



CNN Architectures for Dense Recognition: Image segmentation

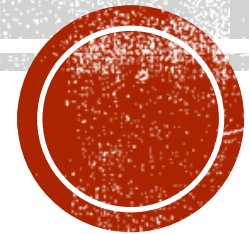
By

Dr. Shiv Ram Dubey

Computer Vision and Biometrics Lab

Department of Information Technology

Indian Institute of Information Technology, Allahabad



DISCLAIMER

The content (text, image, and graphics) used in this slide are adopted from many sources for academic purposes. Broadly, the sources have been given due credit appropriately. However, there is a chance of missing out some original primary sources. The authors of this material do not claim any copyright of such material.



PREVIOUS CLASS: OBJECT BOUNDING BOX DETECTIONS

Deep learning based detectors

Two-stage detectors

R-CNN

Fast R-CNN

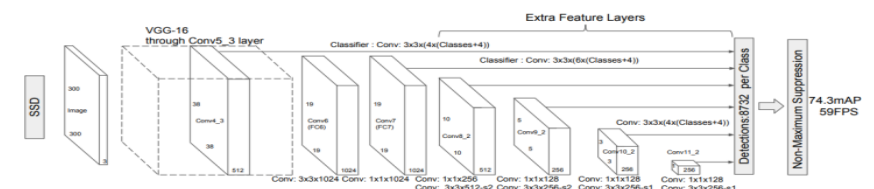
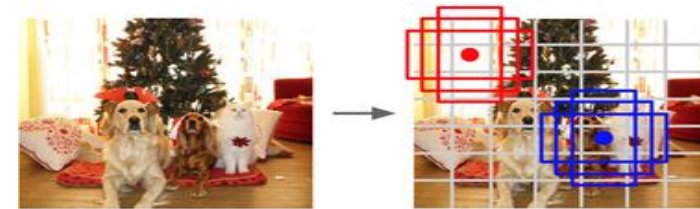
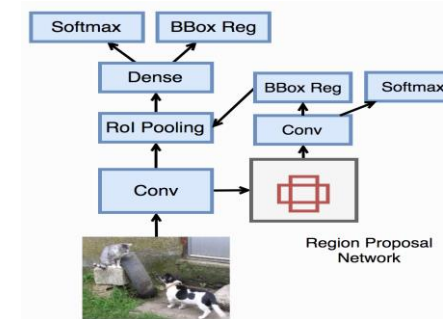
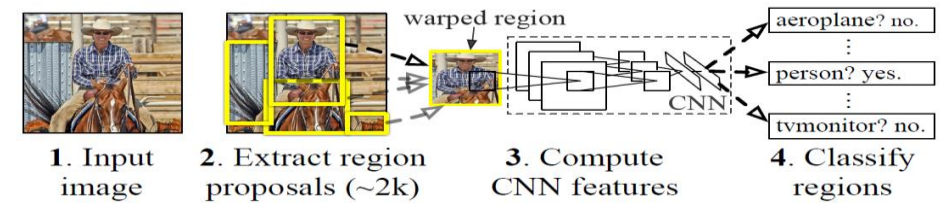
Faster R-CNN

One-stage detectors

YOLO

SSD

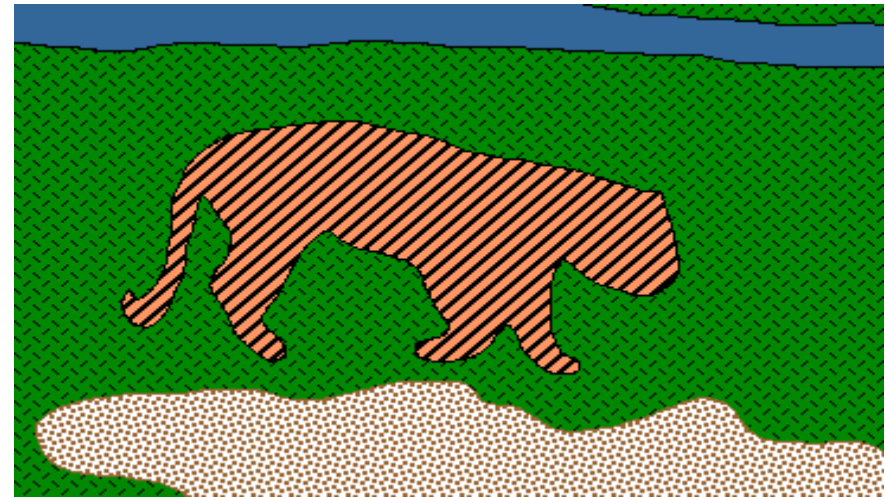
RetinaNet



TODAY'S CLASS

Image Segmentation

Group pixels into meaningful or perceptually similar regions



OUTLINE

- Early “hacks”
 - Sliding window
 - Fully convolutional networks
- Deep network operations for dense prediction
 - Transposed convolutions
 - Unpooling
 - Dilated convolutions
- Instance segmentation
 - Mask R-CNN
- Other dense prediction problems

