Deep Learning for Object Detection

Lluis Gomez i Bigorda

Computer Vision Center, Universitat Autònoma de Barcelona.

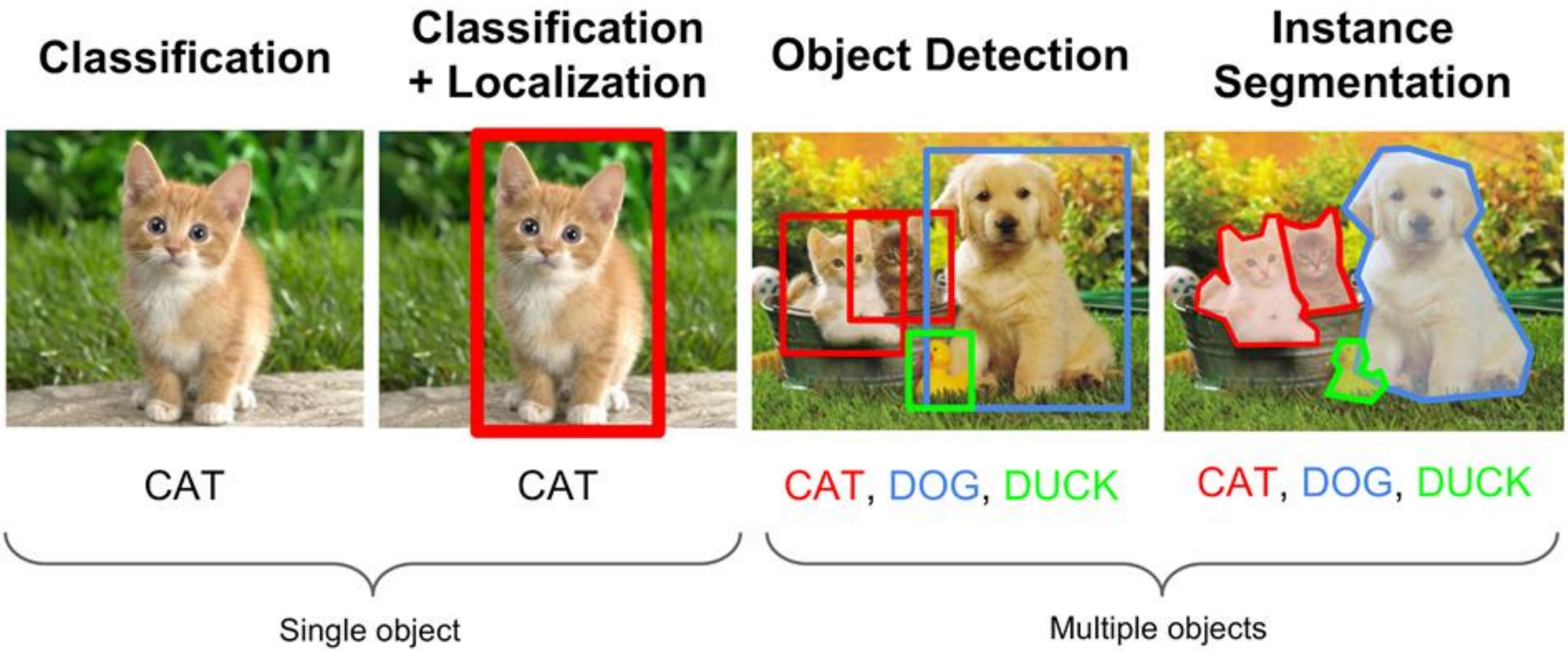
Deep learning for object detection: Outline

- Introduction
- Basic blocks and concepts
- Models

Deep learning for object detection: Outline

- Introduction
- Basic blocks and concepts
- Models

Computer Vision Tasks



https://chaosmail.github.io/deeplearning/2016/10/22/intro-to-deep-learning-for-computer-vision/

Firsts Localization and Detection Datasets

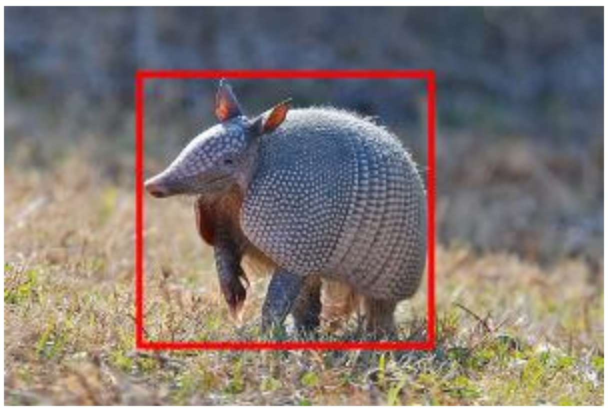
VOC Challenge: 2005-2012

- Classification
- Detection
- Segmentation

- 20 categories
- 6k training images (17k objects)
- 6k validation + 10k test

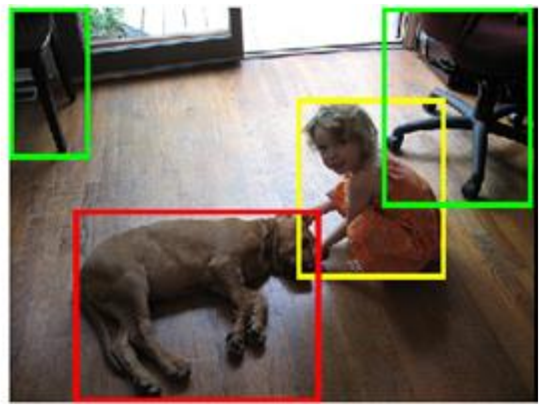
IMAGENET

Localization



- 1000 categories
- 1.2M training images
- 150k val + test images

Detection



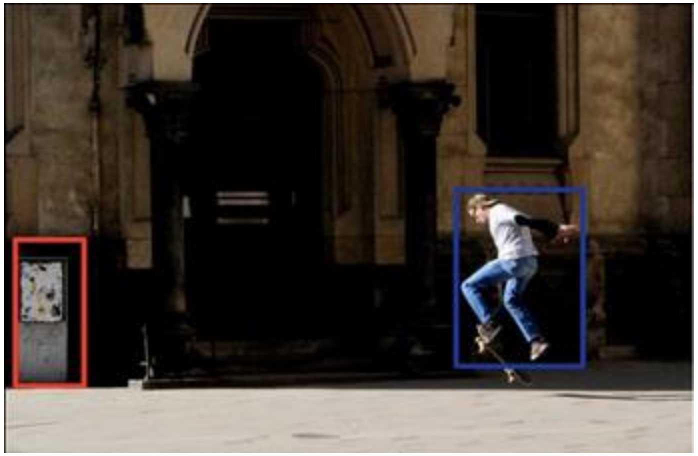
- 200 categories
- 456K training images
- 60K val + test images

Detection from Video

- 30 categories
- 6K videos

Microsoft COCO
Common Objects in Context

Detection



Segmentation

- 80 categories
- 160K training images
- 1M instances

http://presentations.cocodataset.org/COCO17-Detect-Overview.pdf

CVC

Recent Datasets

Open Images Dataset V5+ (2019)

- 15,851,536 boxes on 600 categories
- 2,785,498 instance segmentations on 350 categories
- 36,464,560 image-level labels on 19,959 categories

Objects365 (2019)

- 365 categories
- 600k images
- 10 million bounding boxes

Visual Genome (2017)

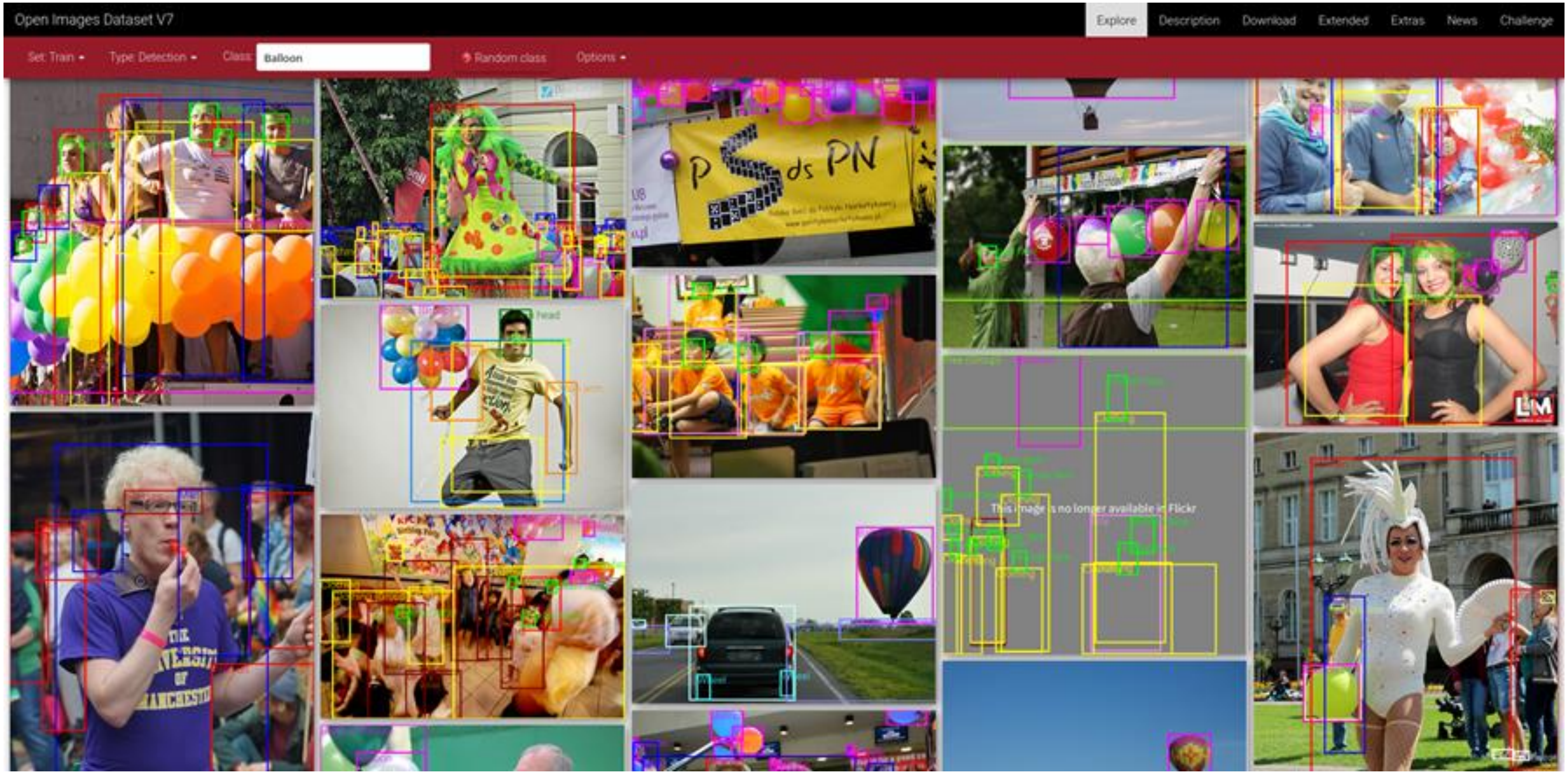
- 108,077 Images
- 5.4 Million Region Descriptions
- 3.8 Million Object Instances
- 2.8 Million Attributes
- 2.3 Million Relationships

++

LVIS: Large Vocabulary Instance Segmentation (2019)

- 1200+ Categories
- 164k images.
- Long Tail (large number of rare categories)
- > 2 million high quality instance segmentation masks.

Open Images Dataset V7



15,851,536 boxes on 600 classes, 2,785,498 instance segmentations on 350 classes, 3,284,280 relationship annotations on 1,466 relationships, ++

https://storage.googleapis.com/openimages/web/index.html

Computer Vision Tasks

Let's define an input and an output for each task

image → Classification Model → Class

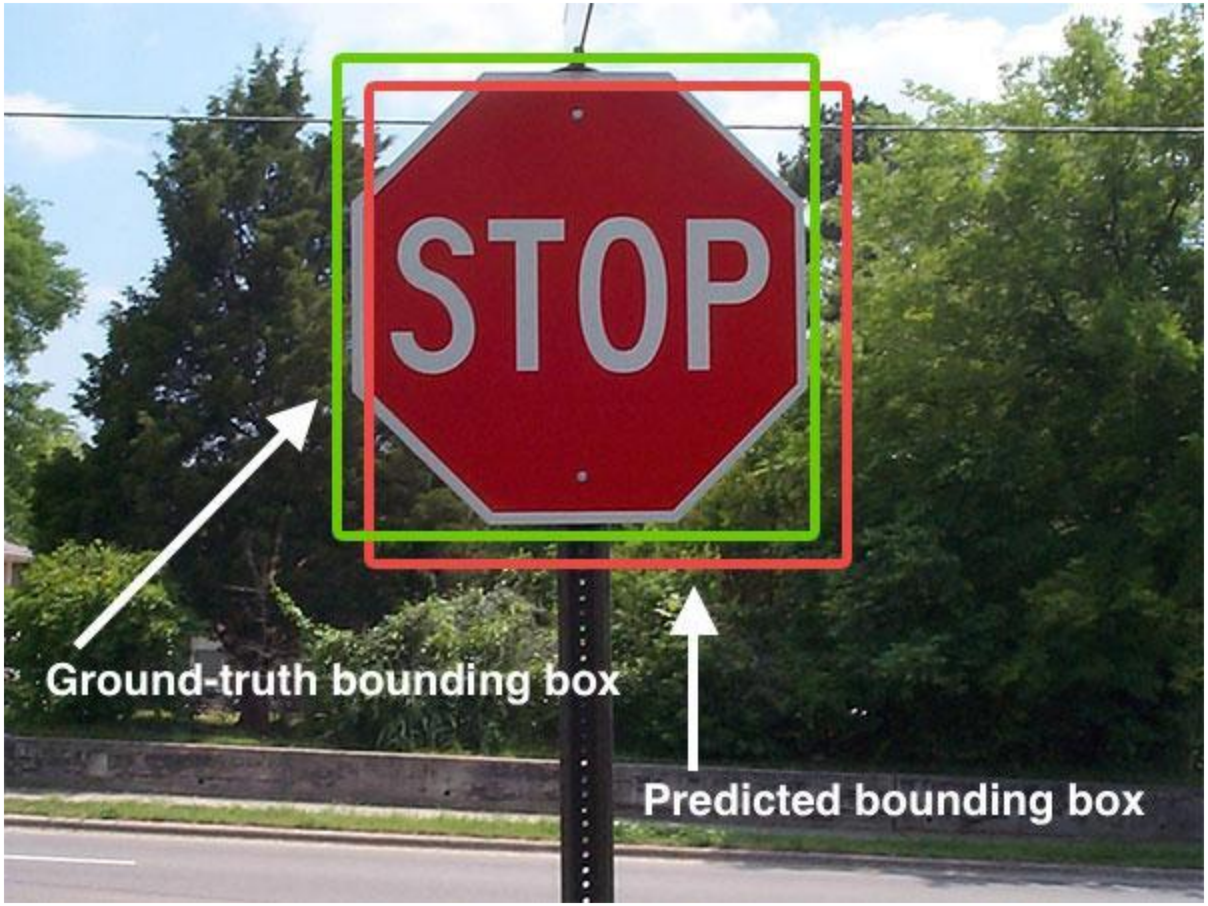
image → Localization Model → Class, Bounding Box

image → Segmentation Model → Class per pixel

image → Detection Model → Class, Confidence, Bounding Box

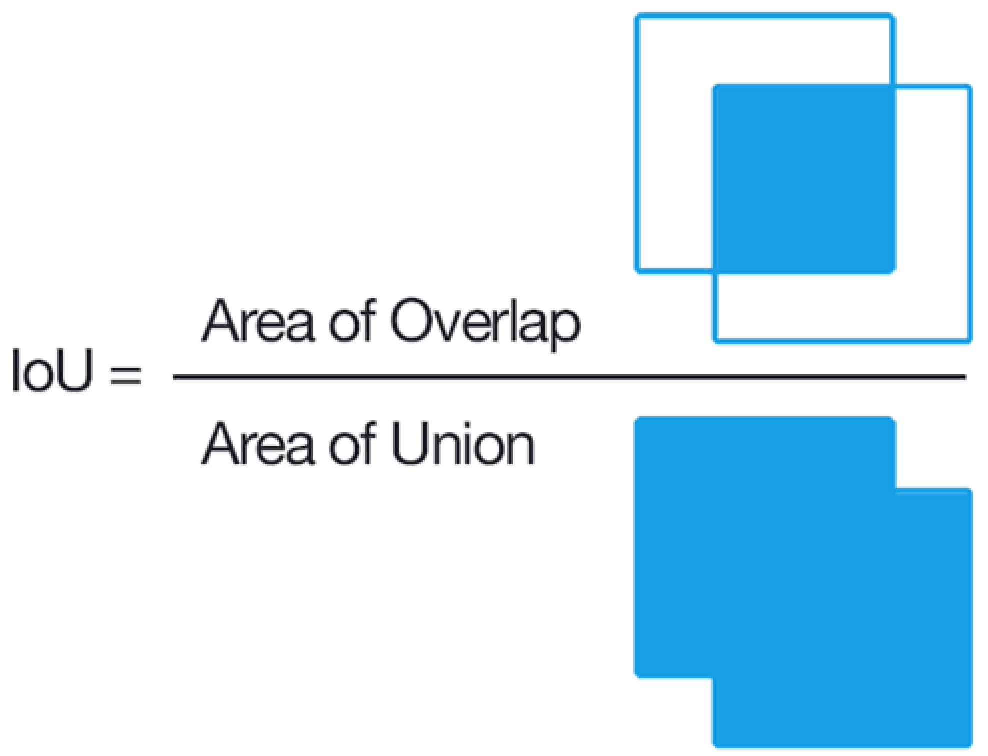
image → Detection in Video Model → Class, Frame, Confidence, Bounding Box

Evaluation



https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Intersection over Union (IoU)

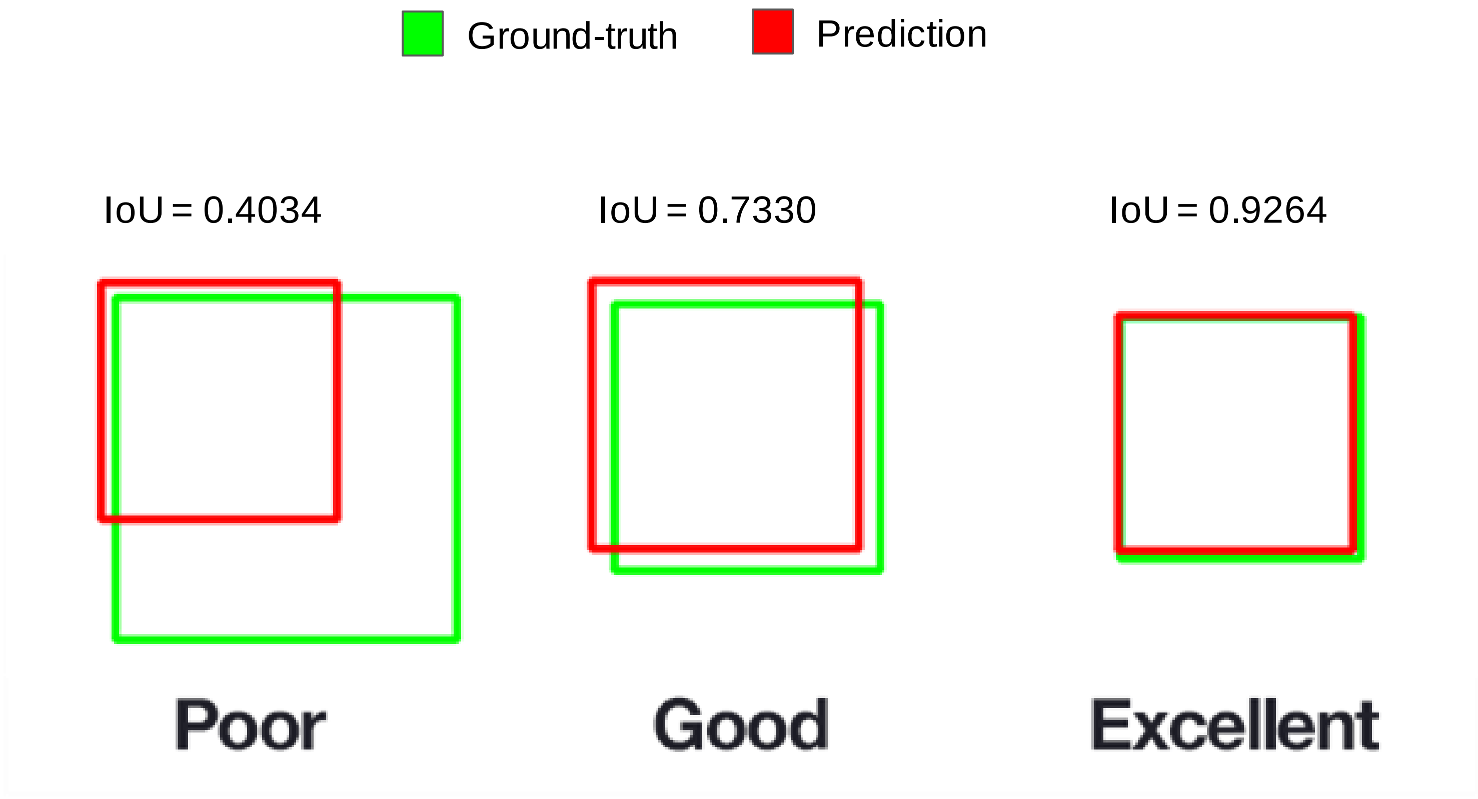


Intersection over Union (IoU)

```
Intersection over Union (IoU) for object det                Python
def bb_intersection_over_union(boxA, boxB):
    # determine the (x, y)-coordinates of the intersection rectangle
    xA = max(boxA[0], boxB[0])
    yA = max(boxA[1], boxB[1])
    xB = min(boxA[2], boxB[2])
    yB = min(boxA[3], boxB[3])

    # compute the area of intersection rectangle
    interArea = max(0, xB - xA + 1) * max(0, yB - yA + 1)

    # compute the area of both the prediction and ground-truth
    # rectangles
    boxAArea = (boxA[2] - boxA[0] + 1) * (boxA[3] - boxA[1] + 1)
    boxBArea = (boxB[2] - boxB[0] + 1) * (boxB[3] - boxB[1] + 1)

    # compute the intersection over union by taking the intersection
    # area and dividing it by the sum of prediction + ground-truth
    # areas - the interesection area
    iou = interArea / float(boxAArea + boxBArea - interArea)

    # return the intersection over union value
    return iou
```

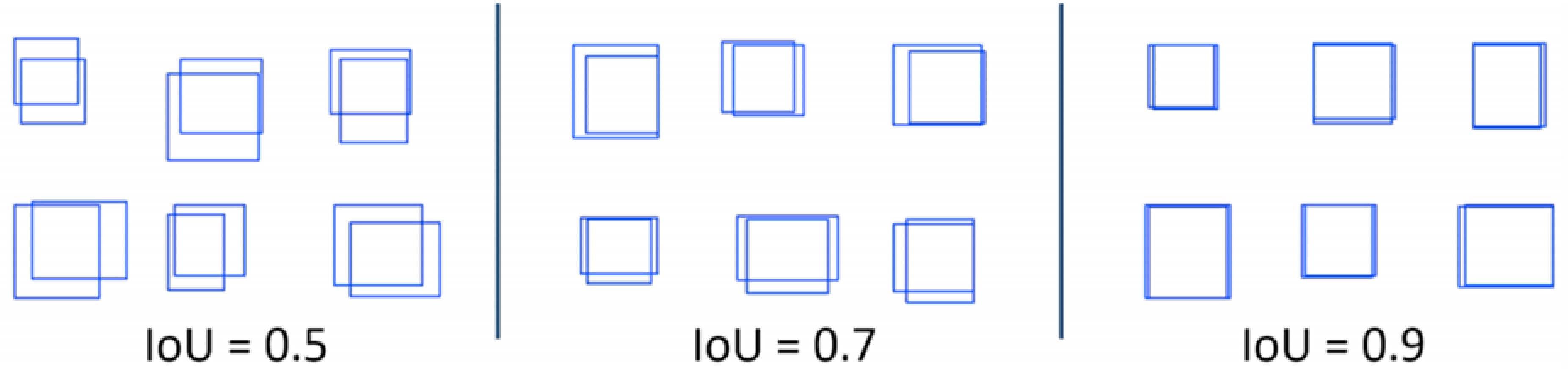
https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Intersection over Union (IoU)



https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Correct/Incorrect detections: IoU threshold



