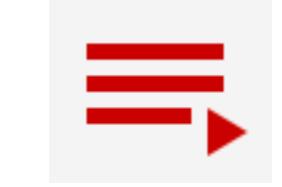




Boulder

Computer Vision; Object Detection; One Stage Detectors



[YouTube Playlist](#)

Maziar Raissi

Assistant Professor

Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu



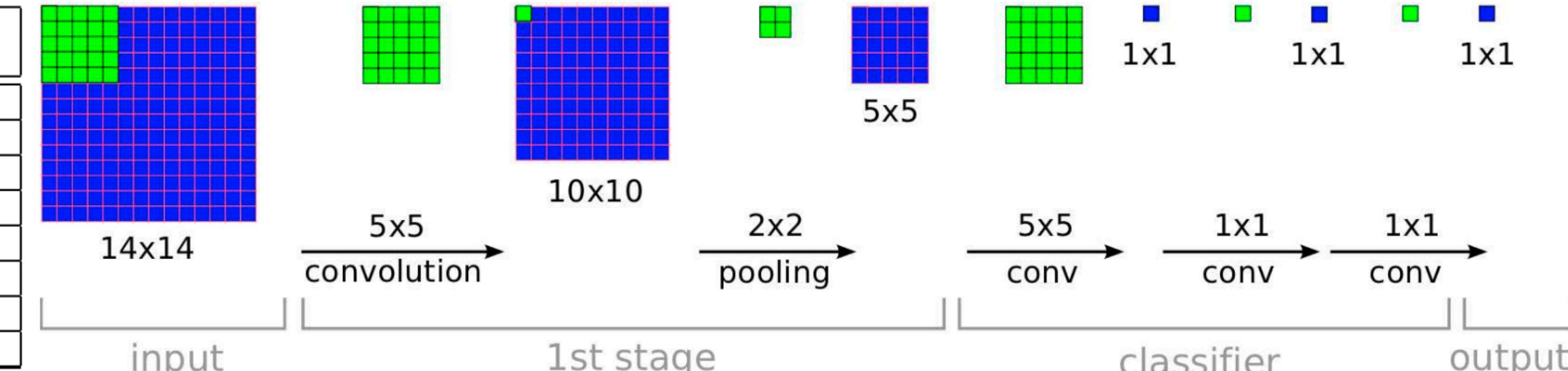
Boulder

OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks



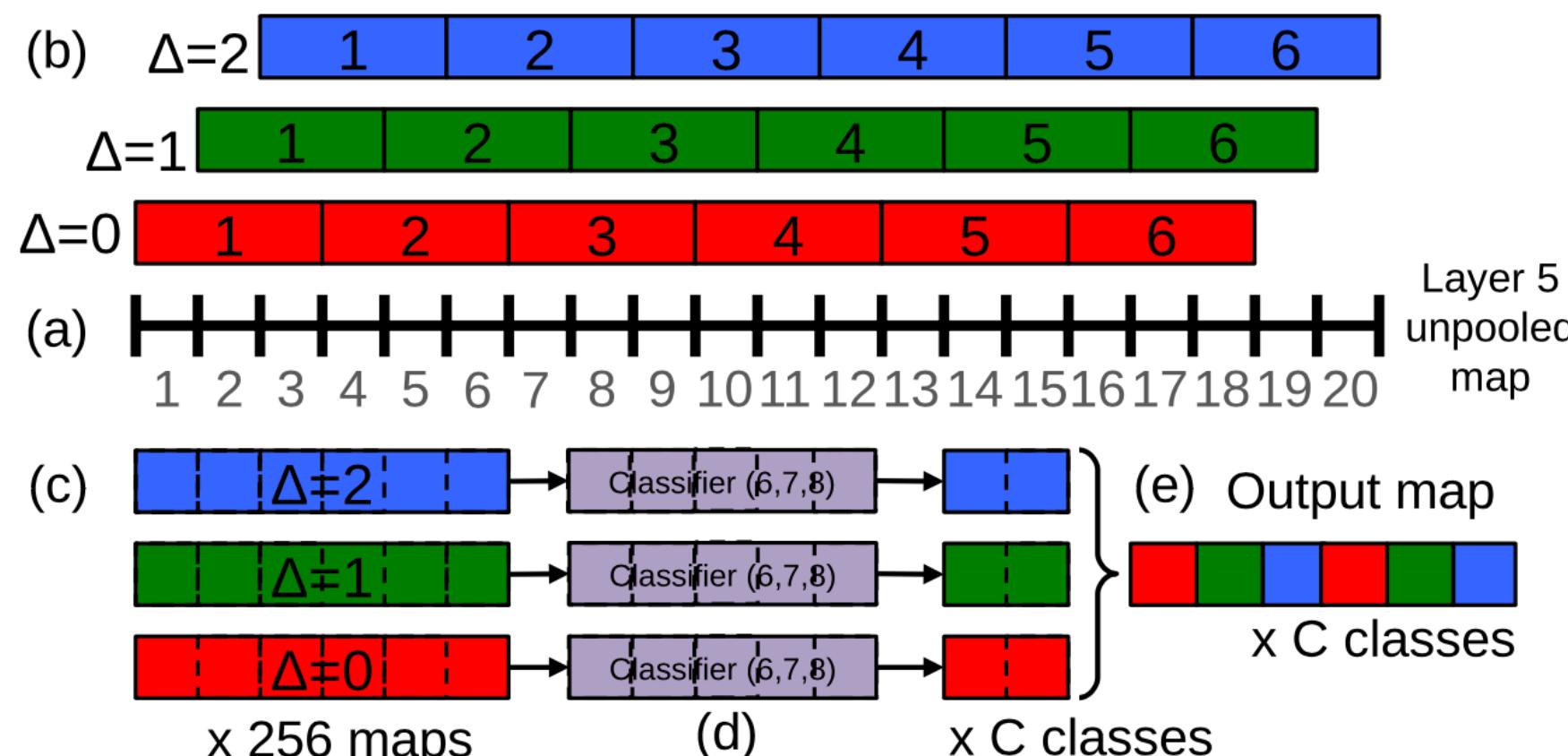
[YouTube Video](#)

Layer	1	2	3	4	5	6	7	8	Output 9
Stage	conv + max	conv + max	conv	conv	conv	conv + max	full	full	full
# channels	96	256	512	512	1024	1024	4096	4096	1000
Filter size	7x7	7x7	3x3	3x3	3x3	3x3	-	-	-
Conv. stride	2x2	1x1	1x1	1x1	1x1	1x1	-	-	-
Pooling size	3x3	2x2	-	-	-	3x3	-	-	-
Pooling stride	3x3	2x2	-	-	-	3x3	-	-	-
Zero-Padding size	-	-	1x1x1x1	1x1x1x1	1x1x1x1	1x1x1x1	-	-	-
Spatial input size	221x221	36x36	15x15	15x15	15x15	15x15	5x5	1x1	1x1

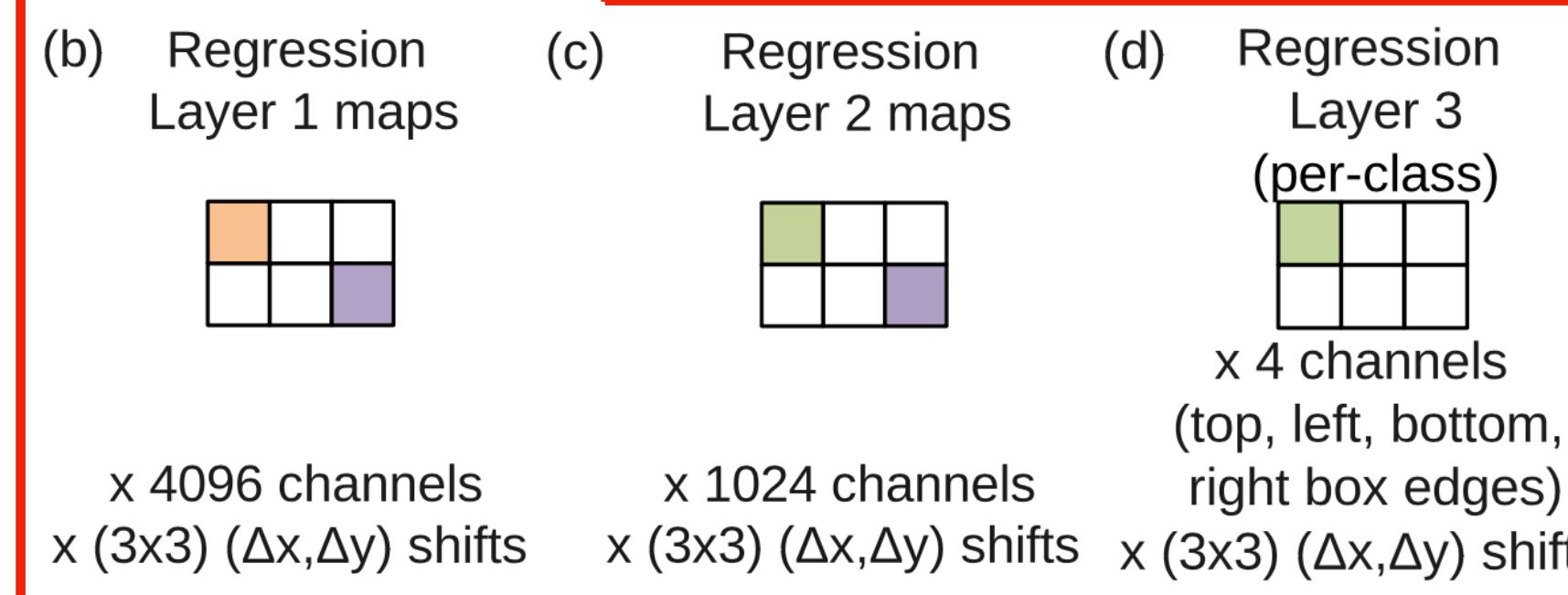
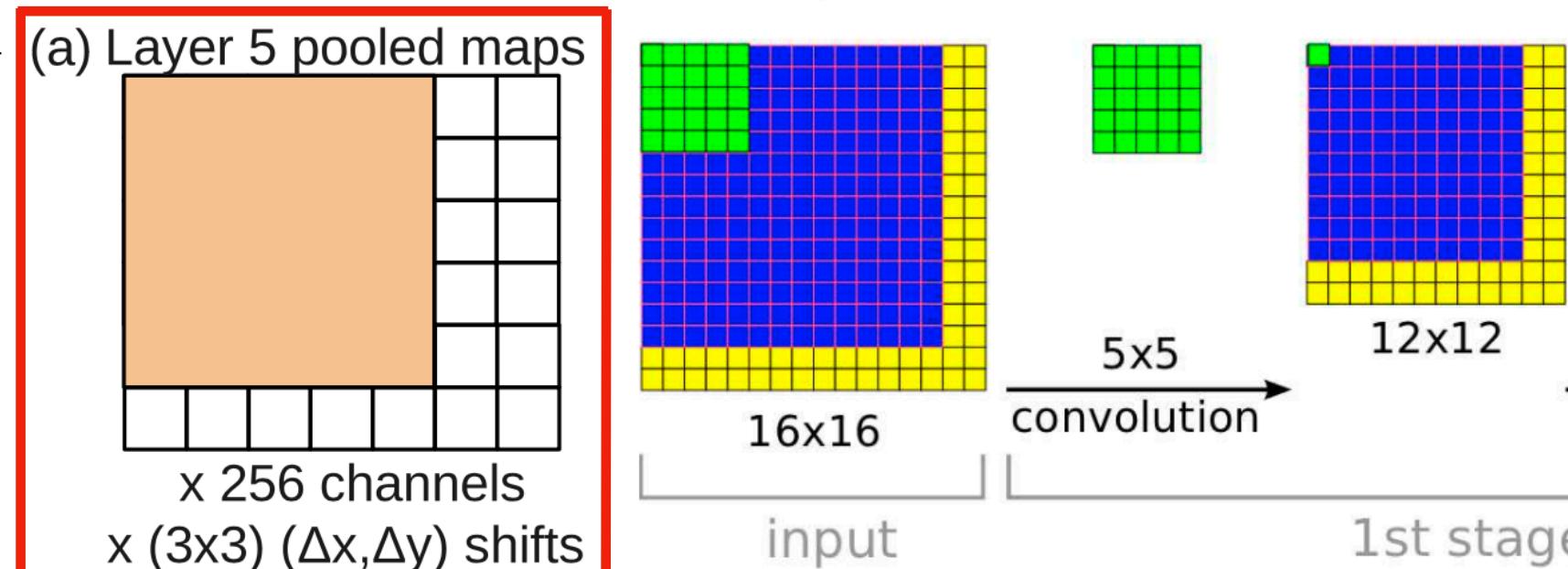