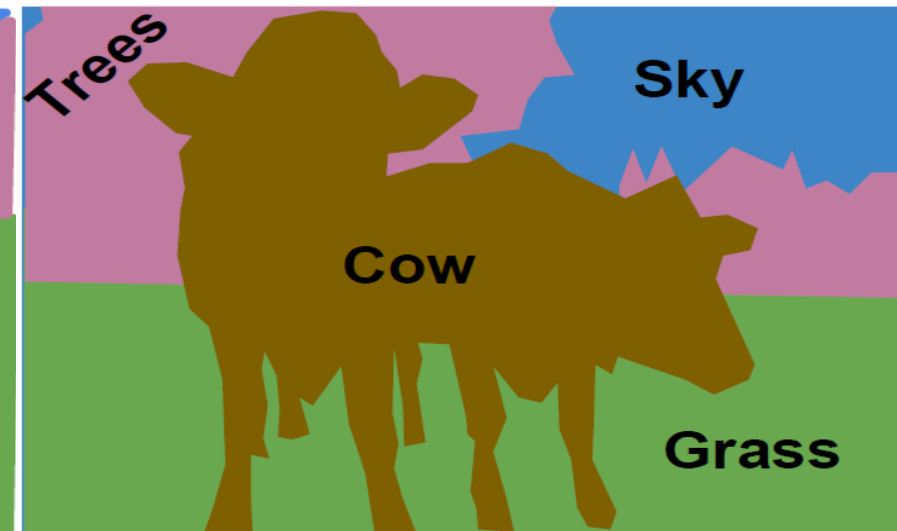
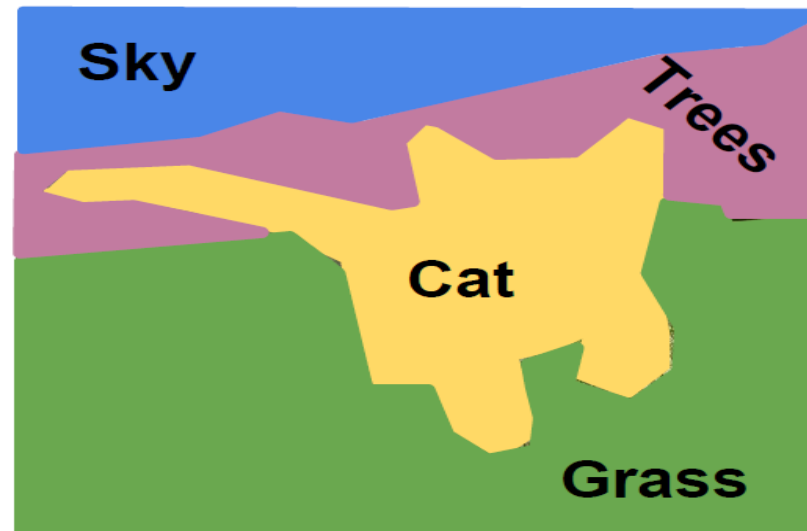


SEMANTIC SEGMENTATION

These images are CC0 public domain [source1](#) [source2](#)

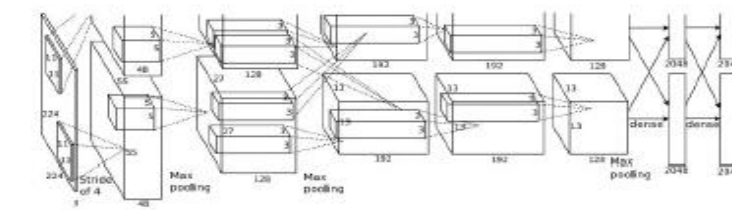
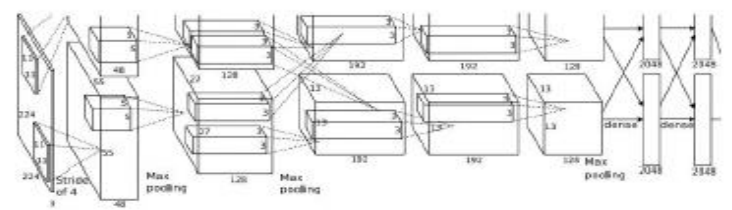
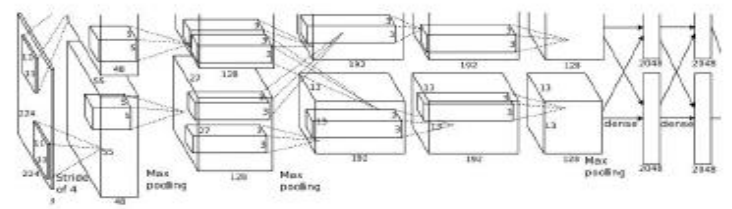
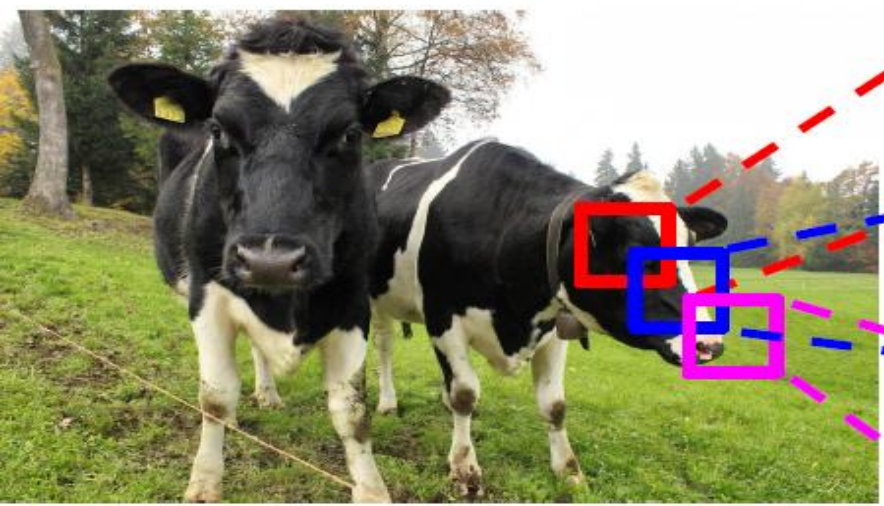
Label each pixel in the image with a category label