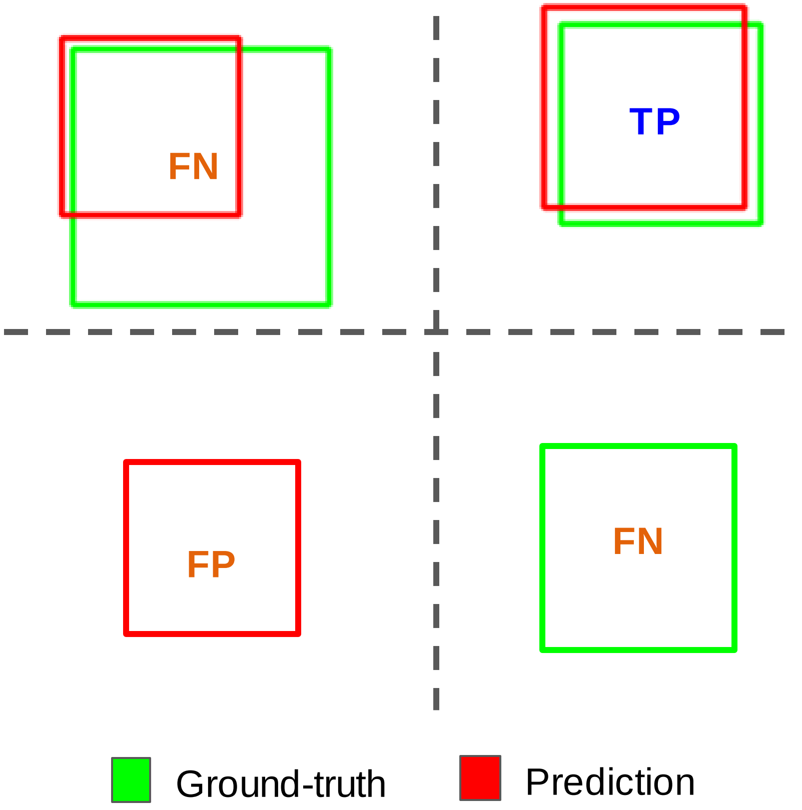
http://presentations.cocodataset.org/COCO17-Detect-Overview.pdf

Correct/Incorrect detections: IoU threshold

Being correct, being wrong

Predicted Values \ Actual Values	Positive (1)	Negative (0)
Positive (1)	TP	FP
Negative (0)	FN	TN

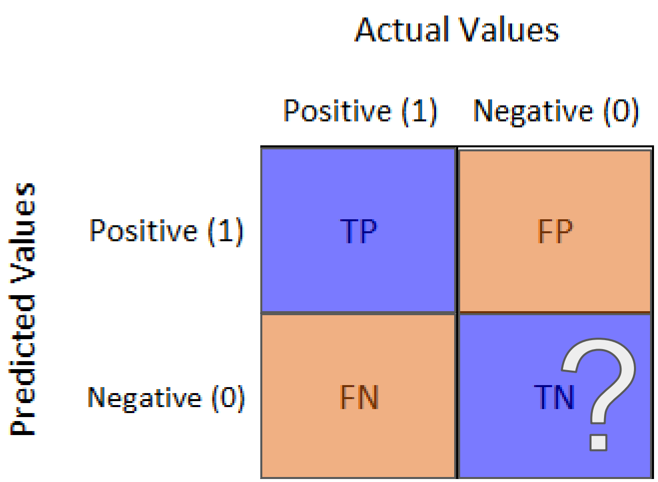
IoU > threshold; threshold = 0.5



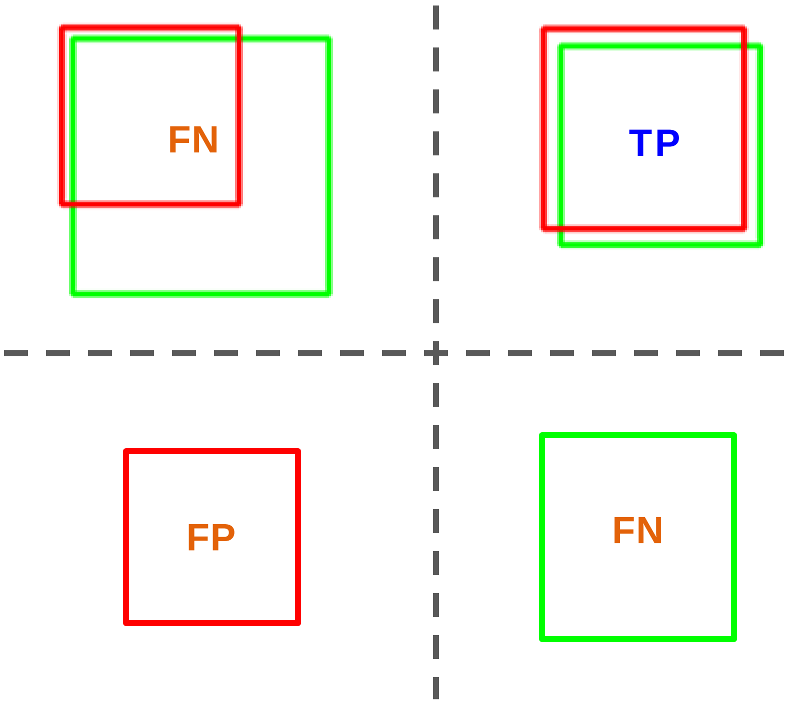
https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Correct/Incorrect detections: IoU threshold

Being correct, being wrong



IoU > threshold; threshold = 0.5

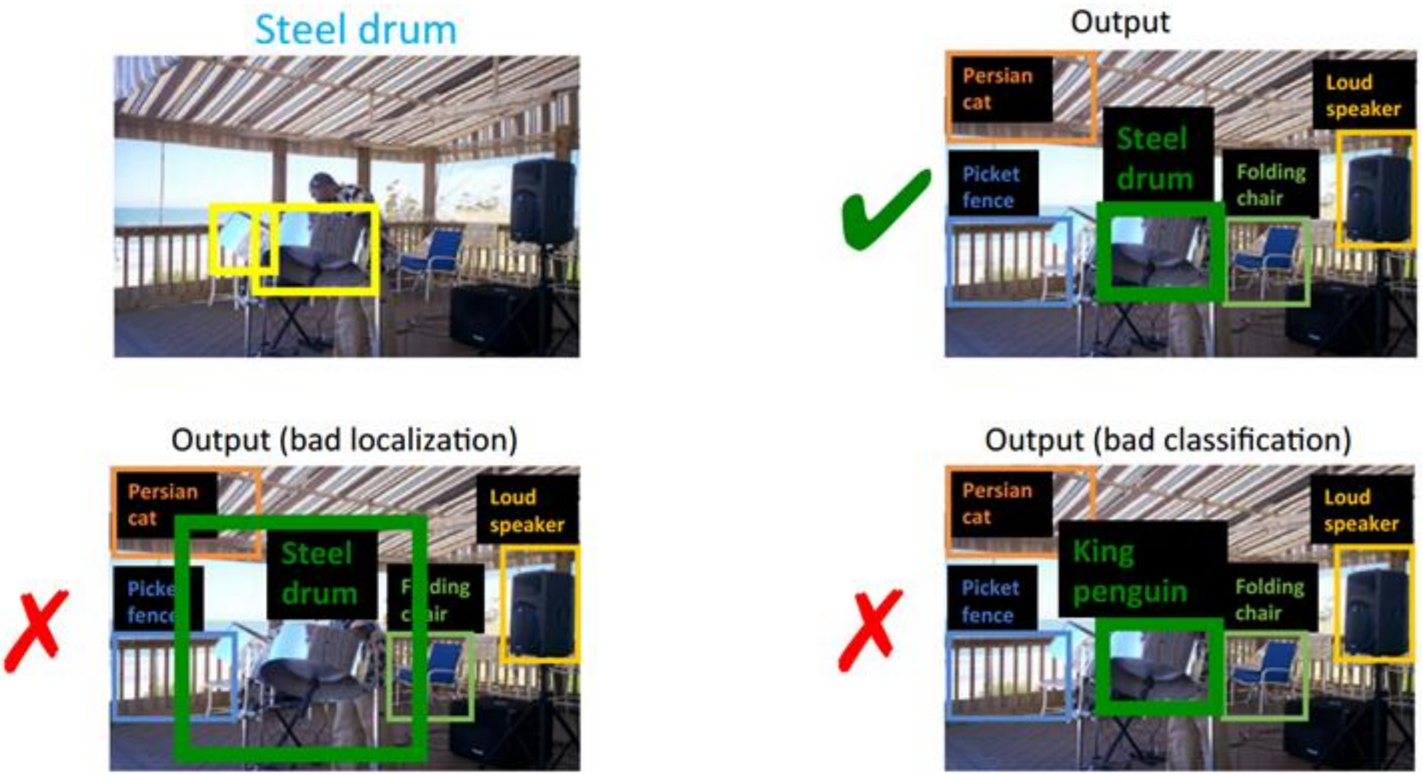


https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Object Localization evaluation

image ⇨ Detection Model ⇨ Class Bounding Box

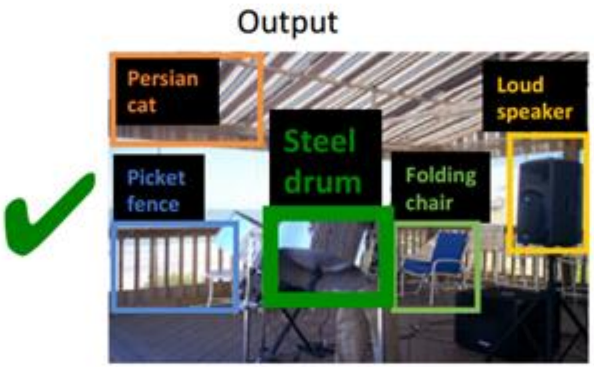
Wrong class or bad localization (IoU<0.5)



http://image-net.org/challenges/talks/2016/ILSVRC2016_10_09_clsloc.pdf

Object Localization evaluation

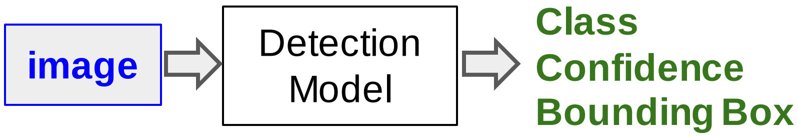
Wrong class or bad localization (IoU<0.5)

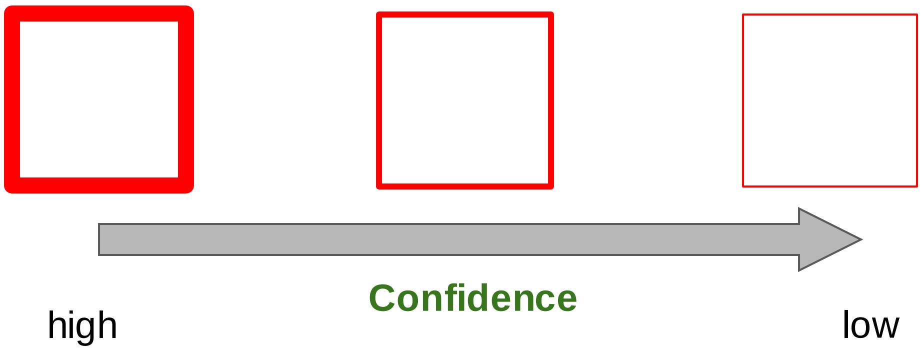


$$\text{Error} = \frac{1}{100{,}000} \sum_{\substack{100{,}000 \\ \text{images}}} 1[\text{incorrect on image i}]$$

http://image-net.org/challenges/talks/2016/ILSVRC2016_10_09_clsloc.pdf

Object Detection evaluation





Metrics:

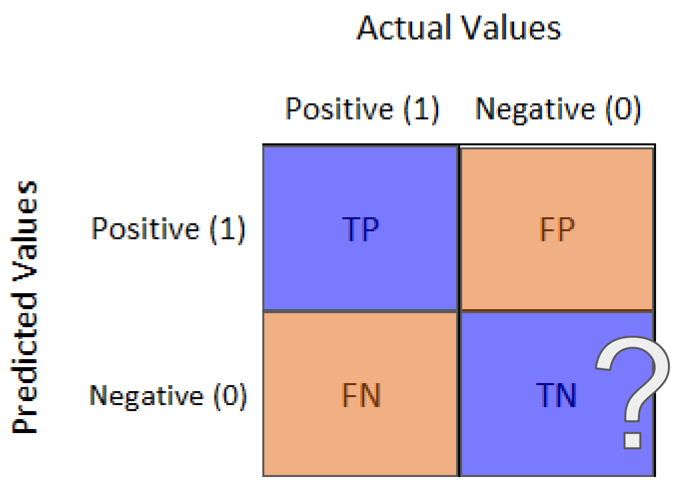
1) Average Precision (AP)
2) Average Recall (AR)

Object Detection evaluation

Computing Average Precision
(A four step procedure)

1. Order the predictions using confidence
2. Compute Precision and Recall (considering all possible operation points / confidence thresholds)
3. Plot Precision Recall plot (optional)
4. Compute Average Precision (AP)

Object Detection evaluation



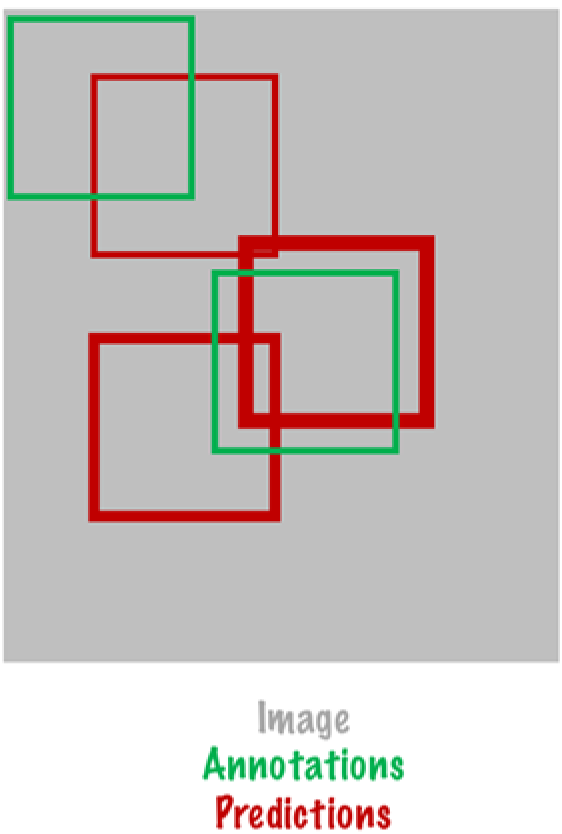
$$TP = \text{True positive}$$
$$TN = \text{True negative}$$
$$FP = \text{False positive}$$
$$FN = \text{False negative}$$

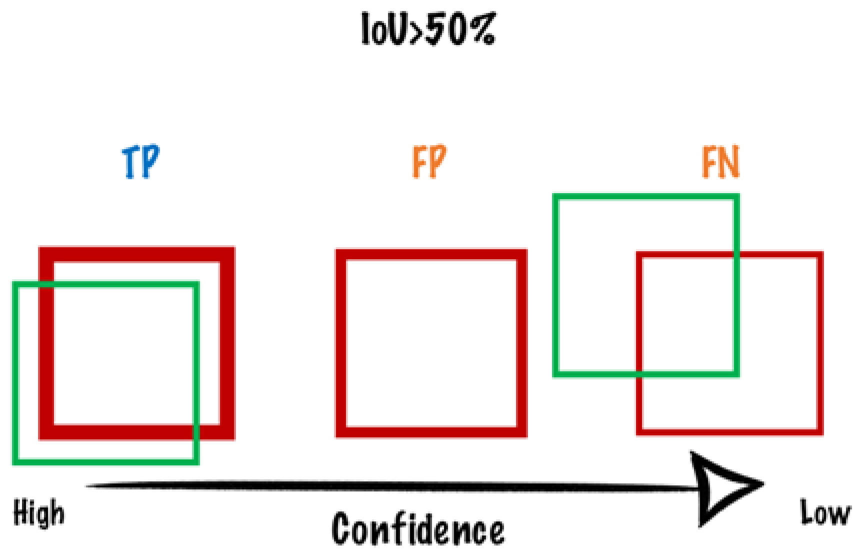
$$Precision = \frac{TP}{TP + FP}$$

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

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

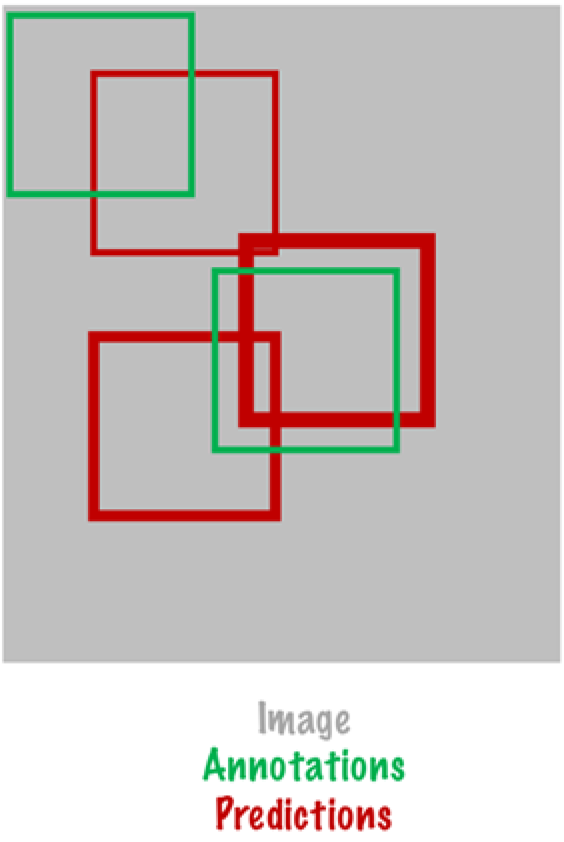
1. Predictions Ordering

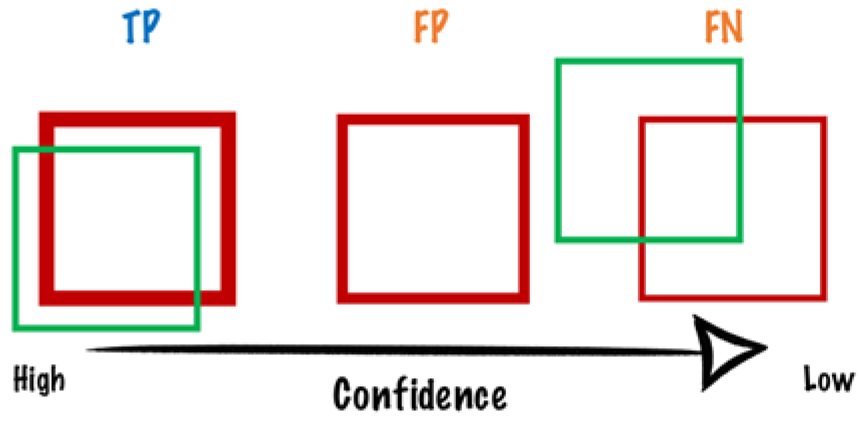




TP	1	1	1
FP	0	1	1
FN	1	1	1

2. Compute Precision and Recall





TP	1	1	1
FP	0	1	1
FN	1	1	1
Precision	1	.5	.5
Recall	.5	.5	.5

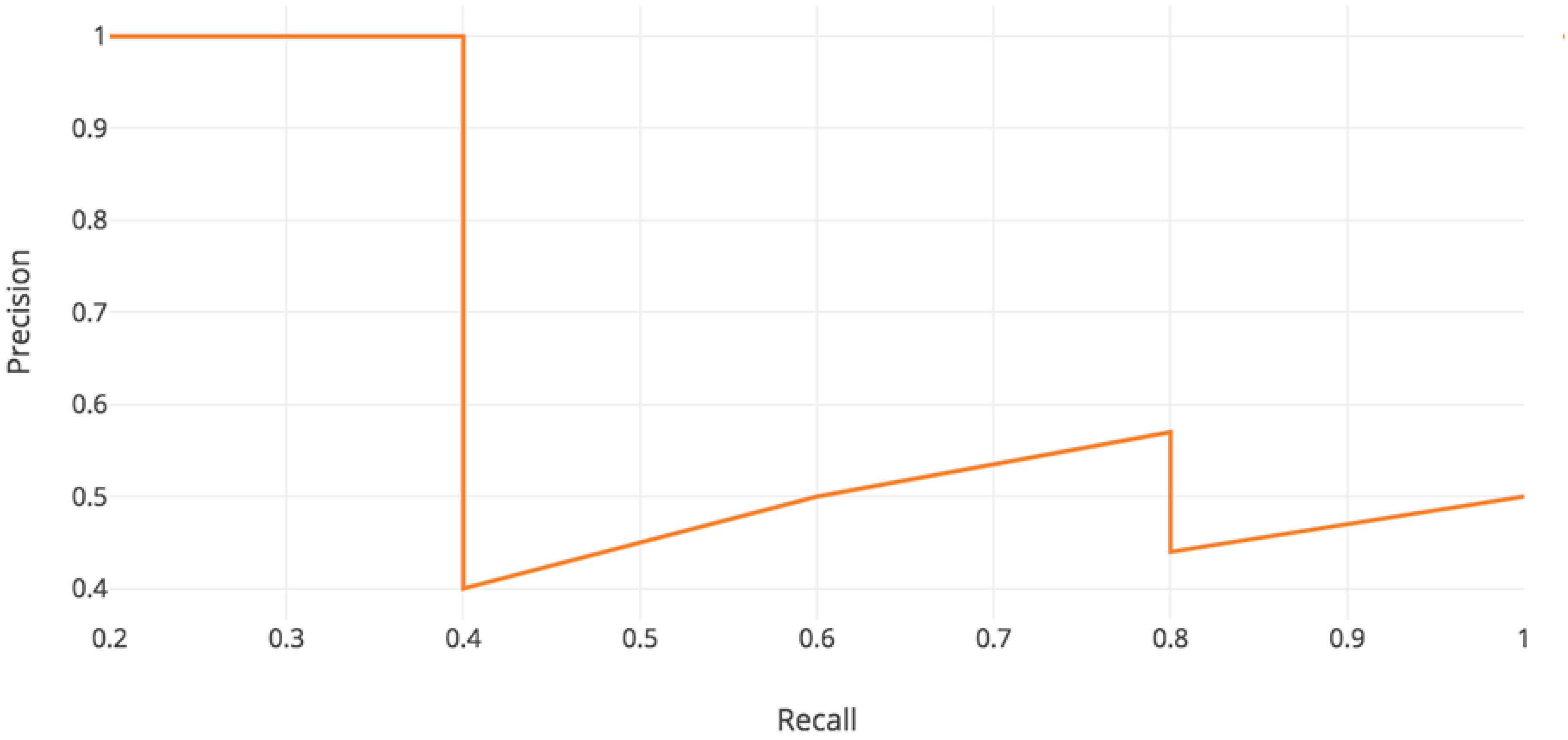
2. Compute Precision and Recall

Example: the whole dataset contains 5 objects, our model has predicted 10 bounding boxes.