$36 = 2 \cdot 3 \cdot 2 \cdot 3 \rightarrow$ total subsampling ratio in the network

Scale	Input size	Layer 5 pre-pool	Layer 5 post-pool	Classifier map (pre-reshape)	Classifier map size
1	245x245	17x17	(5x5)x(3x3)	(1x1)x(3x3)x C	3x3xC
2	281x317	20x23	(6x7)x(3x3)	(2x3)x(3x3)x C	6x9xC
3	317x389	23x29	(7x9)x(3x3)	(3x5)x(3x3)x C	9x15xC
4	389x461	29x35	(9x11)x(3x3)	(5x7)x(3x3)x C	15x21xC
5	425x497	32x35	(10x11)x(3x3)	(6x7)x(3x3)x C	18x24xC
6	461x569	35x44	(11x14)x(3x3)	(7x10)x(3x3)x C	21x30xC



Yields a total subsampling ratio of $12 = 36/3$ instead of 36
 $245 = 221 + 2 \cdot 12 \rightarrow$ input size



Greedy Merge Strategy

$C_s \rightarrow$ set of classes in the top k for each scale $s = 1, \dots, 6$

$B_s \rightarrow$ set of bounding boxes predicted by the regressor network for each class in C_s

Assign $B \leftarrow \bigcup_s B_s$

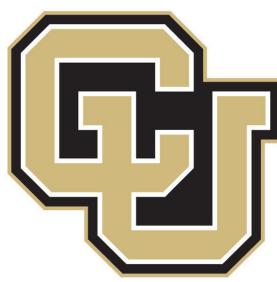
Repeat merging until done:

$$(b_1^*, b_2^*) = \operatorname{argmin}_{b_1 \neq b_2 \in B} \text{match_score}(b_1, b_2)$$

If $\text{match_score}(b_1^*, b_2^*) > t$, stop.

Otherwise, set $B \leftarrow B \setminus \{b_1^*, b_2^*\} \cup \text{box_merge}(b_1^*, b_2^*)$

Compute match score using the sum of the distance between centers of the two bounding boxes and the intersection area of the boxes. `box_merge` computes the average of the bounding boxes' coordinates

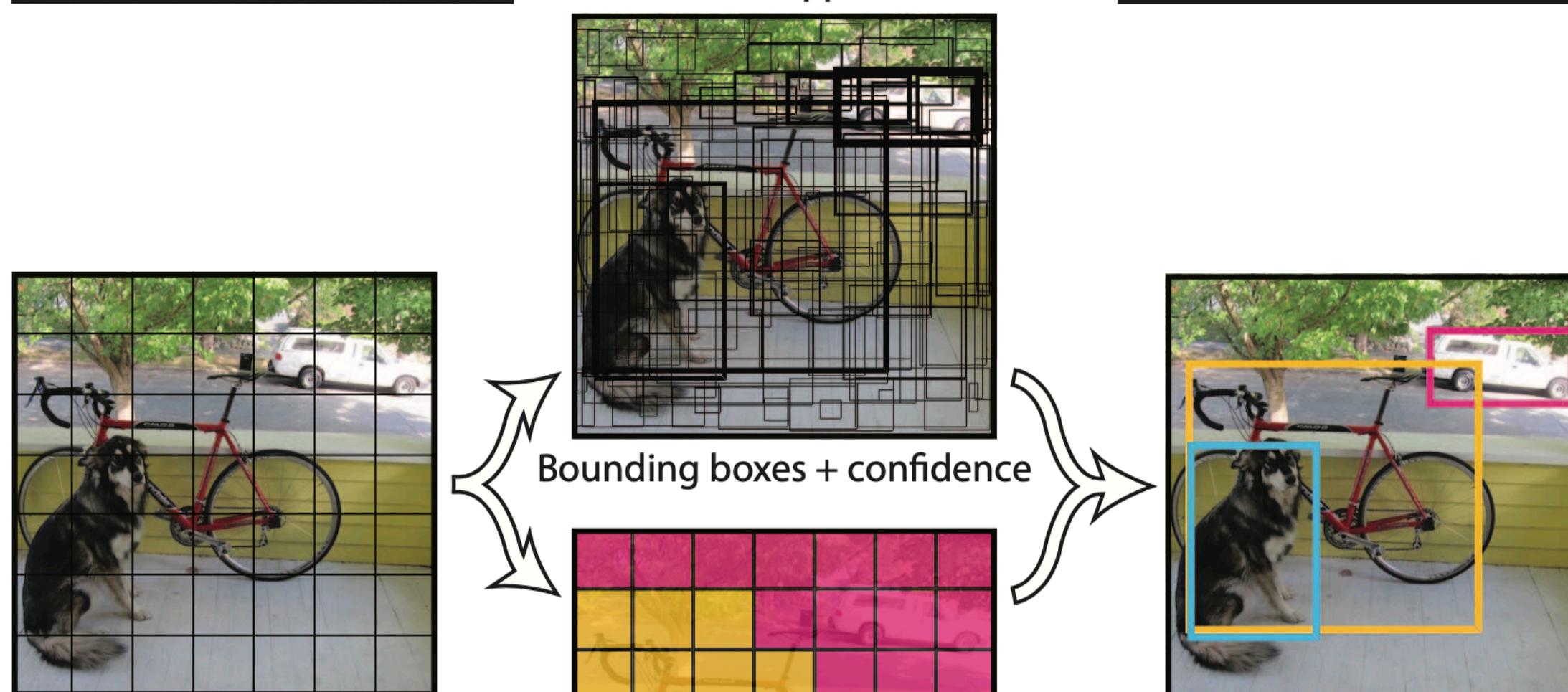
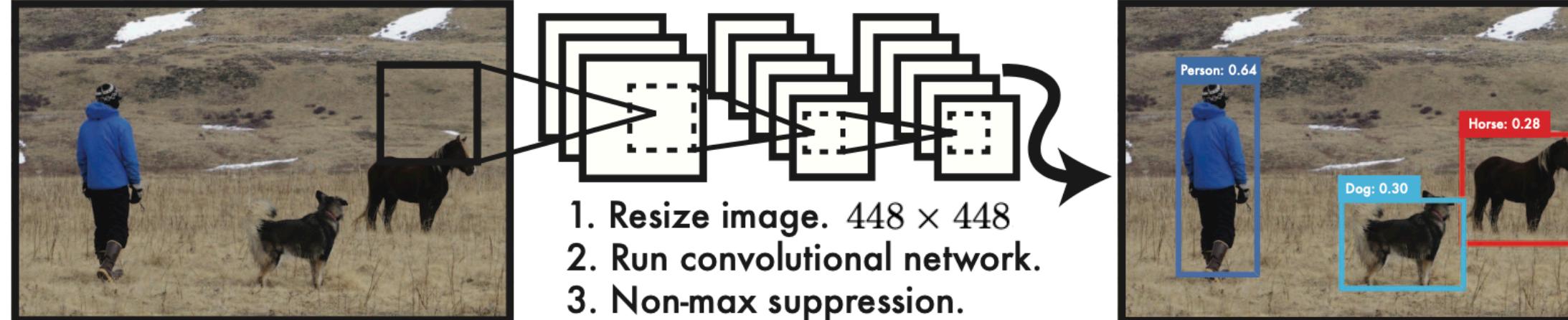


Boulder

You Only Look Once: Unified, Real-Time Object Detection



[YouTube Video](#)



- Divide the input image into an $S \times S$ grid
- If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.
- Each grid cell predicts B bounding boxes and confidence scores for those boxes

Confidence $\rightarrow \Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}}$
The confidence score reflects how confident the model is that the box contains an object and also how accurate it thinks the box is that it predicts.

$(x, y) \rightarrow$ center of the box relative to the bounds of the grid cell

$(w, h) \rightarrow$ width and height relative to the whole image

Each grid cell also predicts C conditional class probabilities, $\Pr(\text{Class}_i | \text{Object})$

$$\Pr(\text{Class}_i | \text{Object}) * \Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}}$$

class-specific confidence scores

$S \times S \times (B * 5 + C)$ tensor

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right]$$

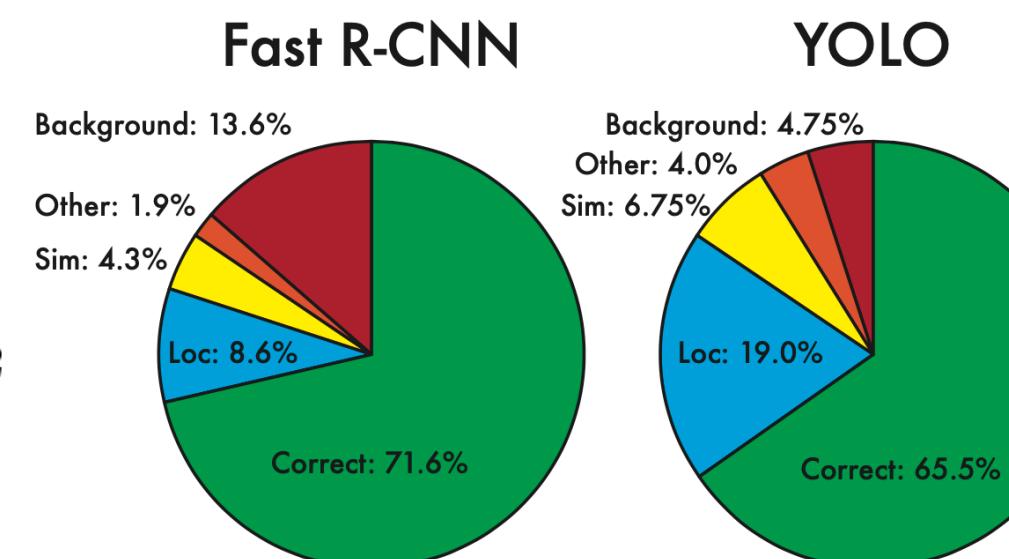
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{noobj}} (C_i - \hat{C}_i)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

