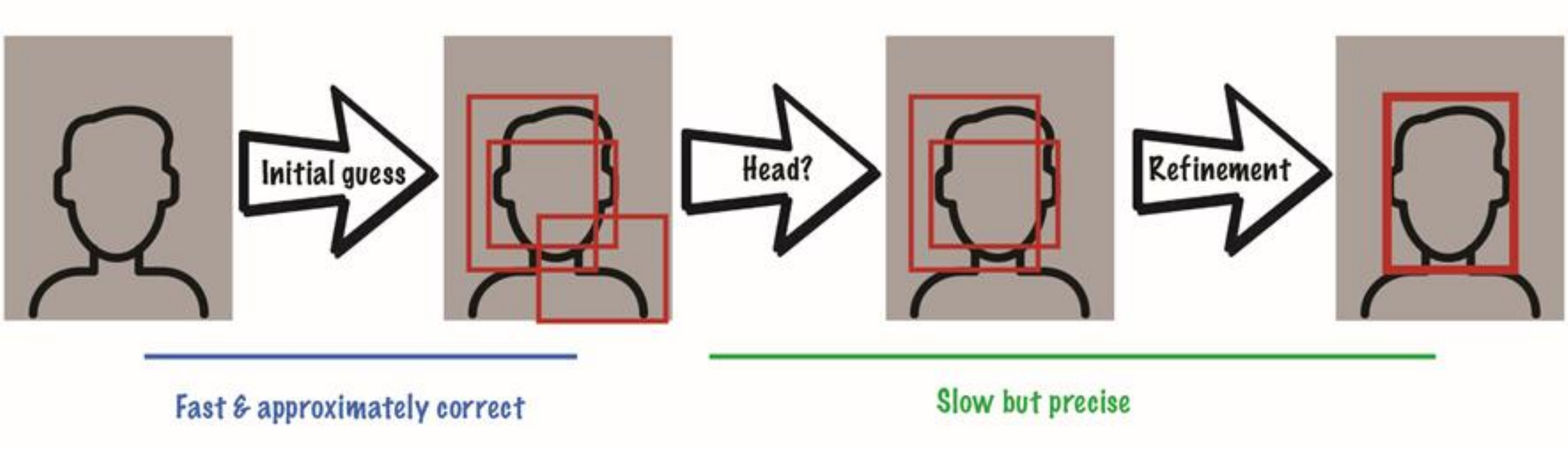
Deep Learning for Object Detection

Lluis Gomez i Bigorda

Computer Vision Center, Universitat Autònoma de Barcelona.

Object detection pipeline (recap of previous class)

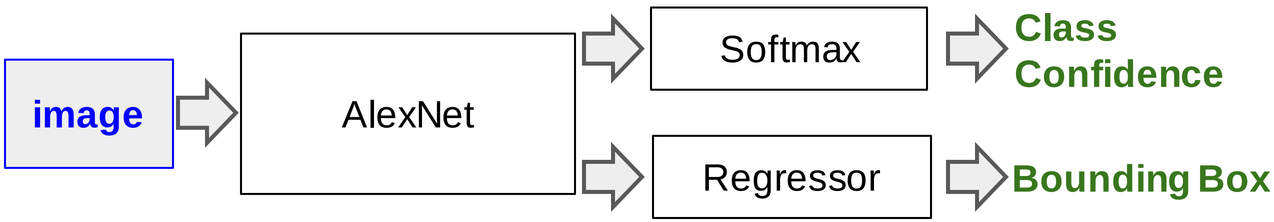
Given the unbalanced nature of detection. What do we need? **Two Stage Framework**



1st Stage **2nd Stage**

Object Detection Models (recap of previous class)

OverFeat

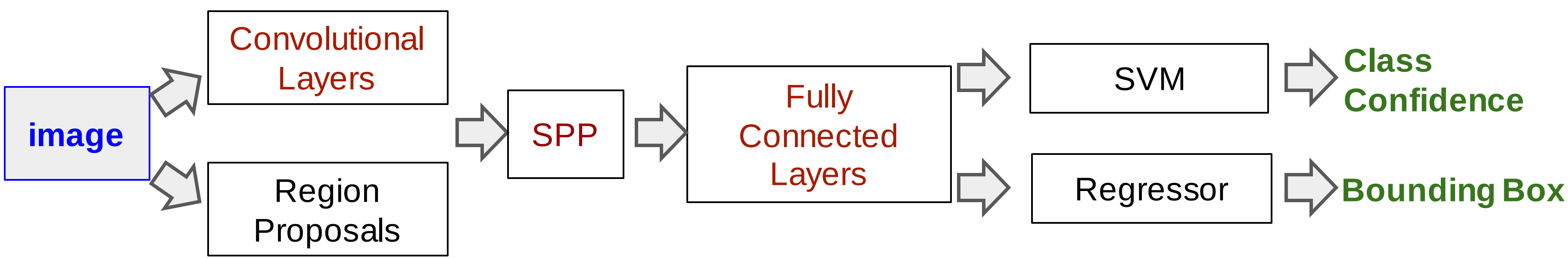


R-CNN

Object Detection Models (recap of previous class)

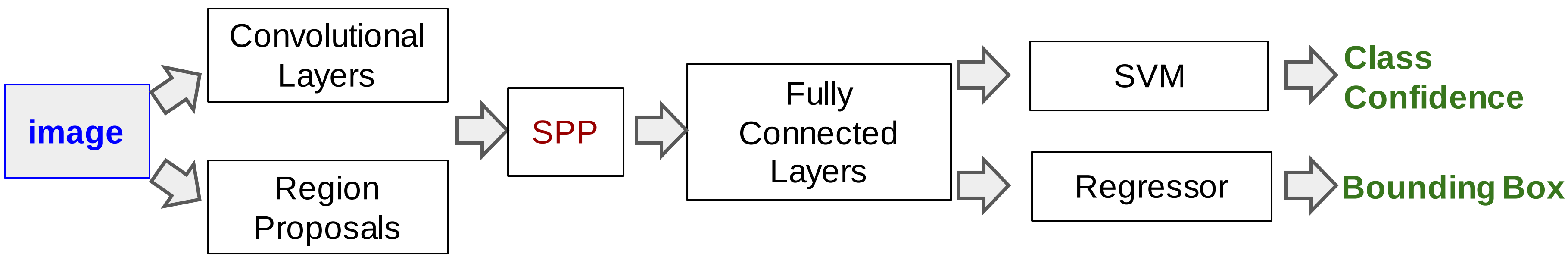
R-CNN

SPP

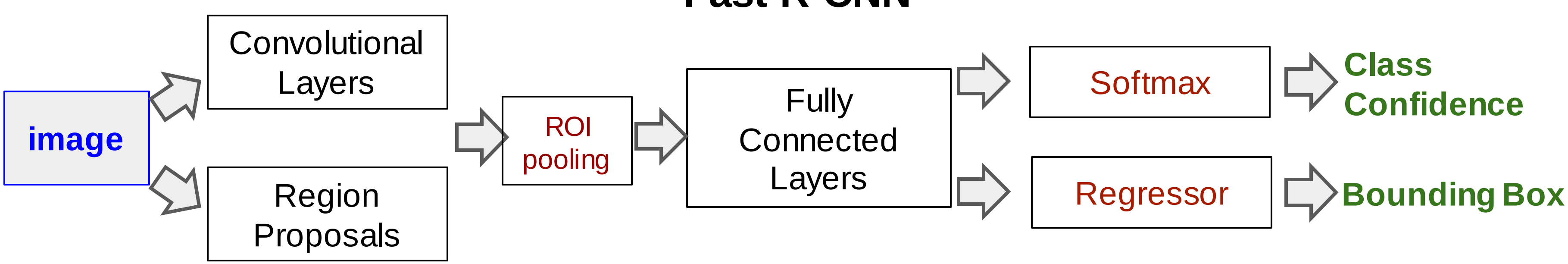


Object Detection Models (recap of previous class)

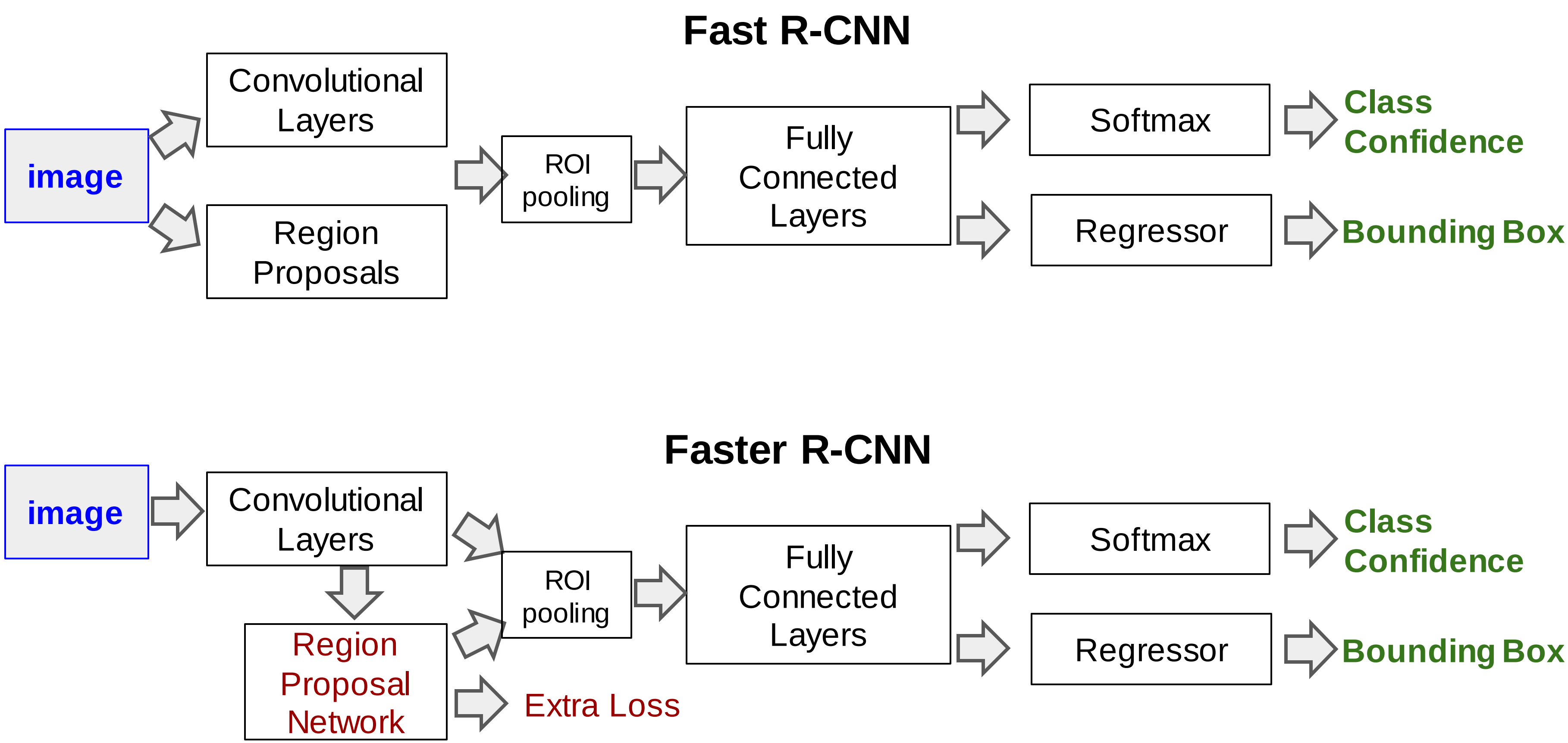
SPP



Fast R-CNN



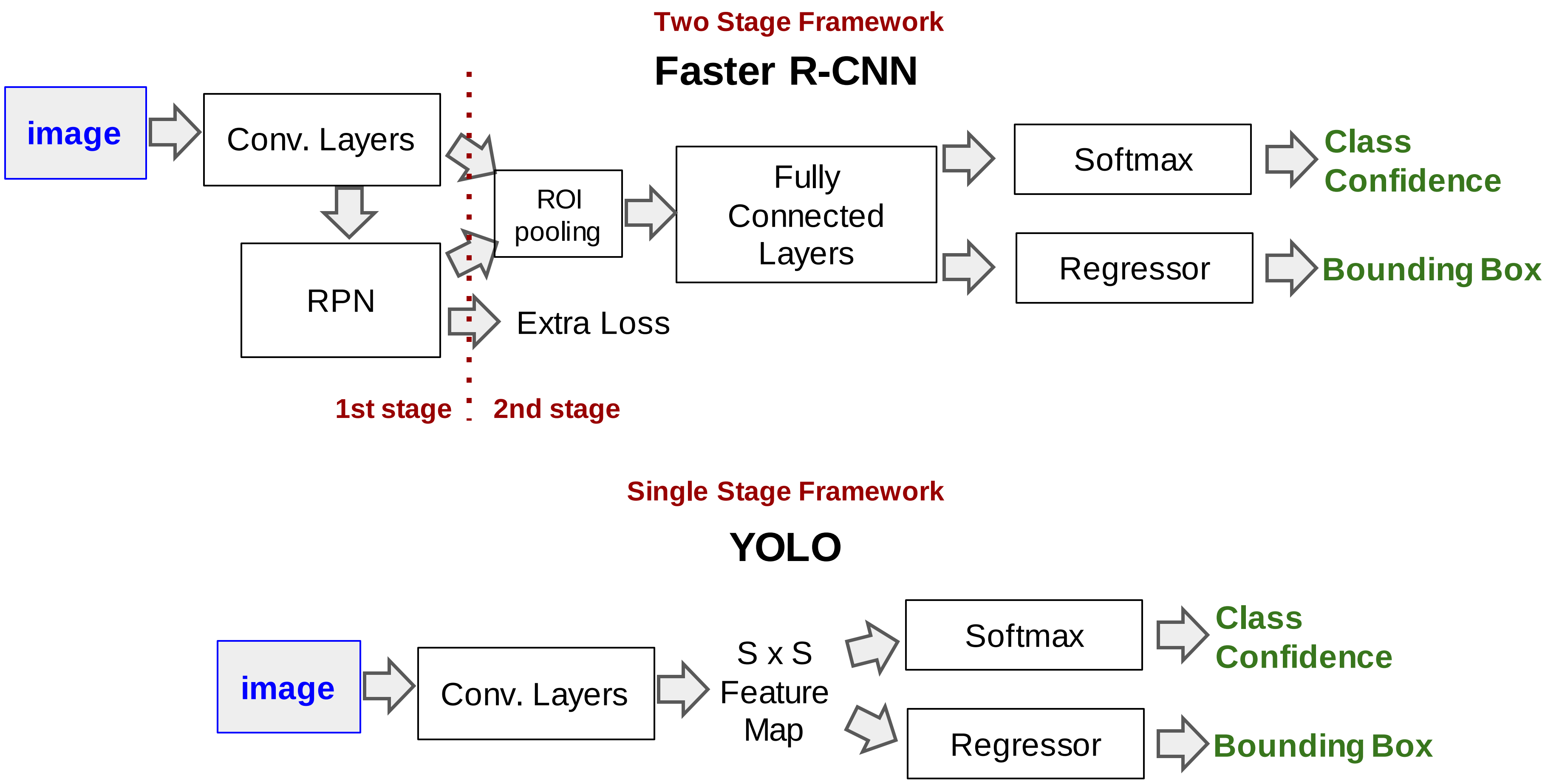
Object Detection Models (recap of previous class)



