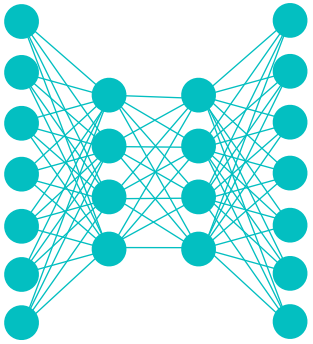
Lecture Notes for

Neural Networks and Machine Learning

Fully Convolutional Learning



What about transpose convolution?

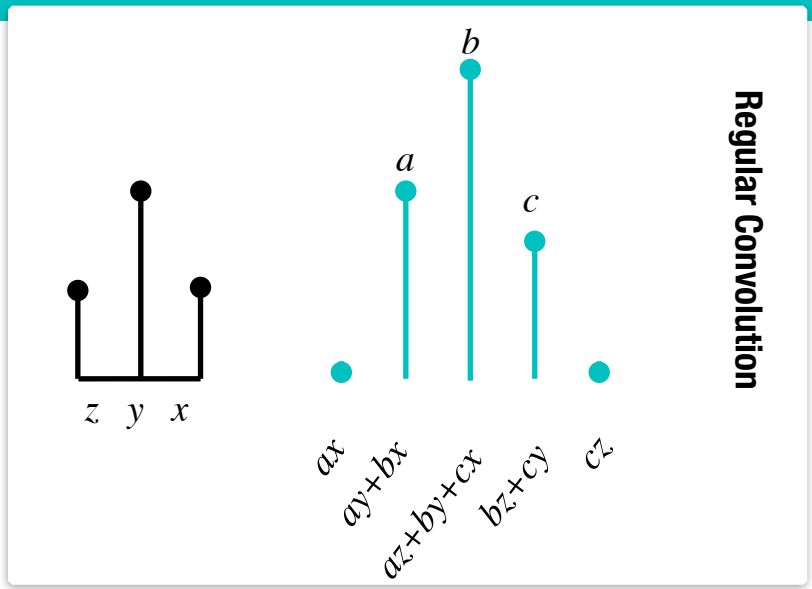
Convolution as Matrix Multiplication

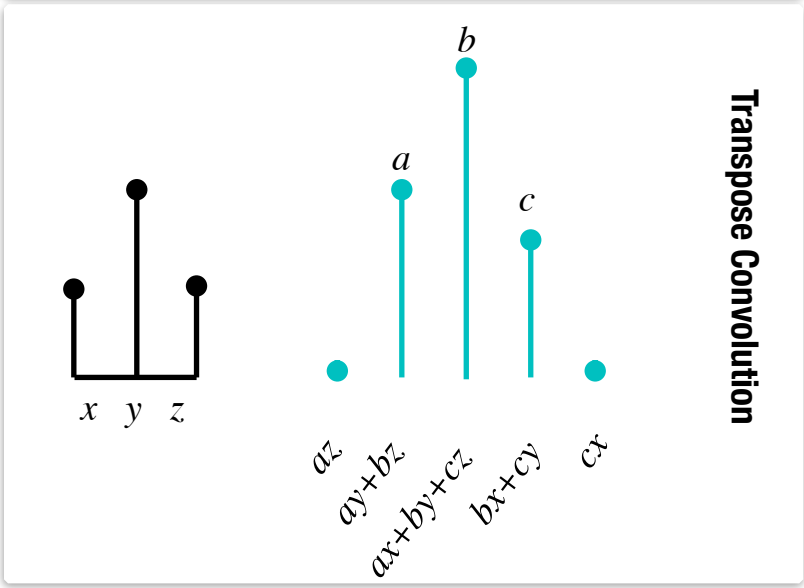
y	x	0	0	0		0	ax
z	y	x	0	0		a	$ay+bx$
0	z	y	x	0	x	b	$az+by+cx$
0	0	z	y	x		c	$bz+cy$
0	0	0	z	y		0	cz

Transpose

y	z	0	0	0		0	az
x	y	z	0	0		a	$ay+bz$
0	x	y	z	0	x	b	$ax+by+cz$
0	0	x	y	z		c	$bx+cy$
0	0	0	x	y		0	cx

like convolving with “reversed coefficients”





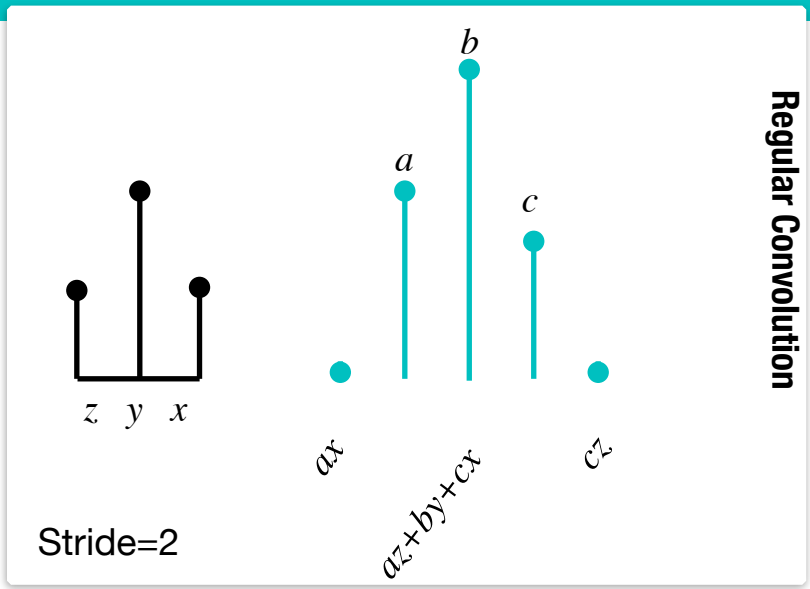
Transpose Convolution: Strides

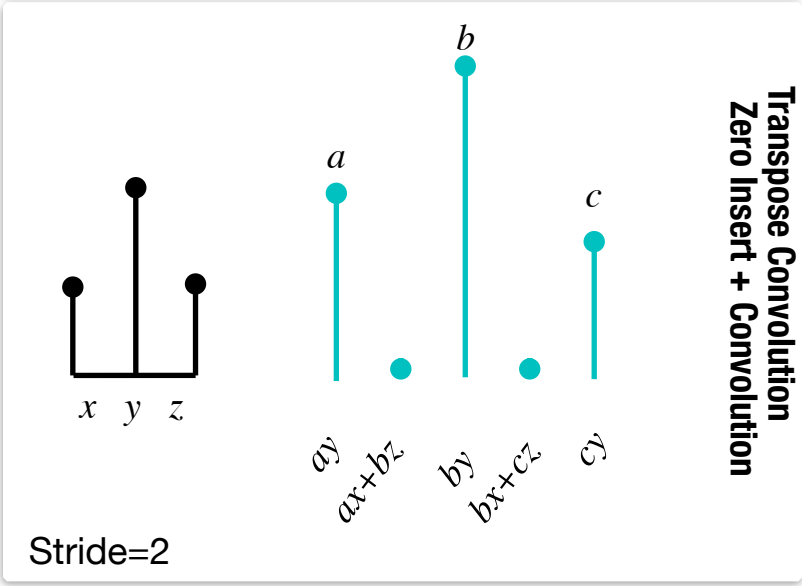
Strided Convolution as Matrix Multiplication

$$\begin{bmatrix} y & x & 0 & 0 & 0 \\ 0 & z & y & x & 0 \\ 0 & 0 & 0 & z & y \end{bmatrix} \times \begin{bmatrix} 0 \\ a \\ b \\ c \\ 0 \end{bmatrix} = \begin{bmatrix} ax \\ az+by+cx \\ cz \end{bmatrix}$$

Transpose

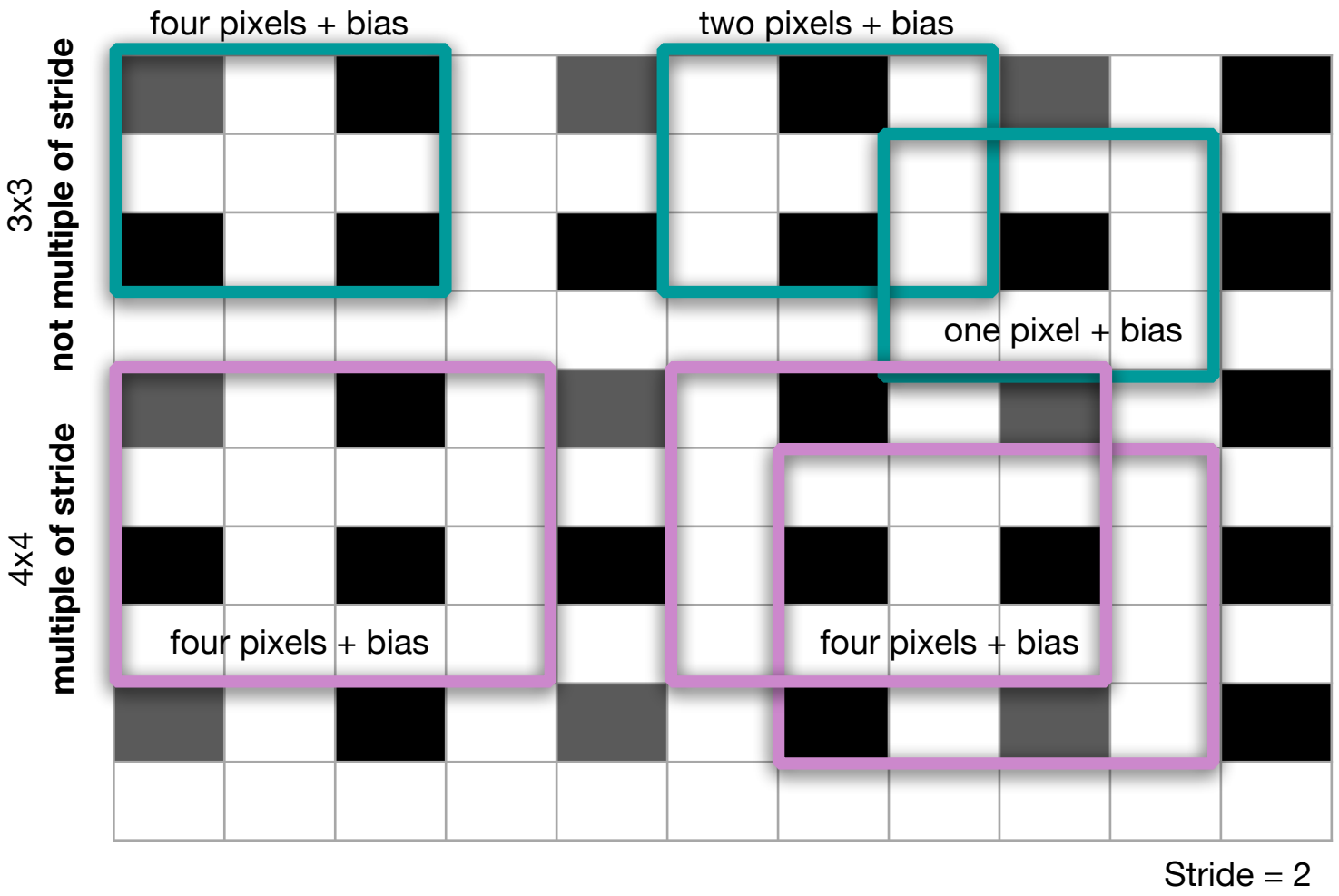
$$\begin{bmatrix} y & 0 & 0 \\ x & z & 0 \\ 0 & y & 0 \\ 0 & x & z \\ 0 & 0 & y \end{bmatrix} \times \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} ay \\ ax+bz \\ by \\ bx+cz \\ cy \end{bmatrix}$$