$$\lambda_{\text{coord}} = 5 \text{ and } \lambda_{\text{noobj}} = .5.$$

$S = 7$
 $B = 2$
 $C = 20$



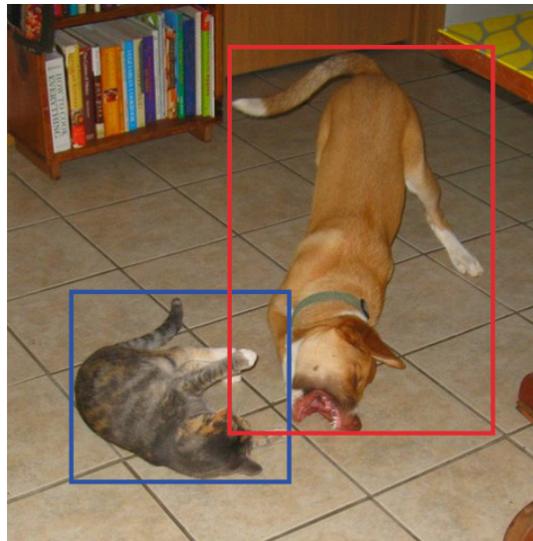


Boulder

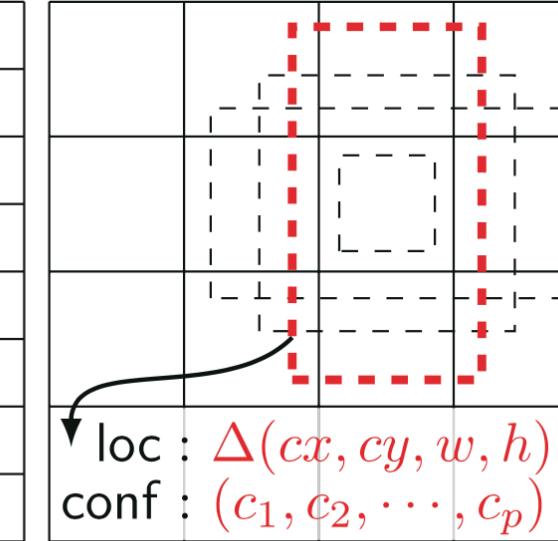
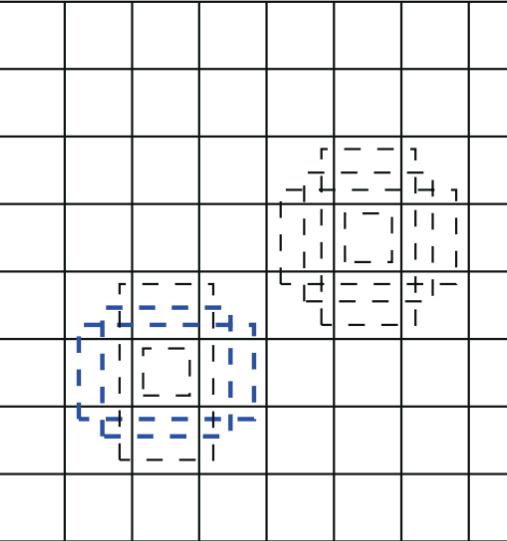


[YouTube Video](#)

SSD: Single Shot MultiBox Detector



(a) Image with GT boxes



$$8732 = 4 * 38 * 38 + 6 * 19 * 19 + 6 * 10 * 10 + 6 * 5 * 5 + 4 * 3 * 3 + 4 * 1 * 1$$

Training Objective

$$L(x, c, l, g) = \frac{1}{N} (L_{conf}(x, c) + \alpha L_{loc}(x, l, g))$$

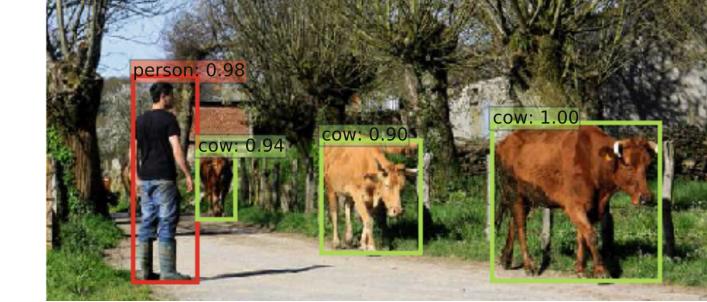
predicted box parameters
ground truth box parameters

number of matched default boxes

L_{conf} → softmax loss

L_{loc} → Smooth L1 loss

Extra Feature Layers



Scales and Aspect Ratios for Default Boxes

$$s_k = s_{\min} + \frac{s_{\max} - s_{\min}}{m-1}(k-1), \quad k \in [1, m]$$

scale of the highest layer
scale of the lowest layer

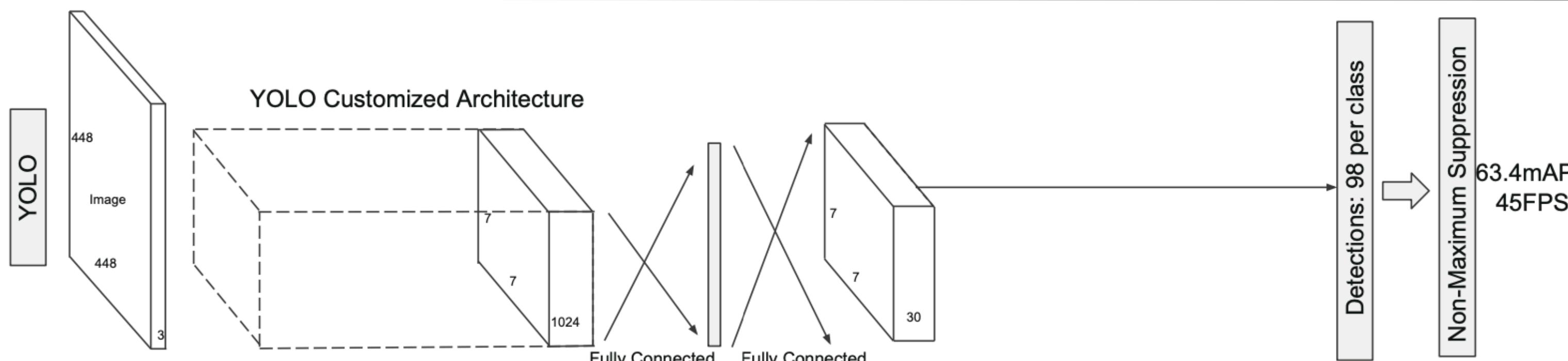
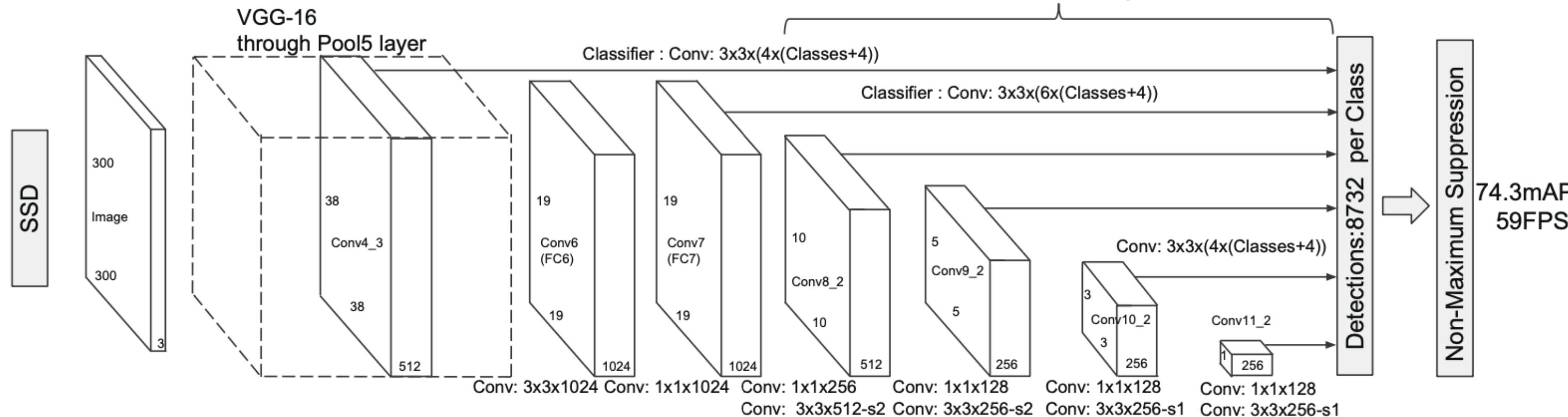
$a_r \in \{1, 2, 3, 1/2, 1/3\} \rightarrow$ aspect ratios

$$w_k^a = s_k \sqrt{a_r} \rightarrow \text{width}$$

$$h_k^a = s_k / \sqrt{a_r} \rightarrow \text{height}$$

$$s'_k = \sqrt{s_k s_{k+1}} \rightarrow \text{scale of the additional default box for aspect ratio 1}$$

6 default boxes per feature map location



Method	mAP	FPS	Test batch size	# Boxes
Faster R-CNN [2] (VGG16)	73.2	7	1	300
Faster R-CNN [2] (ZF)	62.1	17	1	300
YOLO [5]	63.4	45	1	98
Fast YOLO [5]	52.7	155	1	98
SSD300	74.3	46	1	8732
SSD512	76.8	19	1	24564
SSD300	74.3	59	8	8732
SSD512	76.8	22	8	24564



Boulder

YOLO9000: Better, Faster, Stronger



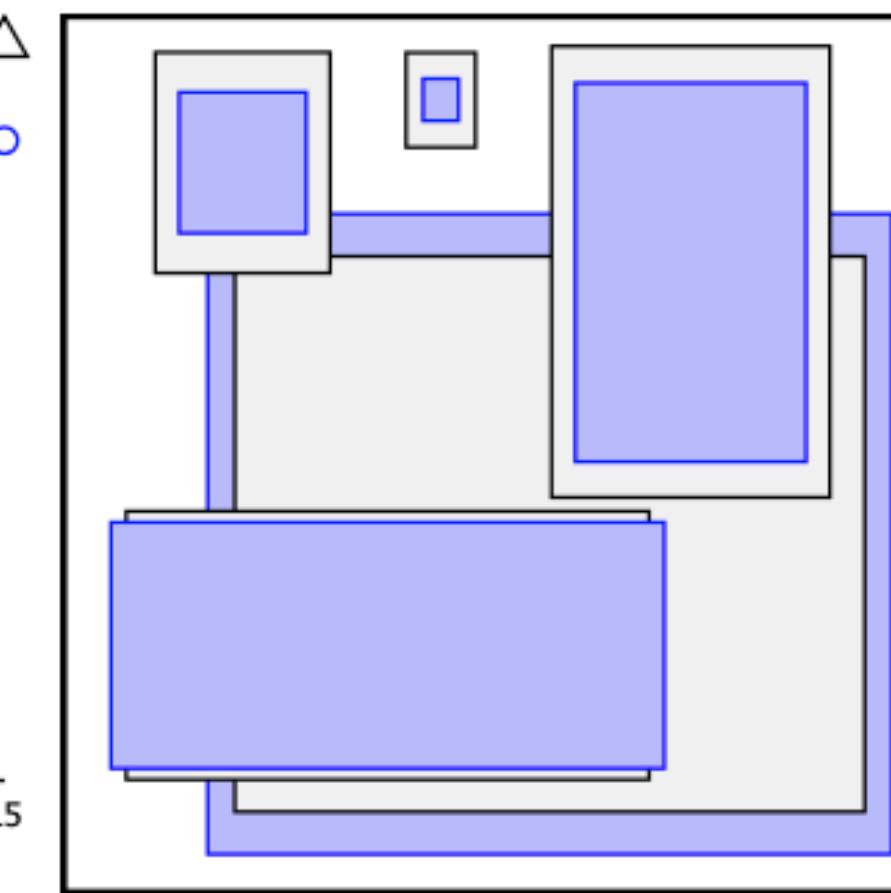
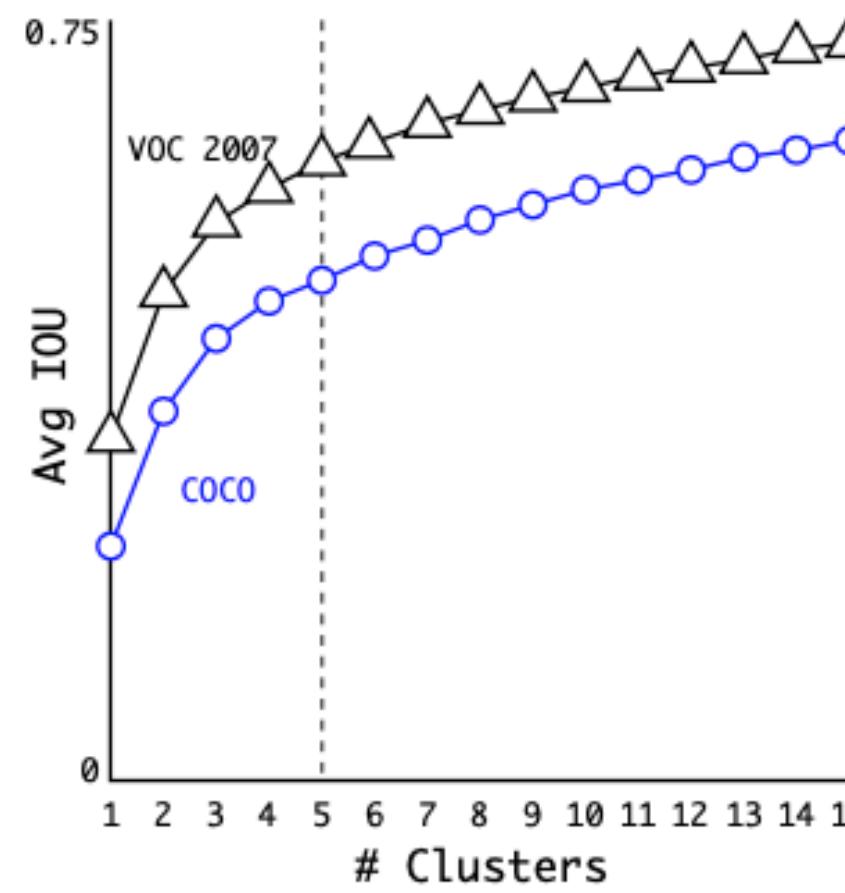
[YouTube Video](#)