Rank	Correct?	Precision	Recall
1	True	1.0	0.2
2	True	1.0	0.4
3	False	0.67	0.4
4	False	0.5	0.4
5	False	0.4	0.4
6	True	0.5	0.6
7	True	0.57	0.8
8	False	0.5	0.8
9	False	0.44	0.8
10	True	0.5	1.0

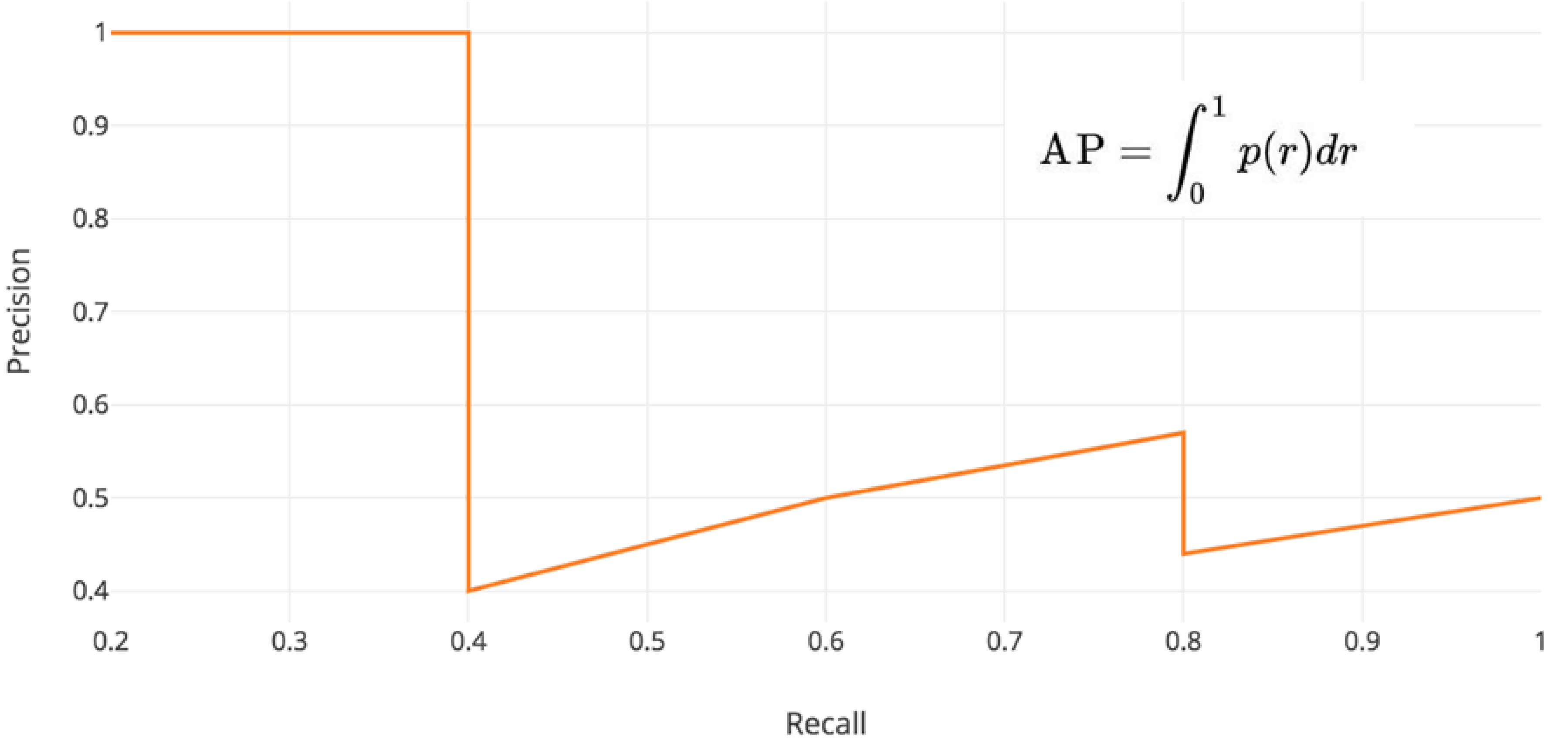
https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173

3. Precision - Recall plots



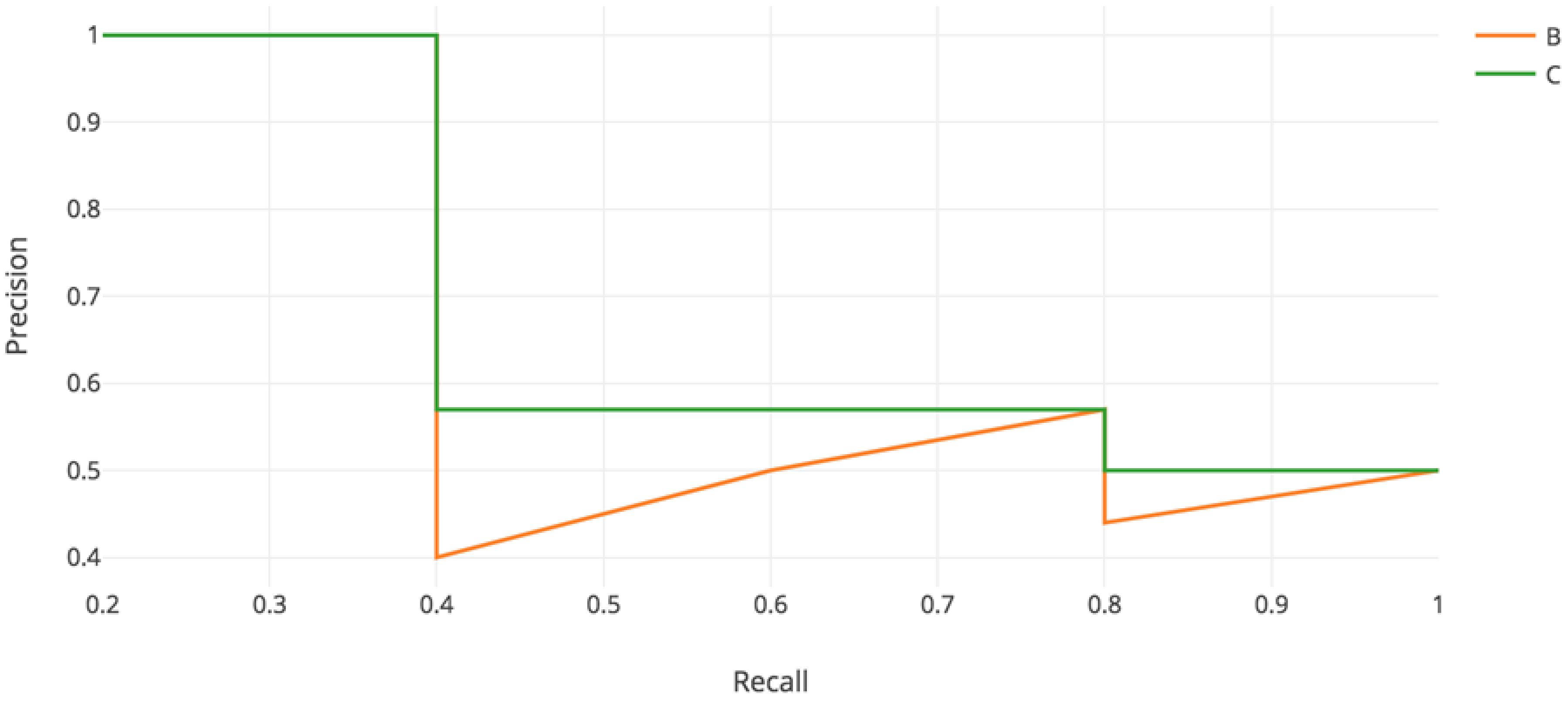
https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173

4. Average Precision (AP)



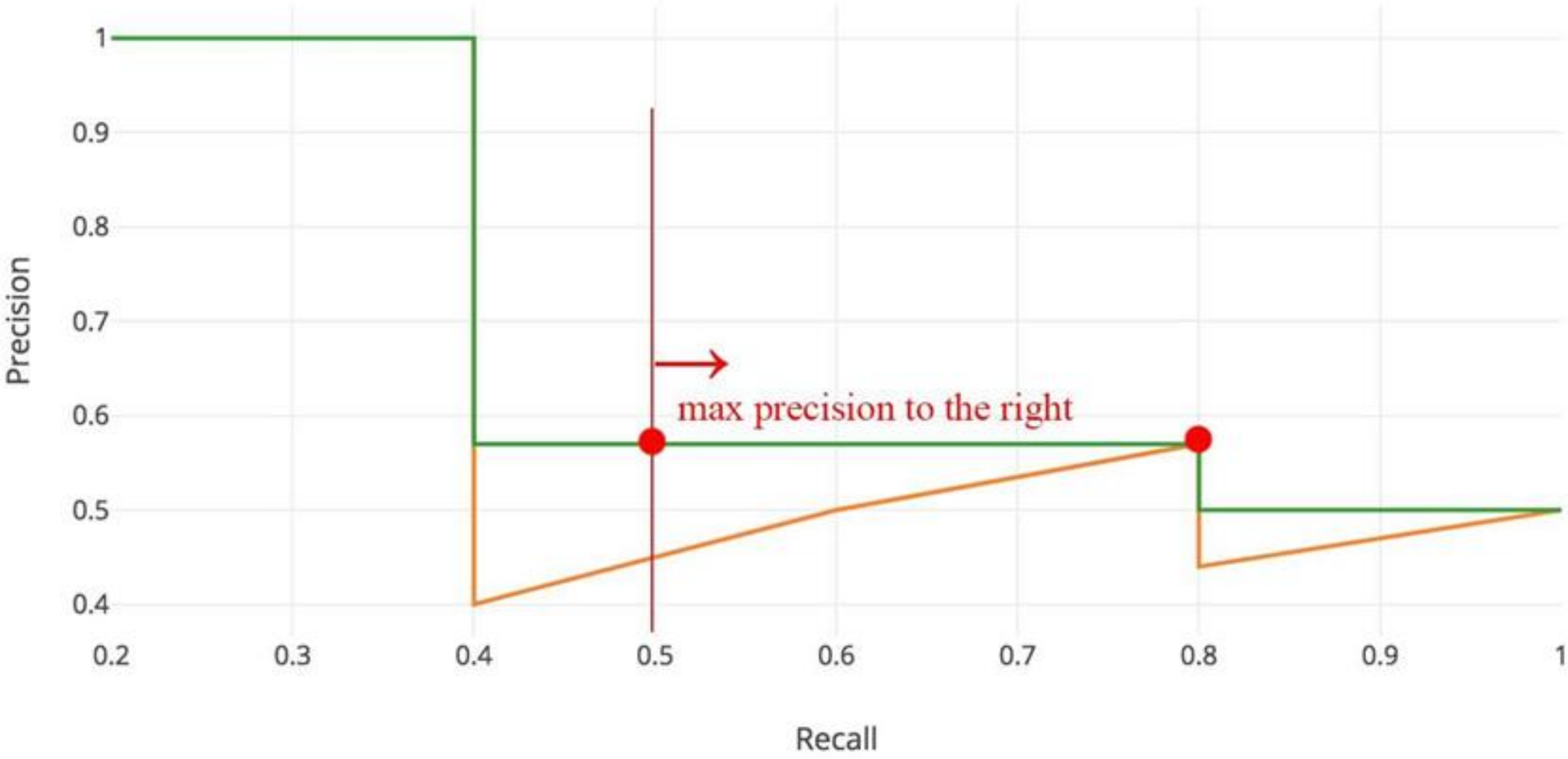
https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173

4. Average Precision (AP)



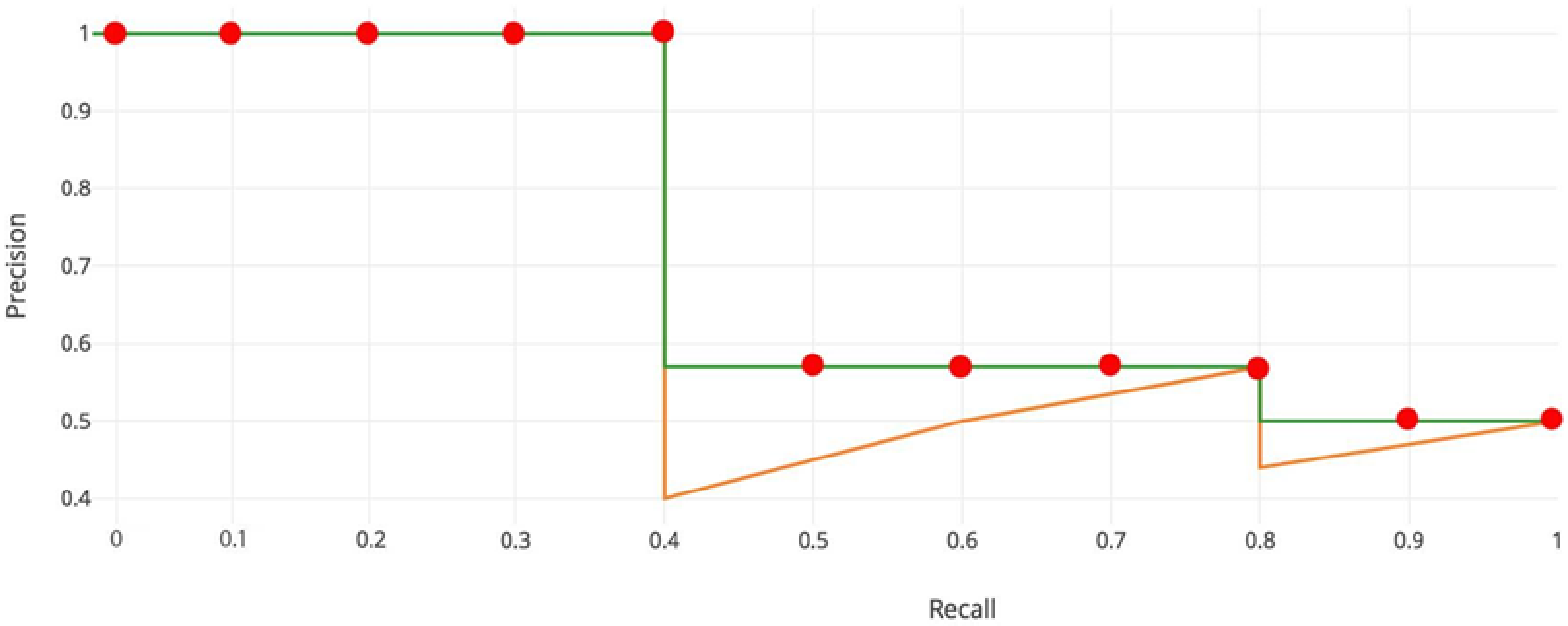
https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173

4. Average Precision (AP)



https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173

Interpolated AP



https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173

Object Detection evaluation

Microsoft COCO
Common Objects in Context

The following 12 metrics are used for characterizing the performance of an object detector on 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: $area > 96^2$
Average Recall (AR):	
$AR^{max=1}$	% AR given 1 detection per image
$AR^{max=10}$	% AR given 10 detections per image
$AR^{max=100}$	% AR given 100 detections per image
AR Across Scales:	
AR^{small}	% AR for small objects: $area < 32^2$
AR^{medium}	% AR for medium objects: $32^2 < area < 96^2$
AR^{large}	% AR for large objects: $area > 96^2$

http://cocodataset.org/#detection-eval

CVC
Centre de Visió per Computador

Average Recall

Other Scores: AR

- Measures the maximum recall over a fixed number of detections allowed in the image of 1, 10, 100.
- AR is averaged over small (A < 32 x 32), medium (32x 32 < A < 96 x 96) and large (A > 96 x 96) instances of objects.

Average Recall (AR):

$AR^{max=1}$	% AR given 1 detection per image
$AR^{max=10}$	% AR given 10 detections per image
$AR^{max=100}$	% AR given 100 detections per image

AR Across Scales:

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

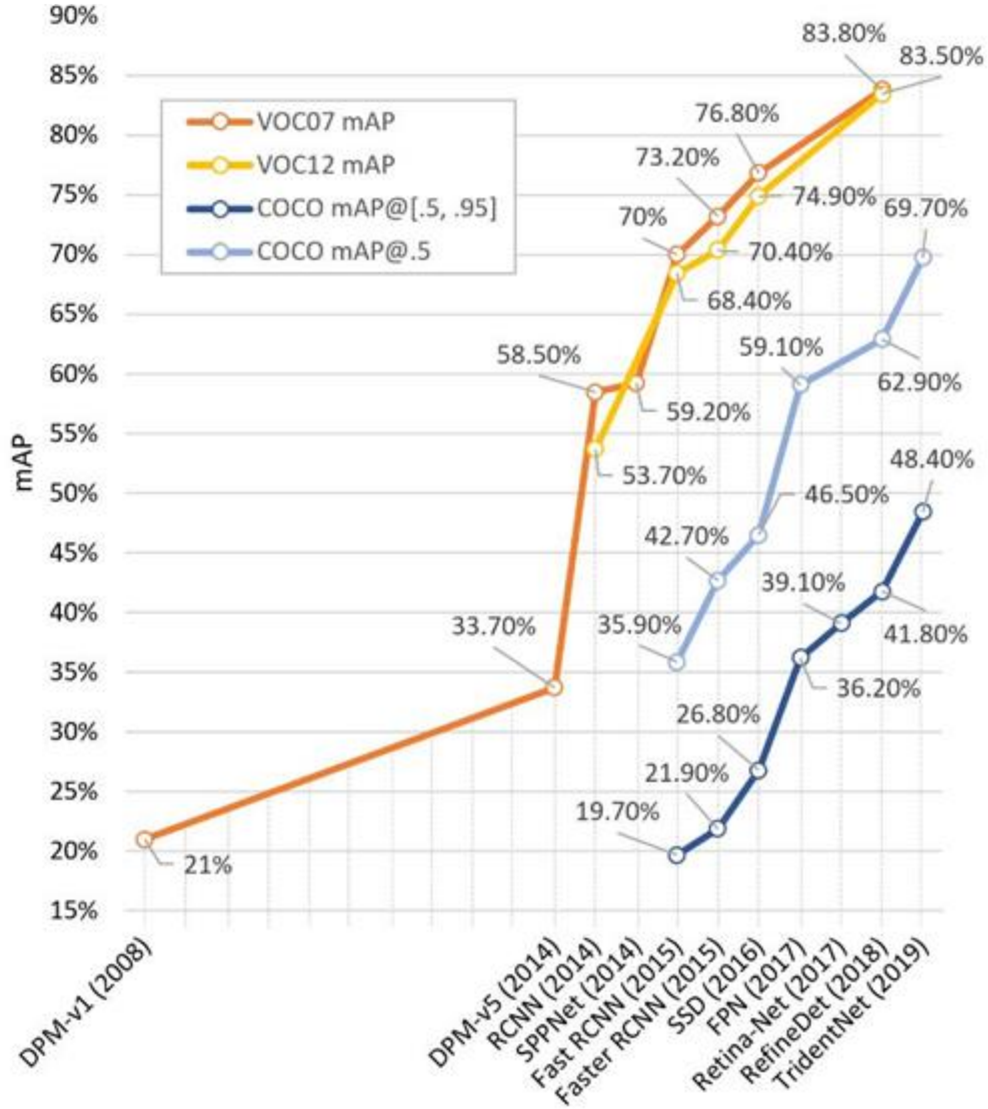
http://cocodataset.org/#detection-eval

Object Localization/Detection evolution

http://image-net.org/challenges/talks_2017/ILSVRC2017_overview.pdf

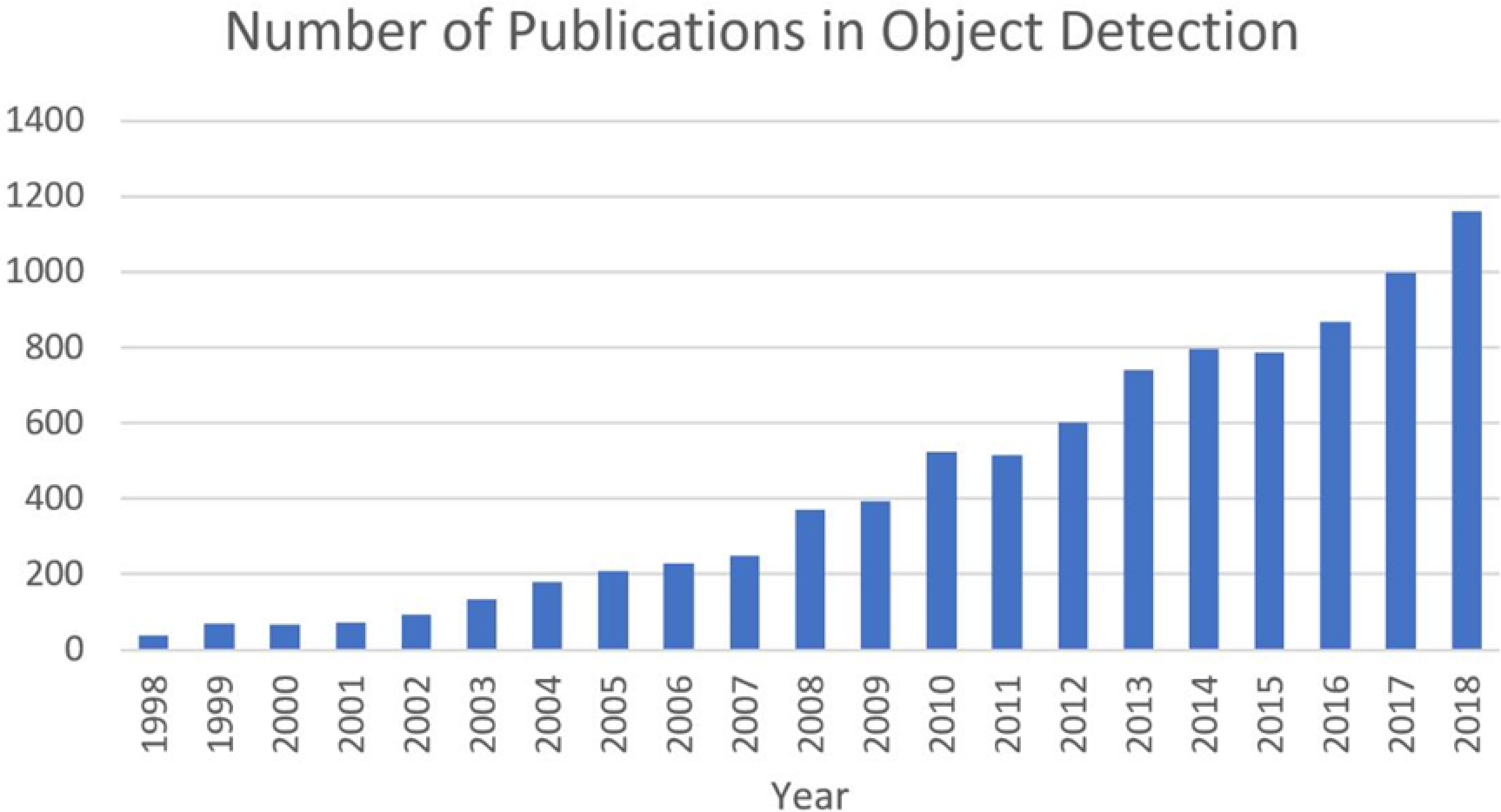
http://presentations.cocodataset.org/COCO17-Detect-Overview.pdf

Object Localization/Detection evolution



[Zou, Zhengxia, et al. "Object detection in 20 years: A survey." arXiv preprint arXiv:1905.05055 (2019).]()

Object Localization/Detection evolution



Zou, Zhengxia, et al. "Object detection in 20 years: A survey." arXiv preprint arXiv:1905.05055 (2019).

Deep learning for object detection: Outline

- Introduction
- Basic blocks and concepts
- Models

What is an outcome of an object detector?