Aside: Convolution after zero insertion

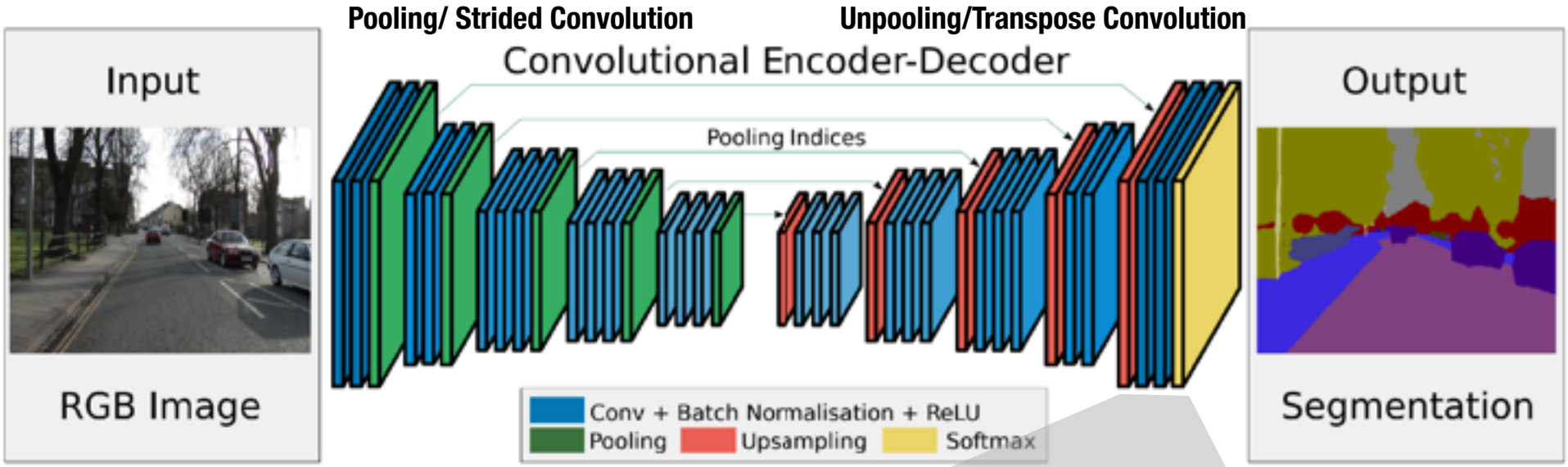
- Kernel size should be a symmetric multiple of the stride



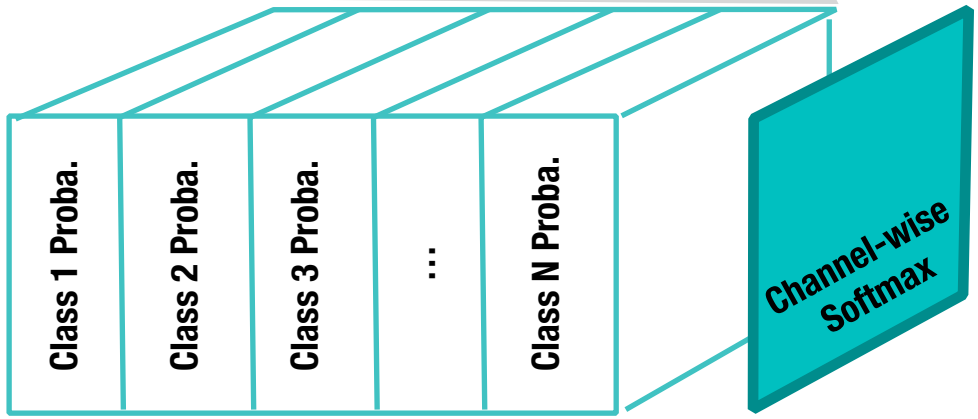
Bias needs to account for both when different numbers of pixels overlap with the kernel

Multiple of stride ensures that same number of active pixels overlap the kernel.

Putting it all together

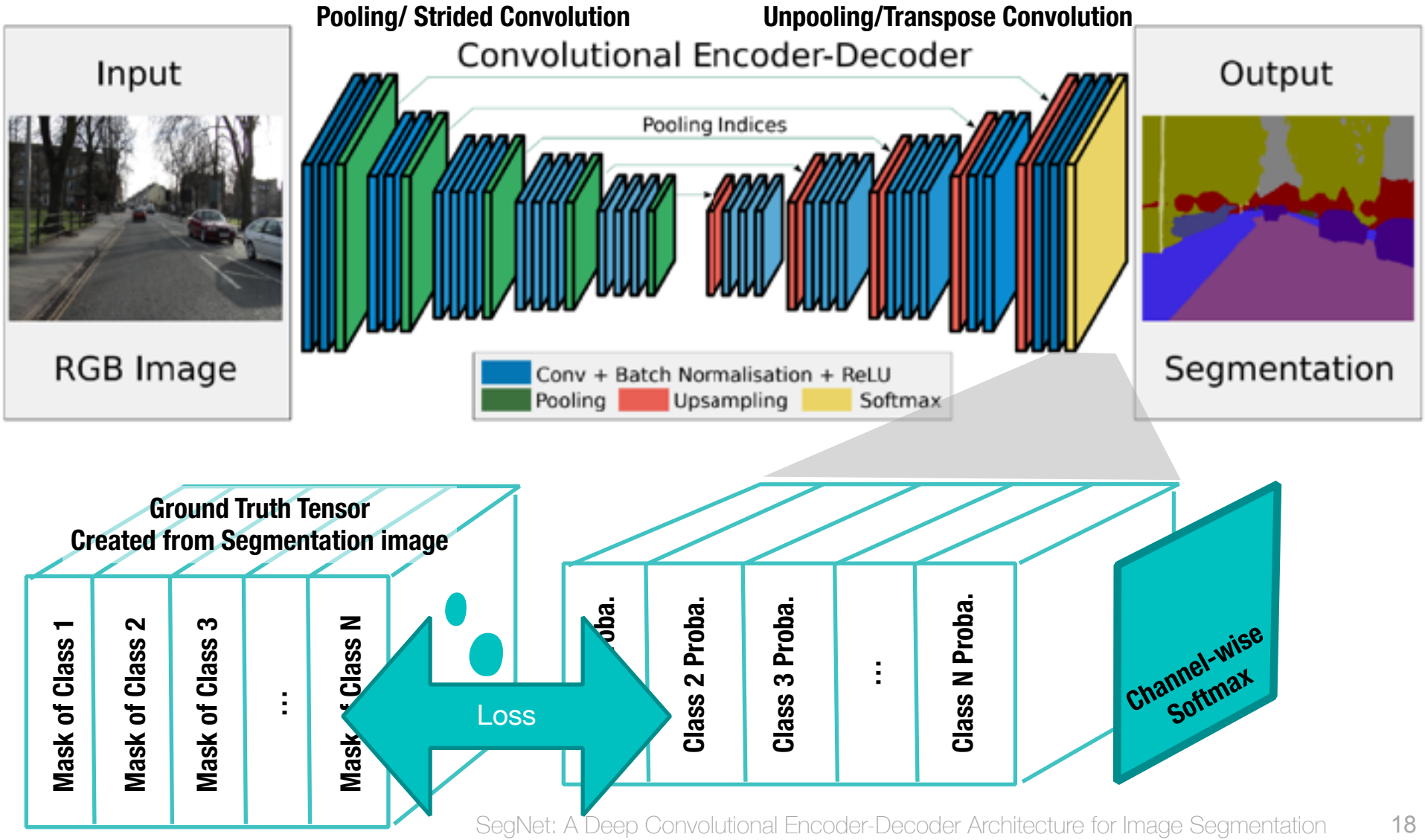


Self Test:
Does it change the architecture if the Image input size changes?



SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

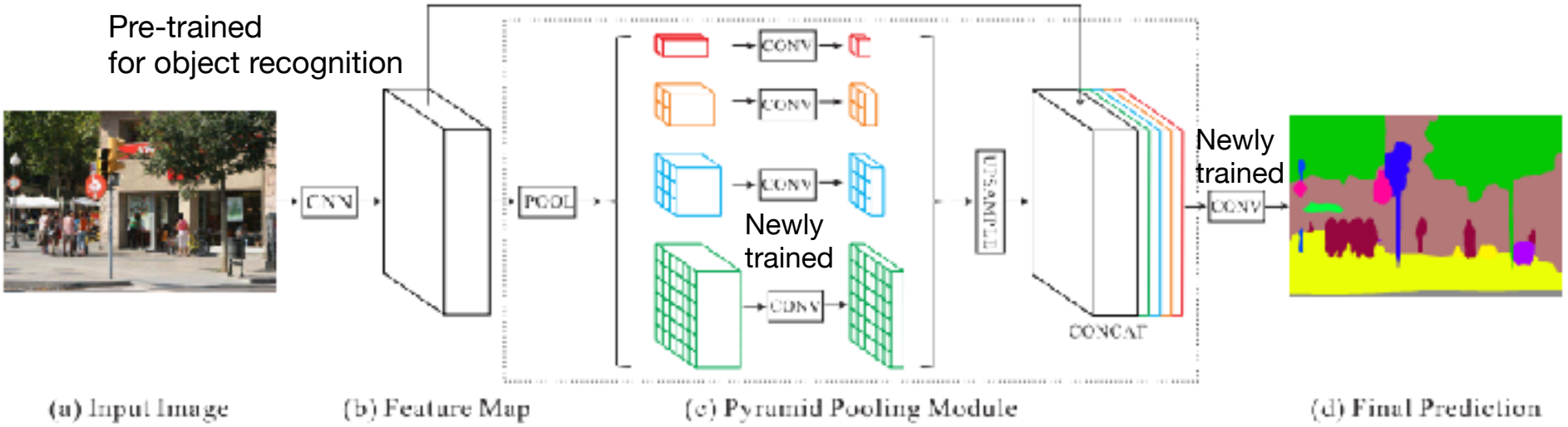
Putting it all together



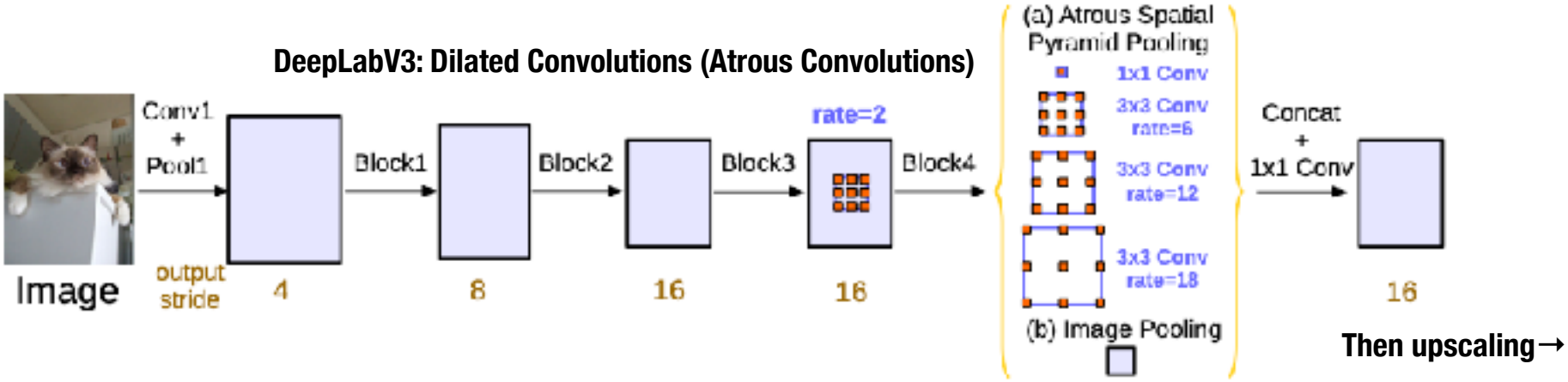
SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

Some Examples

Pyramid Scene Parsing Network (PSPNet)

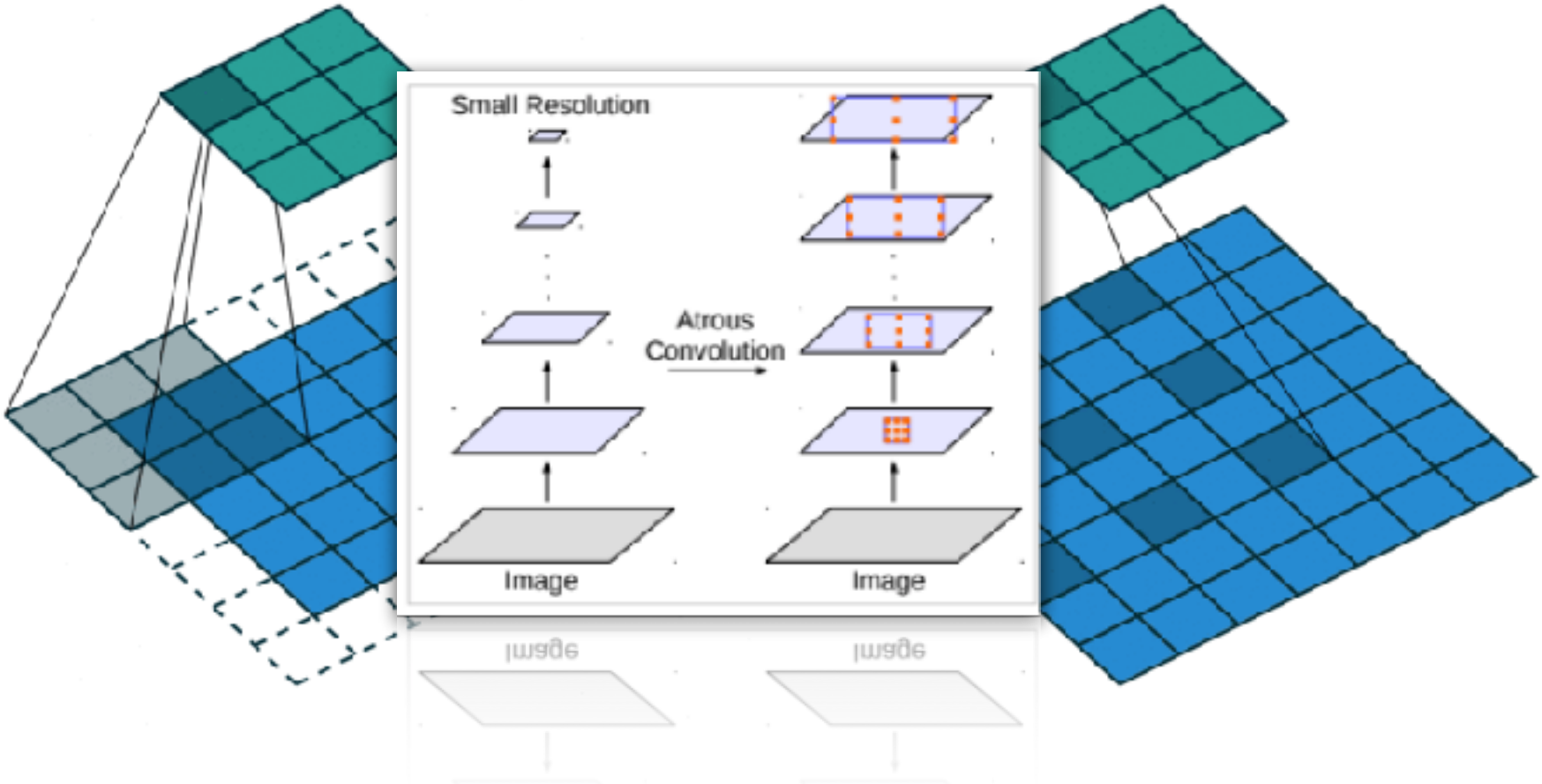


DeepLabV3: Dilated Convolutions (Atrous Convolutions)



https://towardsdatascience.com/semantic-segmentation-with-deep-learning-a-guide-and-code-e52fc8958823

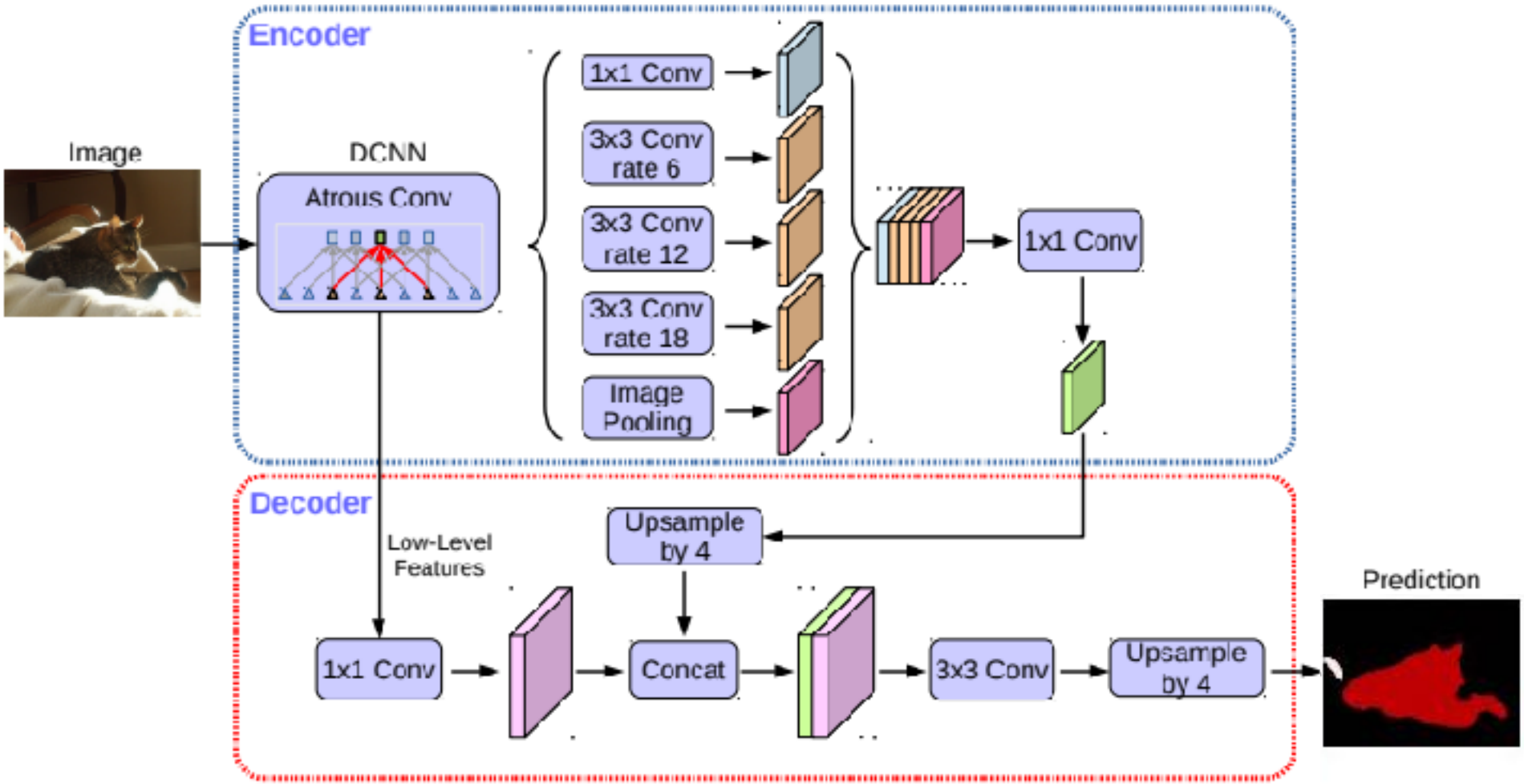
Dilated Convolution (Atrous)



Outputs of convolution are the same size, except for edge effects!
But have advantage of processing at a different scale.

https://towardsdatascience.com/review-dilated-convolution-semantic-segmentation-9d5a5bd768f5

DeepLabV3+



https://github.com/tensorflow/models/tree/master/research/deeplab

https://towardsdatascience.com/semantic-segmentation-with-deep-learning-a-guide-and-code-e52fc8958823

Gated-SCNN (Gate Shape CNN)