YOLO9000 can detect over 9000 object categories.

	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

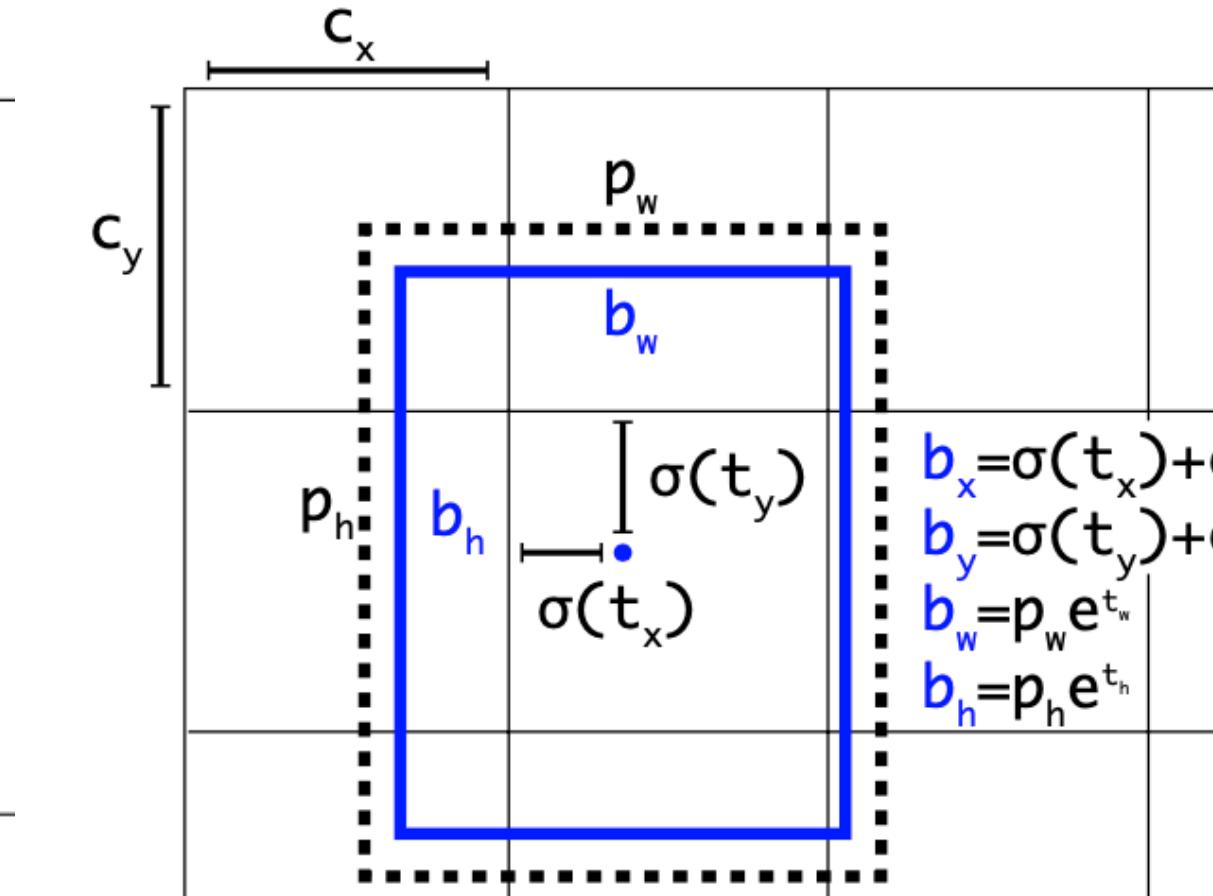
Dimension Clusters

$$d(\text{box}, \text{centroid}) = 1 - \text{IOU}(\text{box}, \text{centroid})$$



k-means clustering on the training set box dimensions

Direct location prediction



$$b_x = \sigma(t_x) + c_x$$

$$b_y = \sigma(t_y) + c_y$$

$$b_w = p_w e^{t_w}$$

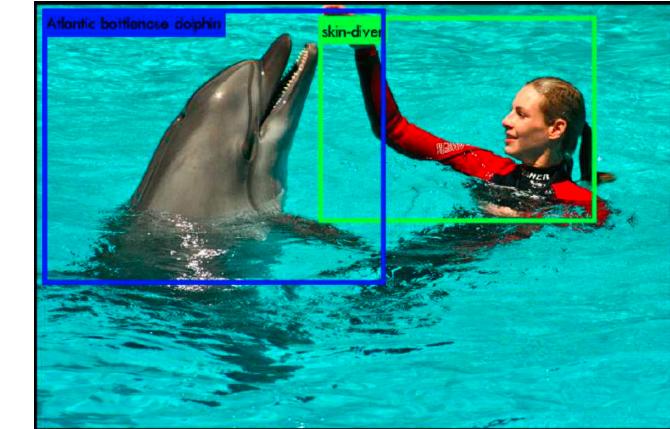
$$b_h = p_h e^{t_h}$$

$$\Pr(\text{object}) * \text{IOU}(b, \text{object}) = \sigma(t_o)$$

$(c_x, c_y) \rightarrow$ cell offset from top left corner of the image

$(t_x, t_y, t_w, t_h, t_o) \rightarrow$ predictions of the network

$(p_w, p_h) \rightarrow$ bounding box prior width & height



Joint classification and detection

See the paper!

Passthrough Layer

To detect small objects well, the $26 \times 26 \times 512$ feature maps from earlier layer is mapped into $13 \times 13 \times 2048$ feature map, then concatenated with the original 13×13 feature maps for detection.

Hierarchical classification

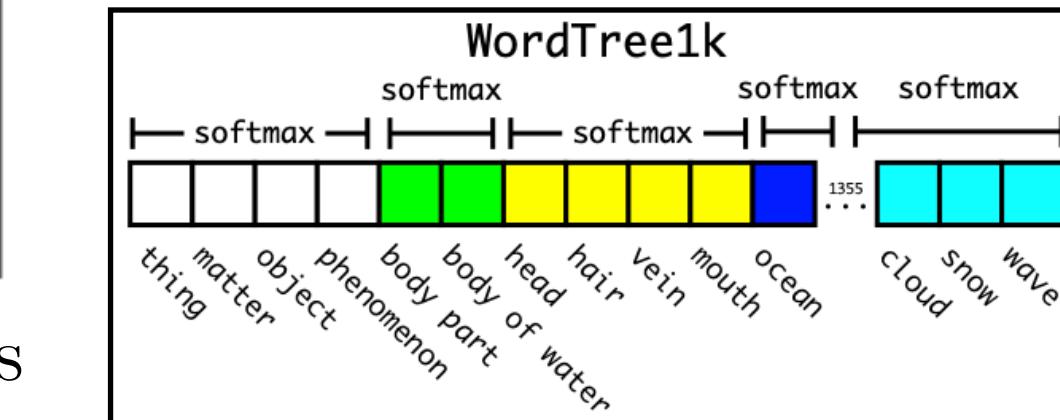
$$\Pr(\text{Norfolk terrier}) = \Pr(\text{Norfolk terrier}|\text{terrier})$$