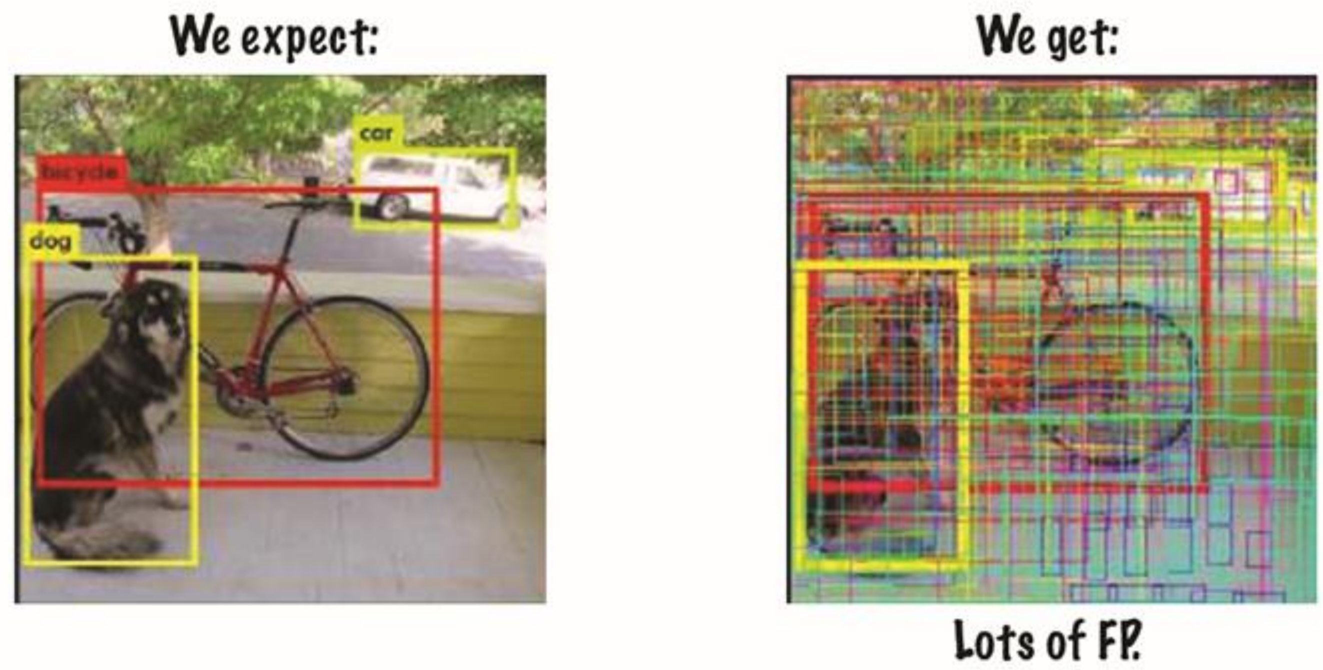
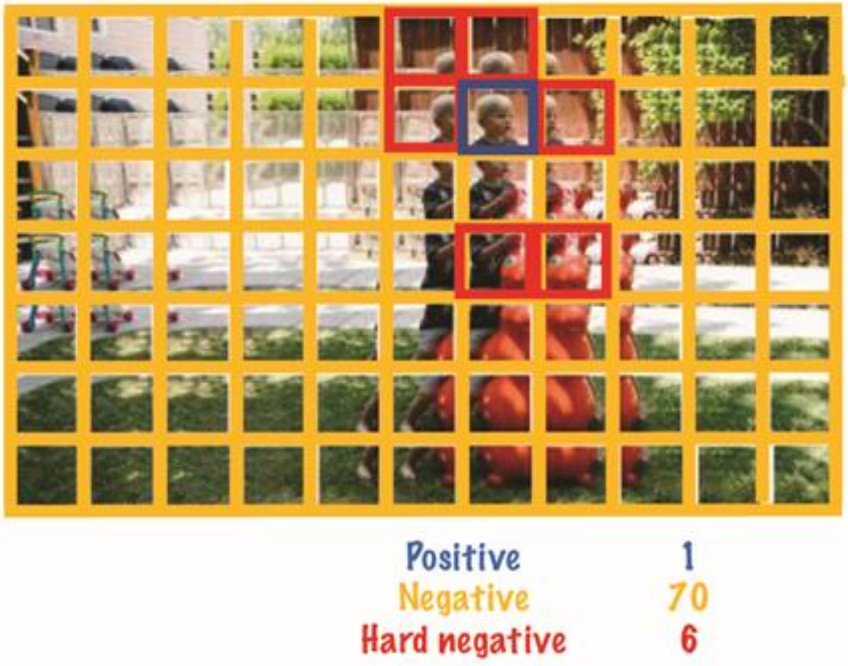


Image credit: https://pjreddie.com/darknet/yolo/

The unbalanced nature of detection.
Hard Negative Mining

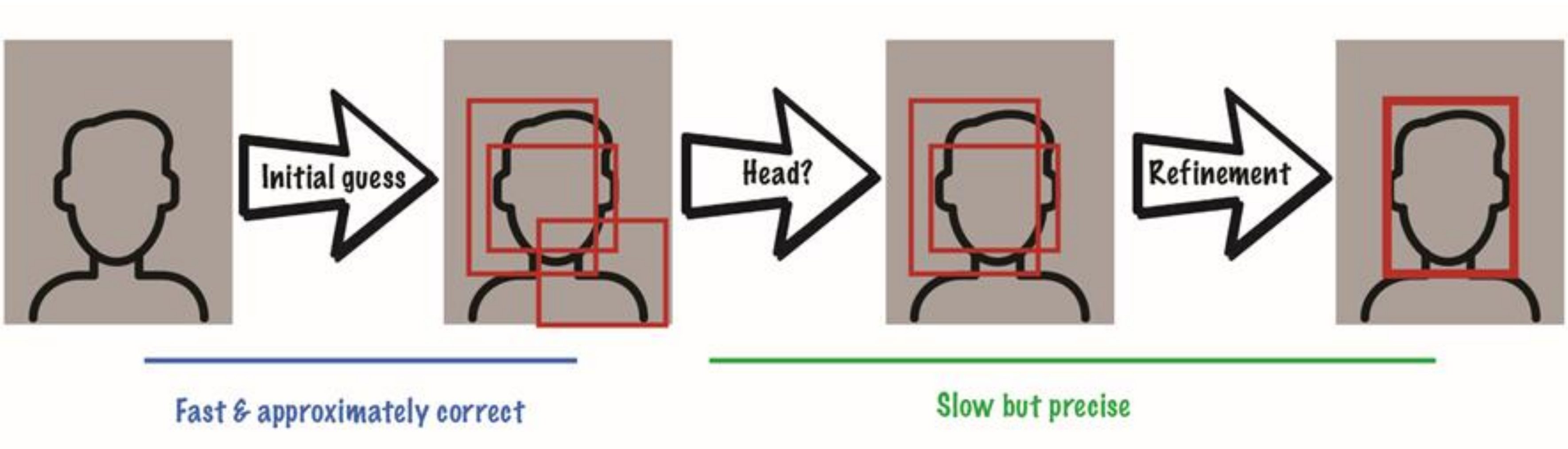
Build a head detector.



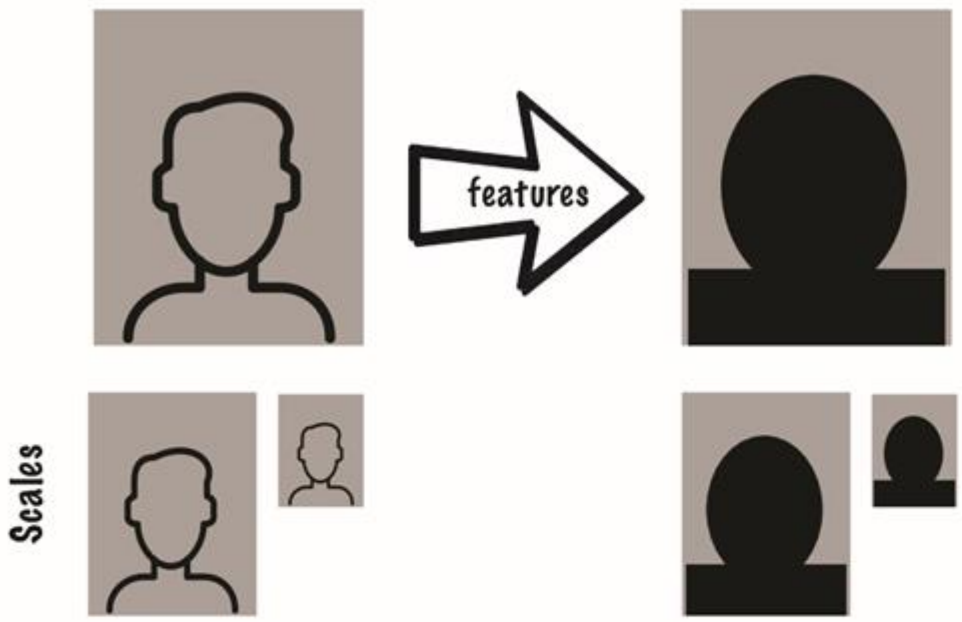
https://medium.com/@ageitgey/machine-learning-is-fun-part-3-deep-learning-and-convolutional-neural-networks-f40359318721#.dnbyjd6zg

Object detection pipeline

Given the unbalanced nature of detection. What do we need?



Initial guess



Features (in the past)

http://sharky93.github.io/docs/gallery/auto_examples/plot_hog.html

Features (currently)

Use pre-trained neural networks!

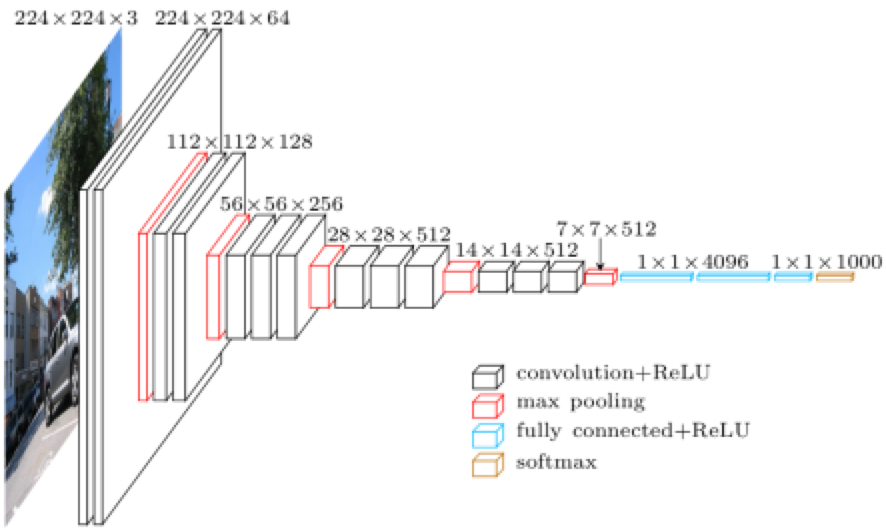
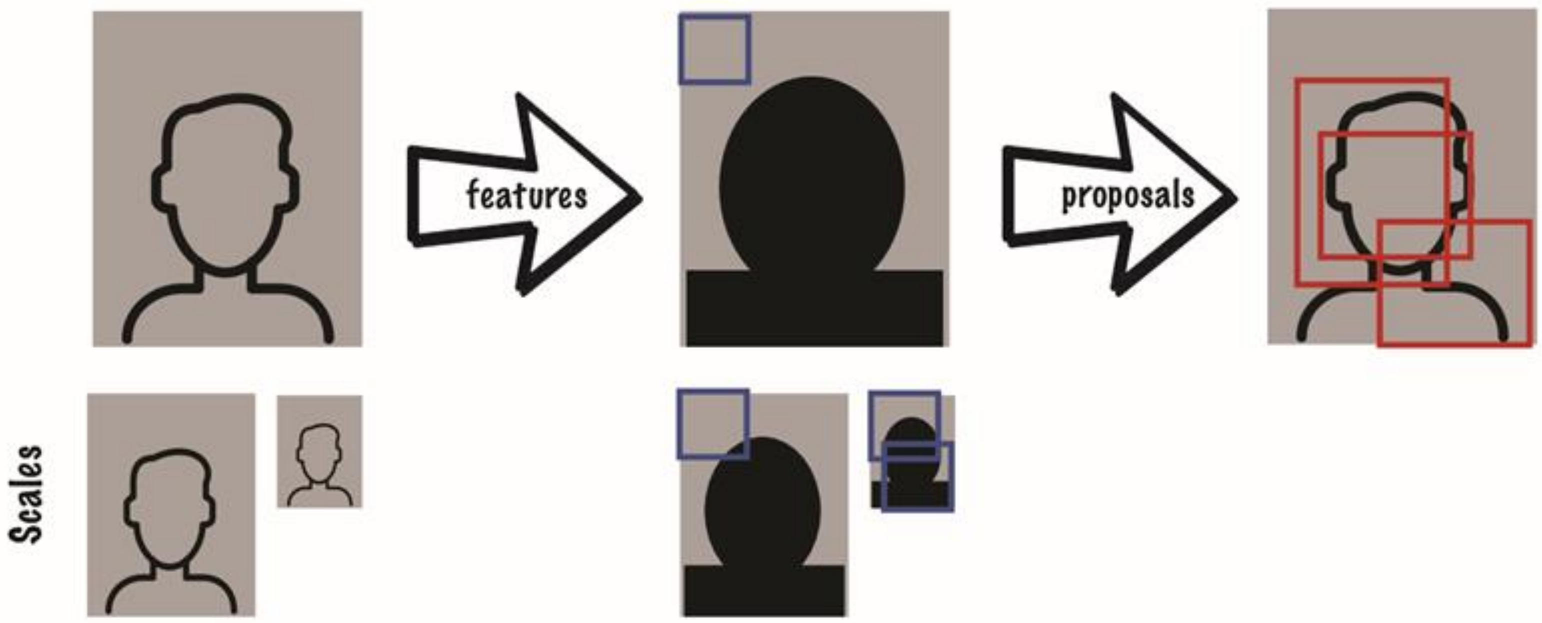


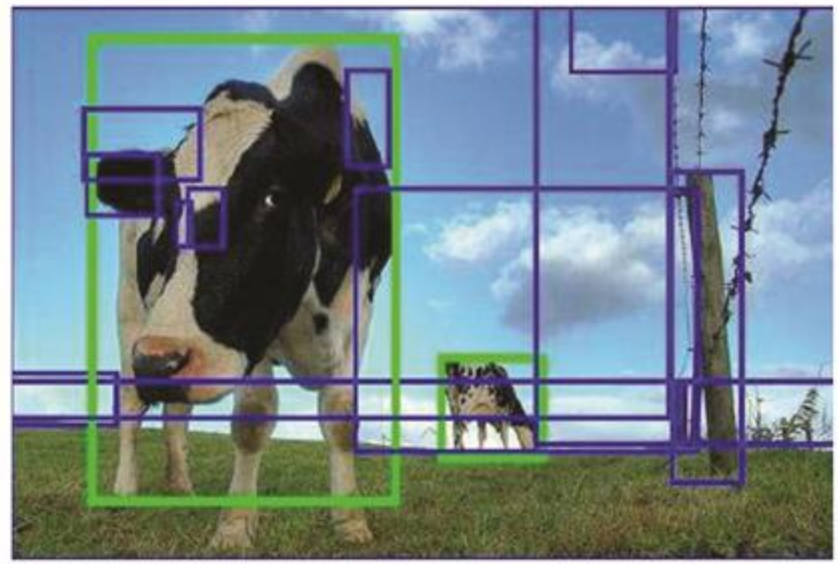
Image source: https://www.cs.toronto.edu/~frossard/post/vgg16/

Region proposals (Class agnostic classifiers)



Region proposals (Class agnostic classifiers)

Objectness [1]
Selective search [2]
Category-independent object proposals [3]
Constrained parametric min-cuts (CPMC) [4]
Multi-scale combinatorial grouping [5]

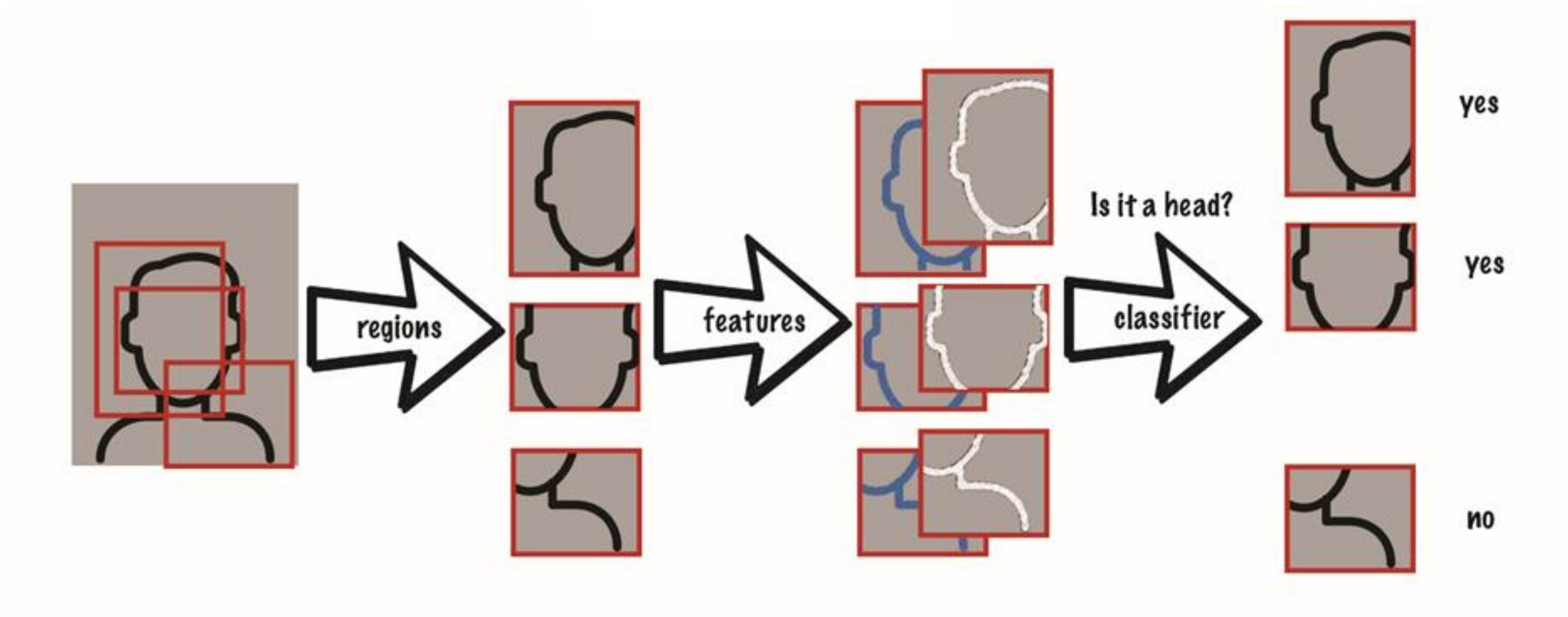


[1] B.Alexe et al.Measuring the objectness of image windows.
[2] J. Uijlings et al. Selective search for object recognition
[3] I. Endres and D. Hoiem. Category independent object proposals
[4] J. Carreira and C. Sminchisescu. CPMC: Automatic object segmentation using constrained parametric min-cuts
[5] P.Arbelaez et al. Multiscale combinatorial grouping

Image: https://ivi.fnwi.uva.nl/isis/publications/bibtexbrowser.php?key=UijlingsIJCV2013&bib=all.bib

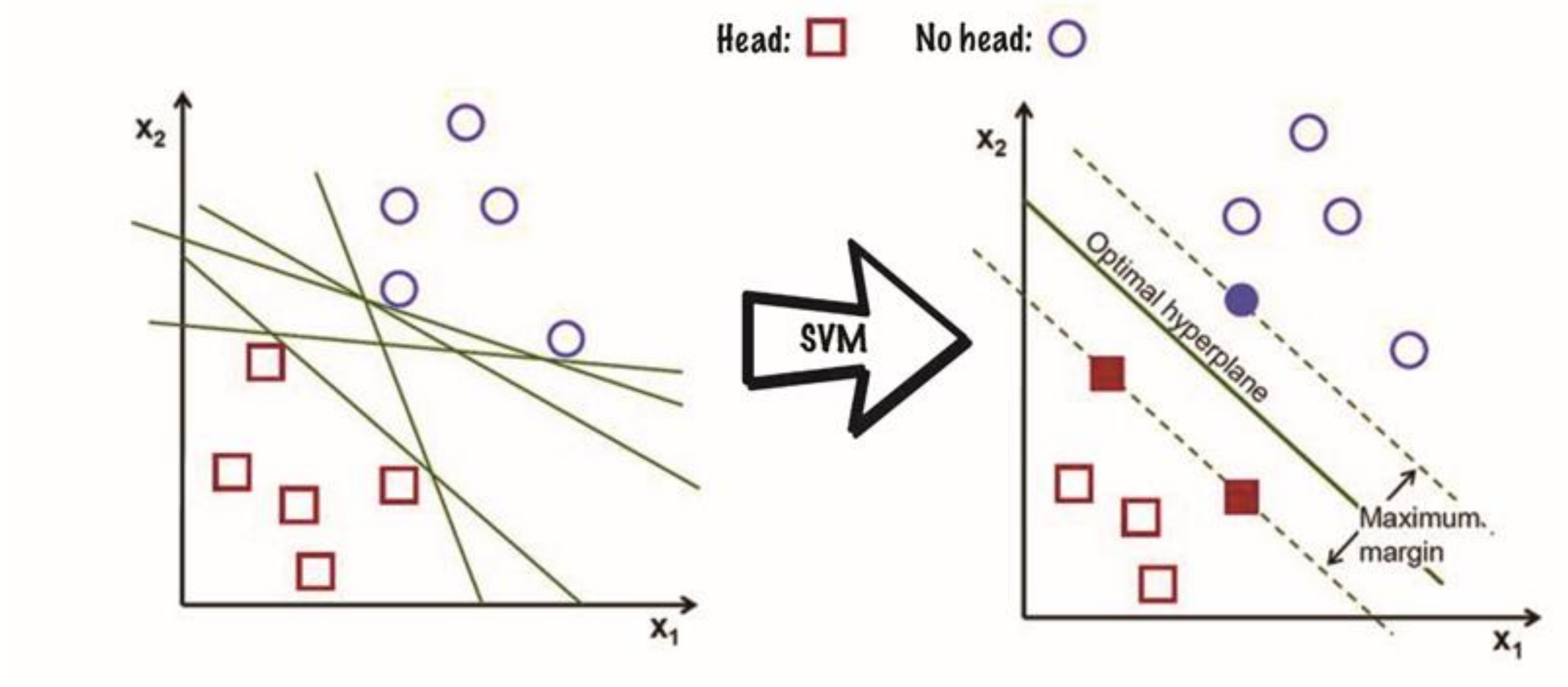
Region Proposals. Selective search.

