



Boulder

Computer Vision; Object Detection; Two Stage Detectors



[YouTube Playlist](#)

Maziar Raissi

Assistant Professor

Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu



Boulder

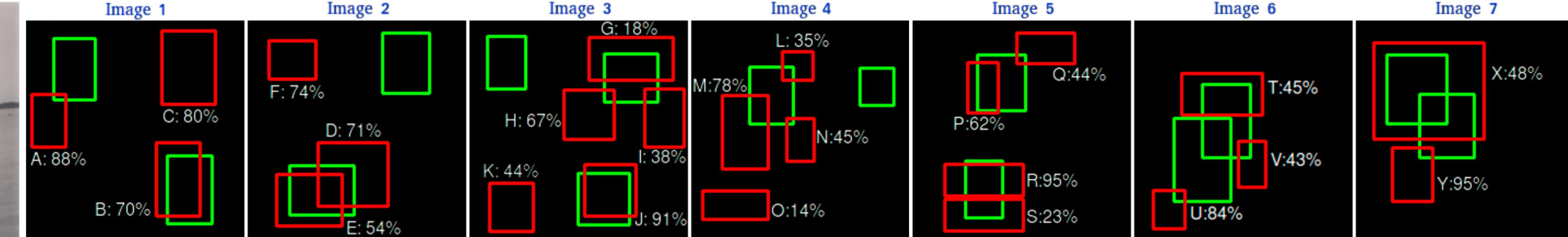


[YouTube Video](#)

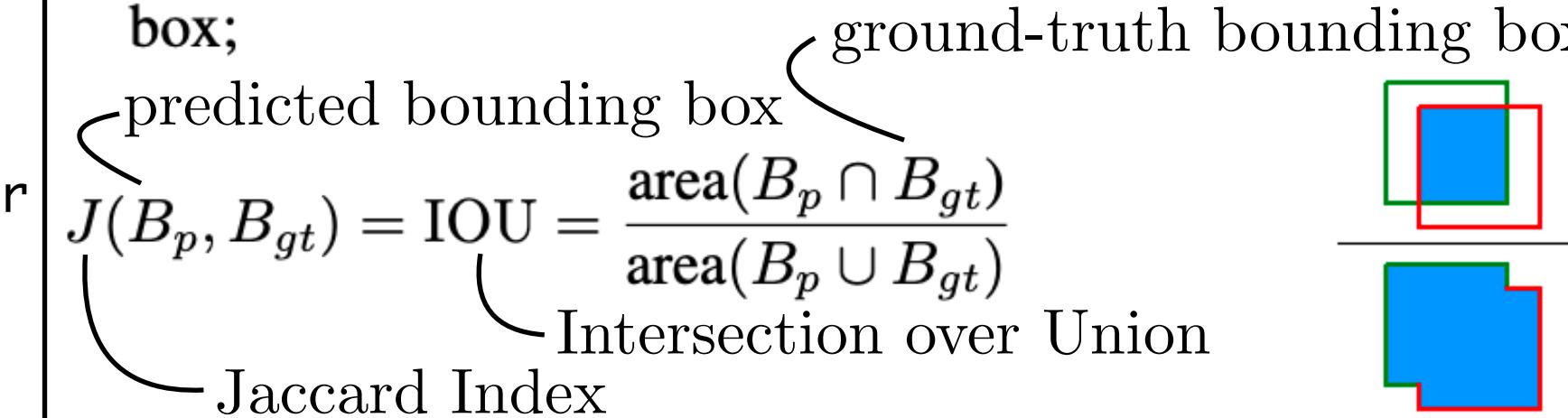
A Survey on Performance Metrics for Object-Detection Algorithms



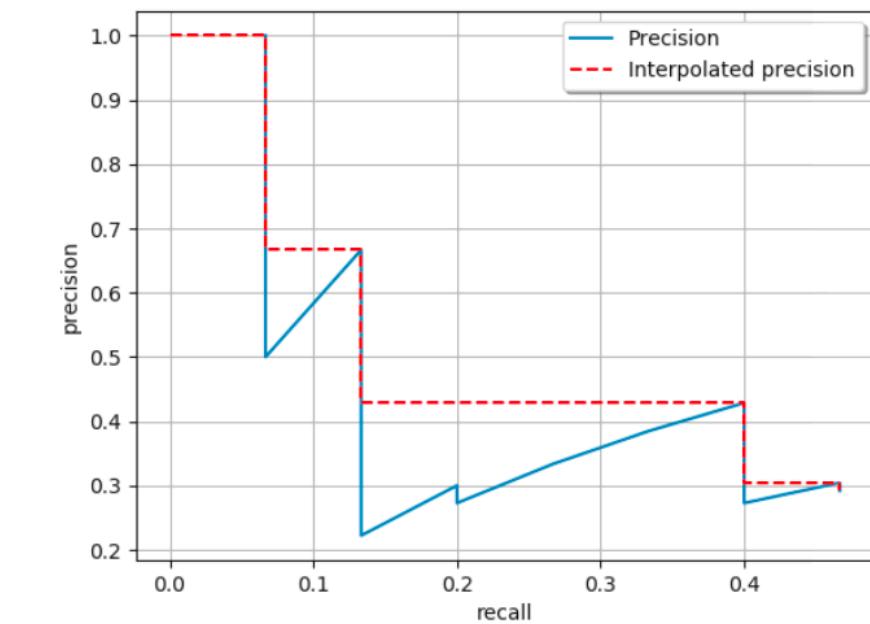
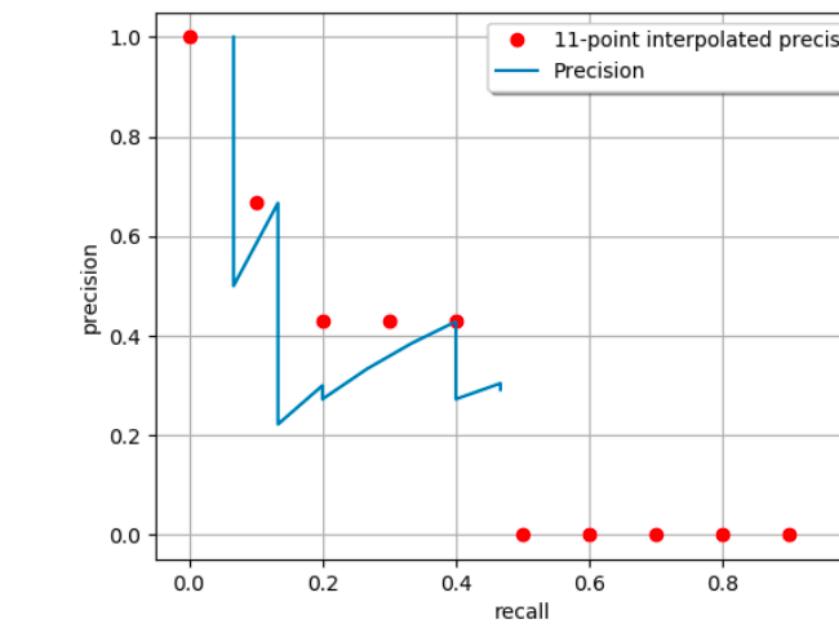
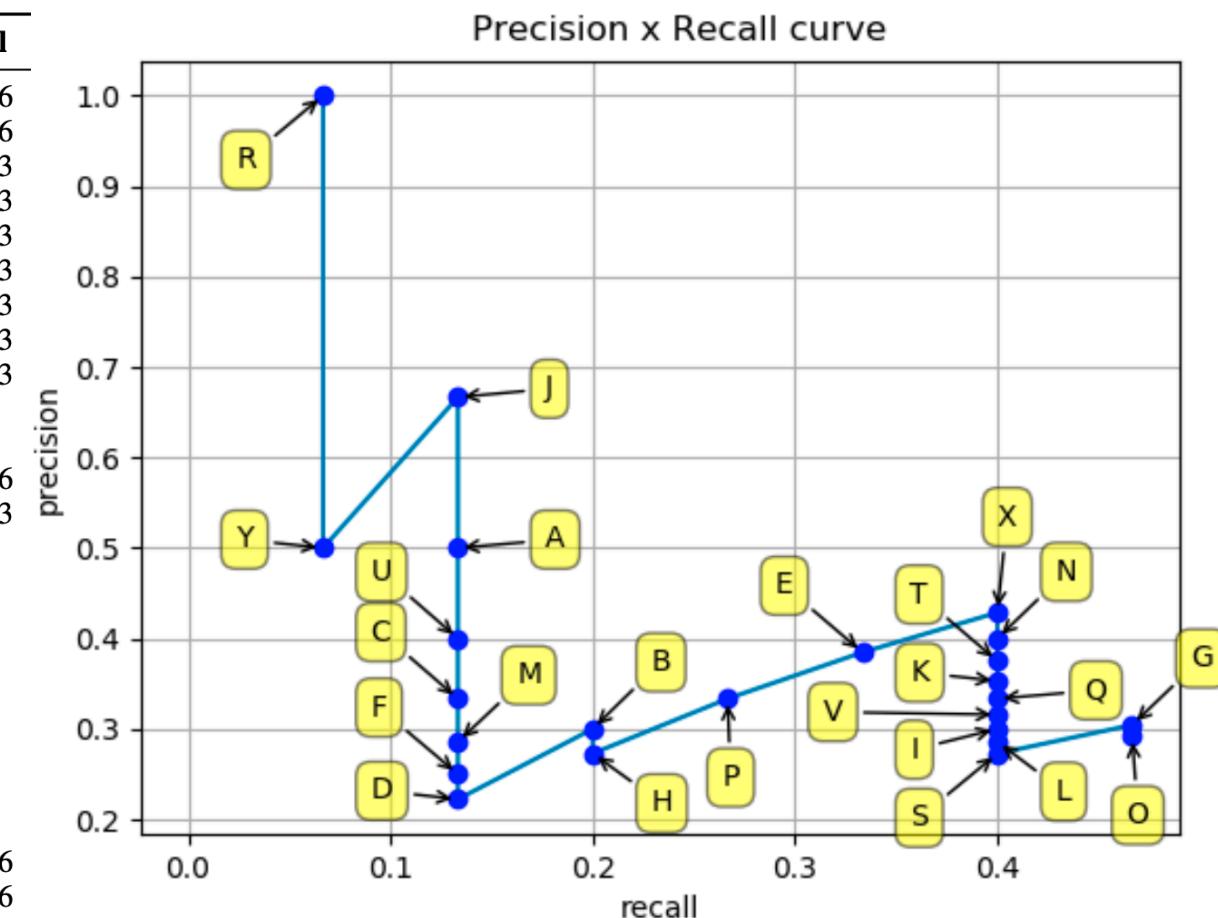
"In the object detection context, a true negative (TN) result does not apply, as there are infinite number of bounding boxes that should not be detected within any given image."



- True positive (TP): A correct detection of a ground-truth bounding box;
- False positive (FP): An incorrect detection of a nonexistent object or a misplaced detection of an existing object;
- False negative (FN): An undetected ground-truth bounding box;



detection	confidence	TP	FP	acc	TP	acc	FP	precision	recall
R	95%	1	0	1	0	1	0	0.0666	
Y	95%	0	1	1	1	0.5	0.0666		
J	91%	1	0	2	1	0.6666	0.1333		
A	88%	0	1	2	2	0.5	0.1333		
U	84%	0	1	2	3	0.4	0.1333		
C	80%	0	1	2	4	0.3333	0.1333		
M	78%	0	1	2	5	0.2857	0.1333		
F	74%	0	1	2	6	0.25	0.1333		
D	71%	0	1	2	7	0.2222	0.1333		
B	70%	1	0	3	7	0.3	0.2		
H	67%	0	1	3	8	0.2727	0.2		
P	62%	1	0	4	8	0.3333	0.2666		
E	54%	1	0	5	8	0.3846	0.3333		
X	48%	1	0	6	8	0.4285	0.4		
N	45%	0	1	6	9	0.4	0.4		
T	45%	0	1	6	10	0.375	0.4		
K	44%	0	1	6	11	0.3529	0.4		
Q	44%	0	1	6	12	0.3333	0.4		
V	43%	0	1	6	13	0.3157	0.4		
I	38%	0	1	6	14	0.3	0.4		
L	35%	0	1	6	15	0.2857	0.4		
S	23%	0	1	6	16	0.2727	0.4		
G	18%	1	0	7	16	0.3043	0.4666		
O	14%	0	1	7	17	0.2916	0.4666		



$t \rightarrow$ threshold
 $\text{IOU} > t \rightarrow$ correct
 $\text{IOU} < t \rightarrow$ incorrect

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\text{all detections}},$$

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{all ground truths}}$$

$$\text{AP}_{11} = \frac{1}{11} \sum_{R \in \{0, 0.1, \dots, 0.9, 1\}} P_{\text{interp}}(R)$$

$$P_{\text{interp}}(R) = \max_{\tilde{R}: \tilde{R} \geq R} P(\tilde{R})$$

$$\text{AP}_{\text{all}} = \sum_n (R_{n+1} - R_n) P_{\text{interp}}(R_{n+1})$$

$$P_{\text{interp}}(R_{n+1}) = \max_{\tilde{R}: \tilde{R} \geq R_{n+1}} P(\tilde{R})$$