$$* \Pr(\text{terrier}|\text{hunting dog})$$

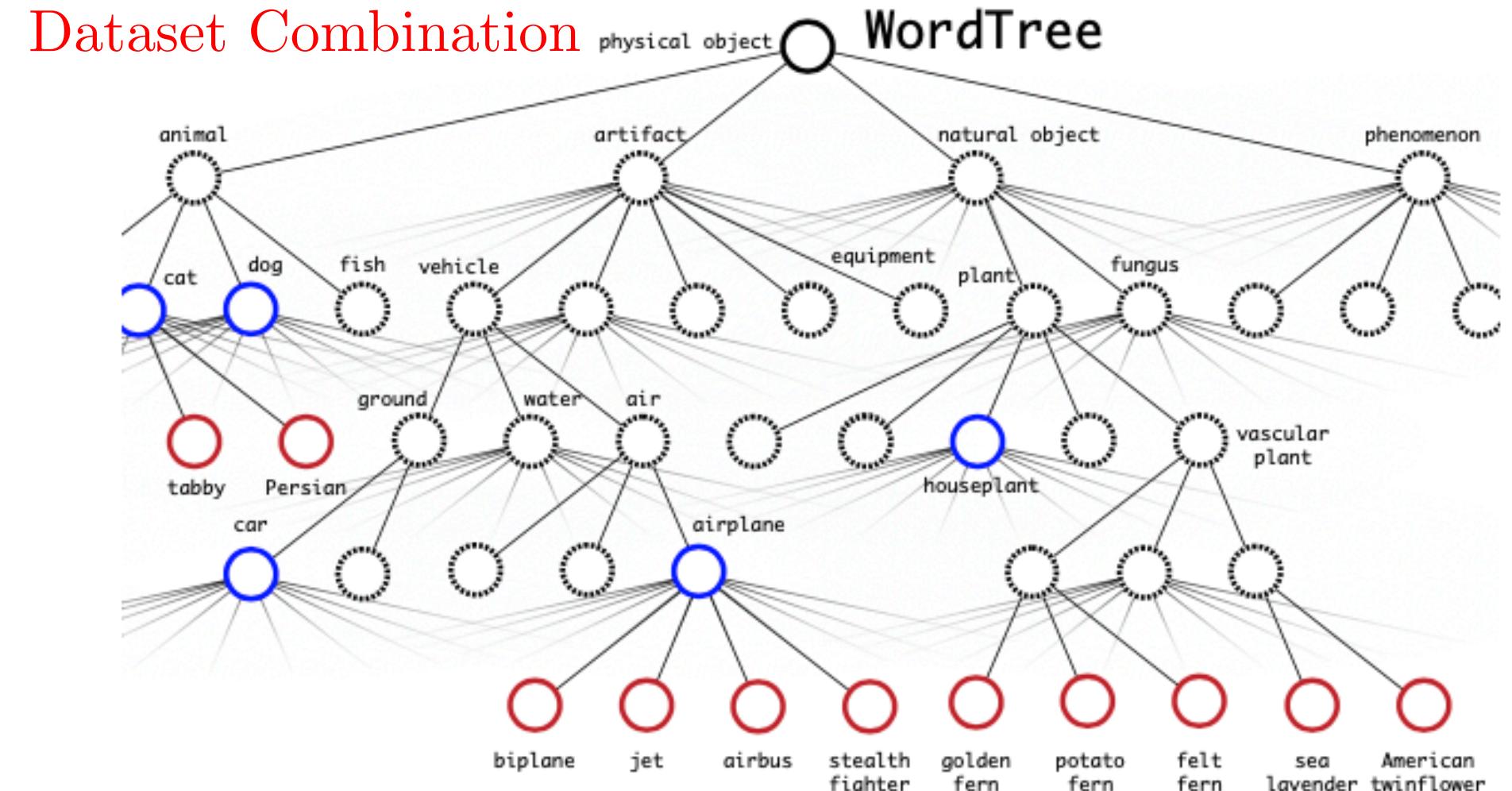
* ... *

$$* \Pr(\text{mammal}|\Pr(\text{animal}))$$

$$* \Pr(\text{animal}|\text{physical object})$$



Dataset Combination



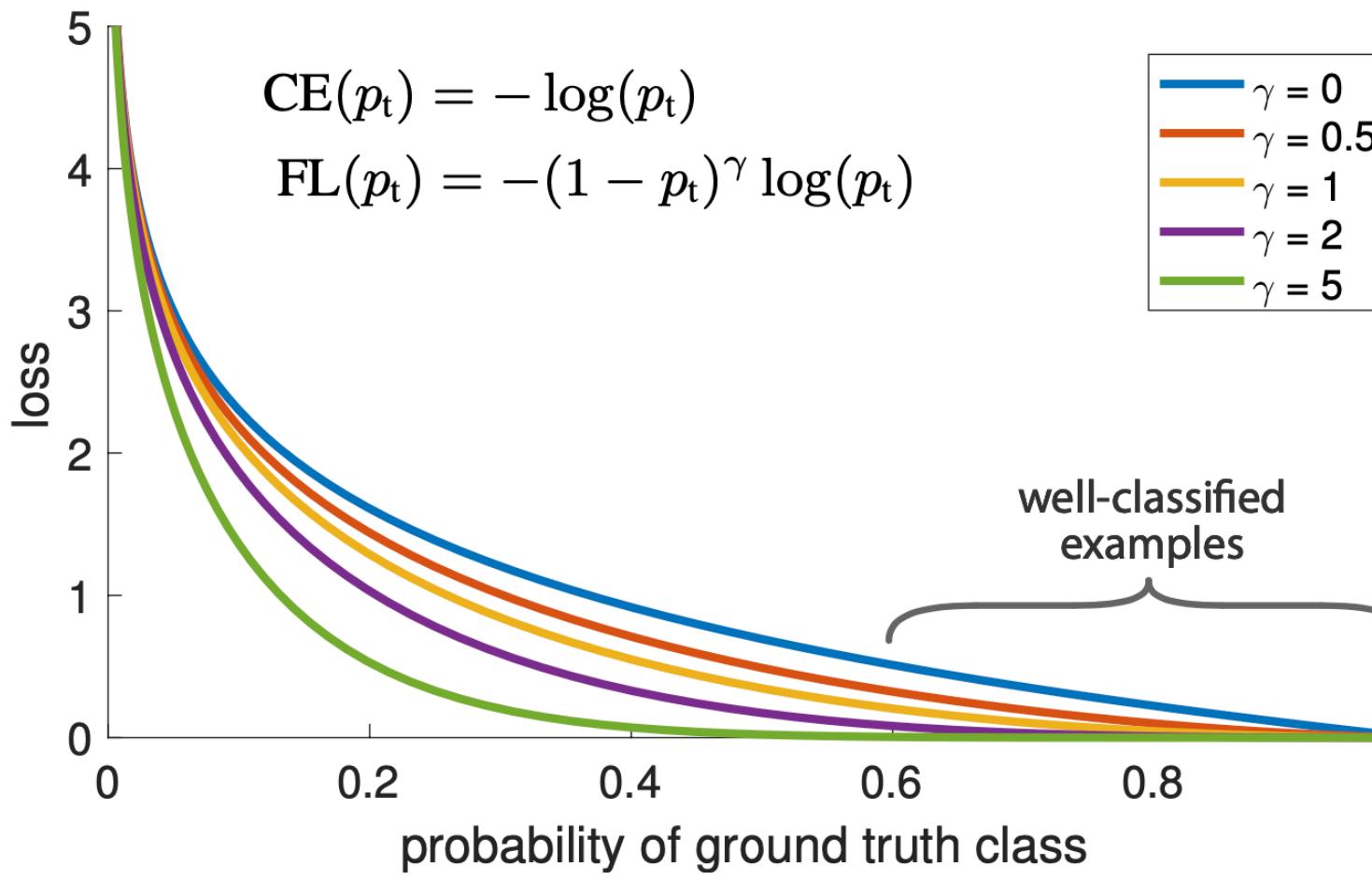


Boulder



Focal Loss for Dense Object Detection

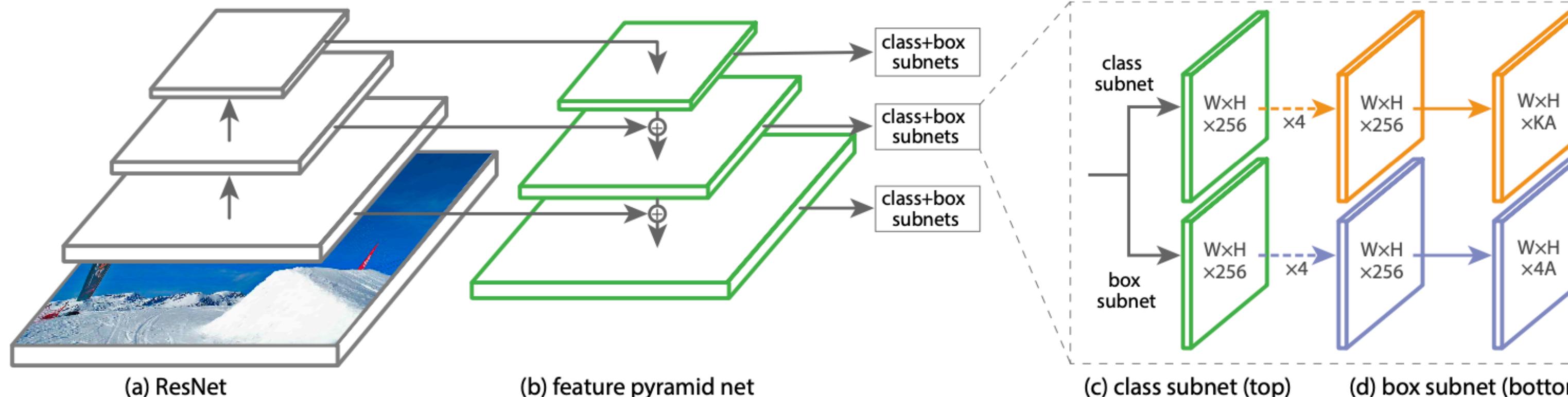
[YouTube Video](#)



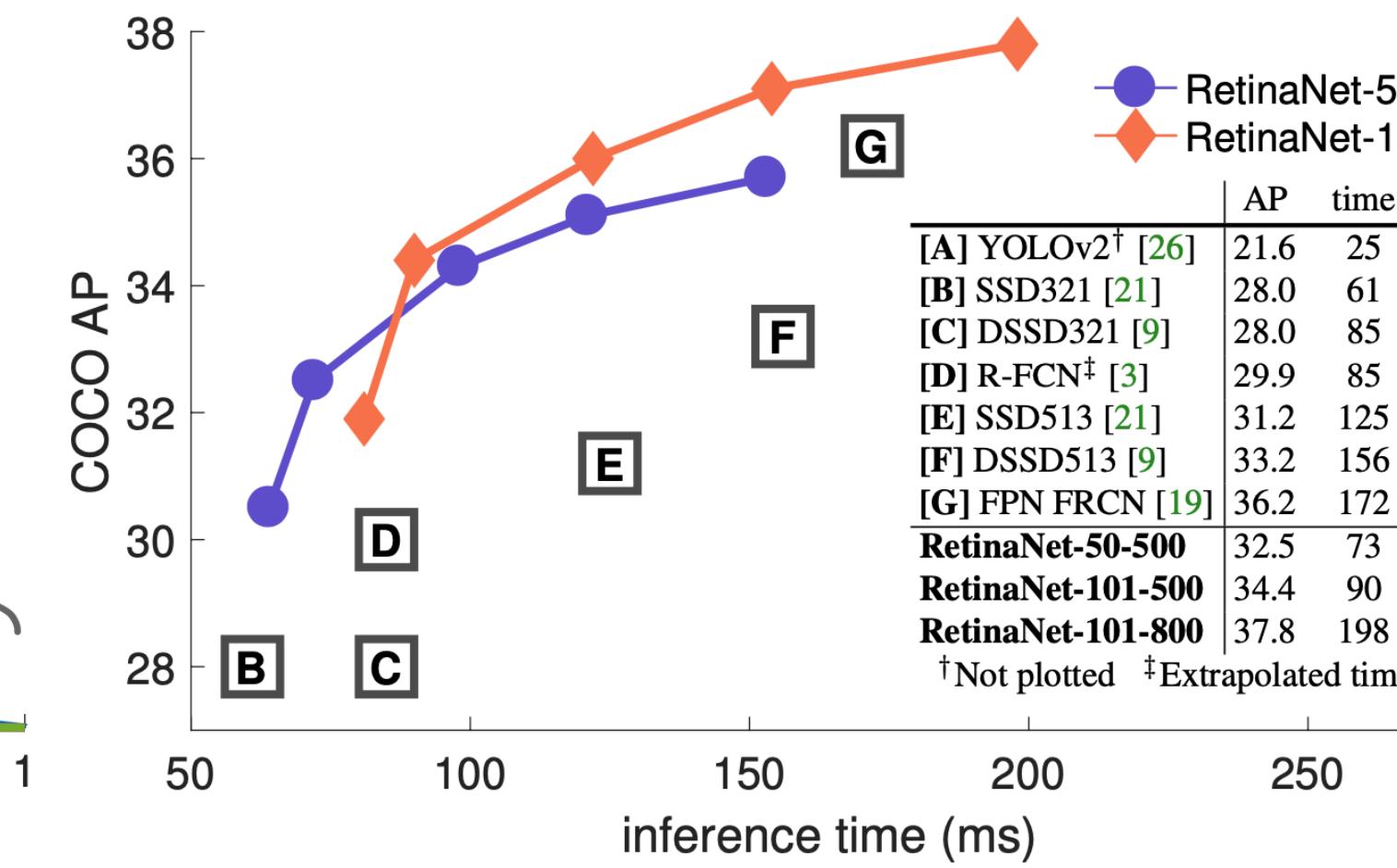
$$FL(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t) \rightarrow \alpha\text{-balanced focal loss}$$

Identify class imbalance during training as the main obstacle impeding one-stage detector from achieving state-of-the-art accuracy and propose a new loss function that eliminates this barrier.

RetinaNet



Lin, Tsung-Yi, et al. "Focal loss for dense object detection." Proceedings of the IEEE international conference on computer vision. 2017.