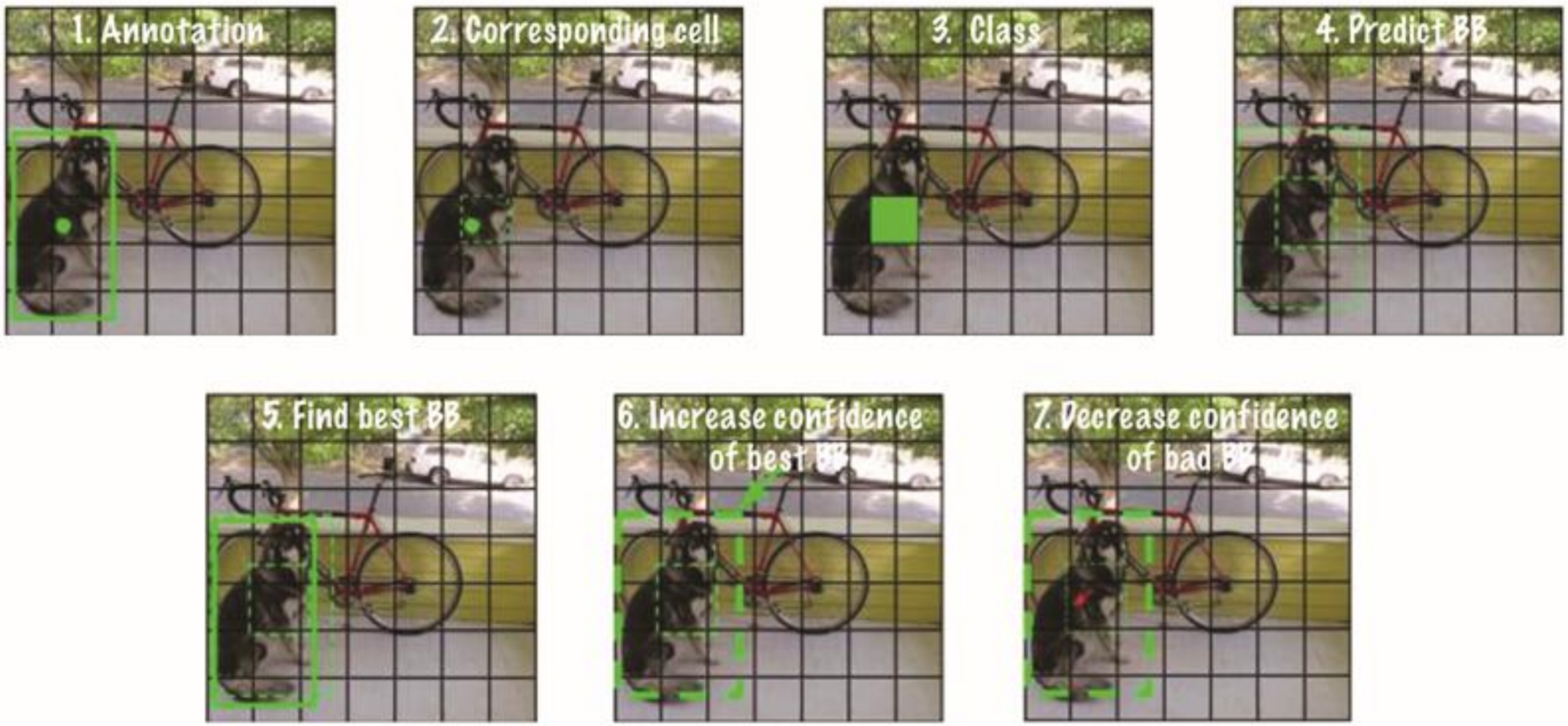
Deep learning for object detection: Outline

- Introduction
- Basic blocks and concepts
- Models (i)
- Models (ii)
    - Single Stage Object Detectors
    - Feature Pyramid Networks
    - Focal Loss
    - Mask R-CNN
    - DETR
    - Other ideas

You Only Look Once - YOLO (2015, CVPR2016)

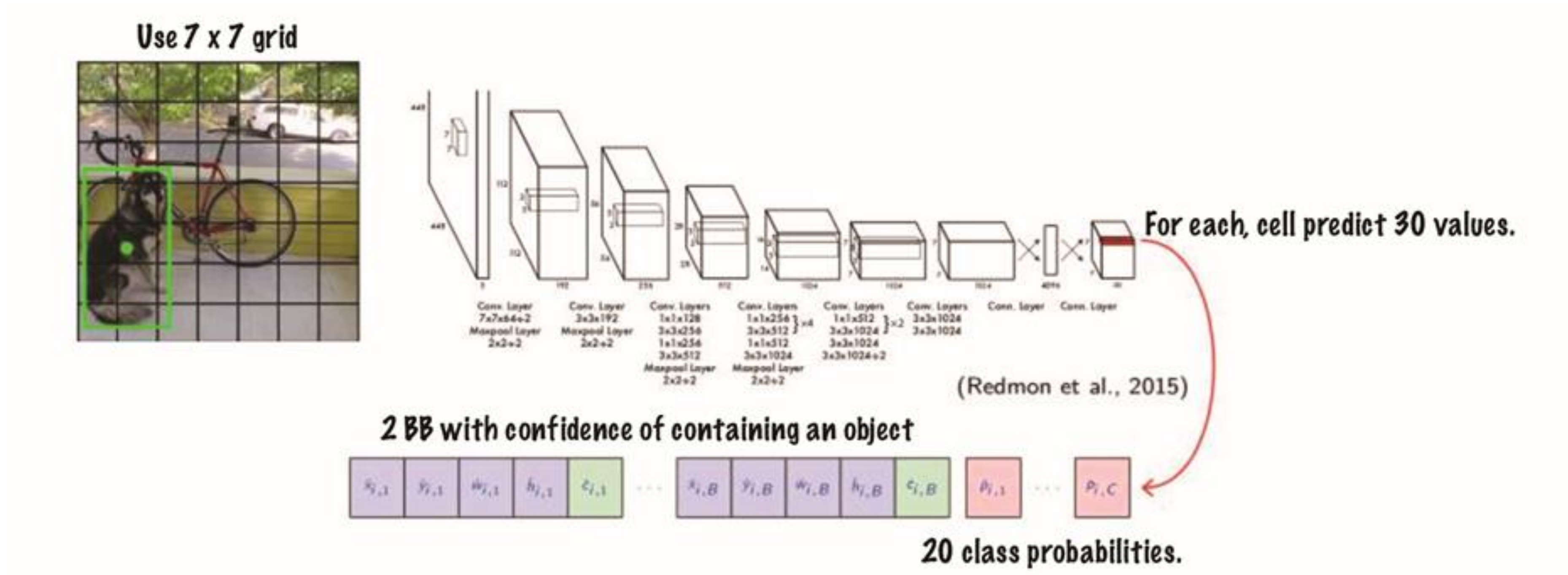


YOLO: Key idea



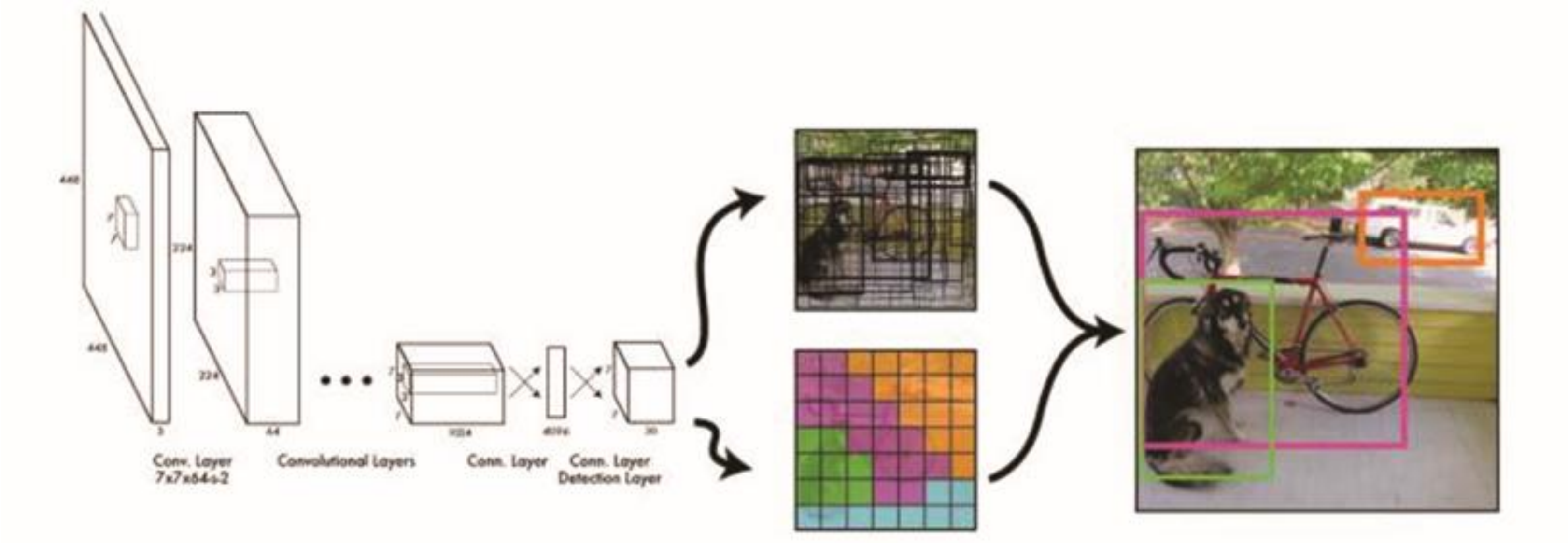
https://docs.google.com/presentation/d/1kAa7NOamBt4calBU9iHgT8a86RRHz9Yz2oh4-GTdX6M/edit#slide=id.g151008b386_0_0

YOLO: Architecture



Francois Fleuret : Deep Learning Course, https://fleuret.org/dlc/

YOLO: Training



1. Pre-train network on Imagenet classification task
2. Train the model with joint loss (quite engineered loss function)

https://docs.google.com/presentation/d/1kAa7NOamBt4calBU9iHgT8a86RRHz9Yz2oh4-GTdX6M/edit#slide=id.g151008b386_0_0

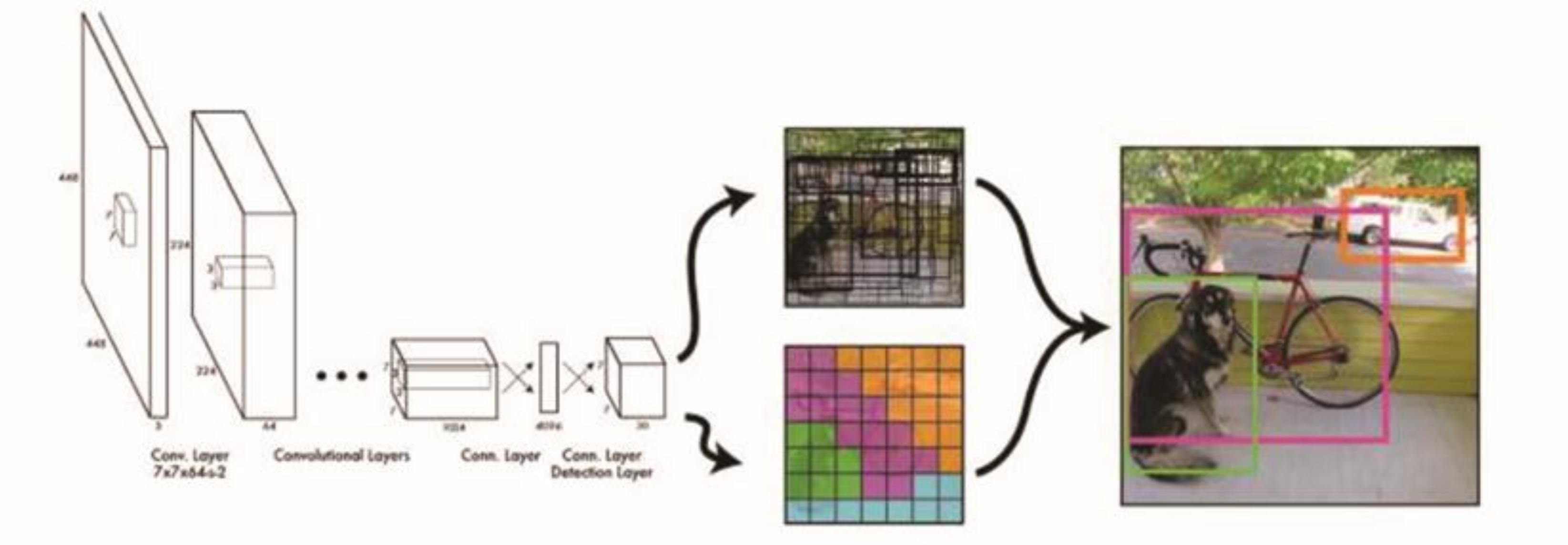
YOLO: Training tricks

1. Use 448×448 input for detection, instead of 224×224,
2. Use Leaky ReLU for all layers,
3. Dropout after the first fully connected layer,
4. Normalize bounding boxes parameters in [0, 1],
5. Use a quadratic loss not only for the bounding box coordinates, but also for the confidence and the class scores,

1. Reduce the weight of large bounding boxes by using the square roots of the size in the loss,
2. Reduce the importance of empty cells by weighting less the confidence-related loss on them,

1. Use momentum 0.9, decay $5e-4$,
2. Data augmentation with scaling, translation, and HSV transformation.

YOLO: Inference



Single pass through the network.

