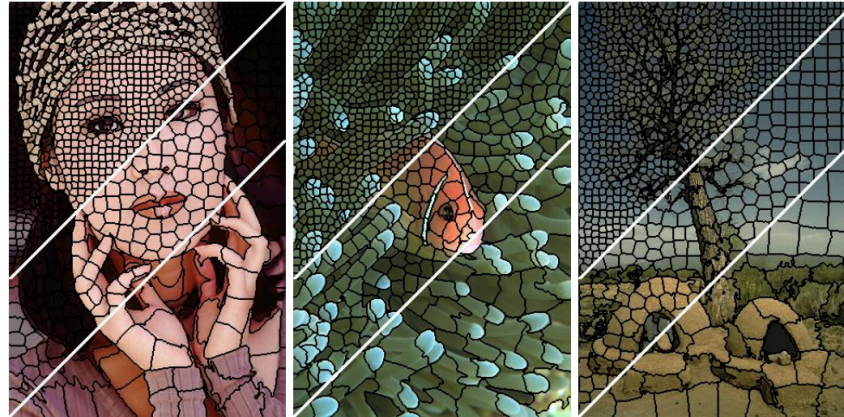


# Segmentation

---



Bottom-up Segmentation



Semantic / instance segmentation

# Outline

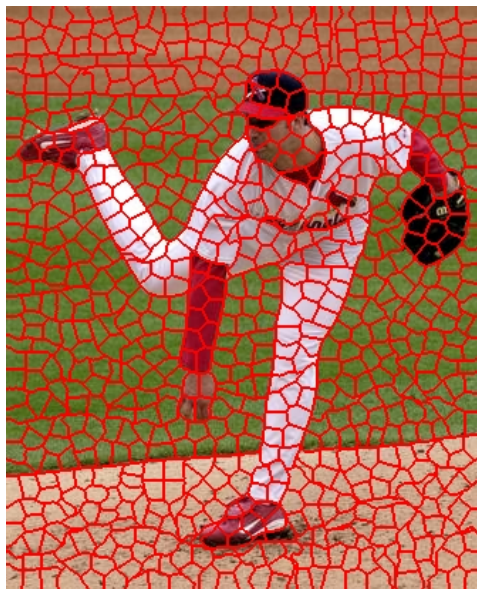
---

- Bottom-up segmentation
  - Superpixel segmentation
- Semantic segmentation
  - Metrics
  - Architectures
    - “Convolutionalization”
    - Dilated convolutions
    - Hyper-columns / skip-connections
    - Learned up-sampling architectures
- Instance segmentation
  - Metrics, RoI Align
- Other dense prediction problems

# Supervoxel segmentation

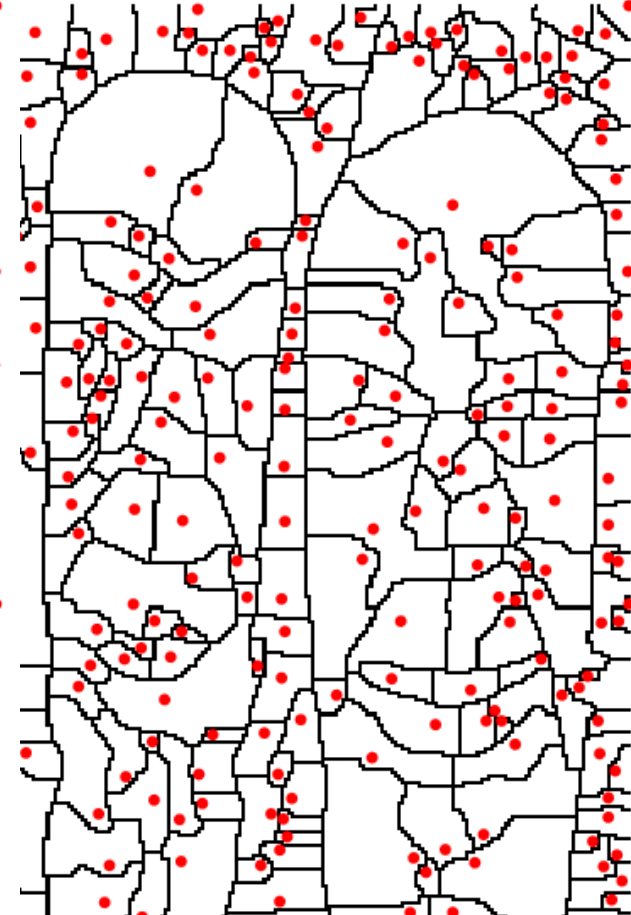
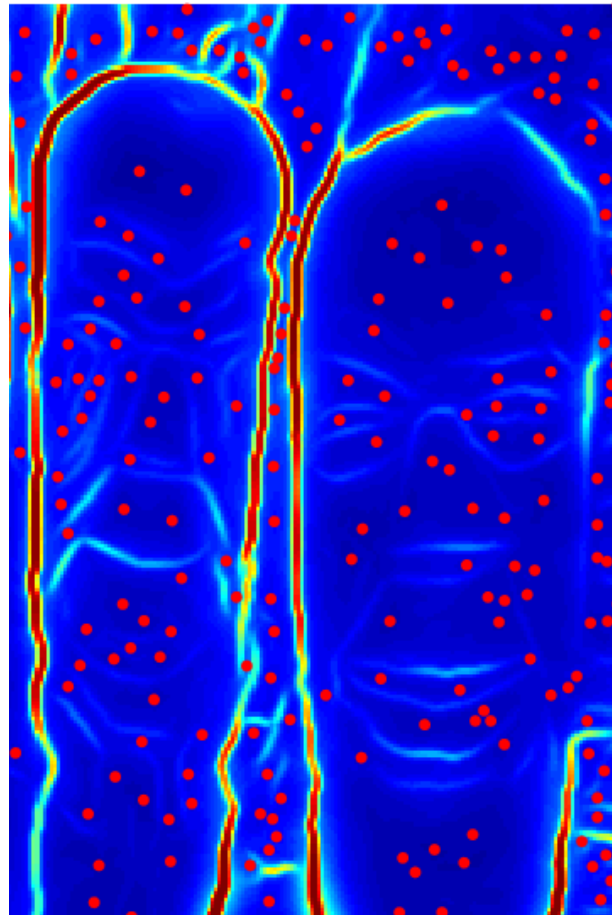
---

- Group together similar-looking pixels as an intermediate stage of processing
  - “Bottom-up” process
  - Typically unsupervised
  - Should be fast
  - Typically aims to produce an over-segmentation



# Superpixel segmentation

---



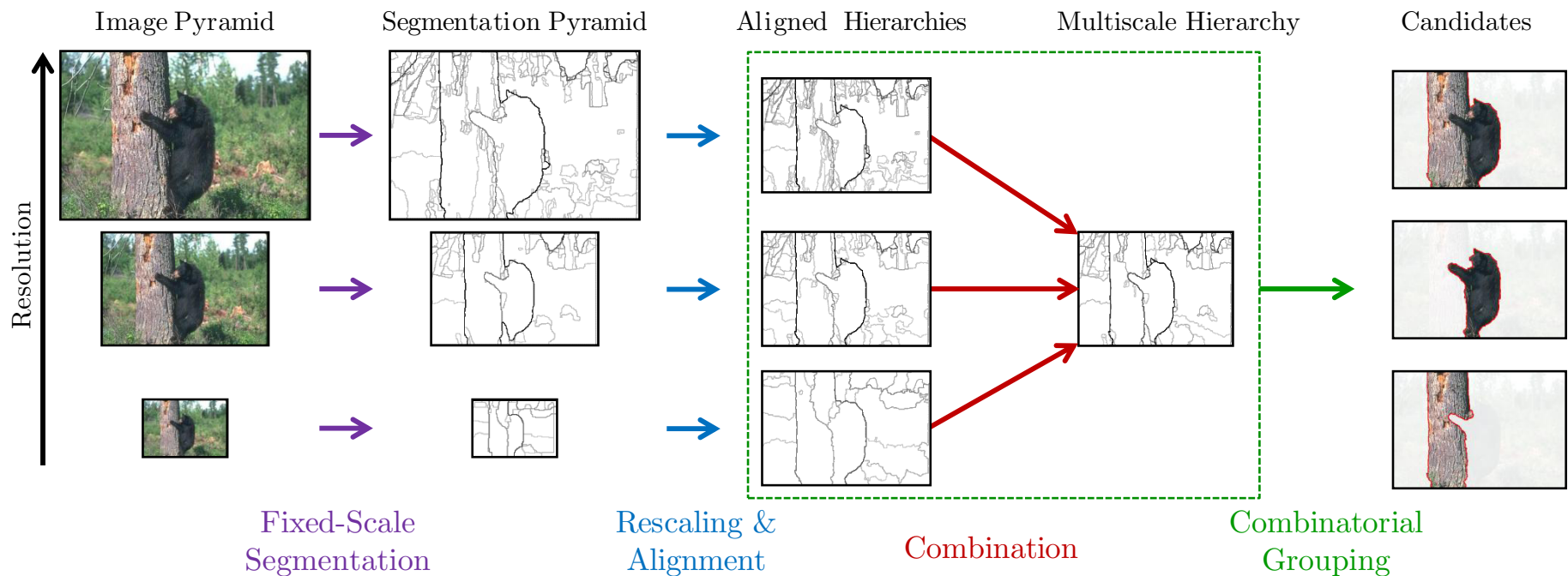
# Superpixel segmentation

---



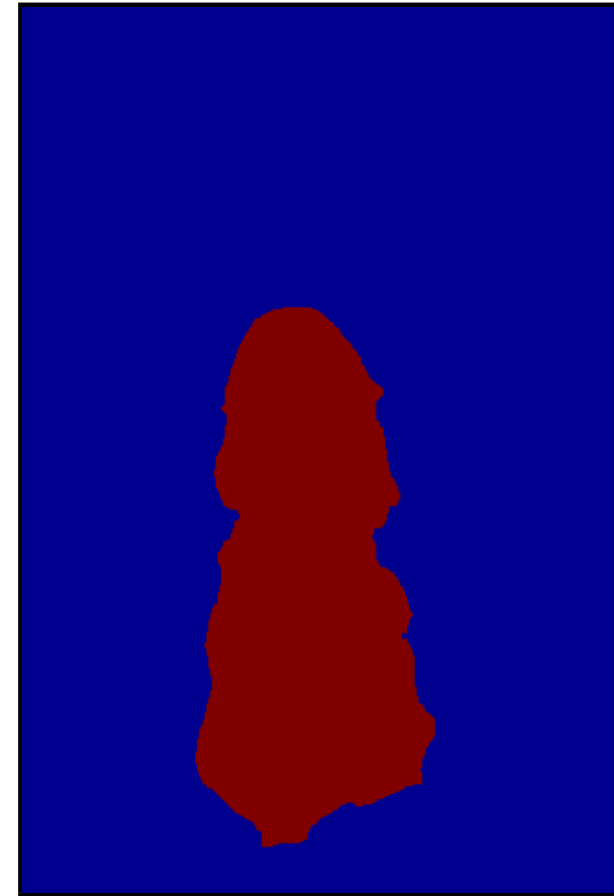
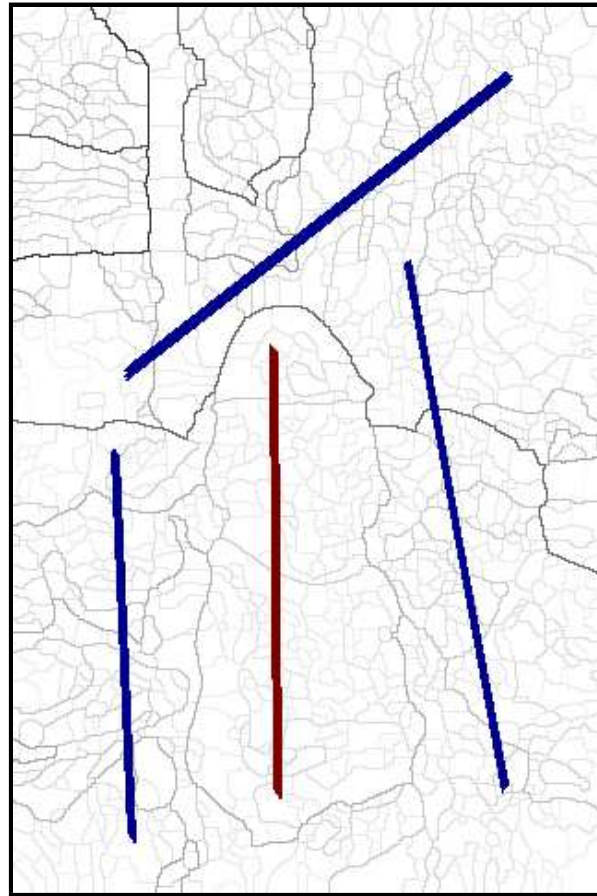
# Multiscale Combinatorial Grouping

- Use hierarchical segmentation: start with small *superpixels* and merge based on diverse cues