α	AP	AP ₅₀	AP ₇₅
.10	0.0	0.0	0.0
.25	10.8	16.0	11.7
.50	30.2	46.7	32.8
.75	31.1	49.4	33.0
.90	30.8	49.7	32.3
.99	28.7	47.4	29.9
.999	25.1	41.7	26.1

γ	α	AP	AP ₅₀	AP ₇₅
0	.75	31.1	49.4	33.0
0.1	.75	31.4	49.9	33.1
0.2	.75	31.9	50.7	33.4
0.5	.50	32.9	51.7	35.2
1.0	.25	33.7	52.0	36.2
2.0	.25	34.0	52.5	36.5
5.0	.25	32.2	49.6	34.8

(a) Varying α for CE loss ($\gamma = 0$)

(b) Varying γ for FL (w. optimal α)

depth	scale	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L	time
50	400	30.5	47.8	32.7	11.2	33.8	46.1	64
50	500	32.5	50.9	34.8	13.9	35.8	46.7	72
50	600	34.3	53.2	36.9	16.2	37.4	47.4	98
50	700	35.1	54.2	37.7	18.0	39.3	46.4	121
50	800	35.7	55.0	38.5	18.9	38.9	46.3	153
101	400	31.9	49.5	34.1	11.6	35.8	48.5	81
101	500	34.4	53.1	36.8	14.7	38.5	49.1	90
101	600	36.0	55.2	38.7	17.4	39.6	49.7	122
101	700	37.1	56.6	39.8	19.1	40.6	49.4	154
101	800	37.8	57.5	40.8	20.2	41.1	49.2	198

(e) Accuracy/speed trade-off RetinaNet (on test-dev)

	backbone	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
<i>Two-stage methods</i>							
Faster R-CNN+++ [15]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [19]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [16]	Inception-ResNet-v2 [33]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [31]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1
<i>One-stage methods</i>							
YOLOv2 [26]	DarkNet-19 [26]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [21, 9]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [9]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet (ours)	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2



Boulder

Speed/Accuracy Trade-Offs For Modern Convolutional Object Detectors

The right speed/memory/accuracy balance for a given application and platform!

$[x_0, y_0, x_1, y_1]$ are min/max coordinates of a box

Self-driving cars: real-time performance

Mobile devices: small memory footprint

Faster R-CNN v.s. R-FCN v.s. SSD

Single model/single pass

Convolutional detection meta-architectures

$a \rightarrow$ anchor

$b \rightarrow$ best matching ground truth box (if one exists)

corresponding to anchor a

$a \rightarrow$ positive anchor (if such a match exists)

$y_a \in \{1, 2, \dots, K\} \rightarrow$ label class assigned to a

$\phi(b_a; a) \rightarrow$ vector encoding of box b w.r.t. anchor a

↪ box encoding

$$\phi(b_a; a) = [10 \frac{x_c}{w_a}, 10 \frac{y_c}{h_a}, 5 \log w, 5 \log h]$$

$x_c, y_c \rightarrow$ center coordinates

$w, h \rightarrow$ width and height

$w_a, h_a \rightarrow$ width and height of anchor a

$a \rightarrow$ negative anchor (if no match is found)

$$y_a = 0$$

$f_{loc}(\mathcal{I}; a, \theta) \rightarrow$ predicted box encoding for anchor a

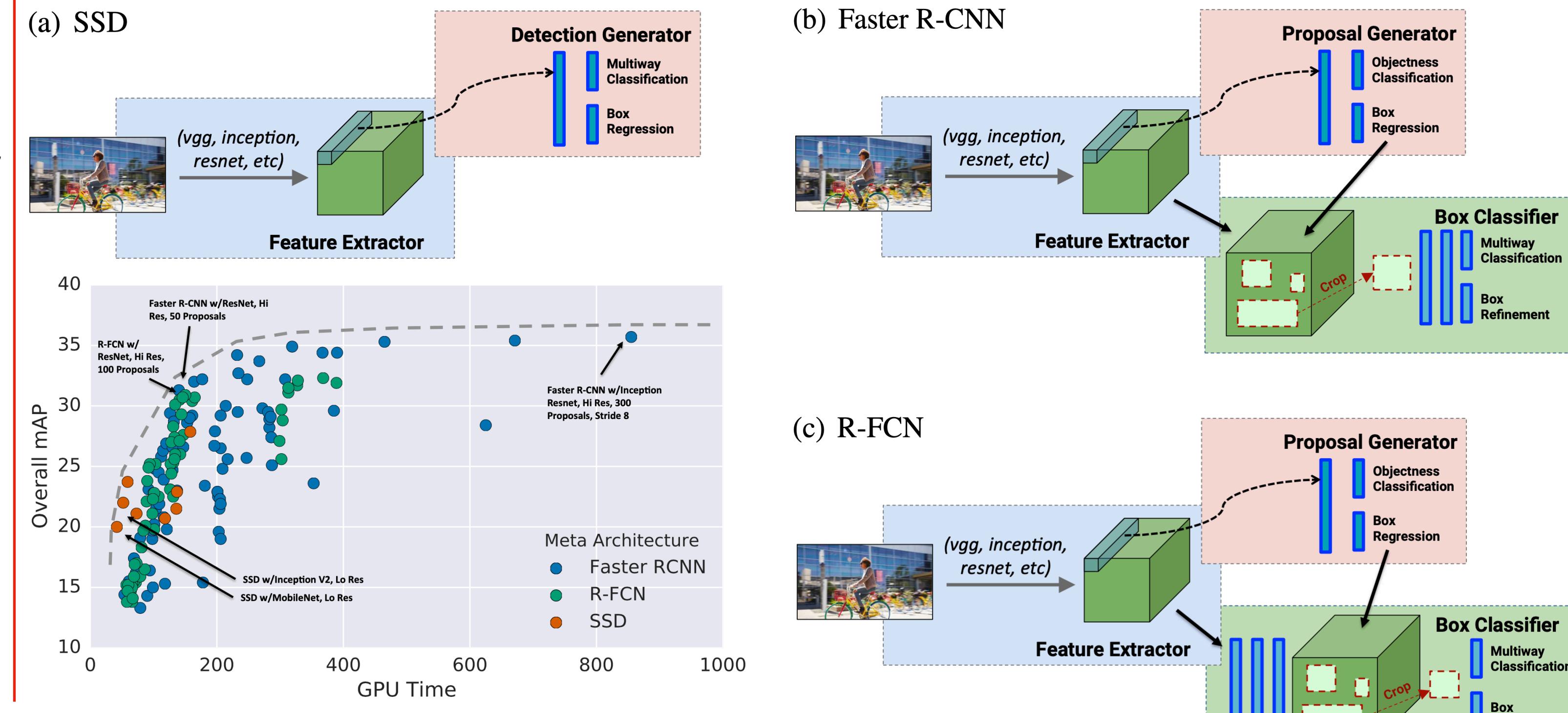
$f_{cls}(\mathcal{I}; a, \theta) \rightarrow$ predicted class for anchor a

$\mathcal{I} \rightarrow$ image

$\theta \rightarrow$ model parameters

$$\mathcal{L}(a, \mathcal{I}; \theta) = \alpha \cdot \mathbb{1}[a \text{ is positive}] \cdot \ell_{loc}(\phi(b_a; a) - f_{loc}(\mathcal{I}; a, \theta)) + \beta \cdot \ell_{cls}(y_a, f_{cls}(\mathcal{I}; a, \theta))$$

Paper	Meta-architecture	Feature Extractor	Matching	Box Encoding $\phi(b_a, a)$	Location Loss functions
Szegedy et al. [39]	SSD	InceptionV3	Bipartite	$[x_0, y_0, x_1, y_1]$	L_2
Redmon et al. [28]	SSD	Custom (GoogLeNet inspired)	Box Center	$[x_c, y_c, \sqrt{w}, \sqrt{h}]$	L_2
Ren et al. [30]	Faster R-CNN	VGG	Argmax	$[\frac{x_c}{w_a}, \frac{y_c}{h_a}, \log w, \log h]$	Smooth L_1
He et al. [13]	Faster R-CNN	ResNet-101	Argmax	$[\frac{x_c}{w_a}, \frac{y_c}{h_a}, \log w, \log h]$	Smooth L_1
Liu et al. [25] (v1)	SSD	InceptionV3	Argmax	$[x_0, y_0, x_1, y_1]$	L_2
Liu et al. [25] (v2, v3)	SSD	VGG	Argmax	$[\frac{x_c}{w_a}, \frac{y_c}{h_a}, \log w, \log h]$	Smooth L_1
Dai et al [6]	R-FCN	ResNet-101	Argmax	$[\frac{x_c}{w_a}, \frac{y_c}{h_a}, \log w, \log h]$	Smooth L_1