(Object specific) Classifier



Classifier (in the past)

Support Vector Machines



Classifier (currently)

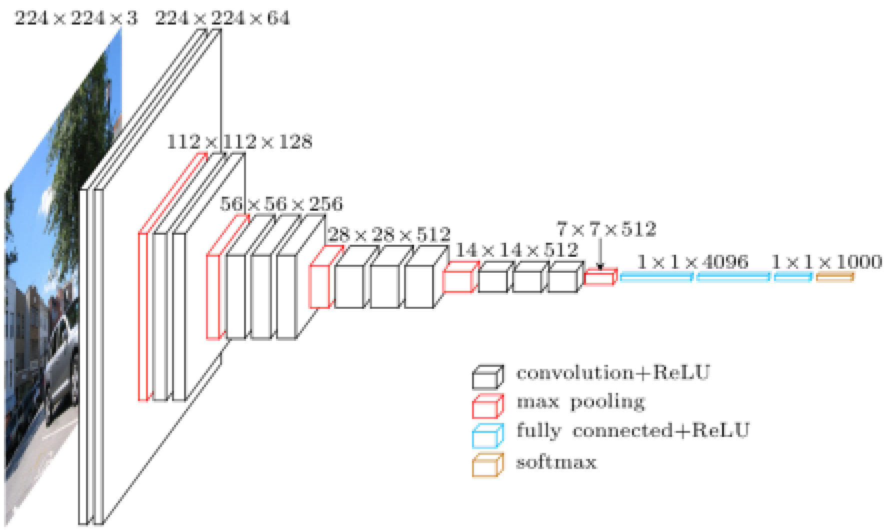
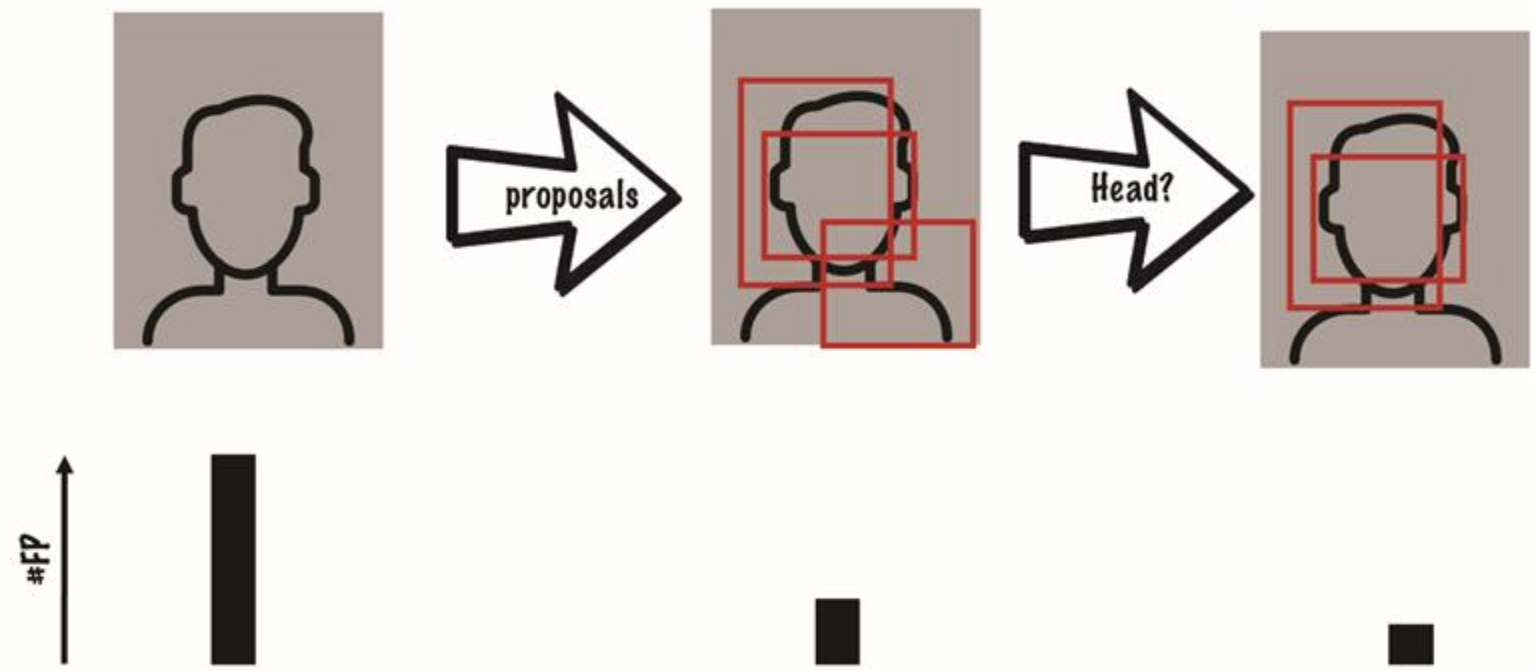


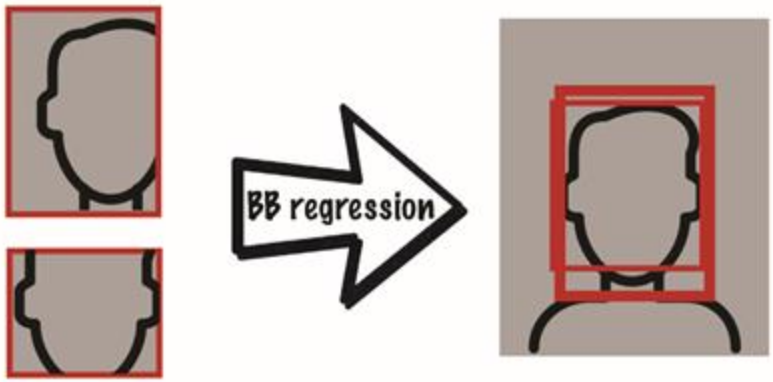
Image source: https://www.cs.toronto.edu/~frossard/post/vgg16/

What happens with FP?

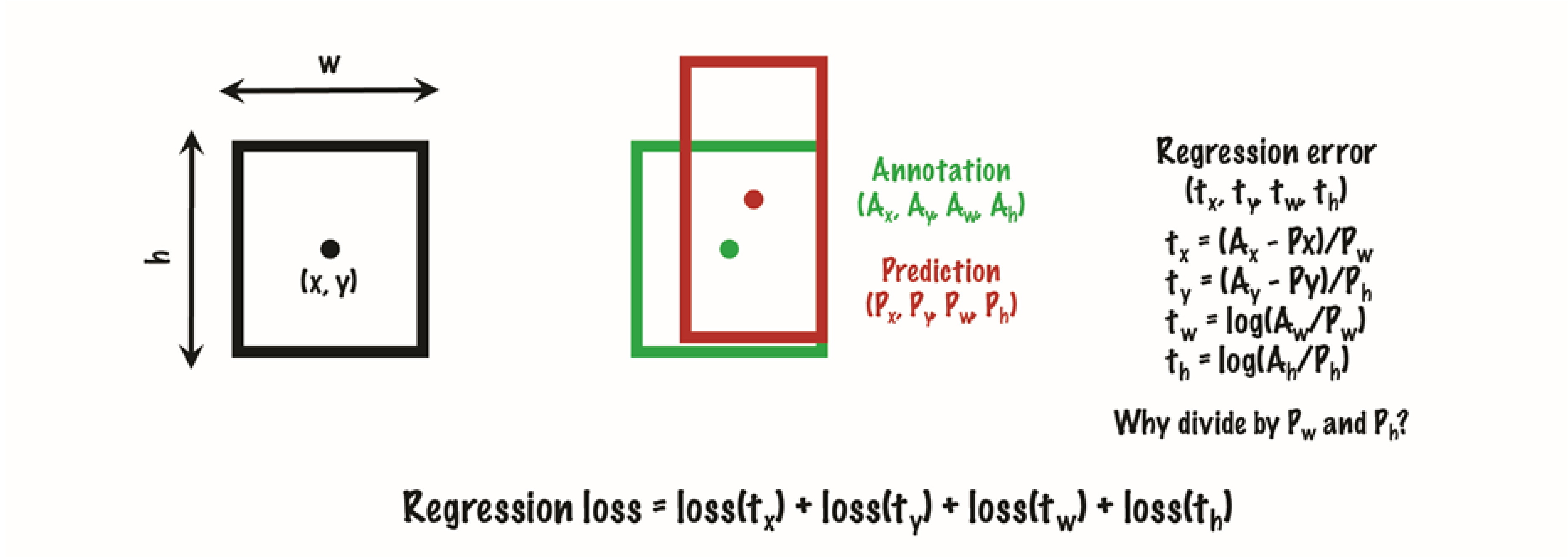


Regressor (Refinement)

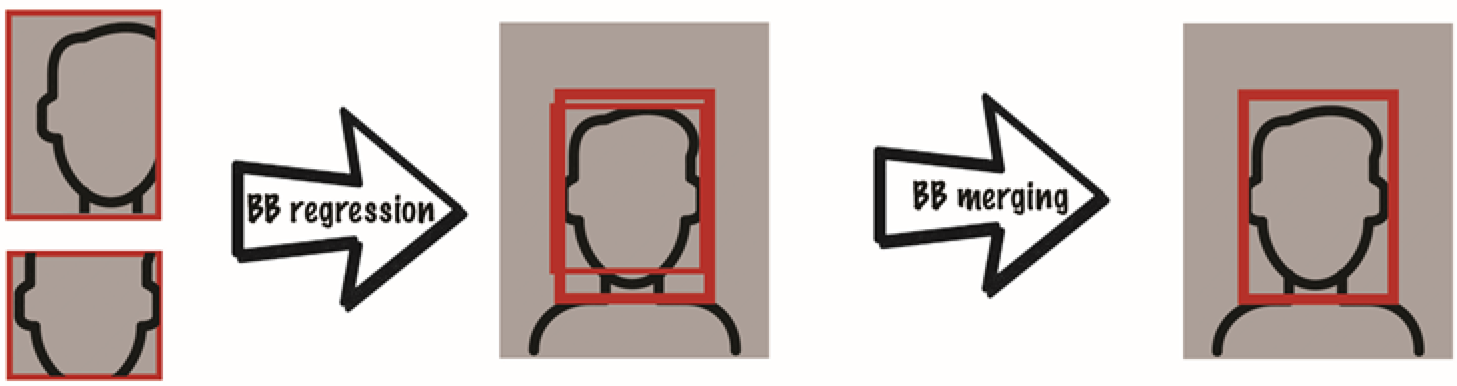
Regressor is used to adjust the position of class specific bounding boxes.



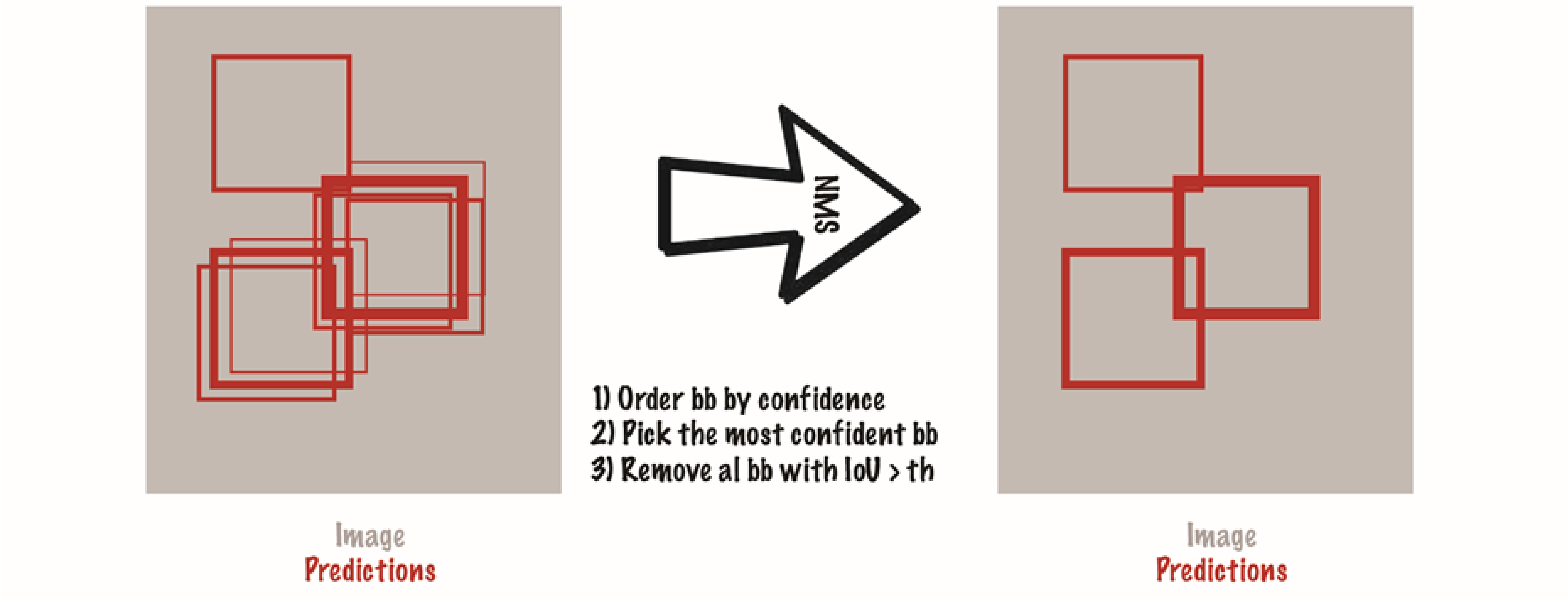
Bounding box regression



Regressor (Refinement)

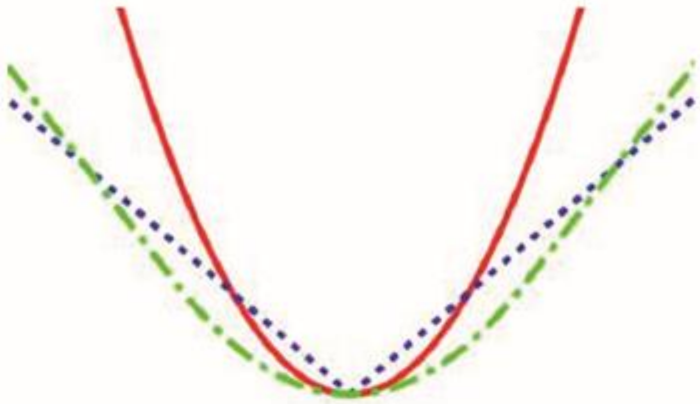


Non-Maximum Suppression (NMS)



Detection: Loss Functions

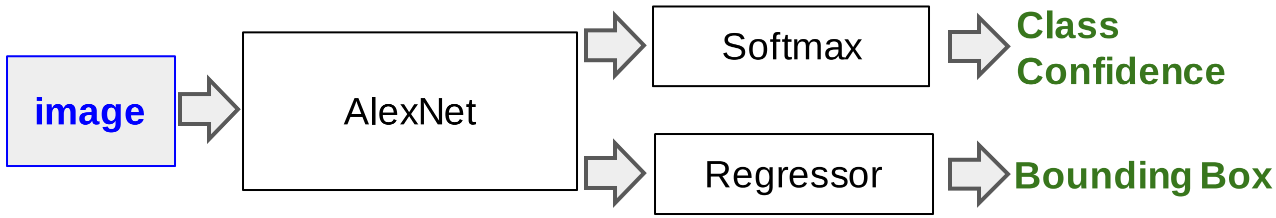
- Classification losses:
  - Cross entropy (softmax)
  - Hinge loss (SVM)
- Regression losses:
  - L1
  - Smooth L1
  - L2

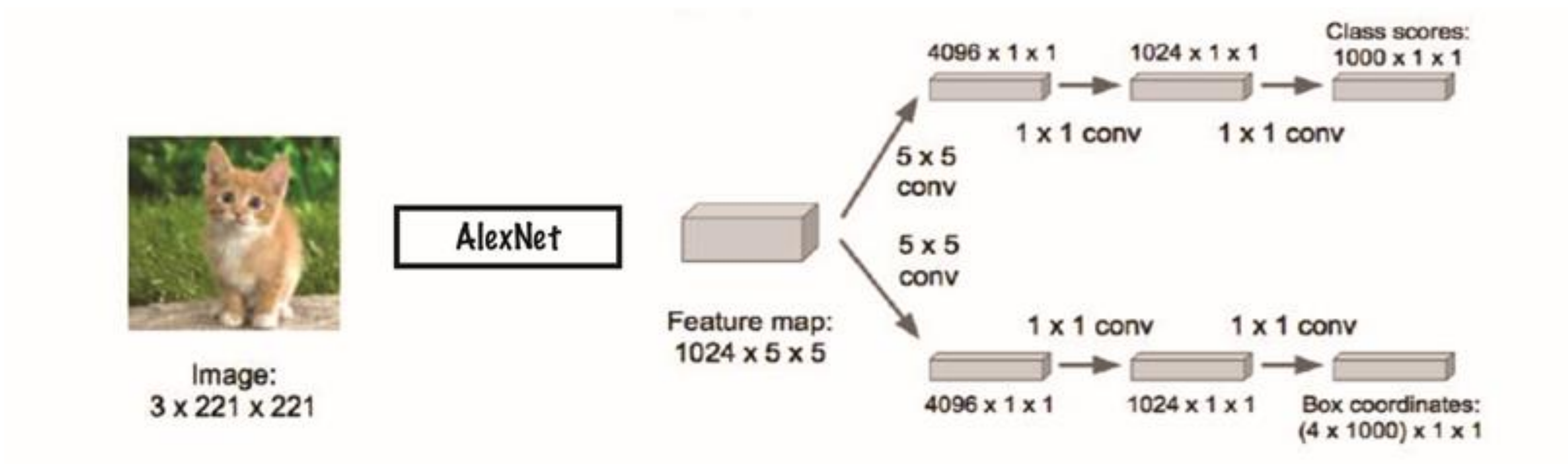


Deep learning for object detection: Outline

- Introduction
- Basic blocks and concepts
- Models

OverFeat (2013, ICLR2014)



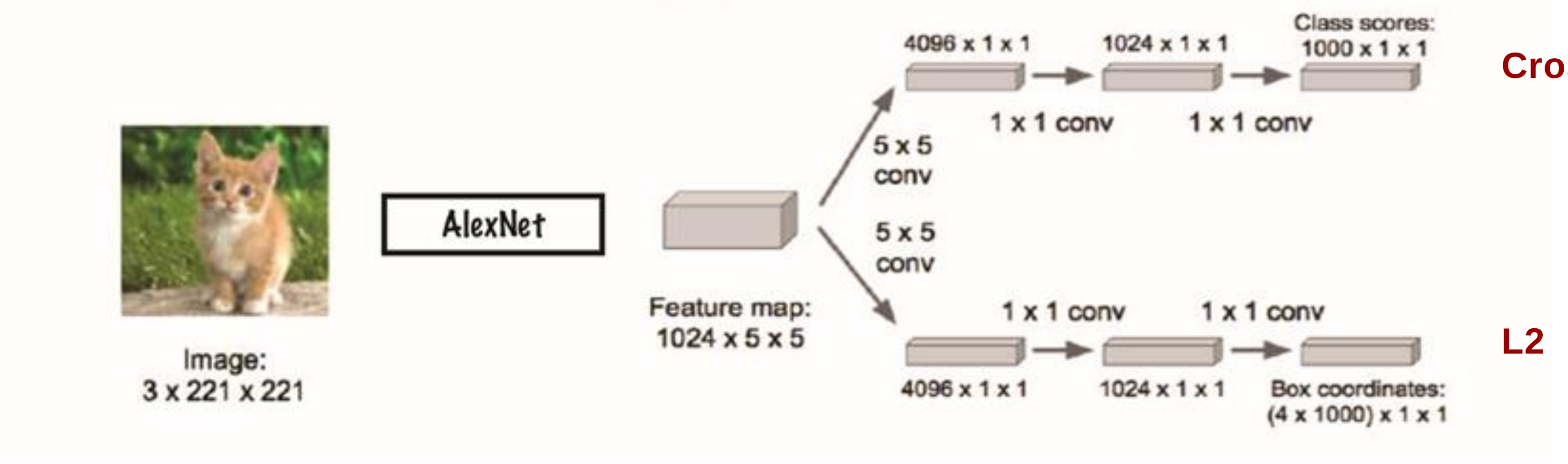


Sermanet, Pierre, et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks." arXiv preprint arXiv:1312.6229 (2013).

OverFeat: Training

Two stage training:

1. Train the classifier (cross entropy)
2. Train the regressor (L2)



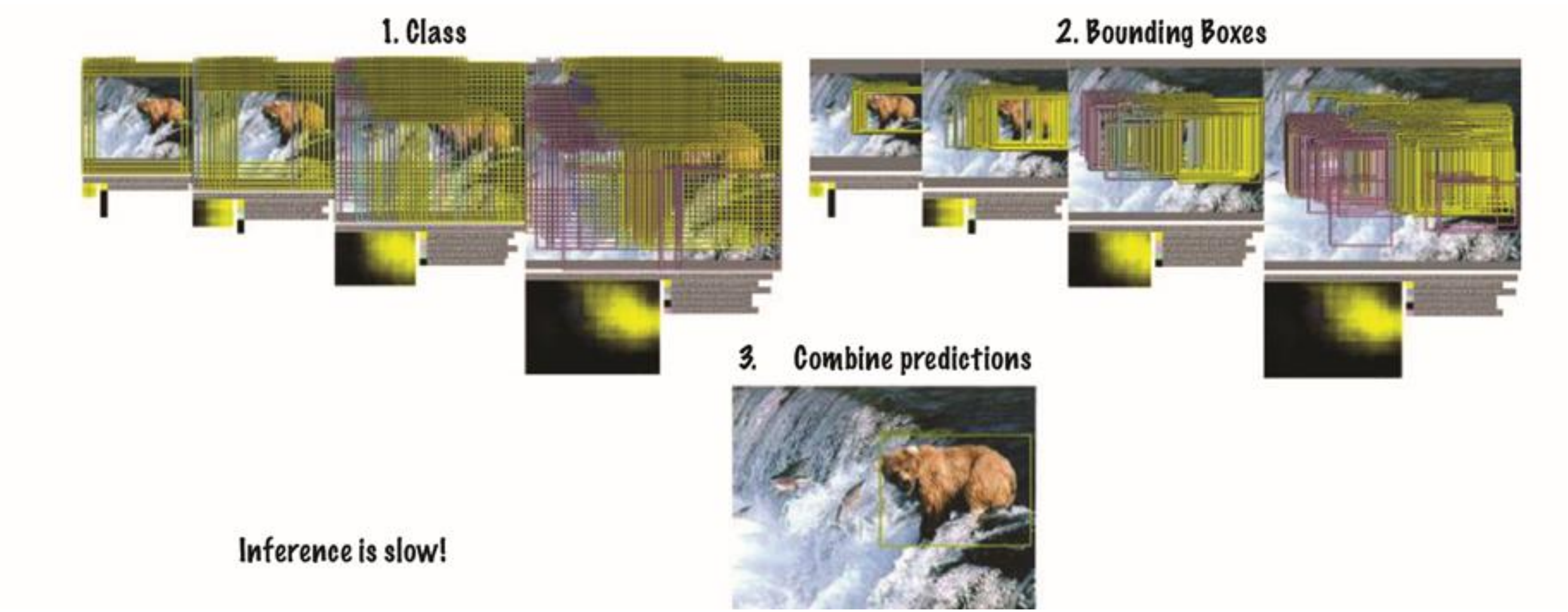
Cross entropy

L2

Sermanet, Pierre, et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks." arXiv preprint arXiv:1312.6229 (2013).

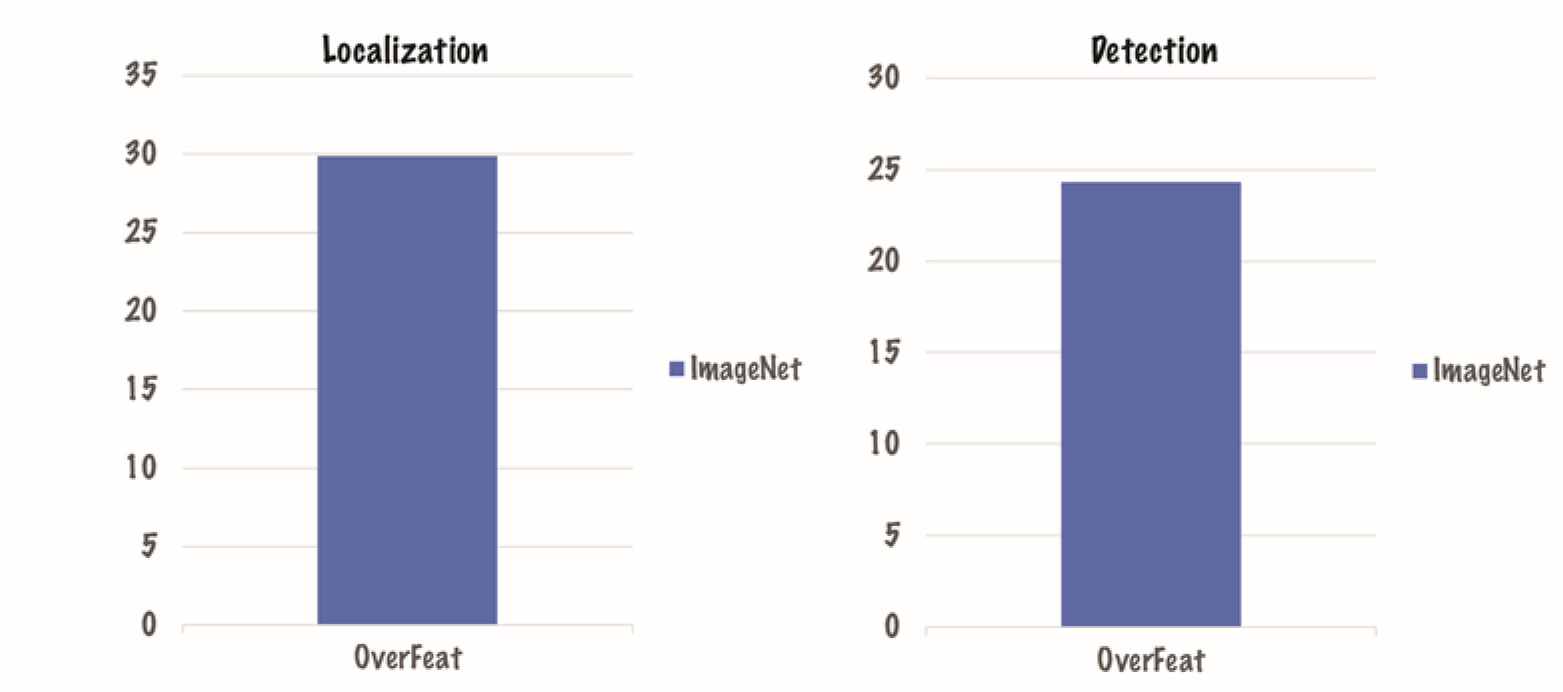
OverFeat: Inference

Apply the network at all positions and scales and predict:



[Sermanet, Pierre, et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks." arXiv preprint arXiv:1312.6229 (2013).]