# Applications: Interactive Segmentation

---



# Semantic Segmentation: Metrics

---



Image



Ground Truth

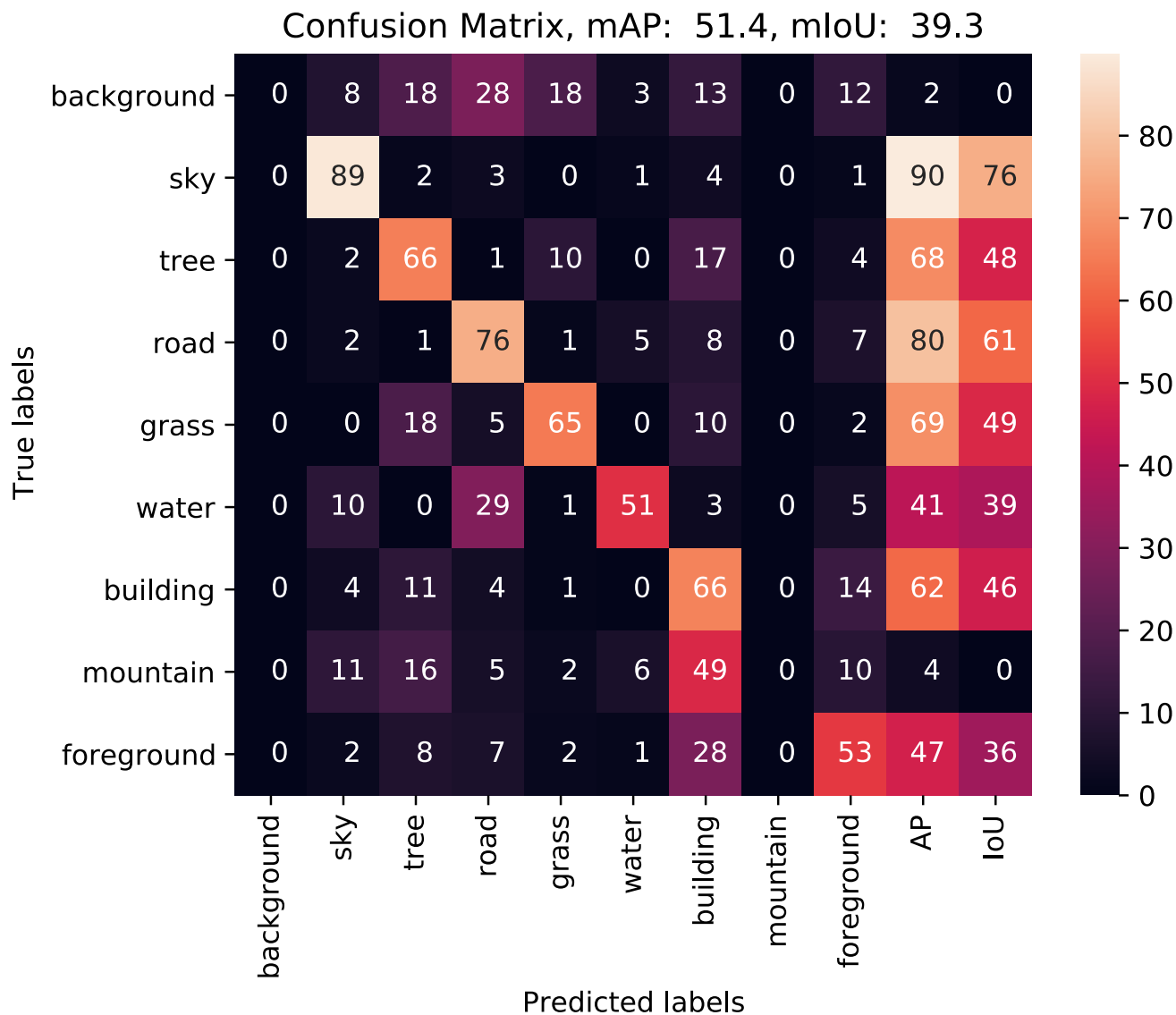


Prediction

- Pixel Classification Accuracy
- Intersection over Union
- Average Precision

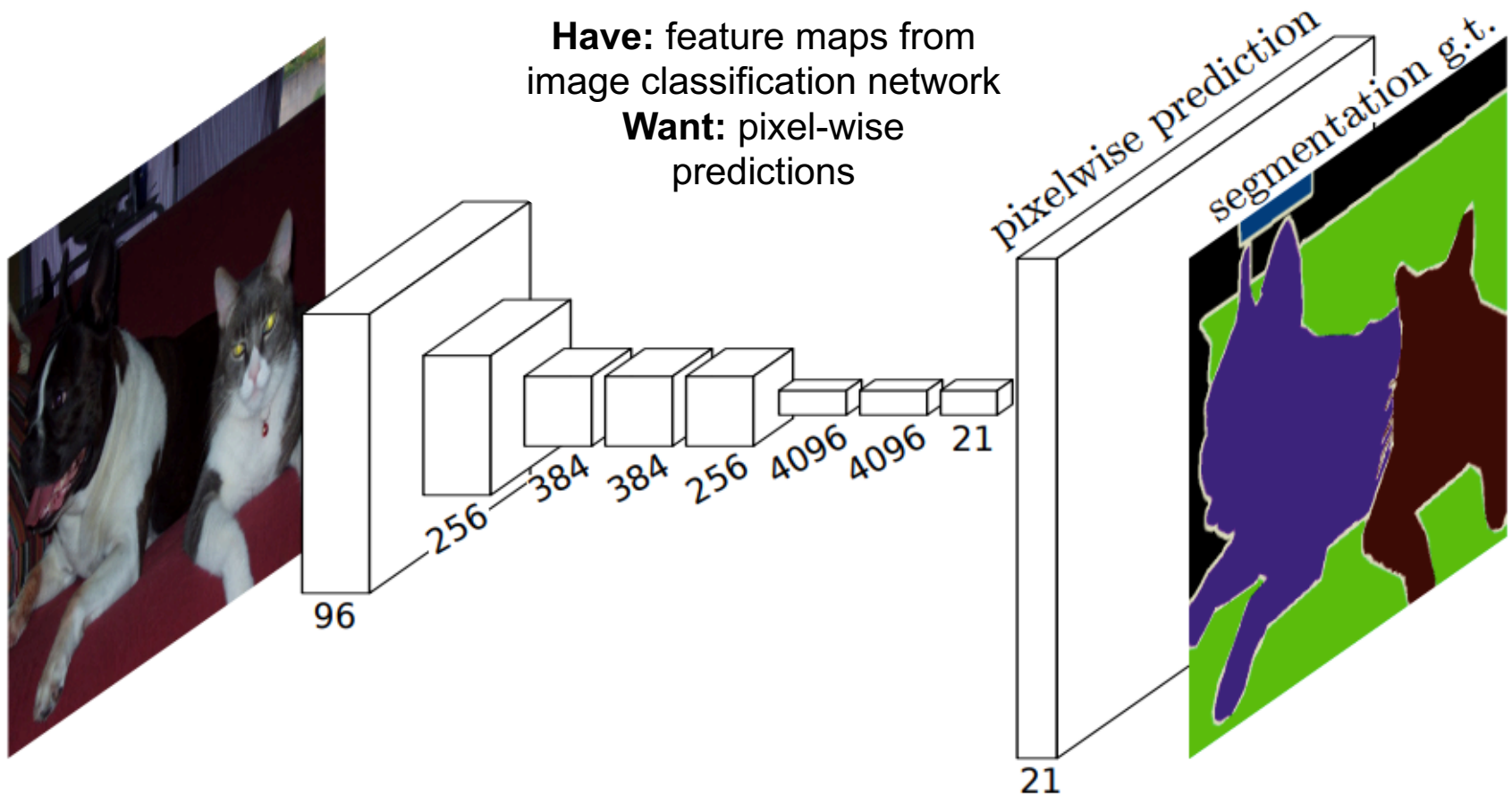


# Semantic Segmentation: Metrics



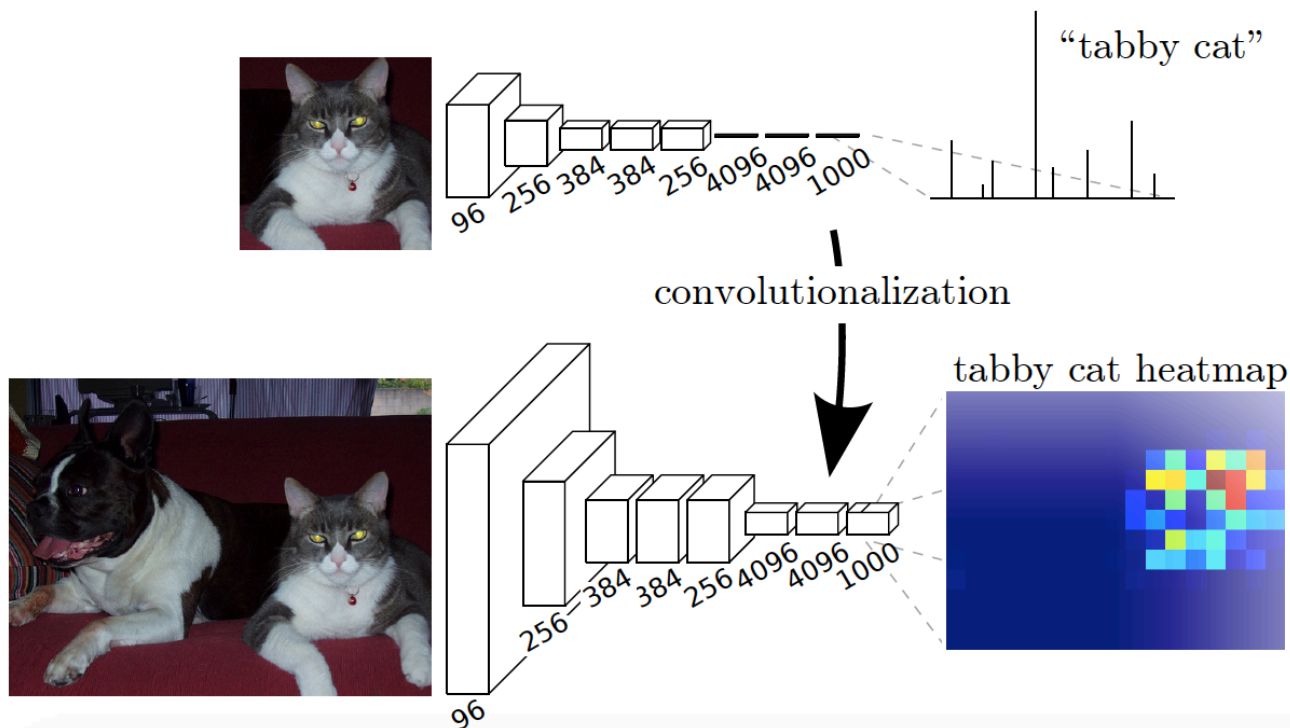
# Semantic Segmentation

- Do dense prediction as a post-process on top of an image classification CNN



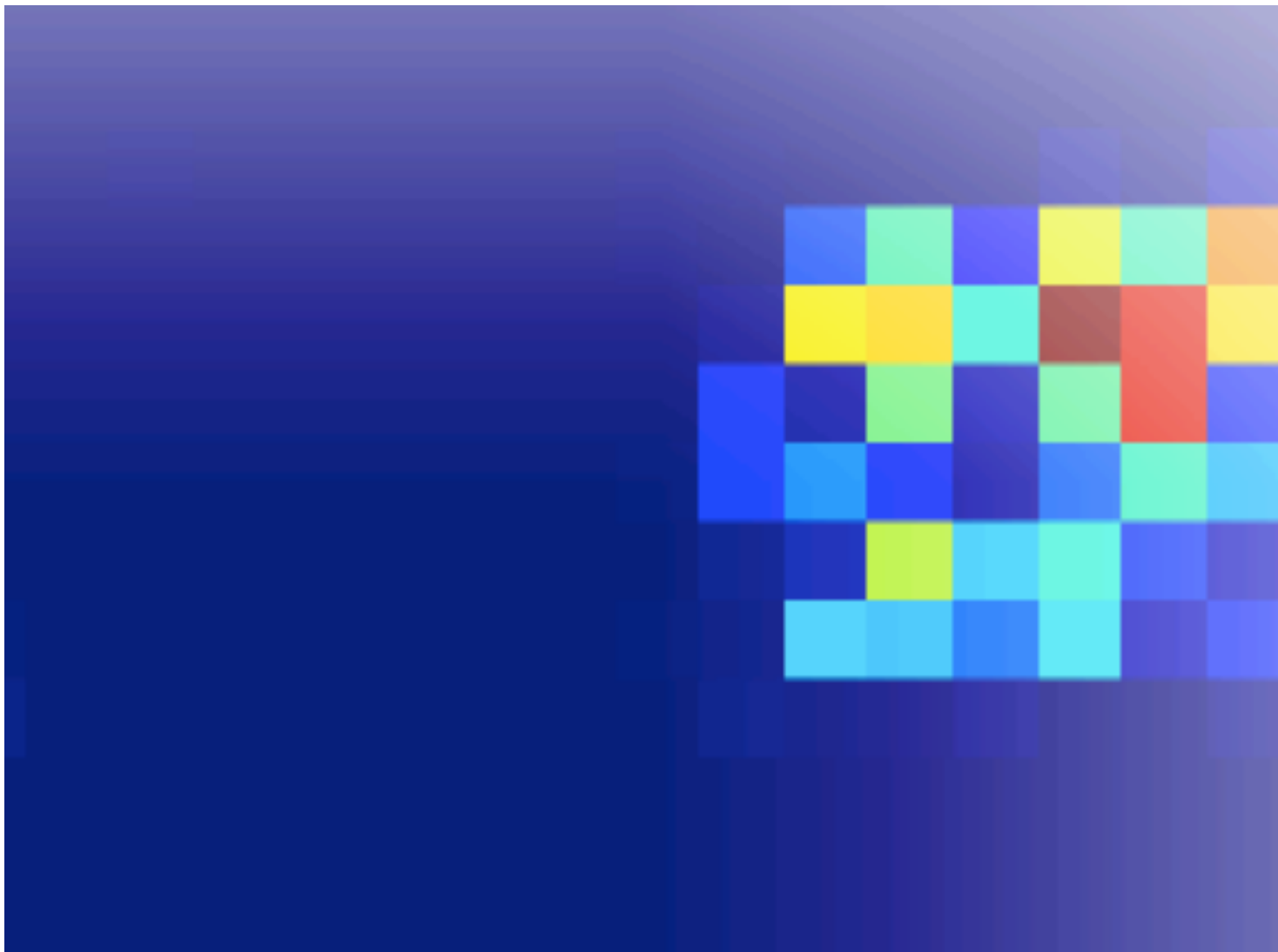
# Convolutionalization

- Design a network with only convolutional layers, make predictions for all pixels at once