Inference is very fast.

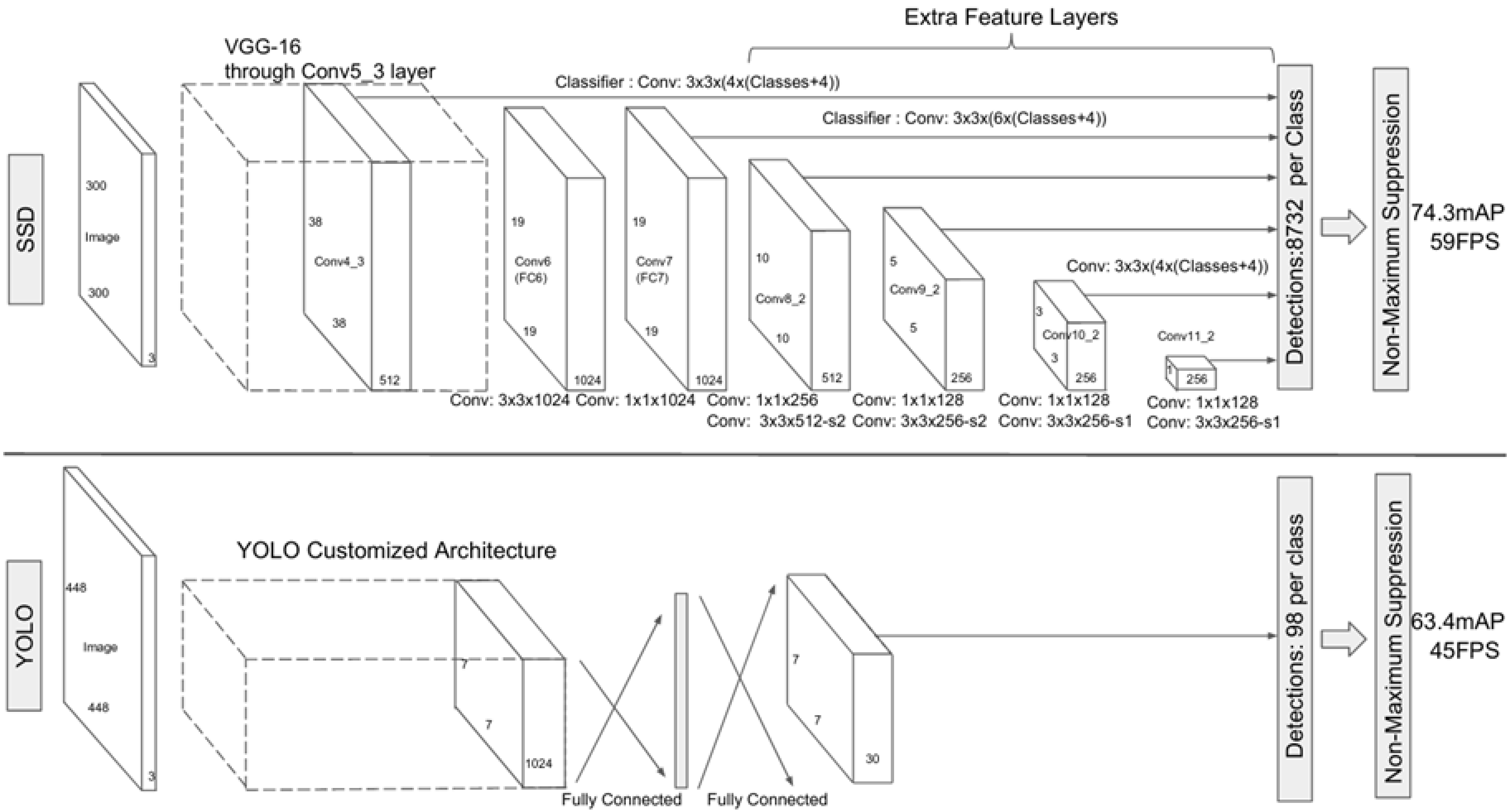
https://docs.google.com/presentation/d/1kAa7NOamBt4calBU9iHgT8a86RRHz9Yz2oh4-GTdX6M/edit#slide=id.g151008b386_0_0

YOLO: Results

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	**73.2**	7 FPS	140 ms/img
YOLO	63.4	**45 FPS**	22 ms/img

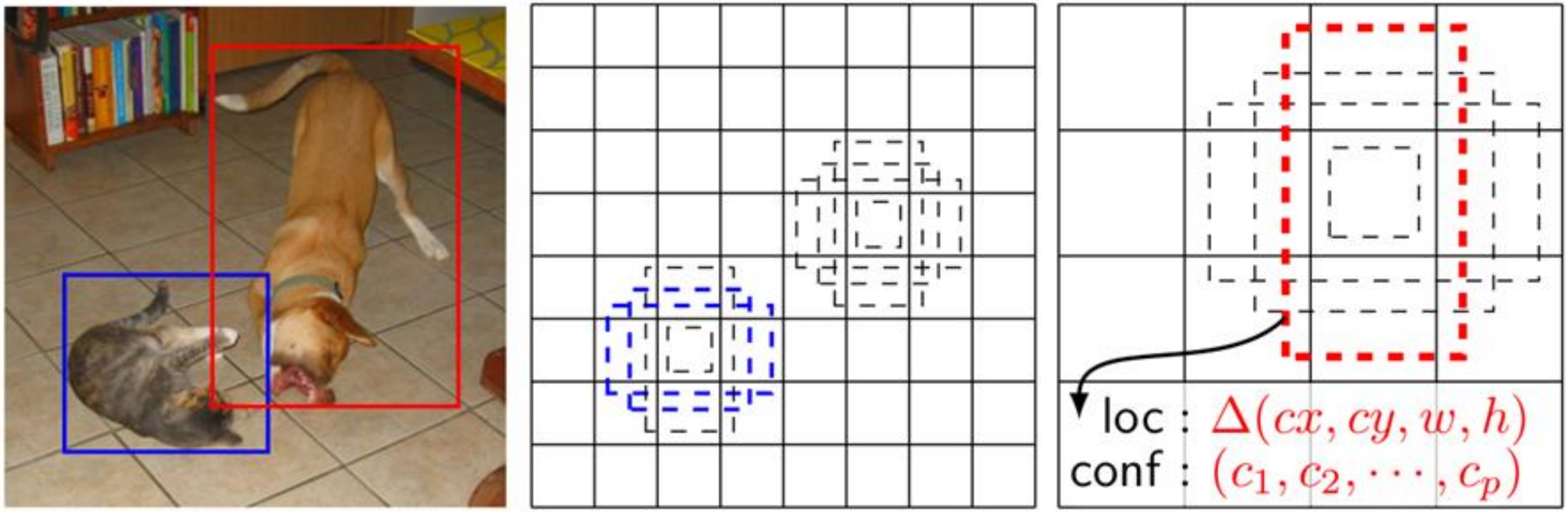
https://docs.google.com/presentation/d/14qBAiyhMOFI_wZW4dA1CkixgXwf0zKGbpw_0oHK8yEM/edit?usp=sharing

Single Shot Detector / SSD (ECCV 2016)



LIU, Wei, et al. [SSD: Single shot multibox detector.] ECCV 2016.

Single Shot Detector / SSD (ECCV 2016)



(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

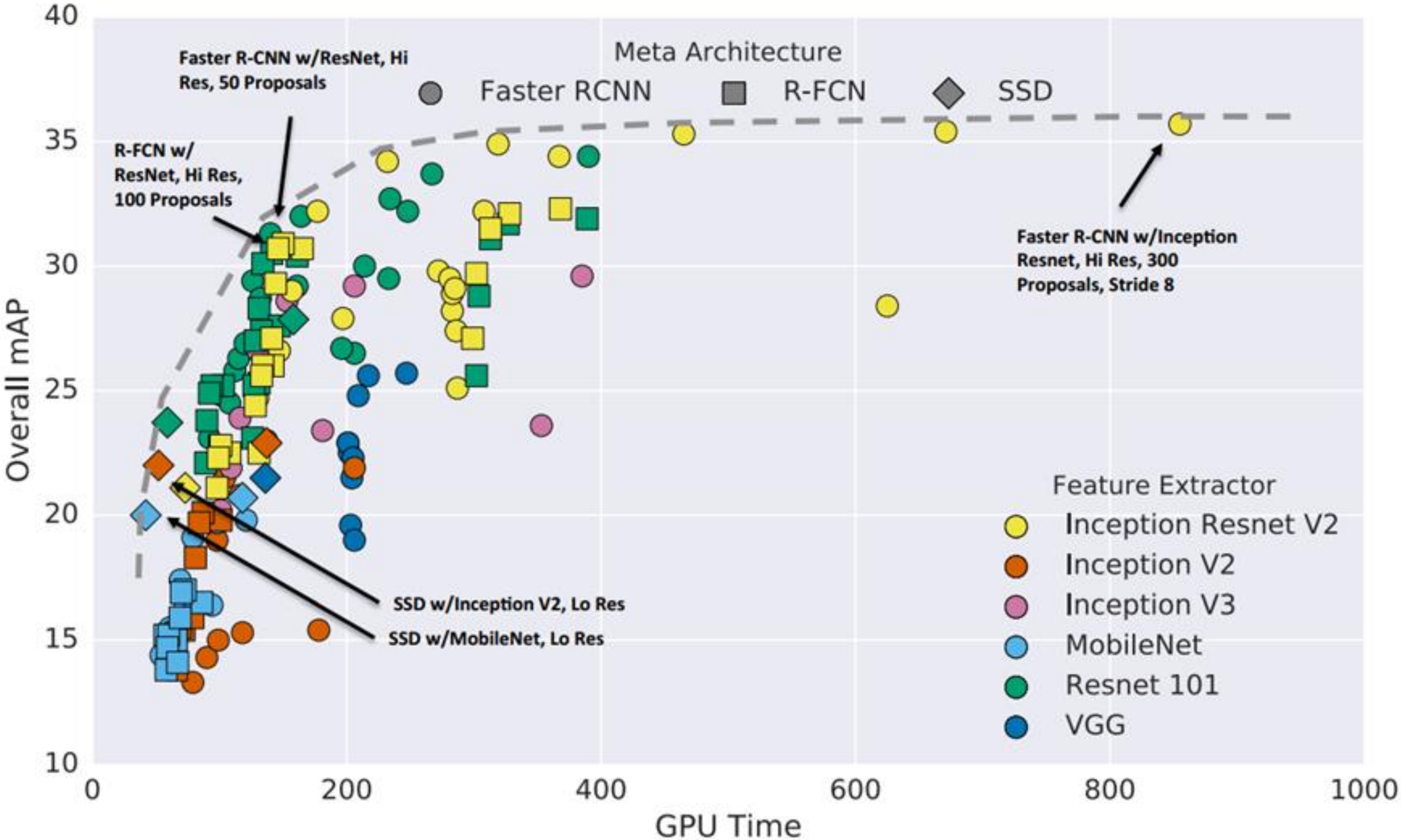
LIU, Wei, et al. SSD: Single shot multibox detector. ECCV 2016.

YOLOv2 (2016)

	YOLO								YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓	✓
anchor boxes?				✓	✓				
new network?					✓	✓	✓	✓	✓
dimension priors?						✓	✓	✓	✓
location prediction?						✓	✓	✓	✓
passthrough?							✓	✓	✓
multi-scale?								✓	✓
hi-res detector?									✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

There are a lot of tricks to get a good architecture for object detection...

Comparison



Huang et al. [Speed/accuracy trade-offs for modern convolutional object detectors.](#) CVPR 2017

The YOLO family from v1 to v7

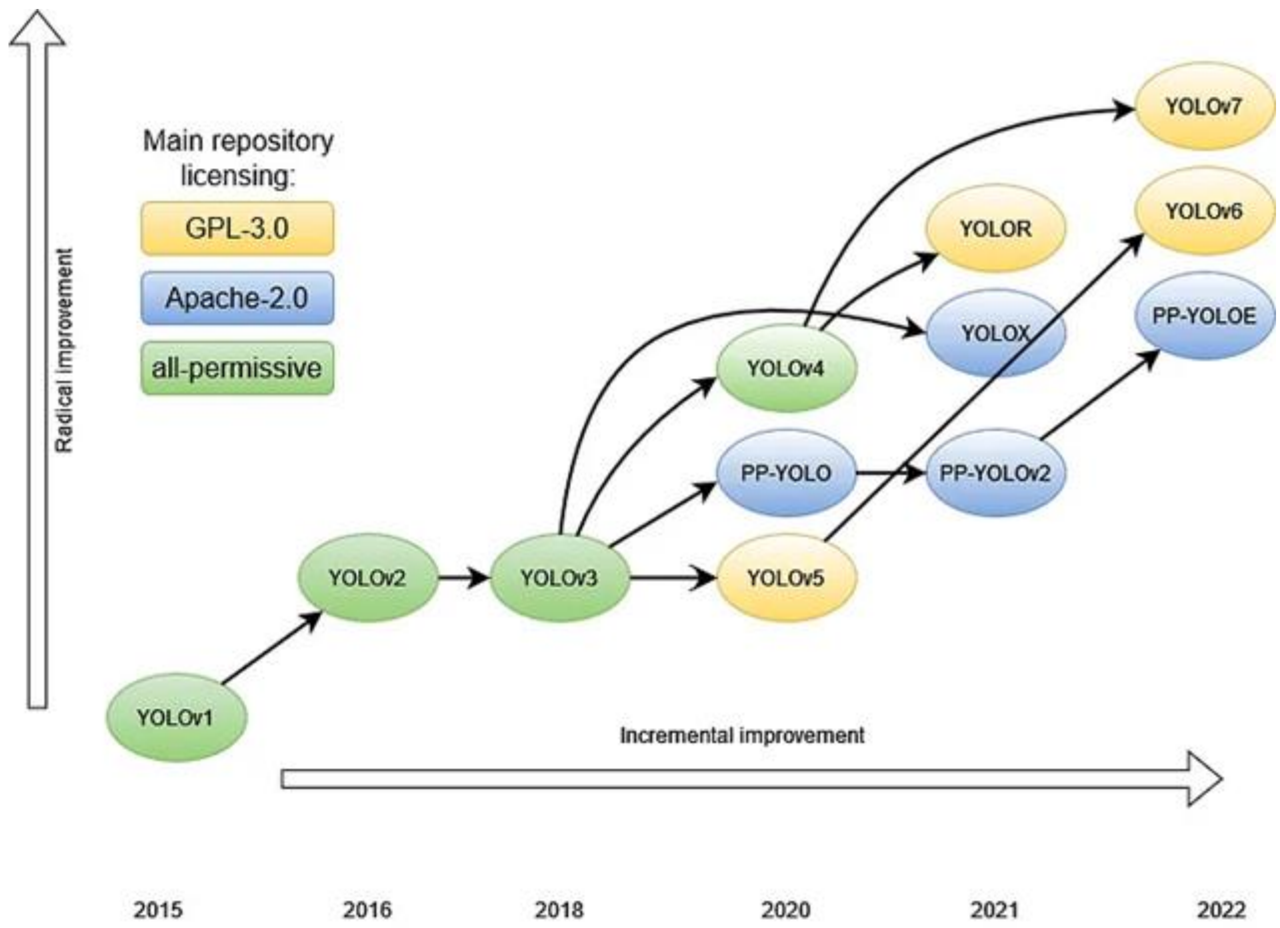


Figure source: [The evolution of the YOLO neural networks family from v1 to v7](#).