1 Shape stream employs Traditional Image Processing for edge detection (**image gradients**)

2 Uses activations to "gate" the image gradient. $\sigma(A) \odot I_{grad}$

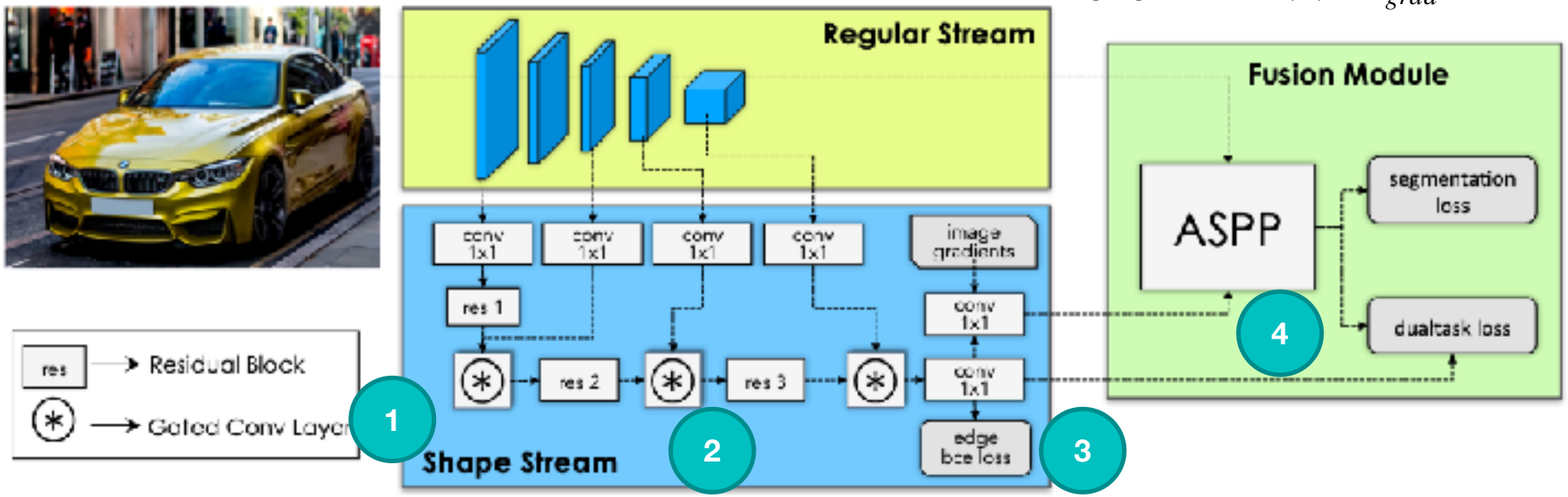


Figure 2: **GSCNN architecture.** Our architecture constitutes of two main streams. The regular stream and the shape stream. The regular stream can be any backbone architecture. The shape stream focuses on shape processing through a set of residual blocks, Gated Convolutional Layers (GCL) and supervision. A fusion module later combines information from the two streams in a multi-scale fashion using an Atrous Spatial Pyramid Pooling module (ASPP). High quality boundaries on the segmentation masks are ensured through a Dual Task Regularizer .

3 Also uses Labeled Boundaries in BCE Edge Loss Function

4 Merges segmentation with edges for finer masks. Concatenate + atrous convolution

https://heartbeat.fritz.ai/a-2019-guide-to-semantic-segmentation-ca8242f5a7fc

Performance

Figure 3: Illustration of the crops used for the distance-based evaluation.

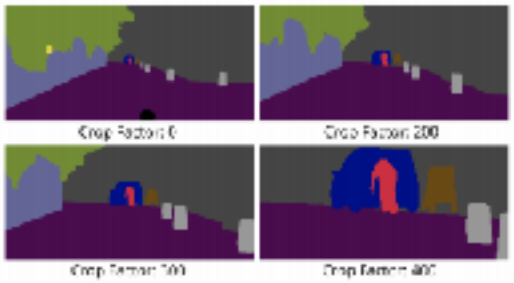


Figure 4: Predictions at diff. crop factors.

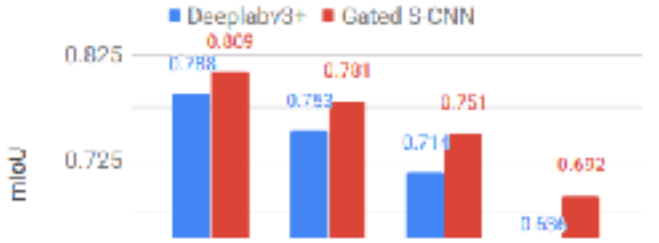


Figure 5: **Distance-based evaluation**: Comparison of mIoU at different crop factors.

Method	road	s.walk	build.	wall	fence	pole	t-light	t-sign	veg	terrain	sky	person	rider	car	truck	bus	train	motor	bike	mean
LRR [18]	97.7	79.9	90.7	44.4	48.6	58.6	68.2	72.0	92.5	69.3	94.7	81.6	60.0	94.0	43.6	56.8	47.2	54.8	69.7	69.7
DeepLabV2 [9]	97.9	81.3	90.3	48.8	47.4	49.6	57.9	67.3	91.9	69.4	94.2	79.8	59.8	93.7	56.5	67.5	57.5	57.7	68.8	70.4
Piecewise [32]	98.0	82.6	90.6	44.0	50.7	51.1	65.0	71.7	92.0	72.0	94.1	81.5	61.1	94.3	61.1	65.1	53.8	61.6	70.6	71.6
PSP-Net [38]	98.2	85.8	92.8	57.5	65.9	62.6	71.8	80.7	92.4	64.5	94.8	82.1	61.5	95.1	78.6	88.3	77.9	68.1	78.0	78.8
DeepLabV3+ [11]	98.2	84.9	92.7	**57.3**	62.1	65.2	68.6	78.9	92.7	63.5	95.3	82.3	62.8	95.4	**85.3**	89.1	80.9	64.6	77.3	78.8
Ours (GSCNN)	**98.3**	**86.3**	**93.3**	55.8	**64.0**	**70.8**	**75.9**	**83.1**	**93.0**	**65.1**	**95.2**	**85.3**	**67.9**	**96.0**	80.8	**91.2**	**83.3**	**69.6**	**80.4**	**80.8**

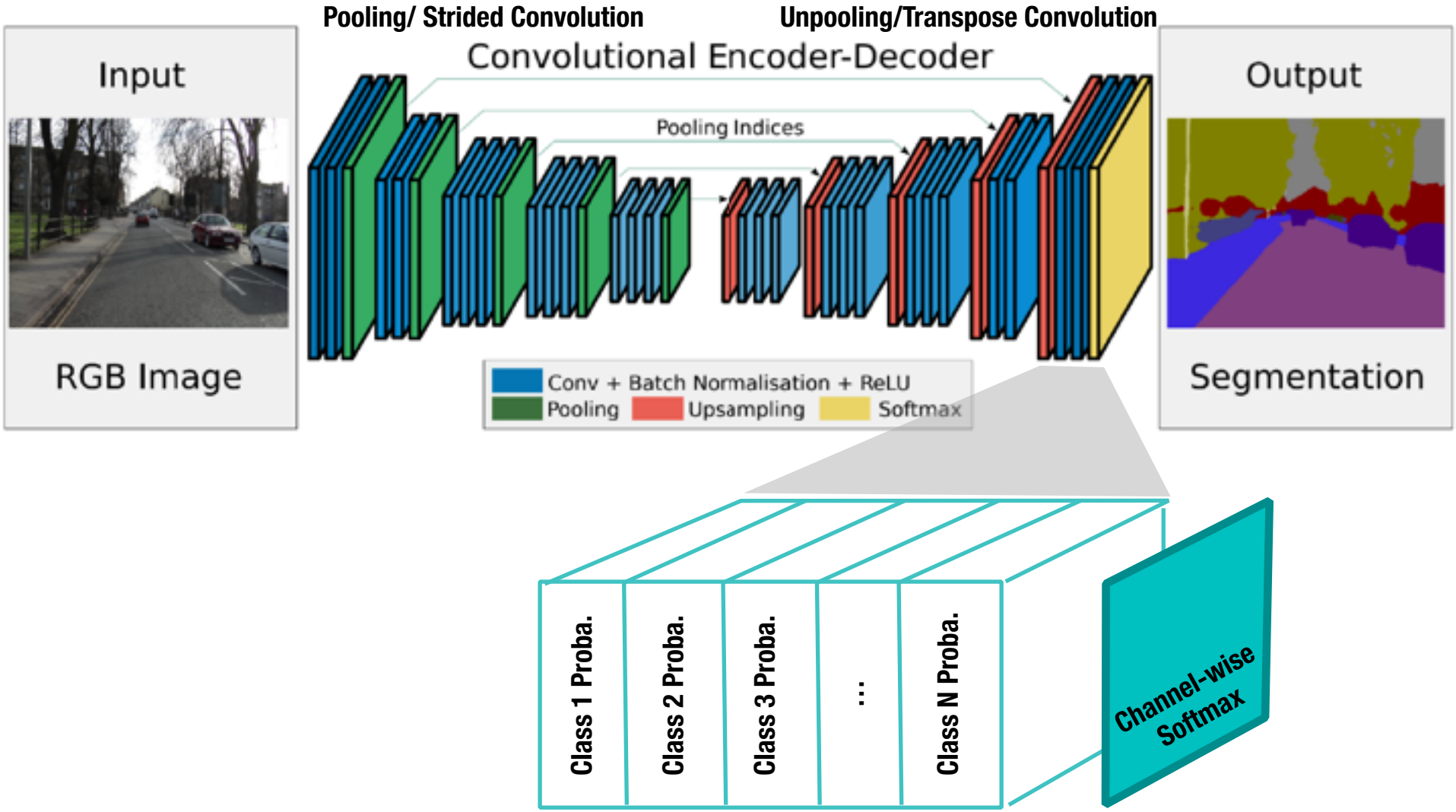
Table 1: Comparison in terms of IoU vs state-of-the-art baselines on the Cityscapes val set.

mIoU == mean Intersection over Union = $\frac{\text{Area of Overlap}}{\text{Area of Union}}$

Logistics and Agenda

- Logistics
  - Lab One
- Agenda
  - Segmentation
    - Semantic (last time)
    - Object (this time)
    - Instance (next time, probably)

Last Time

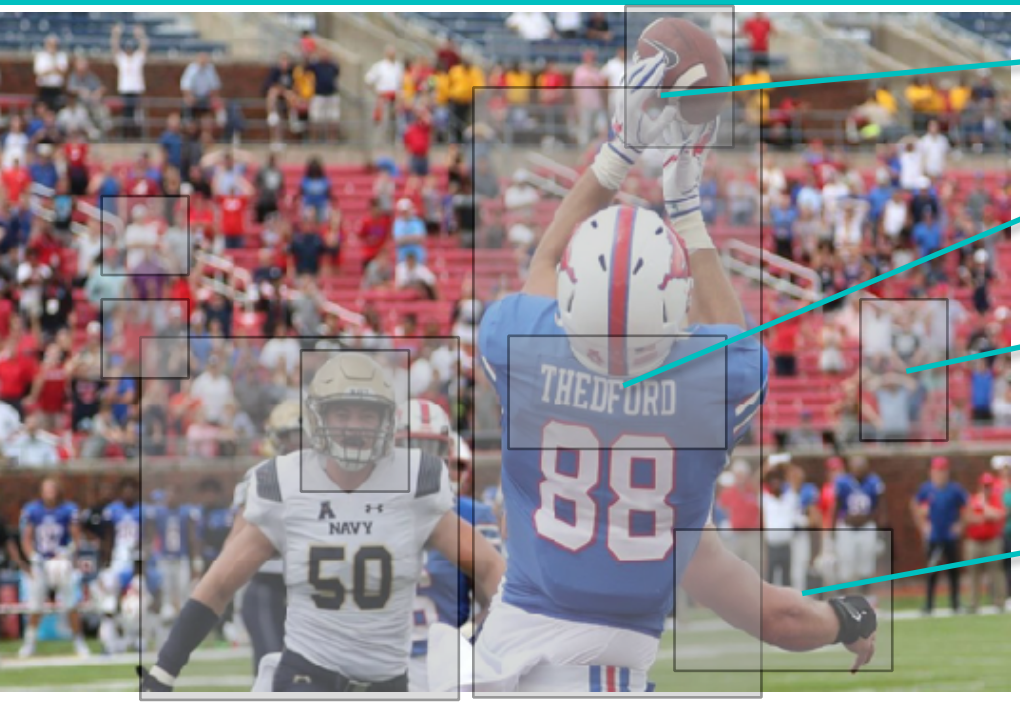

Pooling/ Strided Convolution
Unpooling/Transpose Convolution
Input
Convolutional Encoder-Decoder
Output
Pooling Indices
RGB Image
Conv + Batch Normalisation + ReLU
Pooling
Upsampling
Softmax
Segmentation
Class 1 Proba.
Class 2 Proba.
Class 3 Proba.
...
Class N Proba.
Channel-wise Softmax

Object Segmentation

A history in naming one network five different times with five different papers each time changing one thing about the architecture

Research!