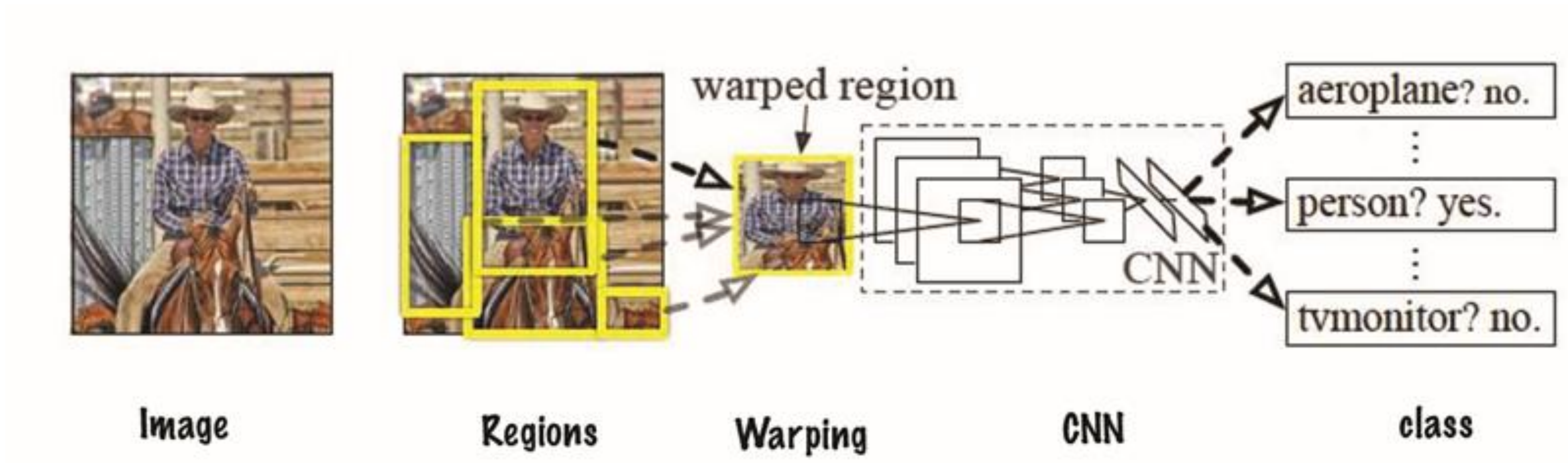
OverFeat: Results



R-CNN (2013, CVPR2014)

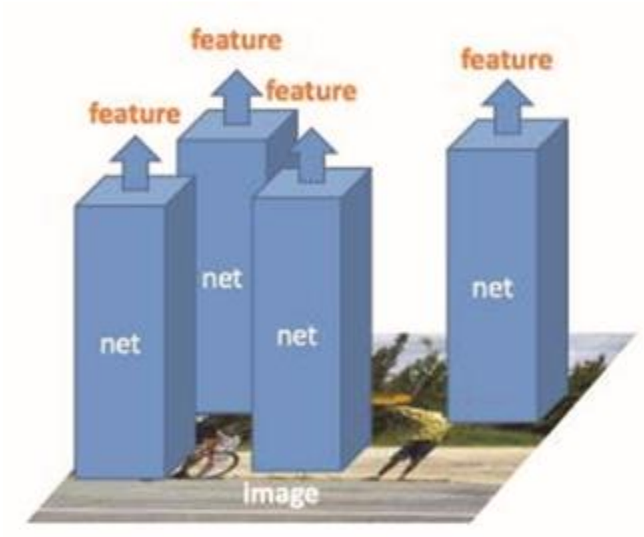
image → Region Proposals → Warped region → AlexNet/VGG → SVM → Class Confidence

AlexNet/VGG → Regressor → Bounding Box



[Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.]

R-CNN: Training

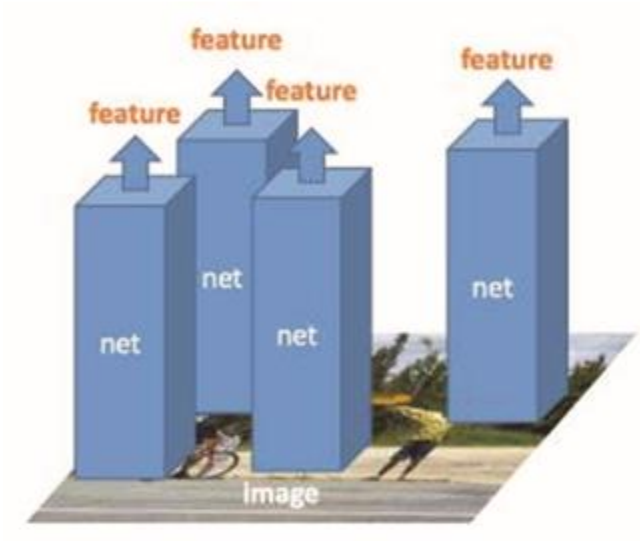


1. Pre-train network on Imagenet (image classification task)
2. Finetune network with softmax classifier (log loss)
3. Extract features
4. Train linear SVMs with hard negative mining (hinge loss)
5. Train bounding box regressions (least squares)

Training is slow (84h) !!!!

Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

R-CNN: Inference

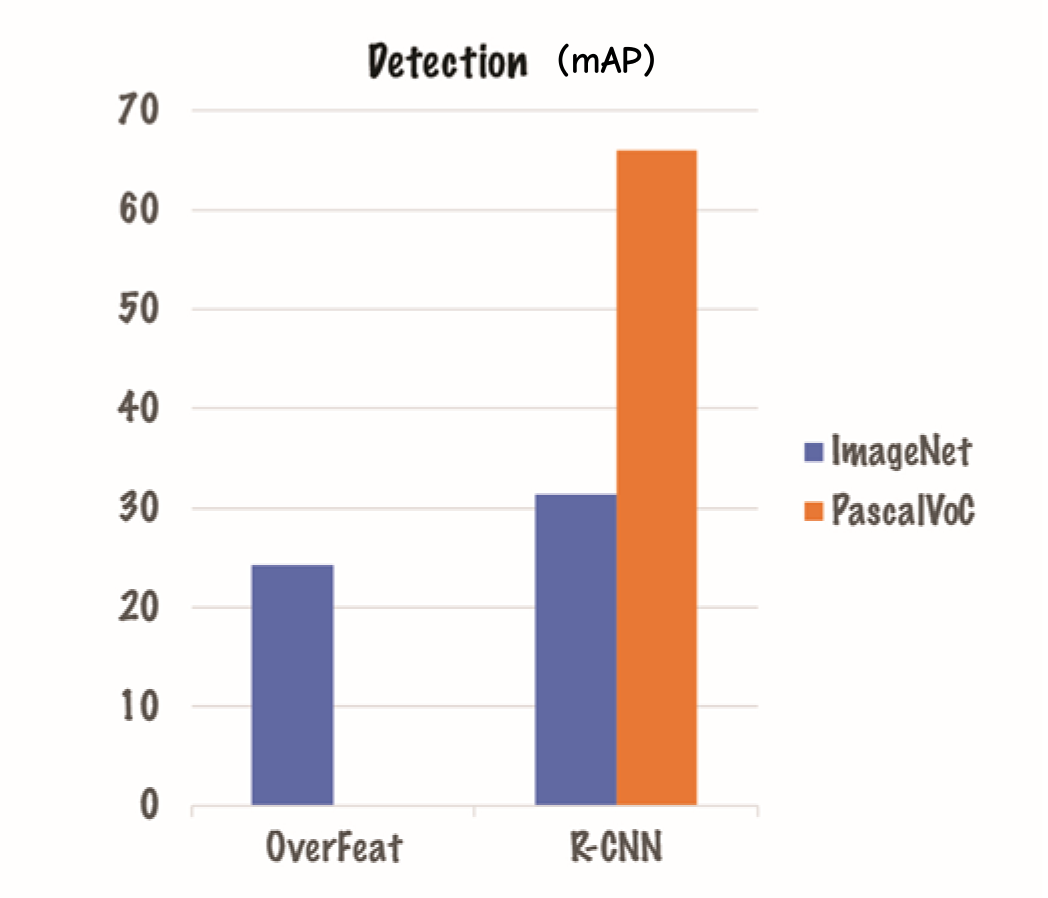


1. Extract 2000 region proposals per image
2. Extract features for each proposal
3. Infer class, confidence and bounding box for each proposal

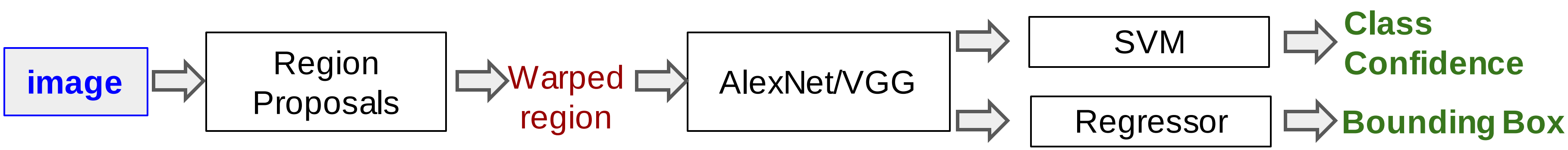
Inference is slow (2k passes of CNN per image).

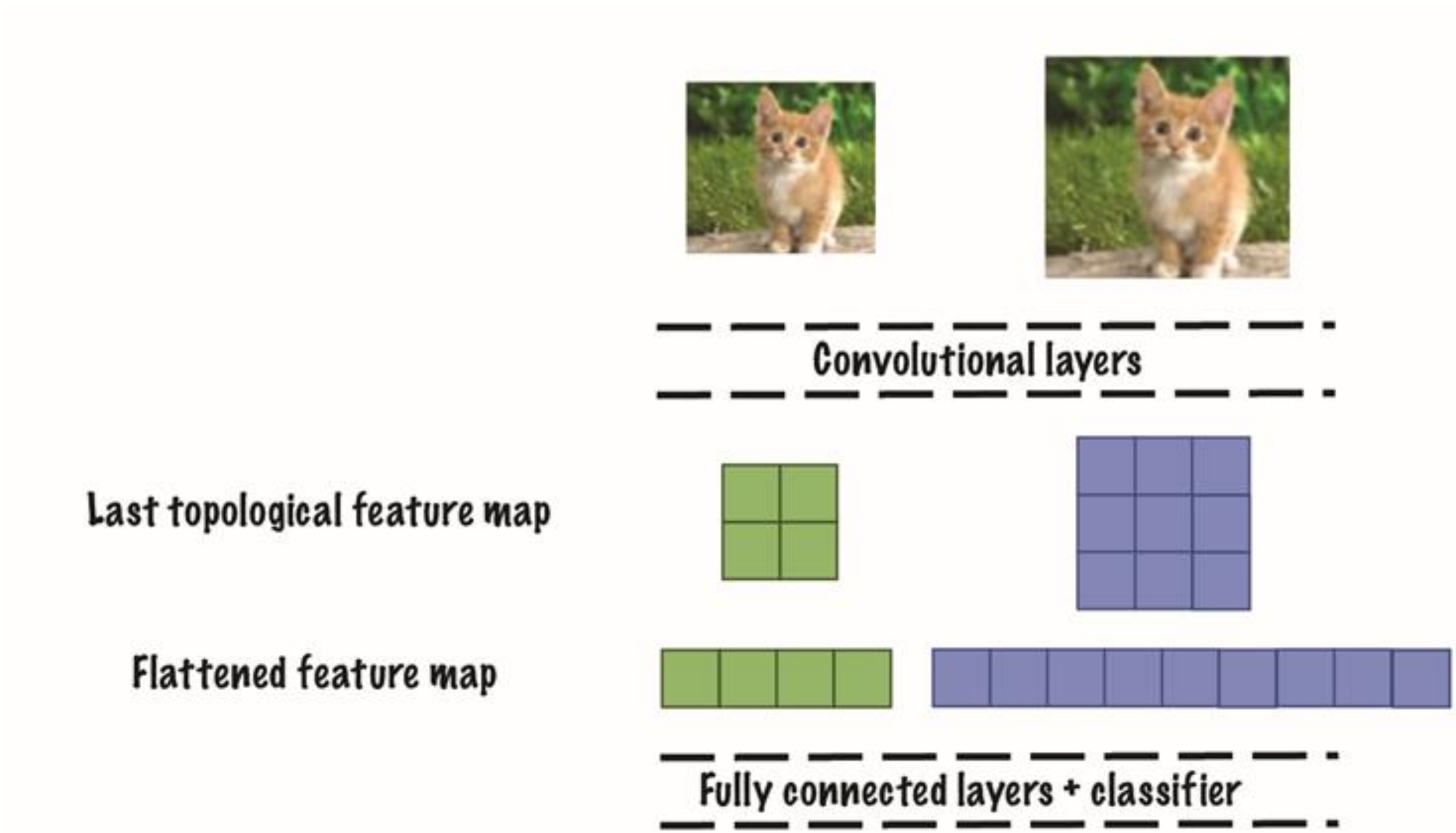
Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

R-CNN: Results

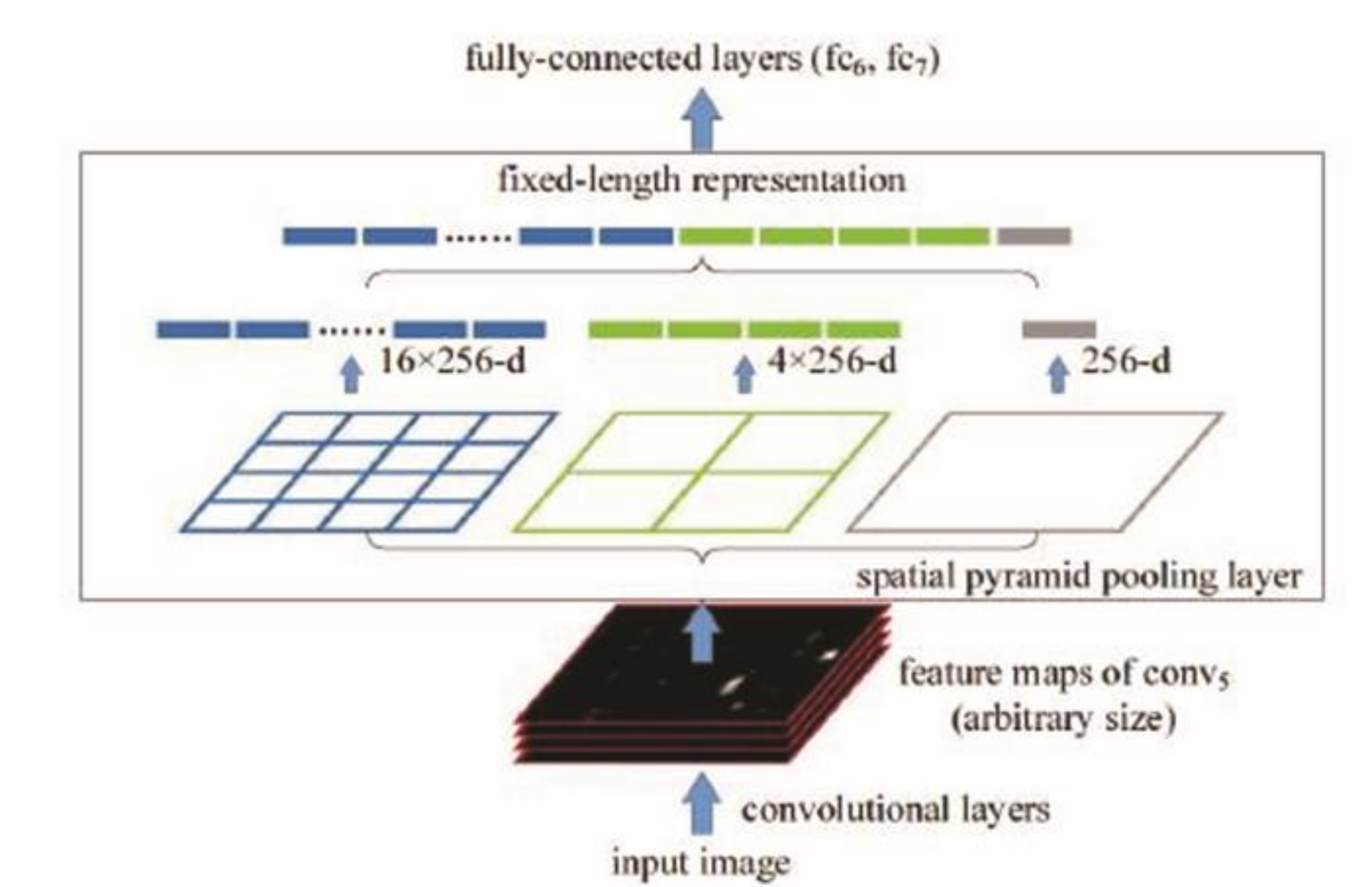


Why do we need to warp images?





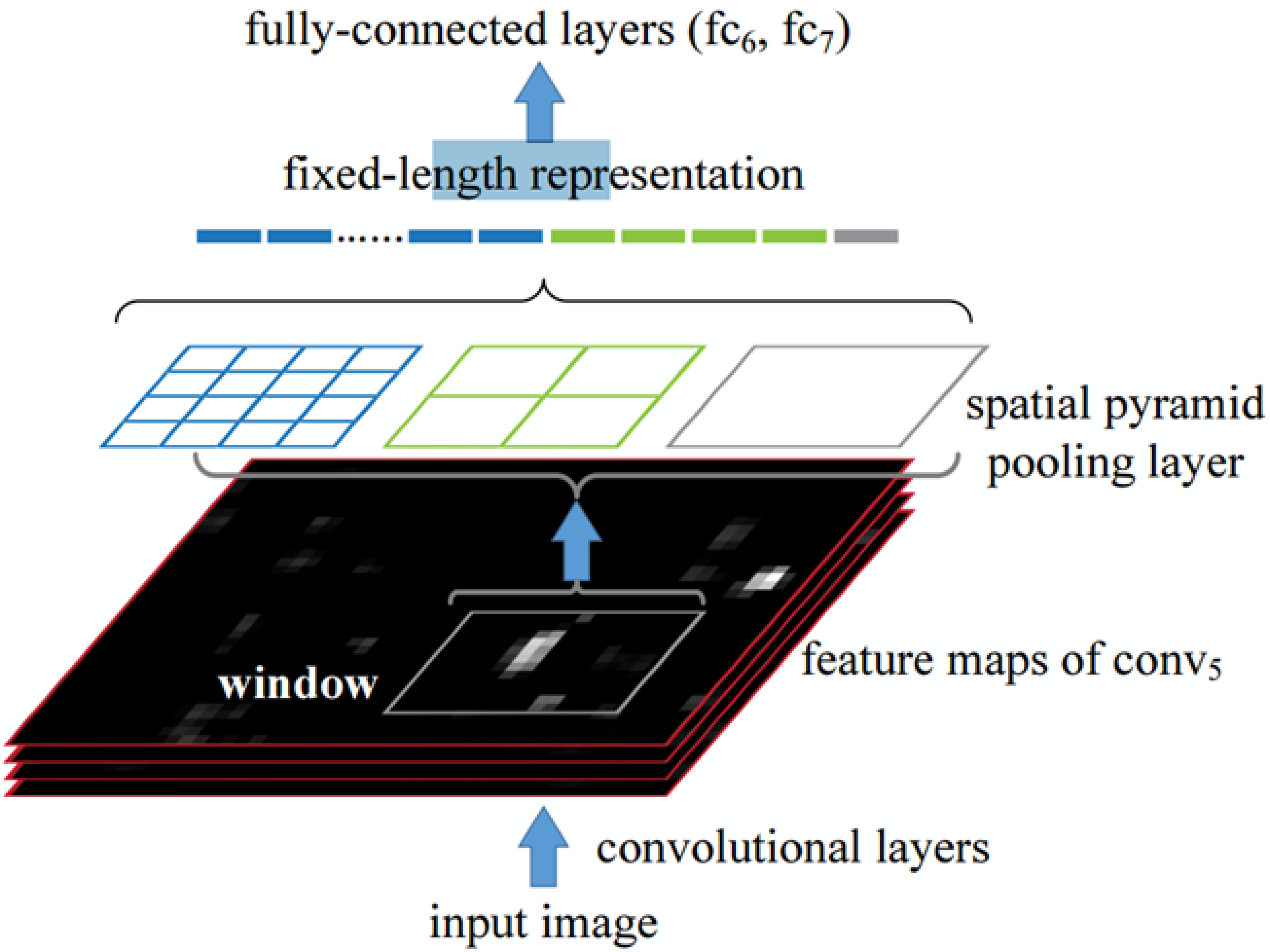
Spatial Pyramid Pooling (2014, TPAMI2015)



http://kaiminghe.com/eccv14sppnet/sppnet_ilsvrc2014.pdf

He, Kaiming, et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition." IEEE transactions on pattern analysis and machine intelligence 37.9 (2015).