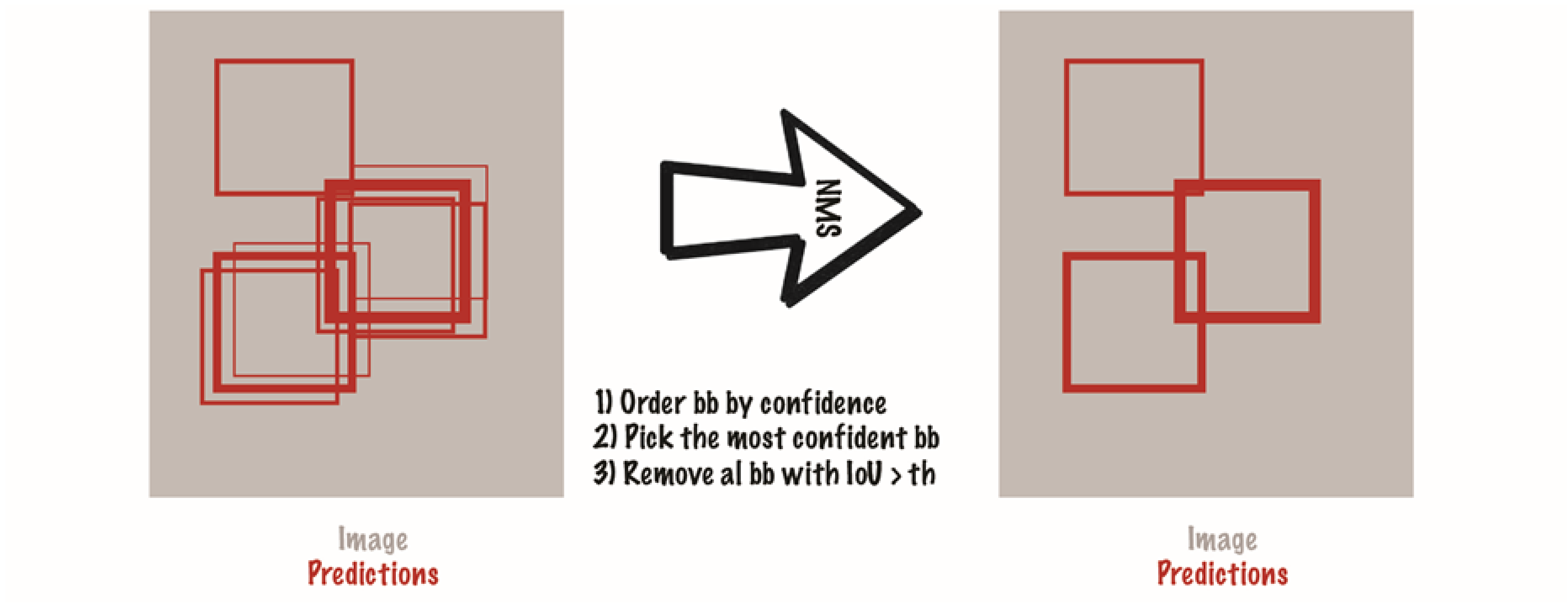
The YOLO family from v1 to v7

YOLO version	backbone	neck	head(s)	augmentations
YOLOv1	GoogLeNet, VGG-16	2x fully connected layers	combined classes + bboxes	random scaling & translations up to 20%; random adjust exposure & saturation up to x1.5 in HSV
YOLOv2	Darknet-19	fully convolutional layers	combined hierarchical classes + bboxes, anchor-based	random crops, rotations, and hue, saturation, and exposure shifts
YOLOv3	Darknet-53	FPN	combined multilabel + bboxes, anchor-based	no specific info, seems like the same as in YOLOv2
YOLOv5	CSPDarknet53	SPPF, CSP-PAN	combined multilabel + bboxes, anchor-based	Mosaic, copy-paste, random affine, MixUp, random adjust HSV, random horizontal flip
PP-YOLO	ResNet50-vd + deformable convolutions	FPN, SPP	combined multilabel + bboxes, anchor-based	MixUp
YOLOv4	CSPDarknet53	PANet, SPP	combined multilabel + bboxes, anchor-based	CutMix, Mosaic, MixUp, CutOut, Self-Adversarial Training, bilateral blurring
PP-YOLOv2	ResNet50-vd + deformable convolutions	PANet	combined multilabel + bboxes, anchor-based	MixUp; random color distortion, expand, crop, flip
YOLOX	Darknet-53	FPN	decoupled multilabel + bboxes, anchor-free	Mosaic, MixUp, random horizontal flip, colorjitter
YOLOR	sequence of convolutional layers with downscaling	FPN, CSP, SPP	multi-head (object detection, multi-label classification, feature embedding)	CutMix, Mosaic, MixUp, CutOut, Self-Adversarial Training, bilateral blurring
PP-YOLOE	CSPRepResNet	PANet	Efficient Task-aligned Head (decoupled), anchor-free	random crop, horizontal flip, color distortion, multi-scale
YOLOv6	EfficientRep	Rep-PAN	Efficient decoupled head, anchor-free	Mosaic, MixUp
YOLOv7	Extended-ELAN	-	multiple (lead heads & aux heads), anchor-based	random perspective, HSV jitter, flips, Mosaic

Table source: The evolution of the YOLO neural networks family from v1 to v7.

Non-Maximum Suppression (NMS)

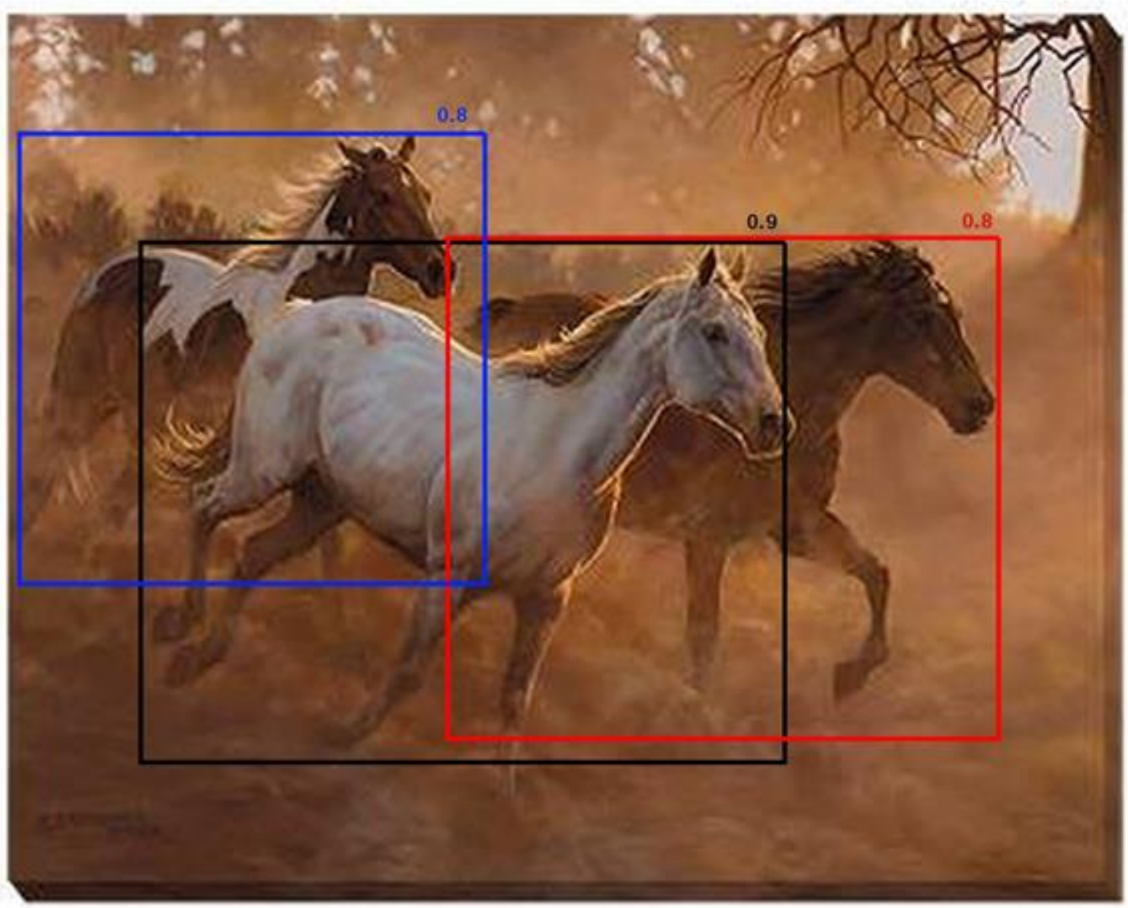
(remember) Common component to all Object Detection architectures!



Non-Maximum Suppression (NMS)

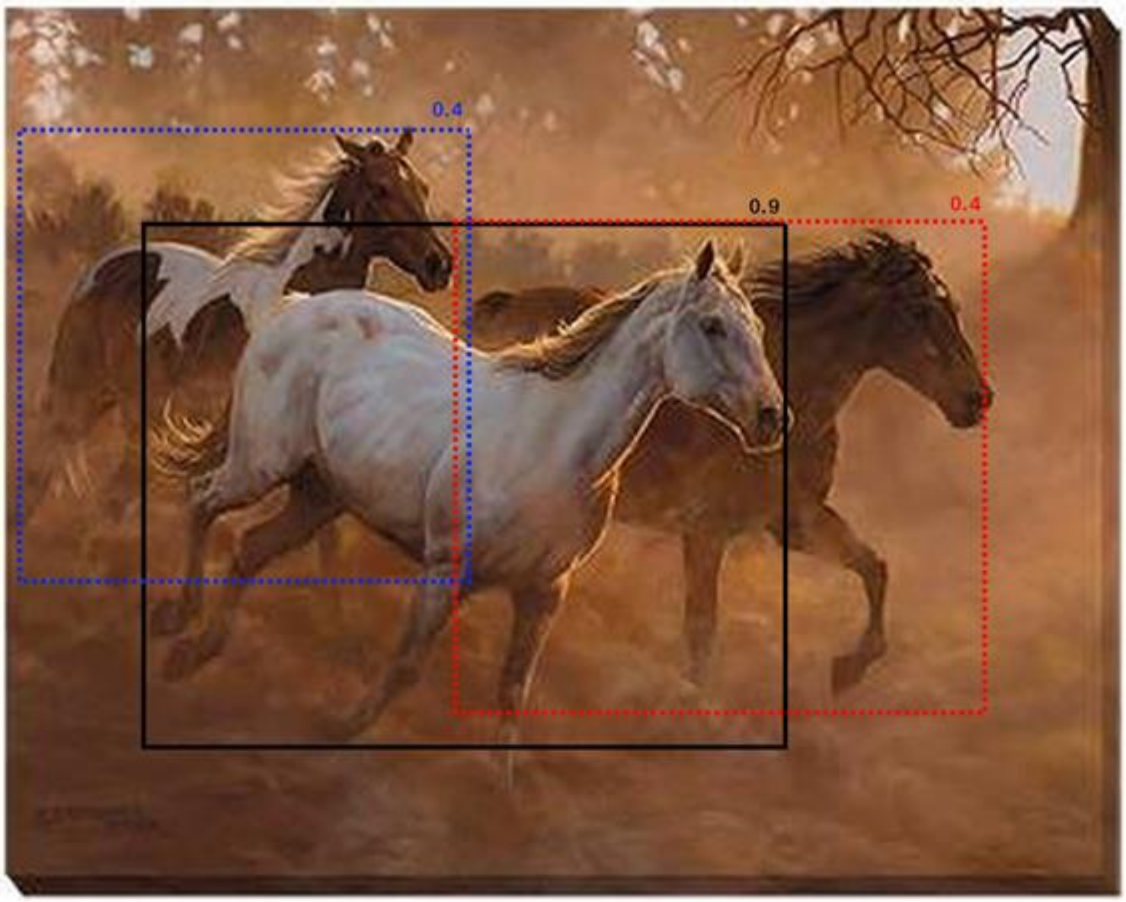
https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c

Non-Maximum Suppression (NMS)



https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c

Non-Maximum Suppression (NMS)



Bodla et al. [Soft-NMS -- Improving Object Detection With One Line of Code]() ICCV 2017

Non-Maximum Suppression (NMS)

Input : $\mathcal{B} = \{b_1, .., b_N\}$, $\mathcal{S} = \{s_1, .., s_N\}$, N_t
$\mathcal{B}$ is the list of initial detection boxes
$\mathcal{S}$ contains corresponding detection scores
N_t is the NMS threshold

begin
    $\mathcal{D} \leftarrow \{\}$
    while $\mathcal{B} \neq empty$ **do**
        $m \leftarrow \text{argmax } \mathcal{S}$
        $\mathcal{M} \leftarrow b_m$
        $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{M}; \mathcal{B} \leftarrow \mathcal{B} - \mathcal{M}$
        for b_i *in* $\mathcal{B}$ **do**
            if $iou(\mathcal{M}, b_i) \geq N_t$ **then**
                $\mathcal{B} \leftarrow \mathcal{B} - b_i; \mathcal{S} \leftarrow \mathcal{S} - s_i$
            end
            NMS

