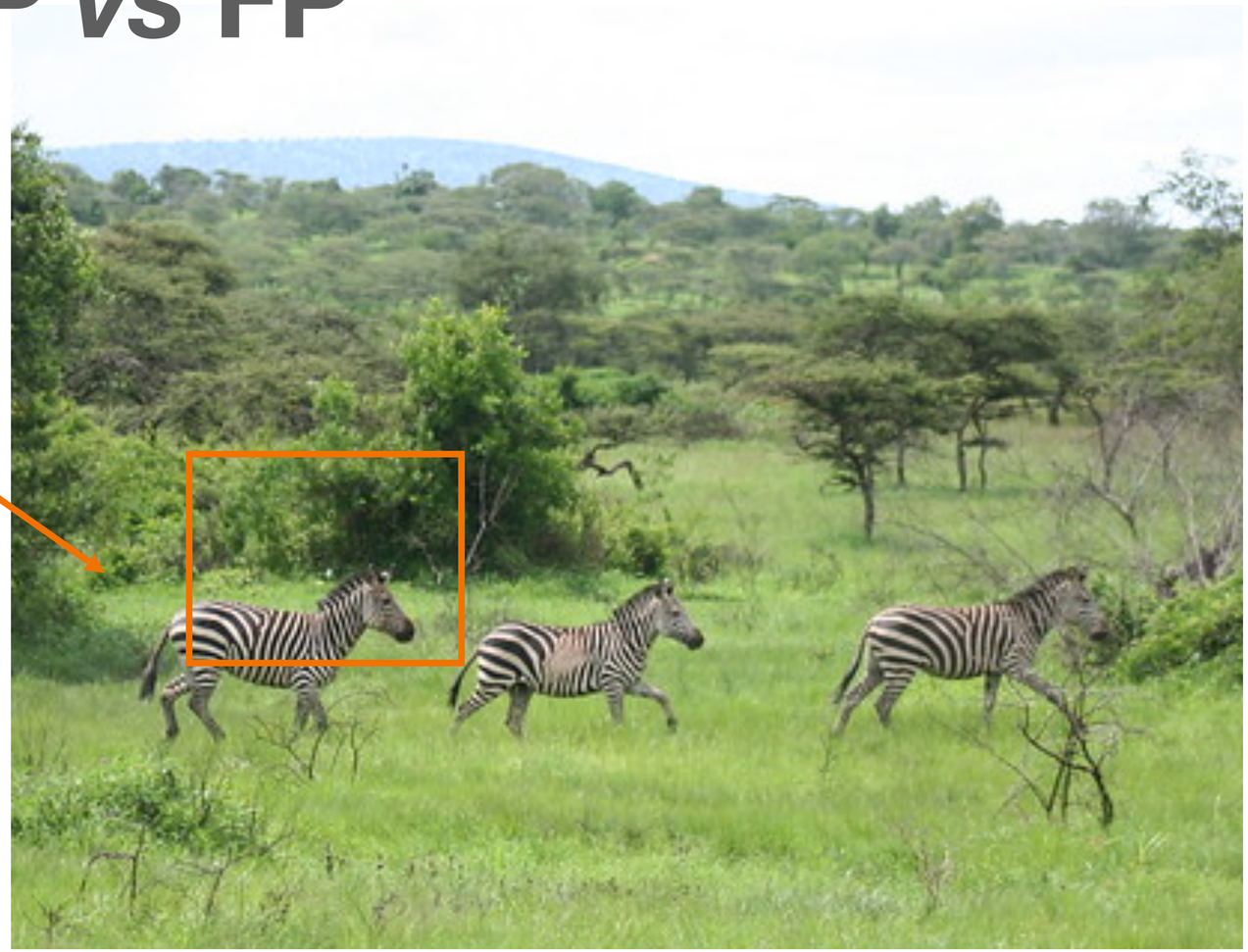
Counting TP *vs* FP

TP = True positive



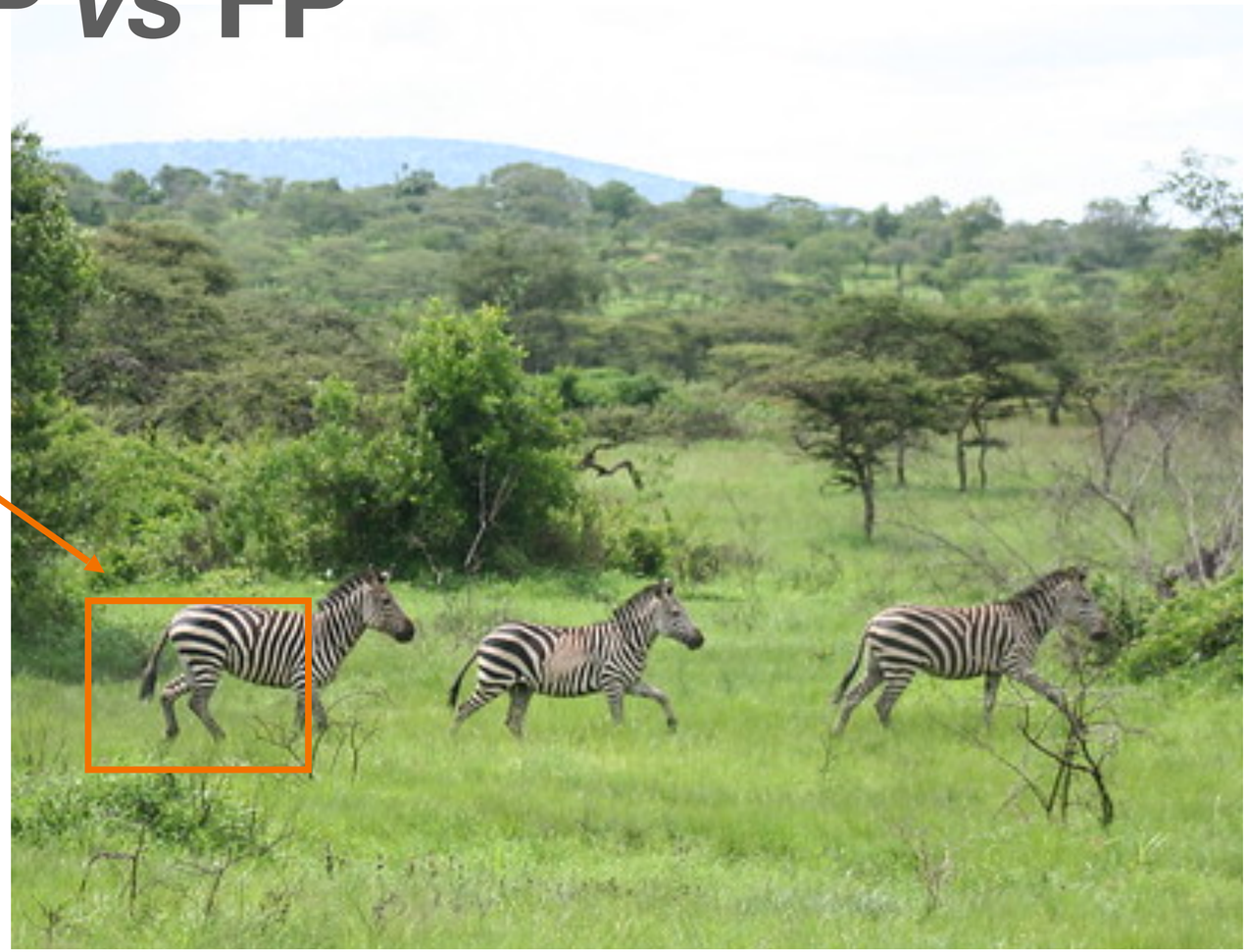
Counting TP *vs* FP

TP = True positive?



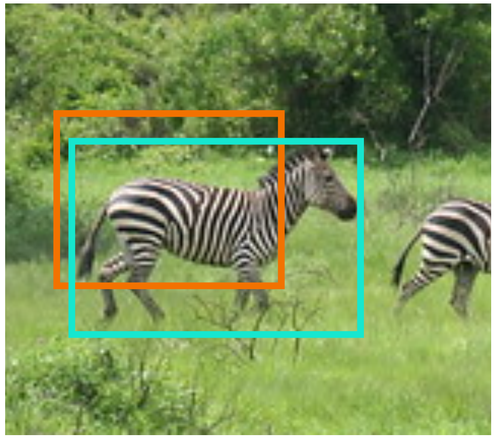
Counting TP *vs* FP

TP = True positive?



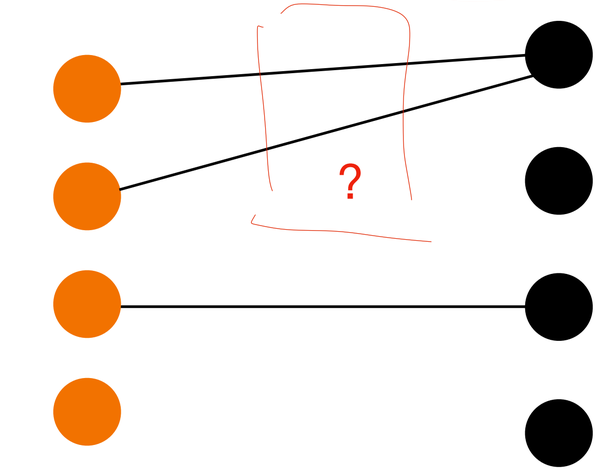
Counting TP *vs* FP

- What is a true positive?
  - Use the Intersection over Union (IoU)
  - e.g. if IoU $> 0.5 \rightarrow$ positive match
- The criterion is defined by the benchmarks (MS-COCO, Pascal VOC)



Resolving conflicts

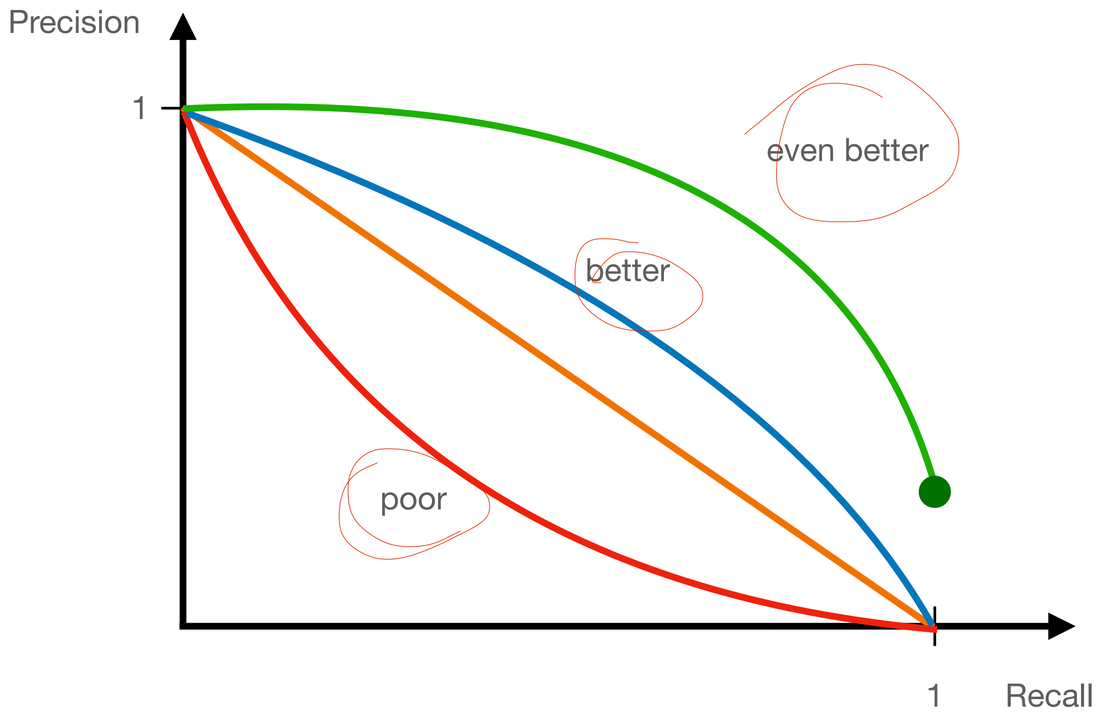
- Each prediction can match at most 1 ground-truth box
- Each ground-truth box can match at most 1 prediction



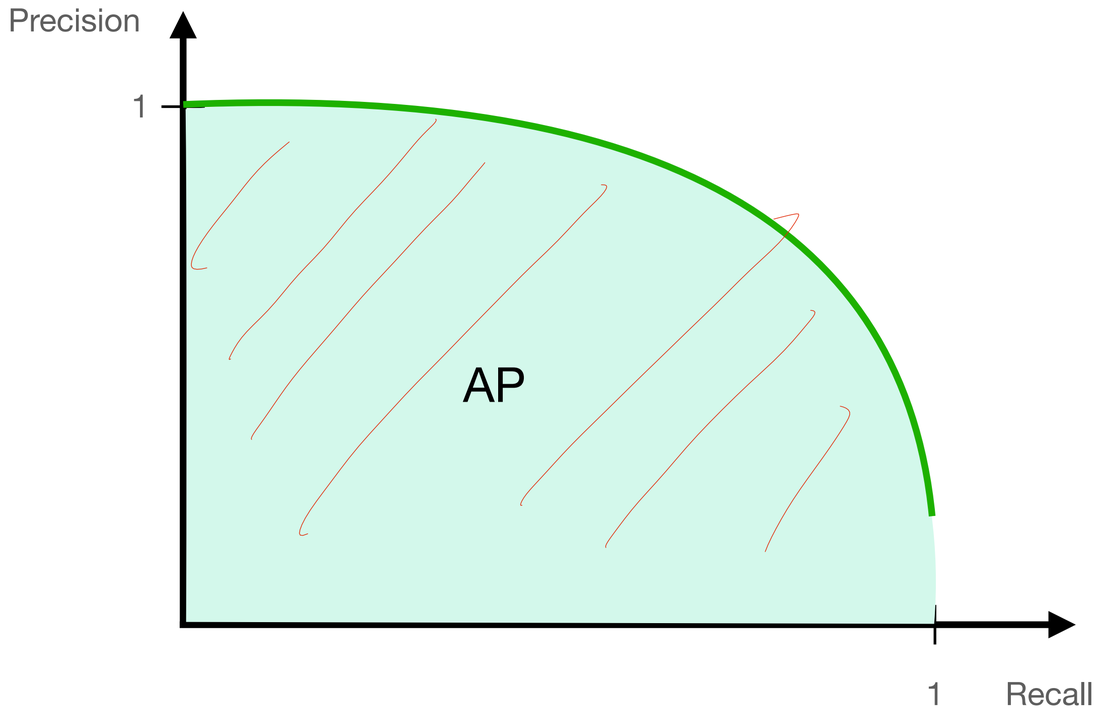
Select the most confident prediction first

All-in-one metric: Average Precision (AP)

- There is often a trade-off between Precision and Recall



Average Precision (AP)



- AP = the area under the Precision-Recall curve (AUC)
- How to fix the a particular Recall value?

Leverage confidence of model predictions

Computing Average Precision

1. Sort the predicted boxes by confidence score
2. For each prediction find the associated ground truth
   - The ground-truth should be unassigned and pass IoU threshold
3. Compute cumulative TP and FP
   - TP: [1,0,1] FP: [0,1,0] → cTP = [1,1,2], cFP = [0,1,1]
4. Compute Precision and Recall (#GT = 3 → TP + FN = 3)
   - Precision: [1/1, 1/2, 2/3]; Recall (#GT = 3): [1/3,1/3,2/3]
5. Plot the Precision-Recall curve and the area beneath (num. integration)

Average Precision: Example

Rank	True/false	Precision	Recall
1	T		
2	T		
3	F		
4	T		
5	F		
6	F		
7	T		
8	F		
9	T		
10	T		
11	F		
12	F		

My method predicts 12 boxes;
There are 6 objects (ground-truth)

Average Precision: Example

Rank	True/false	Precision	Recall
1	T	1.0	0.17
2	T		
3	F		
4	T		
5	F		
6	F		
7	T		
8	F		
9	T		
10	T		
11	F		
12	F		

My method predicts 12 boxes;
There are 6 objects (ground-truth)

Average Precision: Example

Rank	True/false	Precision	Recall
1	T	1.0	0.17
2	T	1.0	0.33
3	F		
4	T		
5	F		
6	F		
7	T		
8	F		
9	T		
10	T		
11	F		
12	F		

My method predicts 12 boxes;
There are 6 objects (ground-truth)

Average Precision: Example

Rank	True/false	Precision	Recall
1	T	1.0	0.17
2	T	1.0	0.33
3	F	0.67	0.33
4	T		
5	F		
6	F		
7	T		
8	F		
9	T		
10	T		
11	F		
12	F		

My method predicts 12 boxes;
There are 6 objects (ground-truth)

Average Precision: Example

Rank	True/false	Precision	Recall
1	T	1.0	0.17
2	T	1.0	0.33
3	F	0.67	0.33
4	T	0.75	0.5
5	F	0.6	0.5
6	F	0.5	0.5
7	T	0.57	0.67
8	F	0.5	0.67
9	T	0.56	0.83
10	T	0.6	1.0
11	F	0.55	1.0
12	F	0.5	1.0

Our method predicts 12 boxes;
There are 6 objects (ground-truth)

Average Precision: Example

- Plot Precision vs Recall

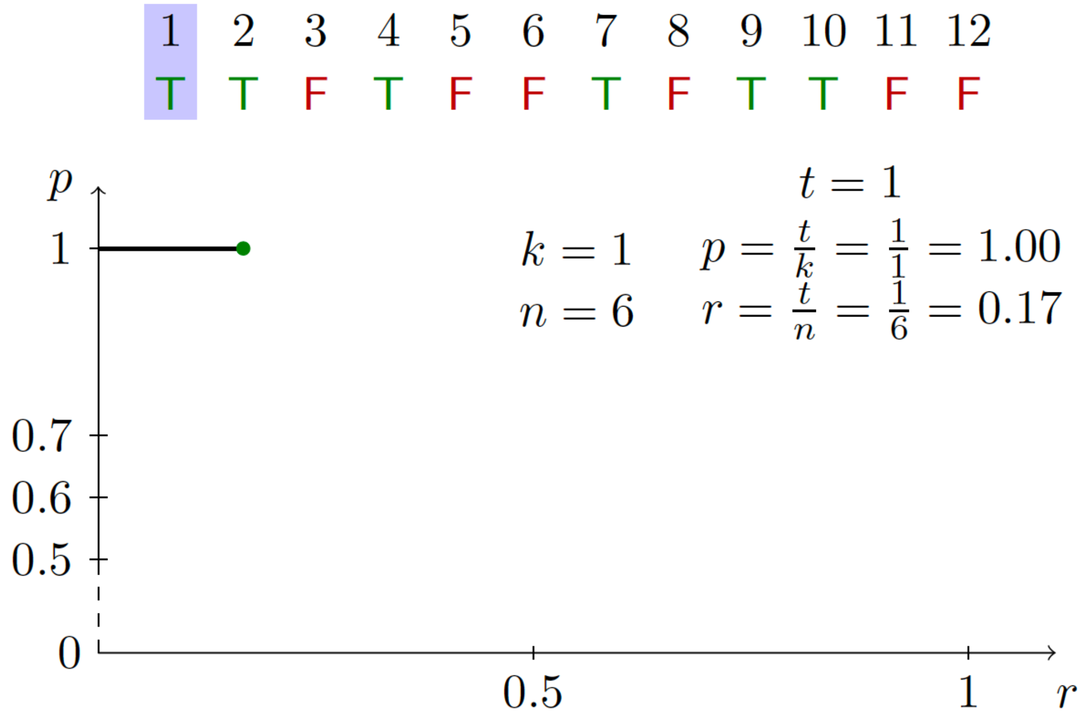


Image Credit: Y. Avrithis, Object detection lecture. INRIA.