# Sparse, Low-resolution Output

---



J. Long, E. Shelhamer, and T. Darrell, [Fully Convolutional Networks for Semantic Segmentation](#), CVPR 2015

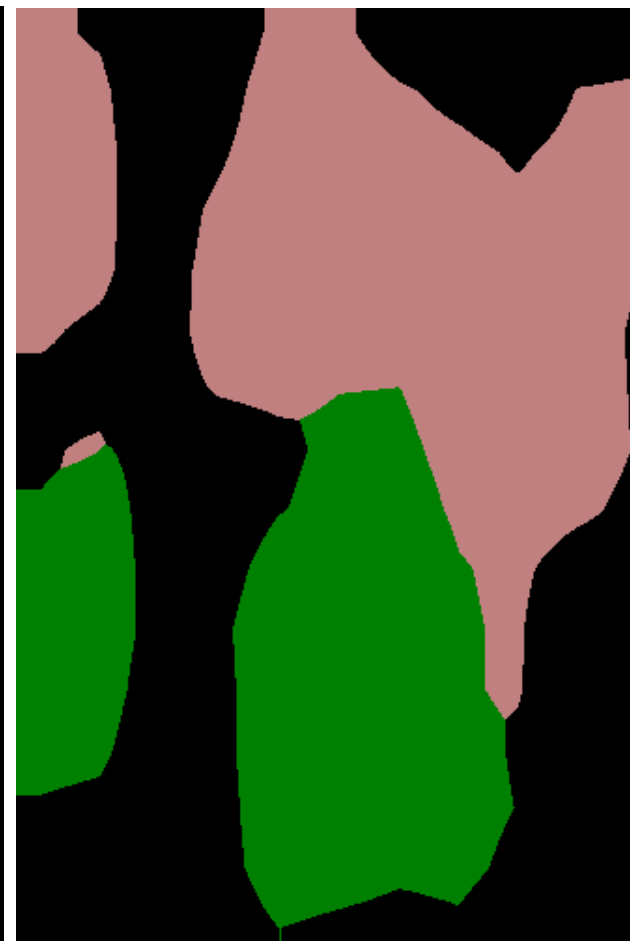
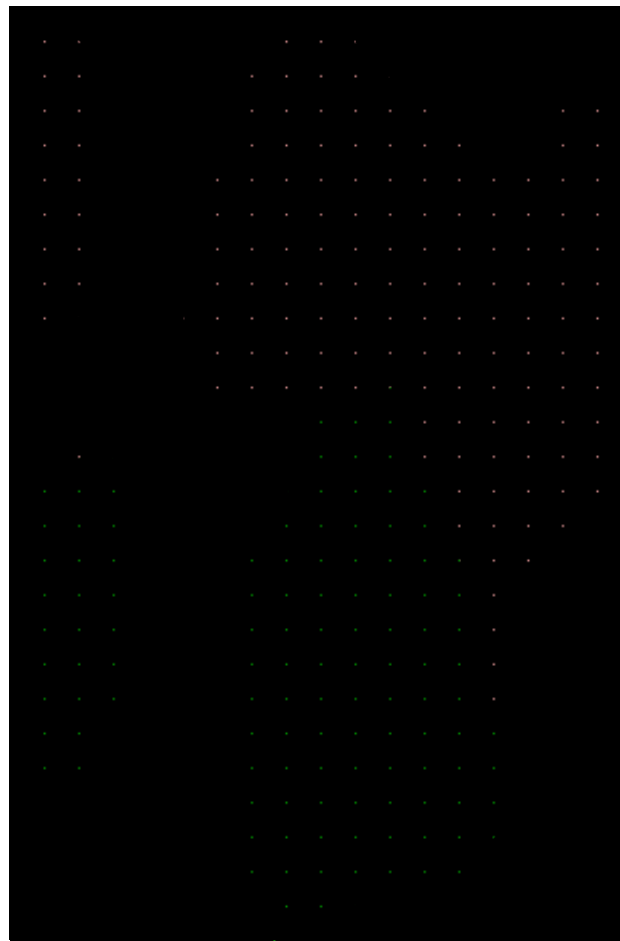
# Aside: Receptive Field, Stride

---

- Receptive Field: Pixels in the image that are “connected” to a given unit.
- Stride: Shift in receptive field between consecutive units in a convolutional feature map.
- See: <https://distill.pub/2019/computing-receptive-fields/>

# Sparse, Low-resolution Output

---

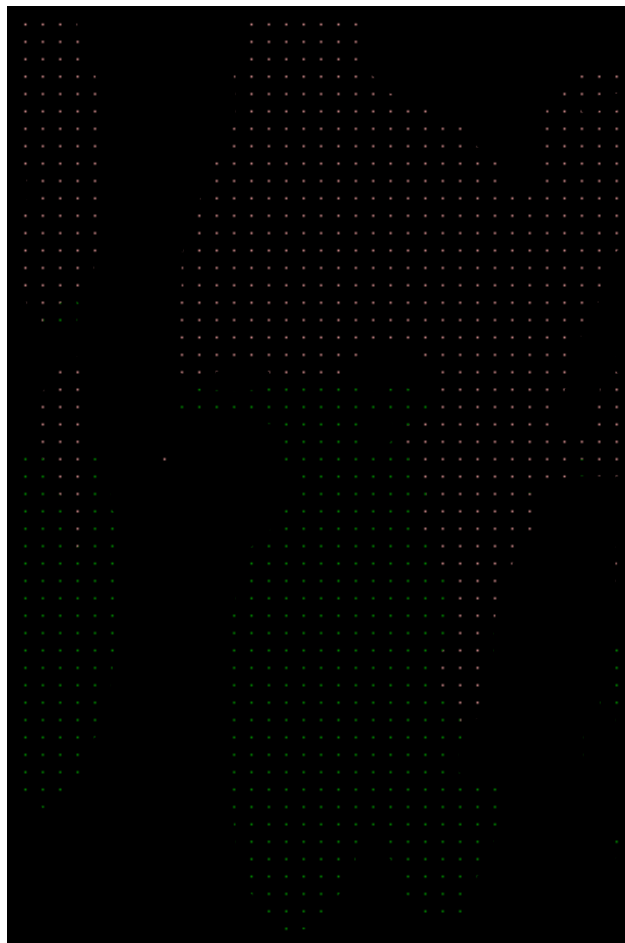


Bilinear Up sampling: Differentiable,  
train through up-sampling.

# Fix 1: Shift and Stitch

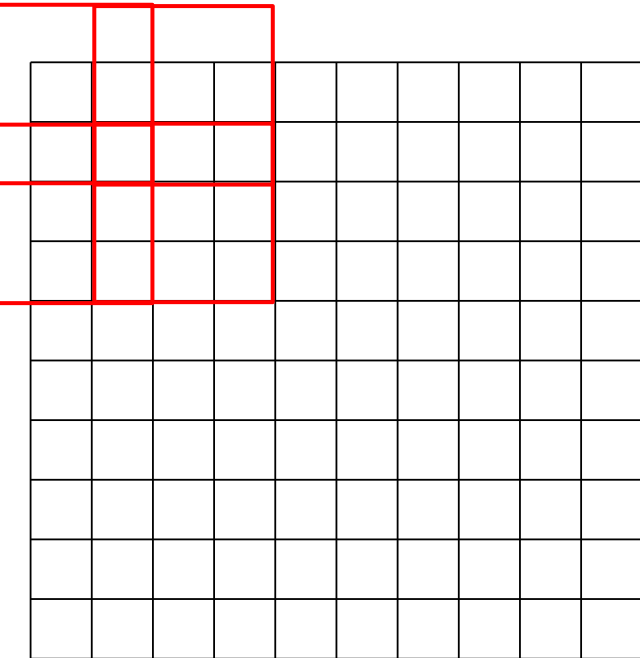
---

- Shift the image, and re-run CNN to get denser output.



# Fix 1: A trous Conv., Dilated Conv.

A. 3x3 conv  
stride 2



B. 3x3 conv, stride1

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

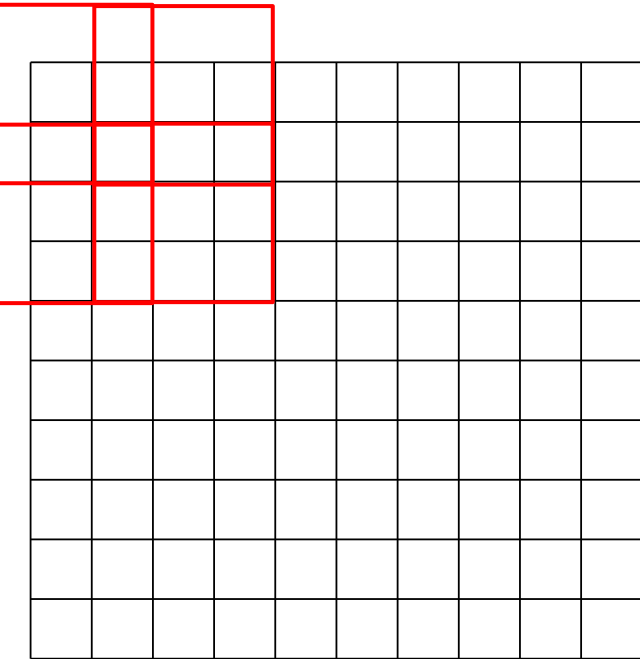
1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

1	1	6	6	11	11	16	16	21	21
1	1	6	6	11	11	16	16	21	21
2	2	7	7	12	12	17	17	22	22
2	2	7	7	12	12	17	17	22	22
3	3	8	8	13	13	18	18	23	23
3	3	8	8	13	13	18	18	23	23
4	4	9	9	14	14	19	19	24	24
4	4	9	9	14	14	19	19	24	24
5	5	10	10	15	15	20	20	25	25
5	5	10	10	15	15	20	20	25	25



# Fix 1: A trous Conv., Dilated Conv.

A. 3x3 conv  
stride 1



B. 3x3 conv, stride1,  
dilation 2

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

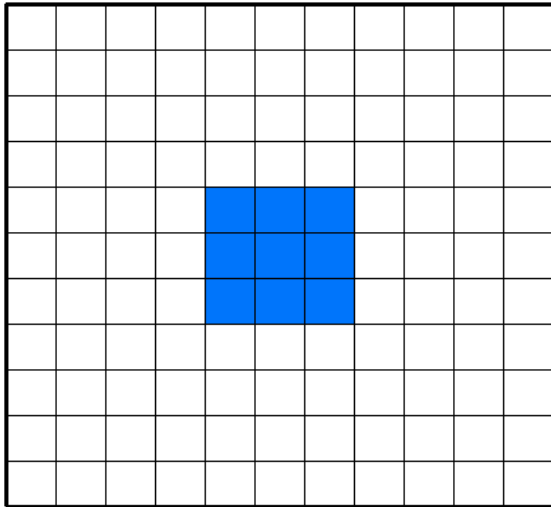
1	1	6	6	11	11	16	16	21	21
1	1	6	6	11	11	16	16	21	21
2	2	7		12		17	17	22	22
2	2						17	22	22
3	3	8		13		18	18	23	23
3	3						18	23	23
4	4	9		14		19	19	24	24
4	4	9	9	14	14	19	19	24	24
5	5	10	10	15	15	20	20	25	25
5	5	10	10	15	15	20	20	25	25

1	1	6	6	11	11	16	16	21	21
1	1	6	6	11	11	16	16	21	21
2	2	7	7	12	12	17	17	22	22
2	2	7	7	12	12	17	17	22	22
3	3	8	8	13	13	18	18	23	23
3	3	8	8	13	13	18	18	23	23
4	4	9	9	14	14	19	19	24	24
4	4	9	9	14	14	19	19	24	24
5	5	10	10	15	15	20	20	25	25
5	5	10	10	15	15	20	20	25	25

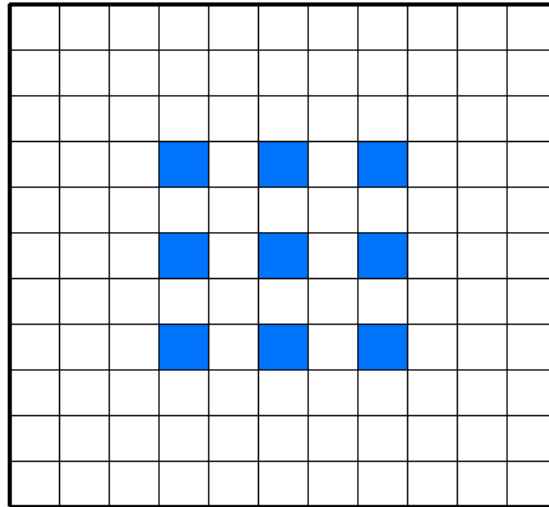
# Fix 1: A trous Conv., Dilated Conv.

---

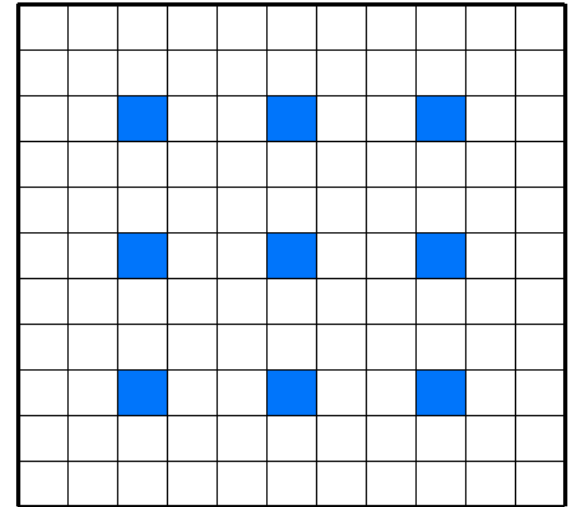
Dilation factor 1



Dilation factor 2



Dilation factor 3



# Fix 1: A trous Conv., Dilated Conv.

---

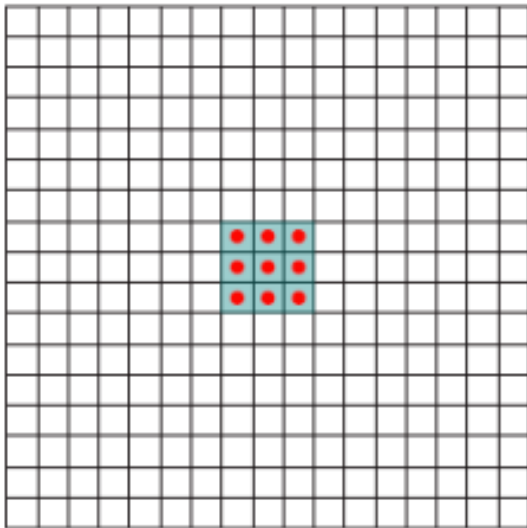
- Use in FCN to remove downsampling:  
change stride of max pooling layer from 2 to 1,  
dilate subsequent convolutions by factor of 2  
(possibly without re-training any parameters)
- Instead of reducing spatial resolution of feature maps, use a large sparse filter

# Fix 1: A trous Conv., Dilated Conv.

---

- Can increase receptive field size exponentially with a linear growth in the number of parameters

Feature map 1 (F1)  
produced from F0 by  
1-dilated convolution



Receptive field: 3x3

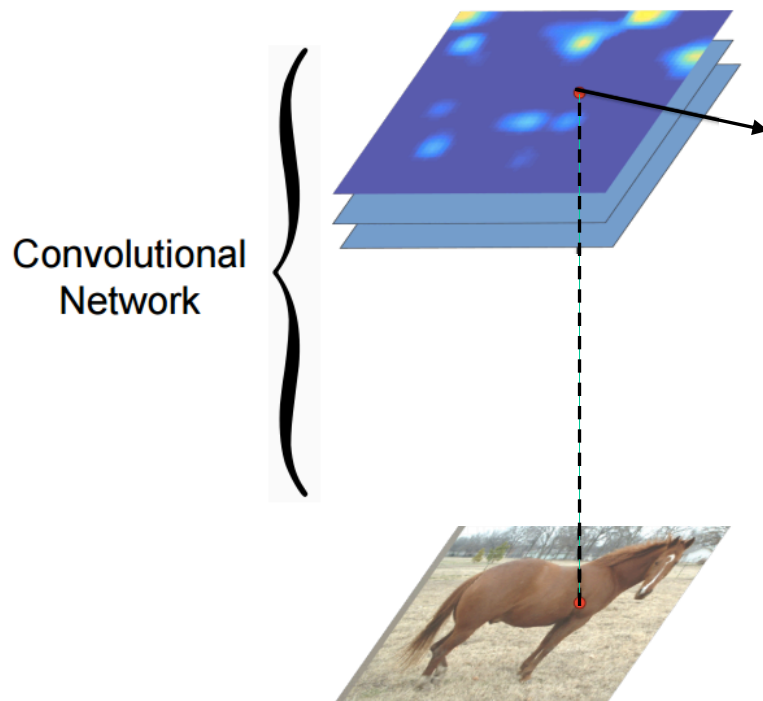
Receptive field: 7x7

Receptive field: 15x15

# Fix 2: Hyper-columns/Skip Connections

---

- Even though with dilation we can predict each pixel, fine-grained information needs to be propagated through the network.
- Idea: Additionally use features from within the network.