            $s_i \leftarrow s_i f(iou(\mathcal{M}, b_i))$
            Soft-NMS
        end
    end
    return $\mathcal{D}, \mathcal{S}$
end

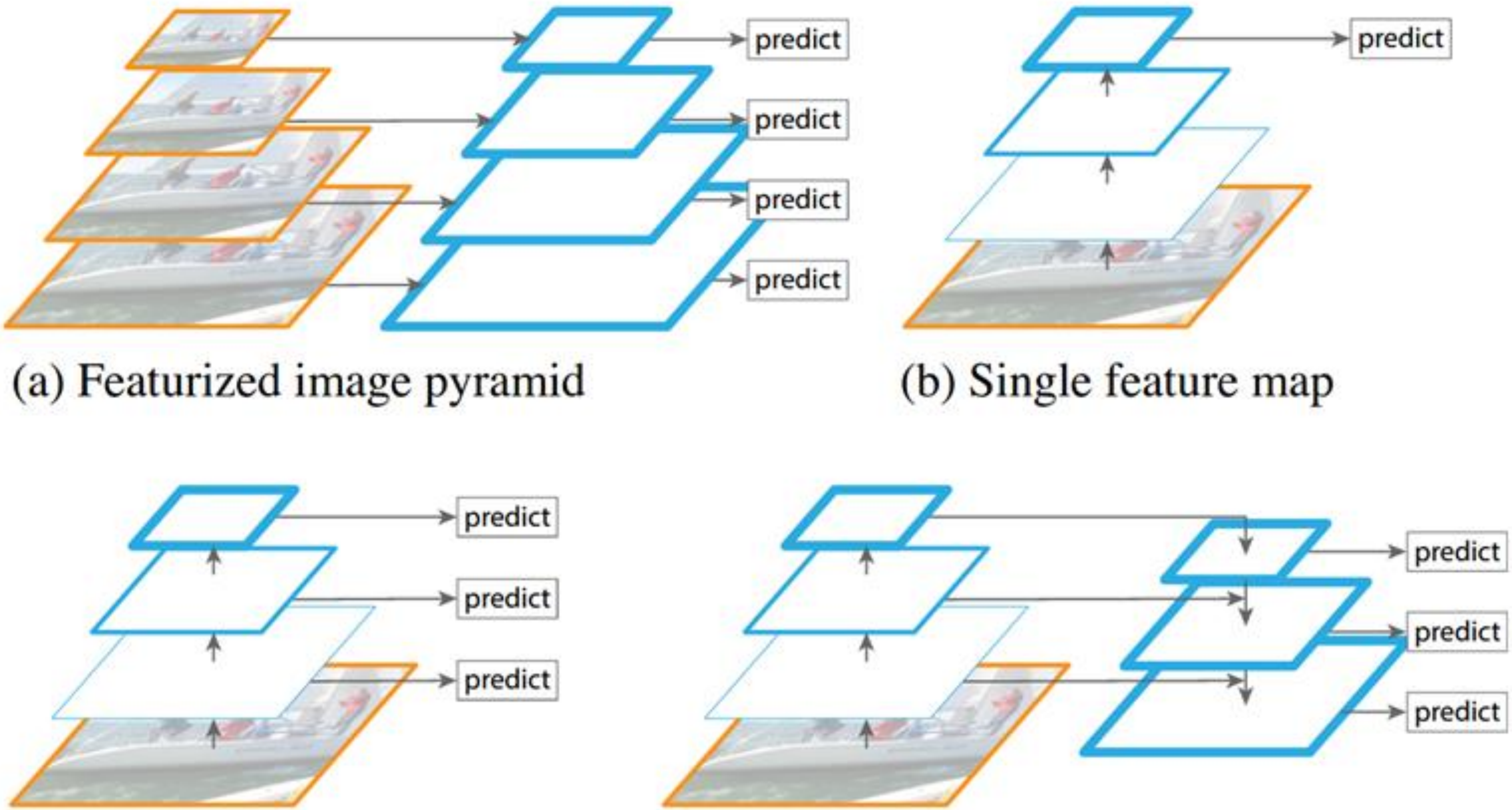
Bodla et al. Soft-NMS -- Improving Object Detection With One Line of Code ICCV 2017

Non-Maximum Suppression (NMS)

NMS : https://github.com/rbgirshick/fast-rcnn/blob/master/lib/utils/nms.py

Soft-NMS : https://github.com/DocF/Soft-NMS/blob/master/soft_nms.py

Feature Pyramid Networks (CVPR 2017)

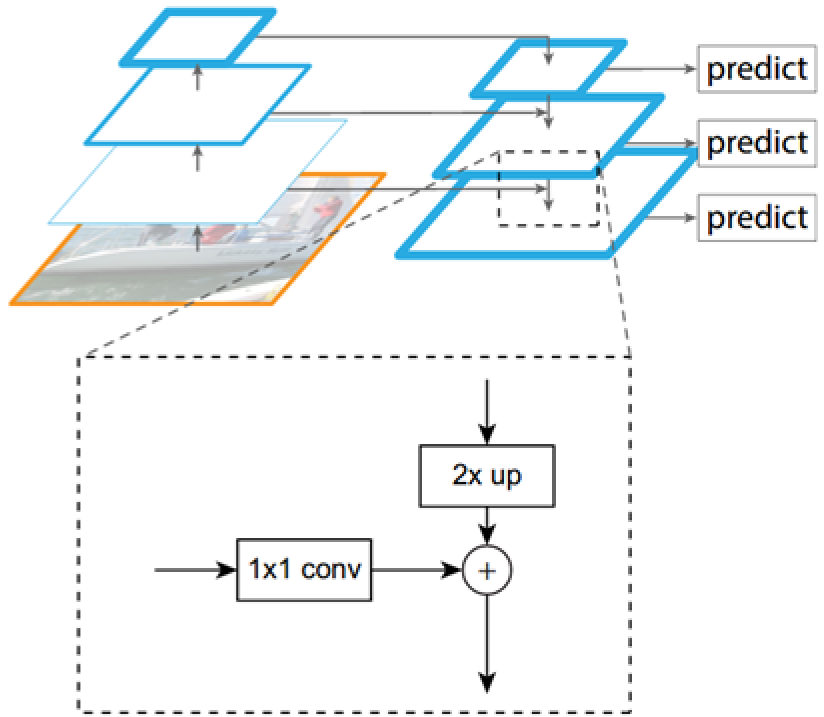


(a) Featurized image pyramid (b) Single feature map

(c) Pyramidal feature hierarchy (d) Feature Pyramid Network

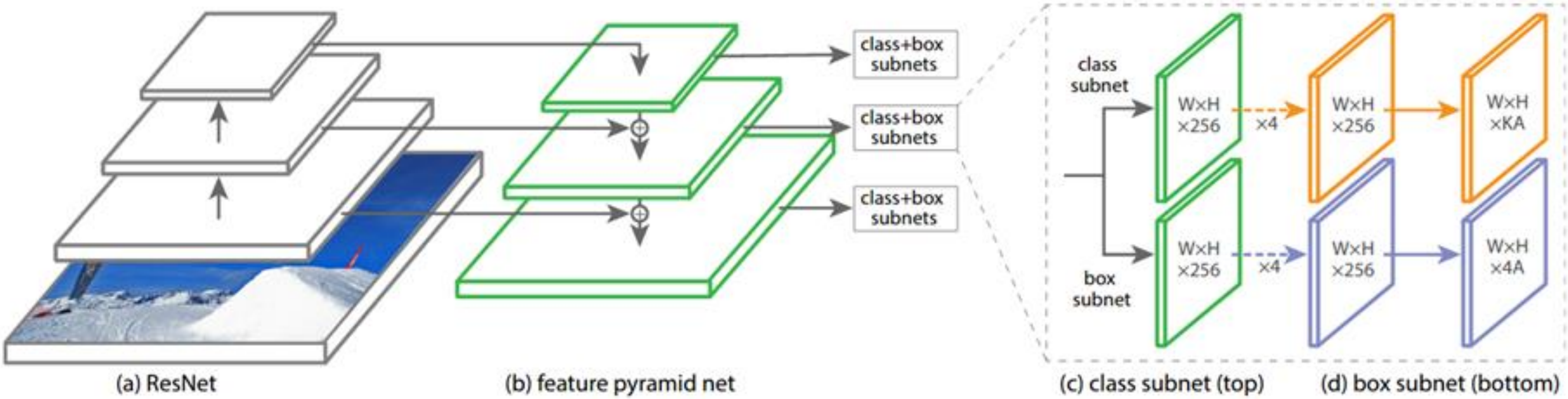
Lin at al. Feature pyramid networks for object detection. CVPR 2017.

Feature Pyramid Networks



Faster R-CNN	proposals	feature	head	lateral?	top-down?	AP@0.5	AP	AP_s	AP_m	AP_l
(*) baseline from He *et al.* [16]†	RPN, C_4	C_4	conv5			47.3	26.3	-	-	-
(a) baseline on conv4	RPN, C_4	C_4	conv5			53.1	31.6	13.2	35.6	**47.1**
(b) baseline on conv5	RPN, C_5	C_5	2*fc*			51.7	28.0	9.6	31.9	43.1
(c) **FPN**	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*	✓	✓	**56.9**	**33.9**	**17.8**	**37.7**	45.8

RetinaNet (ICCV 2017)



4 Conv Layers with 256 3x3 filters

A=9 **anchor boxes**
K=80 object class labels (COCO)

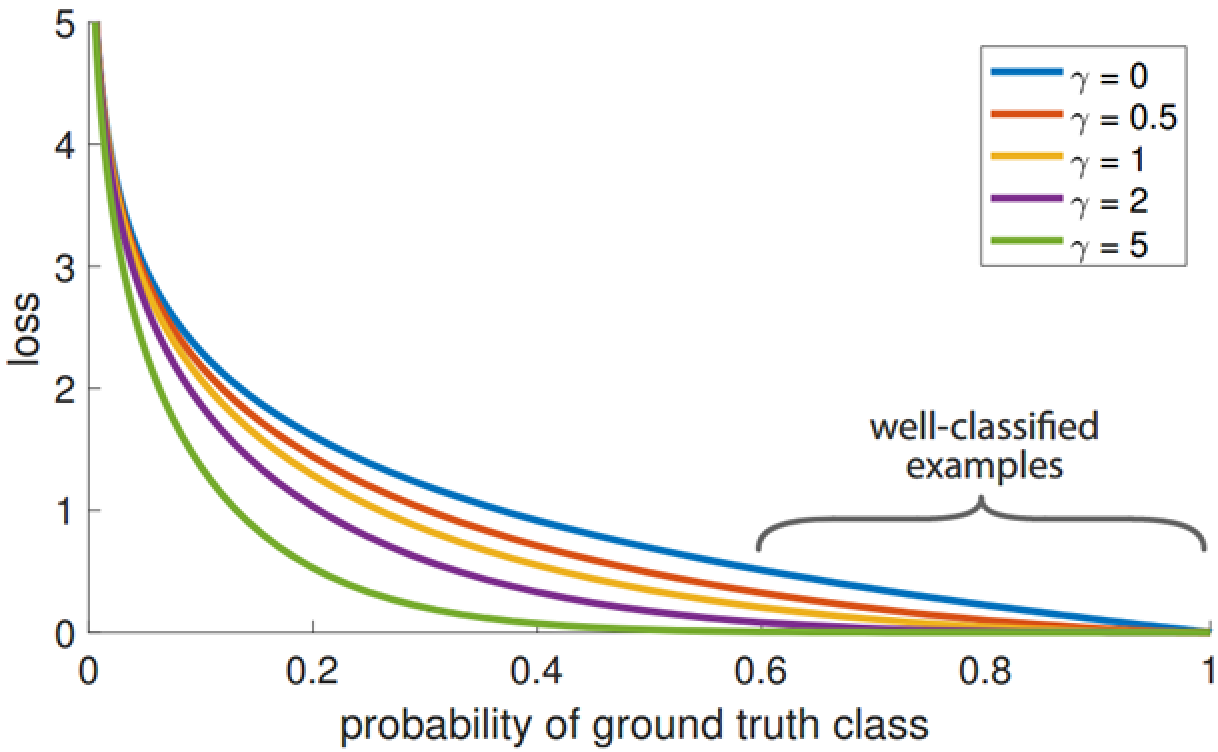
Lin et al. [Focal loss for dense object detection](). ICCV 2017.

Focal Loss

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases}$$

$$\mathrm{CE}(p_t) = -\log(p_t)$$

$$\mathrm{FL}(p_t) = -(1 - p_t)^{\gamma} \log(p_t)$$



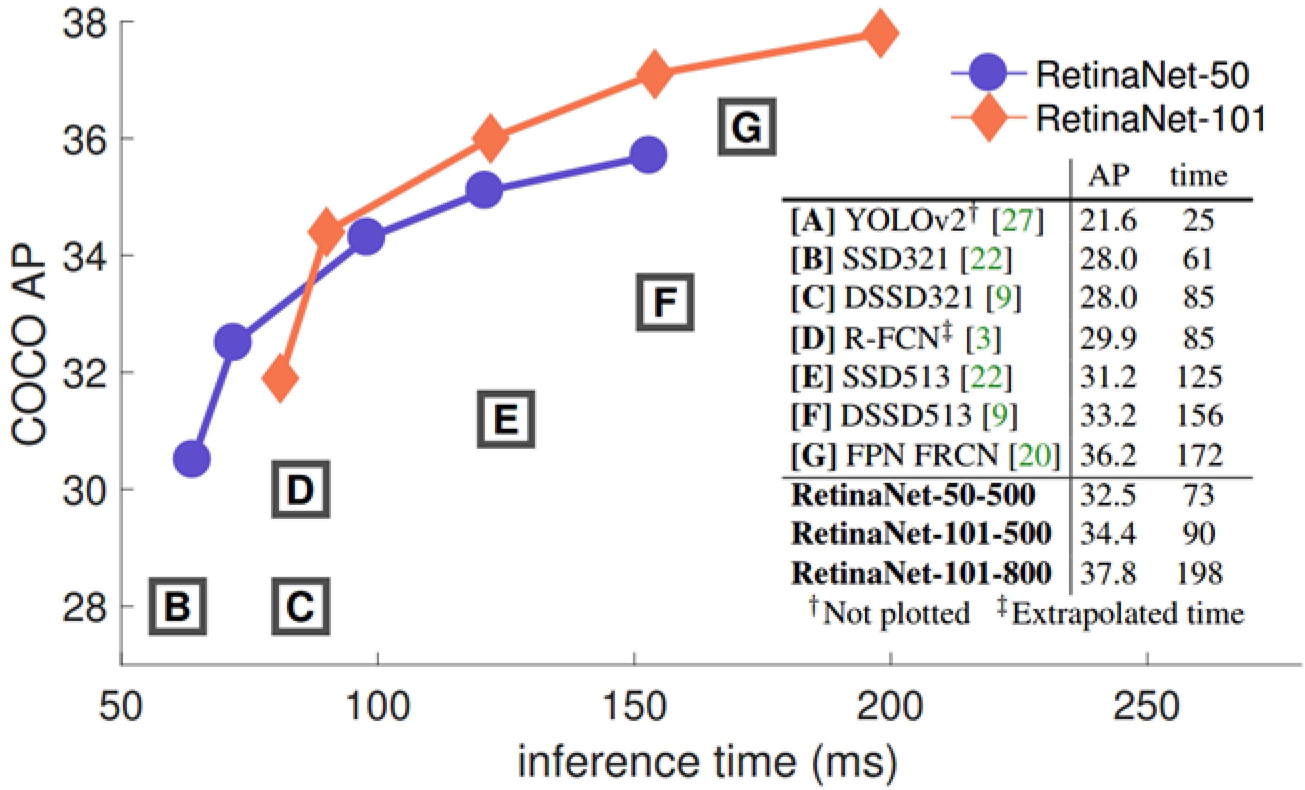
Lin et al. [Focal loss for dense object detection](). ICCV 2017.