Average Precision: Example

- Plot Precision vs Recall

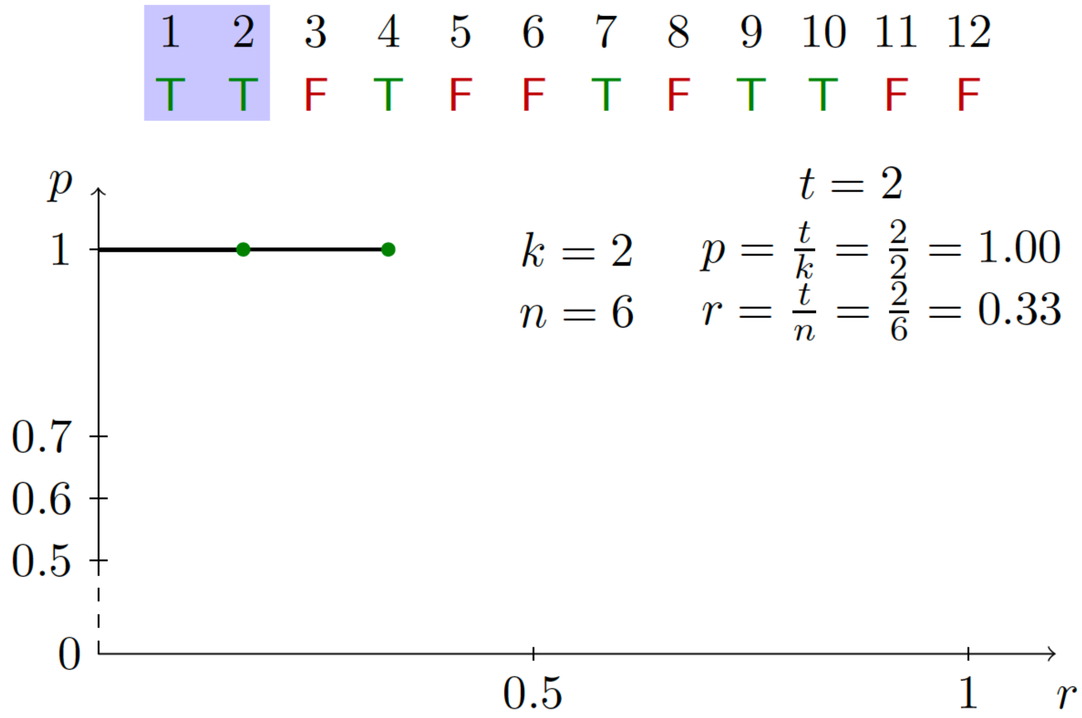


Image Credit: Y. Avrithis, Object detection lecture. INRIA.

Average Precision: Example

- Plot Precision vs Recall

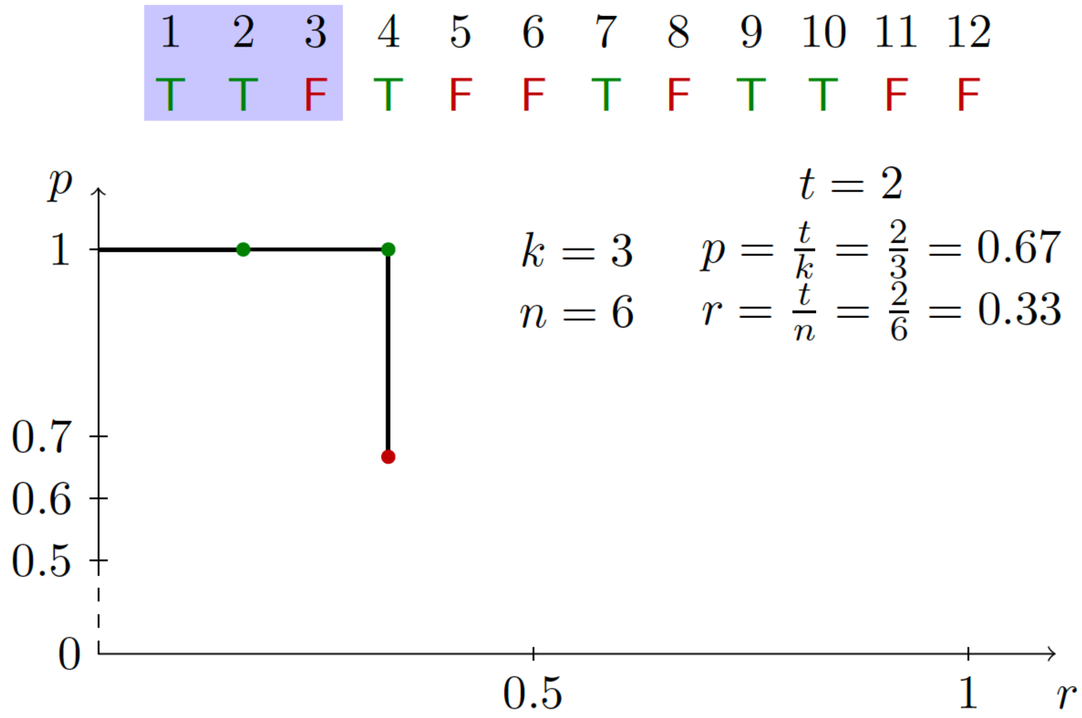


Image Credit: Y. Avrithis, Object detection lecture. INRIA.

Average Precision: Example

- Plot Precision vs Recall

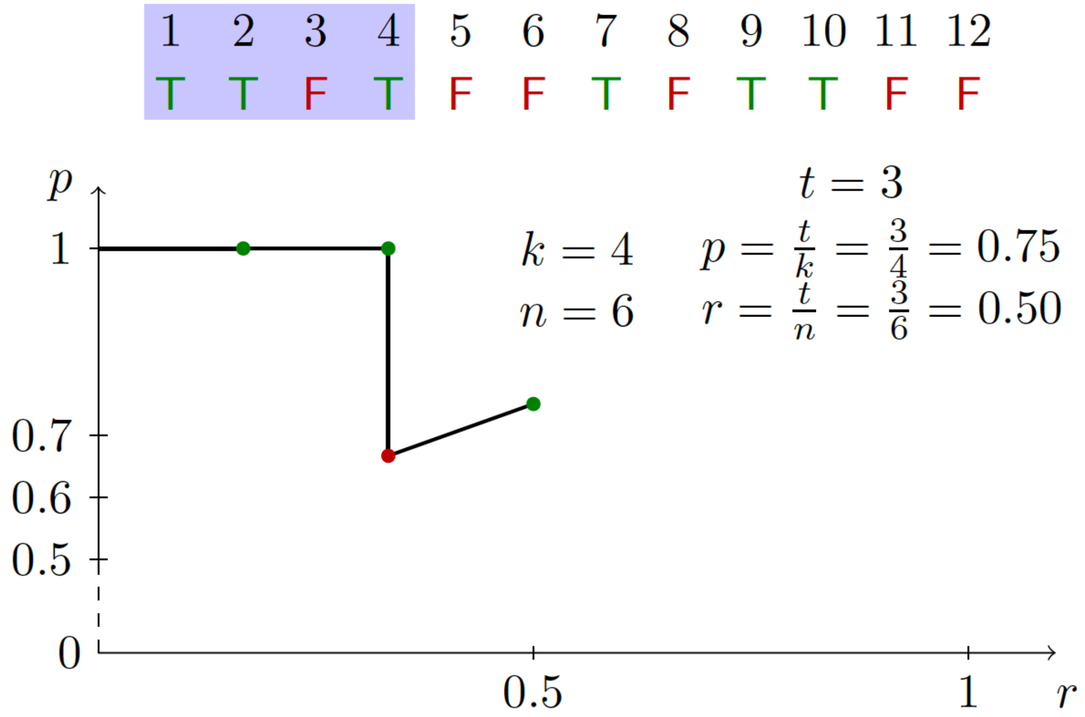


Image Credit: Y. Avrithis, Object detection lecture. INRIA.

Average Precision: Example

- Plot Precision vs Recall

1	2	3	4	5	6	7	8	9	10	11	12
T	T	F	T	F	F	T	F	T	T	F	F

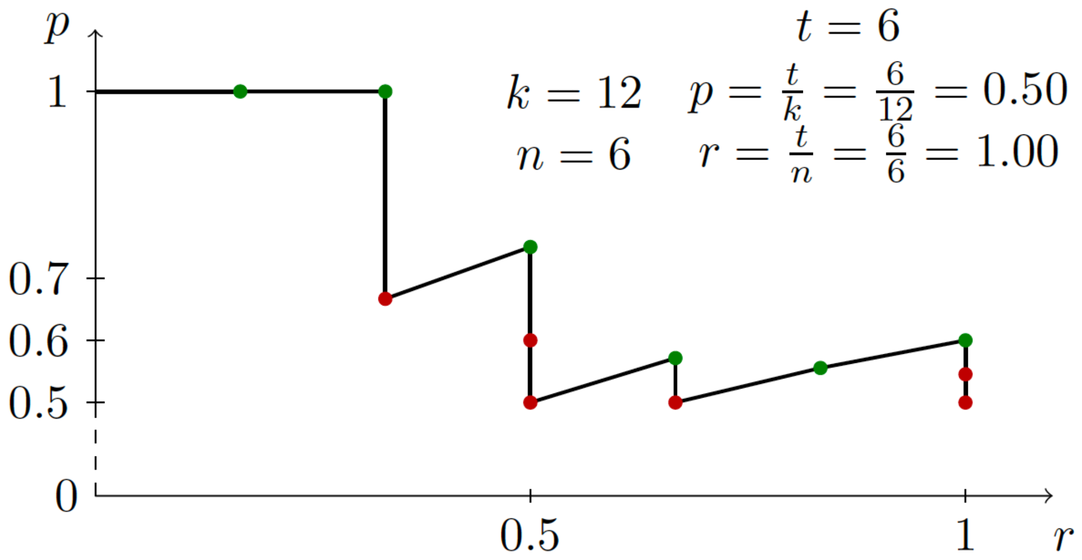


Image Credit: Y. Avrithis, Object detection lecture. INRIA.

Average Precision: Example

- Plot Precision vs Recall

1	2	3	4	5	6	7	8	9	10	11	12
T	T	F	T	F	F	T	F	T	T	F	F

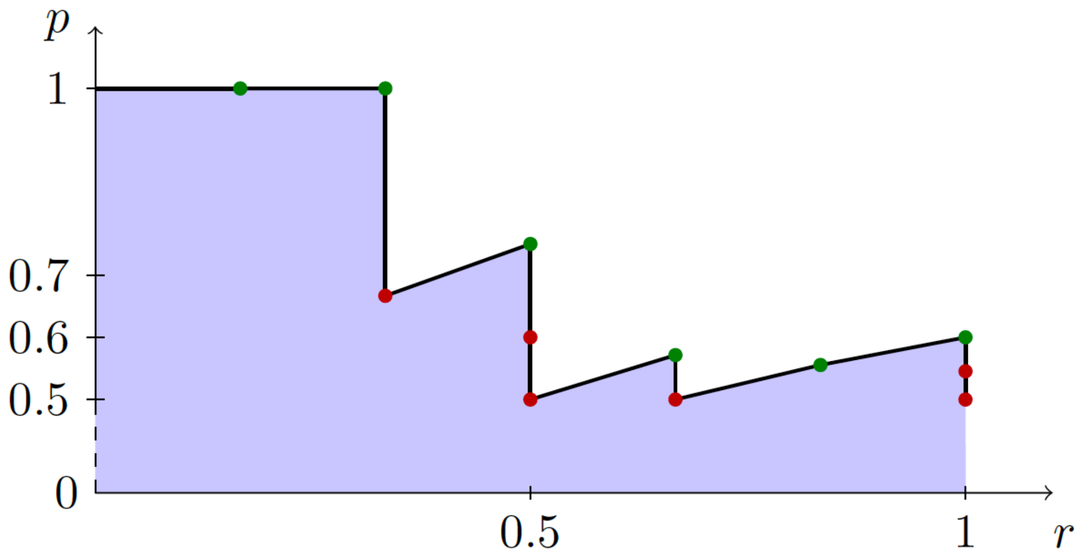


Image Credit: Y. Avrithis, Object detection lecture. INRIA.

Average Precision: Example

- Plot Precision vs Recall

1	2	3	4	5	6	7	8	9	10	11	12
T	T	F	T	F	F	T	F	T	T	F	F

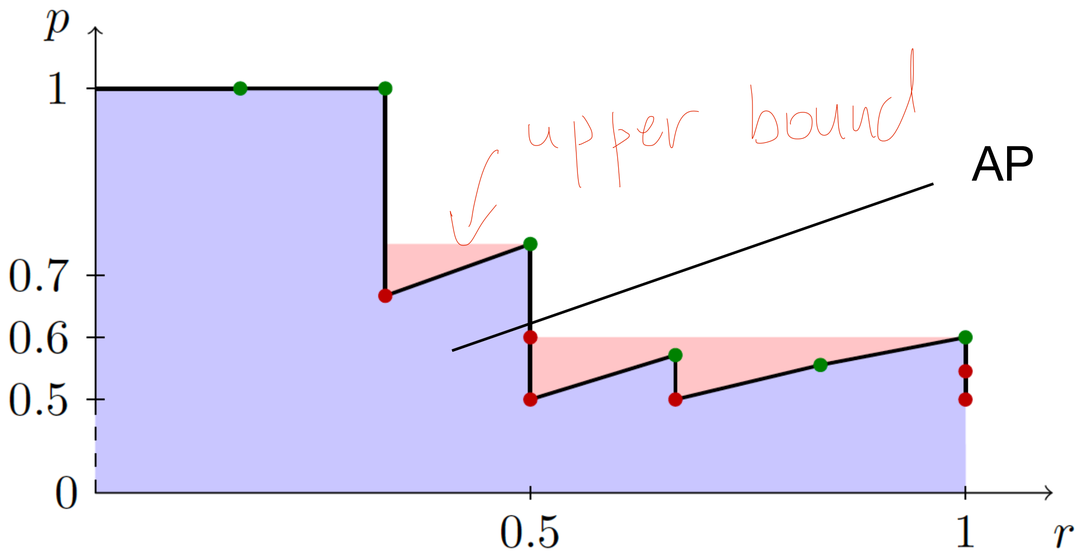


Image Credit: Y. Avrithis, Object detection lecture. INRIA.

AP derivatives

- AP may be averaged over multiple IoU thresholds
- mAP is the average over object categories
- Many other flavours:

MS-COCO

Average Precision (AP):

AP	% AP at IoU=.50:.05:.95 **(primary challenge metric)**
$AP^{IoU=.50}$	% AP at IoU=.50 (PASCAL VOC metric)
$AP^{IoU=.75}$	% AP at IoU=.75 (strict metric)

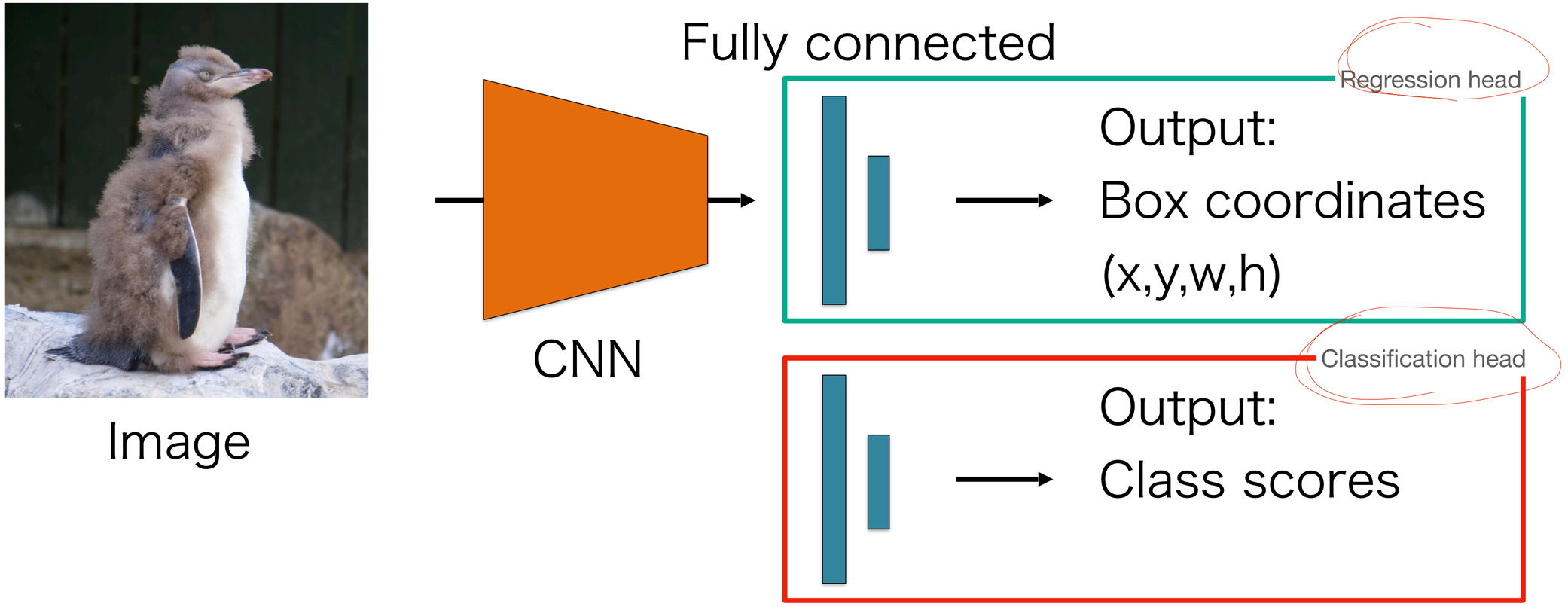
AP Across Scales:

AP^{small}	% AP for small objects: area < 32^2
AP^{medium}	% AP for medium objects: 32^2 < area < 96^2
AP^{large}	% AP for large objects: area > 96^2

Previous lecture

Localisation and classification

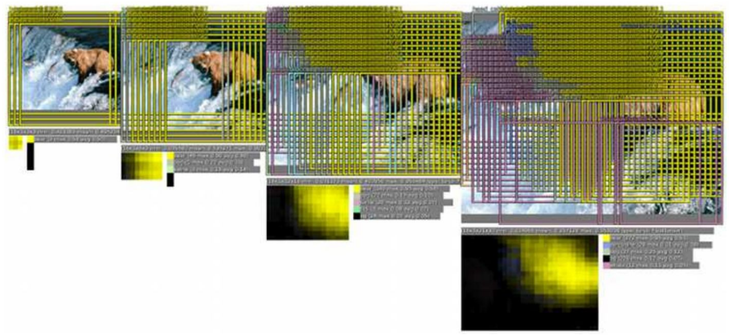
- Bounding box regression and classification