Don't differentiate instances, only care about pixels



SEMANTIC SEGMENTATION: SLIDING WINDOW

Full image

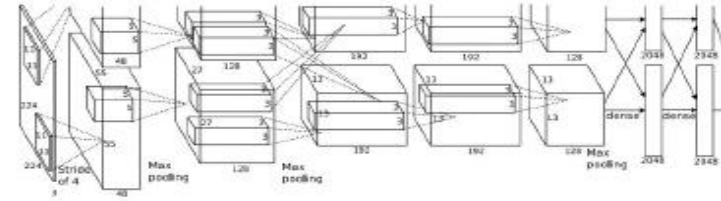
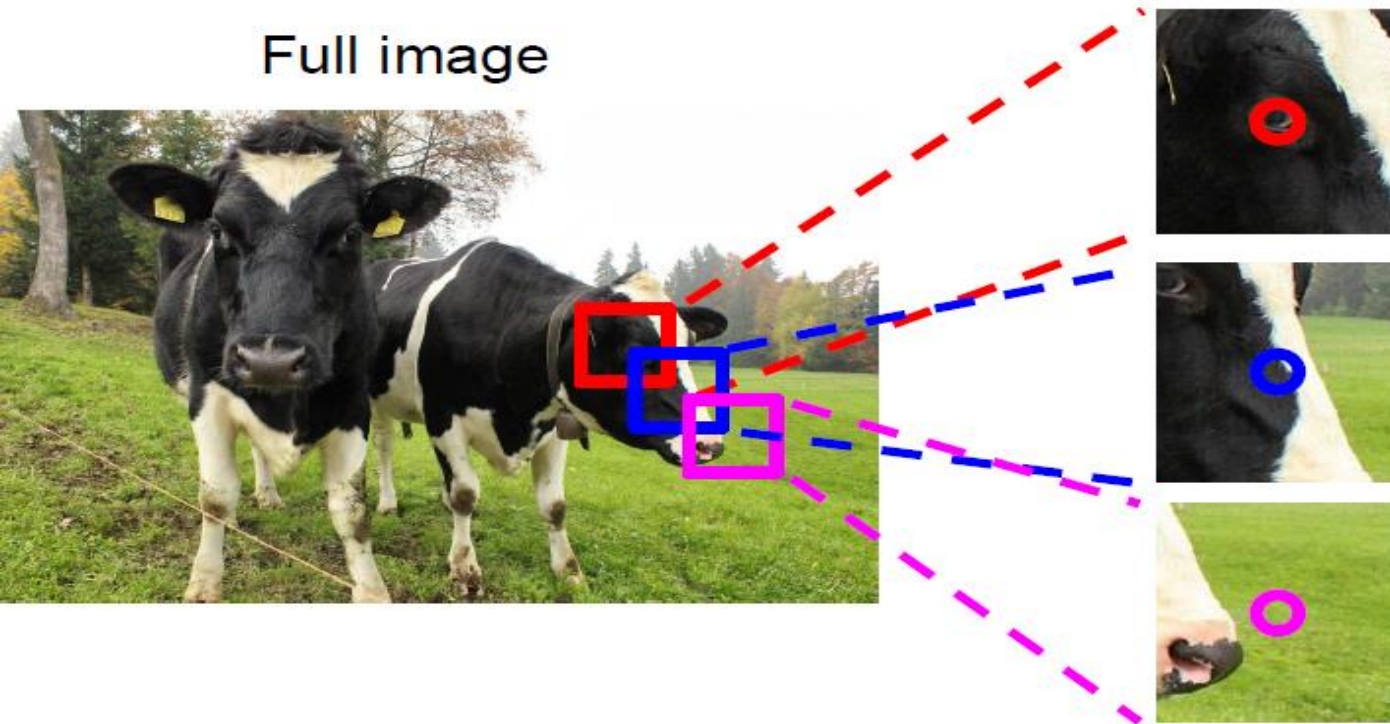


Cow

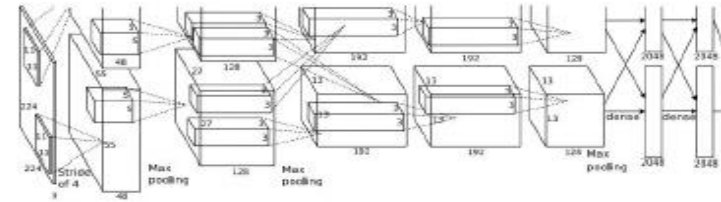
Cow

Grass

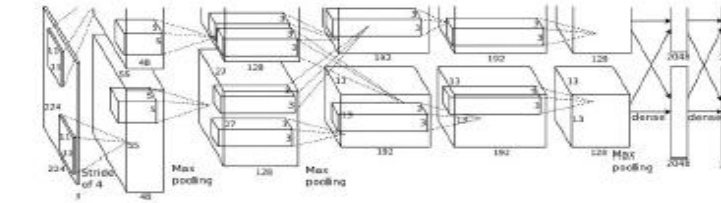
SEMANTIC SEGMENTATION: SLIDING WINDOW



Cow



Cow



Grass

Problem:

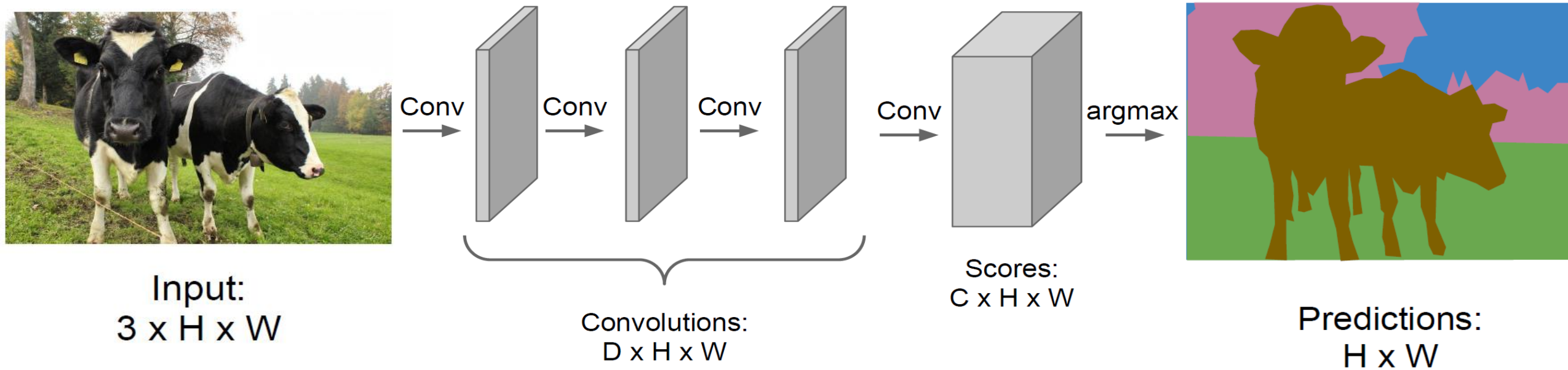
Very inefficient!