The unbalanced nature of detection. Hard Negative Mining vs. Focal Loss

Why putting more focus on hard, misclassified examples?

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

Focal Loss / RetinaNet

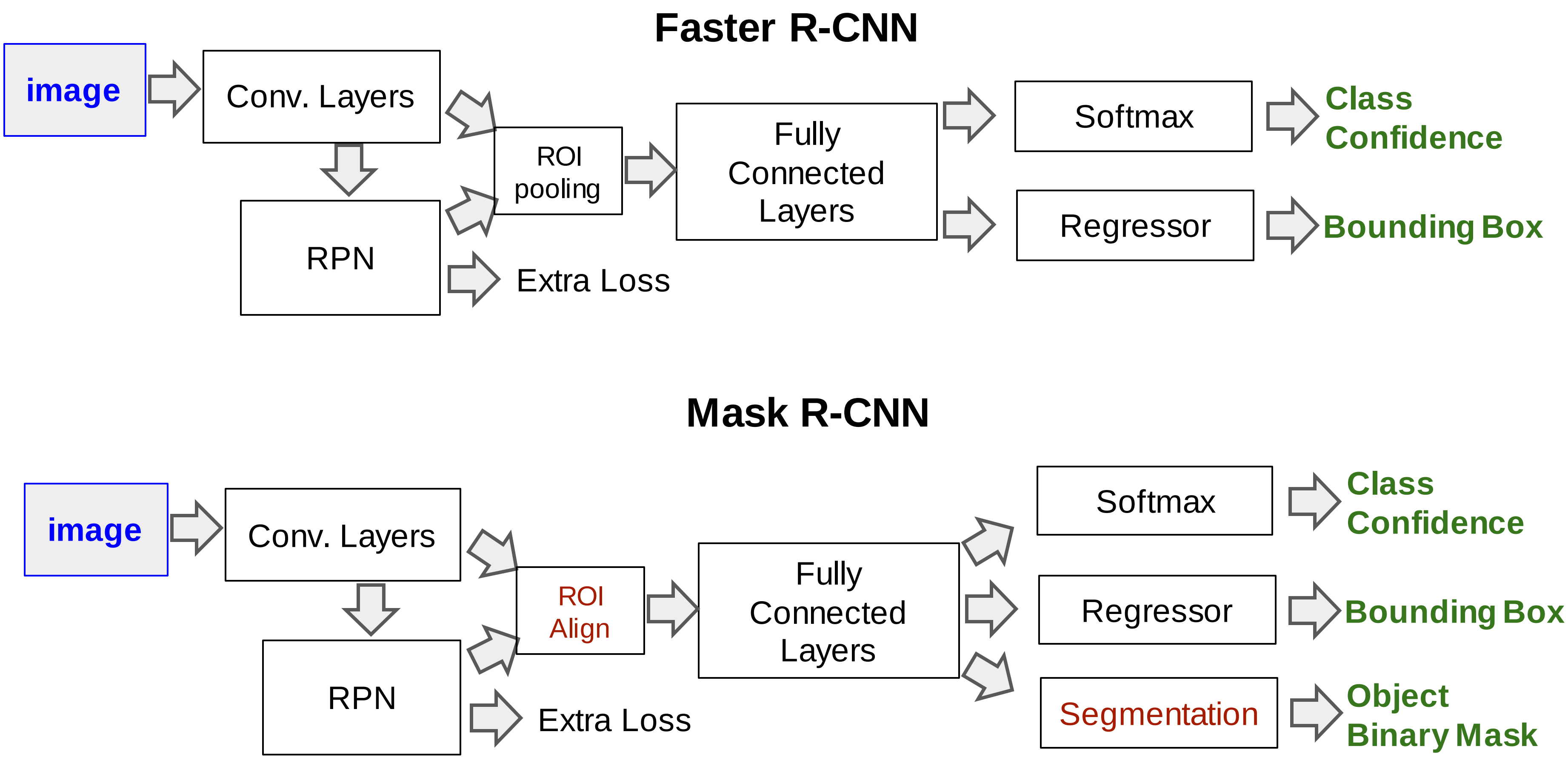


	AP	time
[A] YOLOv2† [27]	21.6	25
[B] SSD321 [22]	28.0	61
[C] DSSD321 [9]	28.0	85
[D] R-FCN‡ [3]	29.9	85
[E] SSD513 [22]	31.2	125
[F] DSSD513 [9]	33.2	156
[G] FPN FRCN [20]	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	37.8	198

†Not plotted ‡Extrapolated time

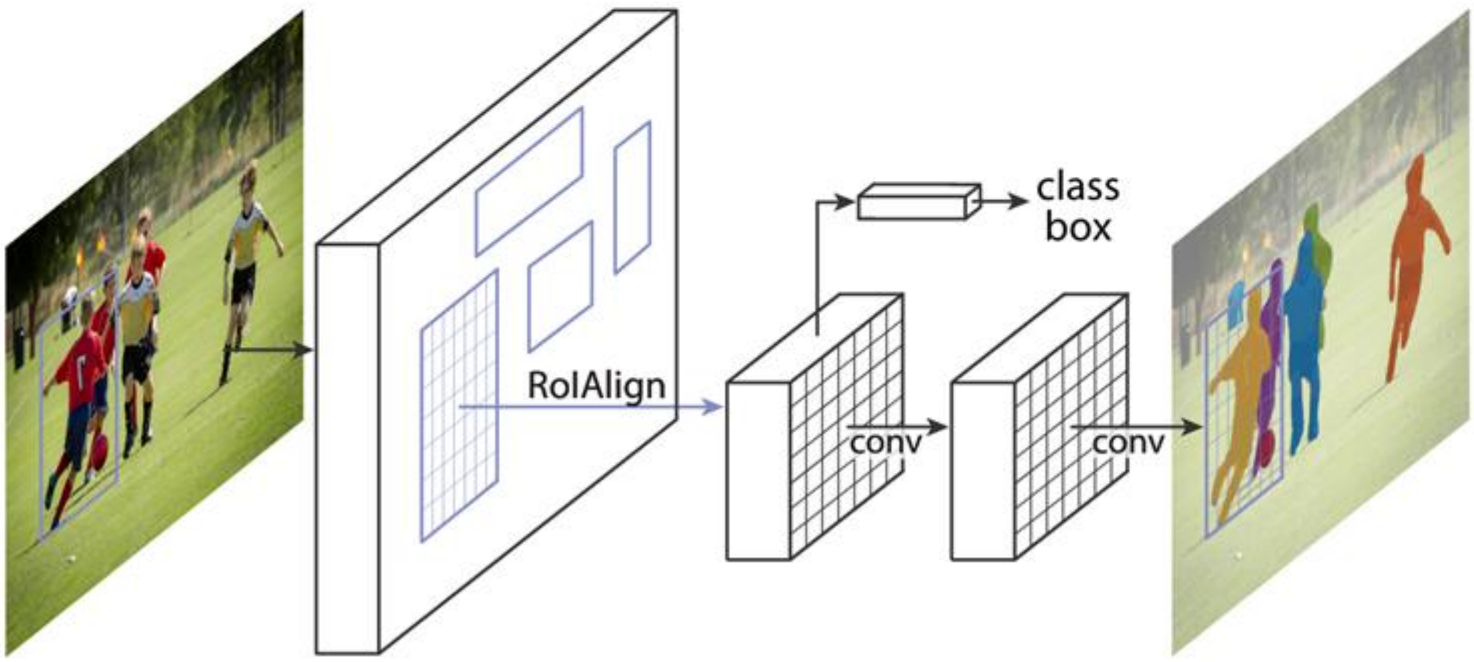
Lin et al. Focal loss for dense object detection. ICCV 2017.

Mask R-CNN (ICCV 2017)



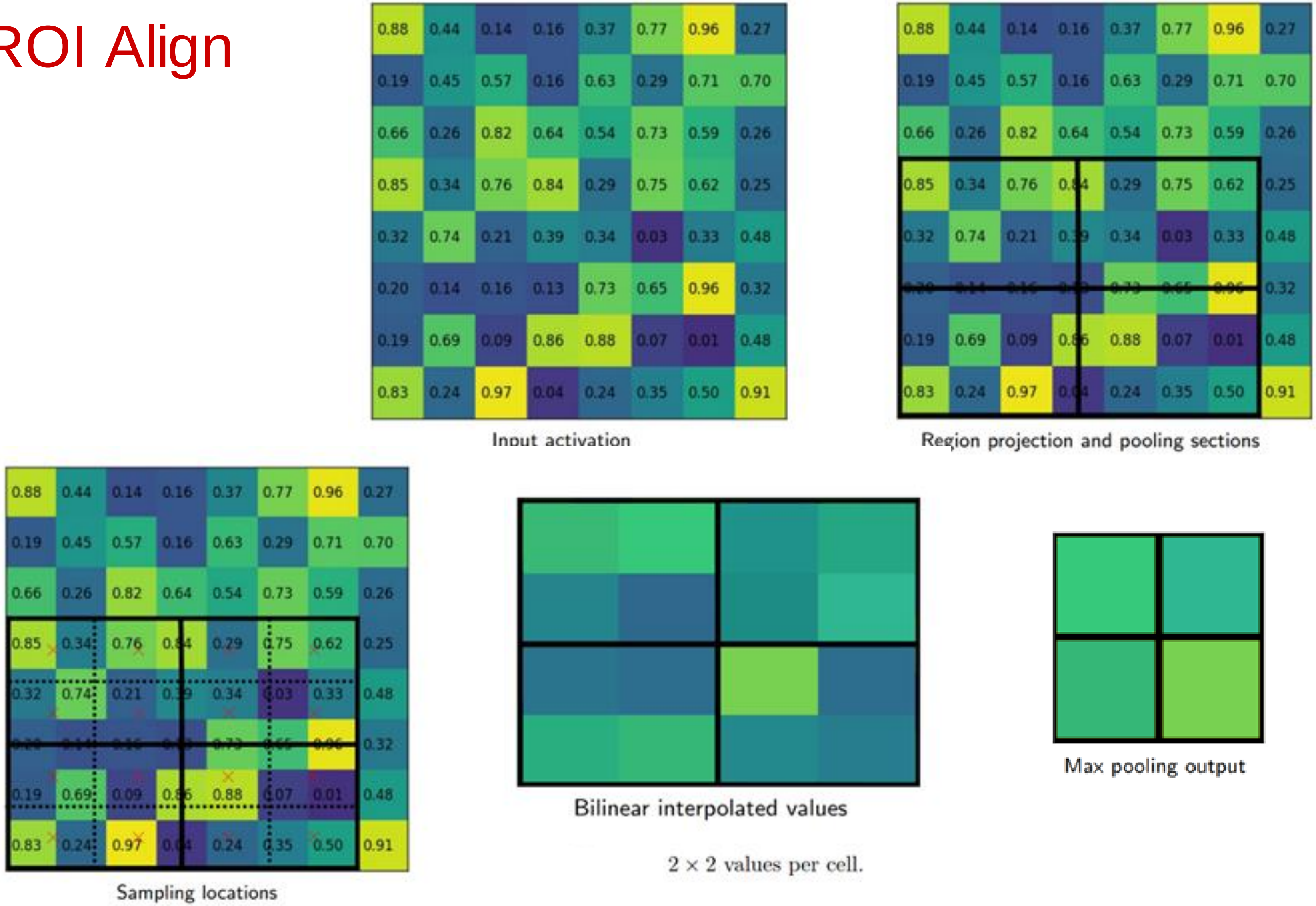
He, K., Gkioxari, G., Dollár, P., & Girshick, R. Mask R-CNN. ICCV 2017.

Mask R-CNN for instance segmentation



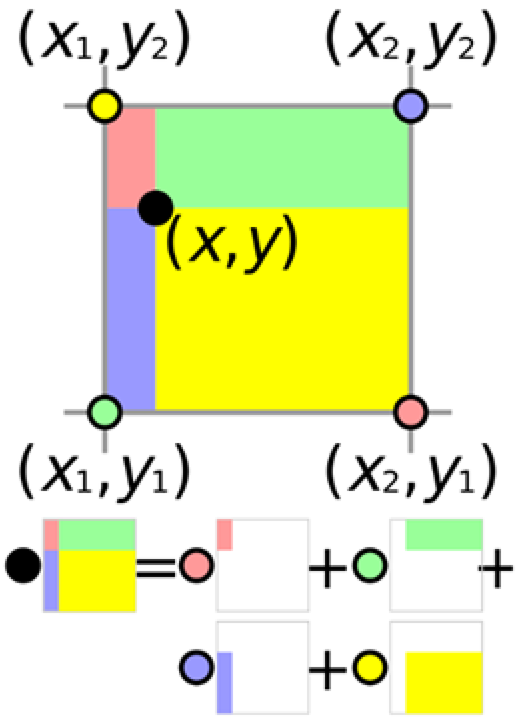
He, K., Gkioxari, G., Dollár, P., & Girshick, R. Mask R-CNN. ICCV 2017.