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^N \text{AP}_i$$

$\text{AP}_i \rightarrow i\text{-th class}$



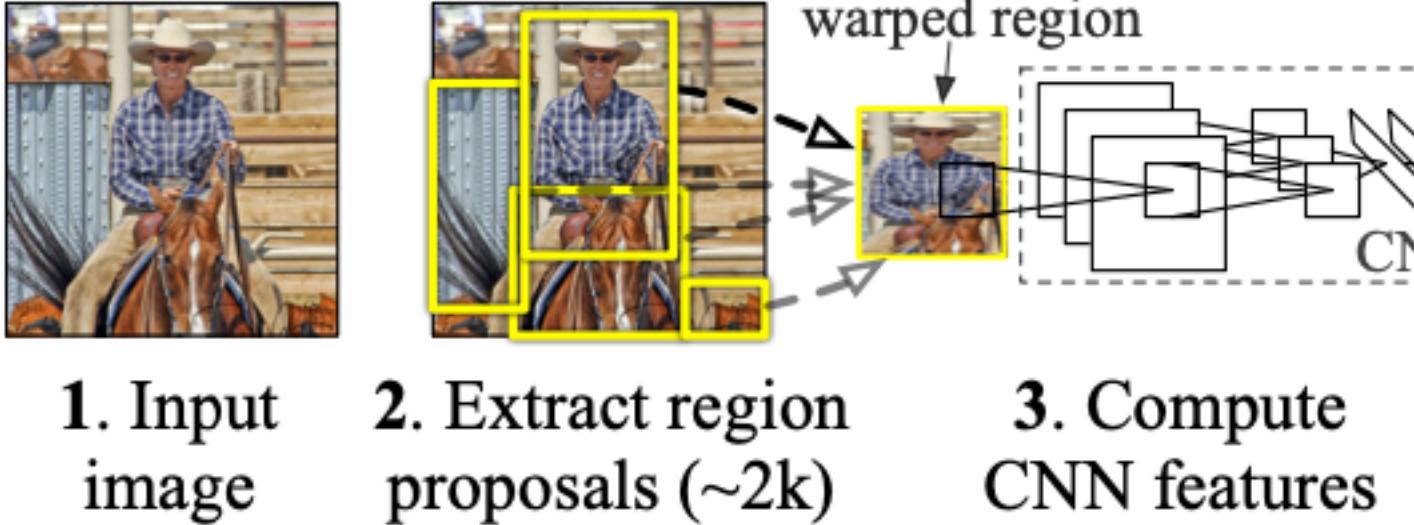
Boulder

Rich feature hierarchies for accurate object detection and semantic segmentation



[YouTube Video](#)

R-CNN: Regions with CNN features



Region proposals: selective search

Feature extraction: 4096-dimensional feature vector from AlexNet



Classify regions: class-specific linear SVM

Greedy Non-maximum Suppression

For each class independently reject a region if it has an intersection-over-union (IoU) overlap with a higher scoring selected region larger than a learned threshold.

Domain-specific fine-tuning

21-way classification layer

Treat all region proposals with ≥ 0.5 IoU overlap with a ground-truth box as positives for that box's class and the rest as negatives.

Object category classifiers

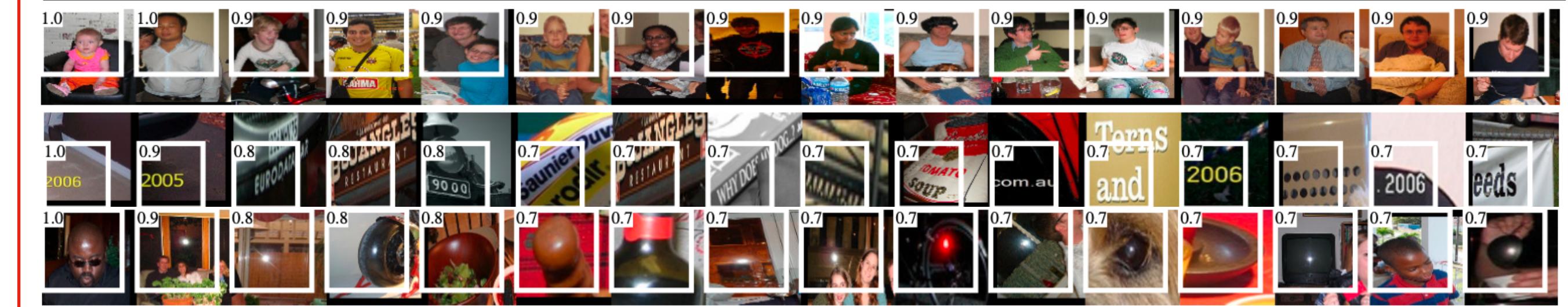
$\text{IoU} < 0.3 \rightarrow$ negative examples

ground-truth bounding boxes \rightarrow positive examples

VOC 2010 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
DPM v5 [17] [†]	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	33.4
UVA [32]	56.2	42.4	15.3	12.6	21.8	49.3	36.8	46.1	12.9	32.1	30.0	36.5	43.5	52.9	32.9	15.3	41.1	31.8	47.0	44.8	35.1
Regionlets [35]	65.0	48.9	25.9	24.6	24.5	56.1	54.5	51.2	17.0	28.9	30.2	35.8	40.2	55.7	43.5	14.3	43.9	32.6	54.0	45.9	39.7
SegDPM [15] [†]	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	40.4
R-CNN	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	50.2
R-CNN BB	71.8	65.8	53.0	36.8	35.9	59.7	60.0	69.9	27.9	50.6	41.4	70.0	62.0	69.0	58.1	29.5	59.4	39.3	61.2	52.4	53.7

Visualizing learned features

The idea is to single out a particular unit (feature) in the network and use it as if it were an object detector in its own right.



Bounding-box regression

$G = (G_x, G_y, G_w, G_h) \rightarrow$ ground-truth bounding box

$P = (\underbrace{P_x, P_y}_{\text{center}}, \underbrace{P_w, P_h}_{\text{width \& height}}) \rightarrow$ proposal bounding box

$$\hat{G}_x = P_w d_x(P) + P_x$$

$$\hat{G}_y = P_h d_y(P) + P_y$$

$$\hat{G}_w = P_w \exp(d_w(P))$$

$$\hat{G}_h = P_h \exp(d_h(P)).$$

$$t_x = (G_x - P_x)/P_w$$

$$t_y = (G_y - P_y)/P_h$$

$$t_w = \log(G_w/P_w)$$

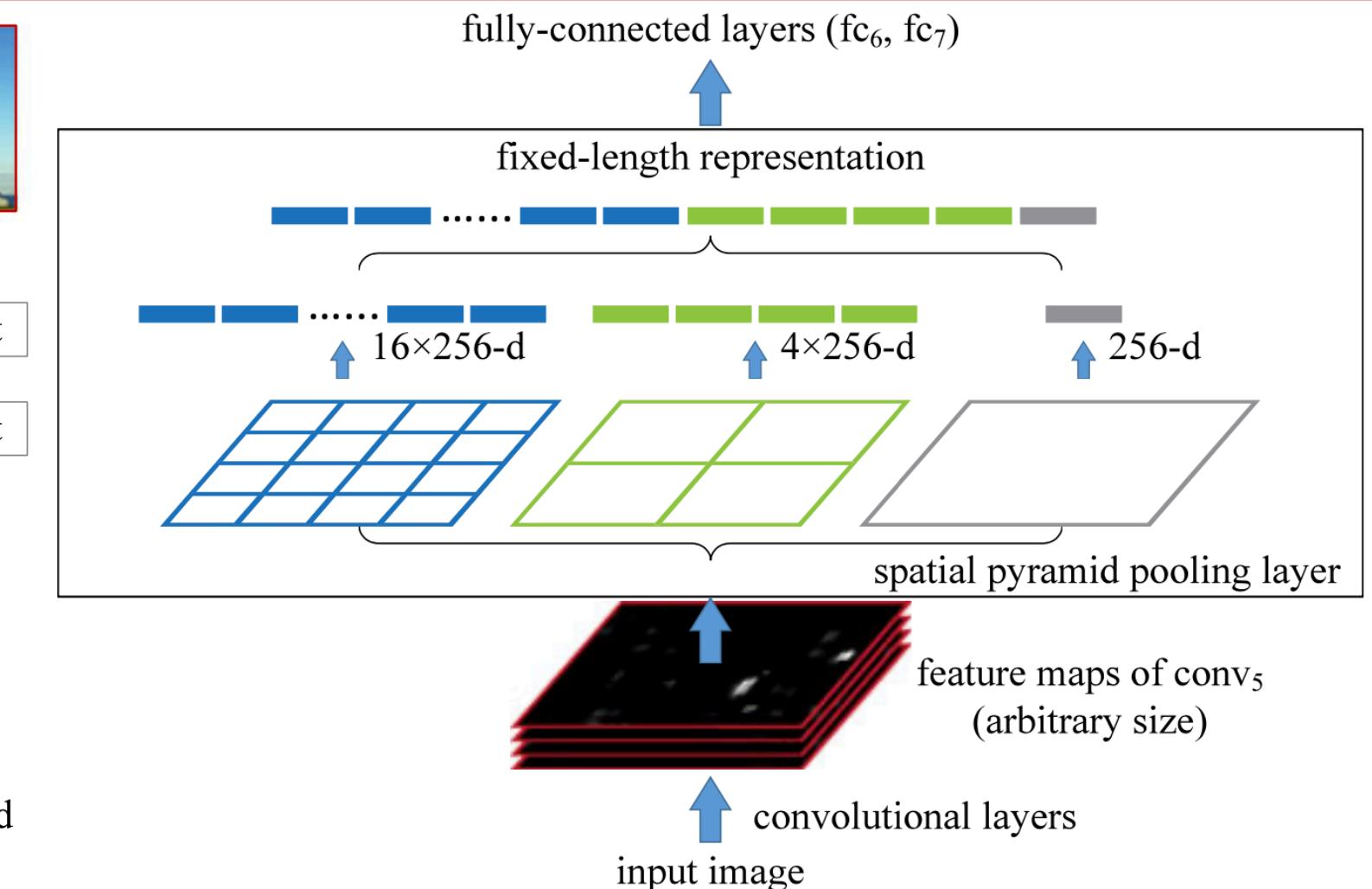
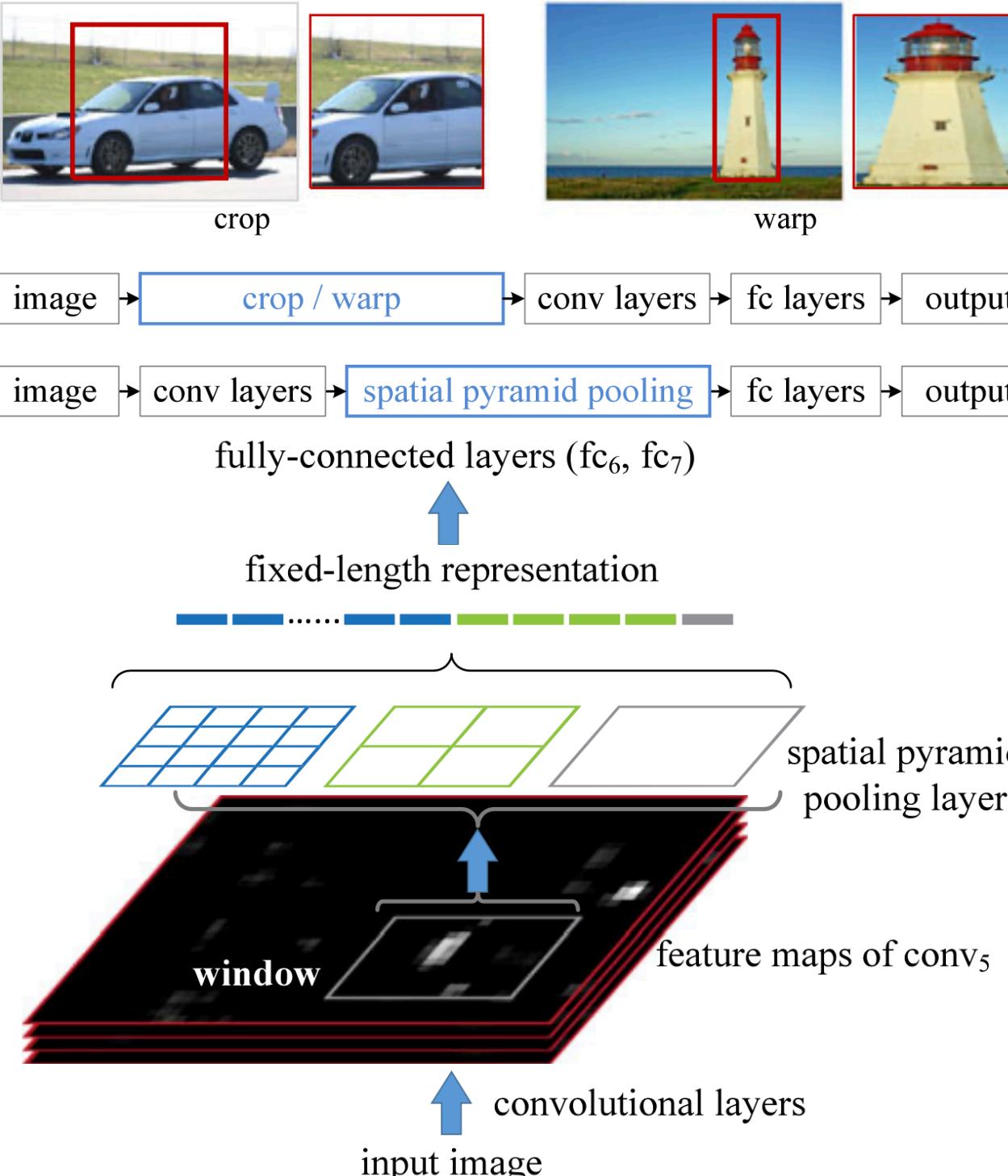
$$t_h = \log(G_h/P_h).$$

$$\mathbf{w}_* = \operatorname{argmin}_{\hat{\mathbf{w}}_*} \sum_i^N (t_*^i - \hat{\mathbf{w}}_*^T \phi_5(P^i))^2 + \lambda \|\hat{\mathbf{w}}_*\|^2$$

$$d_*(P) = \mathbf{w}_*^T \phi_5(P)$$

pool₅ features

Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition


[YouTube Video](#)