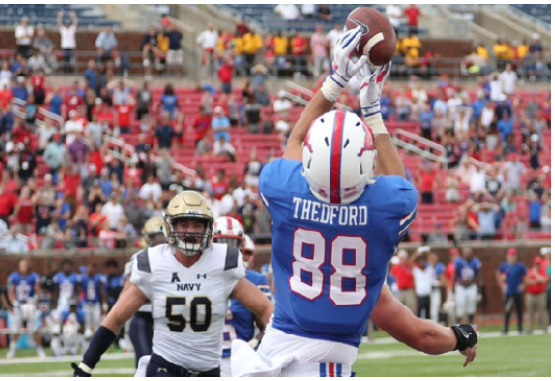
2014: R-CNN

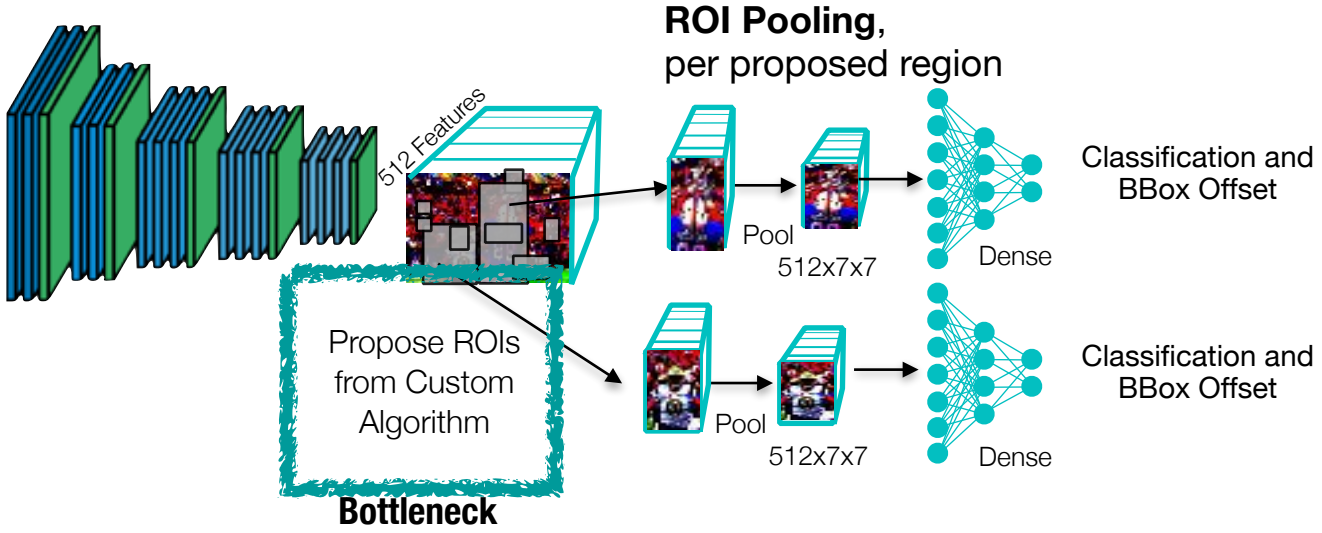


- Too Slow to Be Useful
- SVM and BBox Regression Trained Separately
- Fine Tuned Existing ConvNet (for Warped Images)
- ~50 Seconds per Image when Deployed

Girshick et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, 2014

2015: Fast R-CNN

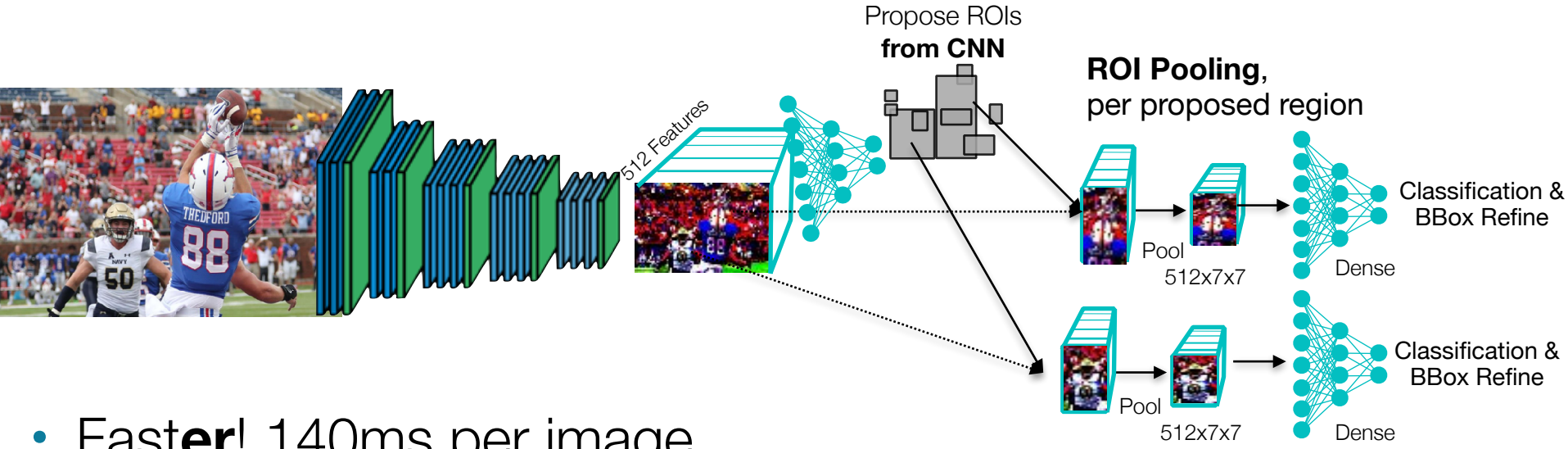




- Fast! 2.3 seconds per image (not ~50)
- But still not real time…

Girshick et al. Fast R-CNN, 2015, February

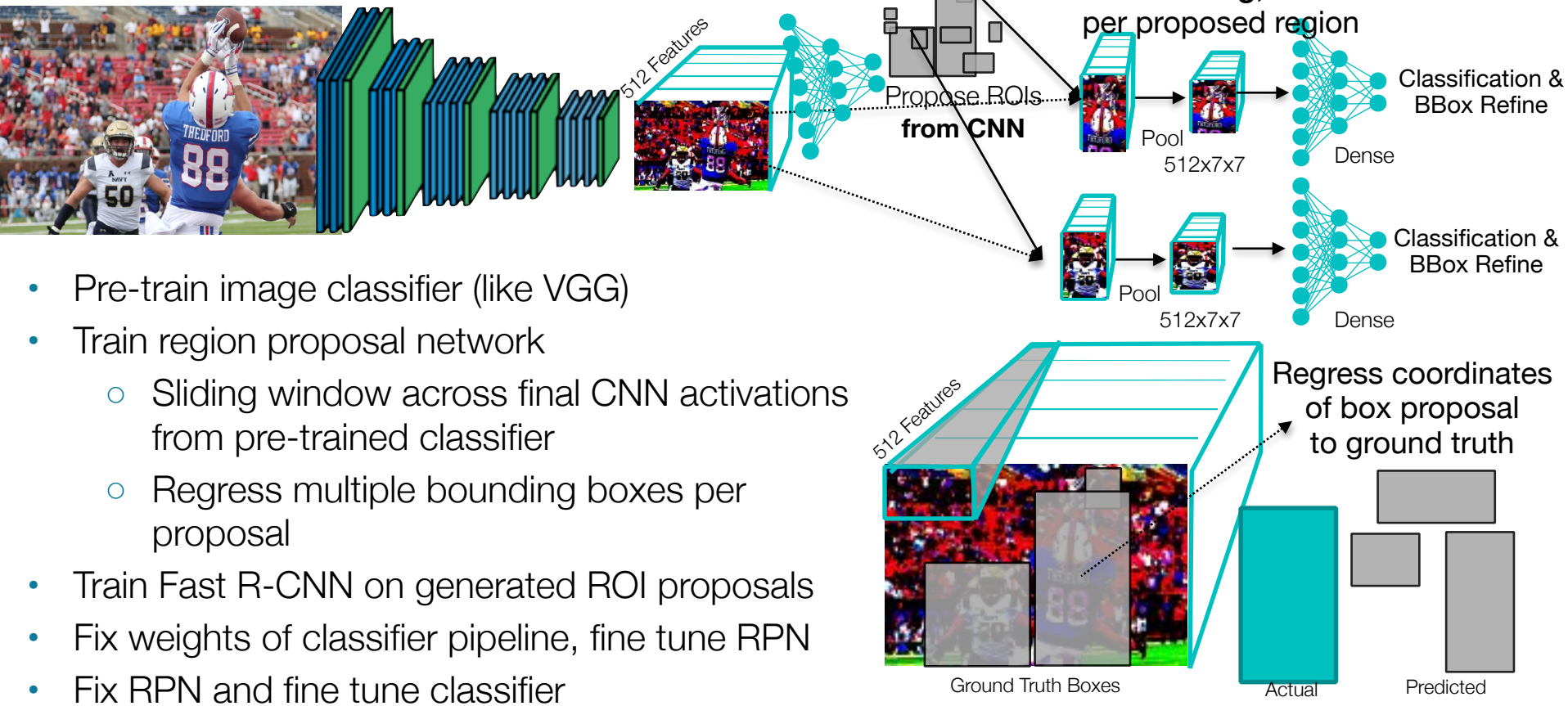
2015: Faster R-CNN



- Fast**er**! 140ms per image (7 FPS)
- Highly Accurate

Ren et al. Faster R-CNN, 2015, November

2015: Faster R-CNN, Training



- Pre-train image classifier (like VGG)
- Train region proposal network
  - Sliding window across final CNN activations from pre-trained classifier
  - Regress multiple bounding boxes per proposal
- Train Fast R-CNN on generated ROI proposals
- Fix weights of classifier pipeline, fine tune RPN
- Fix RPN and fine tune classifier
- Rinse, repeat fine tuning

$$l_{box} = \sum_i \mathbf{1}_i \left[(x - \hat{x}_i)^2 + (y - \hat{y}_i)^2 + (\log w - \log \hat{w}_i)^2 + (\log h - \log \hat{h}_i)^2 \right]$$

$$l_{class} = \sum_c entropy(c, \hat{c})$$

Ren et al. Faster R-CNN, 2015, November

In Parallel…

- Some of the same researchers started working on a similar, but radically different output approach

Don’t want to listen to me explain it?