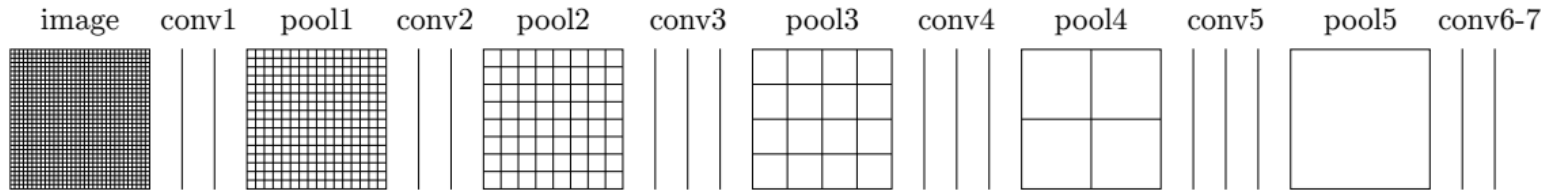


B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik, [Hypercolumns for Object Segmentation and Fine-grained Localization](#), CVPR 2015

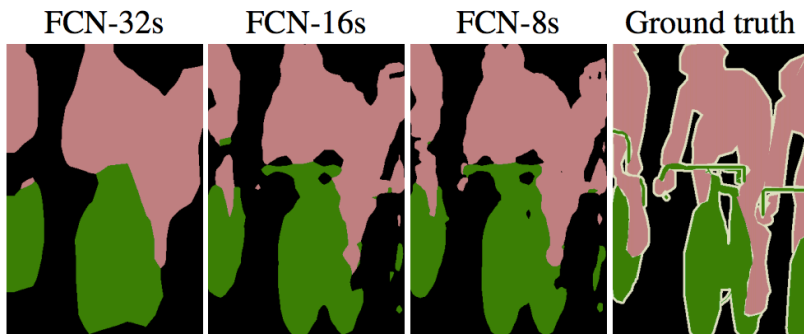
J. Long, et al., [Fully Convolutional Networks for Semantic Segmentation](#), CVPR 2015

# Fix 2: Hyper-columns/Skip Connections

---



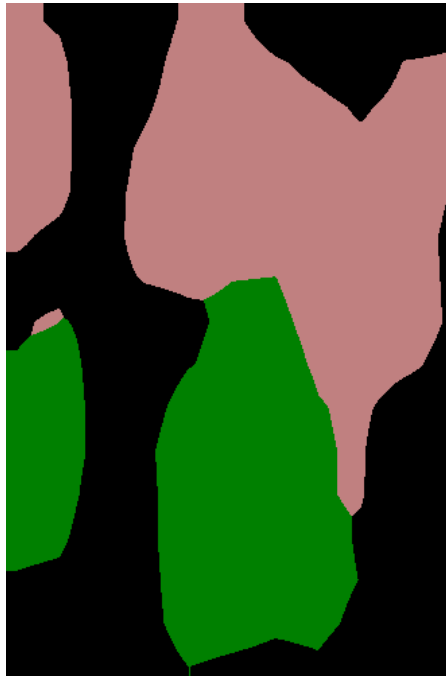
- Predictions by 1x1 conv layers, bilinear upsampling
- Predictions by 1x1 conv layers, learned 2x upsampling, fusion by summing



# Fix 2: Hyper-columns/Skip Connections

---

FCN-32s



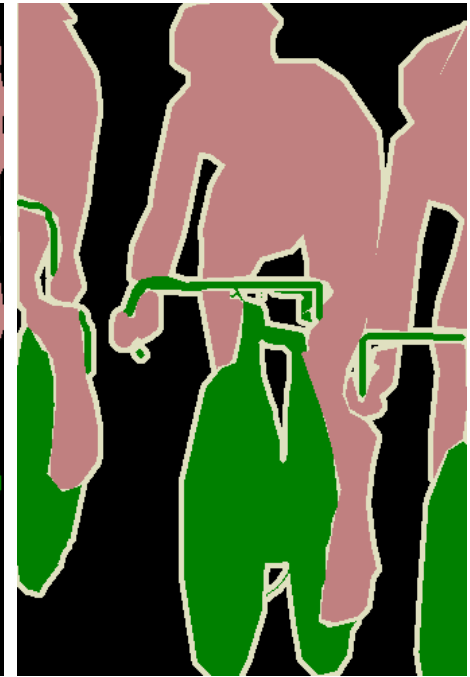
FCN-16s



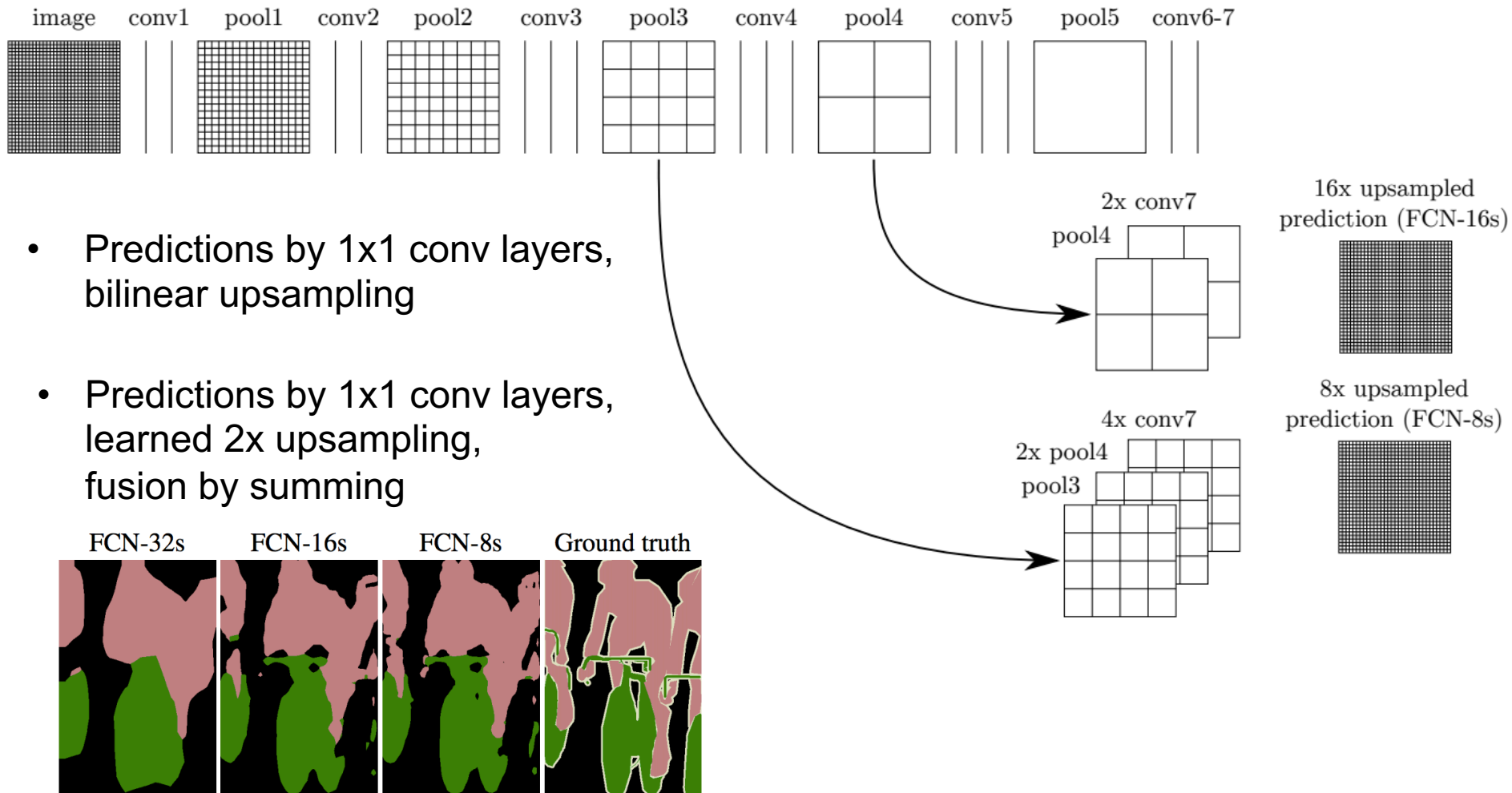
FCN-8s



Ground truth



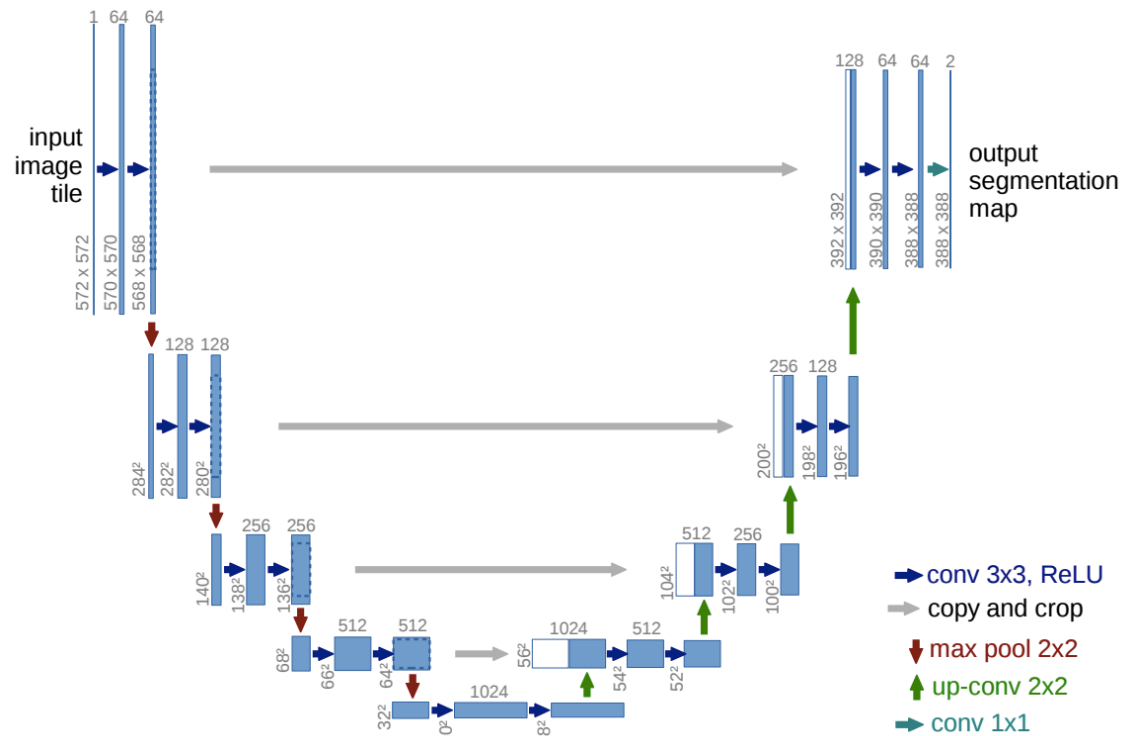
# Fix 2b: Learned Upsampling





# U-Net

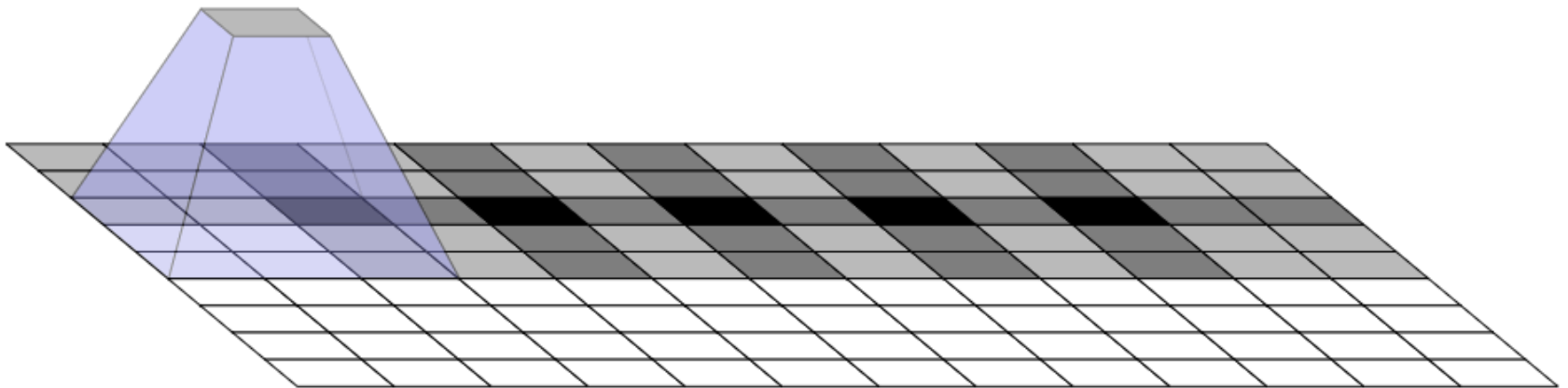
- Like FCN, fuse upsampled higher-level feature maps with higher-res, lower-level feature maps
- Unlike FCN, fuse by concatenation, predict at the end



# Up-convolution

---

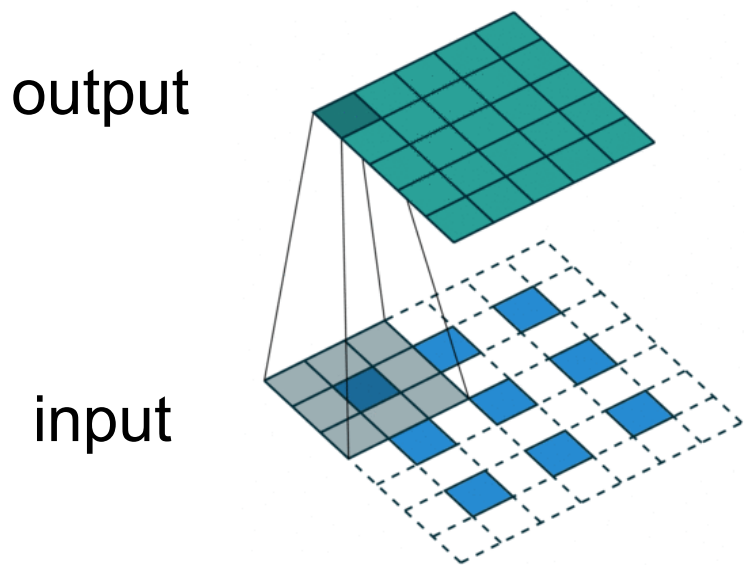
- “Paint” in the output feature map with the learned filter
  - Multiply input value by filter, place result in the output, sum overlapping values