$a \times a \rightarrow$ feature map size (e.g., 13×13)

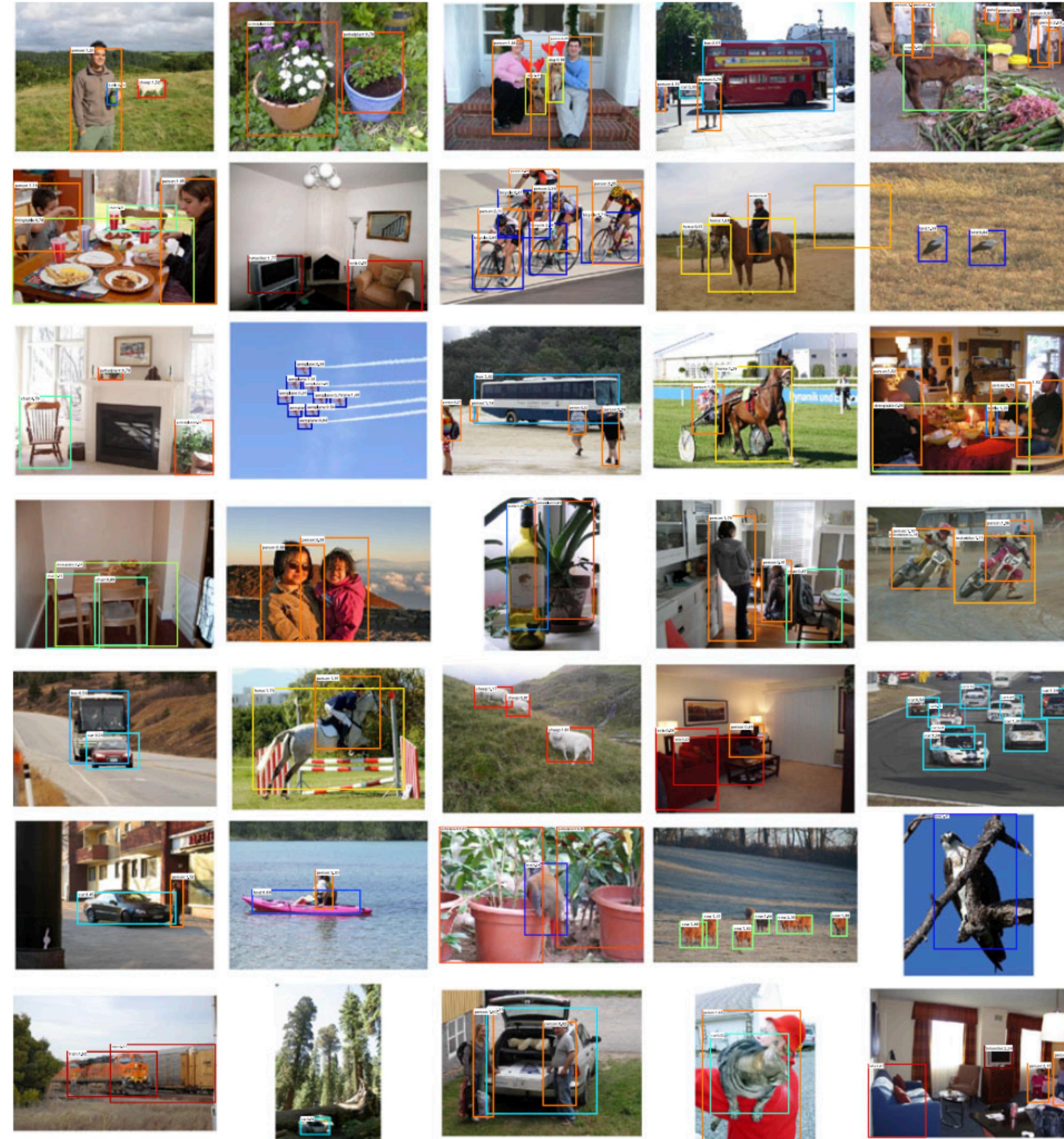
$n \times n \rightarrow$ pyramid level bins (e.g., 2×2)

win = $\lceil a/n \rceil \rightarrow$ window size

str = $\lfloor a/n \rfloor \rightarrow$ stride

Comparisons of Detection Results on Pascal VOC 2007

method	mAP	area	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
DPM [23]	33.7	33.2	60.3	10.2	16.1	27.3	54.3	58.2	23.0	20.0	24.1	26.7	12.7	58.1	48.2	43.2	12.0	21.1	36.1	46.0	43.5
SS [20]	33.8	43.5	46.5	10.4	12.0	9.3	49.4	53.7	39.4	12.5	36.9	42.2	26.4	47.0	52.4	23.5	12.1	29.9	36.3	42.2	48.8
Regionlet [39]	41.7	54.2	52.0	20.3	24.0	20.1	55.5	68.7	42.6	19.2	44.2	49.1	26.6	57.0	54.5	43.4	16.4	36.6	37.7	59.4	52.3
DetNet [40]	30.5	29.2	35.2	19.4	16.7	3.7	53.2	50.2	27.2	10.2	34.8	30.2	28.2	46.6	41.7	26.2	10.3	32.8	26.8	39.8	47.0
RCNN ftfc ₇ (A5)	54.2	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7
RCNN ftfc ₇ (ZF5)	55.1	64.8	68.4	47.0	39.5	30.9	59.8	70.5	65.3	33.5	62.5	50.3	59.5	61.6	67.9	54.1	33.4	57.3	52.9	60.2	62.9
SPP ftfc ₇ (ZF5)	55.2	65.5	65.9	51.7	38.4	32.7	62.6	68.6	69.7	33.1	66.6	53.1	58.2	63.6	68.8	50.4	27.4	53.7	48.2	61.7	64.7
RCNN bb (A5)	58.5	68.1	72.8	56.8	43.0	36.8	66.3	74.2	67.6	34.4	63.5	54.5	61.2	69.1	68.6	58.7	33.4	62.9	51.1	62.5	64.8
RCNN bb (ZF5)	59.2	68.4	74.0	54.0	40.9	35.2	64.1	74.4	69.8	35.5	66.9	53.8	64.2	69.9	69.6	58.9	36.8	63.4	56.0	62.8	64.9
SPP bb (ZF5)	59.2	68.6	69.7	57.1	41.2	40.5	66.3	71.3	72.5	34.4	67.3	61.7	63.1	71.0	69.8	57.6	29.7	59.0	50.2	65.2	68.0



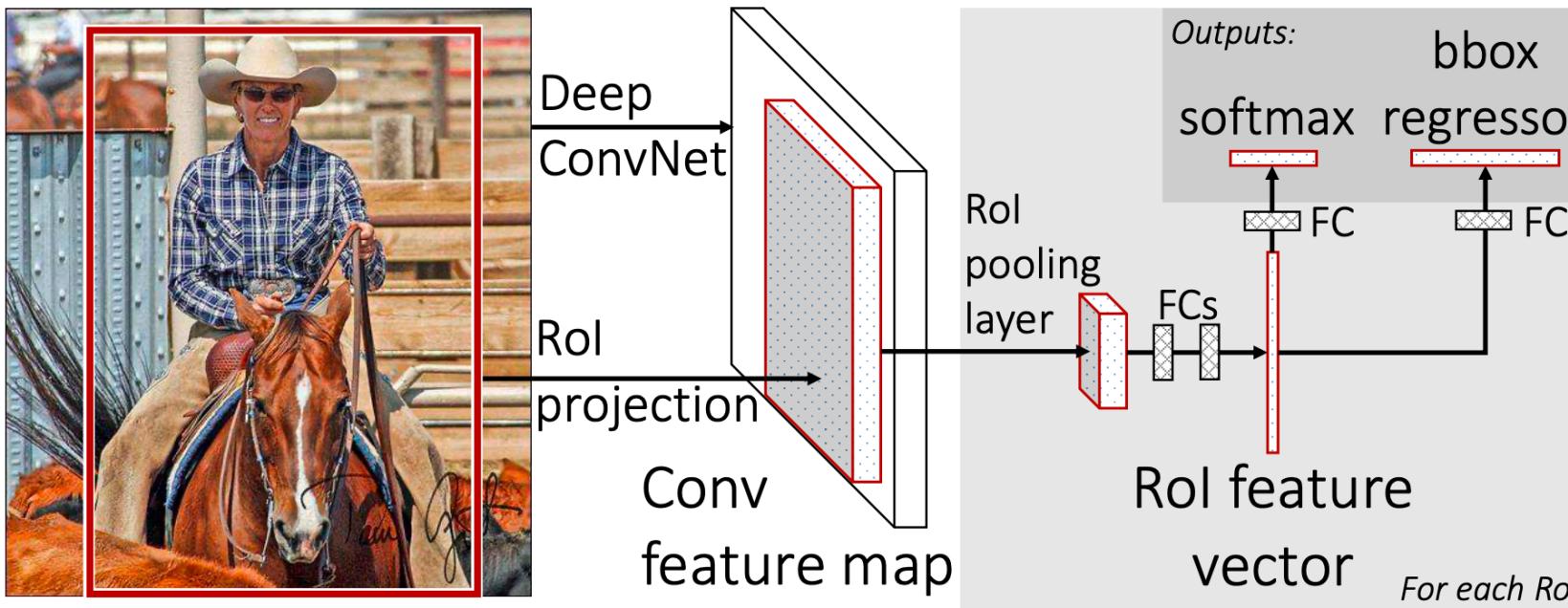


Boulder



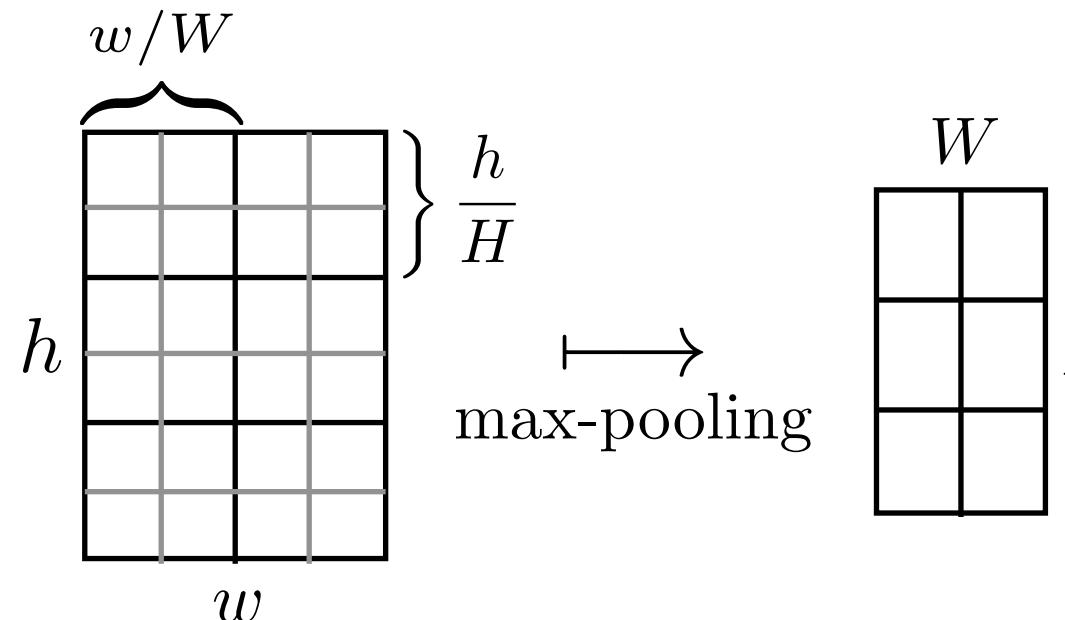
[YouTube Video](#)

Fast R-CNN



ROI pooling layer

height & width
 $(r, c, h, w) \rightarrow \text{RoI} (\text{Regions of Interest})$



H & W are set to be compatible with the first fully connected layer

Fine-tuning

Mini-batches are sampled **hierarchically**, first by sampling N images and then by sampling R/N RoIs from each image ($N=2$, $R = 128$).