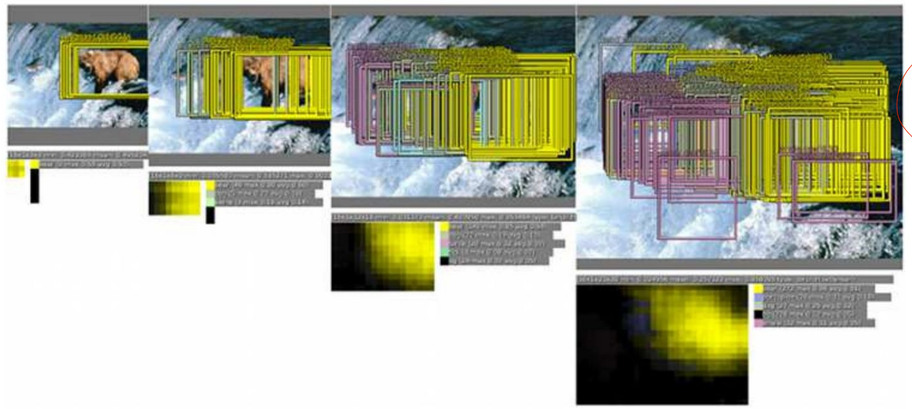
Overfeat

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

Window positions + score maps

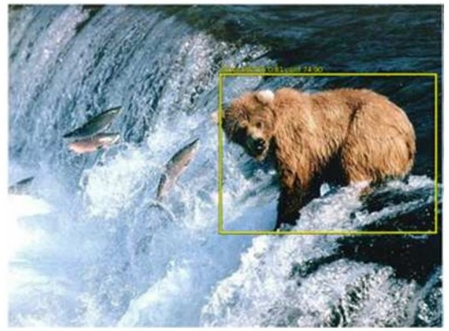


Box regression outputs



NMS

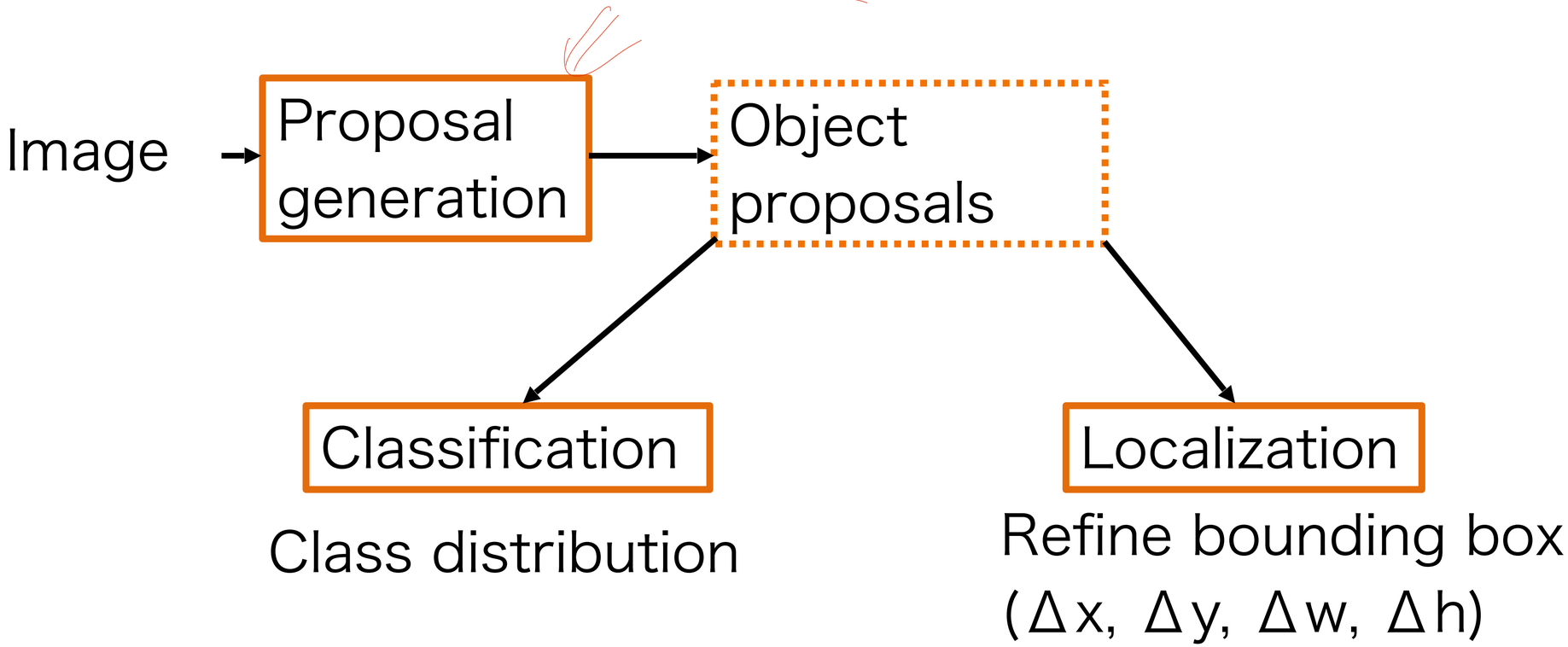
Final Prediction



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

Two-stage detectors

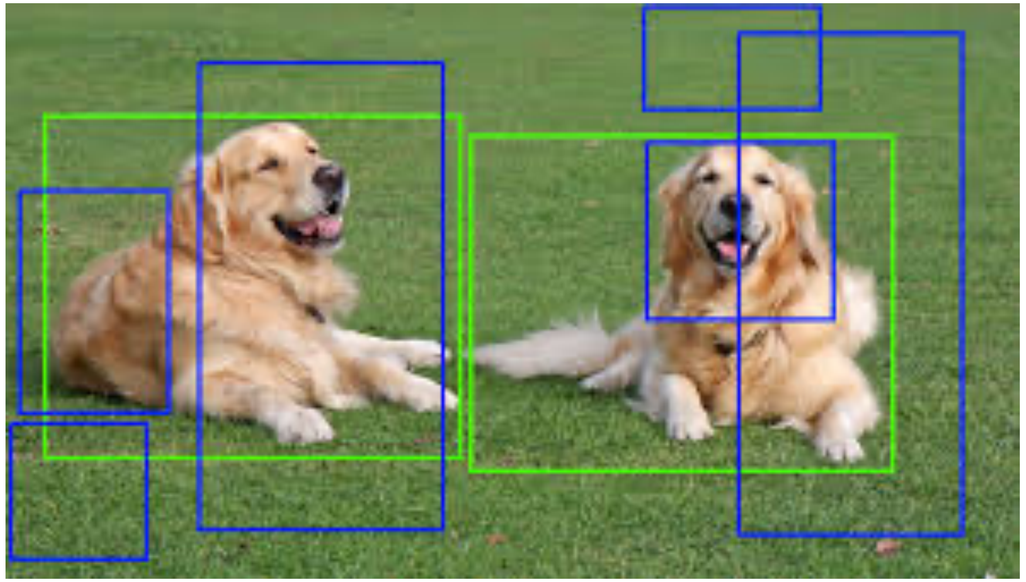
- ~~"Pre-filtering"~~ → proposal generation



Region proposals

- We can use a heuristic-based method to generate “interesting” regions in an image

1. Obtain region proposals.
2. Refine & classify them.



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

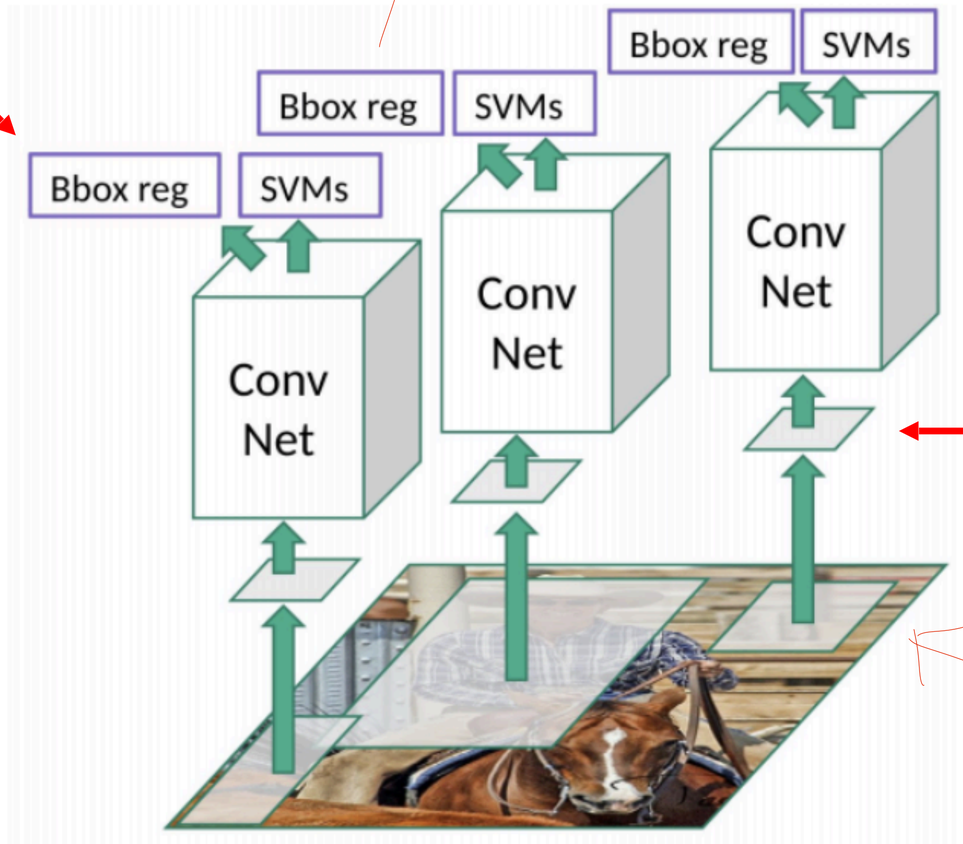
R-CNN family

R-CNN

many net ⇔ not efficient

Regression head to refine the bounding box location

Classification head



Extract features

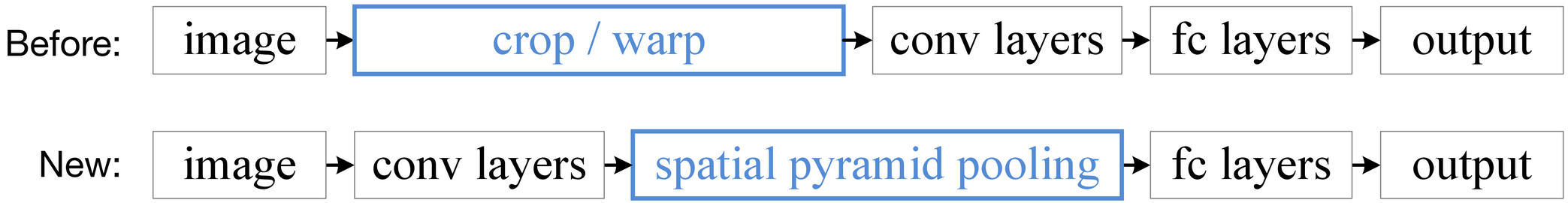
Warping to a fix size 227 x 227

different shape

SPP-Net: Spatial Pyramid Pooling

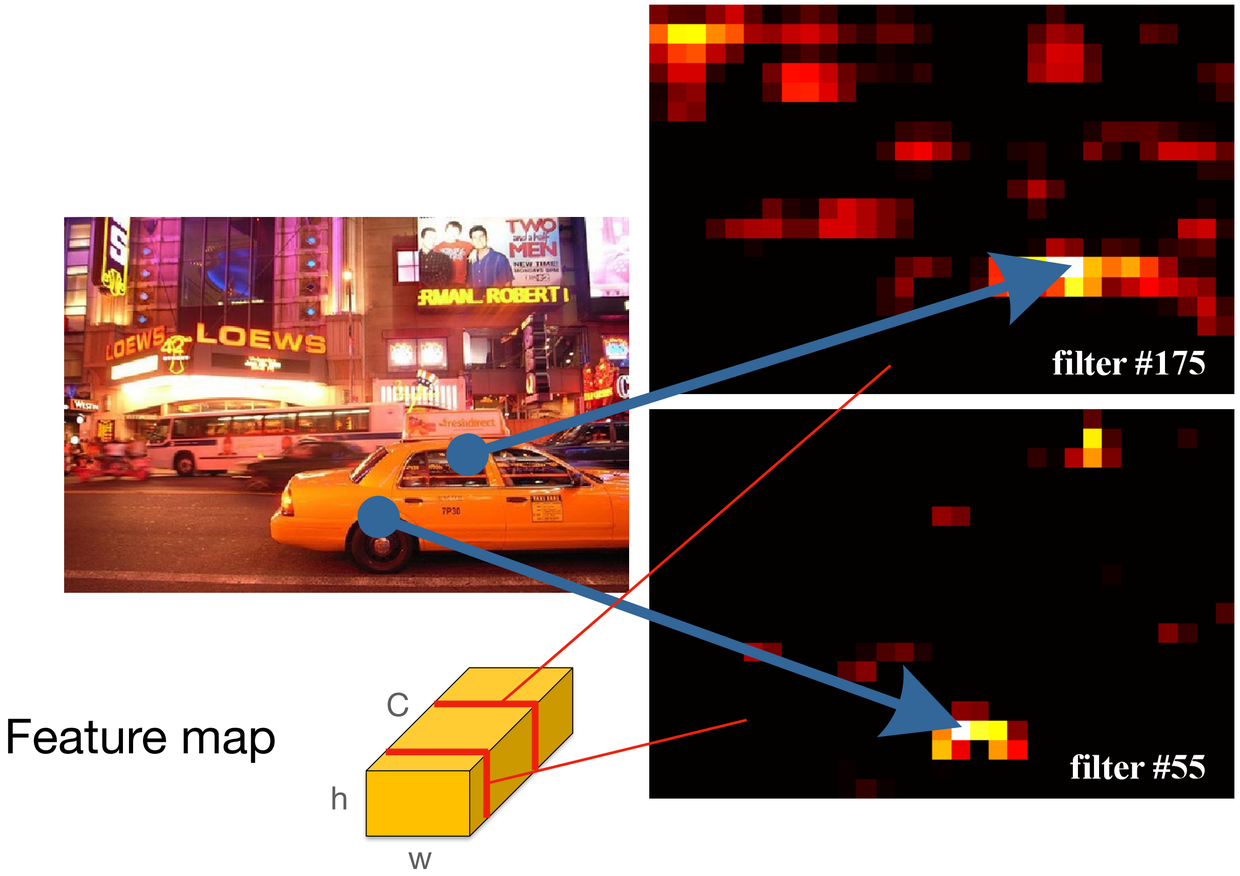
crop

warp



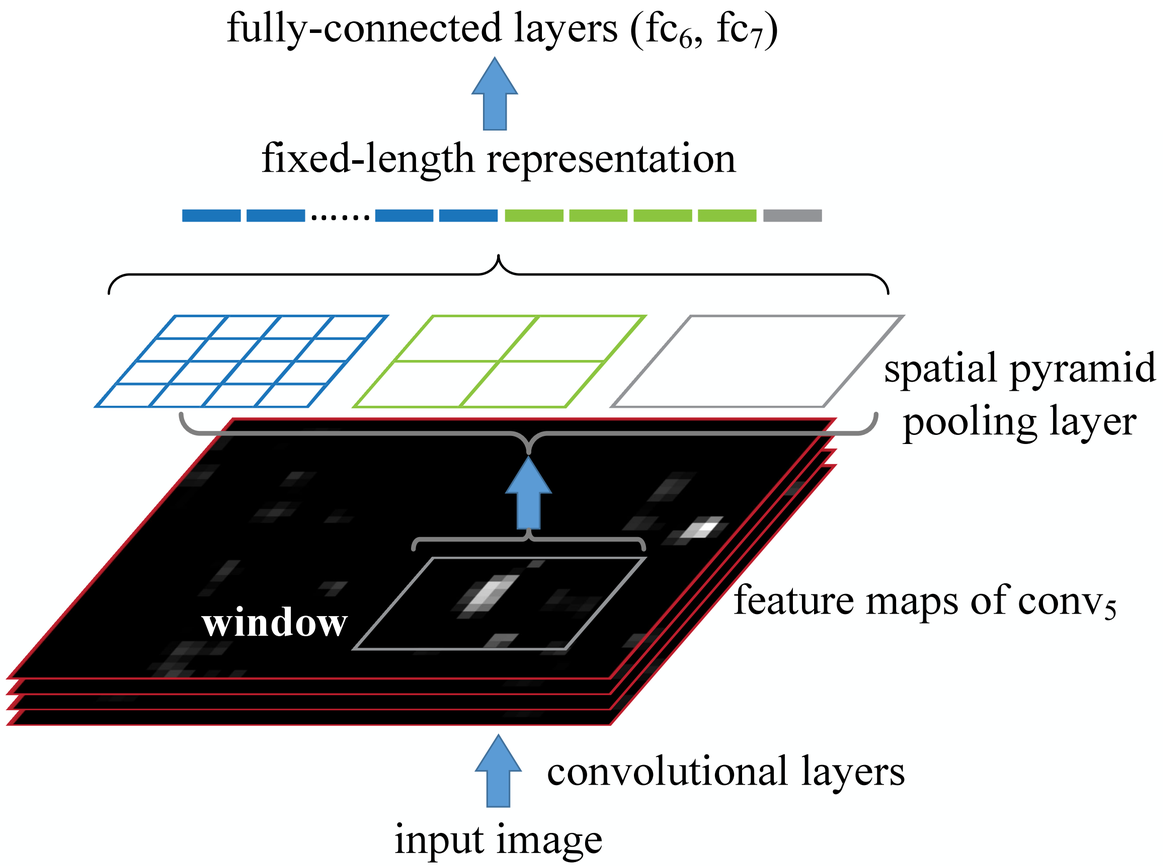
He et al. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. ECCV 2014.

Object detection with deep nets



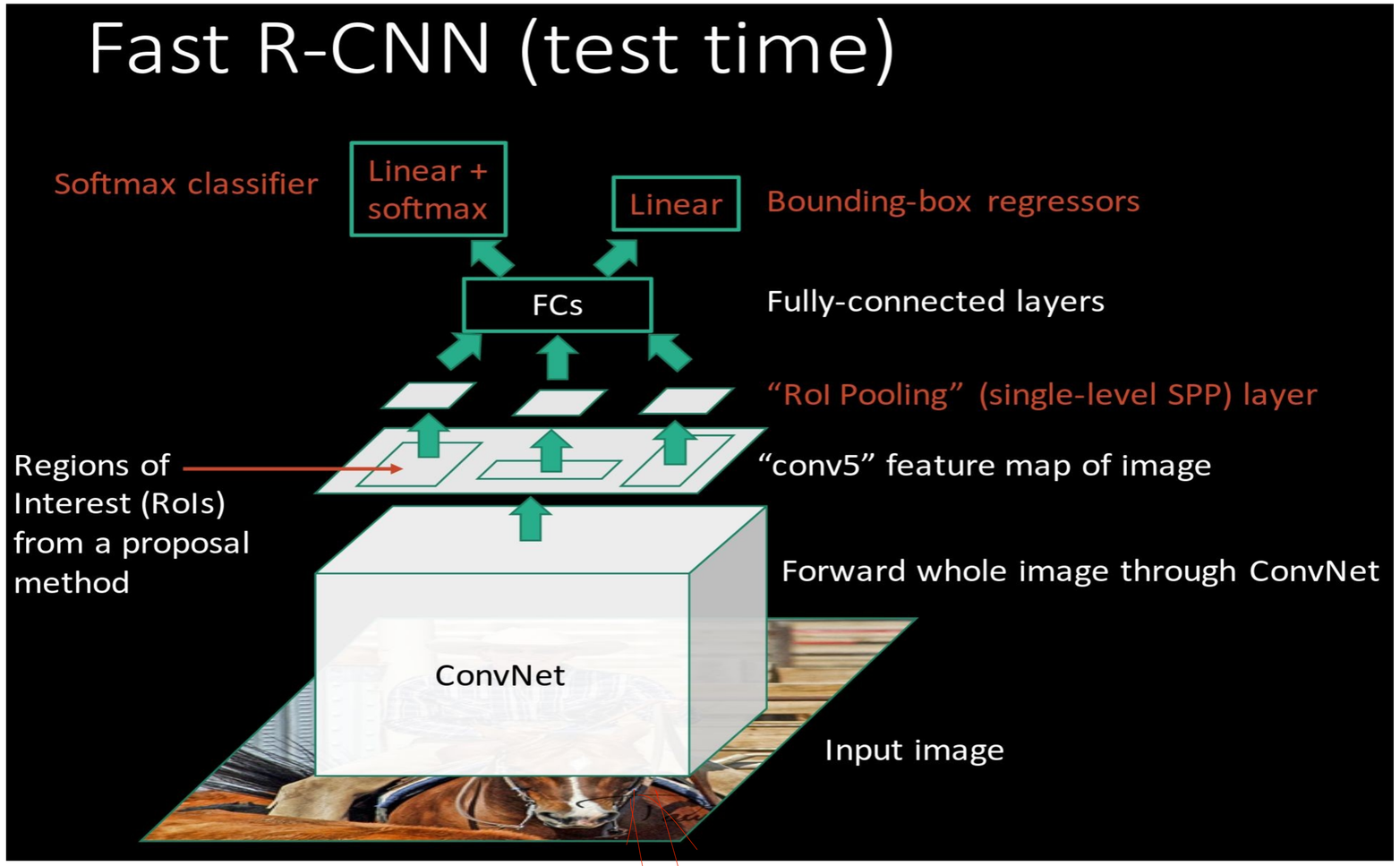
He et al. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. ECCV 2014.