Not reusing shared features between overlapping patches



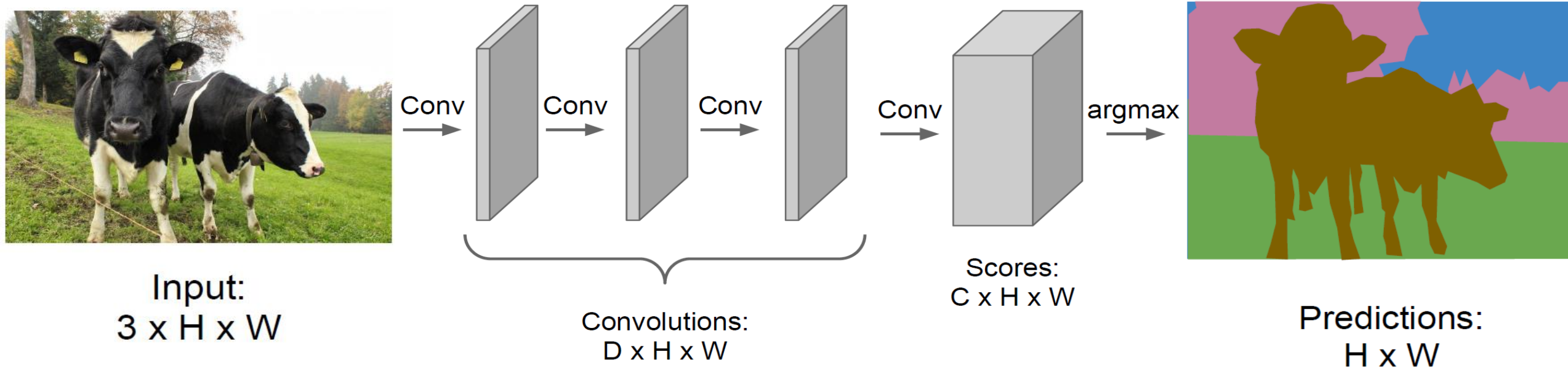
SEMANTIC SEGMENTATION: FULLY CONVOLUTIONAL

Design a network as a bunch of convolutional layers to make predictions for pixels all at once!



SEMANTIC SEGMENTATION: FULLY CONVOLUTIONAL

Design a network as a bunch of convolutional layers to make predictions for pixels all at once!



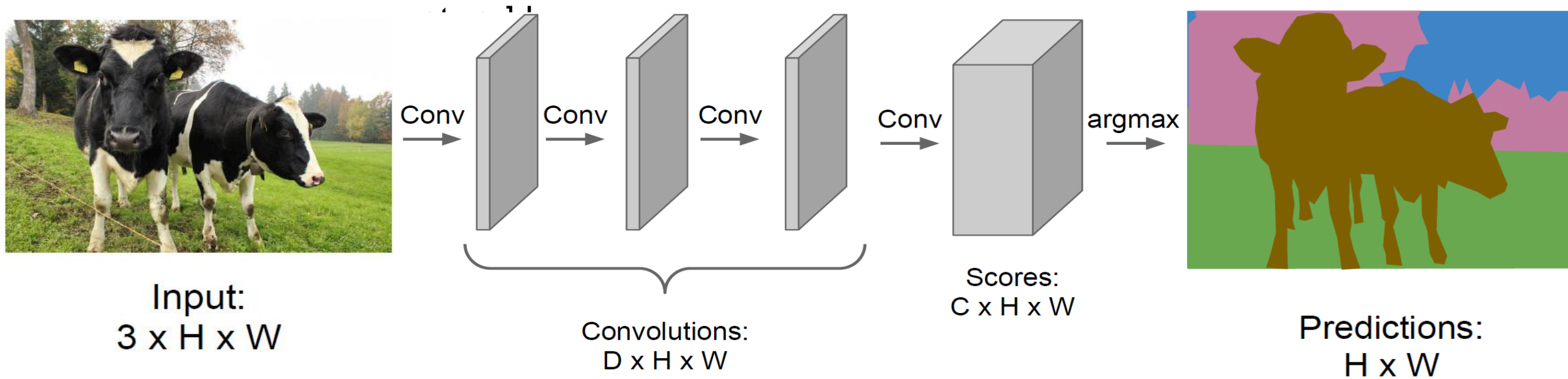
Problem:

Convolutions at original image resolution will be very expensive ...



SEMANTIC SEGMENTATION: FULLY CONVOLUTIONAL

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the



SEMANTIC SEGMENTATION: FULLY CONVOLUTIONAL

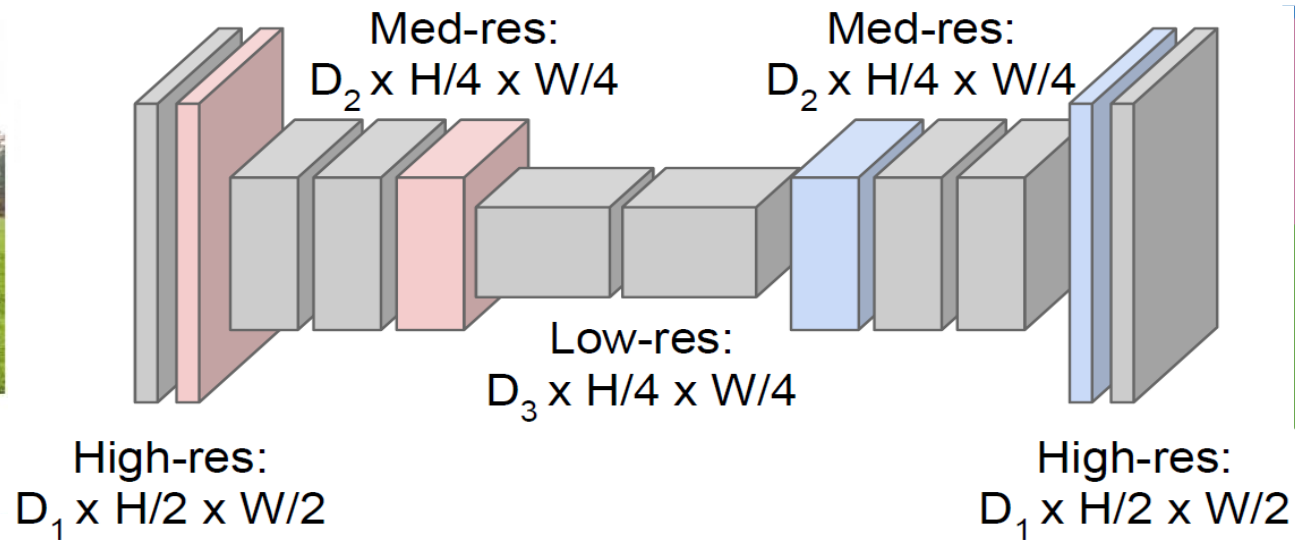
Downsampling:
Pooling, strided
convolution

Design network as a bunch of convolutional layers,
with **downsampling** and **upsampling** inside the
network!