Multi-task loss

$p = (p_0, \dots, p_K) \rightarrow$ probability distribution over $K + 1$ categories

$t^k = (t_x^k, t_y^k, t_w^k, t_h^k) \rightarrow$ bounding box regression offsets

$k = 1, \dots, K \rightarrow$ object classes

category-specific bounding-box regressors

Each training RoI is labeled with:

$u \rightarrow$ ground-truth class

$v \rightarrow$ ground-truth bounding box regression target

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda \underbrace{[u \geq 1]}_{= 1 \text{ iff } u \geq 1} L_{\text{loc}}(t^u, v)$$

$$L_{\text{cls}}(p, u) = -\log p_u$$

$$L_{\text{loc}}(t^u, v) = \sum_{i \in \{\text{x,y,w,h}\}} \text{smooth}_{L_1}(t_i^u - v_i)$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

robust L_1 loss that is less sensitive to outliers

Back-propagation through ROI pooling layers

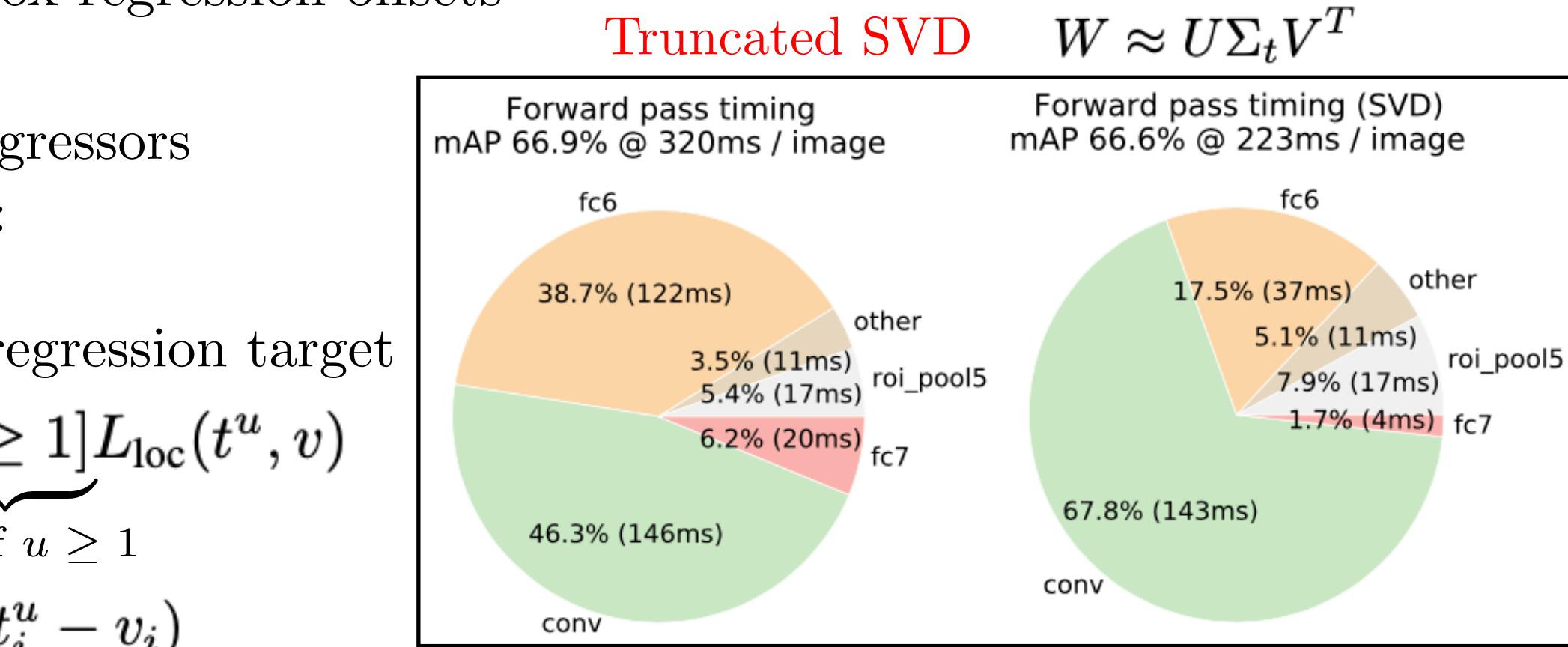
$x_i \in \mathbb{R} \rightarrow$ i -th activation input into the ROI pooling layer

$y_{rj} \in \mathbb{R} \rightarrow$ layer's j -th output from the r -th ROI

$\mathcal{R}(r, j) \rightarrow$ sub-window over which the output unit y_{rj} max-pools

$$i^*(r, j) = \arg \max_{i' \in \mathcal{R}(r, j)} x_{i'}$$

$$y_{rj} = x_{i^*(r, j)}$$



	Fast R-CNN			R-CNN			SPPnet
	S	M	L	S	M	L	$\dagger L$
train time (h)	1.2	2.0	9.5	22	28	84	25
train speedup	18.3x	14.0x	8.8x	1x	1x	1x	3.4x
test rate (s/im)	0.10	0.15	0.32	9.8	12.1	47.0	2.3
\triangleright with SVD	0.06	0.08	0.22	-	-	-	-
test speedup	98x	80x	146x	1x	1x	1x	20x
\triangleright with SVD	169x	150x	213x	-	-	-	-
VOC07 mAP	57.1	59.2	66.9	58.5	60.2	66.0	63.1
\triangleright with SVD	56.5	58.7	66.6	-	-	-	-

A single x_i may be assigned to several different outputs y_{rj} .

$$\frac{\partial L}{\partial x_i} = \sum_r \sum_j [i = i^*(r, j)] \frac{\partial L}{\partial y_{rj}}$$



Boulder

Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

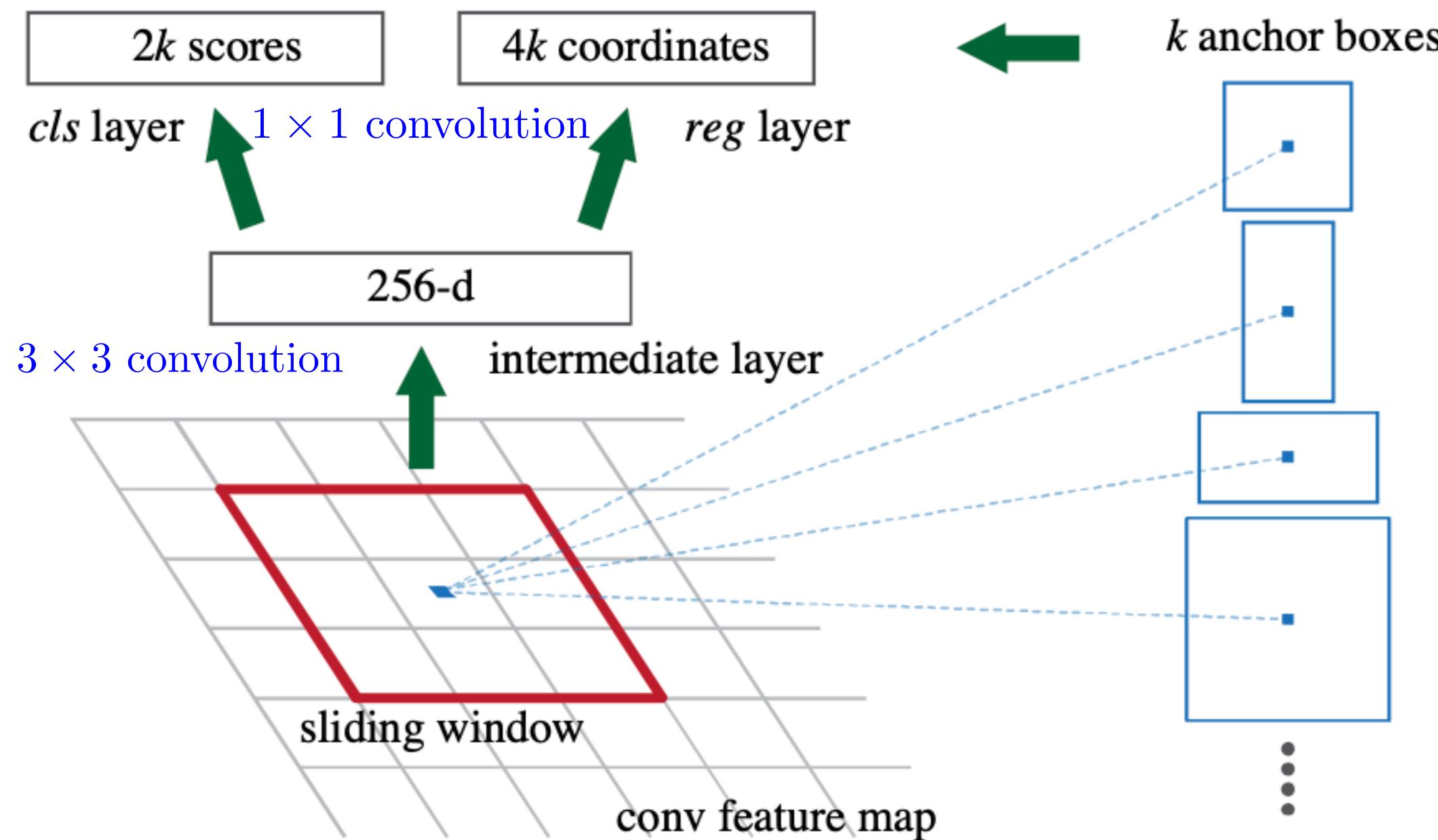


[YouTube Video](#)

Region Proposal Network (RPN)

An RPN is a fully-convolutional network that simultaneously predicts object bounds and objectness scores at each position.

RPN and Fast R-CNN can be trained using a simple alternating optimization to share convolutional features.



Translation-Invariant Anchors

9 anchors: 3 scales & 3 aspect ratios

$W H k \rightarrow$ total anchors for a feature map of size $H \times W$

Loss Function (Region Proposals)

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

$i \rightarrow$ index of an anchor in a minibatch

$p_i \rightarrow$ predicted probability of being an object

$p_i^* = 1 \rightarrow$ if anchor is positive

$p_i^* = 0 \rightarrow$ if anchor is negative

For anchors, use 3 scales with box areas of 128×128 , 256×256 , and 512×512 pixels, and 3 aspect ratios of 1:1, 1:2, and 2:1.

Positive label is assigned to two kinds of anchors: (i) the anchor/anchors with the highest Intersection-over-Union (IoU) overlap with a ground-truth box, or (ii) an anchor that has an IoU overlap higher than 0.7 with any ground-truth box. Negative label is assigned to a non-positive anchor if its IoU ratio is lower than 0.3 for all ground-truth boxes.

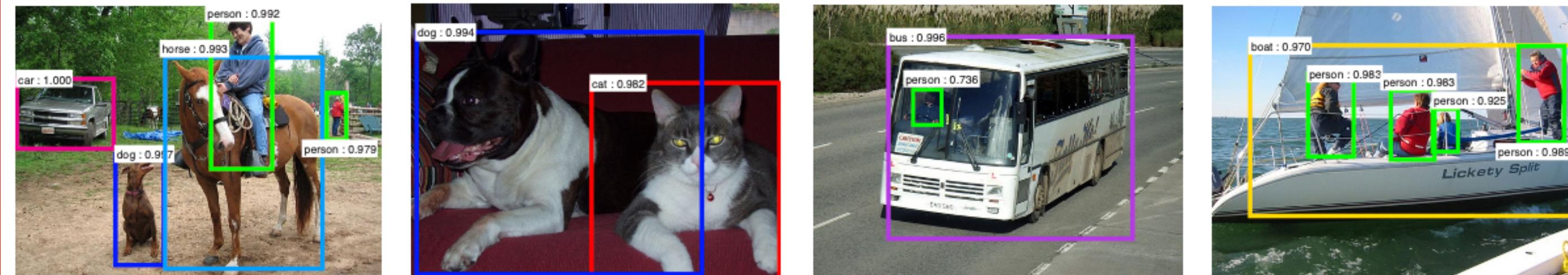
Alternating Training

First step: Train the RPN initialized with an ImageNet-pre-trained model and fine-tuned end-to-end for the region proposal task.

Second step: Train a separate detection network by Fast R-CNN using the proposals generated by the step-1 RPN. This detection network is also initialized by the ImageNet-pre-trained model.

Third step: Use the detector network to initialize RPN training, but we fix the shared conv layers and only fine-tune the layers unique to RPN.

Finally: Keeping the shared conv layers fixed, fine-tune the fc layers of the Fast R-CNN.