Animation: <https://distill.pub/2016/deconv-checkerboard/>

# Up-convolution: Alternate view

- 2D case: for stride 2, dilate the input by inserting rows and columns of zeros between adjacent entries, convolve with flipped filter
- Sometimes called convolution with *fractional input stride 1/2*



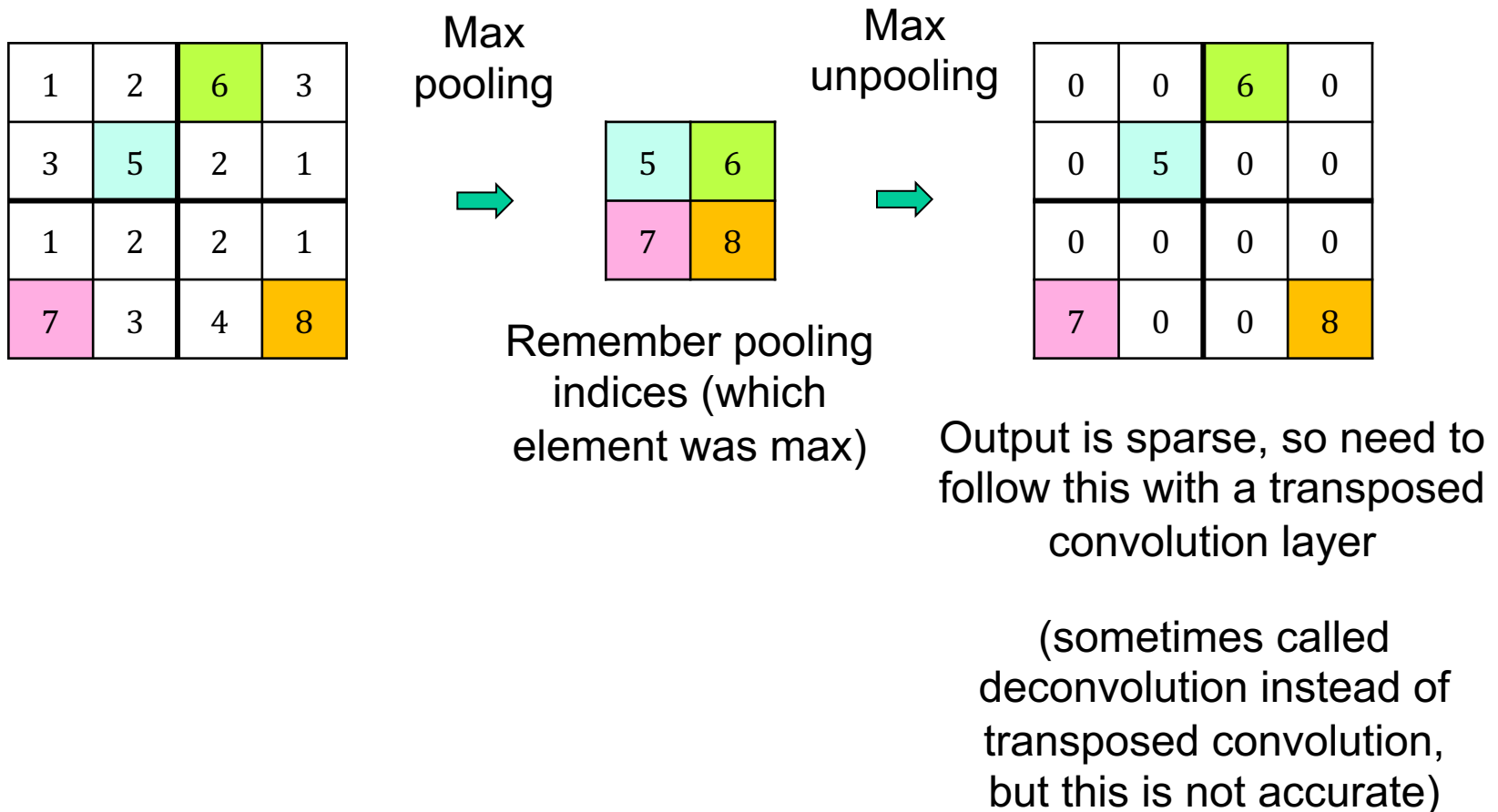
Q: What 3x3 filter would correspond to bilinear upsampling?

$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{4}$
$\frac{1}{2}$	1	$\frac{1}{2}$
$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{4}$

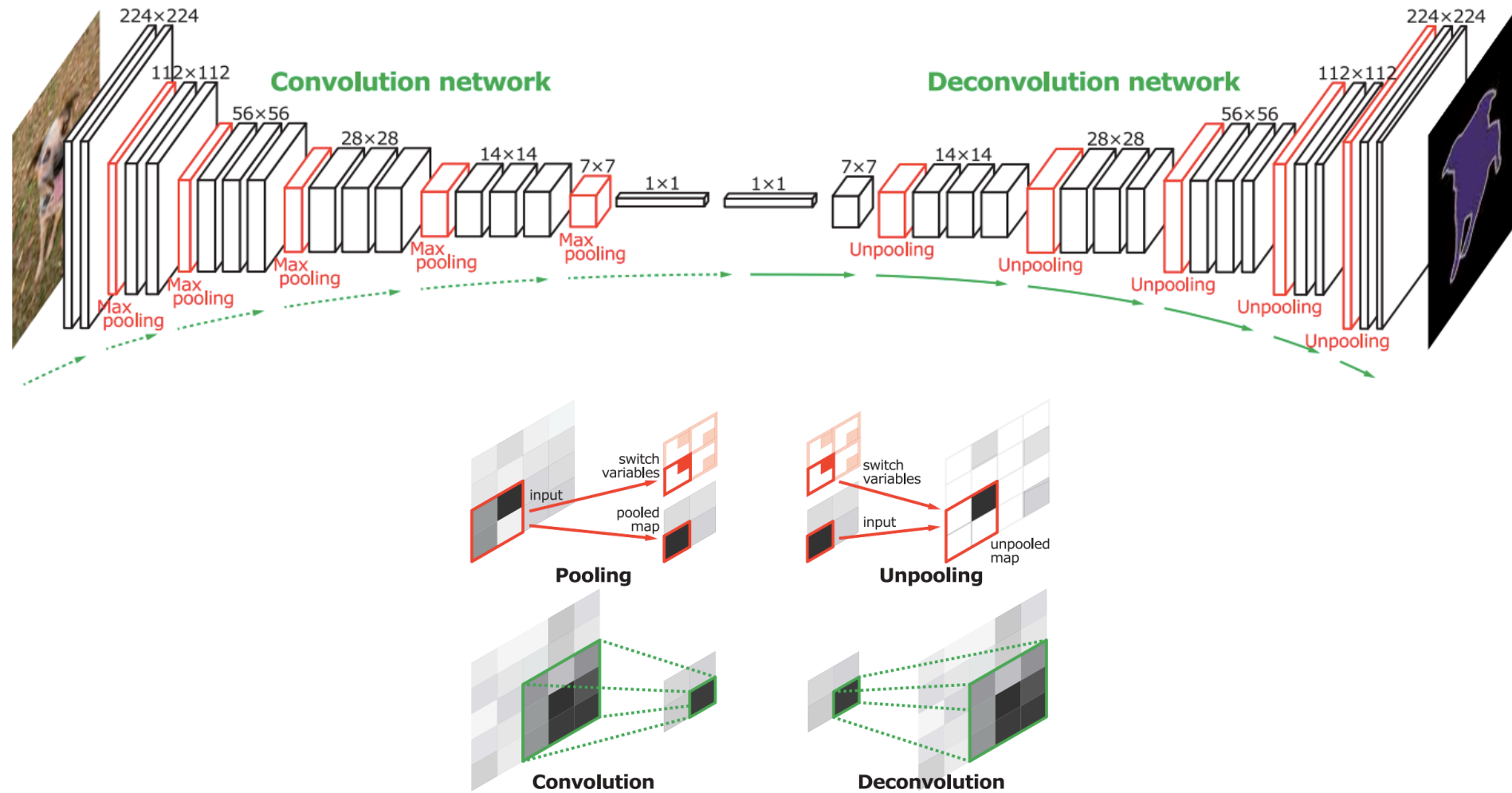
# Upsampling in a deep network

---

- Alternative to transposed convolution:  
max unpooling



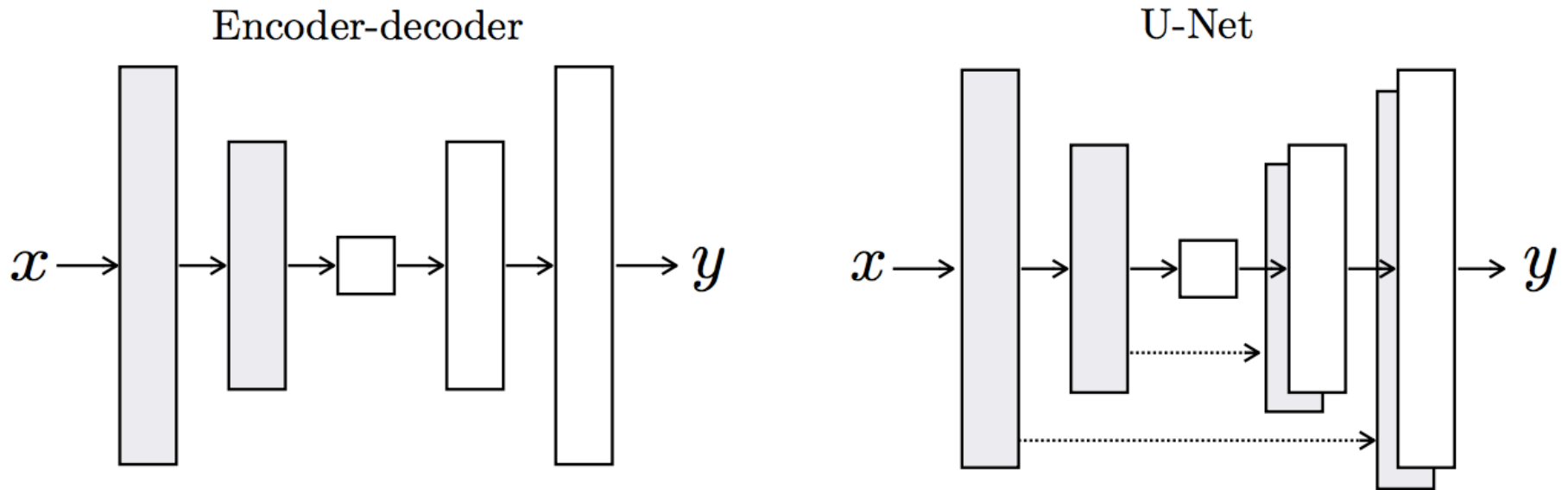
# DeconvNet



H. Noh, S. Hong, and B. Han, [Learning Deconvolution Network for Semantic Segmentation](#), ICCV 2015

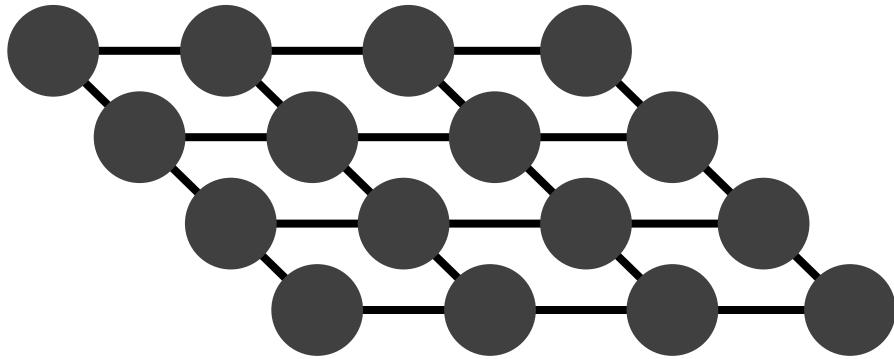
# Summary of upsampling architectures

---



# Fix 3: Use local edge information (CRFs)

---



$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z} e^{-E(\mathbf{y}, \mathbf{x})}$$

$$\mathbf{y}^* = \arg \max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x})$$

$$= \arg \min_{\mathbf{y}} E(\mathbf{y}, \mathbf{x})$$

$$E(\mathbf{y}, \mathbf{x}) = \sum_i E_{data}(y_i, \mathbf{x}) + \sum_{i,j \in \mathcal{N}} E_{smooth}(y_i, y_j, \mathbf{x})$$

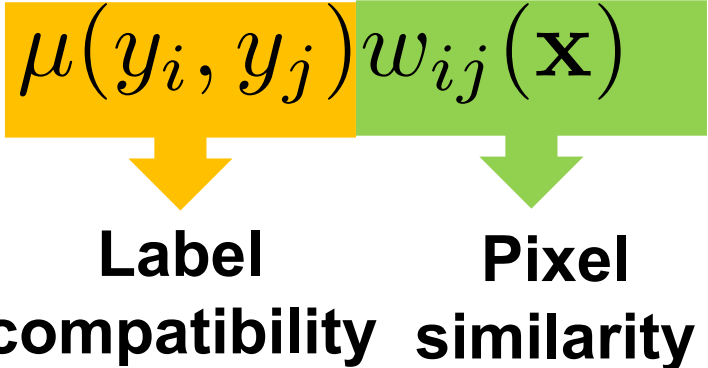
# Fix 3: Use local edge information (CRFs)

---

Idea: take convolutional network prediction and sharpen using classic techniques

*Conditional Random Field*

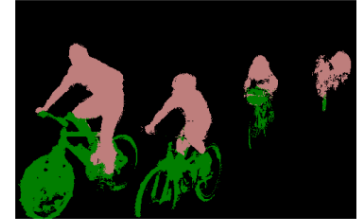
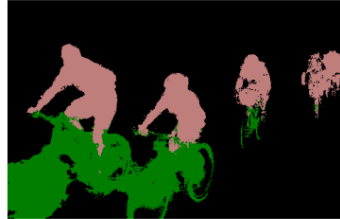
$$\mathbf{y}^* = \arg \min_{\mathbf{y}} \sum_i E_{data}(y_i, \mathbf{x}) + \sum_{i,j \in \mathcal{N}} E_{smooth}(y_i, y_j, \mathbf{x})$$

$$E_{smooth}(y_i, y_j, \mathbf{x}) = \underbrace{\mu(y_i, y_j)}_{\text{Label compatibility}} \underbrace{w_{ij}(\mathbf{x})}_{\text{Pixel similarity}}$$
The diagram shows the equation  $E_{smooth}(y_i, y_j, \mathbf{x}) = \mu(y_i, y_j) w_{ij}(\mathbf{x})$ . The term  $\mu(y_i, y_j)$  is highlighted in a yellow box, and  $w_{ij}(\mathbf{x})$  is highlighted in a green box. Below the yellow box is a yellow arrow pointing to the text "Label compatibility". Below the green box is a green arrow pointing to the text "Pixel similarity".