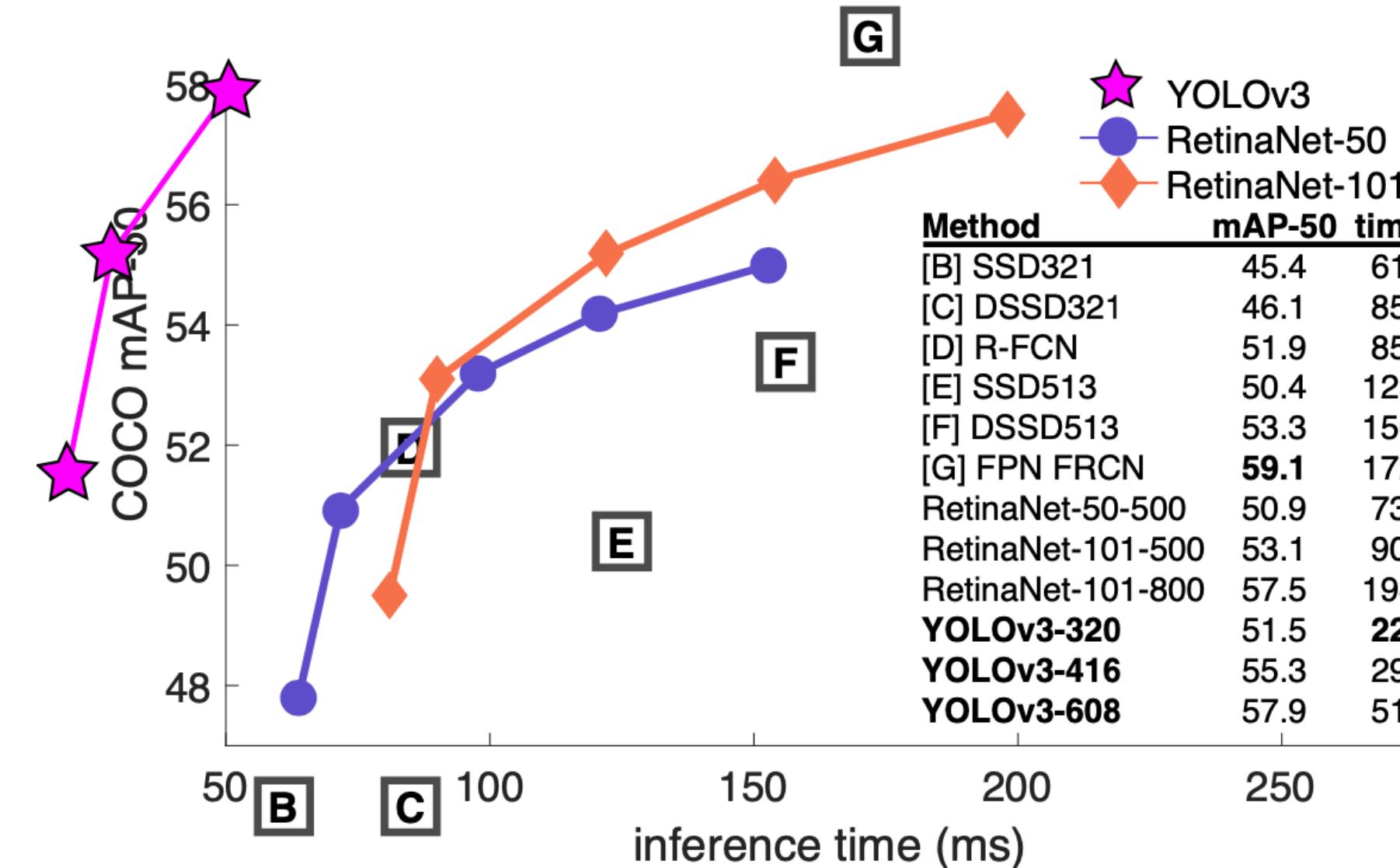
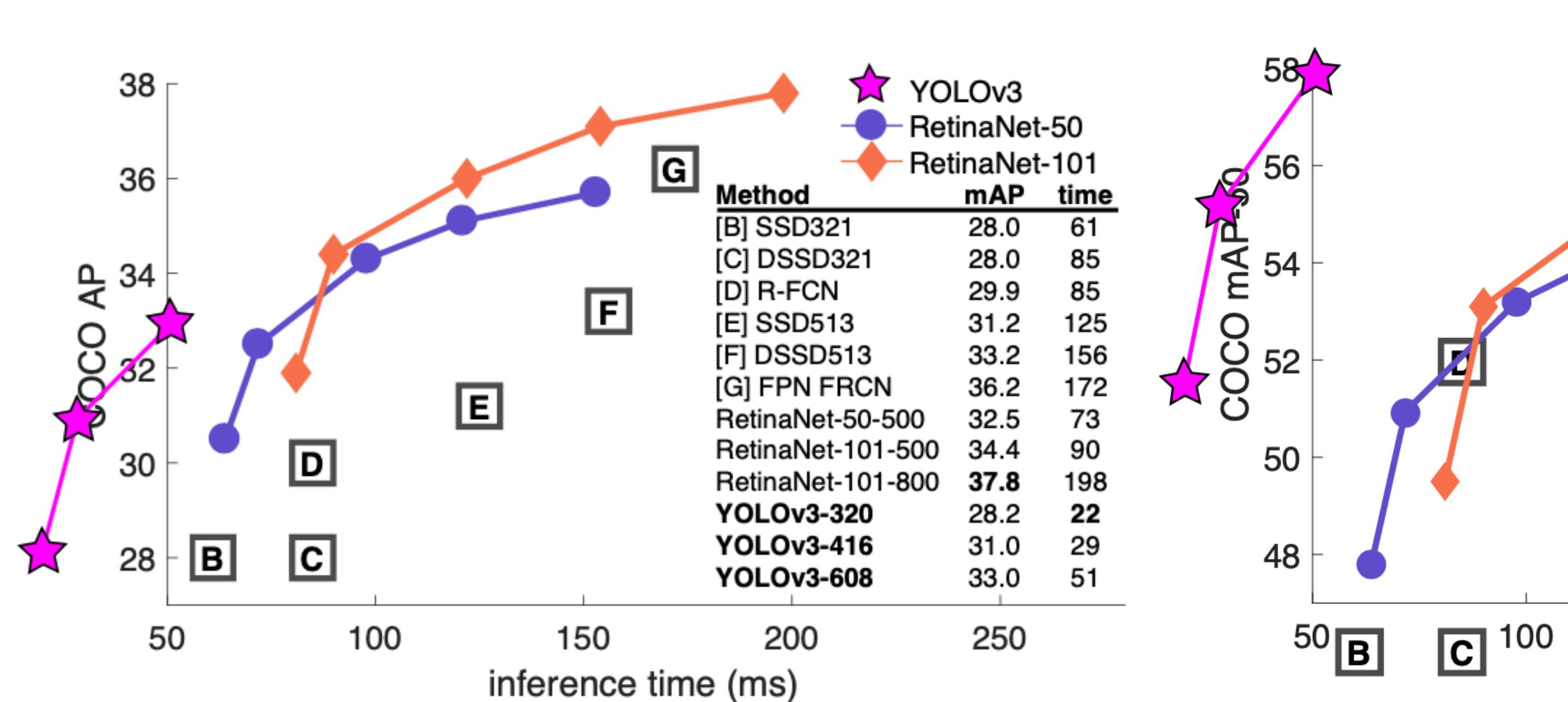


Boulder

YOLOv3: An Incremental Improvement



YouTube Video



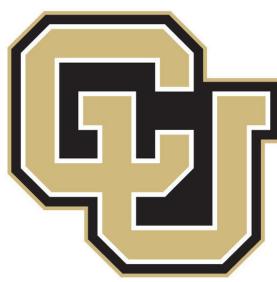
Darknet-53

Type	Filters	Size	Output
Convolutional	32	3 × 3	256 × 256
Convolutional	64	3 × 3 / 2	128 × 128
1x Convolutional	32	1 × 1	
1x Convolutional	64	3 × 3	128 × 128
Residual			
Convolutional	128	3 × 3 / 2	64 × 64
2x Convolutional	64	1 × 1	
2x Convolutional	128	3 × 3	64 × 64
Residual			
Convolutional	128	1 × 1	
8x Convolutional	256	3 × 3 / 2	32 × 32
8x Convolutional	128	1 × 1	
8x Convolutional	256	3 × 3	32 × 32
Residual			
Convolutional	512	3 × 3 / 2	16 × 16
Convolutional	256	1 × 1	
8x Convolutional	512	3 × 3	16 × 16
Residual			
Convolutional	1024	3 × 3 / 2	8 × 8
Convolutional	512	1 × 1	
4x Convolutional	1024	3 × 3	8 × 8
Residual			

	backbone	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
<i>Two-stage methods</i>							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1
<i>One-stage methods</i>							
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	40.8	61.1	44.1	24.1	44.2	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

In experiments with COCO, the model predict 3 boxes at each scale (3 different scales similar to feature pyramid networks) so the tensor is $N \times N \times [3 * (4 + 1 + 80)]$ for the 4 bounding box offsets, 1 objectness prediction, and 80 class predictions.

Class prediction: Does not use a softmax, instead simply uses independent logistic classifiers. During training it uses binary cross-entropy loss for the class predictions.



Boulder

EfficientDet: Scalable and Efficient Object Detection

Multi-Scale Feature Fusion

aggregate features at different resolutions

$$P^{\text{in}} = (P_{l_1}^{\text{in}}, P_{l_2}^{\text{in}}, \dots) \rightarrow \text{list of multi-scale features}$$

$P_{l_i}^{\text{in}}$ → feature at level l_i

$$P^{\text{out}} = f(P^{\text{in}}) \rightarrow \text{design } f$$

FPN (Feature Pyramid Network)

$$P^{\text{in}} = (P_3^{\text{in}}, P_4^{\text{in}}, \dots, P_7^{\text{in}})$$

P_i^{in} → feature level with resolution $\frac{1}{2^i}$ of the input image

$$P_7^{\text{out}} = \text{Conv}(P_7^{\text{in}})$$

$$P_6^{\text{out}} = \text{Conv}(P_6^{\text{in}} + \text{Resize}(P_7^{\text{out}}))$$

⋮

$$P_3^{\text{out}} = \text{Conv}(P_3^{\text{in}} + \text{Resize}(P_4^{\text{out}}))$$

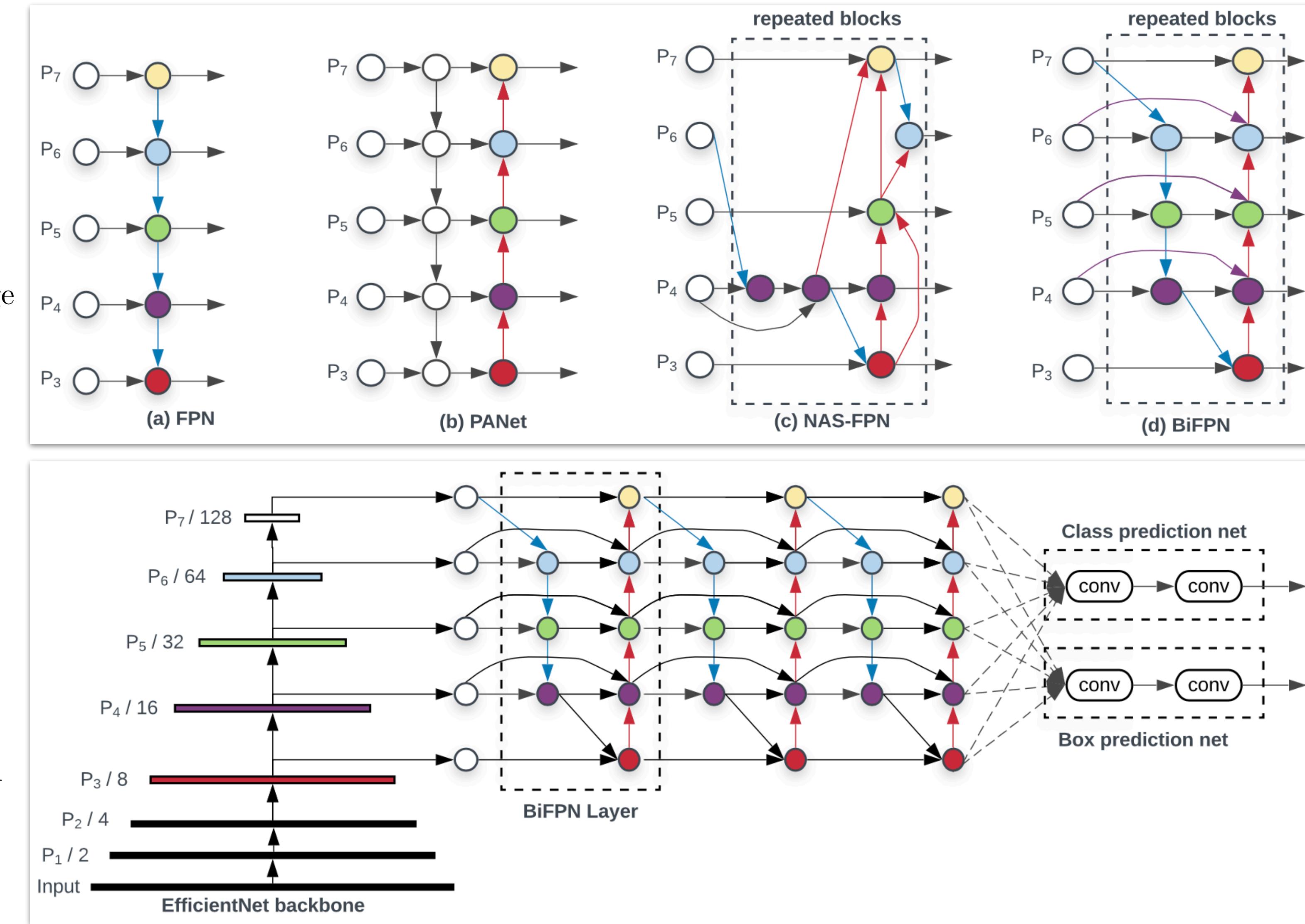
Weighted Bi-directional FPN (BiFPN)

$$P_6^{\text{td}} = \text{Conv}\left(\frac{w_1 P_6^{\text{in}} + w_2 \text{Resize}(P_7^{\text{out}})}{w_1 + w_2 + \varepsilon}\right)$$

$$P_6^{\text{out}} = \text{Conv}\left(\frac{w'_1 P_6^{\text{in}} + w'_2 P_6^{\text{td}} + w'_3 \text{Resize}(P_5^{\text{out}})}{w'_1 + w'_2 + w'_3 + \varepsilon}\right)$$

	Input size R_{input}	Backbone Network	BiFPN W_{bifpn}	Box/class D_{bifpn}	Box/class D_{class}
D0 ($\phi = 0$)	512	B0	64	3	3
D1 ($\phi = 1$)	640	B1	88	4	3
D2 ($\phi = 2$)	768	B2	112	5	3
D3 ($\phi = 3$)	896	B3	160	6	4
D4 ($\phi = 4$)	1024	B4	224	7	4
D5 ($\phi = 5$)	1280	B5	288	7	4
D6 ($\phi = 6$)	1280	B6	384	8	5
D6 ($\phi = 7$)	1536	B6	384	8	5