SPP-Net: Spatial Pyramid Pooling



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

Fast R-CNN

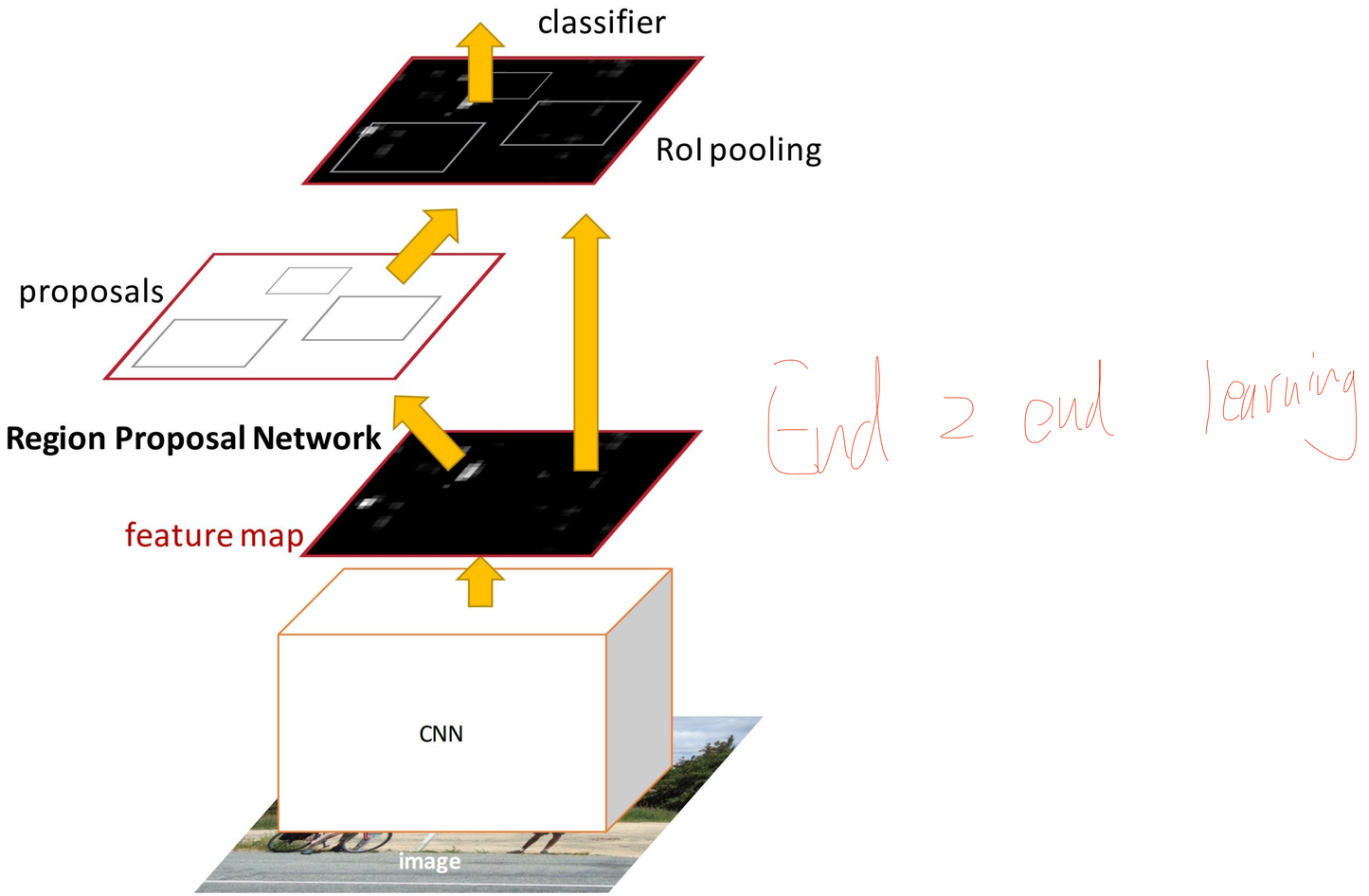


Shared computation at test time (like SPP)

Girschick, "Fast R-CNN", ICCV 2015

Slide credit: Ross Girschick

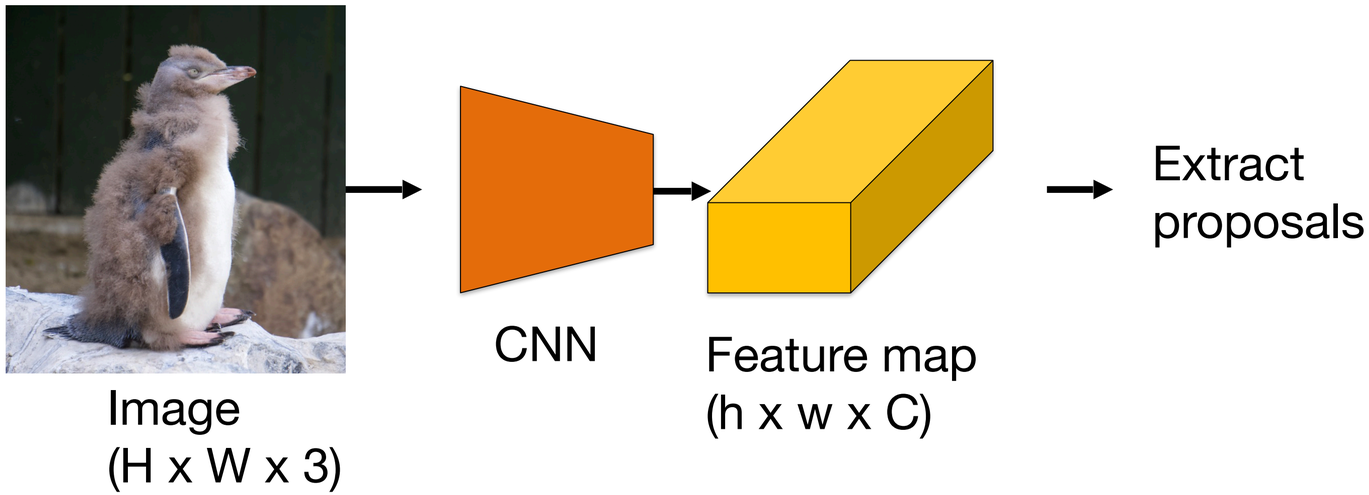
Faster R-CNN



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

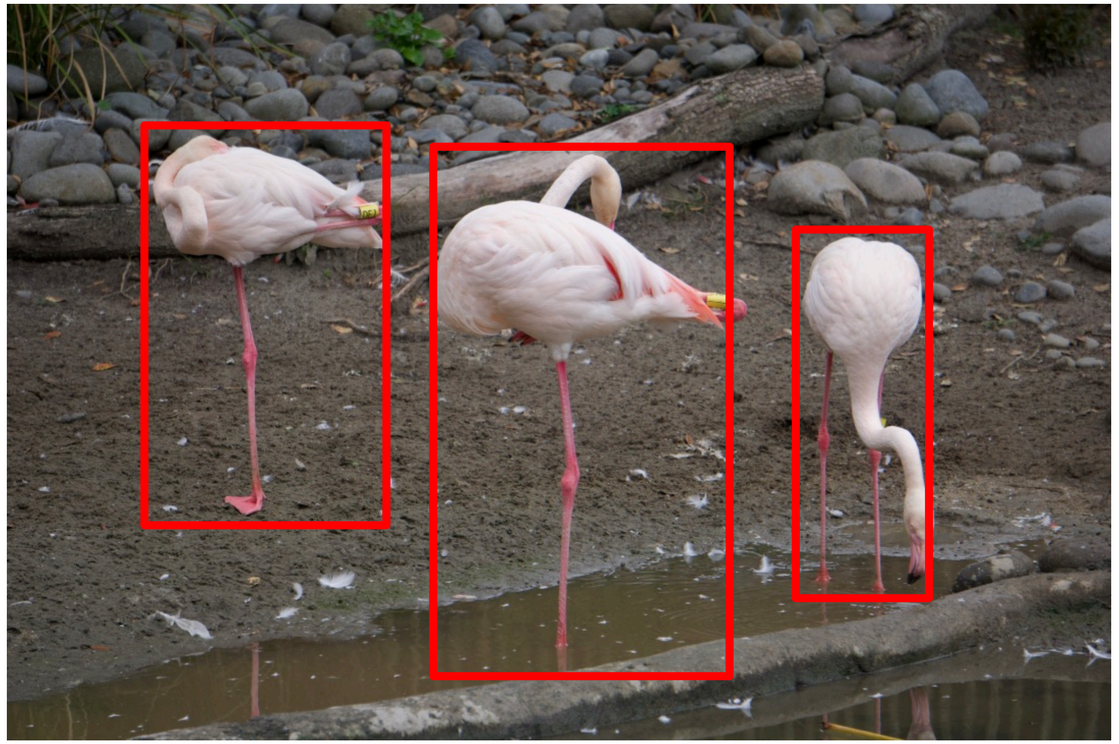
Slide credit: Ross Girschick

Region Proposal Network



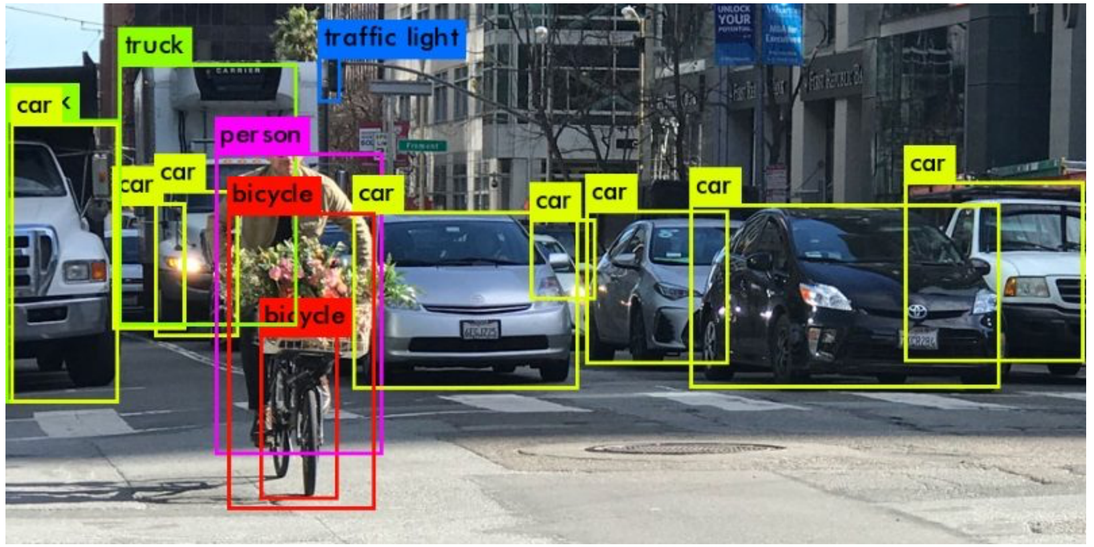
- How to place the proposals? Uniformly, densely.
- How many proposals? N per each feature location.

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)

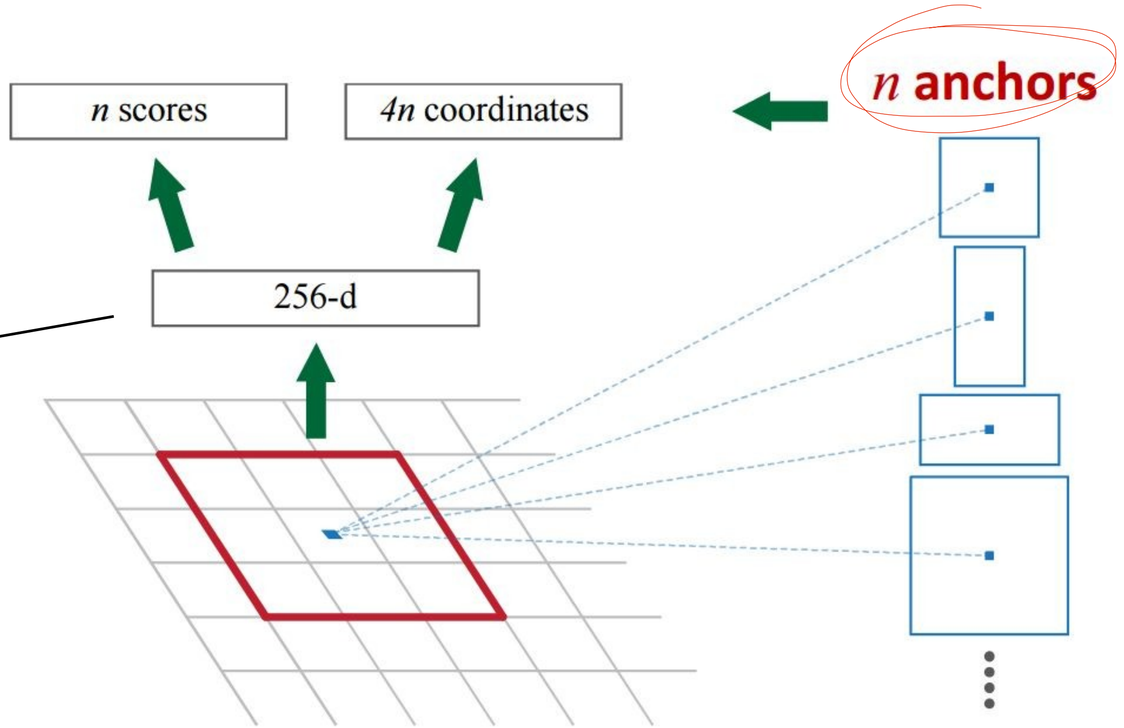
- RPN: Place multiple proposals uniformly densely
  - Learn to predict confidence for each proposal

Region Proposal Network

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

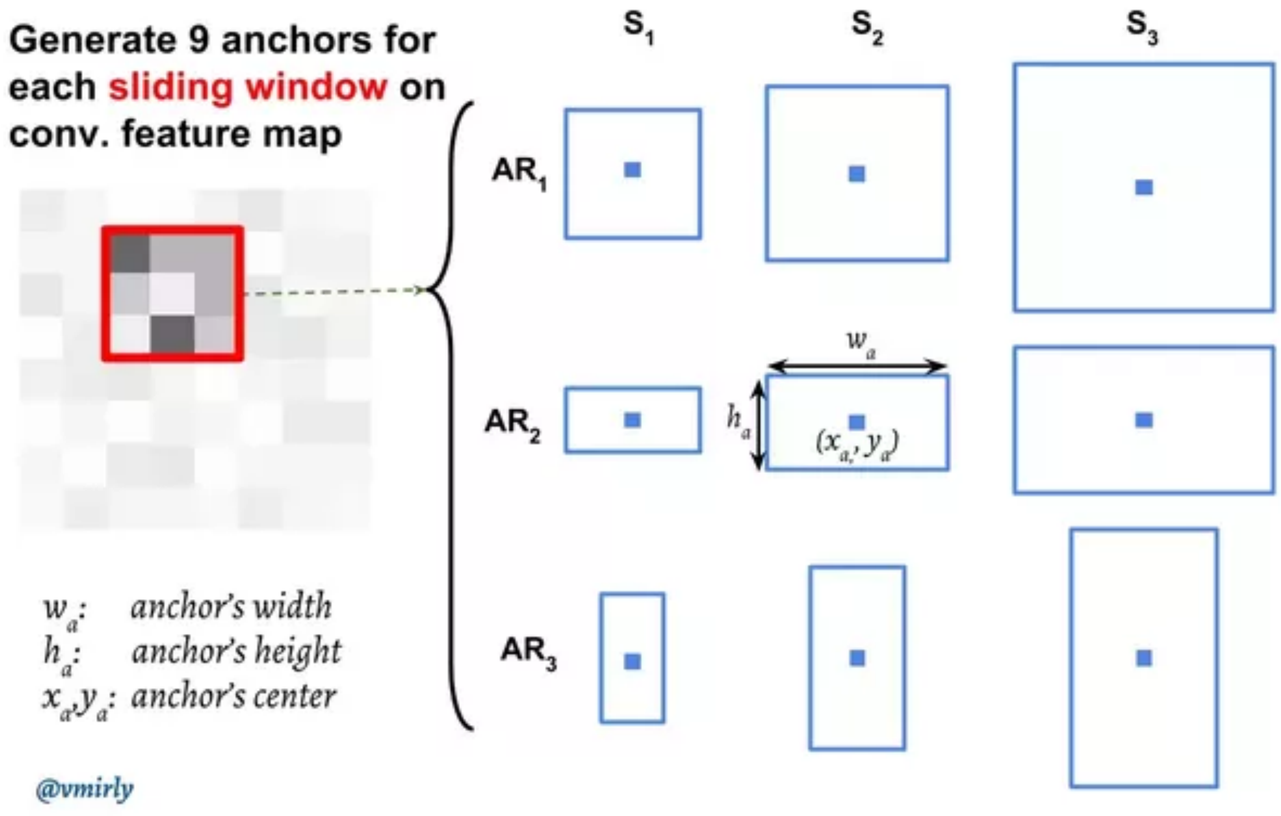


Ren et al, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, NIPS 2015

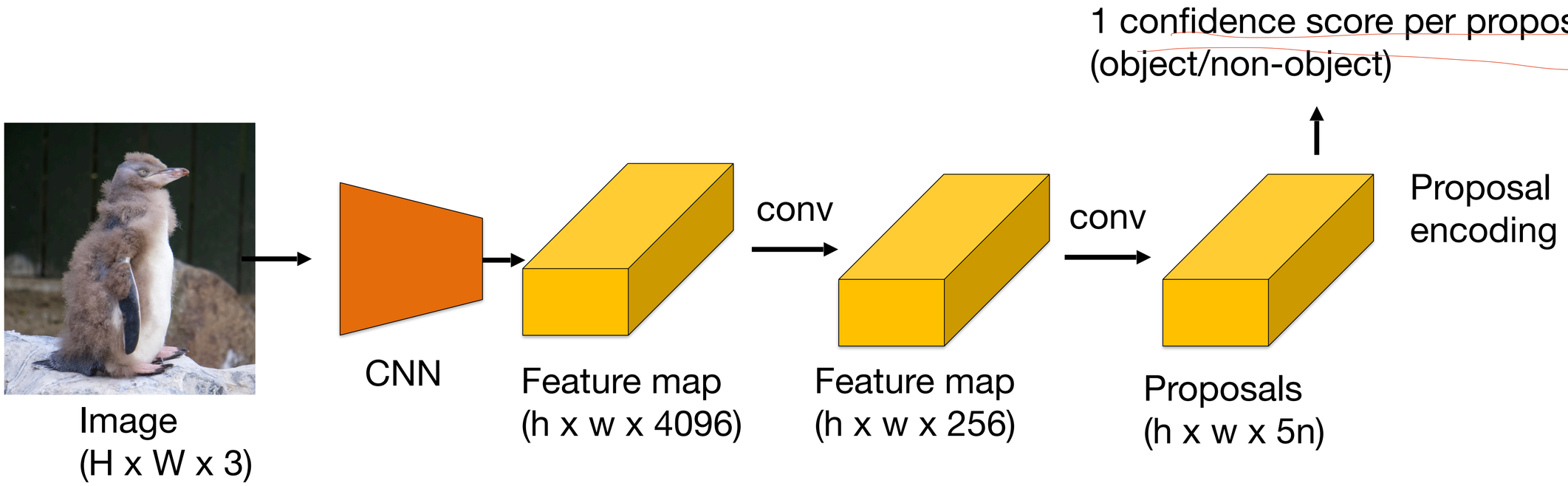
Slide credit: Ross Girschick

RPN: Training

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



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.
- For IoU in [0.3, 0.7], ignore the anchor in 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:

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

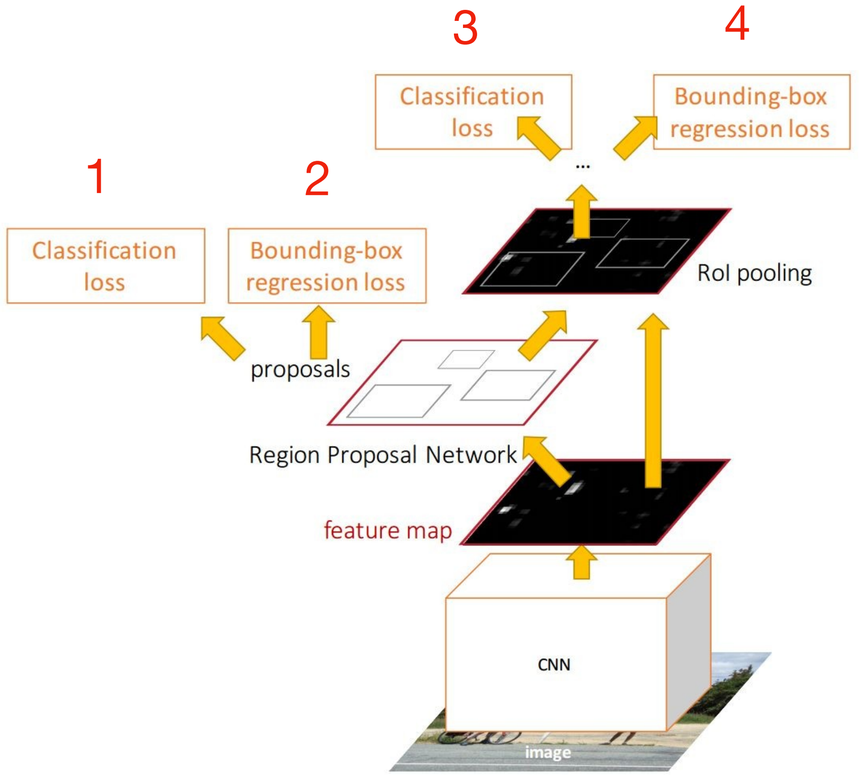
$c \in \{0,1\}$ ground truth; $p \in [0,1]$ model prediction.