# Fix 3: Use local edge information (CRFs)

---



Image

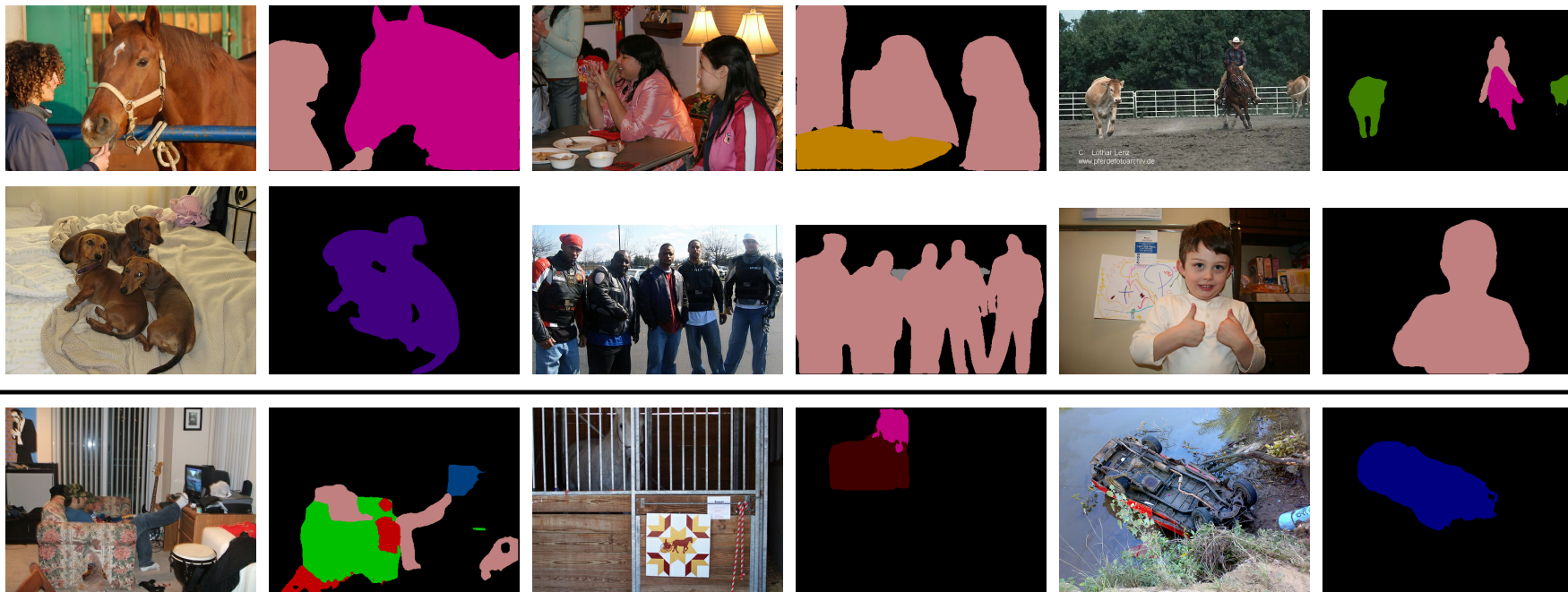
VGG-16 Bef.

VGG-16 Aft.

ResNet Bef.

ResNet Aft.

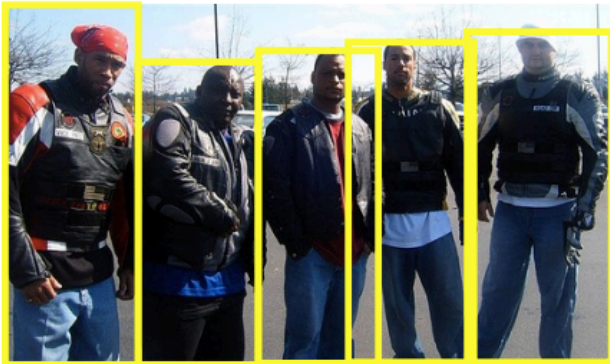
# Semantic Segmentation Results



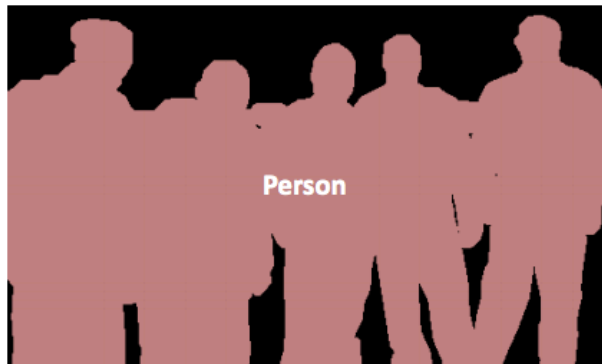
Method	mIOU
Deep Layer Cascade (LC) [82]	82.7
TuSimple [77]	83.1
Large_Kernel_Matters [60]	83.6
Multipath-RefineNet [58]	84.2
ResNet-38_MS_COCO [83]	84.9
PSPNet [24]	85.4
IDW-CNN [84]	86.3
CASIA_IVA_SDN [63]	86.6
DIS [85]	86.8
DeepLabv3 [23]	85.7
DeepLabv3-JFT [23]	86.9
DeepLabv3+ (Xception)	87.8
DeepLabv3+ (Xception-JFT)	89.0

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, Hartwig Adam, [DeepLabv3+: Encoder-Decoder with Atrous Separable Convolution](#), ECCV 2018

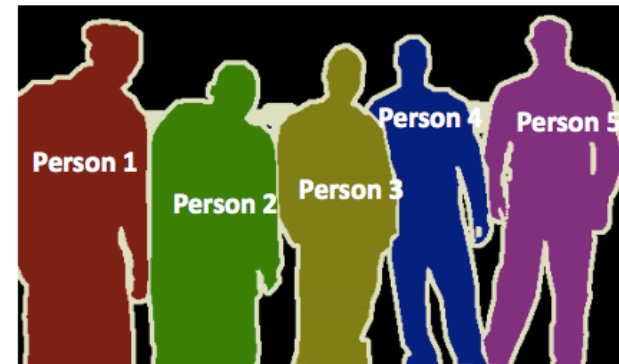
# Instance segmentation



Object Detection



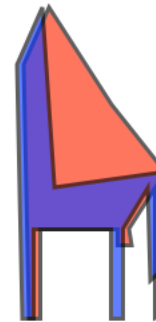
Semantic Segmentation



Instance Segmentation

## Evaluation

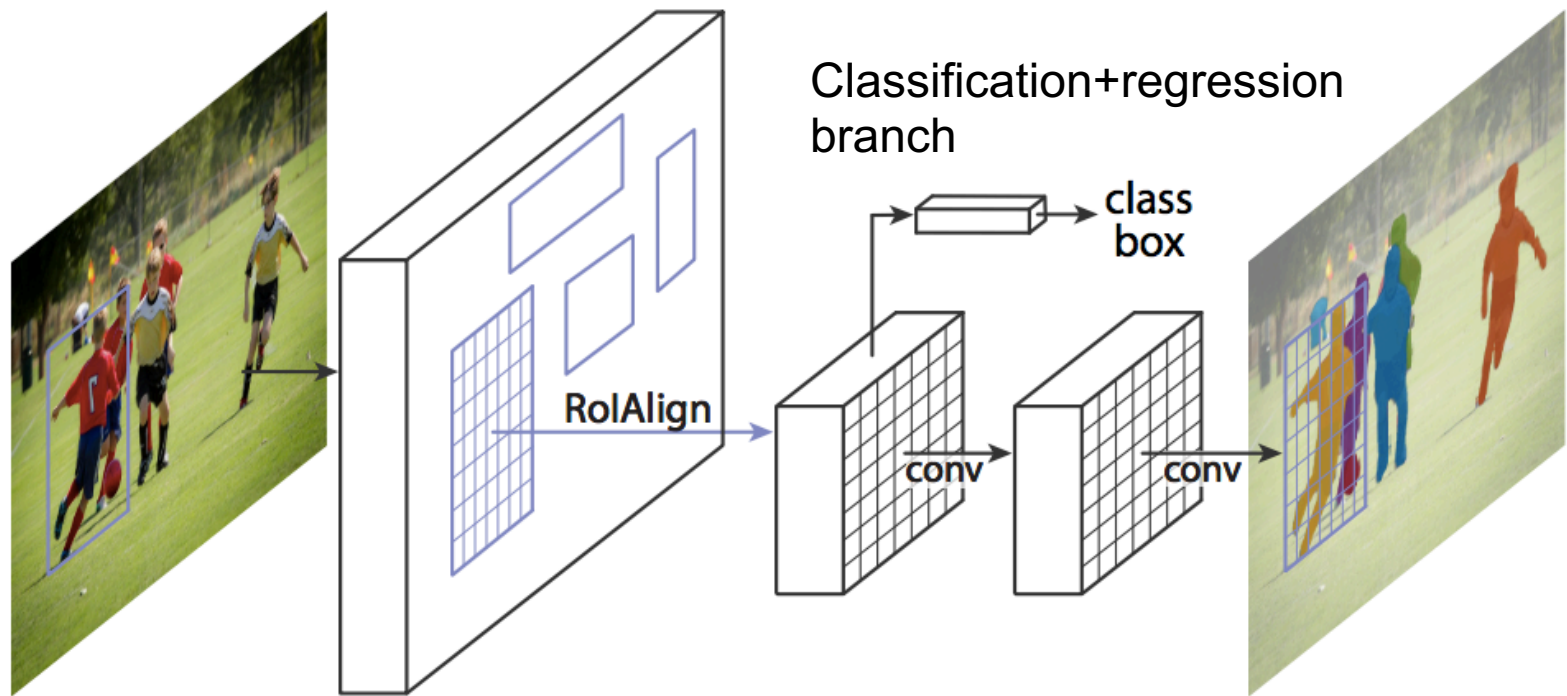
- Average Precision like detection, except region IoU as opposed to box IoU.



$$I/U = \frac{\text{red square}}{\text{red square} + \text{blue square} + \text{purple square}}$$

# Mask R-CNN

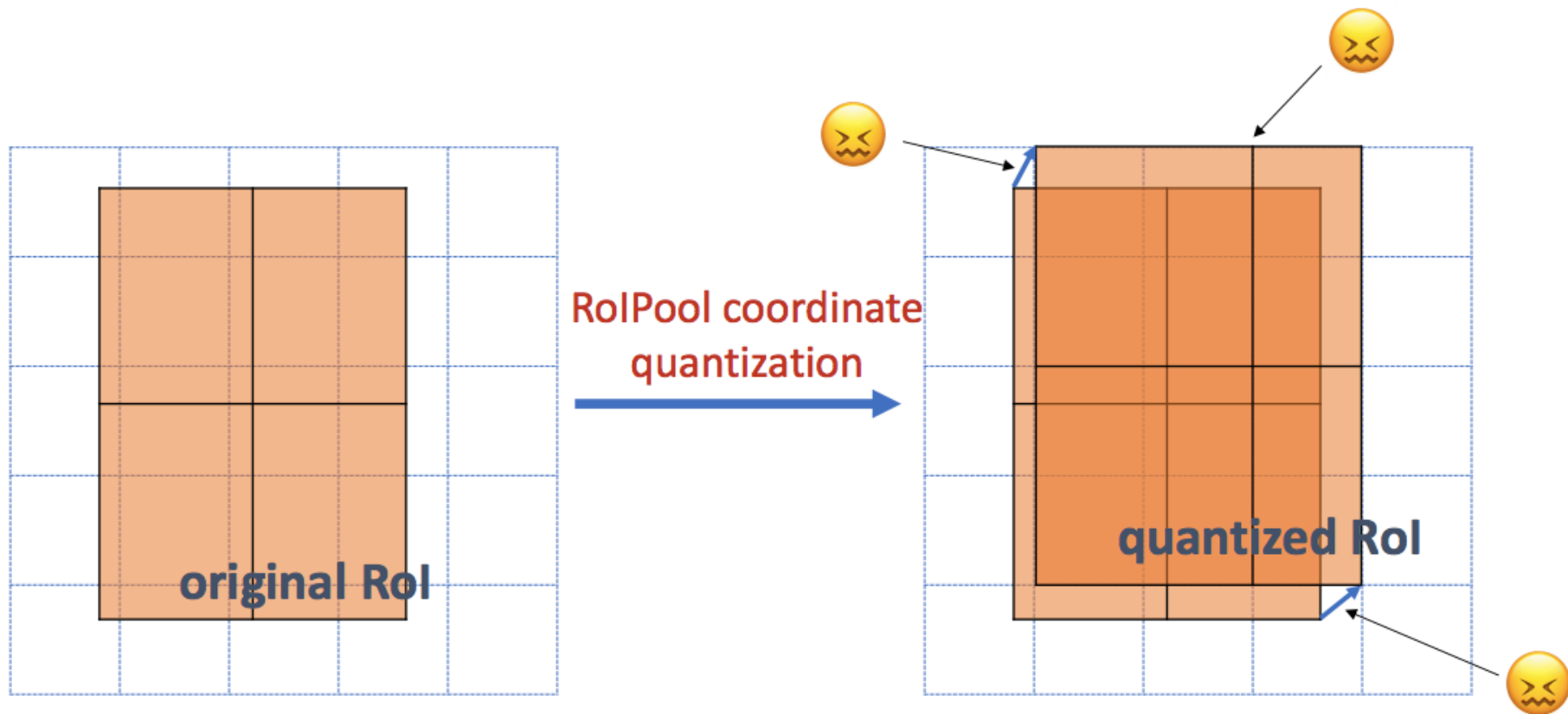
- Mask R-CNN = Faster R-CNN + FCN on Rols