- Check out Joseph Redmon’s Talk at CVPR 2016:
  - https://www.youtube.com/watch?v=NM6lrxy0bxs
  - This is how you give a Technical presentation, plus he is from the school where I did graduate school… so he is clearly superior by grace of proximity…

2018: Everybody has a Gimmick

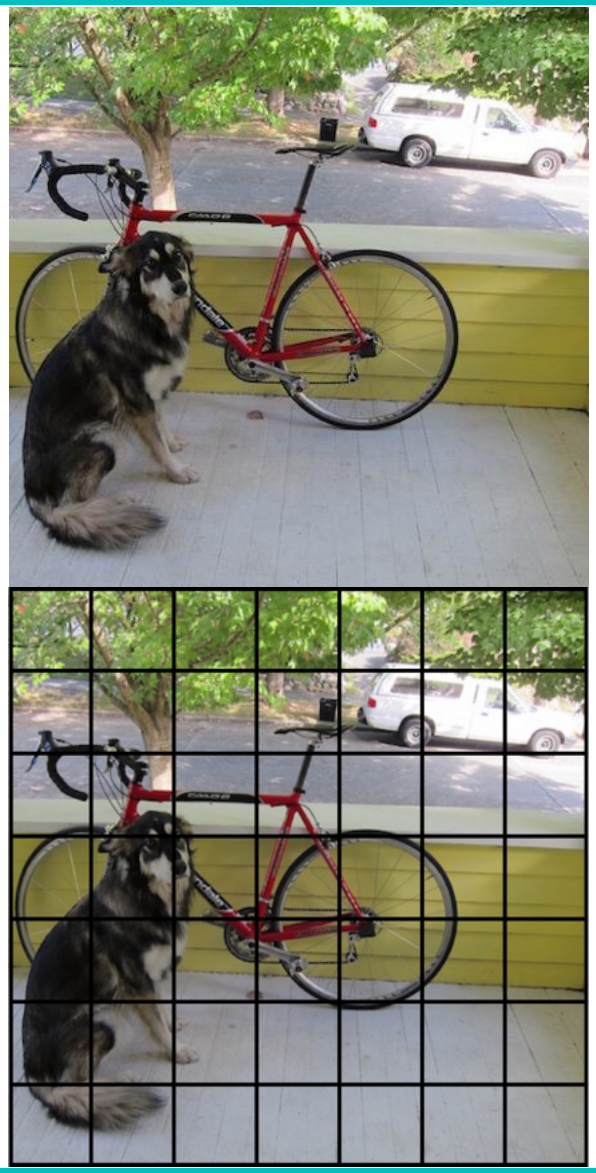
YOLO9000:
Better, Faster, Stronger

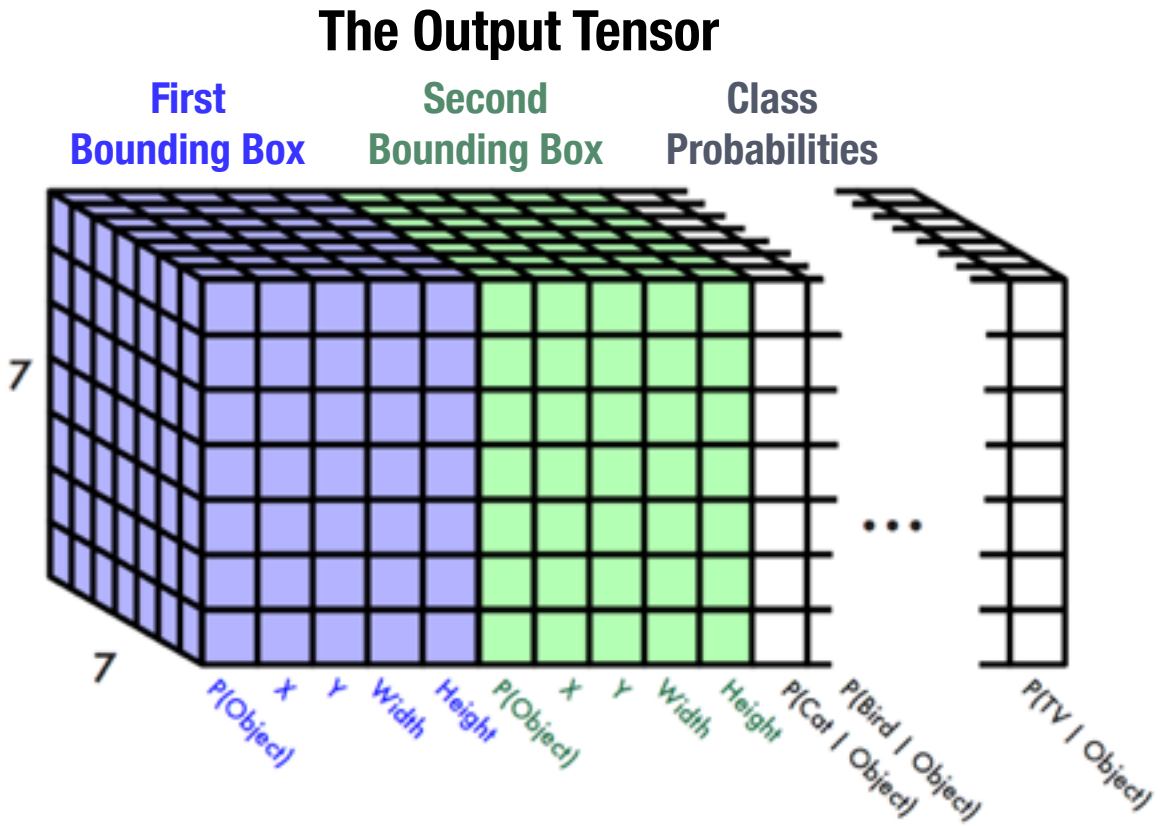
Joseph Redmon*†, Ali Farhadi*†
University of Washington*, Allen Institute for AI†
http://pjreddie.com/yolo9000/

- YOLO ~40-60 FPS
- Slightly more Accurate than **Faster R-CNN**

Redmon and Farhadi, YOLO9000: Better, Faster, Stronger, 2016, December 25 — *Merry Christmas?*