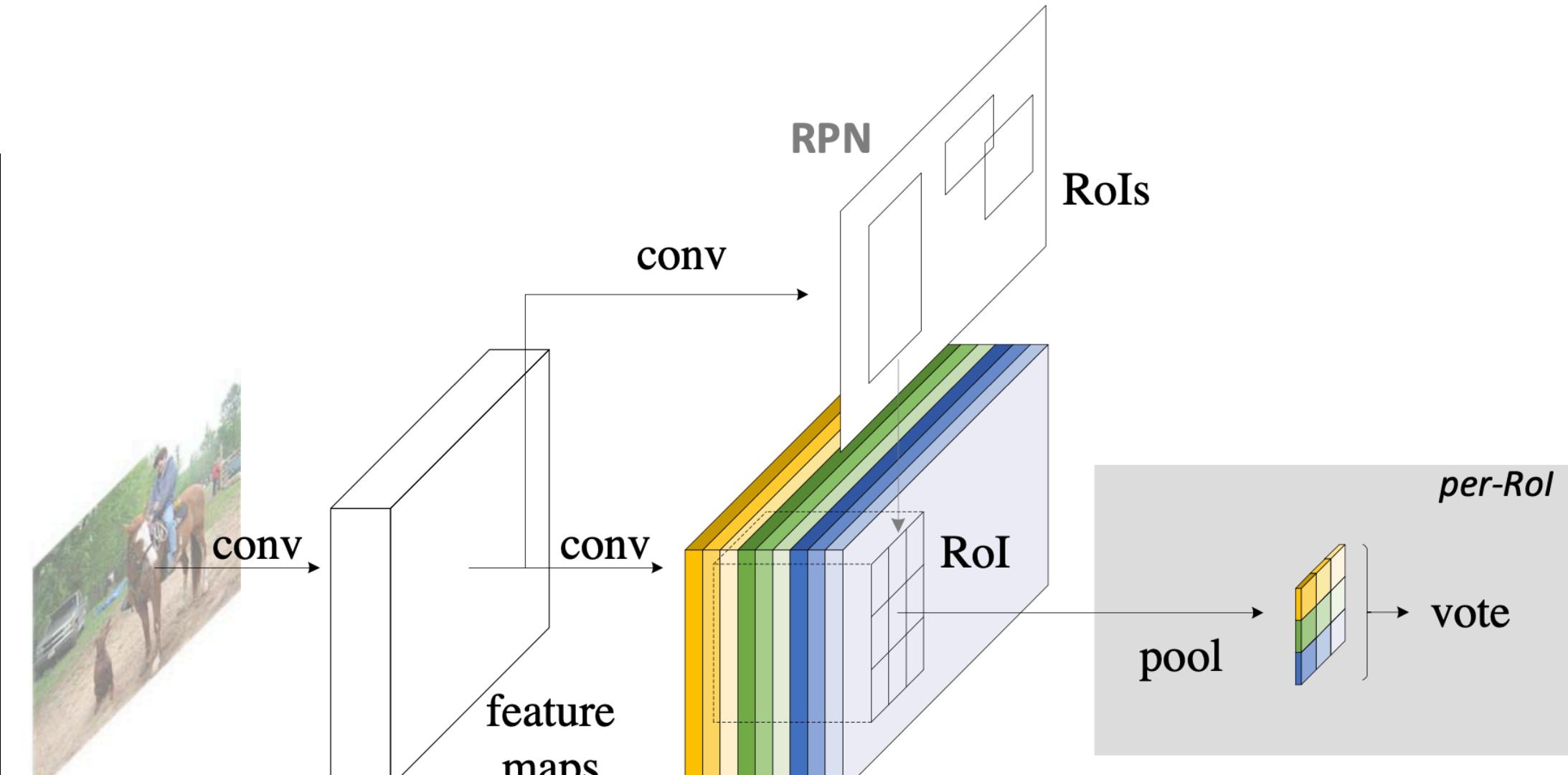
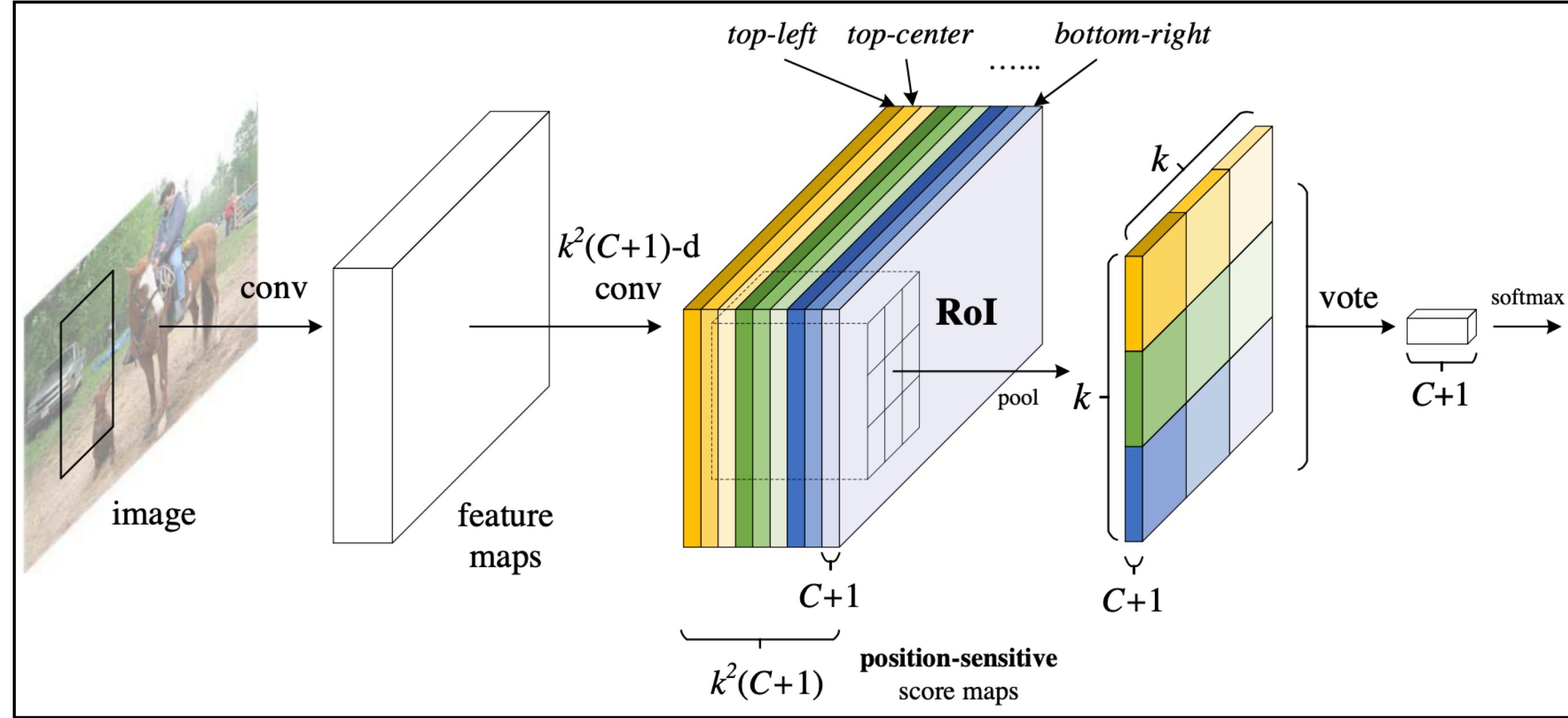
Boulder

R-FCN: Object Detection via Region-based Fully Convolutional Networks



[YouTube Video](#)

There exists a dilemma between translation-invariance in image classification and translation-variance in object detection.



$$r_c(i, j | \Theta) = \sum_{(x,y) \in \text{bin}(i,j)} z_{i,j,c}(x + x_0, y + y_0 | \Theta) / n$$

$$r_c(\Theta) = \sum_{i,j} r_c(i, j | \Theta) \rightarrow \text{voting}$$

Divide each RoI rectangle into $k \times k$ bins by a regular grid

For an RoI rectangle of a size $w \times h$, a bin is of a size $\approx \frac{w}{k} \times \frac{h}{k}$

$r_c(i, j)$ \rightarrow pooled response in the (i, j) -th bin for the c -th category

$z_{i,j,c}$ \rightarrow one score map out of the $k^2(C+1)$ score maps

(x_0, y_0) \rightarrow top-left corner of an RoI

n \rightarrow number of pixels in the bin

The (i, j) -th bin spans $\lfloor i \frac{w}{k} \rfloor \leq x < \lceil (i+1) \frac{w}{k} \rceil$ and $\lfloor j \frac{h}{k} \rfloor \leq y < \lceil (j+1) \frac{h}{k} \rceil$

Similarly, append a sibling $4k^2$ -d convolutional layer for bounding box regression.

	depth of per-RoI subnetwork	training w/ OHEM?	train time (sec/img)	test time (sec/img)	mAP (%) on VOC07
Faster R-CNN	10		1.2	0.42	76.4
R-FCN	0		0.45	0.17	76.6
Faster R-CNN	10	✓ (300 RoIs)	1.5	0.42	79.3
R-FCN	0	✓ (300 RoIs)	0.45	0.17	79.5
Faster R-CNN	10	✓ (2000 RoIs)	2.9	0.42	N/A
R-FCN	0	✓ (2000 RoIs)	0.46	0.17	79.3