Mask branch: separately predict segmentation for each possible class

# RoIAlign vs. RoIPool

- RoIPool: nearest neighbor quantization

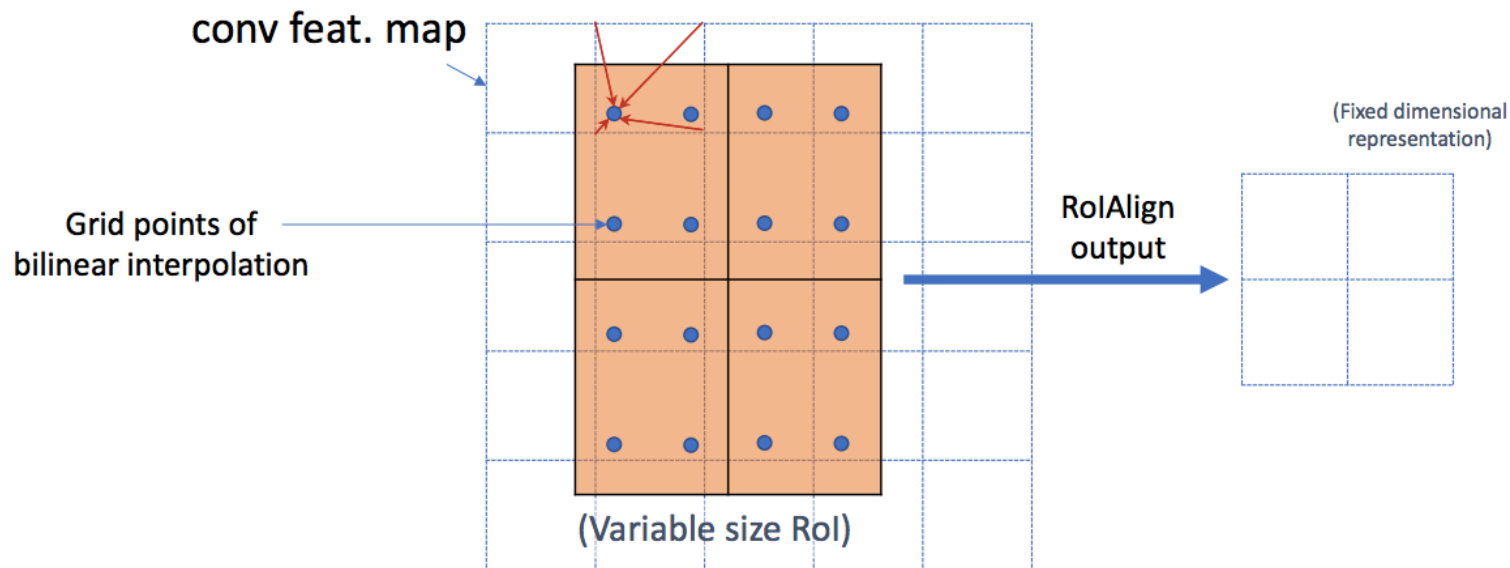


K. He, G. Gkioxari, P. Dollar, and R. Girshick, [Mask R-CNN](#),  
ICCV 2017 (Best Paper Award)

# RoIAlign vs. RoIPool

---

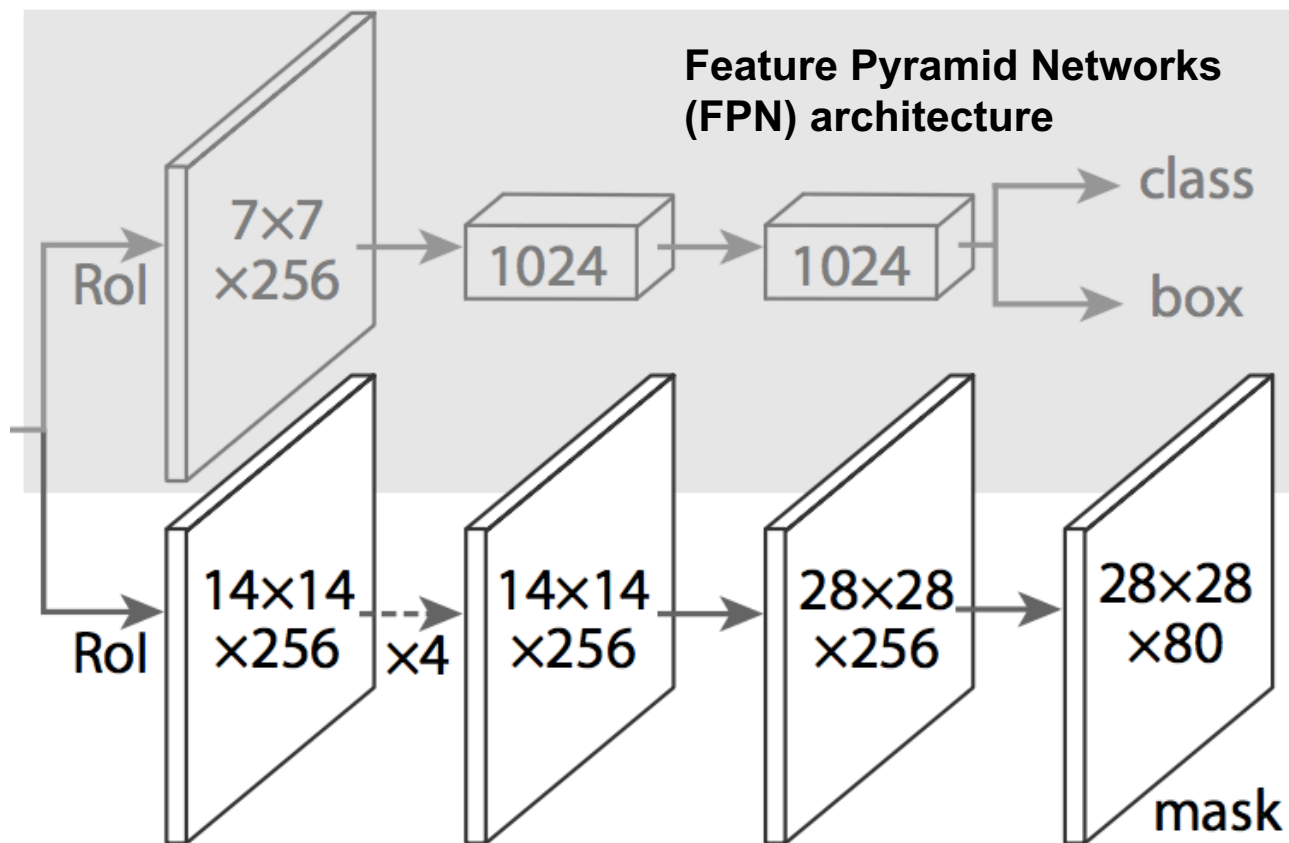
- RoIPool: nearest neighbor quantization
- RoIAlign: bilinear interpolation



K. He, G. Gkioxari, P. Dollar, and R. Girshick, [Mask R-CNN](#),  
ICCV 2017 (Best Paper Award)

# Mask R-CNN

- From RoIAlign features, predict class label, bounding box, and segmentation mask



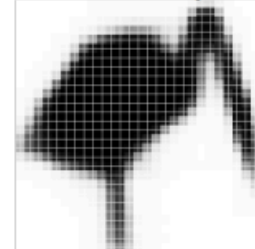
K. He, G. Gkioxari, P. Dollar, and R. Girshick, [Mask R-CNN](#), ICCV 2017 (Best Paper Award)

# Mask R-CNN

---



28x28 soft prediction



Resized Soft prediction



Final mask

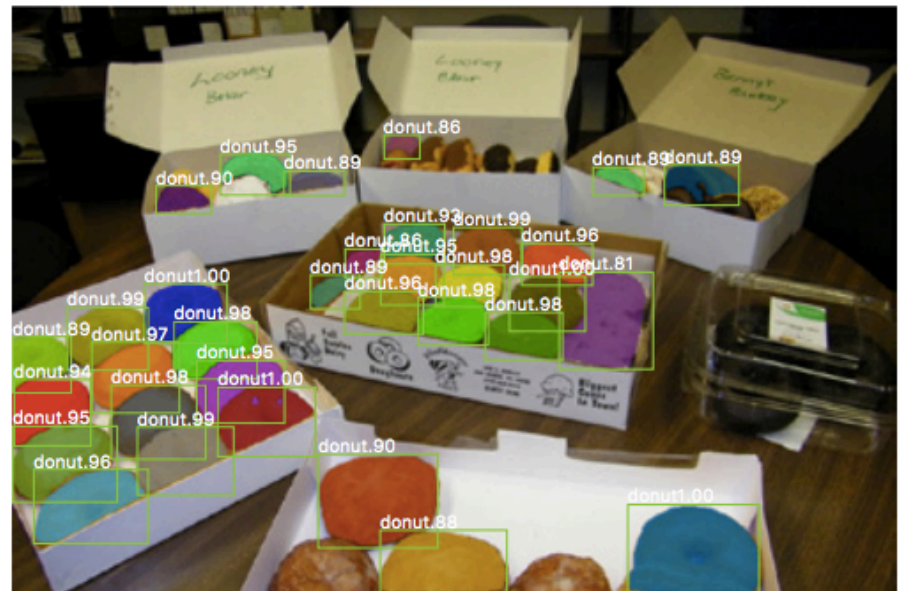
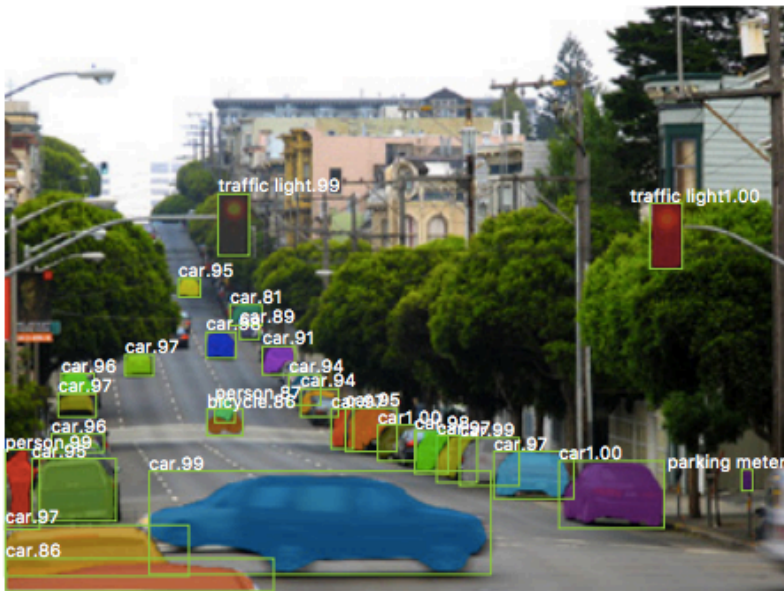


Validation image with box detection shown in red

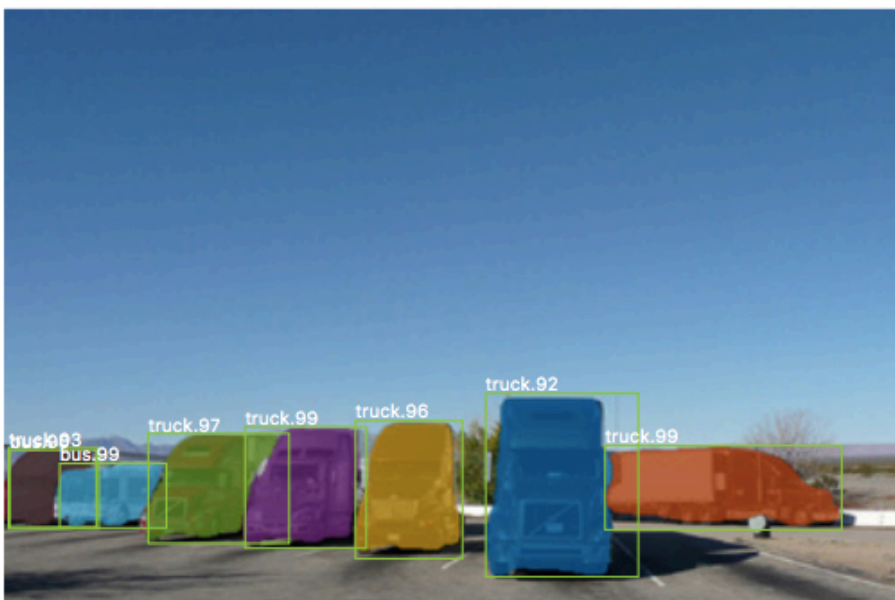
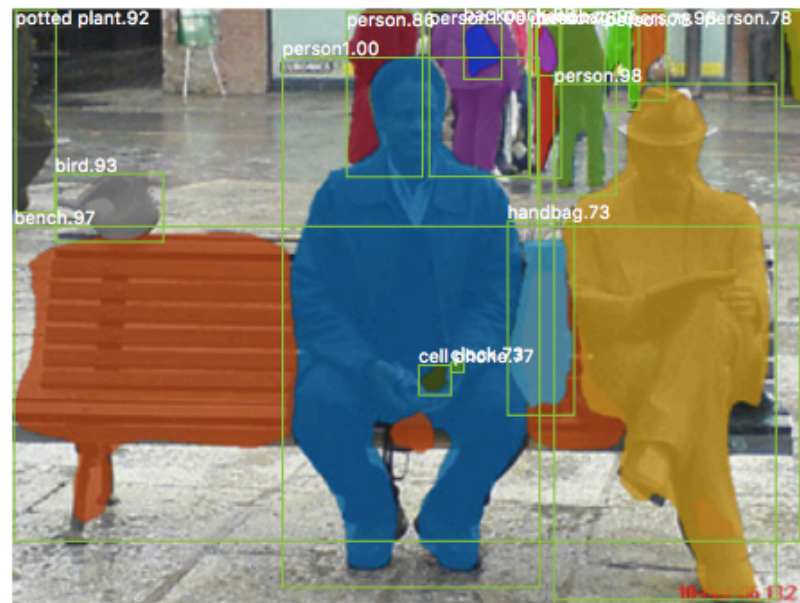
K. He, G. Gkioxari, P. Dollar, and R. Girshick, [Mask R-CNN](#),  
ICCV 2017 (Best Paper Award)



# Example results



# Example results



# Instance segmentation results on COCO

---

	backbone	AP	AP <sub>50</sub>	AP <sub>75</sub>	AP <sub>S</sub>	AP <sub>M</sub>	AP <sub>L</sub>
MNC [10]	ResNet-101-C4	24.6	44.3	24.8	4.7	25.9	43.6
FCIS [26] +OHEM	ResNet-101-C5-dilated	29.2	49.5	-	7.1	31.3	50.0
FCIS+++ [26] +OHEM	ResNet-101-C5-dilated	33.6	54.5	-	-	-	-
<b>Mask R-CNN</b>	ResNet-101-C4	33.1	54.9	34.8	12.1	35.6	51.1
<b>Mask R-CNN</b>	ResNet-101-FPN	35.7	58.0	37.8	15.5	38.1	52.4
<b>Mask R-CNN</b>	ResNeXt-101-FPN	<b>37.1</b>	<b>60.0</b>	<b>39.4</b>	<b>16.9</b>	<b>39.9</b>	<b>53.5</b>