Spatial Pyramid Pooling (2014, TPAMI2015)

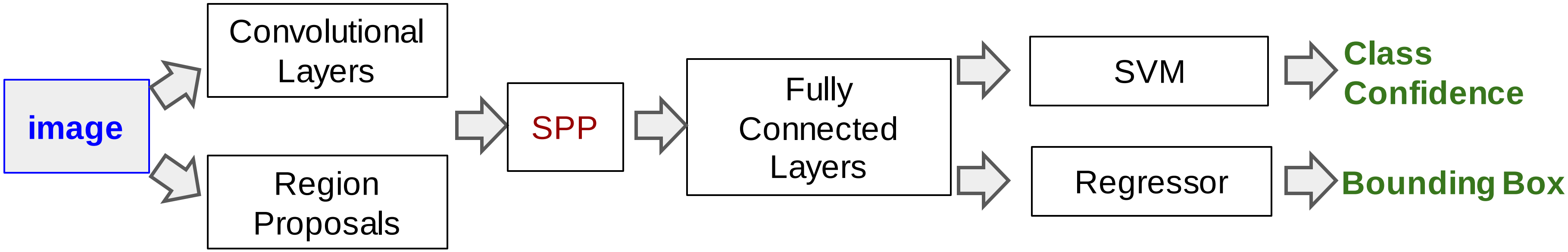


[He, Kaiming, et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition." IEEE transactions on pattern analysis and machine intelligence 37.9 (2015).]()

Spatial Pyramid Pooling (2014, TPAMI2015)

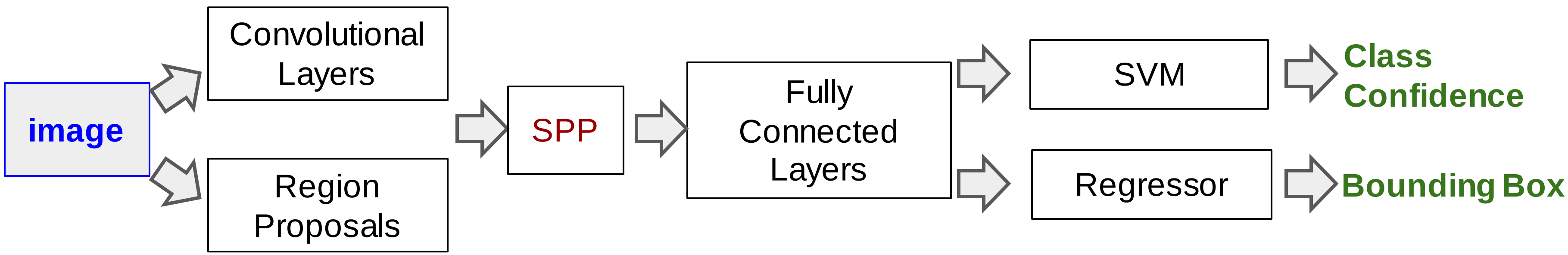
R-CNN

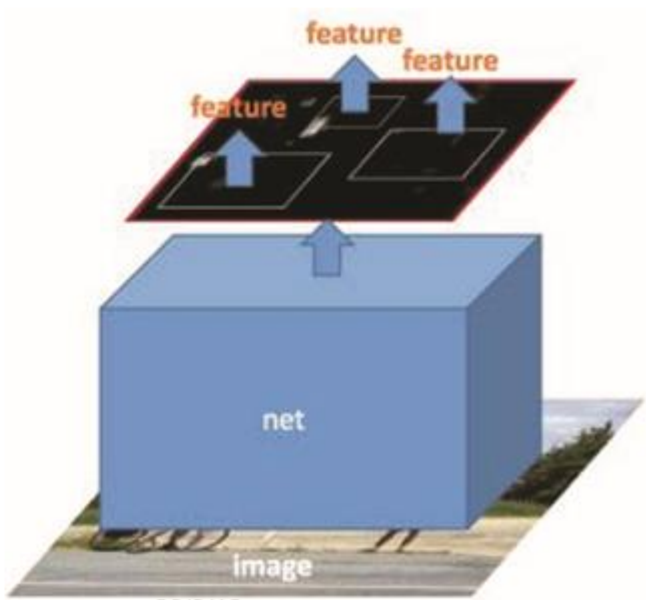
SPP



He, Kaiming, et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition." IEEE transactions on pattern analysis and machine intelligence 37.9 (2015).

SPP: Training



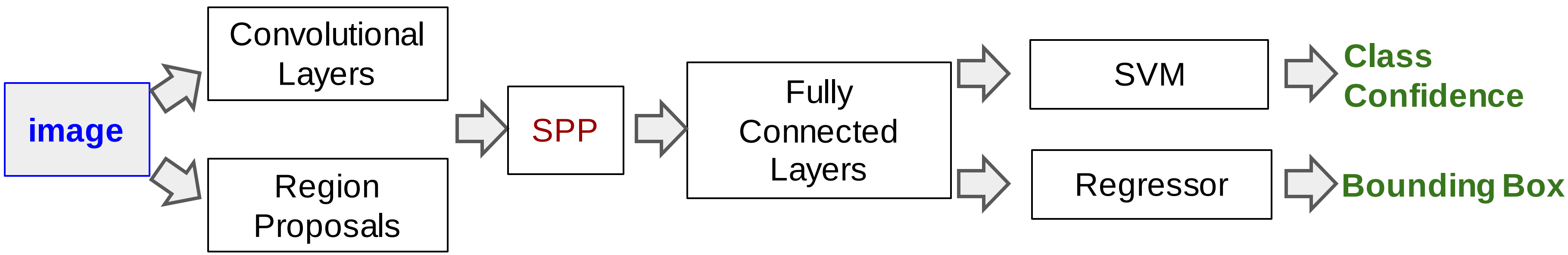


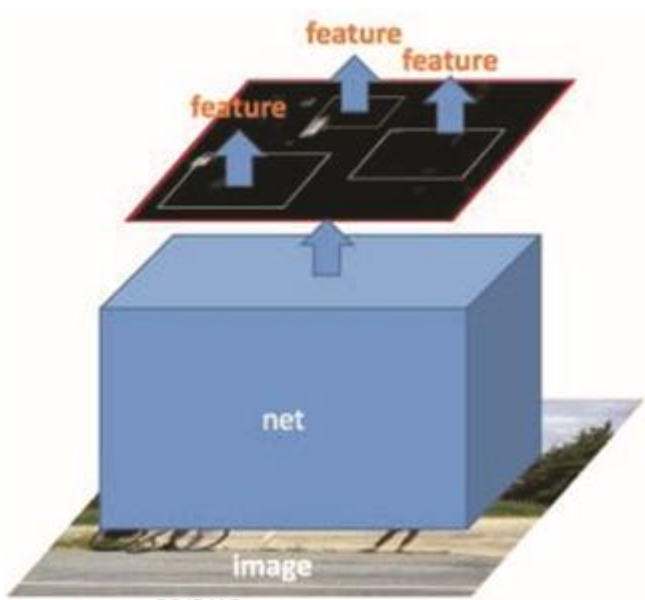
1. Pre-train network on Imagenet (image classification task) (Including Conv + SPP + FC layers)
1. Finetune network with softmax classifier (log loss)
2. Extract features
3. Train linear SVMs with hard negative mining (hinge loss)
4. Train bounding box regressions (least squares)

Training is (faster that R-CNN but) still slow!

[He, Kaiming, et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition." IEEE transactions on pattern analysis and machine intelligence 37.9 (2015).]

SPP: Inference



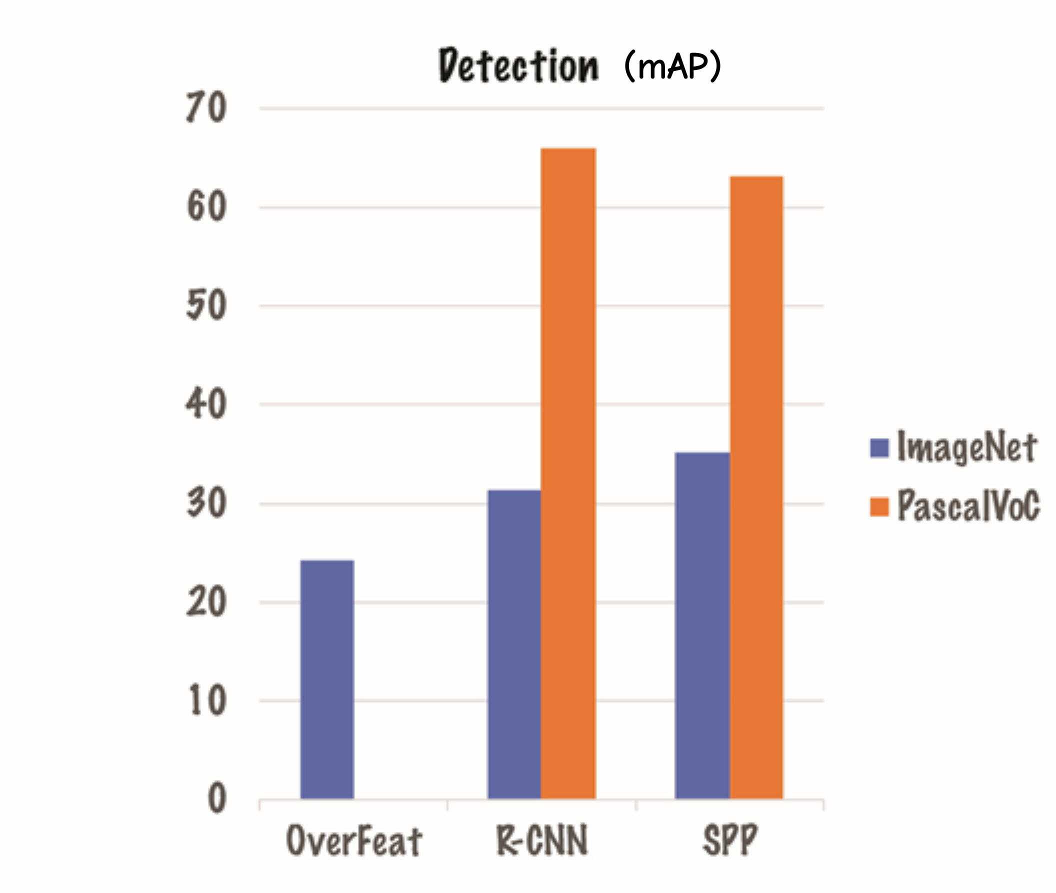


1. Compute the feature map of an image
2. Compute region proposals (2k per image)
3. Project region proposals into feature map
4. Infer class, confidence and bounding box for each proposal

Inference time is OK (up to 100x faster than R-CNN).

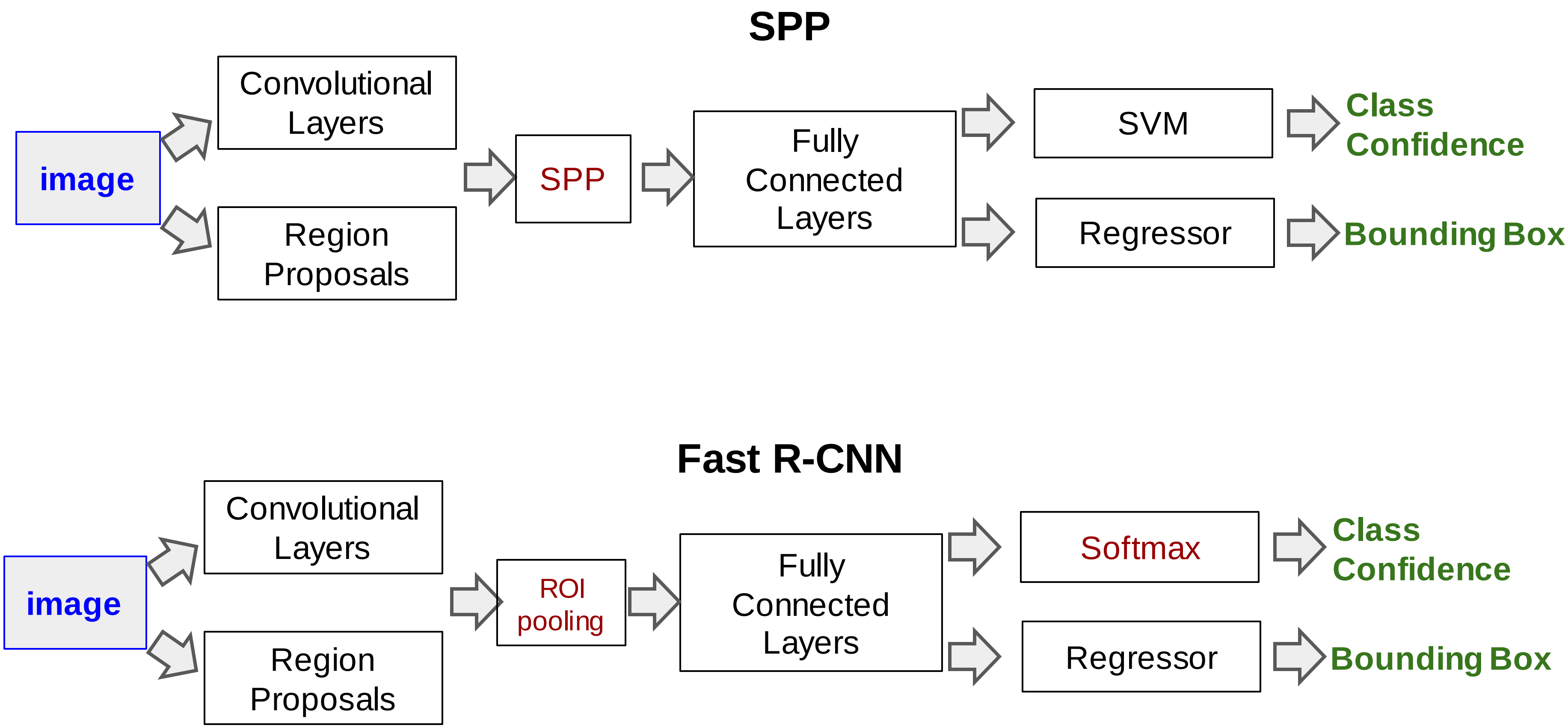
[He, Kaiming, et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition." IEEE transactions on pattern analysis and machine intelligence 37.9 (2015).]

SPP: Results



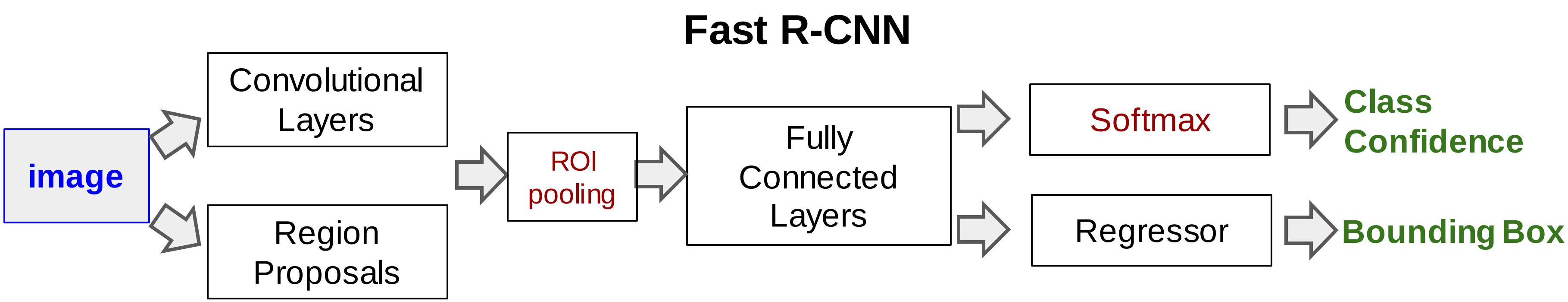
Detection (mAP)
70
60
50
40
30
20
10
0
OverFeat
R-CNN
SPP
ImageNet
PascalVoC

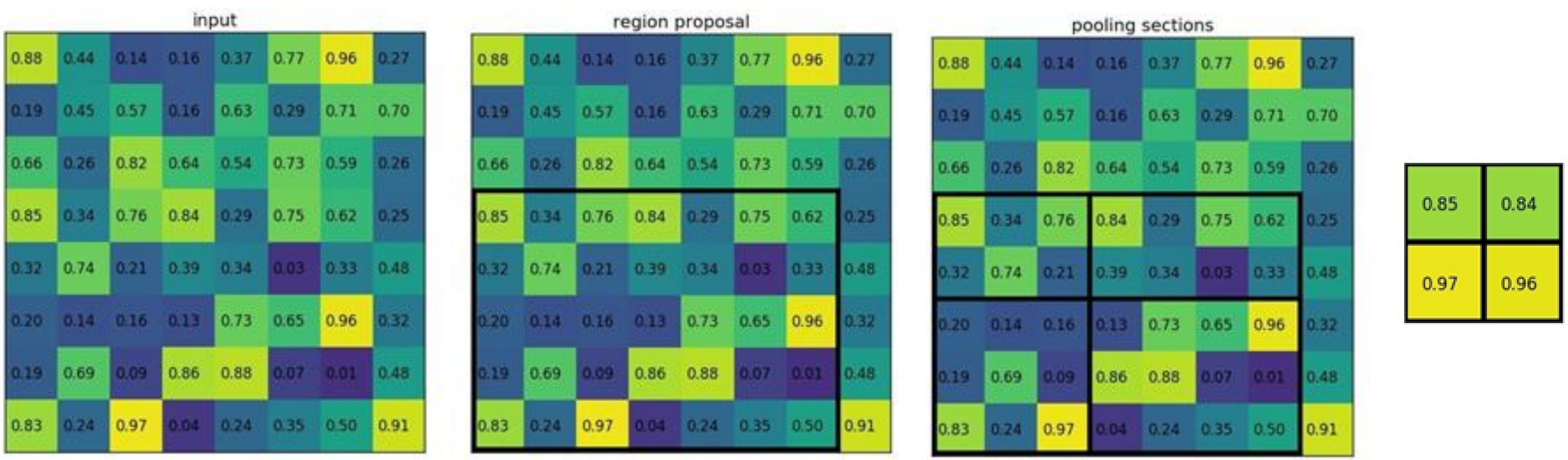
Fast R-CNN (2015, ICCV2015)



Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.

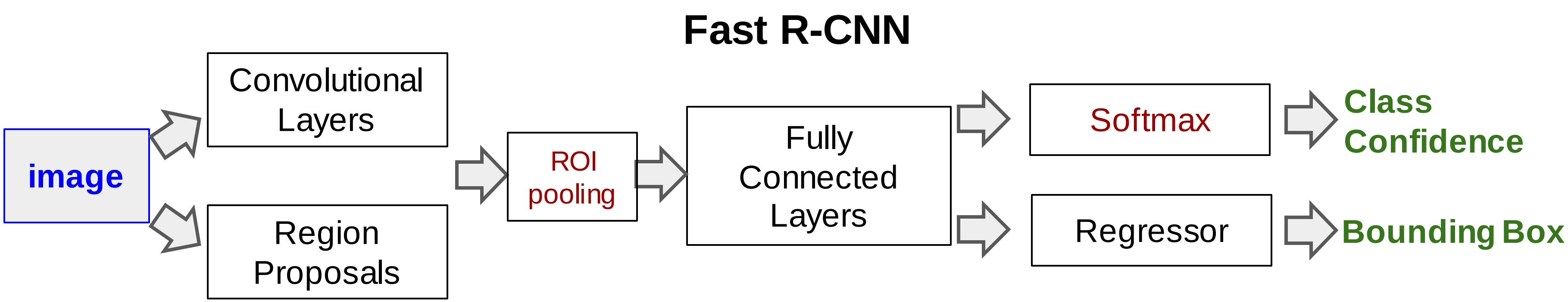
Fast R-CNN: ROI Pooling

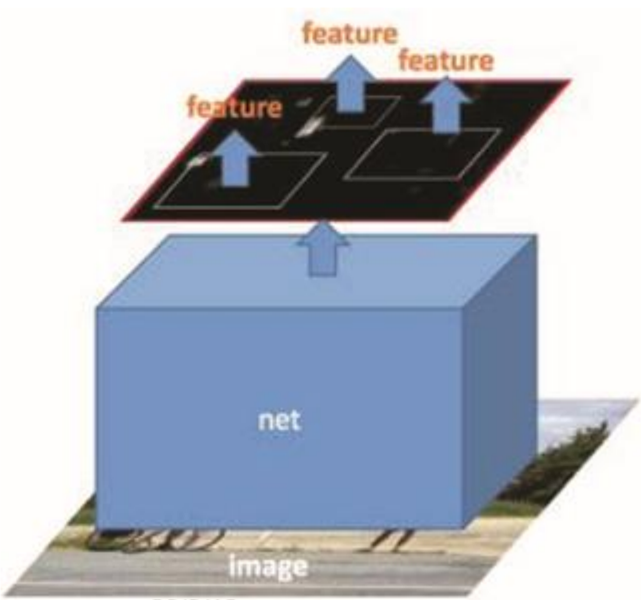




https://deepsense.ai/region-of-interest-pooling-explained/

Fast R-CNN: Training





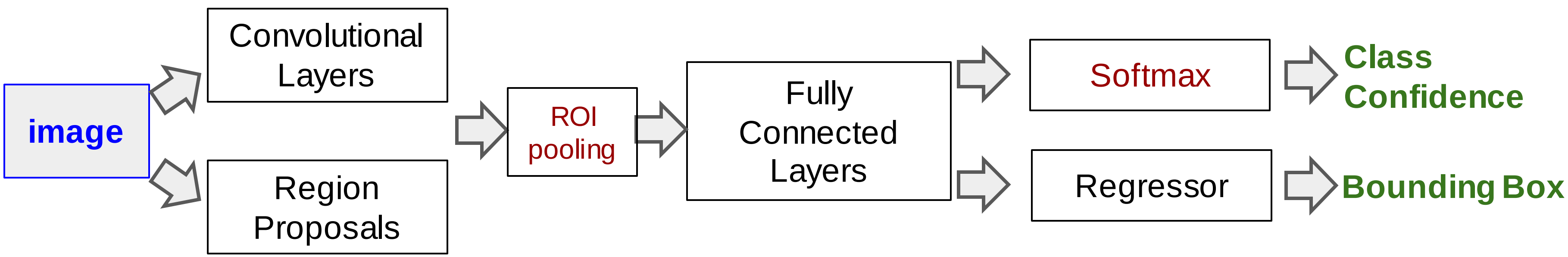
Joint loss: log loss + smooth L1 loss

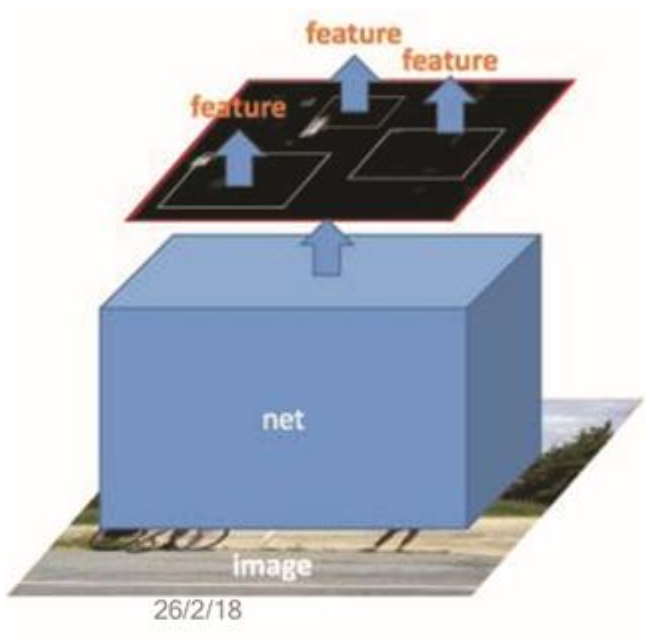
1. Pre-train network on Imagenet classification task
2. Train the model with joint loss

Training is elegant and fast.
Region Proposals are still required....

Fast R-CNN: Inference

Fast R-CNN



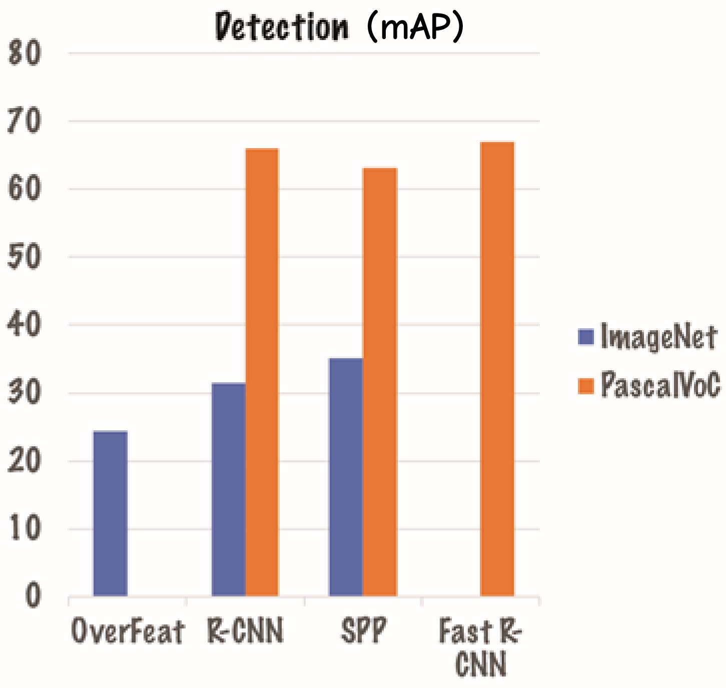


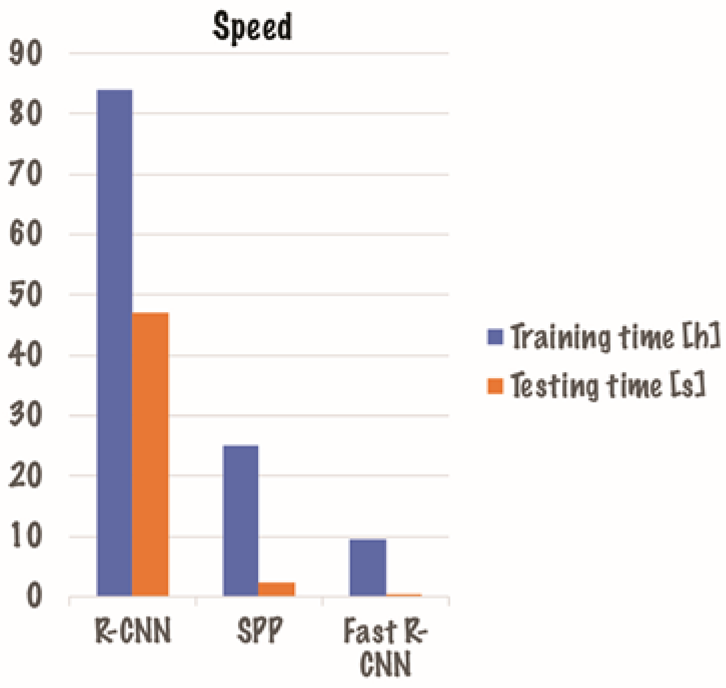
1. Extract feature map
2. Extract region proposals
3. Infer class, confidence and bounding box for each proposal

Inference is fast.