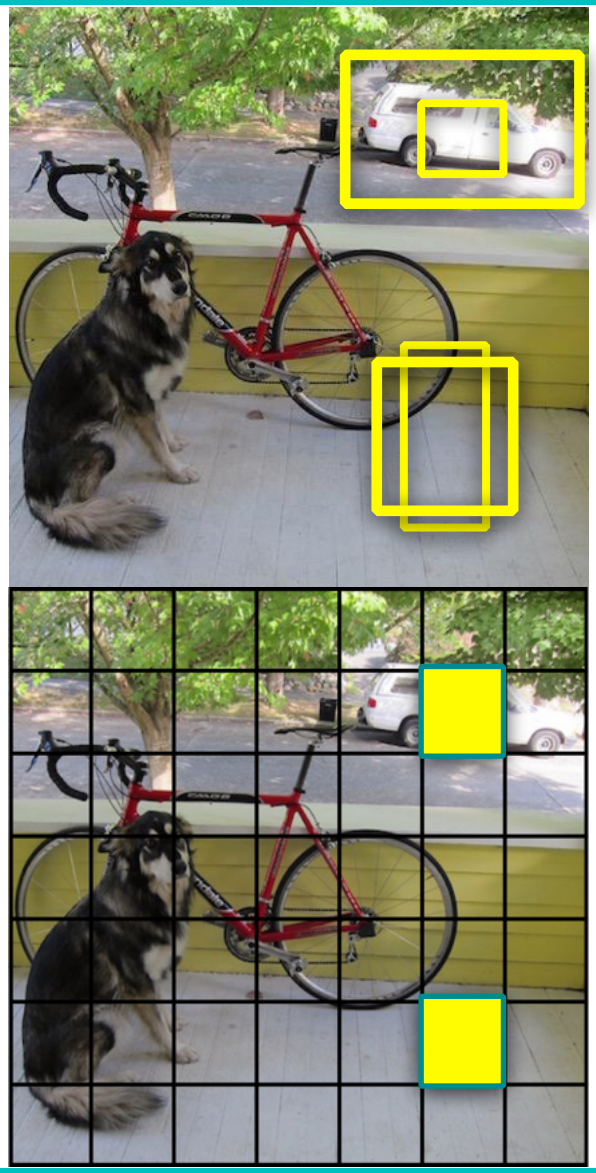
The YOLO Output Tensor

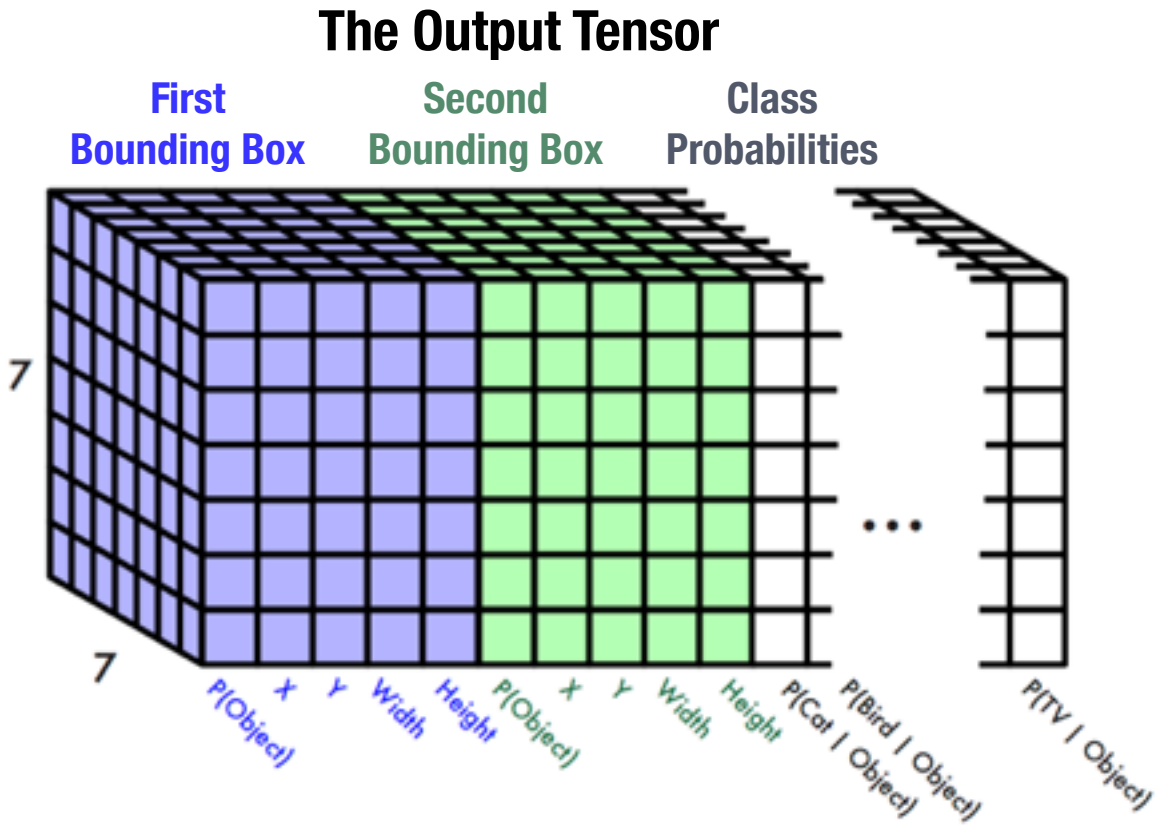




Redmon and Farhadi, YOLO9000: Better, Faster, Stronger, 2016, December 25 — *Merry Christmas?*

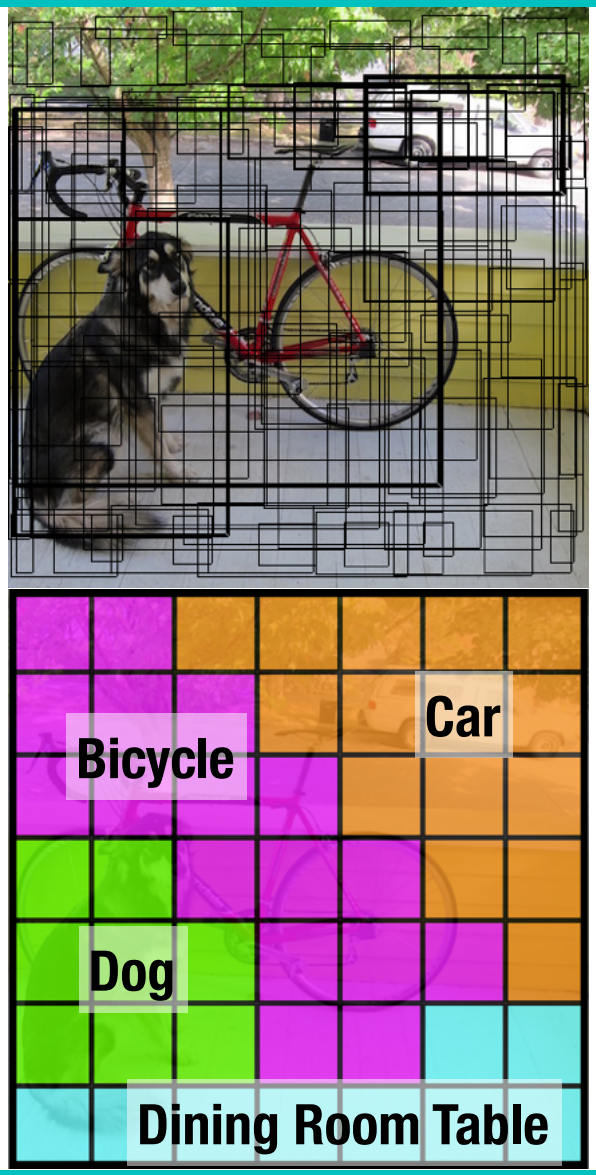
The YOLO Output Tensor



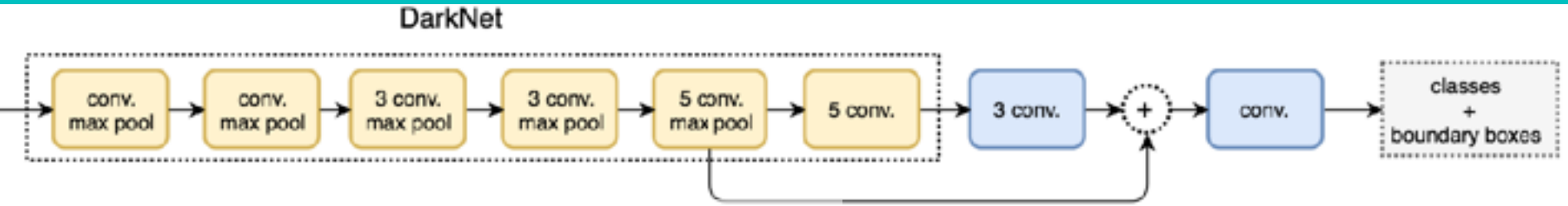


Redmon and Farhadi, YOLO9000: Better, Faster, Stronger, 2016, December 25 — *Merry Christmas?*

The YOLO Output Tensor

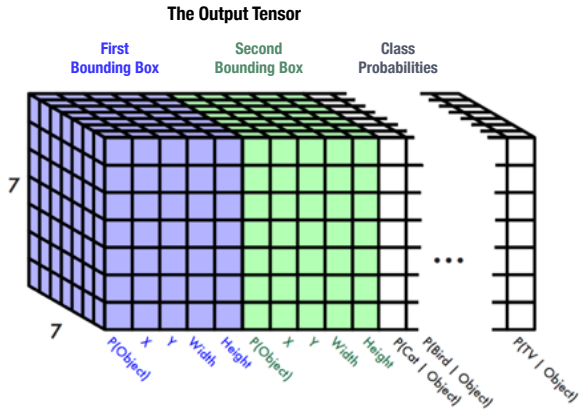


The YOLO Architecture



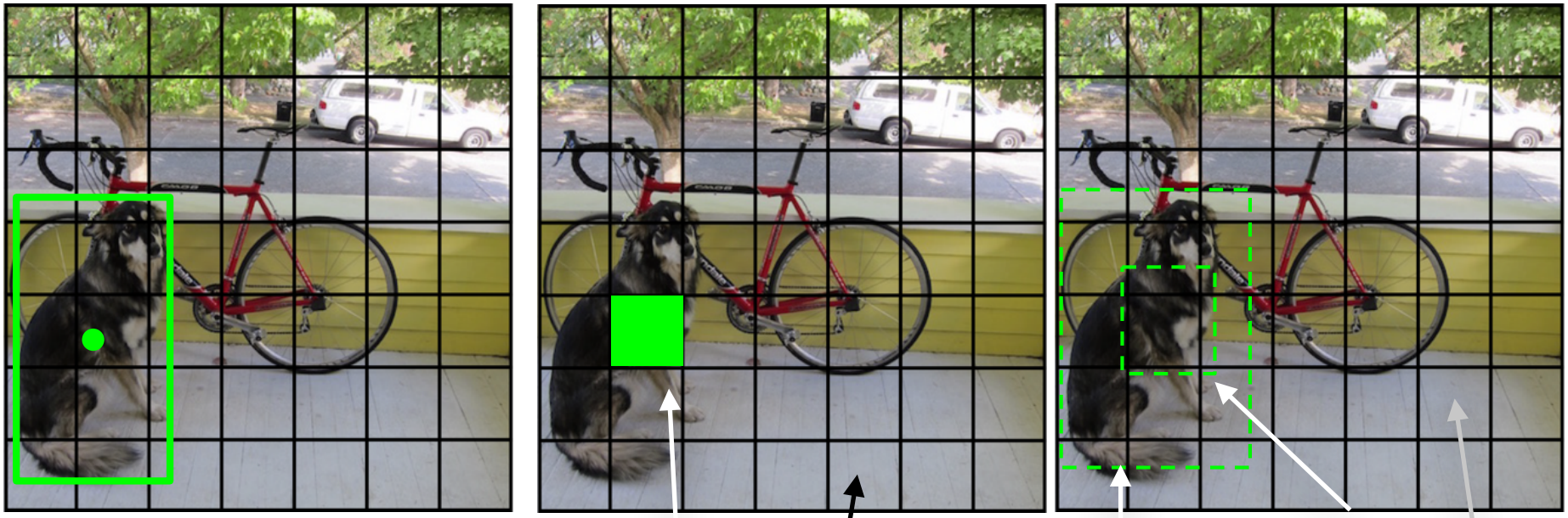
Trained from Traditional Image Dataset.
Architecture usually: DarkNet on ImageNet

Last layers:
Trained on images
with bounding boxes



https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088

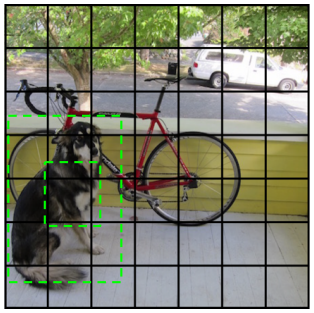
Training the YOLO Architecture



Redmon and Farhadi, YOLO9000: Better, Faster, Stronger, 2016, December 25 — *Merry Christmas?*

The YOLO Loss Function

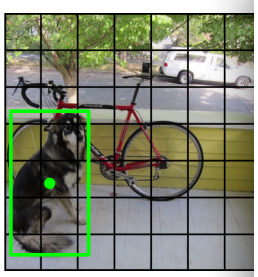
Update Bounding Box



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

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

Localization Loss



SxS cells, i^{th} cell

B boxes per cell, jth box

$\mathbb{1}^{\text{obj}}$ indicator function, from GT

$\hat{C}$ is confidence per box

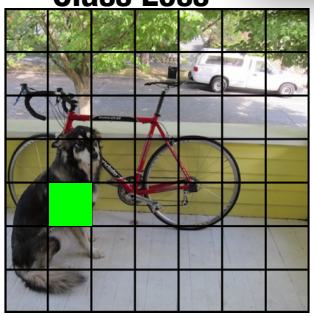
$\hat{p}(c)$ softmax output, per class

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

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

Object Detection Loss

Class Loss



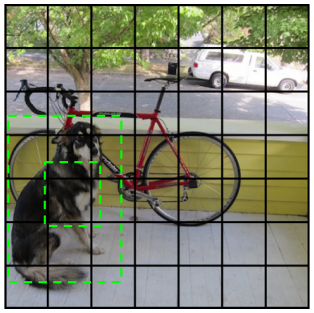
$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