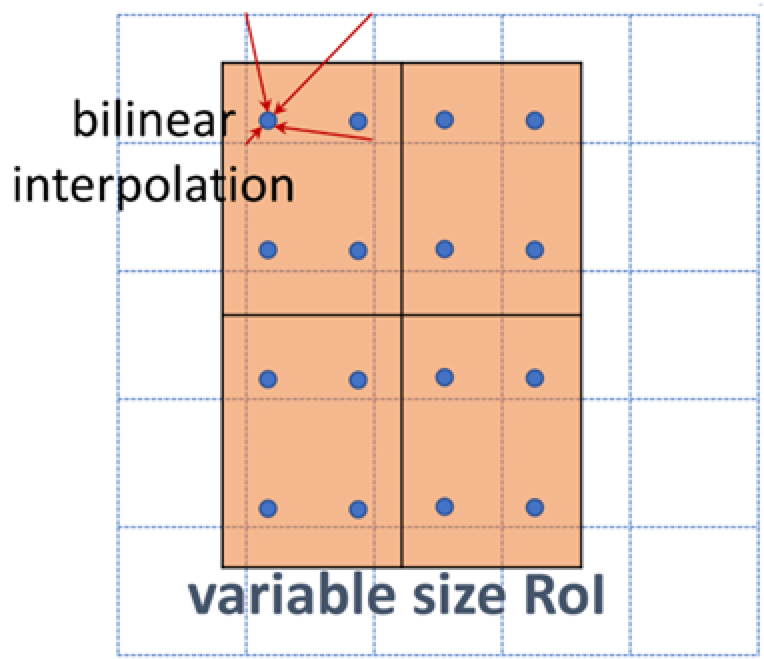
ROI Align



N. Sardana. [Instance Segmentation](https://towardsdatascience.com/understanding-region-of-interest-part-2-roi-align-and-roi-warp-f795196fc193). 2018.
https://towardsdatascience.com/understanding-region-of-interest-part-2-roi-align-and-roi-warp-f795196fc193

ROI Align / Bilinear interpolation





Bilinear interpolation for RoIAlign.

N. Sardana. [Instance Segmentation](https://towardsdatascience.com/understanding-region-of-interest-part-2-roi-align-and-roi-warp-f795196fc193). 2018.
https://towardsdatascience.com/understanding-region-of-interest-part-2-roi-align-and-roi-warp-f795196fc193

Mask R-CNN Bounding Box Detection Results

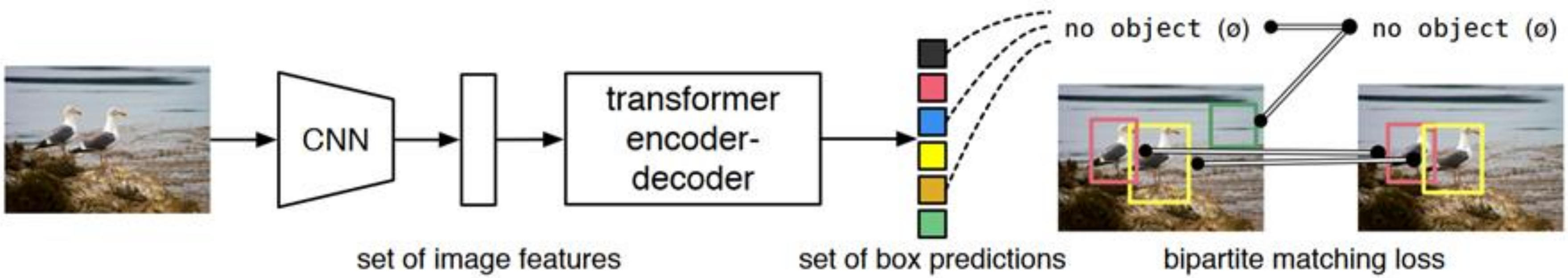
	backbone	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{bb}_{S}	AP^{bb}_{M}	AP^{bb}_{L}
Faster R-CNN+++ [19]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [27]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [21]	Inception-ResNet-v2 [41]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [39]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
Faster R-CNN, RoIAlign	ResNet-101-FPN	37.3	59.6	40.3	19.8	40.2	48.8
Mask R-CNN	ResNet-101-FPN	38.2	60.3	41.7	20.1	41.1	50.2
Mask R-CNN	ResNeXt-101-FPN	**39.8**	**62.3**	**43.4**	**22.1**	**43.2**	51.2

He, K., Gkioxari, G., Dollár, P., & Girshick, R. Mask R-CNN. ICCV 2017.

DETR (ECCV 2020)

End-to-End Object Detection with Transformers

- DETR directly predicts (in parallel) the final set of detections by combining a CNN with a transformer architecture. No need of NMS!
- During training, bipartite matching uniquely assigns predictions with ground truth boxes. Prediction with no match should yield a “no object” (∅) class prediction.

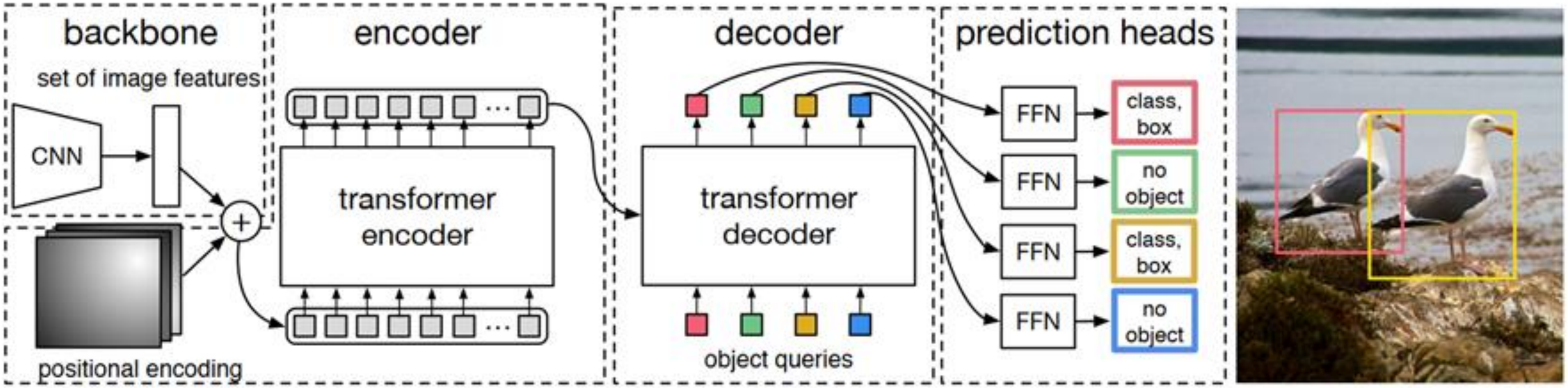


Carion et al. "[End-to-end object detection with transformers.]()" ECCV, 2020.

DETR (ECCV 2020)

End-to-End Object Detection with Transformers

- Encoder input: CNN features + positional encoding.
- Decoder input:
  - a fixed number N of learned positional embeddings (N=100), **object queries.**
  - also attends to the encoder output.
- Output embeddings of the decoder go to a shared feed forward network (FFN) that predicts a detection (class and bbox) or a “no object” class.



Carion et al. "End-to-end object detection with transformers." ECCV, 2020.

DETR (ECCV 2020)

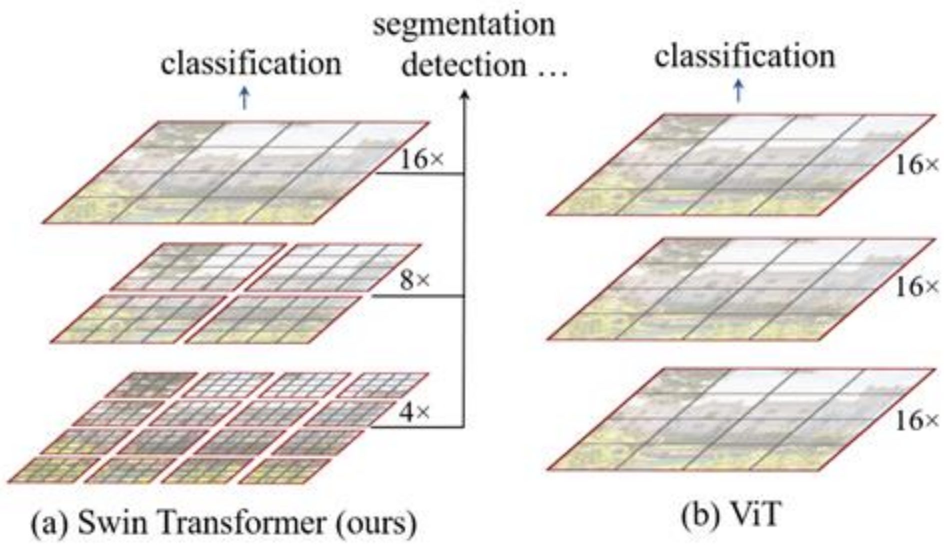
Model	GFLOPS/FPS	#params	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster RCNN-DC5	320/16	166M	39.0	60.5	42.3	21.4	43.5	52.5
Faster RCNN-FPN	180/26	42M	40.2	61.0	43.8	24.2	43.5	52.0
Faster RCNN-R101-FPN	246/20	60M	42.0	62.5	45.9	25.2	45.6	54.6
Faster RCNN-DC5+	320/16	166M	41.1	61.4	44.3	22.9	45.9	55.0
Faster RCNN-FPN+	180/26	42M	42.0	62.1	45.5	26.6	45.4	53.4
Faster RCNN-R101-FPN+	246/20	60M	44.0	63.9	**47.8**	**27.2**	48.1	56.0
DETR	86/28	41M	42.0	62.4	44.2	20.5	45.8	61.1
DETR-DC5	187/12	41M	43.3	63.1	45.9	22.5	47.3	61.1
DETR-R101	152/20	60M	43.5	63.8	46.4	21.9	48.0	61.8
DETR-DC5-R101	253/10	60M	**44.9**	**64.7**	47.7	23.7	**49.5**	**62.3**

Carion et al. "End-to-end object detection with transformers." ECCV, 2020.

Swin Transformer (2021)

The hierarchical Transformers (e.g., Swin Transformers) reintroduce several ConvNet priors.

Makes Transformers practically viable as a generic vision backbone and demonstrate remarkable performance on a wide variety of vision tasks.



Liu et al. "[Swin Transformer: Hierarchical Vision Transformer using Shifted Windows]" ICCV, 2021.

Swin Transformer (2021)

(b) Various backbones w. Cascade Mask R-CNN

	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	param	FLOPs	FPS
DeiT-S†	48.0	67.2	51.7	41.4	64.2	44.3	80M	889G	10.4
R50	46.3	64.3	50.5	40.1	61.7	43.4	82M	739G	18.0
Swin-T	**50.5**	**69.3**	**54.9**	**43.7**	**66.6**	**47.1**	86M	745G	15.3
X101-32	48.1	66.5	52.4	41.6	63.9	45.2	101M	819G	12.8
Swin-S	**51.8**	**70.4**	**56.3**	**44.7**	**67.9**	**48.5**	107M	838G	12.0
X101-64	48.3	66.4	52.3	41.7	64.0	45.1	140M	972G	10.4
Swin-B	**51.9**	**70.9**	**56.5**	**45.0**	**68.4**	**48.7**	145M	982G	11.6

Liu et al. "[Swin Transformer: Hierarchical Vision Transformer using Shifted Windows]" ICCV, 2021.

ConvNext (2022)

"Modernize" a standard ResNet toward the design of a vision Transformer: "Patchify" input, ResNeXt, Larger Kernels, ReLU->GeLU, fewer activations, BN->LN, modern optimizer, better augmentations, etc…

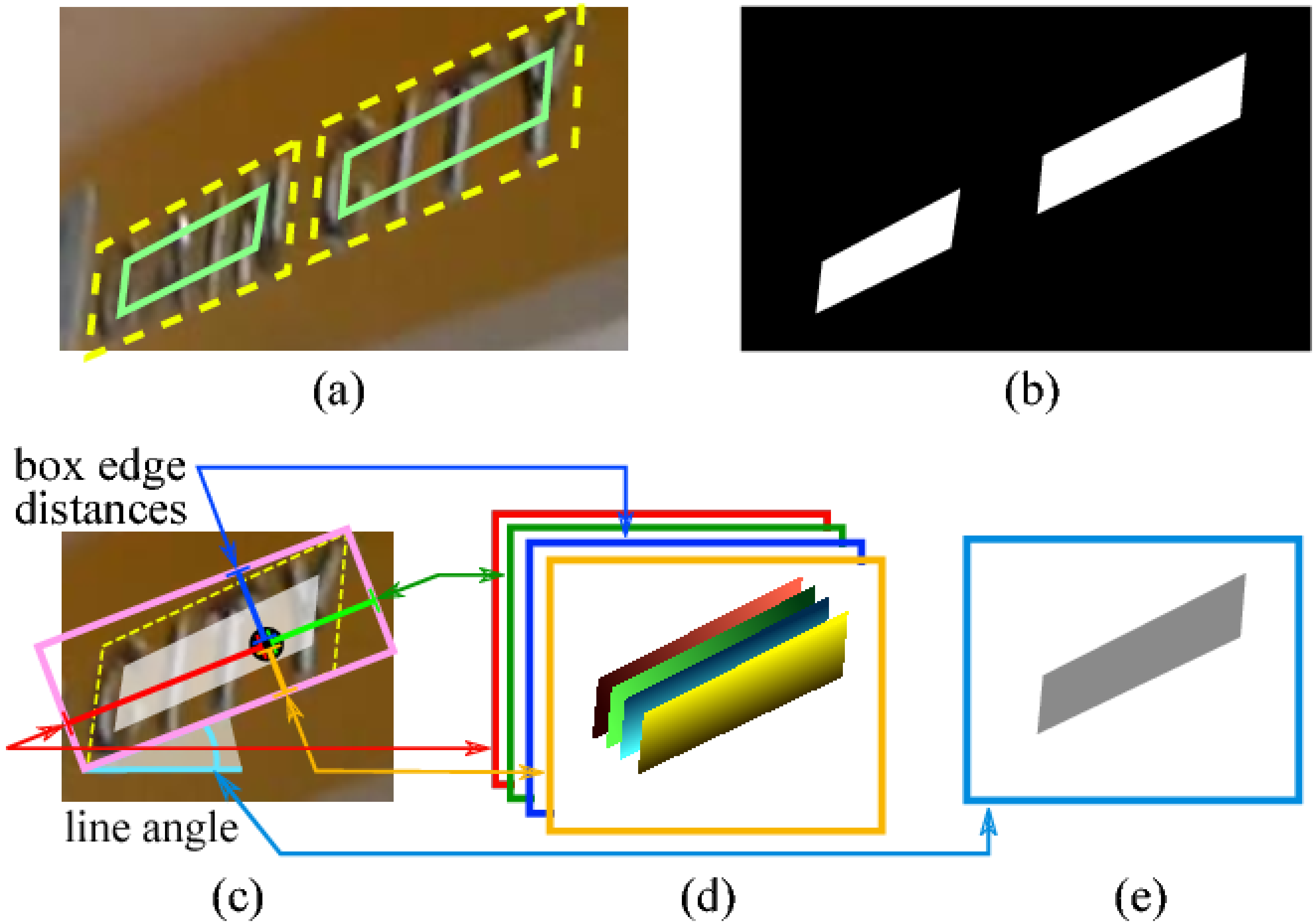
Constructed entirely from standard ConvNet modules.