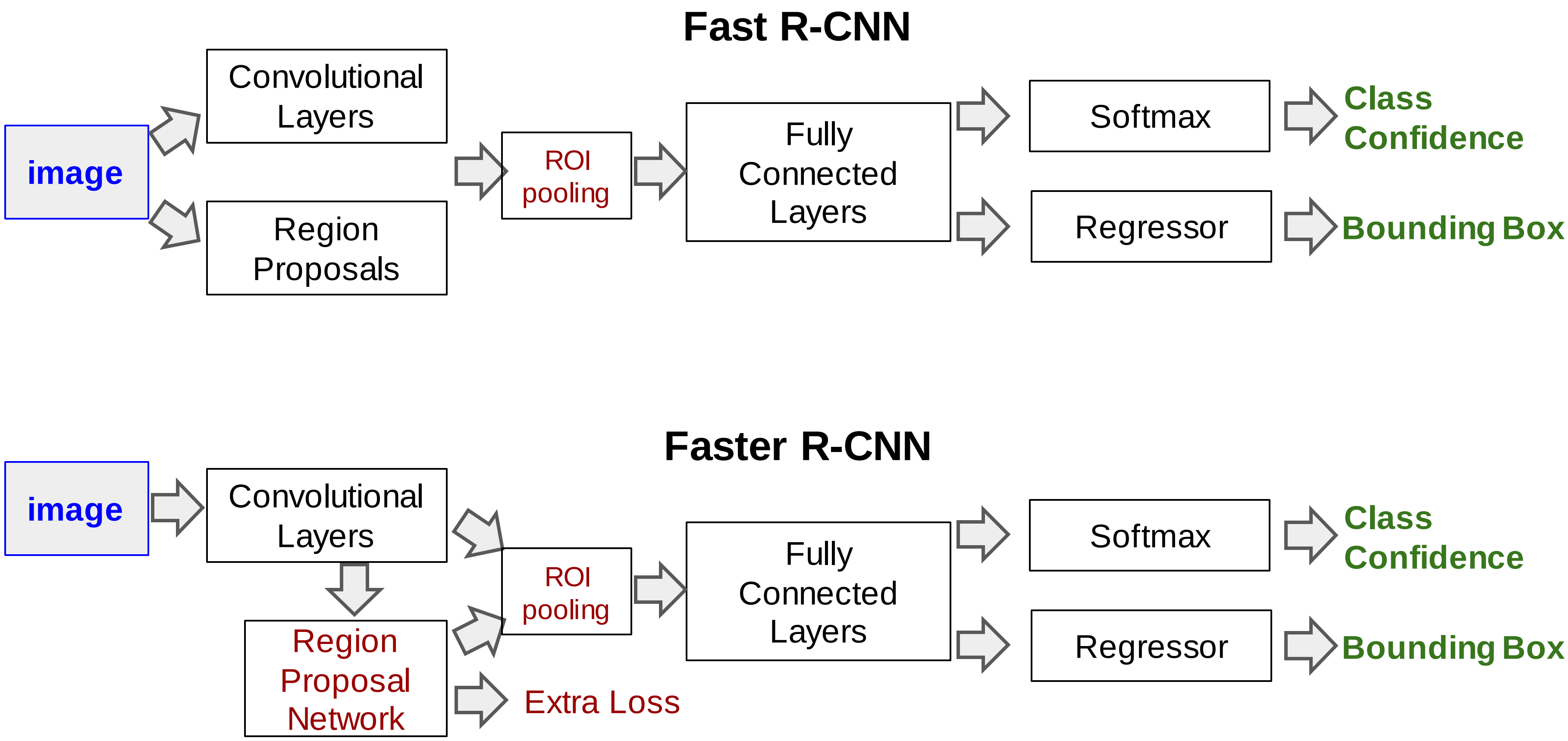
Now the bottleneck is in Selective Search!

Fast R-CNN: Results





Faster R-CNN (2015, NIPS2015)

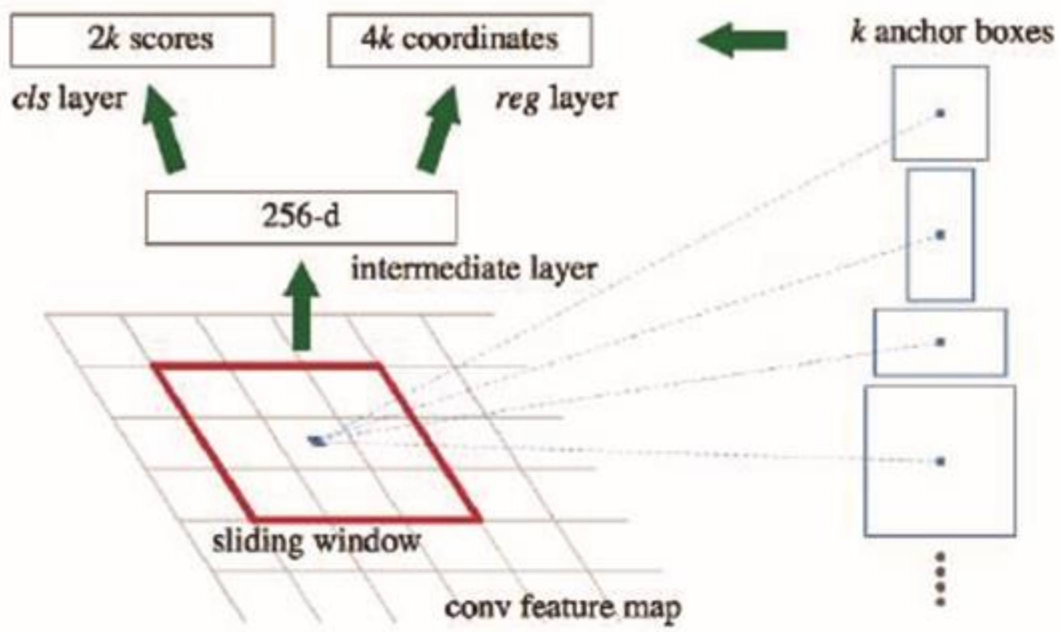


[Ren, Shaoqing, et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." *NIPS*. 2015.]

Faster R-CNN: Region Proposal Network (RPN)

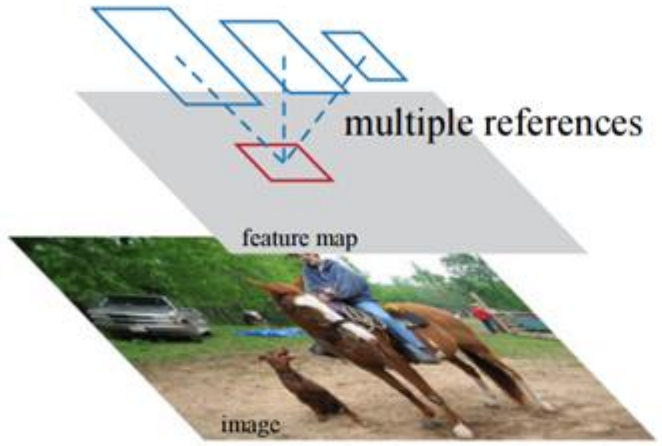
Anchor box: pre-defined "template" bounding box.

The RPN predicts **how good is the Region Proposal** resulting from placing each of the anchor boxes at each of the cells of the feature map (cls layer), and tries to **adjust its coordinates** to better match an overlapping GT object (reg layer).



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

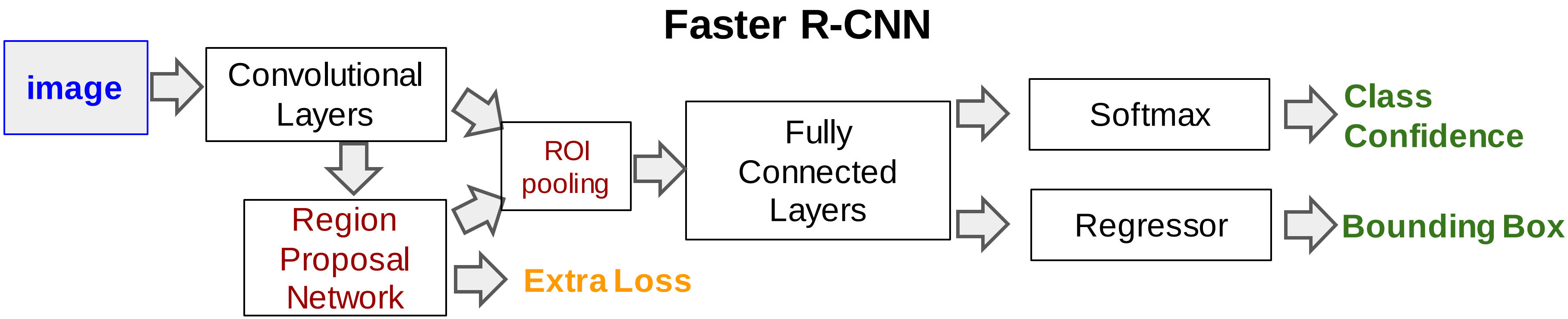
Faster R-CNN: Region Proposal Network (RPN)

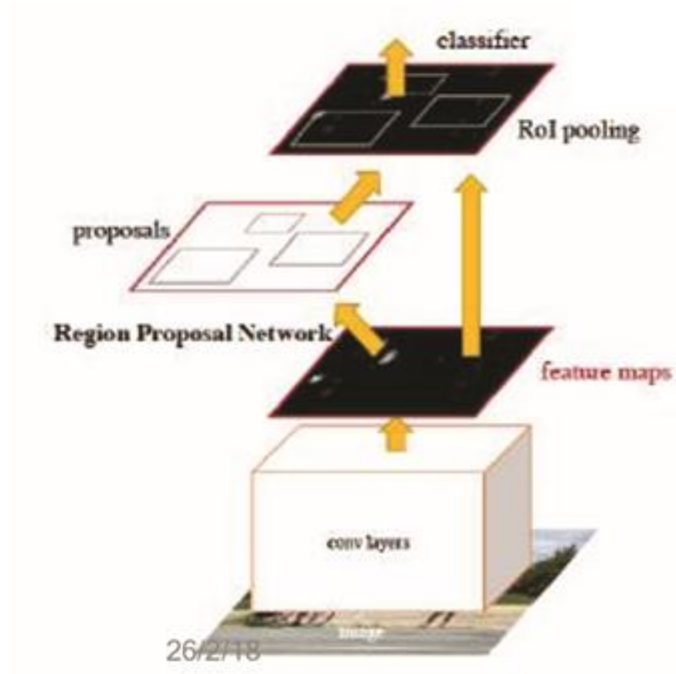


- The k proposals are parameterized relative to k reference boxes (Anchors) generated with cluster analysis over the training set.
- An anchor is centered at the sliding window in question, and is associated with a scale and aspect ratio.
- The original Faster R-CNN uses 3 scales and 3 aspect ratios by default, yielding k= 9 anchors at each sliding position.

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

Faster R-CNN: Training



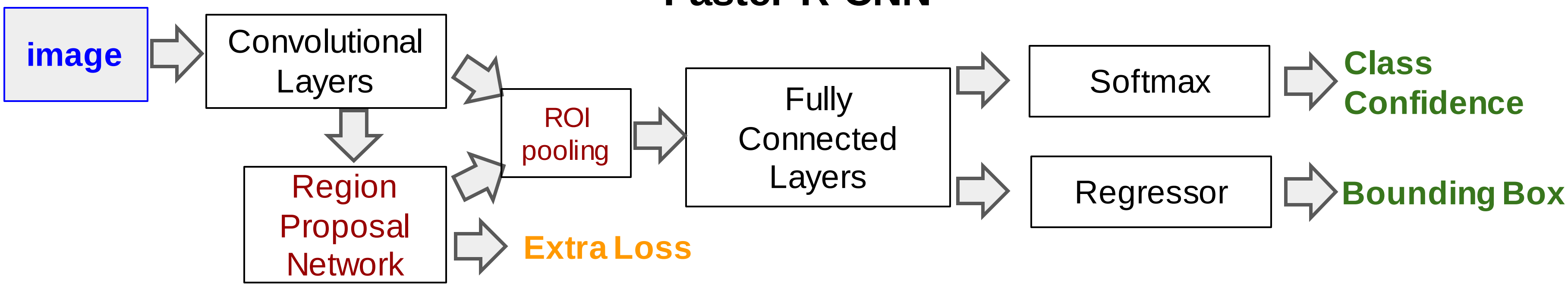


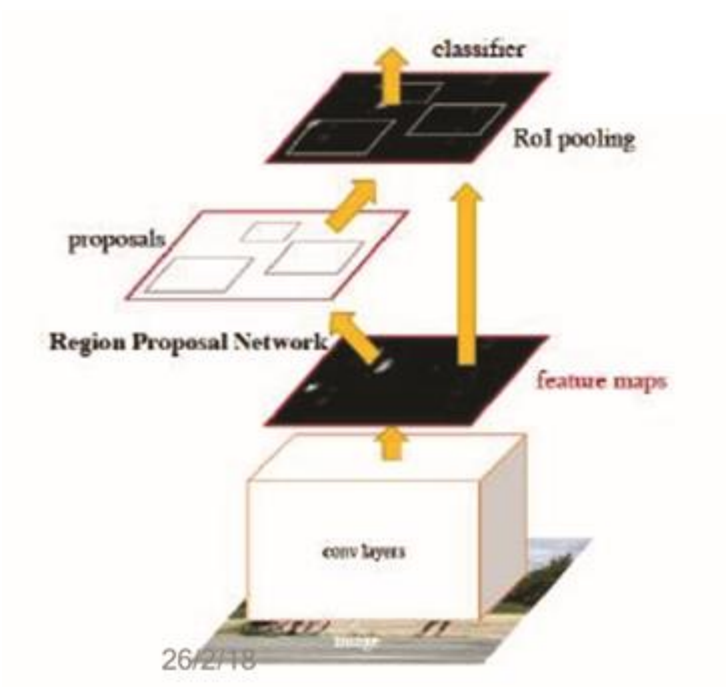
One network, four losses (TPAMI version)

1. **RPN classification (anchor good / bad)**
2. **RPN regression (anchor -> proposal)**
3. Fast R-CNN classification (over classes)
4. Fast R-CNN regression (proposal -> box)

Faster R-CNN: Inference

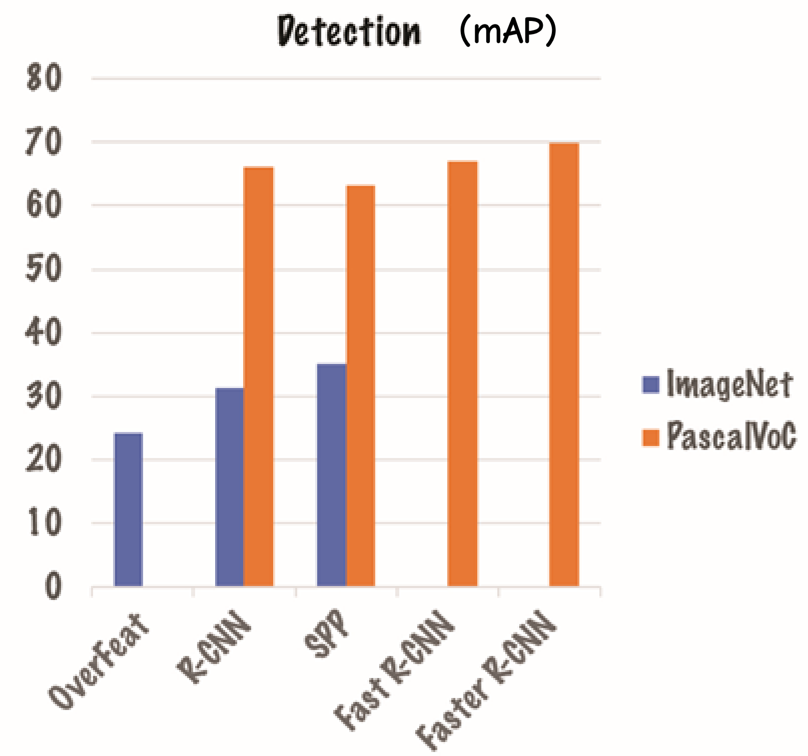
Faster R-CNN

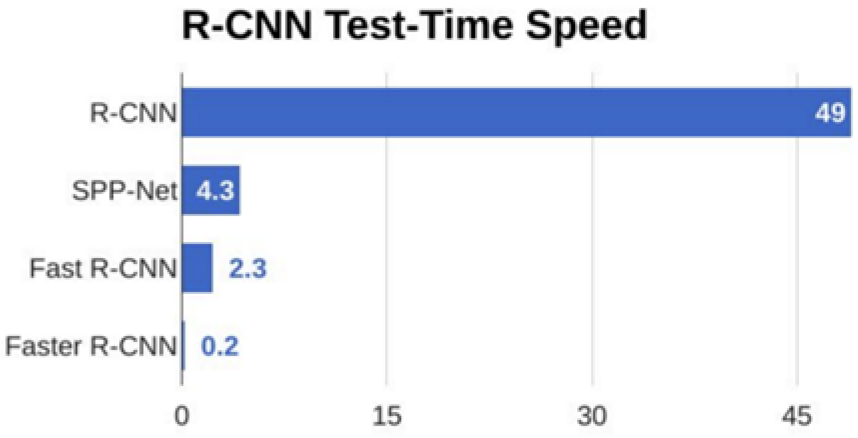




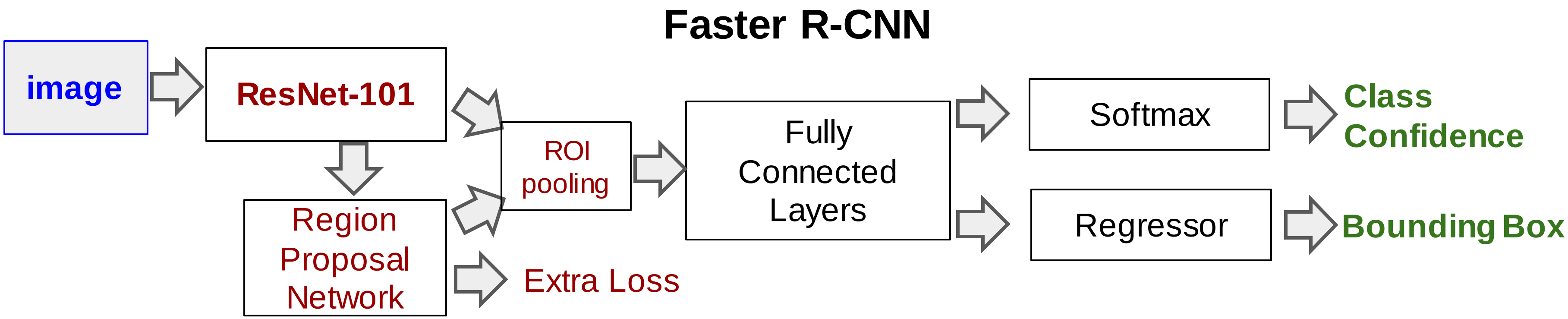
1. Extract feature map and region proposals
2. Infer class, confidence and bounding box for each proposal

Faster R-CNN: Results





Faster R-CNN + ResNets (2015, CVPR2016)



Faster R-CNN baseline	mAP@.5	mAP@.5:.95
VGG-16	41.5	21.5
ResNet-101	**48.4**	**27.2**

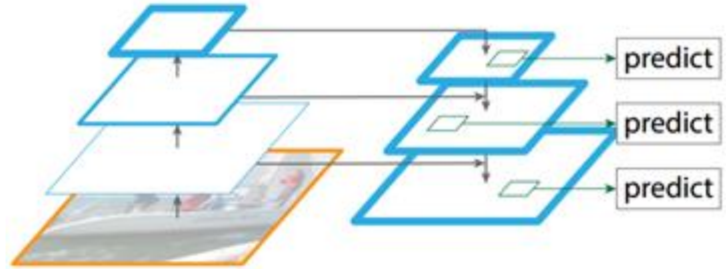
COCO detection results

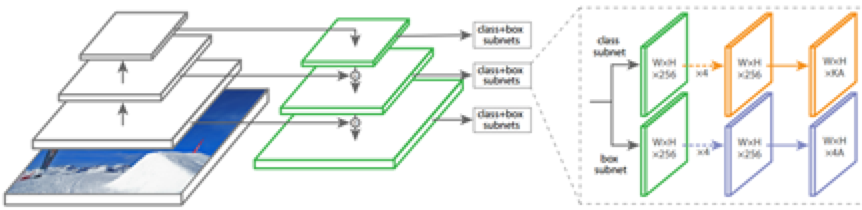
Other ideas (covered in next class)

Feature Pyramid Networks for Object Detection

Tsung-Yi Lin[1,2], Piotr Dollár[1], Ross Girshick[1],
Kaiming He[1], Bharath Hariharan[1], and Serge Belongie[2]

[1]Facebook AI Research (FAIR)
[2]Cornell University and Cornell Tech



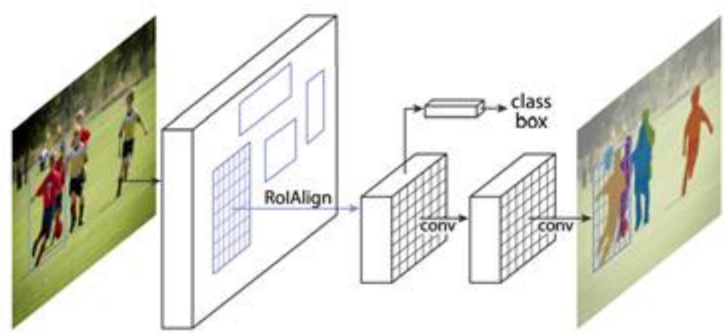


Focal Loss for Dense Object Detection

Tsung-Yi Lin Priya Goyal Ross Girshick Kaiming He Piotr Dollár
Facebook AI Research (FAIR)

Mask R-CNN

Kaiming He Georgia Gkioxari Piotr Dollár Ross Girshick
Facebook AI Research (FAIR)



References

- Ross B. Girshick, Jeff Donahue, Trevor Darrell and Jitendra Malik; Rich feature hierarchies for accurate object detection and semantic segmentation.
- Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, Yann LeCun; OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks.
- Kaiming He and Xiangyu Zhang and Shaoqing Ren and Jian Sun; Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition.
- Ross B. Girshick; Fast R-CNN.
- Shaoqing Ren, Kaiming He, Ross B. Girshick and Jian Sun; Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; Deep Residual Learning for Image Recognition.
- M. Oquab and L. Bottou and I. Laptev and J. Sivic; Is object localization for free? - Weakly-supervised learning with convolutional neural networks.