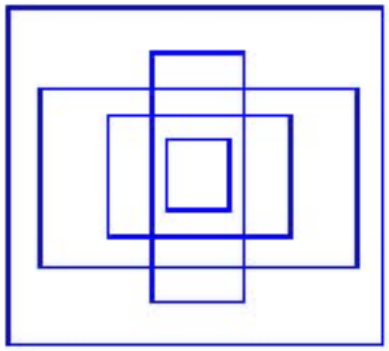
Classification Loss

https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088

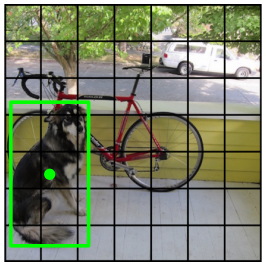
Updated YOLO Localization

Update Bounding Box

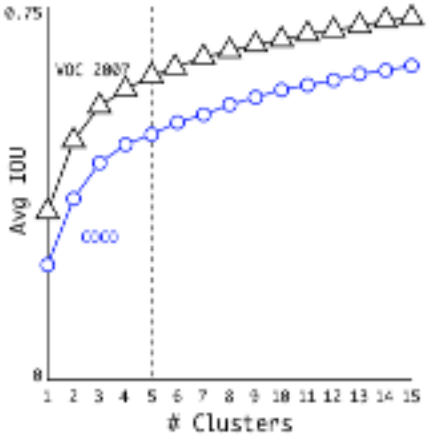


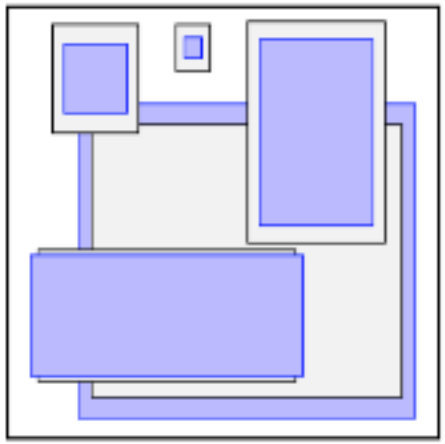


- Define 5 pre-defined box shapes (based on data)
- Regress x, y offset from cell center
- Bound x and y to the bounds of the cell
- And w and h scaling of predefined shape

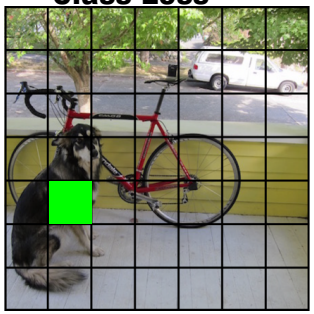


"Good" Shape Priors Found via Clustering





Class Loss



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

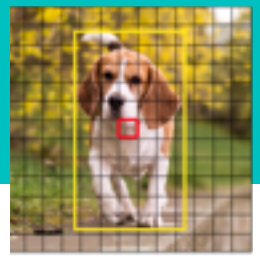
$$+\lambda_{\text{noobj}}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{\text{noobj}}\left(C_i-\hat{C}_{ij}\right)^2$$

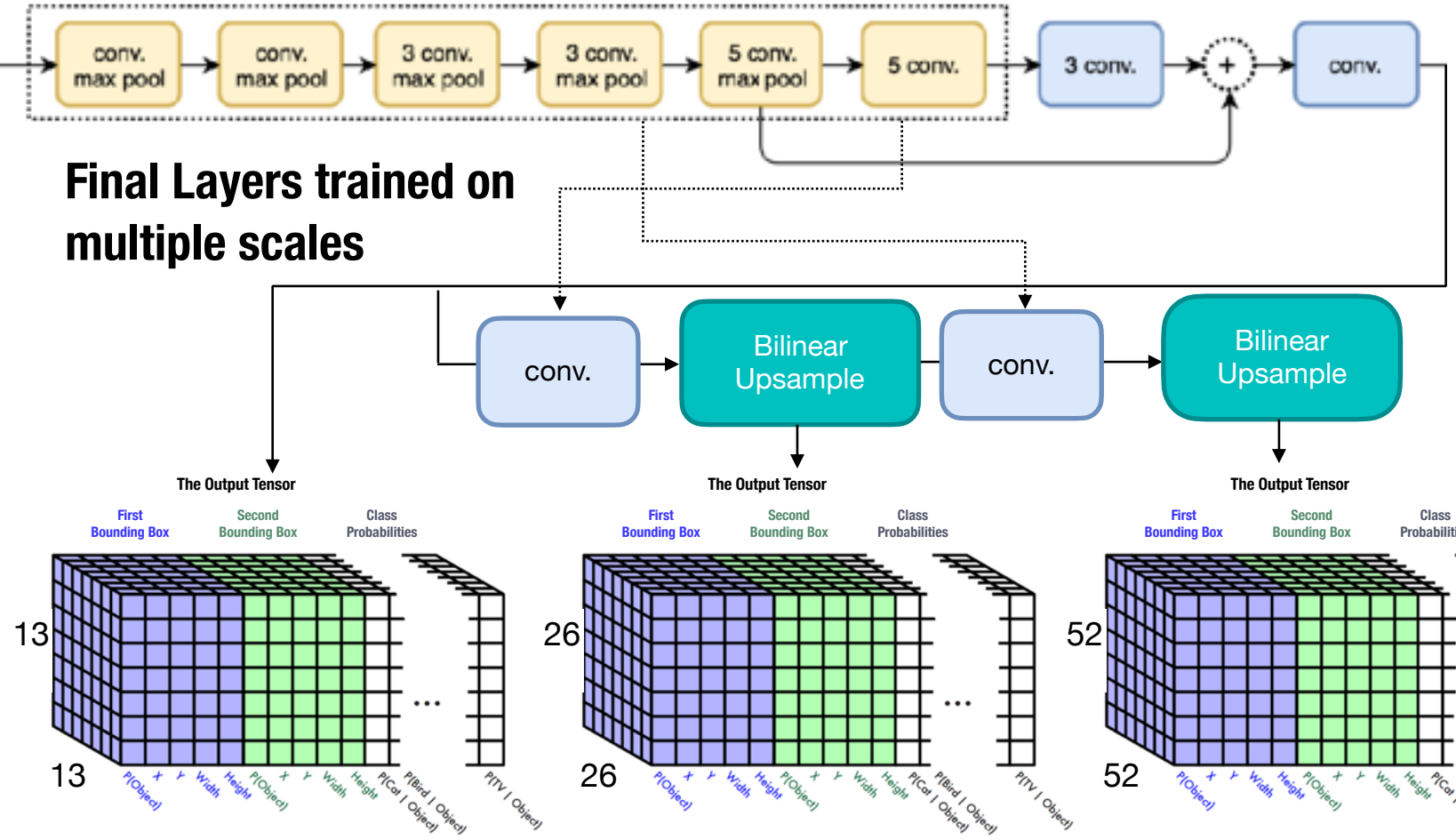
$$+\sum_{i=0}^{S^2}\mathbb{1}_{i}^{\text{obj}}\sum_{c\in\text{classes}}\left(p_i(c)-\hat{p}_i(c)\right)^2$$

Classification Loss

https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088

The YOLOv3 Architecture

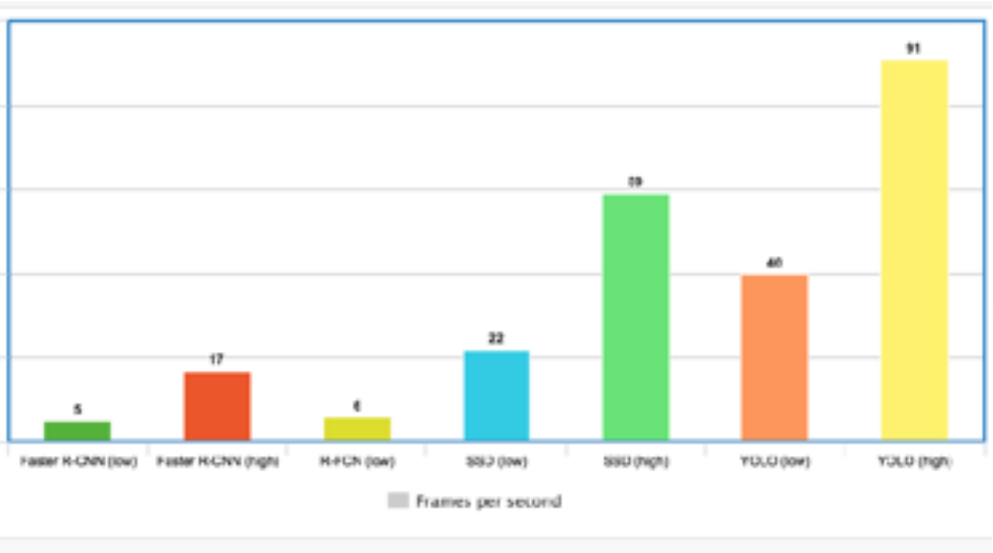




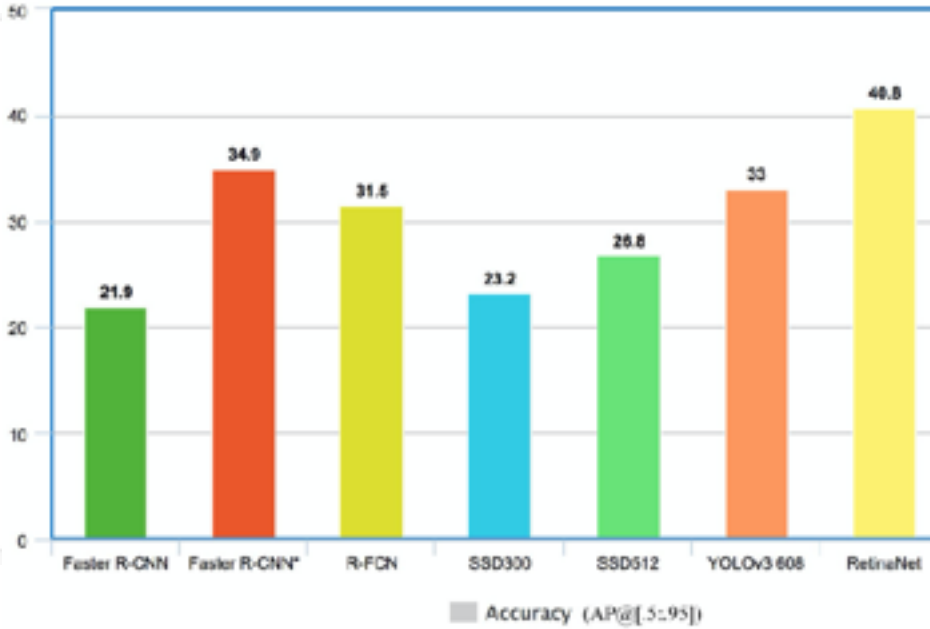
https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088

SSD, YOLO, Faster-RCNN

- Remember, this is an ongoing line of research and everybody wants to win the franchise war



Speed



"Performance"

https://medium.com/@jonathan_hui/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359

Lecture Notes for

Neural Networks and Machine Learning

FCN Learning

Next Time:
Instance Segmentation

Reading: None