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

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)



Addressing scale variance

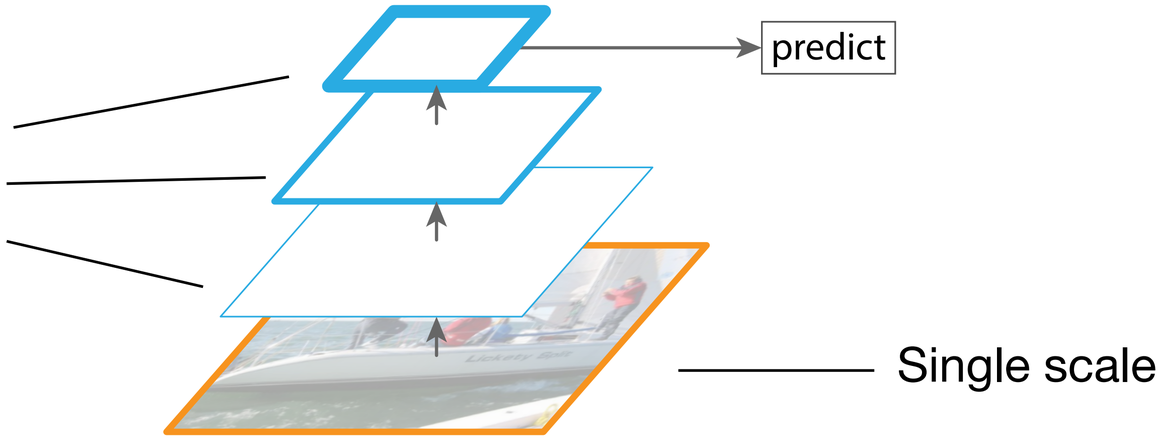
- CNNs are not scale-invariant
- Problem: Object scale can vary drastically
- Proposals help only to some extent:
  - because the features are “canonicalised” in RoI pooling

Credit: MS-COCO

Addressing scale variance

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

Features are progressively downsampled in CNNs

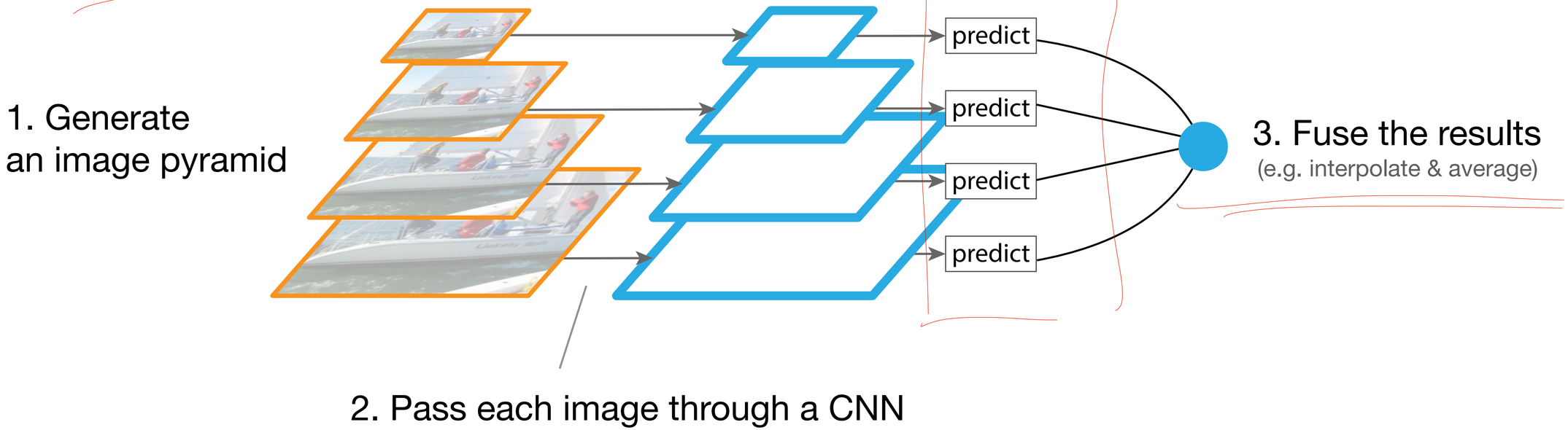


➡ Make CNNs for object detection more robust to scale changes

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

Addressing scale variance

- Idea A: featurised image hierarchy

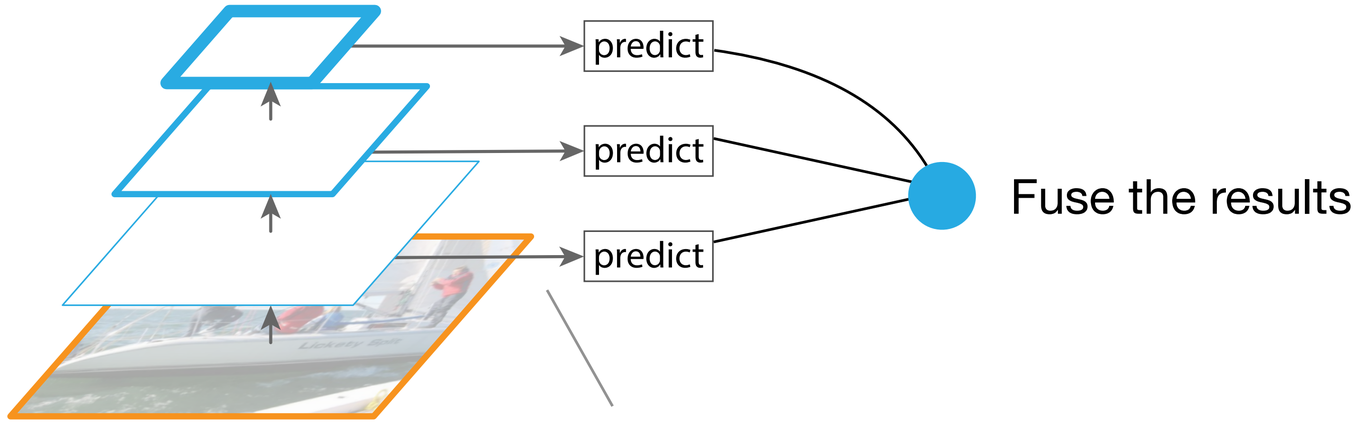


- Pros: Typically boosts accuracy (esp. at test time)
- Cons: Computationally inefficient

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

Addressing scale variance

- Idea B: pyramidal feature hierarchy



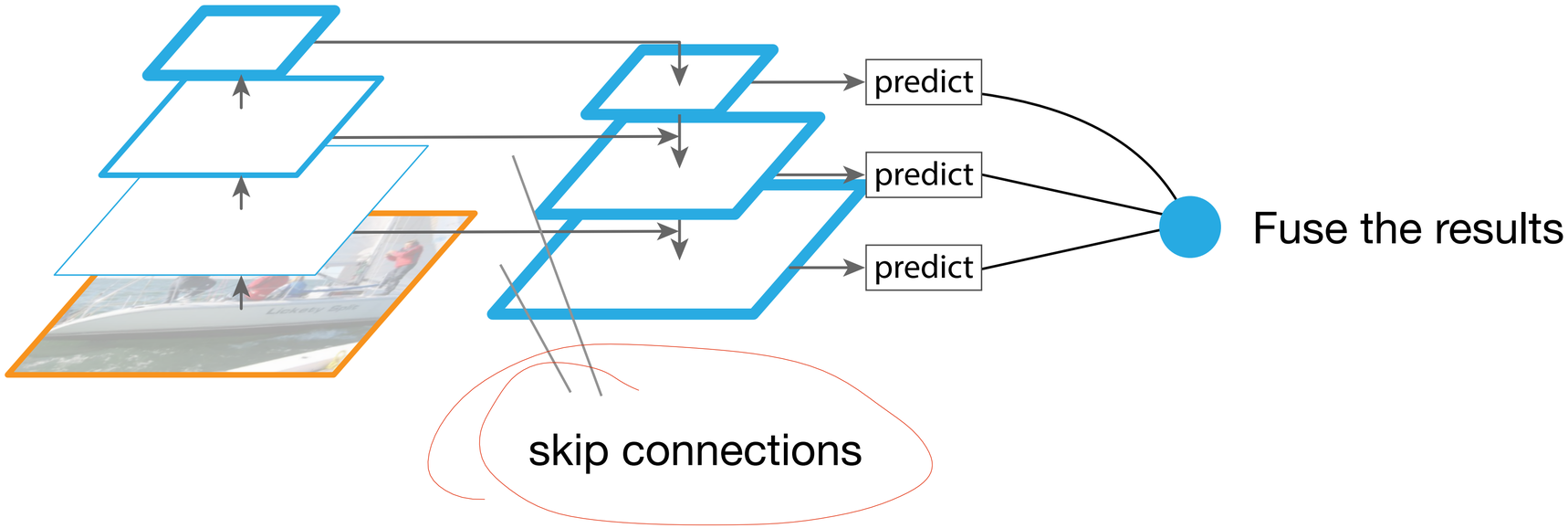
Supervise at every feature scale
(aka “deep supervision”)

- 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

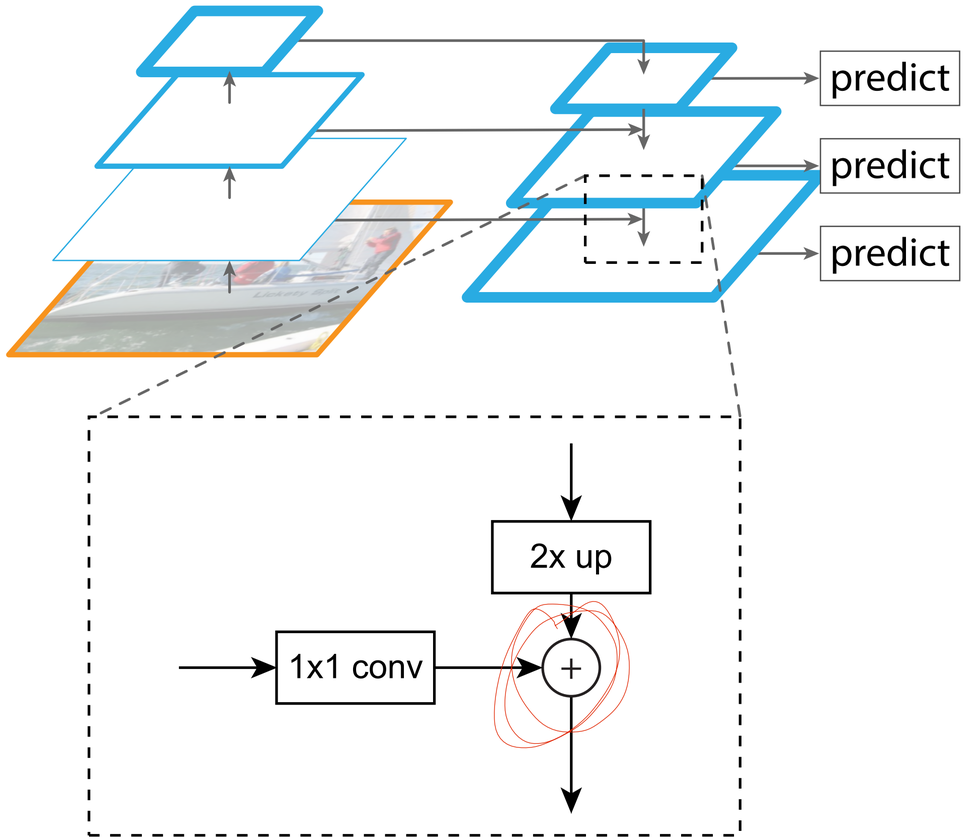


- Higher scales benefit from deeper representation from lower scales
- Efficient and high accuracy

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

Feature Pyramid Network (FPN)

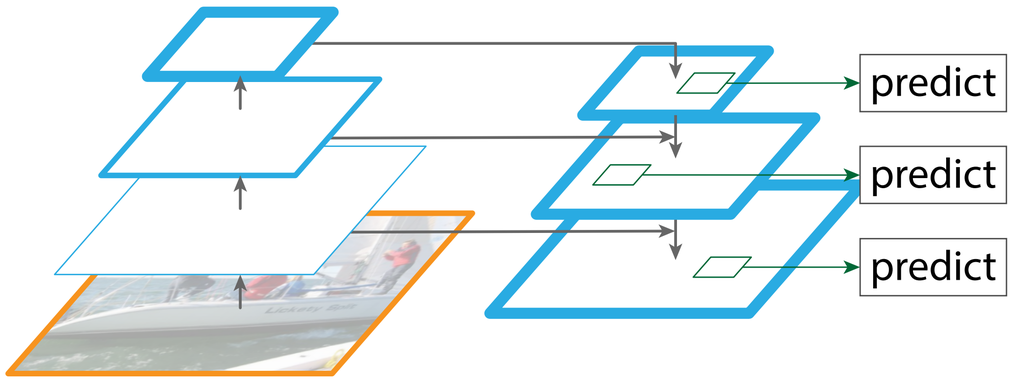
- Straightforward implementation:
  - convolution with 1x1 kernel
  - upsampling (nearest neighbours)
  - element-wise addition



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?

large objects — high scale

small objects — low scale

?

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

Feature Pyramid Network (FPN)

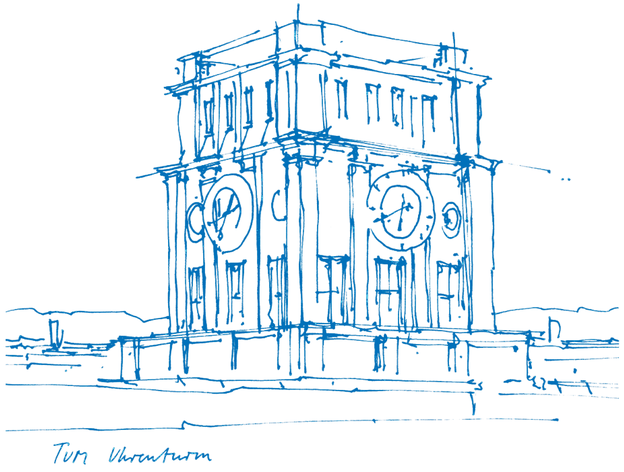
- Pros:
  - improves recall across all scales (esp. small objects);
  - more accurate (in terms of AP);
  - broadly applicable, also for one-stage detectors (next);
  - still in wide use today.
- Cons:
  - increased model complexity.

Computer Vision III:

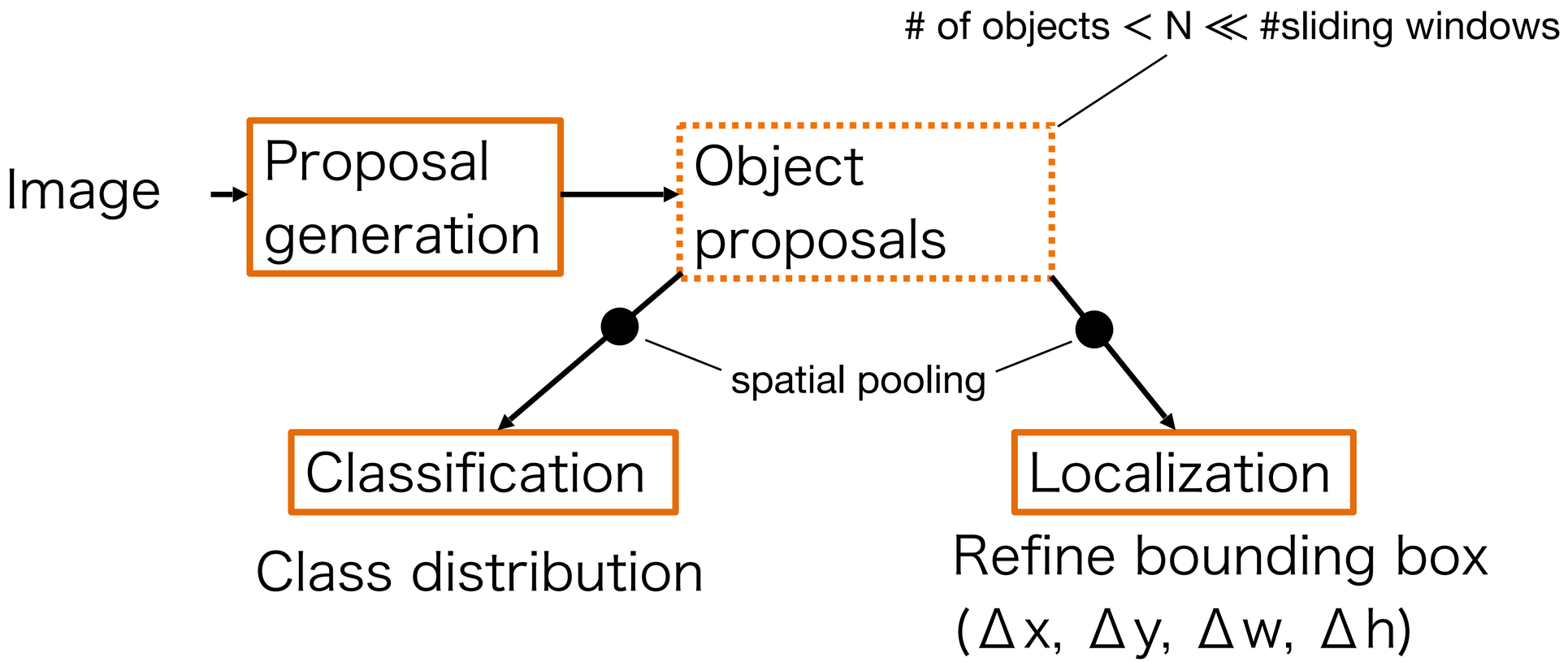
One-stage object detectors

Dr. Nikita Araslanov
31.10.2023

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

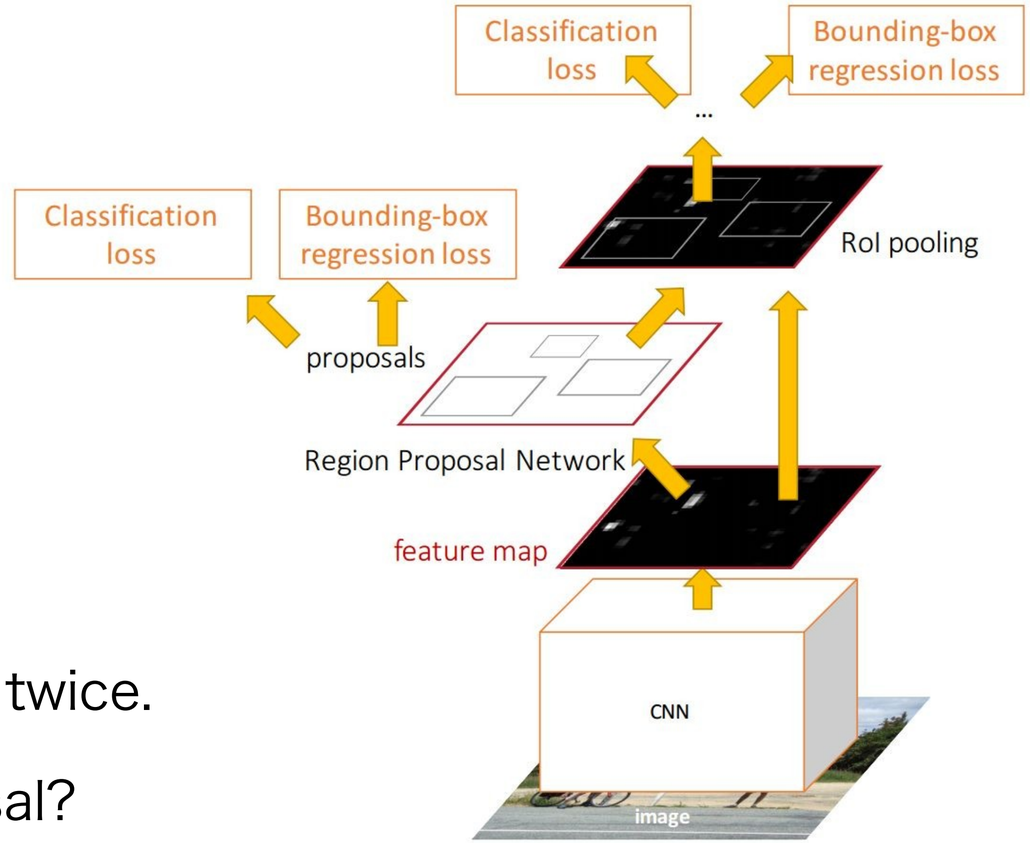


Two-stage detectors



Recall: Faster R-CNN

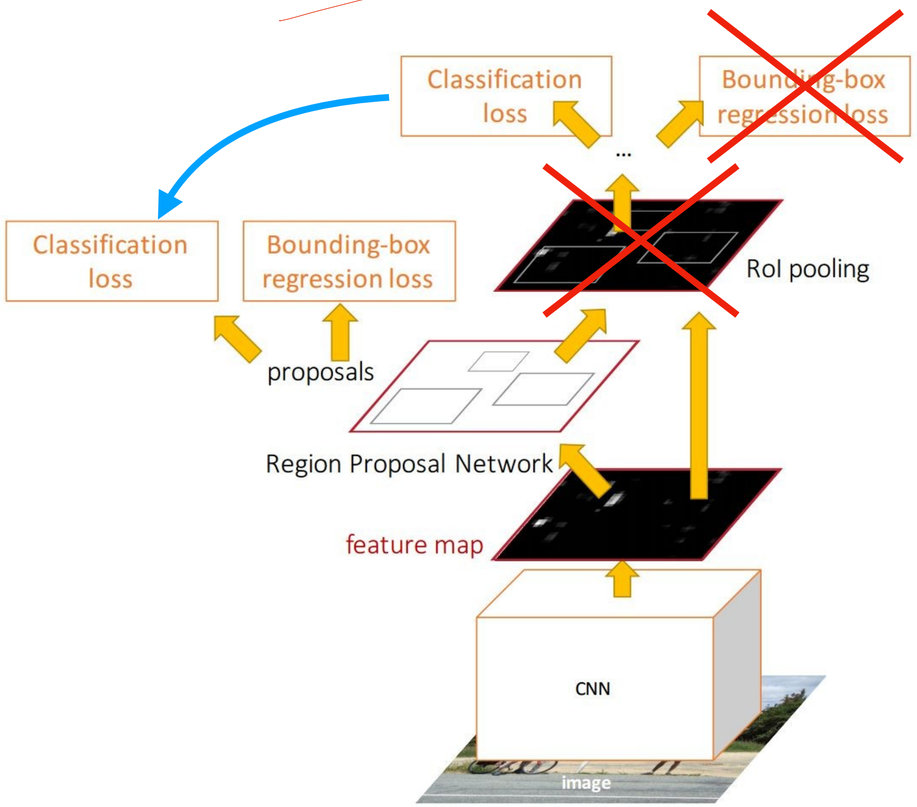
- Four losses:
  1. RPN classification (object/non-object)
  2. RPN regression (anchor → proposal)
  3. Fast R-CNN classification (type of object)
  4. Fast R-CNN regression (proposal → box)



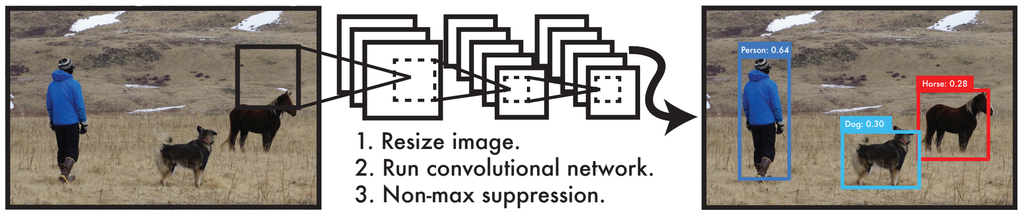
- Conceptually, we seem to be doing the same thing twice.
- Why not already predict the class for each proposal?

One-stage detector: YOLO

What if we remove pooling?



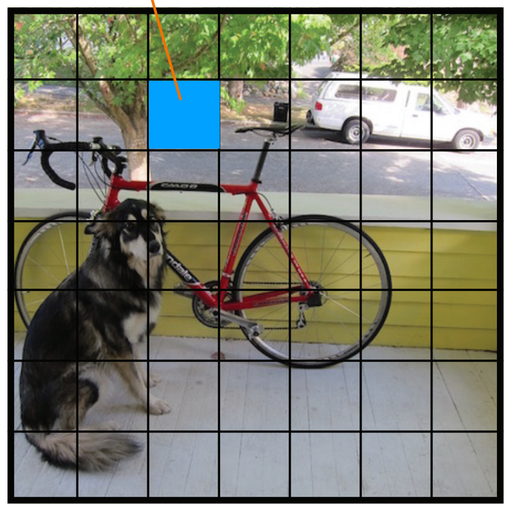
A one-stage detector



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.



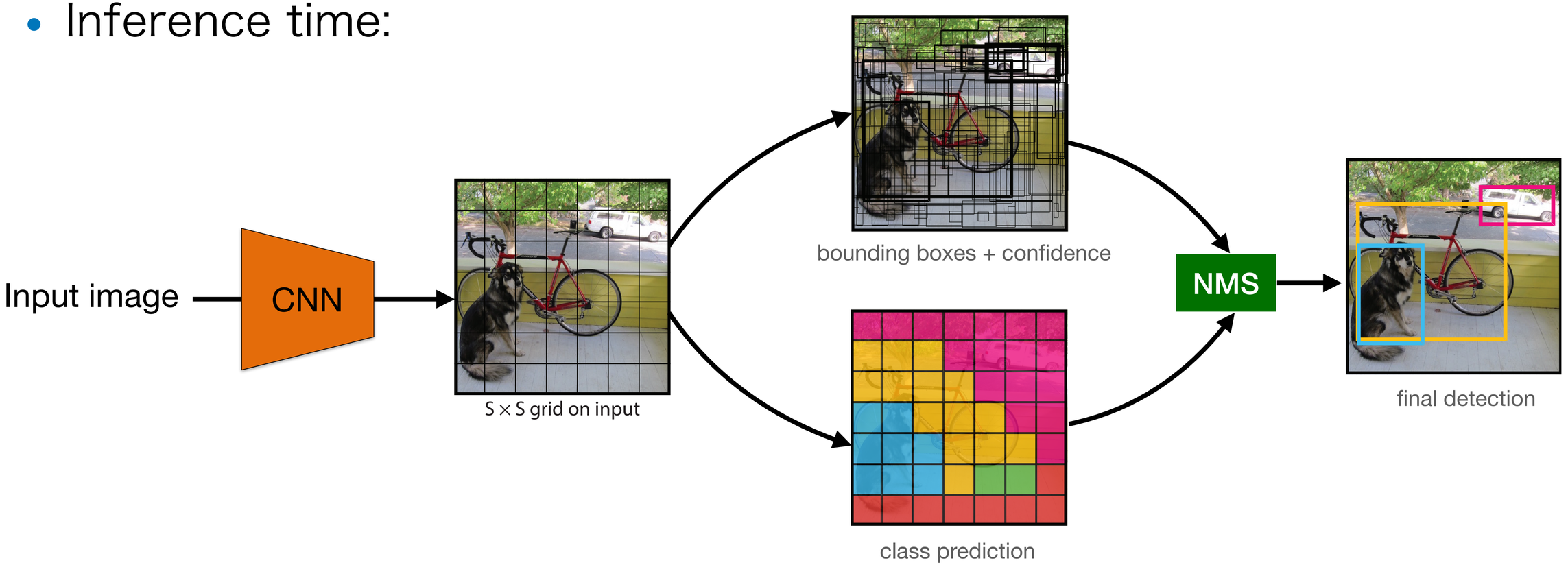
Quiz: What is the dimensionality of the output?

$B \times [C + 5]$

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

YOLO: You Only Look Once

- Inference time:



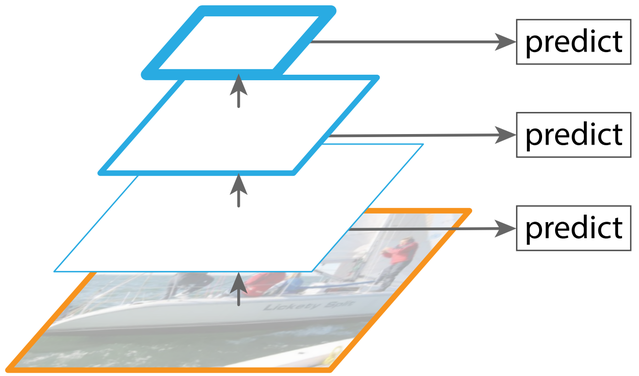
Redmon et al, “You only look once: Unified real-time object detection”, CVPR 2016.

YOLO: You Only Look Once

- 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).

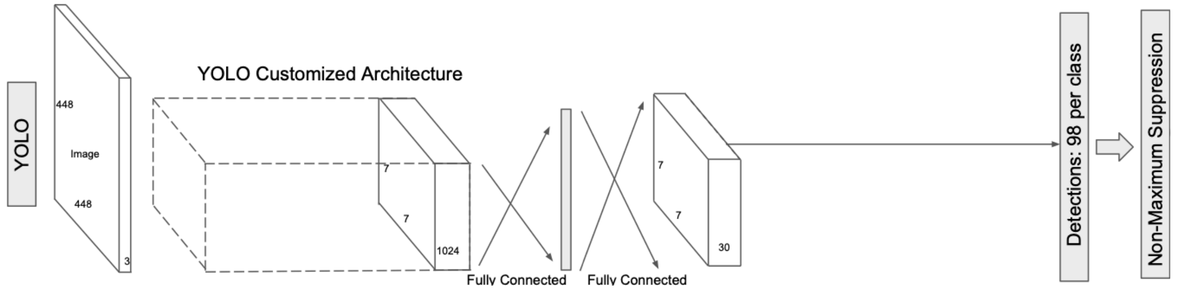
网格分辨率较低，每个单元的锚点较少--处理小型物体时会出现问题；

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



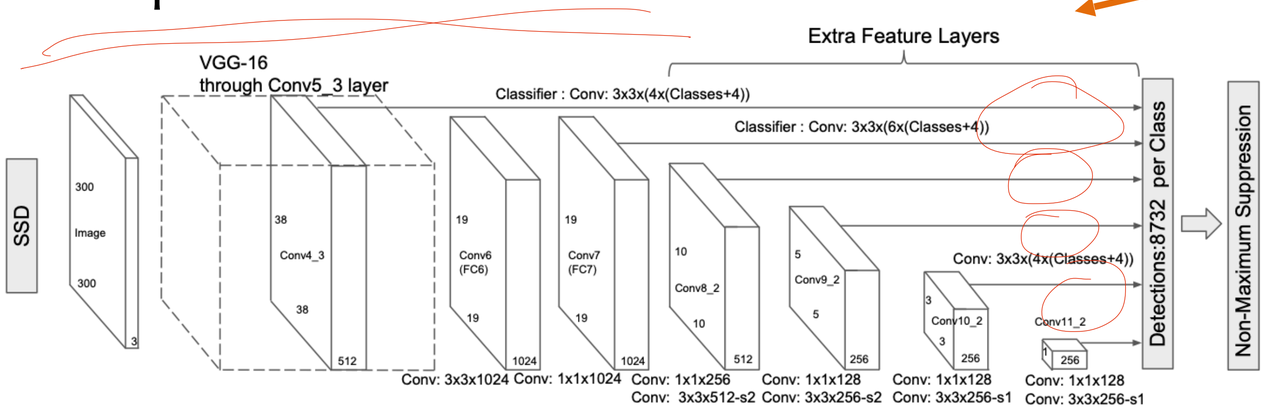
SSD: Single Shot multibox Detector

- 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.

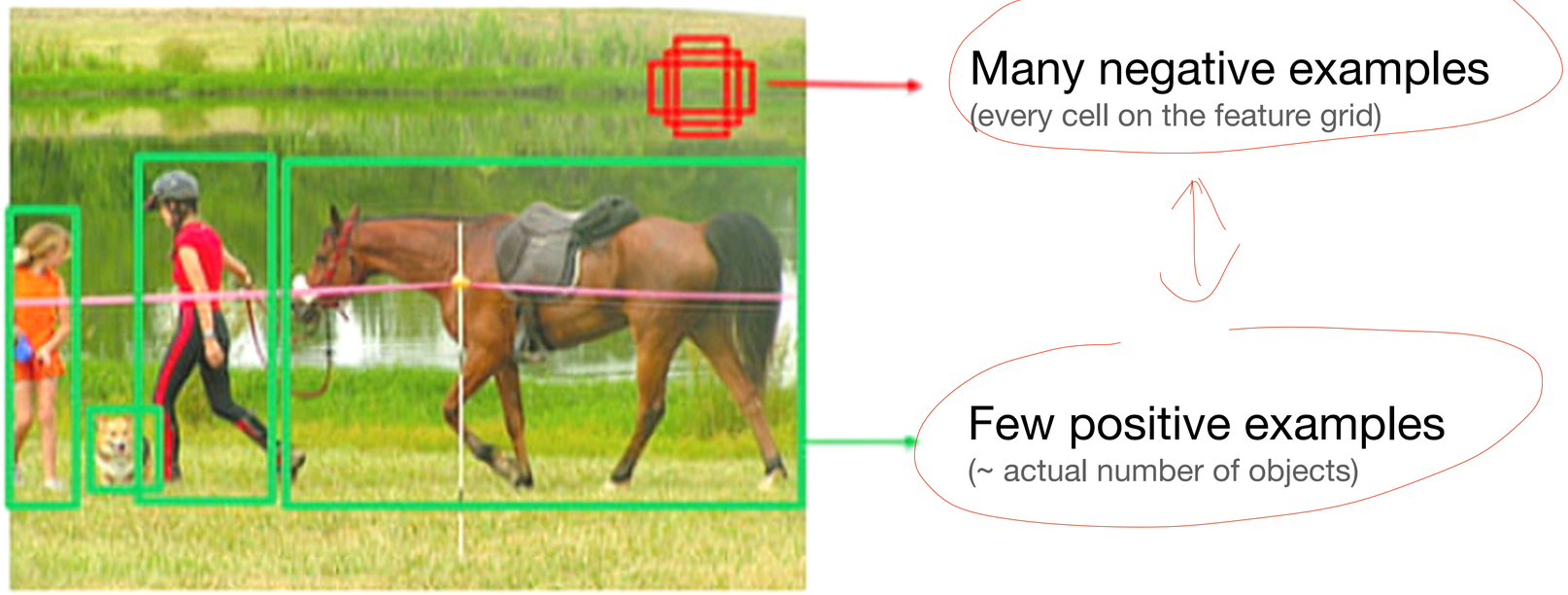
- Cons:
  - still lags behind two-stage detectors;
    - data augmentation is still crucial (esp. random scaling);
  - a bit more complex (due to multi-scale features).

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?

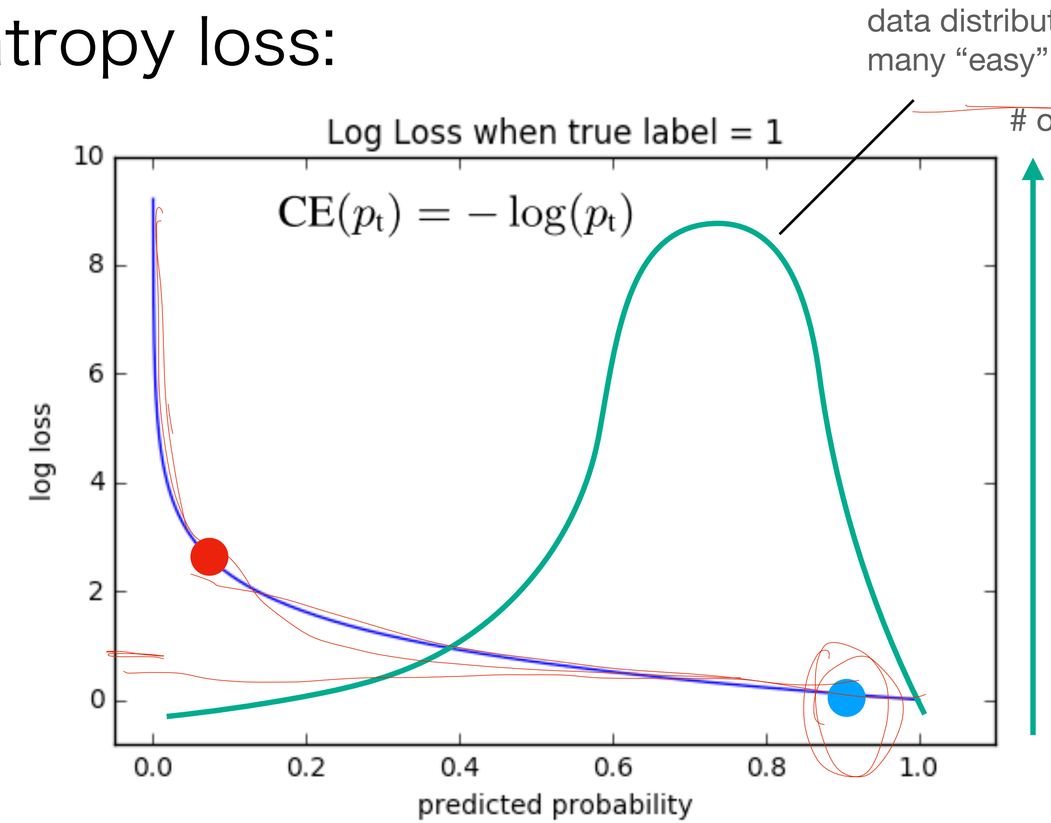
- Many locations need to be analysed (100k) densely covering the image – foreground-background imbalance



- 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:



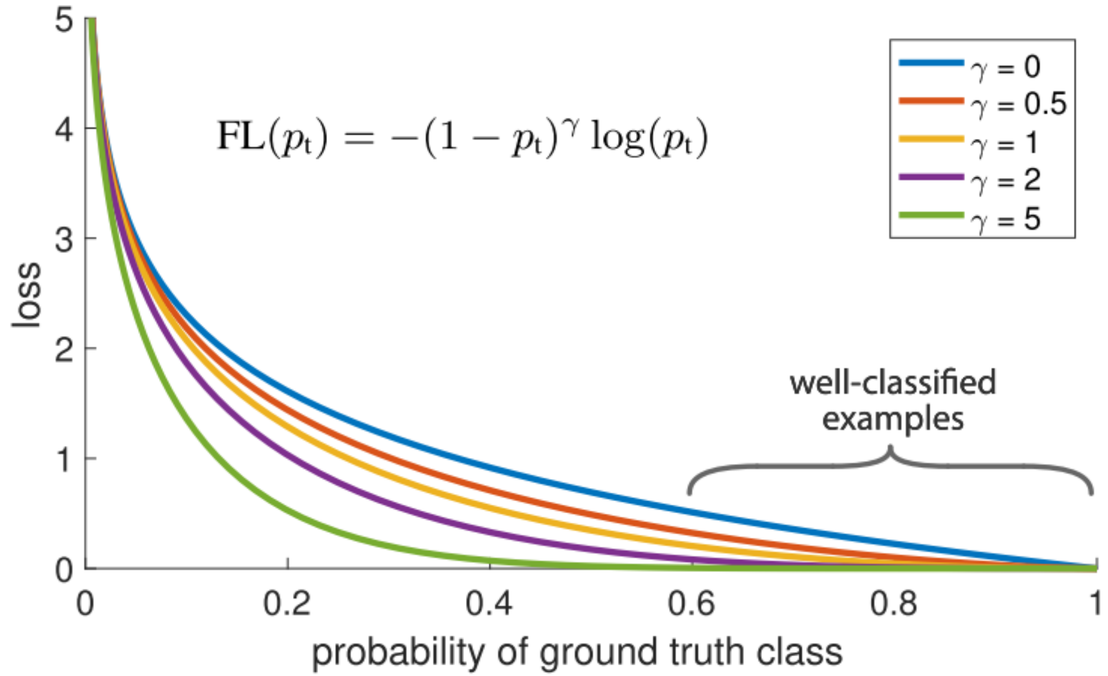
- Very hard example, e.g. $p_t \approx 0.1$, we get loss ≈ 2.3
- An easy example, e.g. $p_t \approx 0.9$, we get loss ≈ 0.1

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):



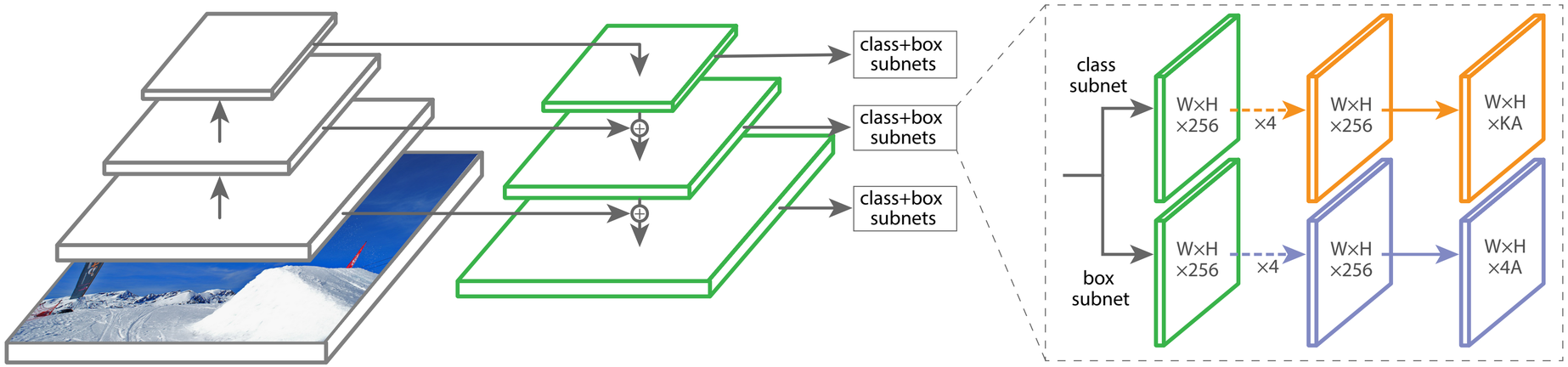
- When $\gamma = 0$, it is equivalent to the cross-entropy loss.
- As γ goes towards 1, the easy examples are down-weighted.
- Example: $\gamma = 2$, if $p_t = 0.9$, FL is $\times 100$ lower than CE.

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

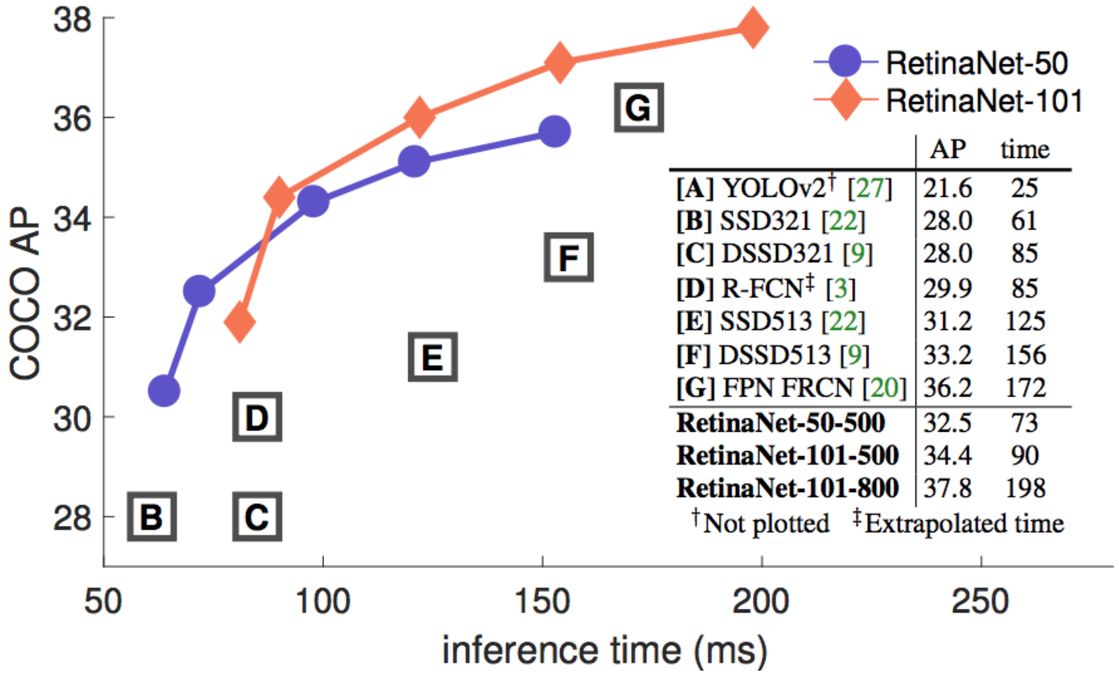
- One-stage (like YOLO and SSD) with Focal Loss
- Feature extraction with ResNet
- Multi-scale prediction – now with FPN



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

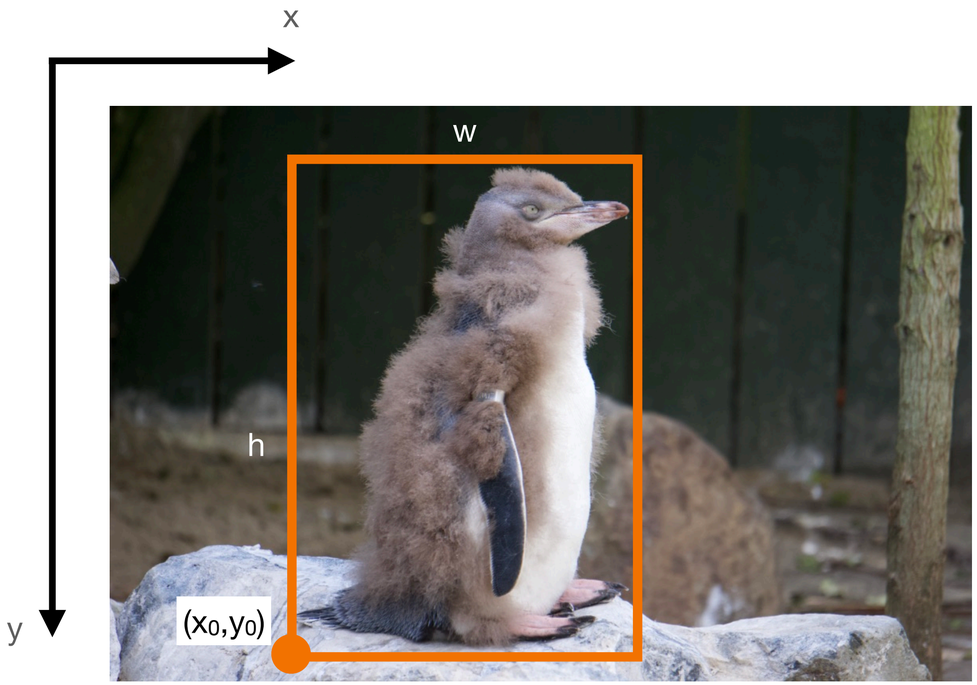
Summary: Two-stage vs one-stage

- 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.

Object detection: advanced topic(s)

Box representation

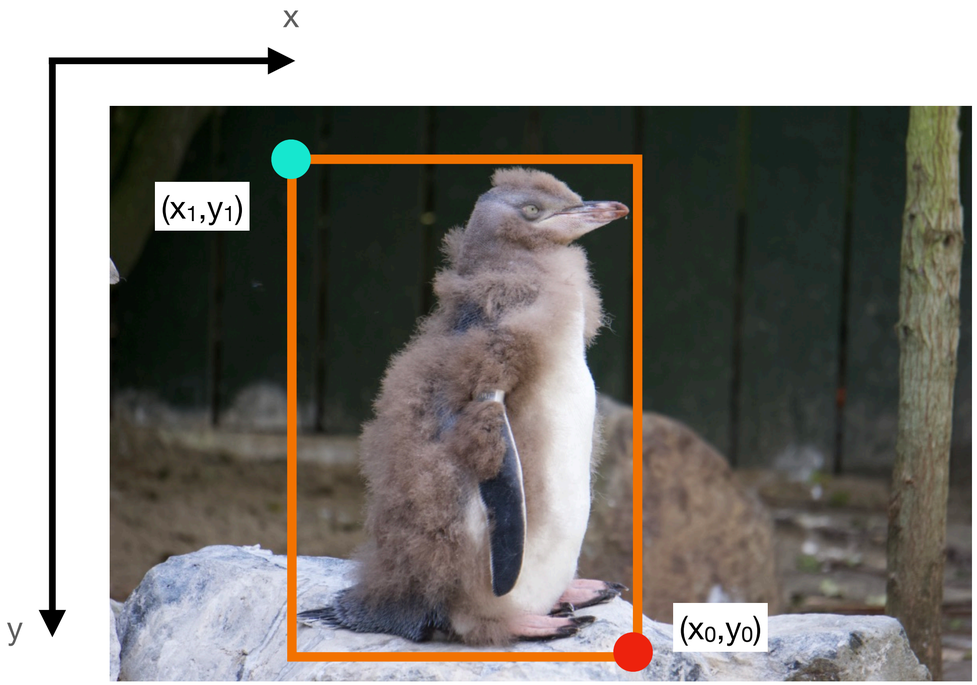
- There are many ways to define a bounding box



- (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

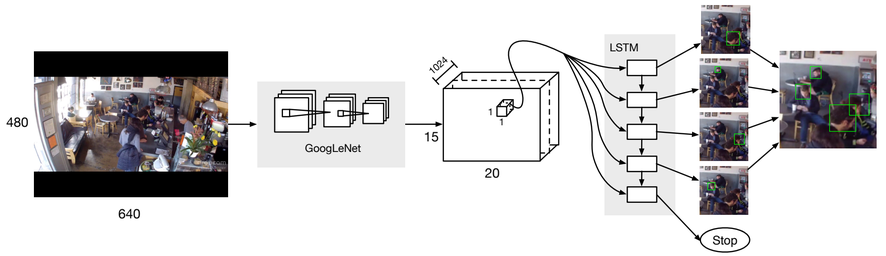
Heatmap A (top-left corner)

Heatmap B (bottom-right corner)

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

Bounding box representation

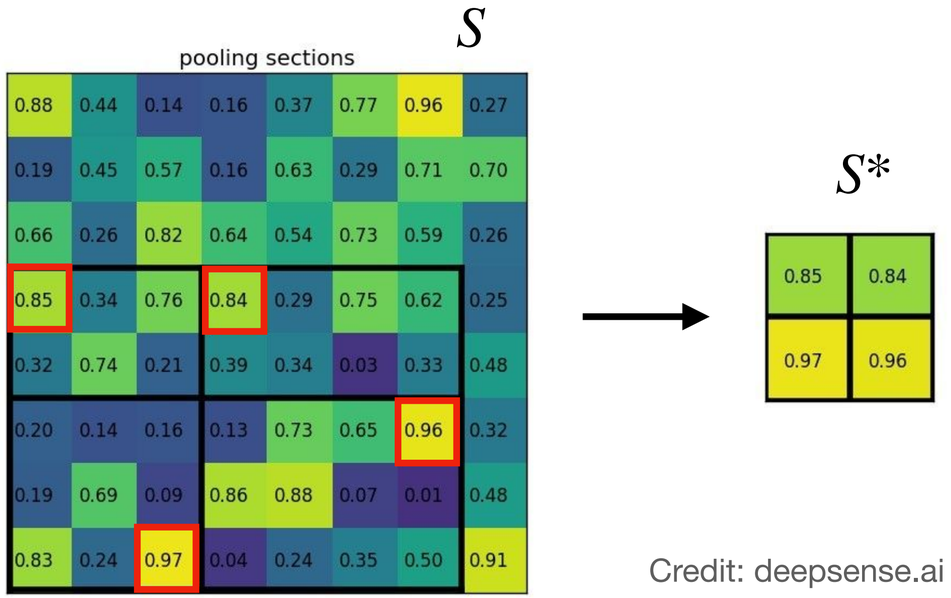
- Additional reading:
  - Law and Deng, “CornerNet: Detecting Objects as Paired Keypoints“ (ECCV 2018).
  - K. Duan et al. „**CenterNet**: Keypoint Triplets for Object Detection“ (ICCV 2019).
  - Sequential detection: Stewart et al., “End-to-End People Detection in Crowded Scenes” (CVPR 2016)



Recall: Spatial Pooling

- Is it differentiable?

- S^* is differentiable w.r.t. S ?
  - yes and no – depends on pooling
- S^* is differentiable w.r.t. (x, y, h, w) ?
  - no

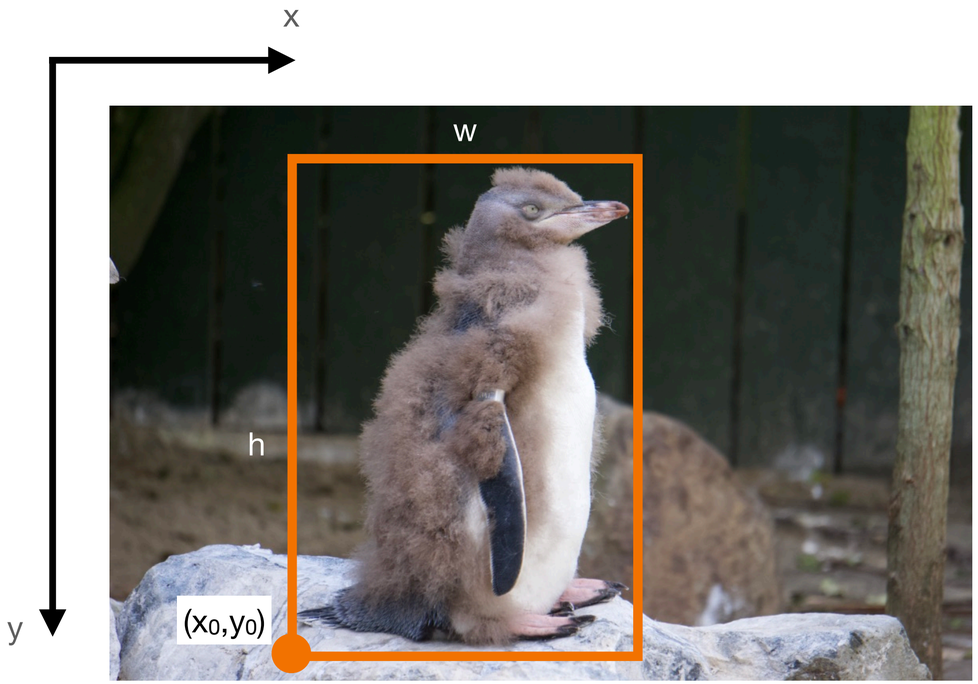


Max-pooling within each window

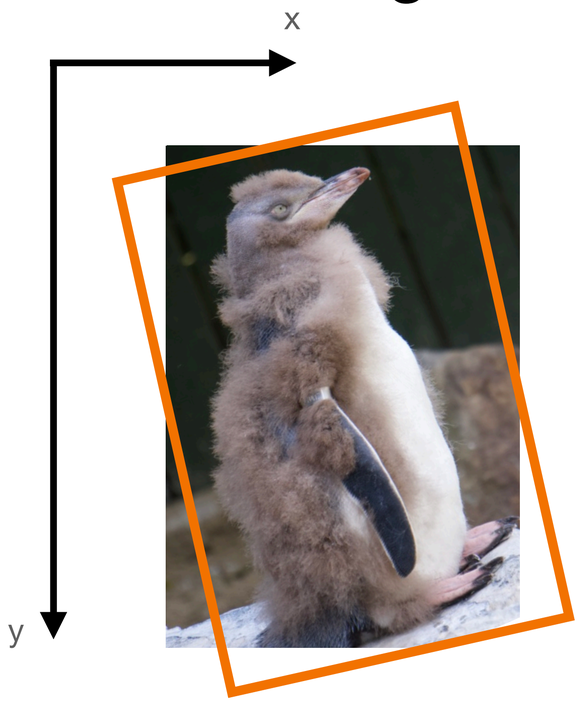
Credit: deepsense.ai

Dataset bias

Training



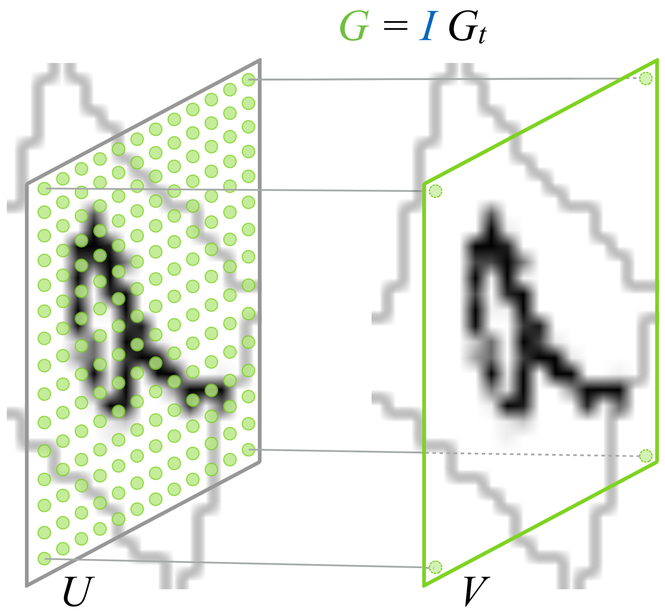
Testing

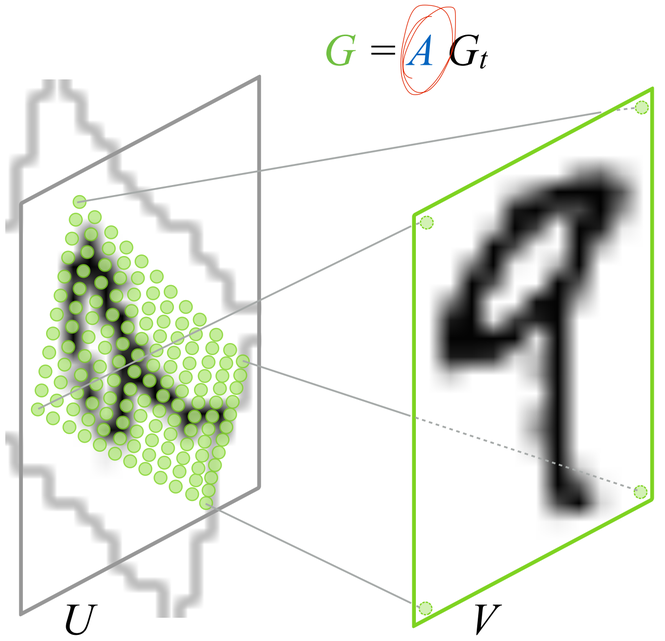


We need equivariance!