OHEM: Online Hard Example Mining

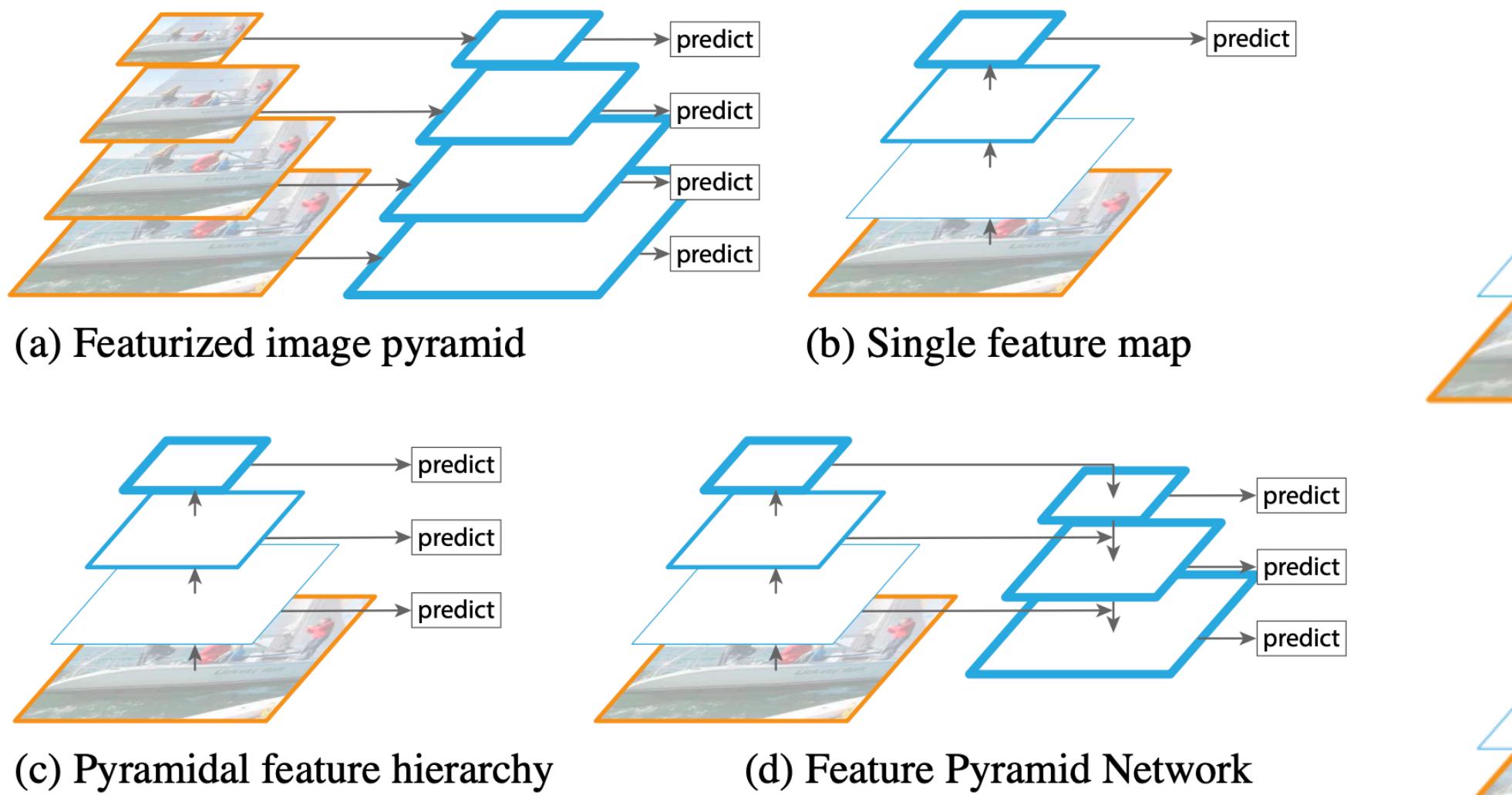


Boulder



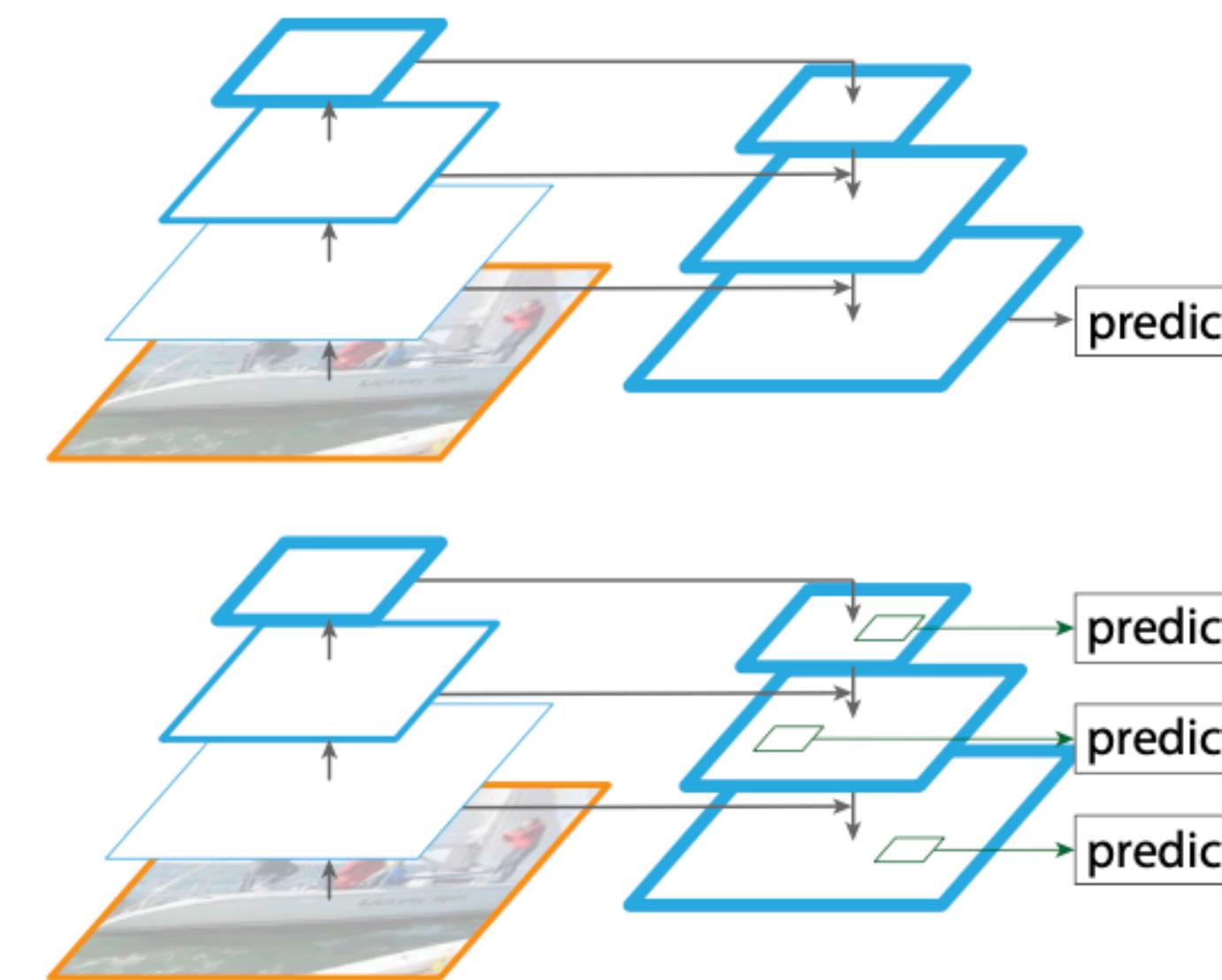
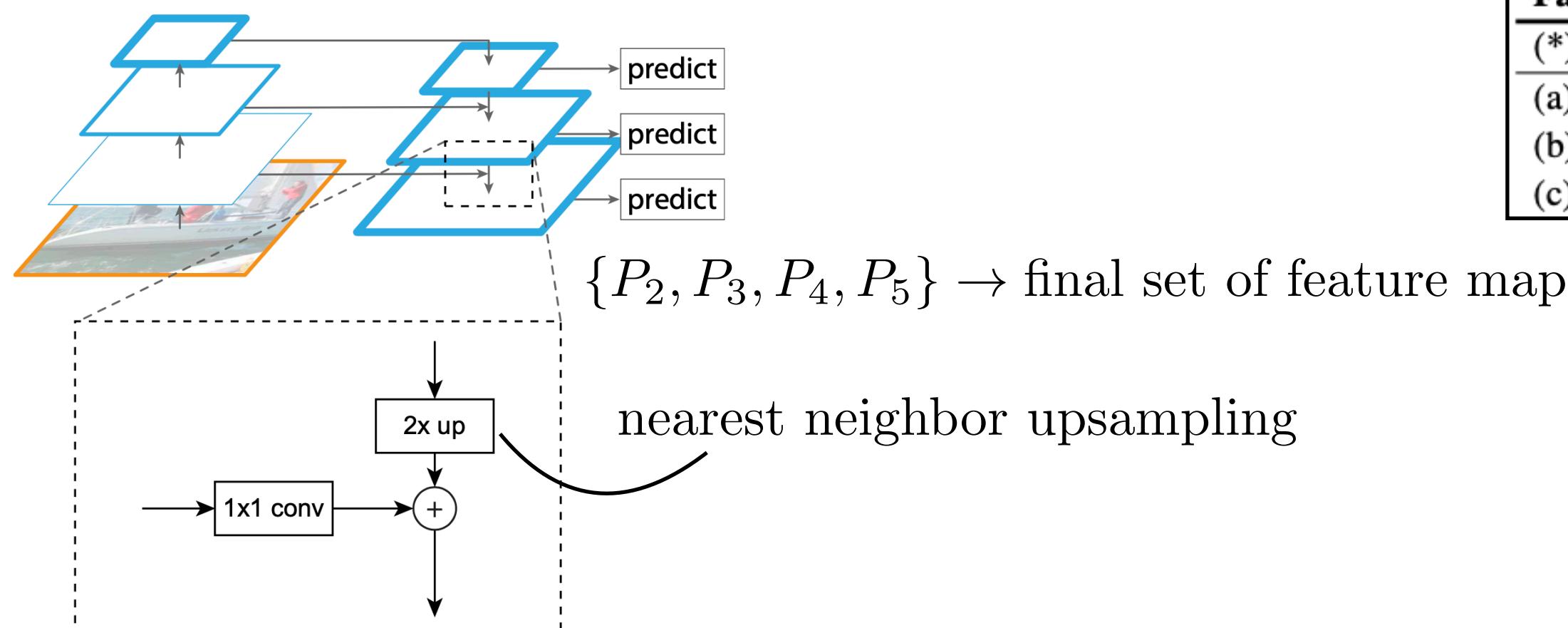
[YouTube Video](#)

Feature Pyramid Networks for Object Detection



Thicker blue outlines denote semantically stronger features.

$\{C_2, C_3, C_4, C_5\}$ → output of the last residual blocks for conv2, conv3, conv4 and conv5 in ResNets
 $\{4, 8, 16, 32\}$ → strides of pixels with respect to the input image



Region Proposal Network (RPN)

15 anchors over the pyramid:

$\{32^2, 64^2, 128^2, 256^2, 512^2\} \rightarrow$ areas of anchors on $\{P_2, P_3, P_4, P_5, P_6\}$, respectively
 $\{1:2, 1:1, 2:1\} \rightarrow$ aspect ratio

Region-of-Interest (RoI) pooling

Intuition: If the ROI's scale becomes smaller (say, 1/2 of 224), it should be mapped into a finer-resolution level (say, $k = 3$).

$w \& h \rightarrow$ width & height of ROI

$$k = \lfloor k_0 + \log_2(\sqrt{wh}/224) \rfloor$$

$$k_0 = 4$$

⇒ assign the ROI to P_k

COCO detection benchmark

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	2fc			51.7	28.0	9.6	31.9	43.1
(c) FPN	RPN, $\{P_k\}$	$\{P_k\}$	2fc	✓	✓	56.9	33.9	17.8	37.7	45.8

method	backbone	competition	image pyramid	test-dev				test-std					
				AP@.5	AP	AP _s	AP _m	AP _l	AP@.5	AP	AP _s	AP _m	AP _l
ours, Faster R-CNN on FPN	ResNet-101	-		59.1	36.2	18.2	39.0	48.2	58.5	35.8	17.5	38.7	47.8
<i>Competition-winning single-model results follow:</i>													
G-RMI [†]	Inception-ResNet	2016	-	34.7	-	-	-	-	-	-	-	-	-
AttractioNet [‡] [10]	VGG16 + Wide ResNet [§]	2016	✓	53.4	35.7	15.6	38.0	52.7	52.9	35.3	14.7	37.6	51.9
Faster R-CNN +++ [16]	ResNet-101	2015	✓	55.7	34.9	15.6	38.7	50.9	-	-	-	-	-
Multipath [40] (on minival)	VGG-16	2015		49.6	31.5	-	-	-	-	-	-	-	-
ION [‡] [2]	VGG-16	2015		53.4	31.2	12.8	32.9	45.2	52.9	30.7	11.8	32.8	44.8



Boulder

Deformable Convolutional Networks

geometric transformations: object scale, pose, view point, and part deformation

Convolution

2D convolution consists of two steps:

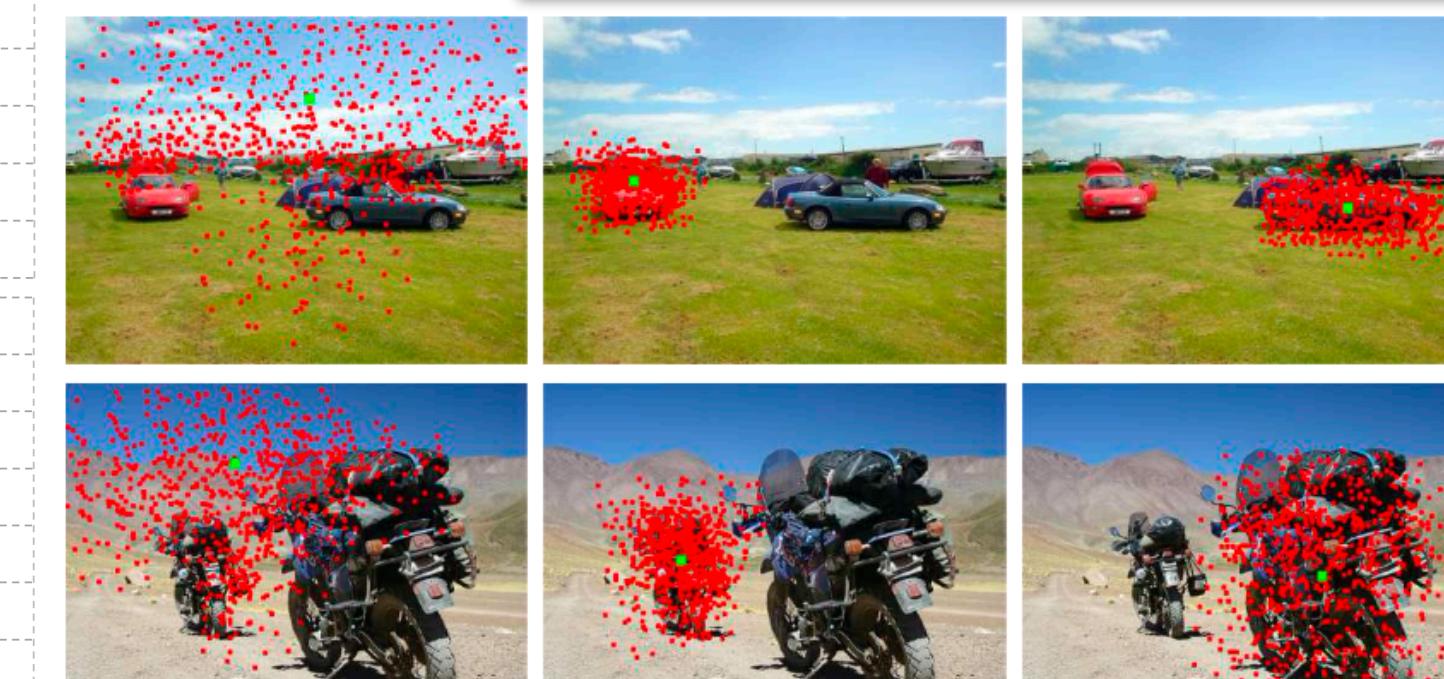
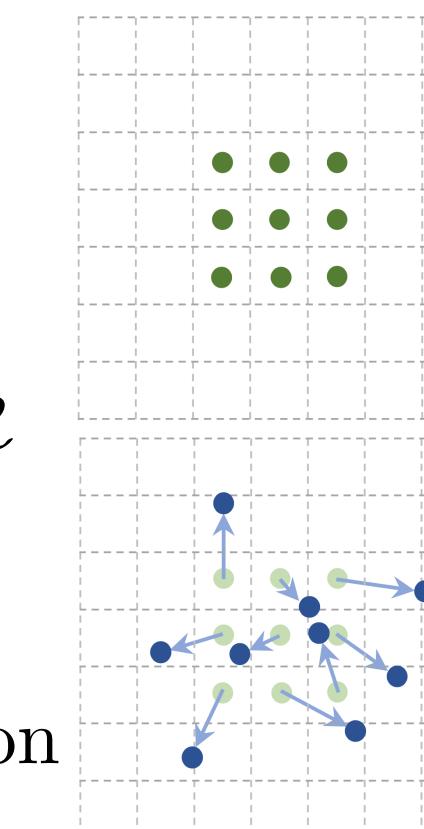
- sampling using a regular grid \mathcal{R} over the input feature map x
- summation of sampled values weighted by w