ConvNeXts compete favorably with Transformers in terms of accuracy and scalability, outperforming Swin Transformers on COCO detection, while maintaining the simplicity and efficiency of standard ConvNets.

backbone	FLOPs	FPS	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	A
		Mask-RCNN 3× schedule				
○ Swin-T	267G	23.1	46.0	68.1	50.3	
• ConvNeXt-T	262G	25.6	**46.2**	67.9	50.8	
		Cascade Mask-RCNN 3× schedule				
• ResNet-50	739G	16.2	46.3	64.3	50.5	
• X101-32	819G	13.8	48.1	66.5	52.4	
• X101-64	972G	12.6	48.3	66.4	52.3	
○ Swin-T	745G	12.2	50.4	69.2	54.7	
• ConvNeXt-T	741G	13.5	**50.4**	69.1	54.8	
○ Swin-S	838G	11.4	51.9	70.7	56.3	
• ConvNeXt-S	827G	12.0	**51.9**	70.8	56.5	
○ Swin-B	982G	10.7	51.9	70.5	56.4	
• ConvNeXt-B	964G	11.4	**52.7**	71.3	57.2	
○ Swin-B‡	982G	10.7	53.0	71.8	57.5	
• ConvNeXt-B‡	964G	11.5	**54.0**	73.1	58.8	
○ Swin-L‡	1382G	9.2	53.9	72.4	58.8	
• ConvNeXt-L‡	1354G	10.0	**54.8**	73.8	59.8	
• ConvNeXt-XL‡	1898G	8.6	**55.2**	74.2	59.9	

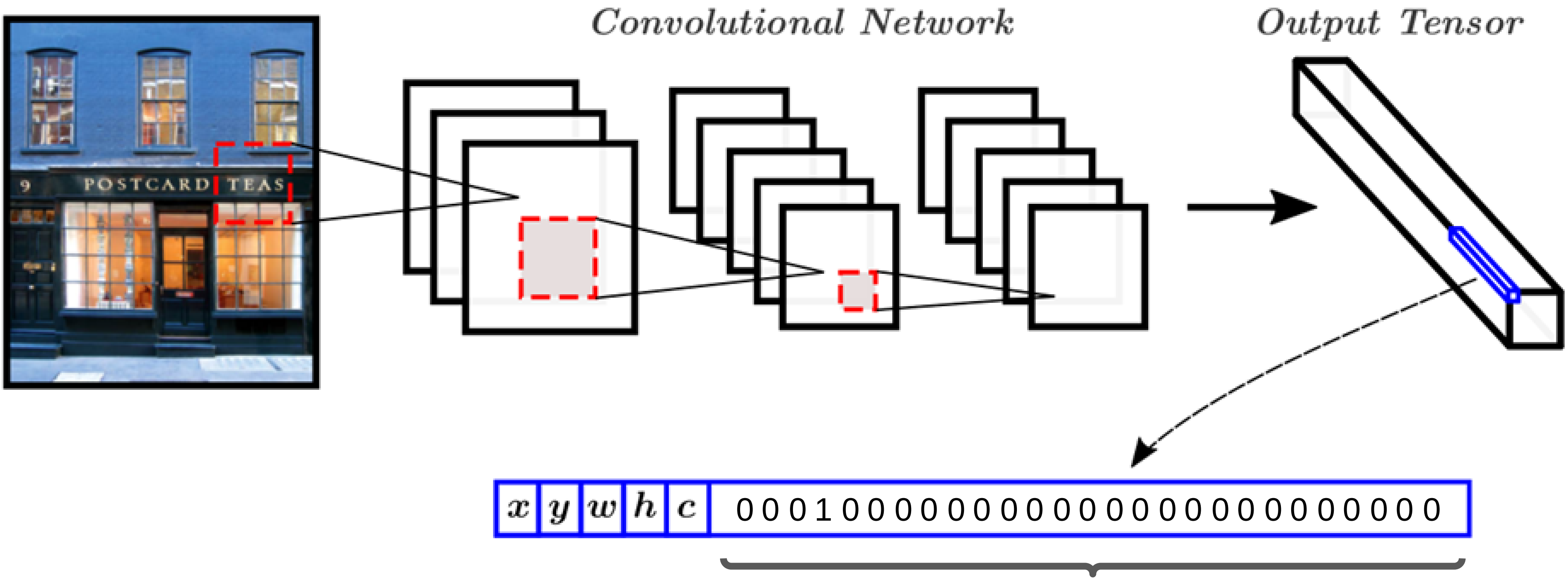
Liu et al. "A ConvNet for the 2020s" CVPR, 2022.

Other ideas in Object Detection: Rotated objects



Zhou, Xinyu, et al. "EAST: an efficient and accurate scene text detector." CVPR 2017

Other ideas in Object Detection: STR



One-hot classification

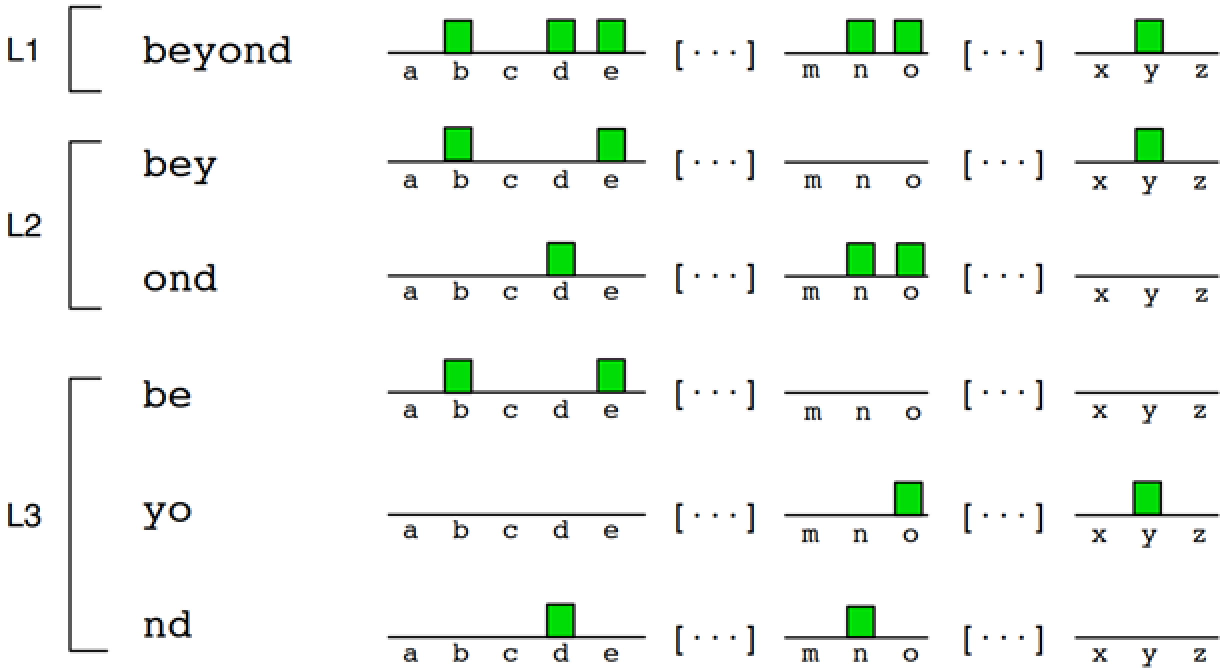
How many classes?

L.Gomez, A. Mafla, M. Rusiñol, D. Karatzas. "Single Shot Scene Text Retrieval", ECCV 2018.

Label Embedding (PHOC)

Text strings are embedded into a d-dimensional binary space: Pyramidal Histogram Of Characters (PHOC)

PHOC encodes if a particular character appears in a particular region of the string



Almazán, Jon, et al. "Word spotting and recognition with embedded attributes." IEEE transactions on pattern analysis and machine intelligence 36.12 (2014): 2552-2566.

Label Embeddings

Attribute-based recognition

Comparison of:

- **Direct Attribute Prediction (DAP):** compute attribute probabilities + combine scores

Lampert, Nickisch, Harmeling, "Learning To Detect Unseen Object Classes by Between-Class Attribute Transfer", CVPR'09

- **Attribute Label Embedding (ALE)**: embed classes + bilinear compatibility

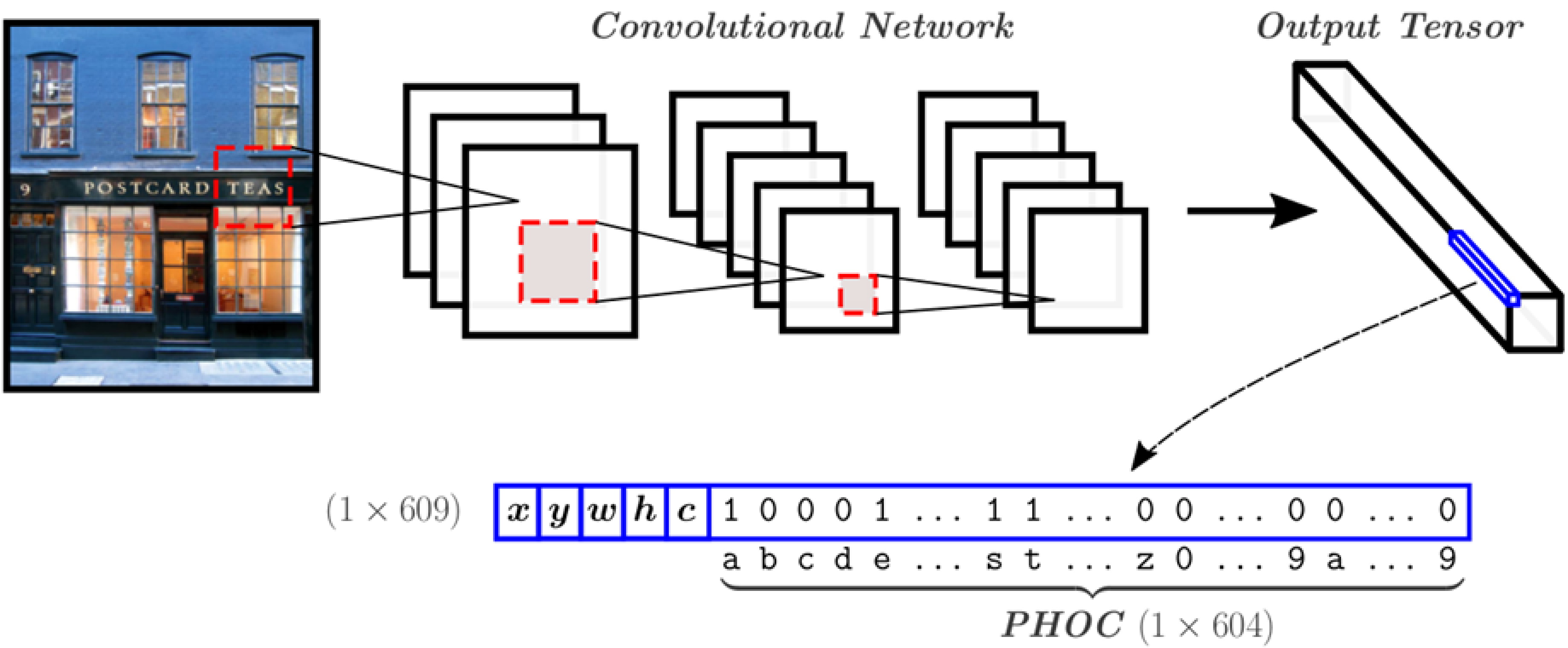
Akata, Perronnin, Harchaoui, Schmid, "Label-embedding for attribute-based classification", CVPR'13

→ ALE outperforms DAP by large margin on zero-shot bird recognition

See also: Alabdulmohsin, Cissé, Zhang, "Is attribute-based zero-shot learning an ill-posed strategy?", EACL'16.

Slide credit: Florent Perronnin. "Output Embedding for Large Scale Computer Vision", ECCV 2018.

Single Shot Text Detection and Recognition



L.Gomez, A. Mafla, M. Rusiñol, D. Karatzas. "Single Shot Scene Text Retrieval", ECCV 2018.

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.
- Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, Ali Farhadi; You Only Look Once: Unified, Real-Time Object Detection.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; Deep Residual Learning for Image Recognition.
- Joseph Redmon and Ali Farhadi; YOLO9000: Better, Faster, Stronger.
- M. Oquab and L. Bottou and I. Laptev and J. Sivic; Is object localization for free? - Weakly-supervised learning with convolutional neural networks.

References

- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). SSD: Single shot multibox detector. ECCV 2016.
- Bodla, N., Singh, B., Chellappa, R., & Davis, L. S. Soft-NMS--improving object detection with one line of code. ICCV 2017.
- Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. CVPR 2017.
- Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. Focal loss for dense object detection. ICCV 2017.
- He, K., Gkioxari, G., Dollár, P., & Girshick, R. Mask R-CNN. ICCV 2017.
- Carion et al."End-to-end object detection with transformers. ECCV, 2020.
- Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., & Liang, J. EAST: an efficient and accurate scene text detector. CVPR 2017.
- L.Gomez, A. Mafla, M. Rusiñol, D. Karatzas. Single Shot Scene Text Retrieval, ECCV 2018.