Upsampling:
??



Input:
 $3 \times H \times W$



Predictions:
 $H \times W$



SEMANTIC SEGMENTATION: FULLY CONVOLUTIONAL

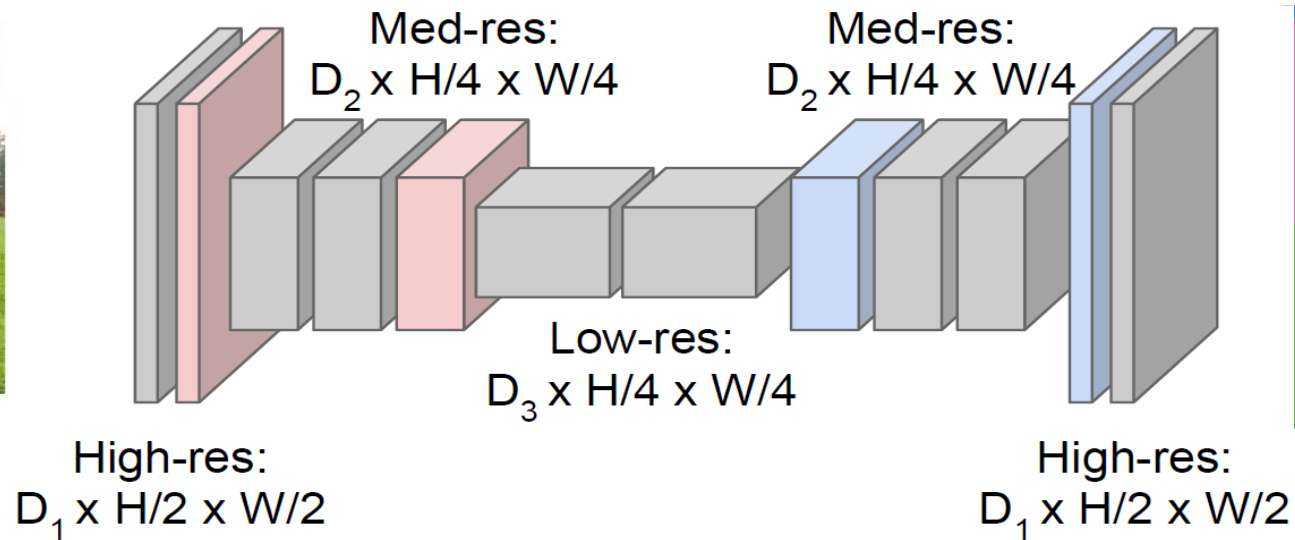
Downsampling:
Pooling, strided
convolution

Design network as a bunch of convolutional layers,
with **downsampling** and **upsampling** inside the
network!

Upsampling:
Unpooling or strided
transpose convolution



Input:
 $3 \times H \times W$



Predictions:
 $H \times W$



UPSAMPLING: UNPOOLING

Nearest Neighbor

1	2
3	4



1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Input: 2 x 2

Output: 4 x 4



UPSAMPLING: UNPOOLING

Nearest Neighbor

1	2
3	4



1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Input: 2 x 2

Output: 4 x 4

“Bed of Nails”

1	2
3	4



1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Input: 2 x 2

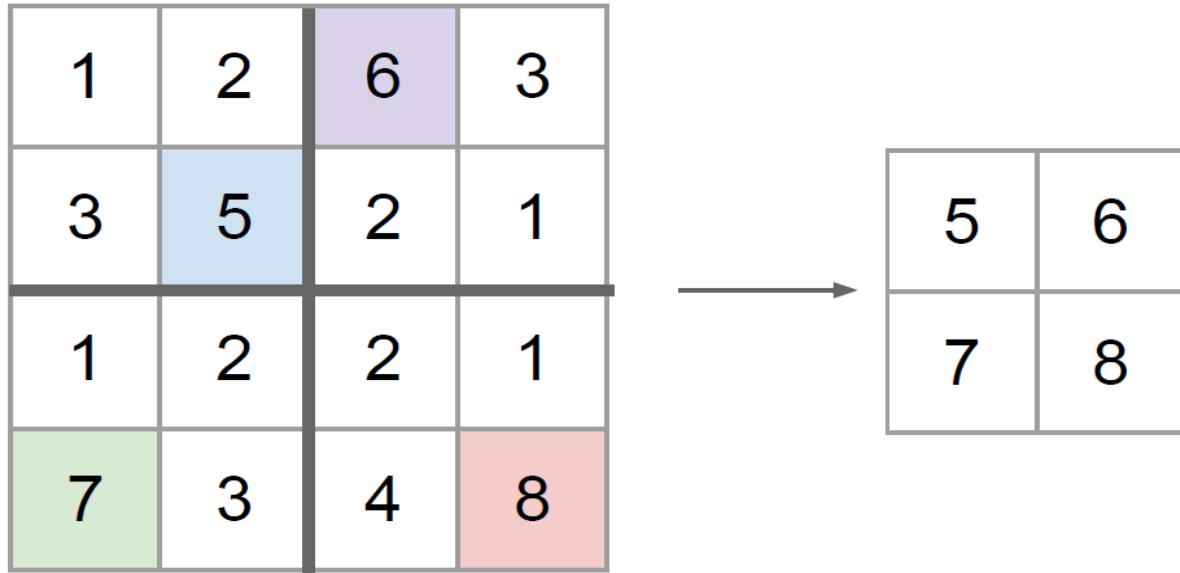
Output: 4 x 4



UPSAMPLING: MAX UNPOOLING

Max Pooling

Remember which element was max!