$\mathcal{R} \rightarrow$ defines receptive field size & dilation

$\mathcal{R} = \{(-1, -1), (-1, 0), \dots, (0, 1), (1, 1)\} \rightarrow 3 \times 3$ convolution with dilation 1

$y \rightarrow$ output feature map

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n)$$



Deformable Convolution

$\{\Delta p_n : n = 1, \dots, |\mathcal{R}|\} \rightarrow$ offsets augmenting \mathcal{R}

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n + \Delta p_n)$$

$$x(p) = \sum_q G(q, p) \cdot x(q) \rightarrow \text{bilinear interpolation}$$

$p \rightarrow$ an arbitrary (fractional) location (e.g., $p = p_0 + p_n + \Delta p_n$)

$q \rightarrow$ enumerates all integral spatial locations in the feature map x

$G(q, p) = g(q_x, p_x) \cdot g(q_y, p_y) \rightarrow$ non-zero only for a few q 's

$$g(a, b) = \max(0, 1 - |a - b|)$$

ROI Pooling

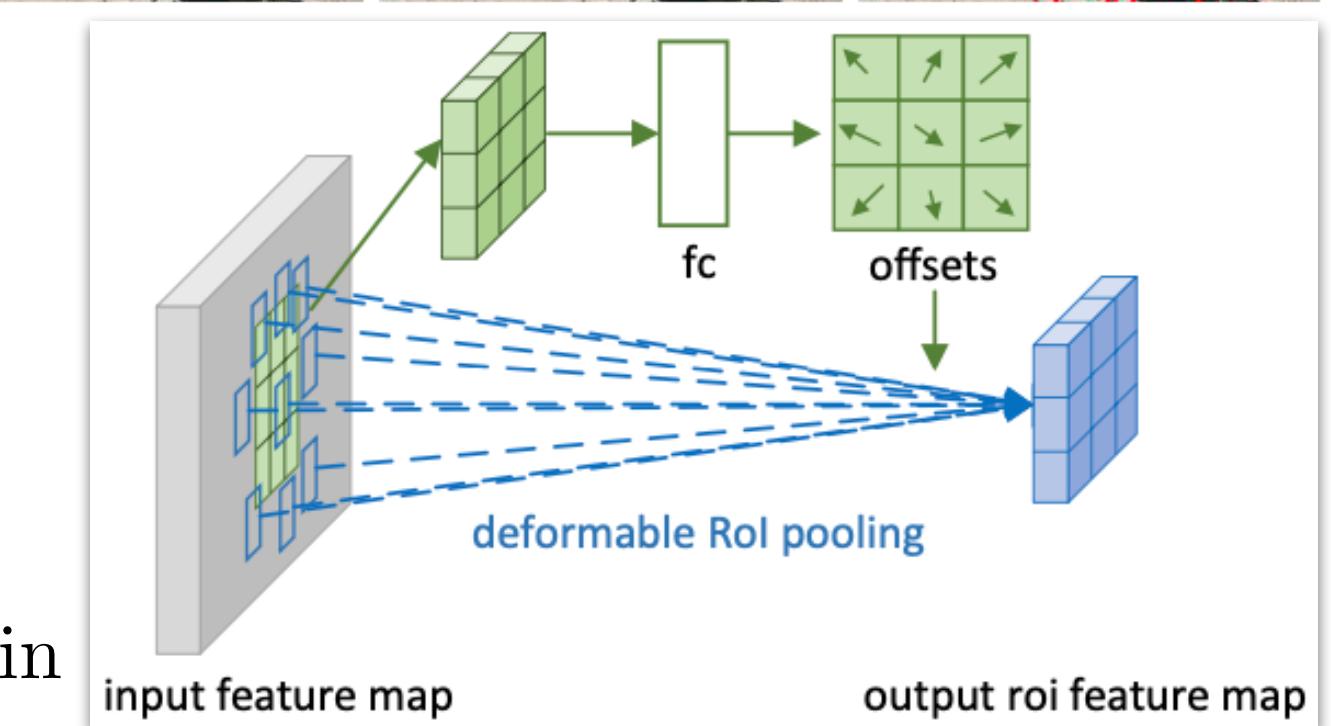
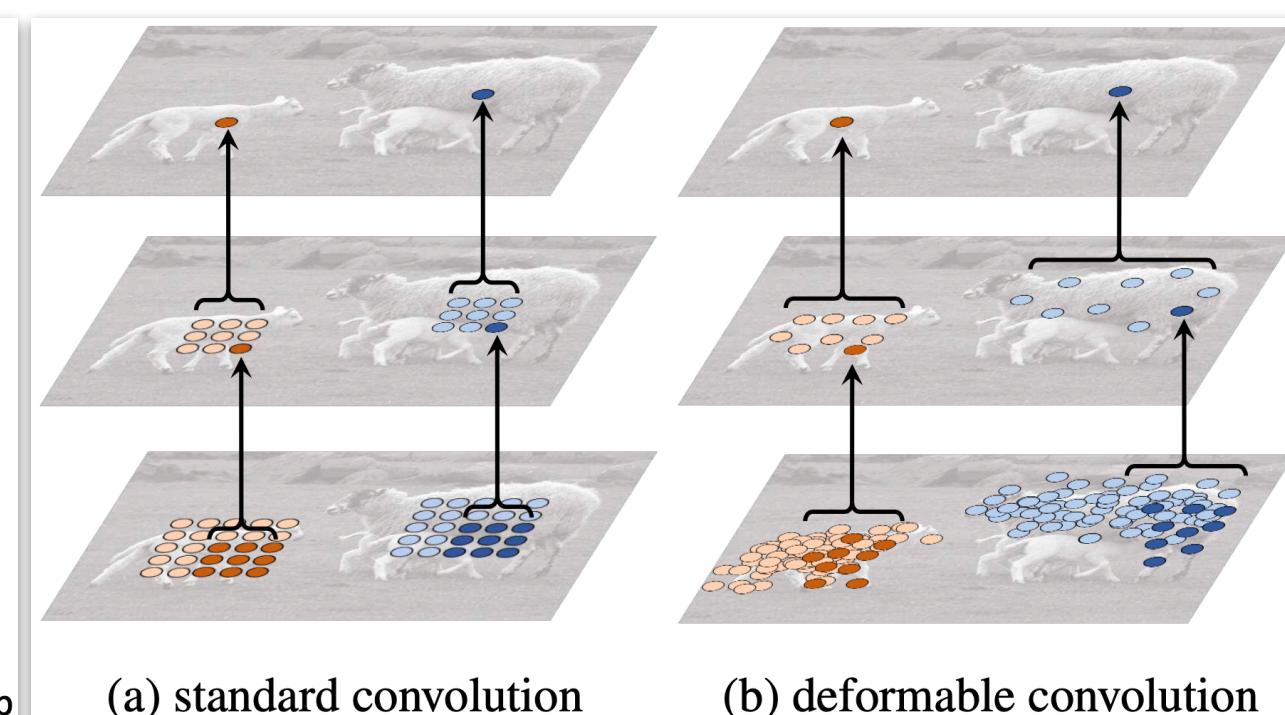
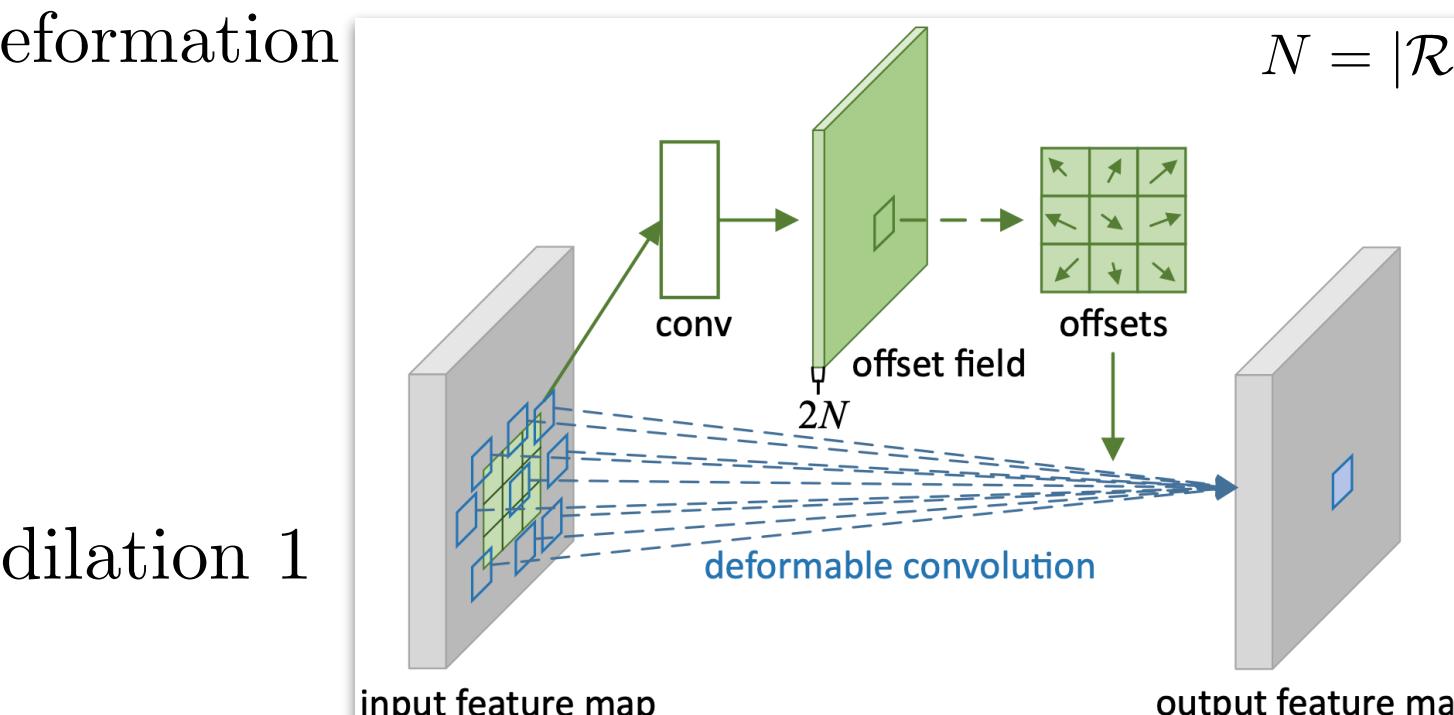
ROI of size $w \times h$ and top-left corner p_0