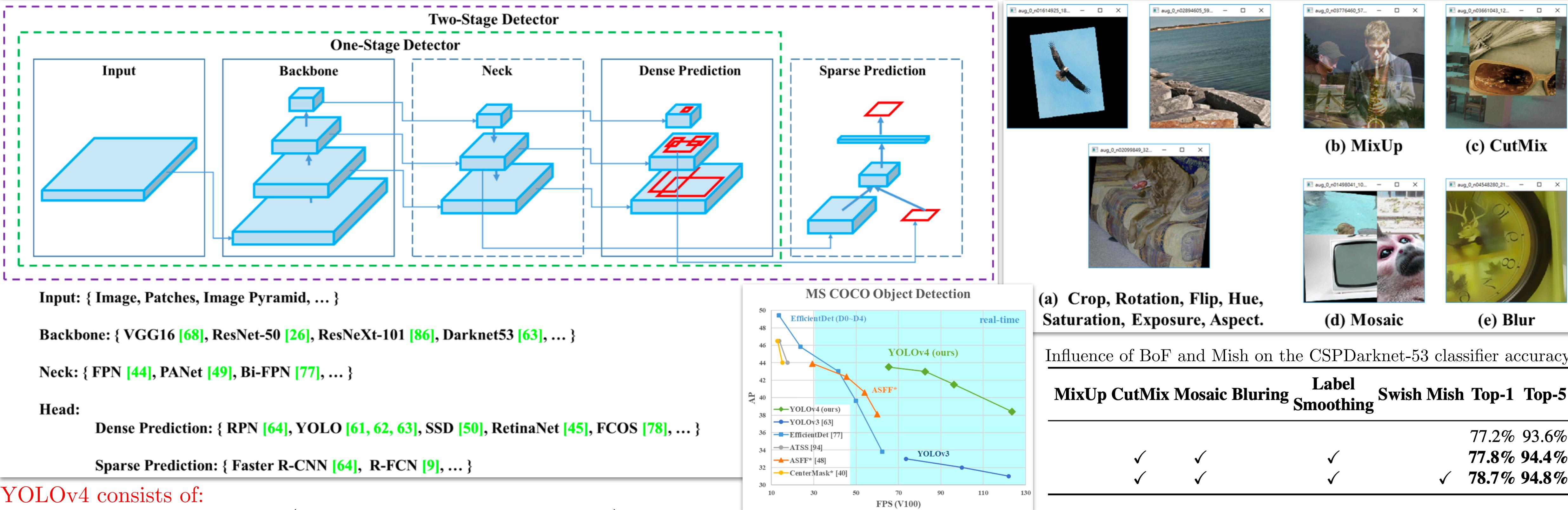
$\phi \rightarrow$ Compound Scaling Coefficient





Boulder

YOLOv4: Optimal Speed and Accuracy of Object Detection



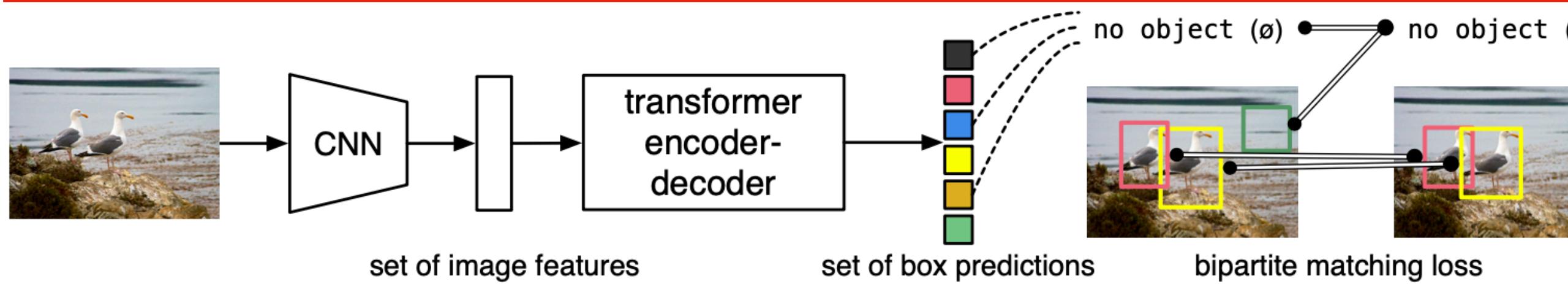
Influence of BoF and Mish on the CSPDarknet-53 classifier accuracy.

MixUp	CutMix	Mosaic	Bluring	Label Smoothing	Swish	Mish	Top-1	Top-5
✓	✓	✓	✓	✓	77.2%	93.6%	77.8%	94.4%
✓	✓	✓	✓	✓	78.7%	94.8%		



Boulder

End-to-End Object Detection with Transformers



Object detection set prediction loss

$N \rightarrow$ number of prediction

$y \rightarrow$ ground truth set of objects

$\hat{y} = \{\hat{y}_i\}_{i=1}^N \rightarrow$ set of N predictions

$N > \#$ of objects in the image

$y \rightarrow$ set of size N padded by \emptyset (no object)

$\sigma \in S_N \rightarrow$ permutation of N elements

$$\hat{\sigma} = \arg \min_{\sigma \in \mathcal{S}_N} \sum_{i=1}^N \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}) \rightarrow \text{Hungarian Algorithm}$$

(pairwise matching cost)

$$y_i = (c_i, b_i)$$

$c_i \rightarrow$ target class label (which may be \emptyset)

$b_i \in [0, 1]^4 \rightarrow$ ground-truth bonding box

center coordinates, height and width relative to the image size

$\hat{p}_{\sigma(i)}(c_i) \rightarrow$ probability of class c_i for the prediction with

$\hat{b}_{\sigma(i)}$ → predicted box for the prediction with index $\sigma(i)$

$$\mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}) = -\mathbb{1}_{\{c_i \neq \emptyset\}} \hat{p}_{\sigma(i)}(c_i) + \mathbb{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)})$$

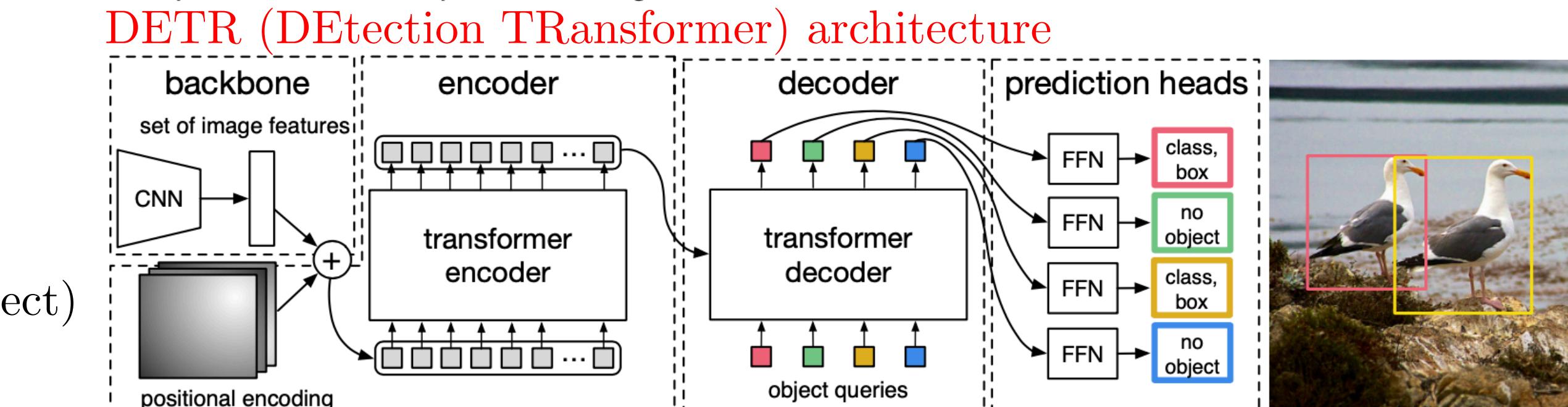
$$f_{\lambda_1}(h; \hat{h}_{-1}) = \lambda_1 f_{\lambda_1}(h; \hat{h}_{-1}) + \lambda_{-1} \|h - \hat{h}_{-1}\|$$

$$\mathcal{L}_{\text{iou}}(b_i, \hat{b}_{\sigma(i)}) = 1 - \left(\frac{|b_i \cap \hat{b}_{\sigma(i)}|}{|b_i \cup \hat{b}_{\sigma(i)}|} - \frac{|B(b_i, \hat{b}_{\sigma(i)}) \setminus b_i \cup \hat{b}_{\sigma(i)}|}{|B(b_i, \hat{b}_{\sigma(i)})|} \right)$$

largest box containing b_i & $\hat{b}_{\sigma(i)}$

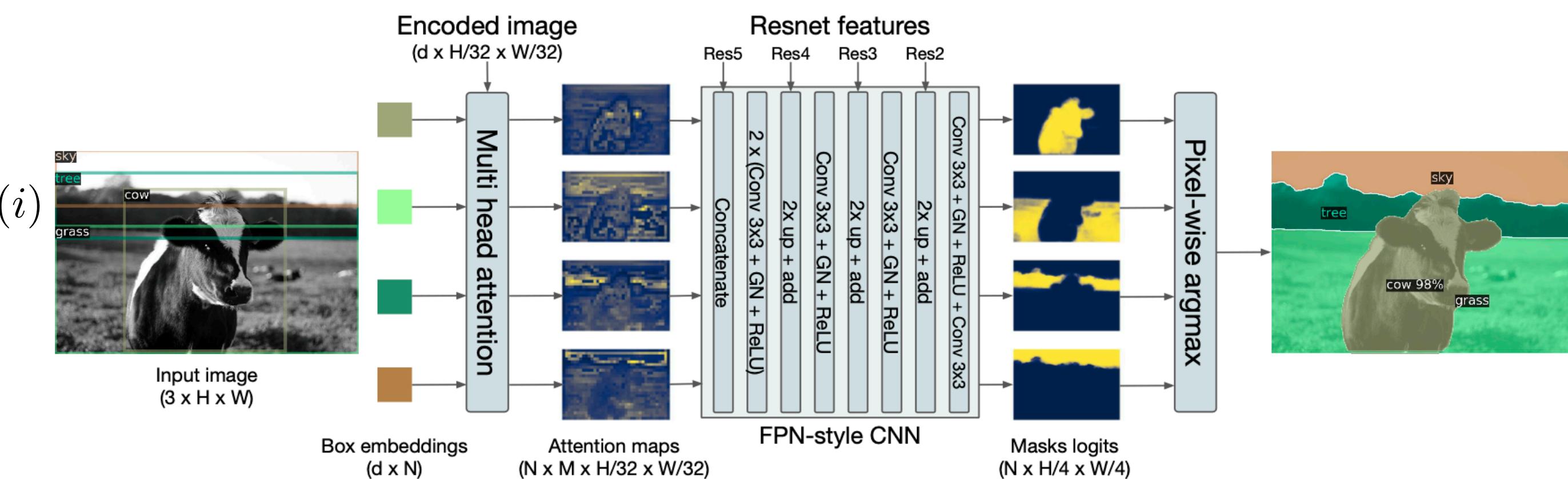
$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^N [-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}(i)})]$$

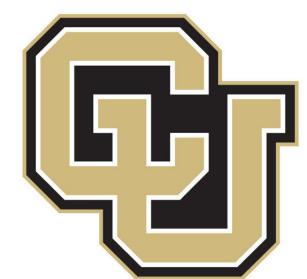
Down-weight the log-probability term when $c_i = \emptyset$
by a factor of 10 to account for class imbalances



Object Queries → Positional Encoding

DETR for panoptic segmentation





Boulder



Questions?

[YouTube Playlist](#)