Input: 4 x 4

Output: 2 x 2



UPSAMPLING: MAX UNPOOLING

Max Pooling

Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4



5	6
7	8

Output: 2 x 2

Max Unpooling

Use positions from pooling layer

1	2
3	4

Input: 2 x 2



0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

Output: 4 x 4



UPSAMPLING: MAX UNPOOLING

Max Pooling

Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4



5	6
7	8

Output: 2 x 2

Max Unpooling

Use positions from pooling layer

1	2
3	4

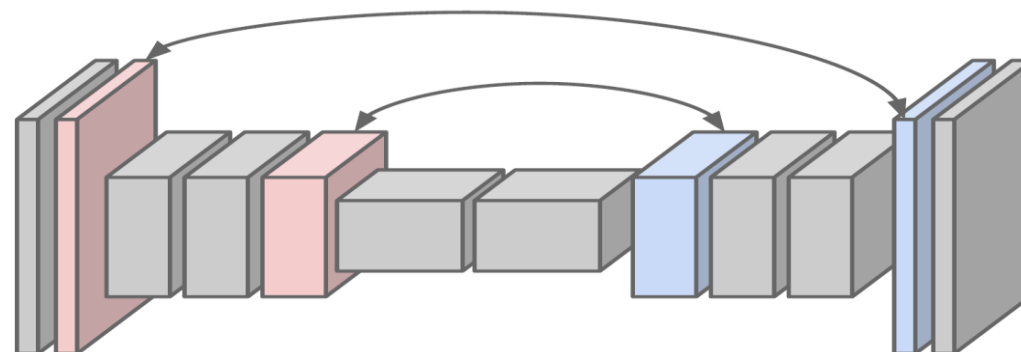
Input: 2 x 2



0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

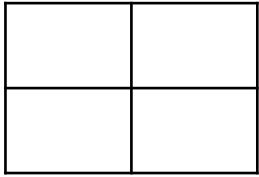
Output: 4 x 4

Corresponding pairs of
downsampling and
upsampling layers

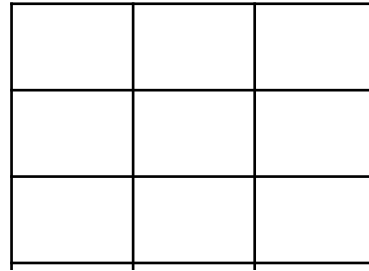


UPSAMPLING: TRANSPOSE CONVOLUTION

3 x 3 **transpose** convolution, stride 2 pad 1



Input: 2 x 2

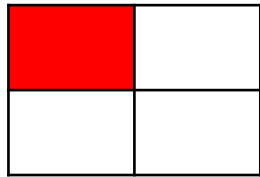


Output: 3 x 3

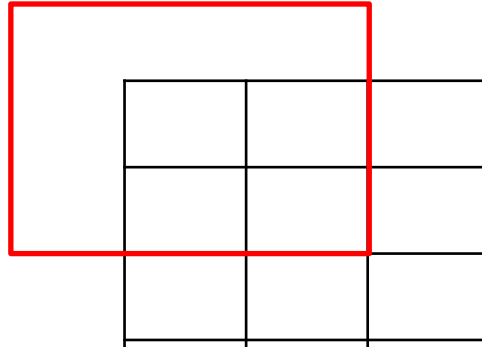


UPSAMPLING: TRANSPOSE CONVOLUTION

3 x 3 **transpose** convolution, stride 2 pad 1



Input gives
weight for
filter



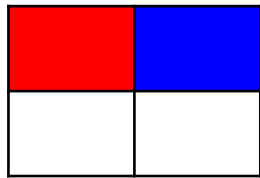
Input: 2 x 2

Output: 3 x 3

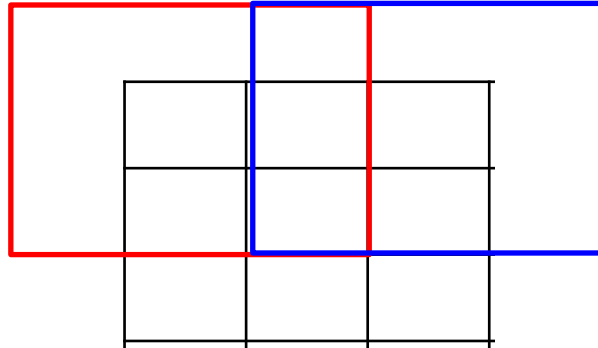


UPSAMPLING: TRANSPOSE CONVOLUTION

3 x 3 **transpose** convolution, stride 2 pad 1



Input gives
weight for
filter



Filter moves 2 pixels in the