Divide the ROI into $k \times k$ bins

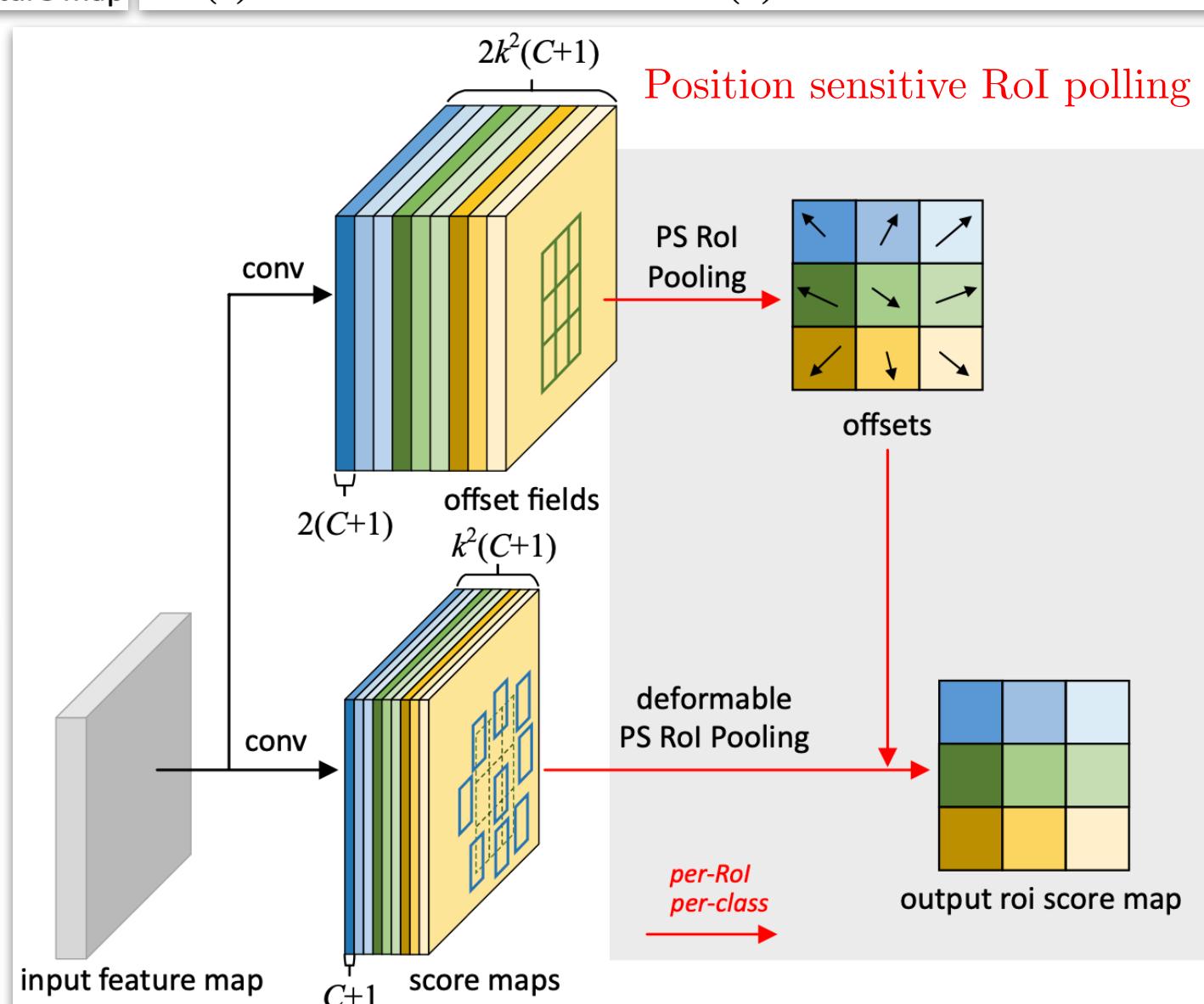
$$n_{ij} \rightarrow \text{number of pixels in the bin}$$

$y \rightarrow k \times k$ output feature map

$$y(i, j) = \sum_{p \in \text{bin}(i, j)} x(p_0 + p)/n_{ij} \text{ for } 0 \leq i, j < k$$



$$\{\Delta p_{ij} : 0 \leq i, j < k\} \rightarrow \text{offsets}$$



Deformable ROI Pooling

$$y(i, j) = \sum_{p \in \text{bin}(i, j)} x(p_0 + p + \Delta p_{ij})/n_{ij}$$

$$\Delta p_{ij} = \gamma \hat{\Delta p}_{ij} \odot (w, h) \quad \hat{\Delta p}_{ij} \rightarrow \text{normalized offsets}$$



Boulder



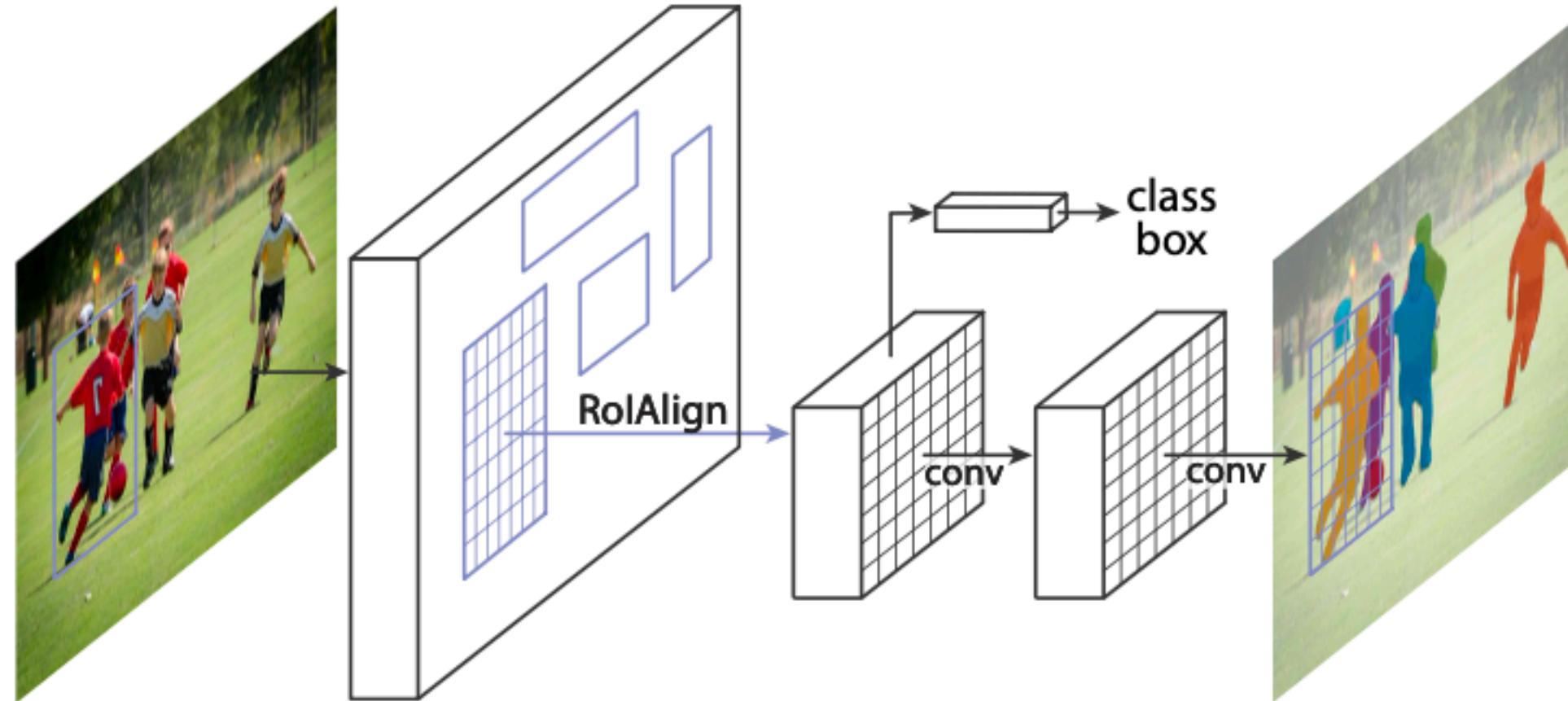
[YouTube Video](#)

Mask R-CNN

Object detection: classify individual objects and localize each using a bounding box

Semantic segmentation: classify each pixel into a fixed set of categories

Instance segmentation



$L = L_{\text{cls}} + L_{\text{box}} + L_{\text{mask}} \rightarrow$ multi-task loss on each sampled ROI

The mask branch has a Km^2 -dimensional output for each ROI

K binary masks of resolution $m \times m$, one for each of the K classes per pixel sigmoid

$L_{\text{mask}} \rightarrow$ average binary cross-entropy loss

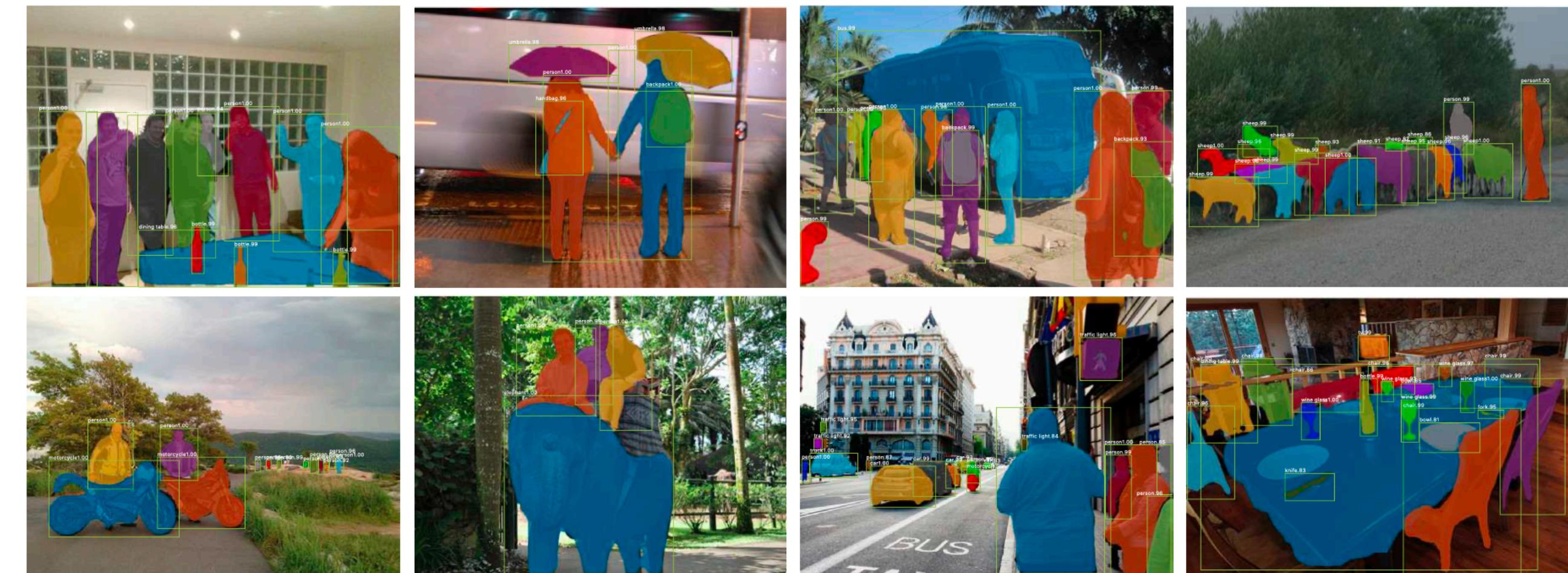
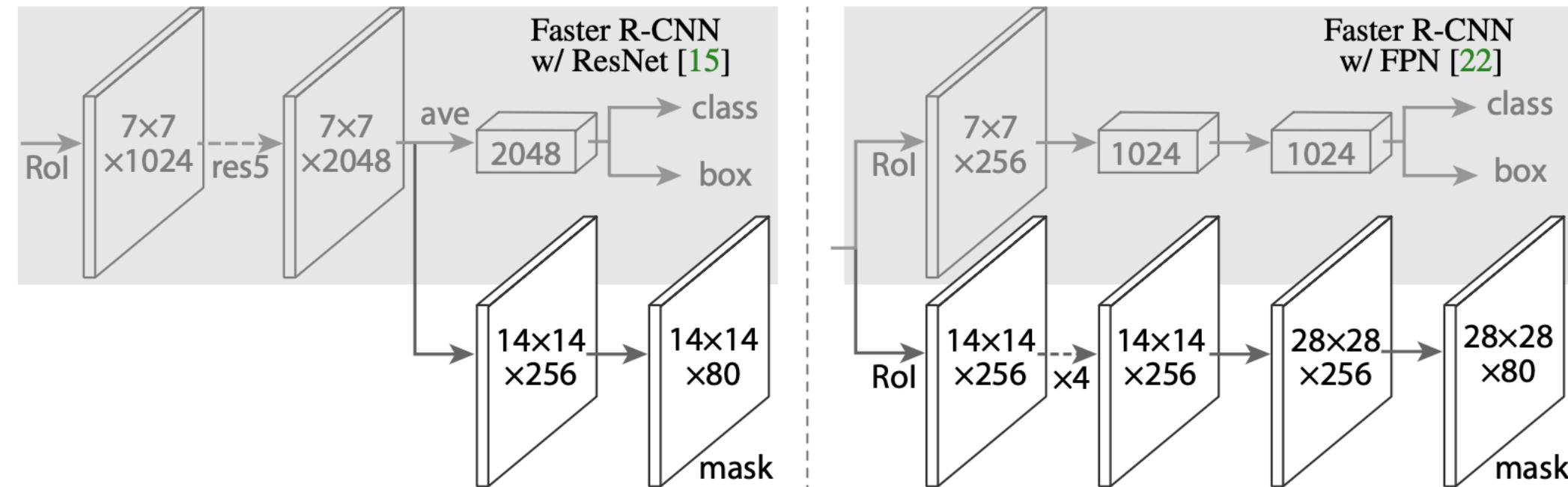
For an ROI associated with ground-truth class k , L_{mask} is only defined on the k -th mask (other mask outputs do not contribute to the loss).

RoIAlign $x \rightarrow$ a continuous coordinate

$\text{RoIPool} \rightarrow [x/16]$ (quantization)

Avoid any quantization of the ROI boundaries or bins (i.e., use $x/16$ instead of $[x/16]$)

Use bilinear interpolation to compute the exact values of the input features at four regularly sampled locations in each "ROI bin", and aggregate the result (using max or average).



A keypoint's location can be modeled as a one-hot mask, and Mask R-CNN can be adopted to predict K masks, one for each of K keypoint types (e.g., left shoulder, right elbow).



Boulder



Questions?

[YouTube Playlist](#)