AP at different IoU thresholds

AP for different size instances

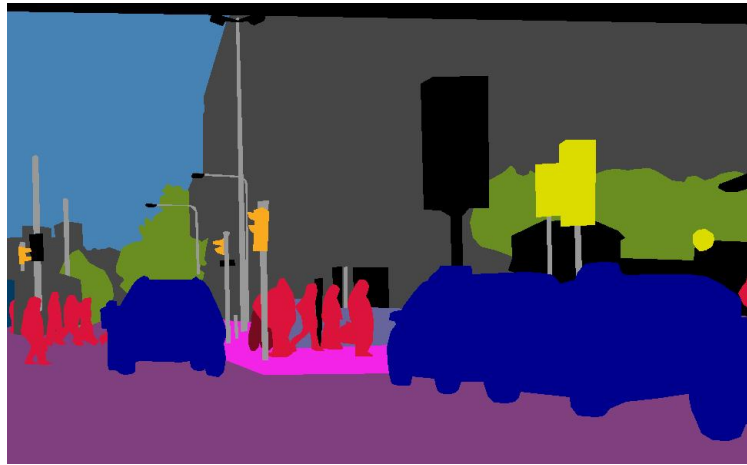
K. He, G. Gkioxari, P. Dollar, and R. Girshick, [Mask R-CNN](#),  
ICCV 2017 (Best Paper Award)

# Unifying Semantic and Instance Segm.

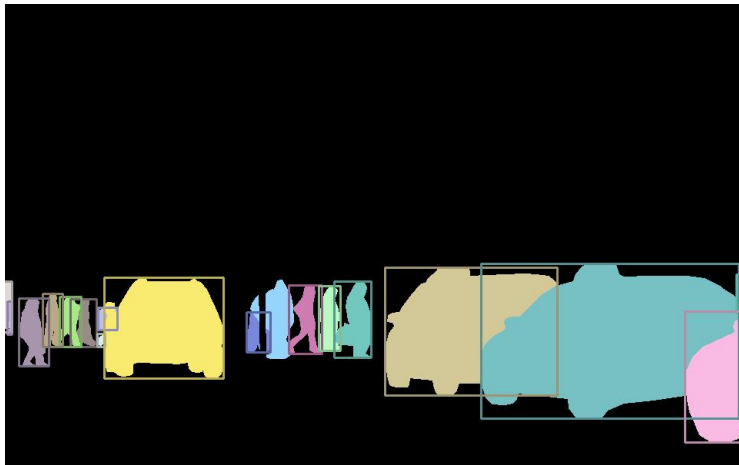
---



(a) image



(b) semantic segmentation



(c) instance segmentation

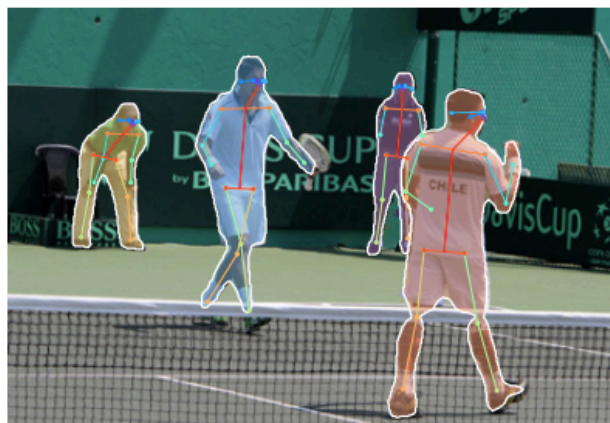


(d) panoptic segmentation

Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, Piotr Dollár,  
[Panoptic Segmentation](#), CVPR 2019.

# Keypoint prediction

- Given  $K$  keypoints, train model to predict  $K$   $m \times m$  *one-hot* maps

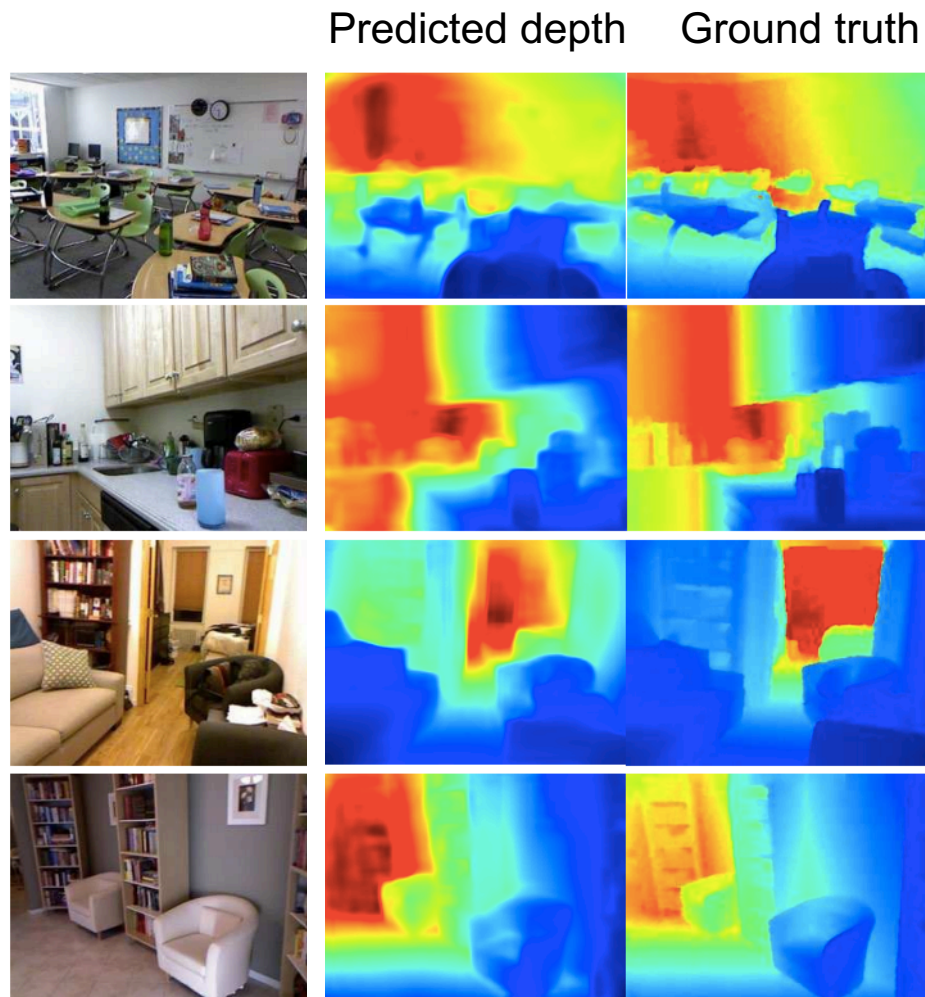
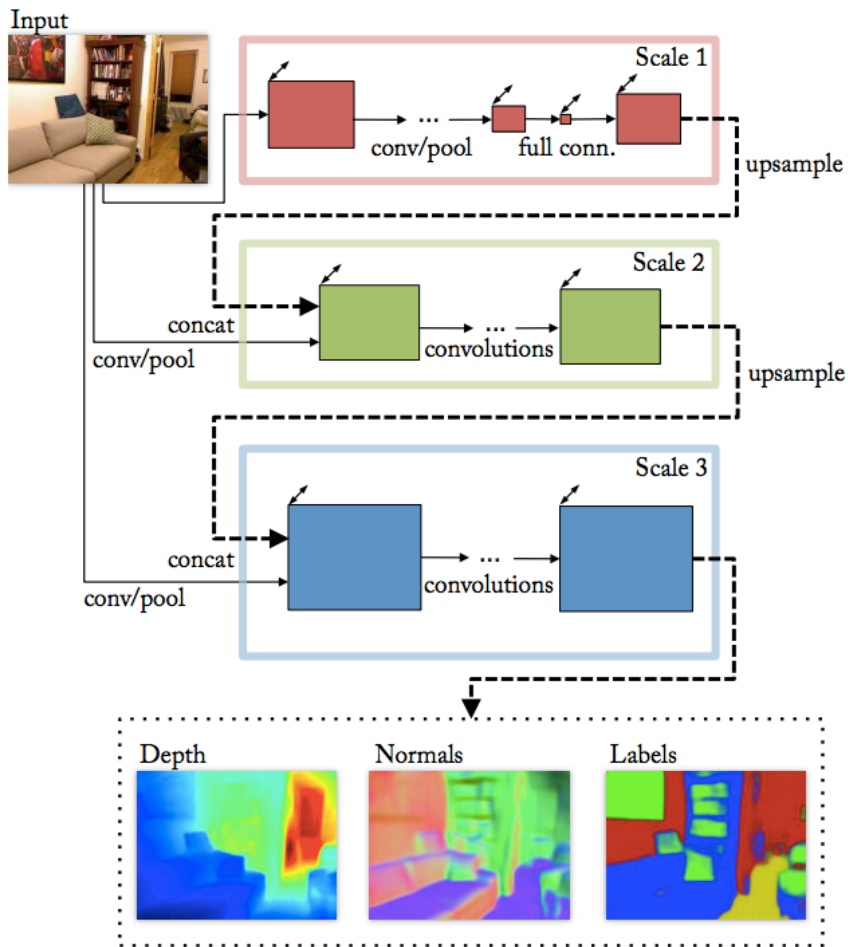


# Other dense prediction tasks

---

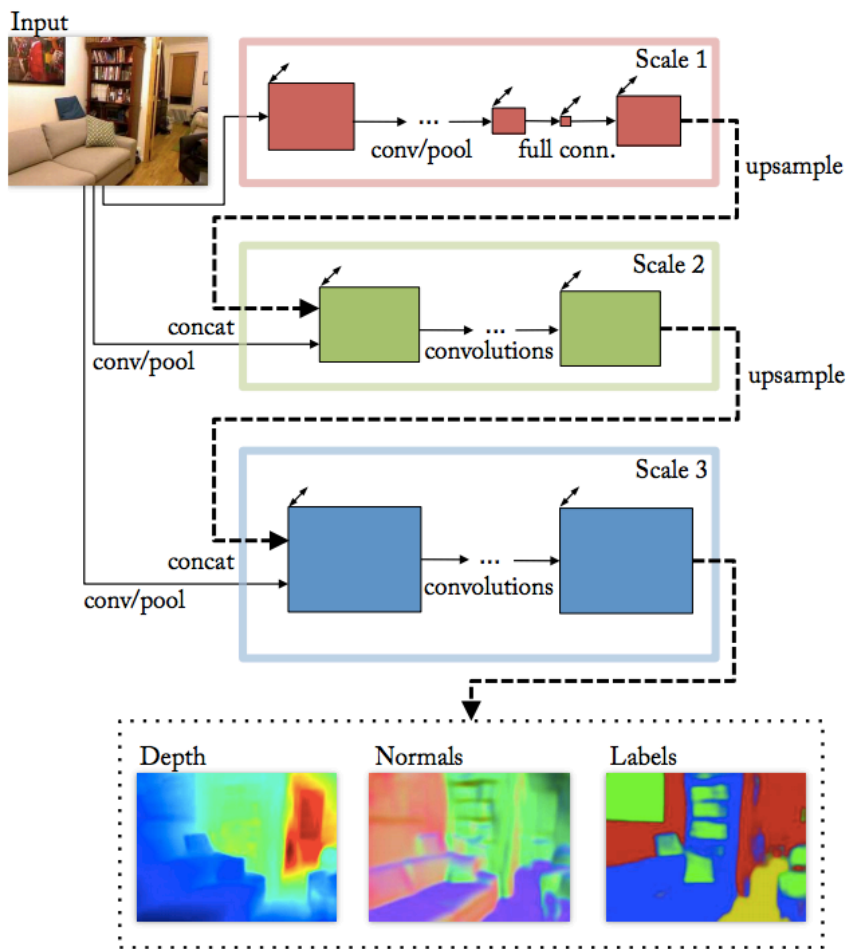
- Depth estimation
- Surface normal estimation
- Colorization
- ....

# Depth and normal estimation

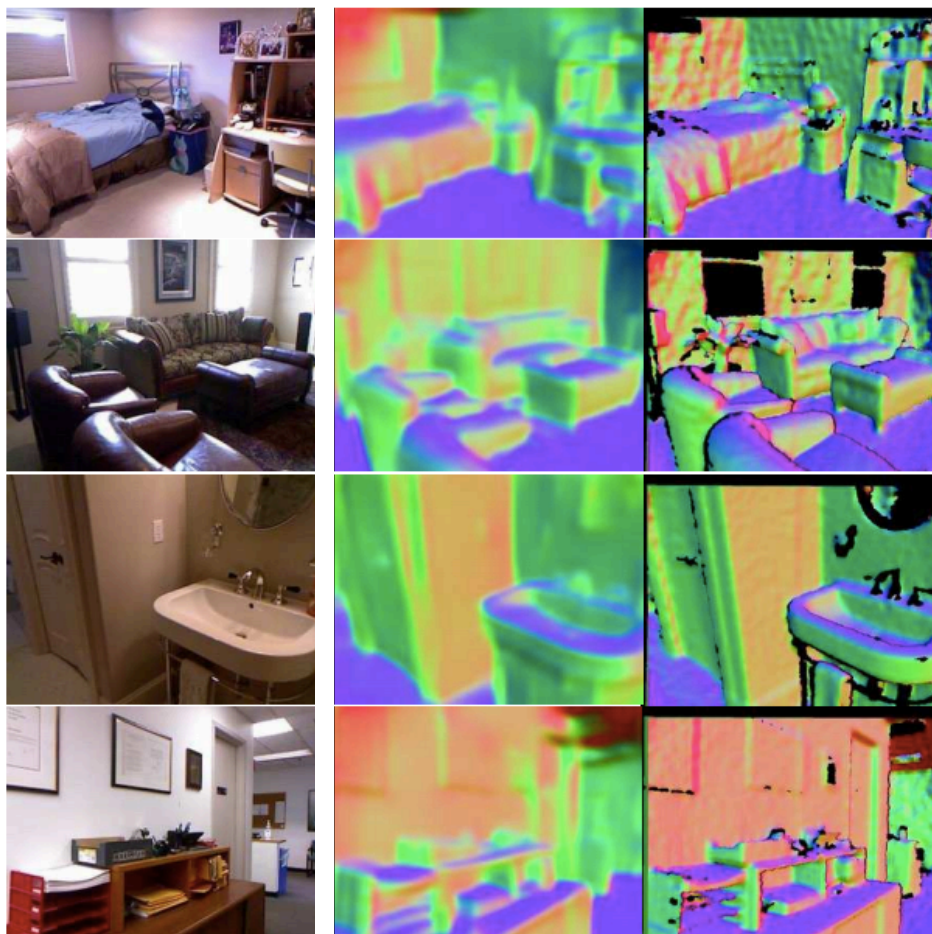


D. Eigen and R. Fergus, [Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture](#), ICCV 2015

# Depth and normal estimation



Predicted normals Ground truth



D. Eigen and R. Fergus, [Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture](#), ICCV 2015



# Colorization