output for every one pixel
in the input

Input: 2 x 2

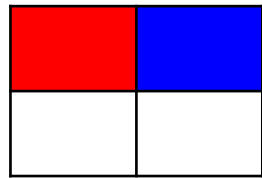
Output: 3 x 3

Stride gives ratio between
movement in output and
input



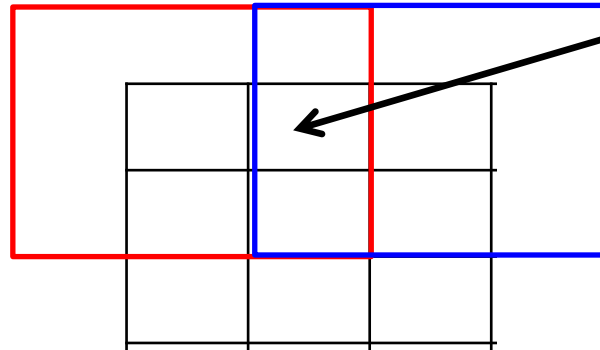
UPSAMPLING: TRANSPOSE CONVOLUTION

3 x 3 **transpose** convolution, stride 2 pad 1



Input gives
weight for
filter

Input: 2 x 2



Sum where
output overlaps

Filter moves 2 pixels in the
output for every one pixel
in the input

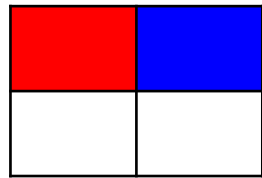
Output: 3 x 3

Stride gives ratio between
movement in output and
input

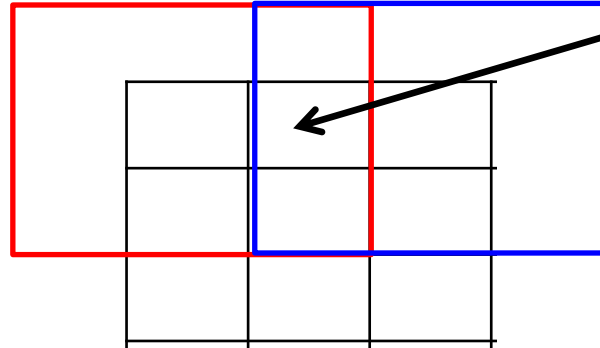


UPSAMPLING: TRANSPOSE CONVOLUTION

3 x 3 **transpose** convolution, stride 2 pad 1



Input gives
weight for
filter



Sum where
output overlaps

Filter moves 2 pixels in the
output for every one pixel
in the input

Input: 2 x 2

Output: 3 x 3

Stride gives ratio between
movement in output and
input

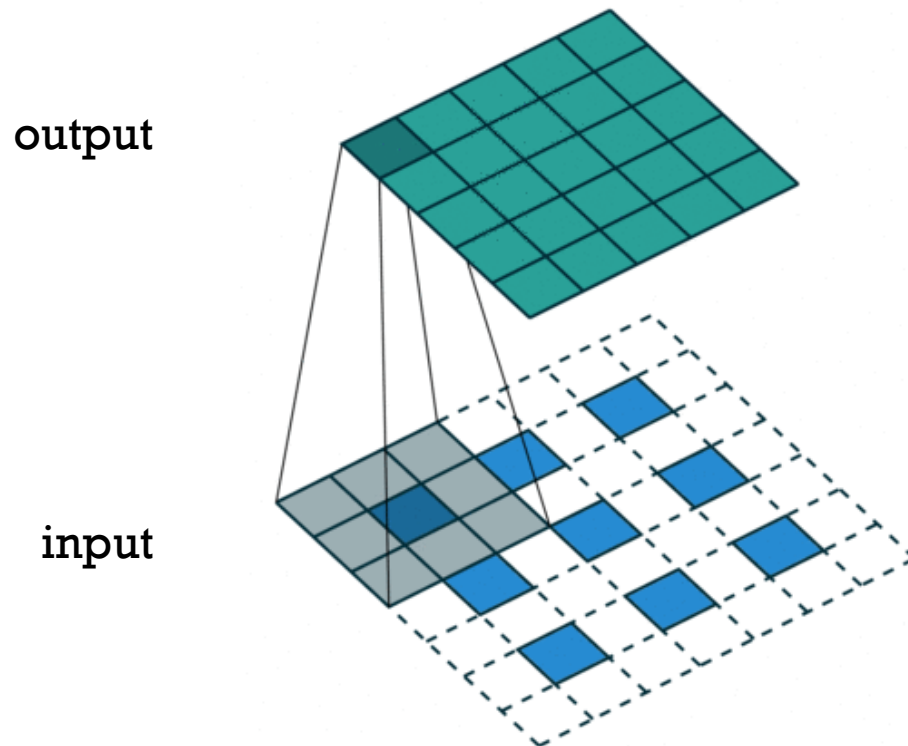
Actual output: 5*5

Need to crop 3*3 centrally



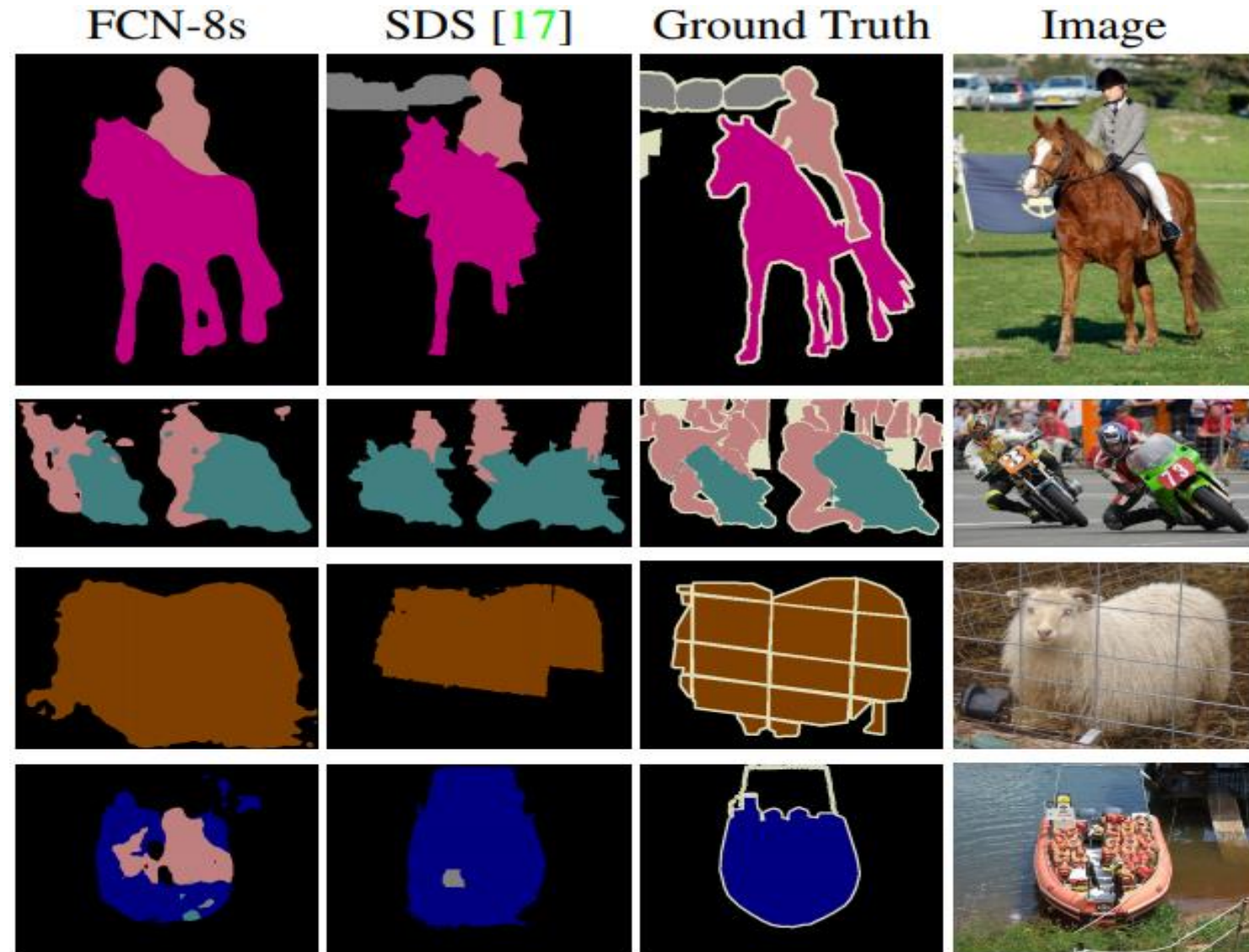
UPSAMPLING IN A DEEP NETWORK

- *Backwards-strided Convolution (Dilated Convolution)*: to increase resolution, use *output stride* > 1
 - For stride 2, dilate the input by inserting rows and columns of zeros between adjacent entries
 - Sometimes called convolution with *fractional input stride* $1/2$