$$f(A(x)) = A(f(x))$$

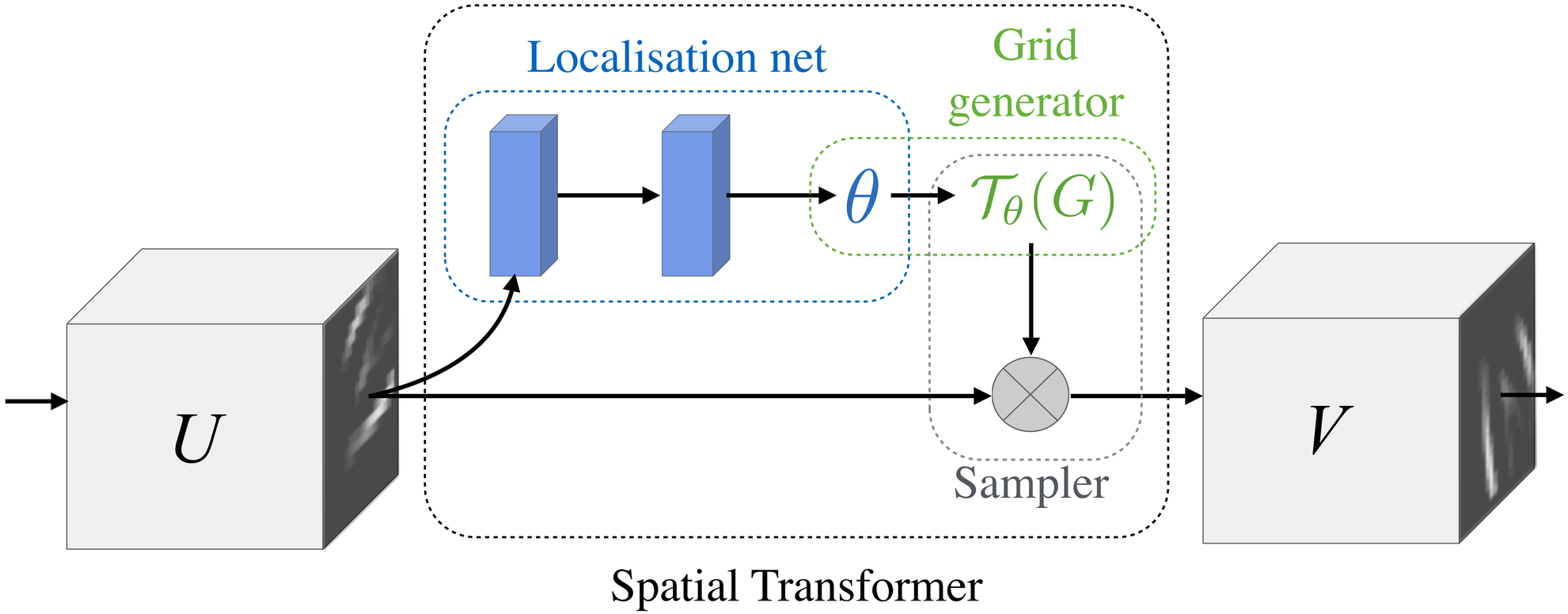
Spatial transformers





Jaderberg et al., “Spatial Transformer Networks”. NIPS 2015

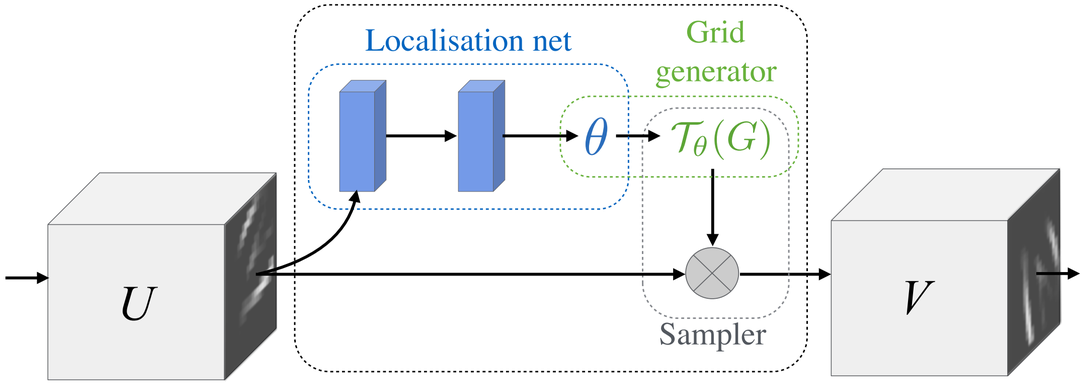
Spatial transformers



Jaderberg et al., “Spatial Transformer Networks”. NIPS 2015

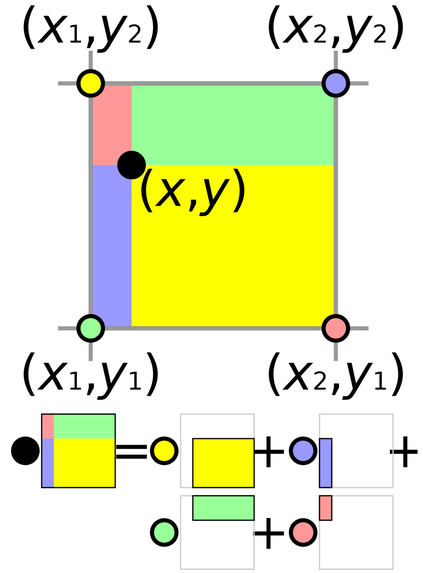
Spatial transformers

- Learn to predict the parameters of the mapping θ:



$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \mathcal{T}_\theta(G_i) = \mathtt{A}_\theta \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

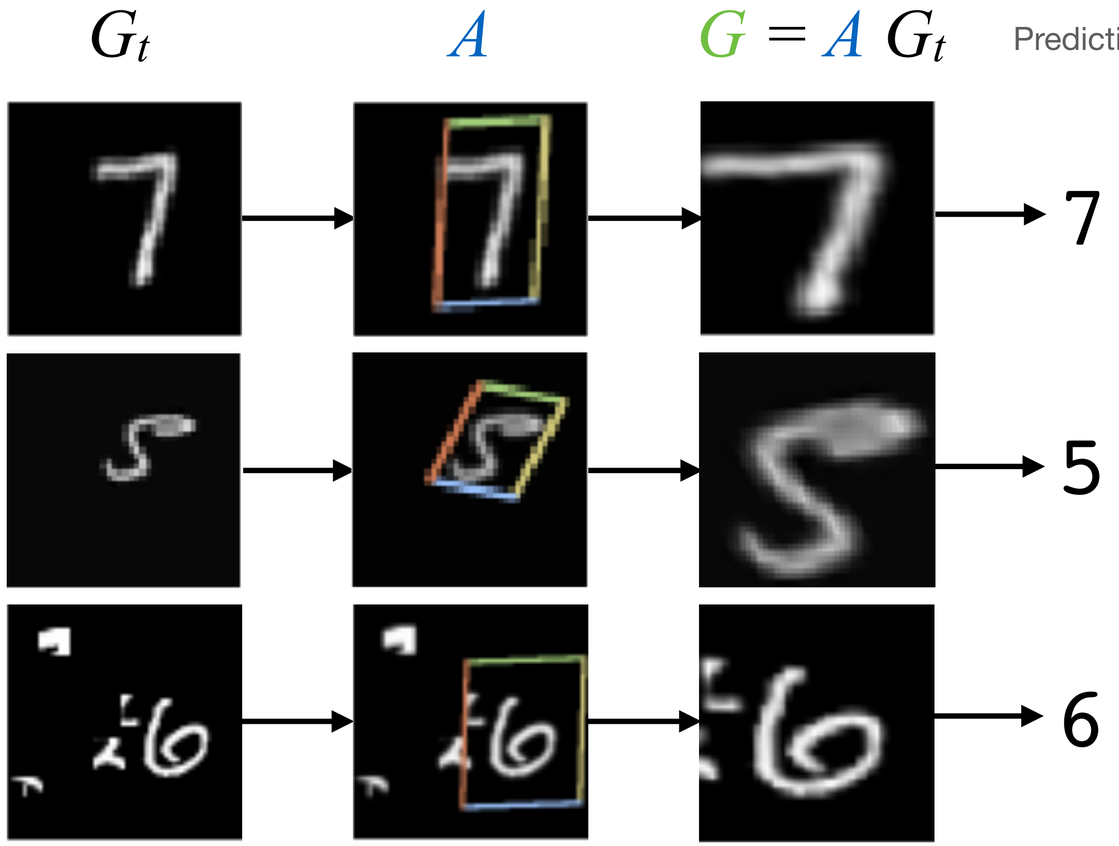
Sampler: bilinear interpolation



Credit: Cmglee (Wikipedia)

Jaderberg et al., “Spatial Transformer Networks”. NIPS 2015

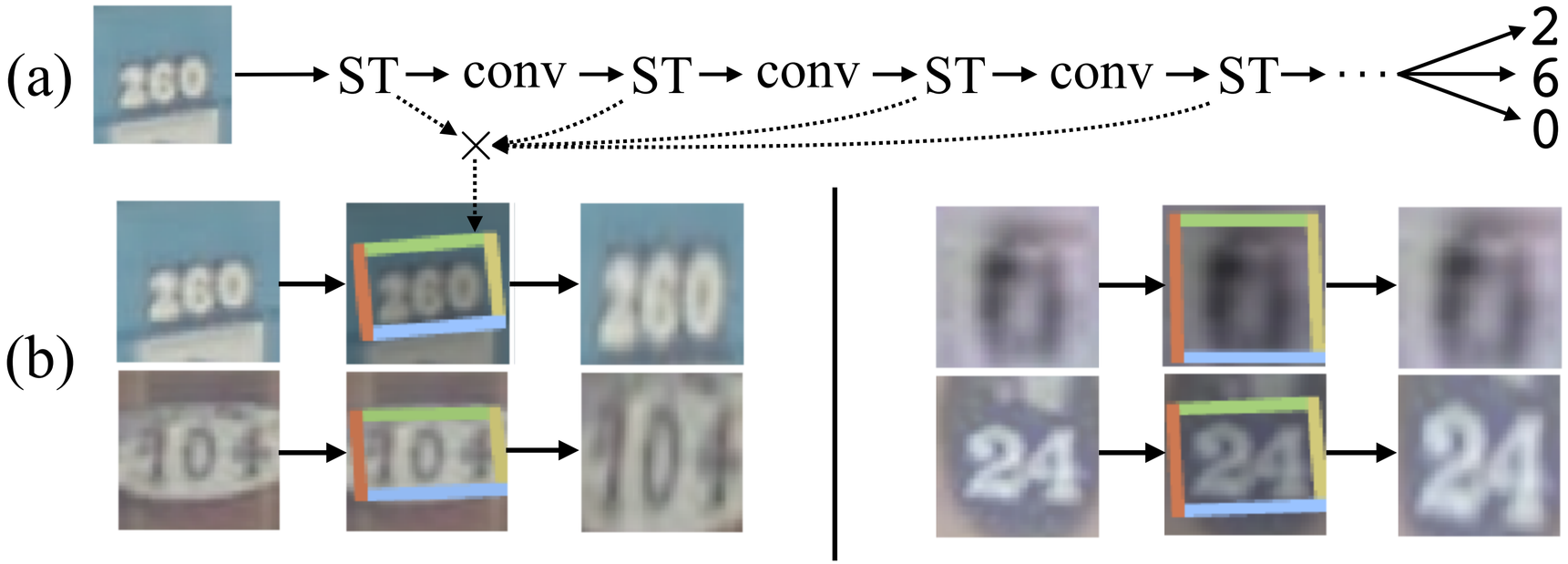
Differentiable image sampling



Jaderberg et al., “Spatial Transformer Networks”. NIPS 2015

Differentiable image sampling

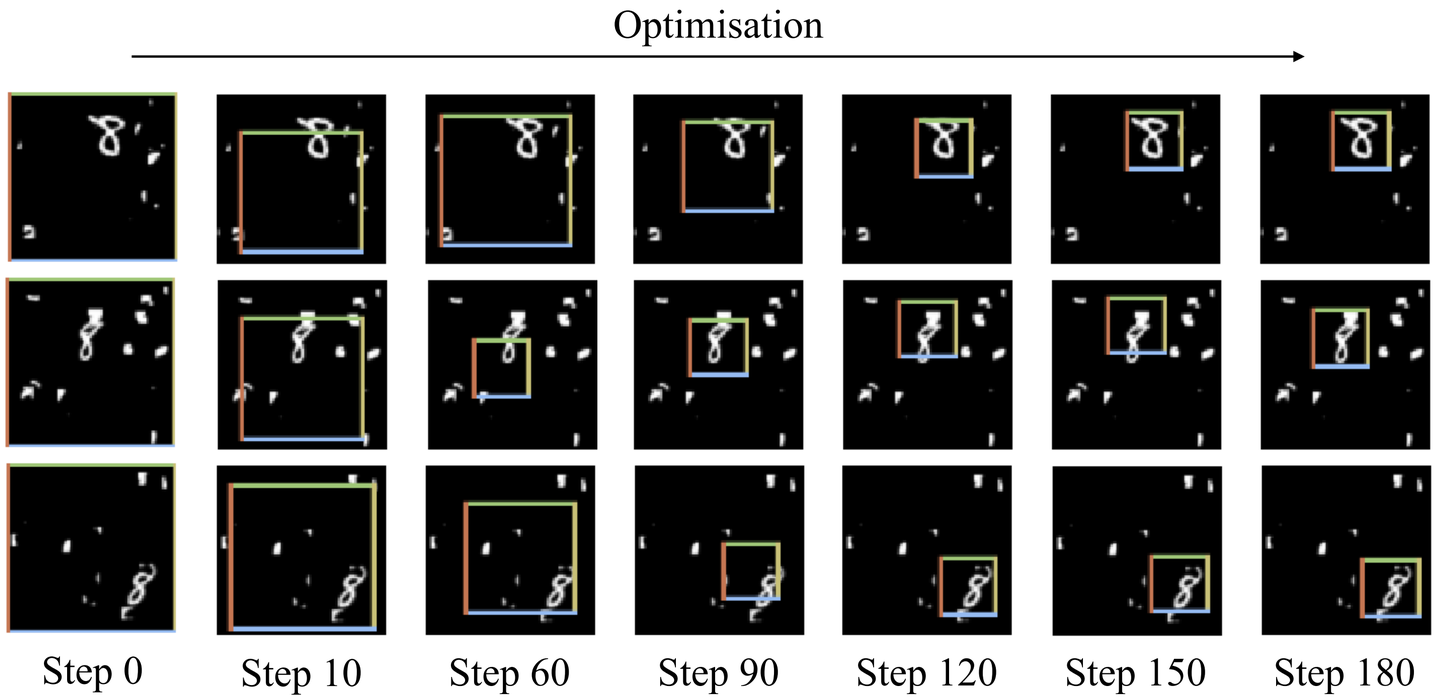
- Learn to predict a chain of transformations:



Jaderberg et al., “Spatial Transformer Networks”. NIPS 2015

Differentiable image sampling

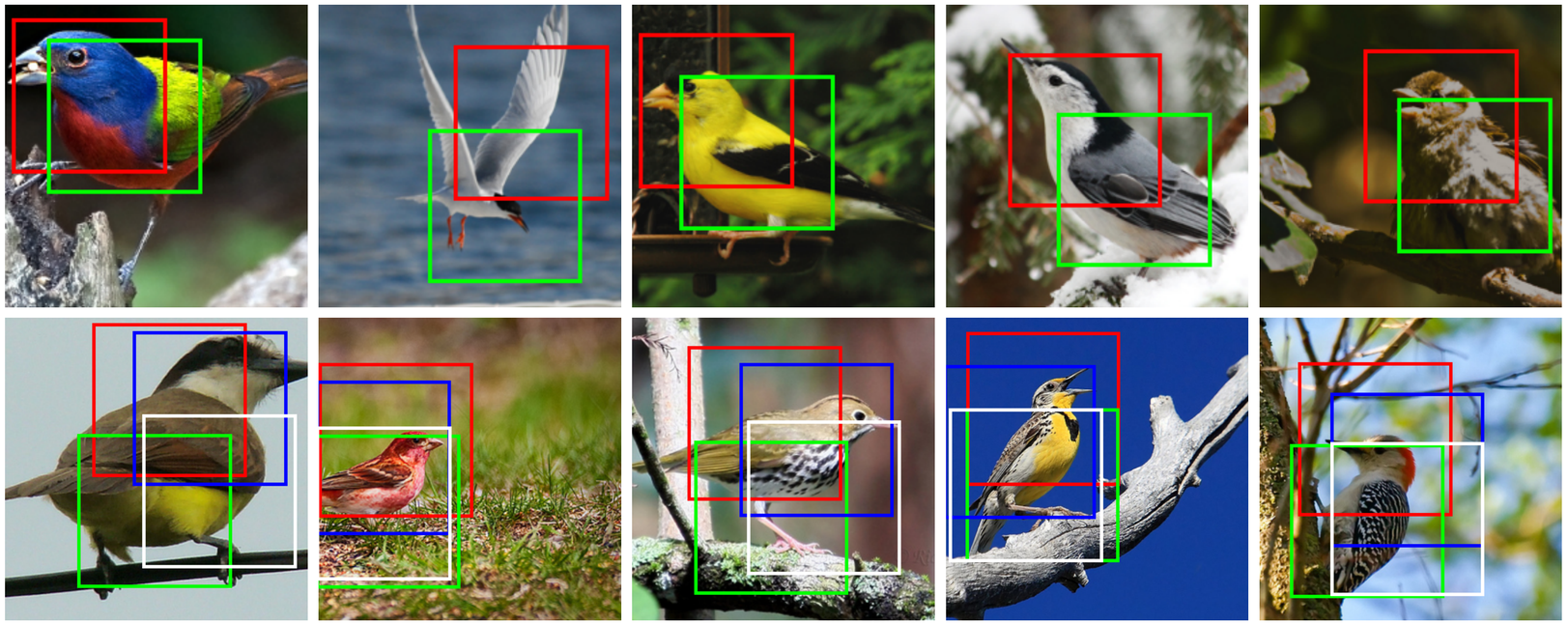
- Learning to localise RoI without dense supervision (only class label):



Jaderberg et al., “Spatial Transformer Networks”. NIPS 2015

Differentiable image sampling

- Training multiple STs: focus on different object parts



Jaderberg et al., “Spatial Transformer Networks”. NIPS 2015

Spatial transformers

- Learning to localise without any supervision
- Makes the network equivariant to certain transformations (e.g. affine)
- Fully differentiable
- Cons:
  - difficulty of training for generalising to more challenging scenes

Jaderberg et al., “Spatial Transformer Networks”. NIPS 2015

Summary

- Evaluating object detection;
- Feature Pyramid Networks (FPNs);
- One-stage detectors (YOLO, SSD, RetinaNet);
- Focal loss;
- Spatial transformers.

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.