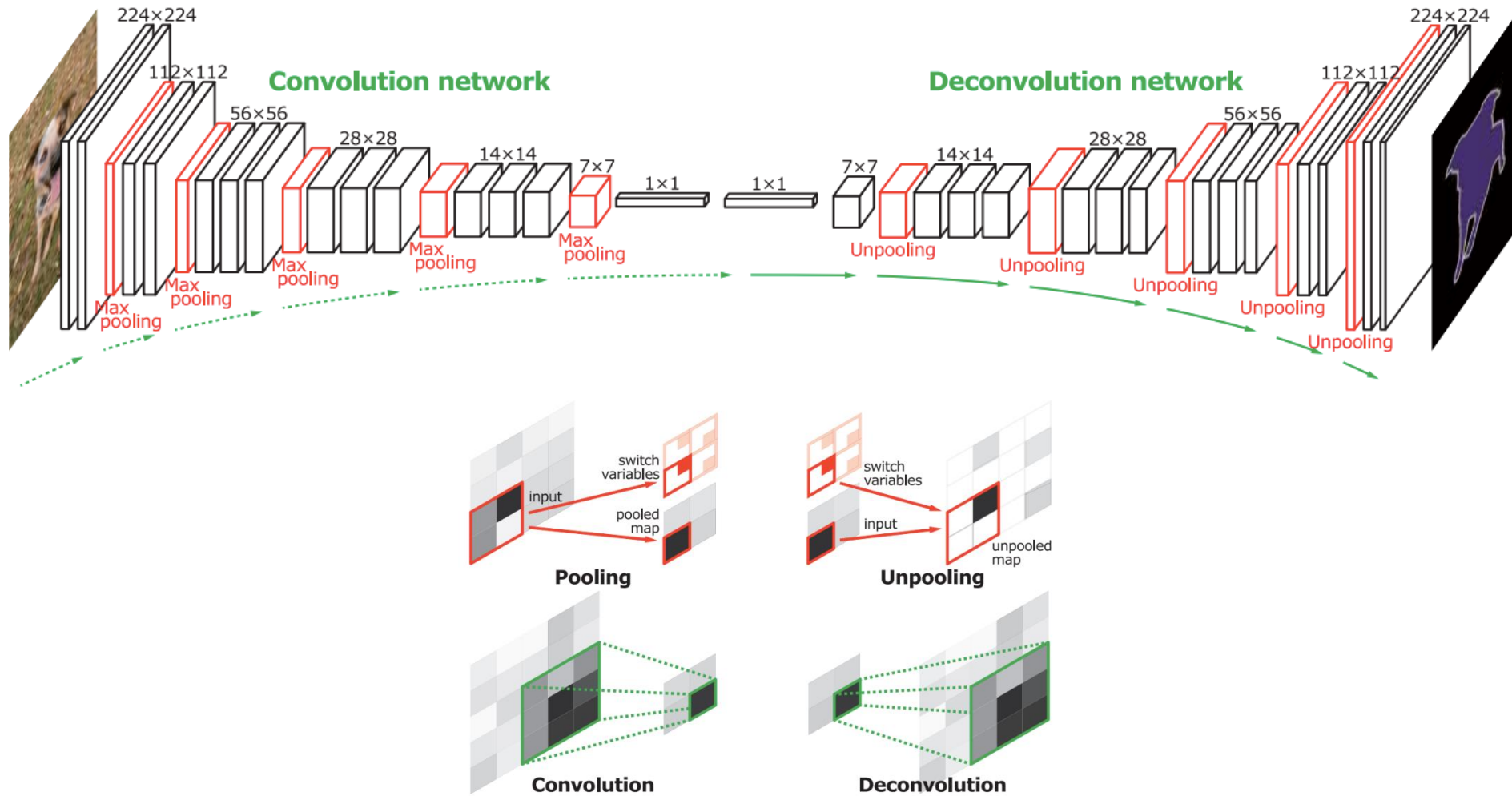


“FULLY CONVOLUTIONAL (FCN)” RESULTS

- Takes advantage of pre-training from classification
- Applied to objects and scenes (NYUd v2)
- But feature pooling reduces spatial sensitivity and resolution



DECONVNET



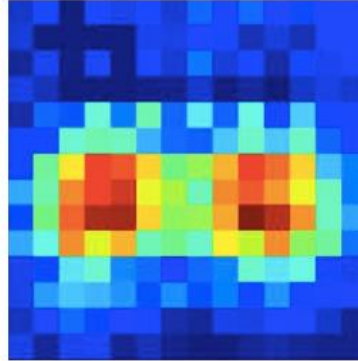
H. Noh, S. Hong, and B. Han, [Learning Deconvolution Network for Semantic Segmentation](#), ICCV 2015



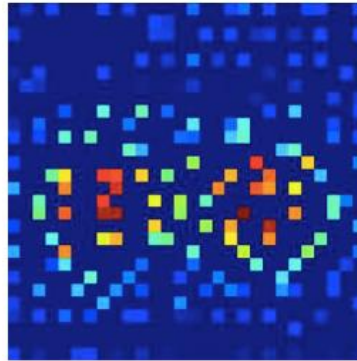
DECONVNET



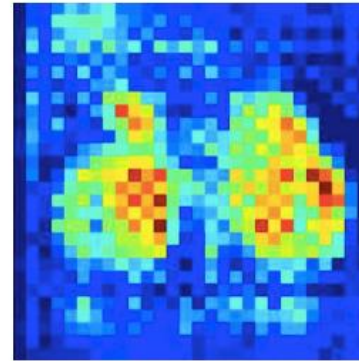
Original image



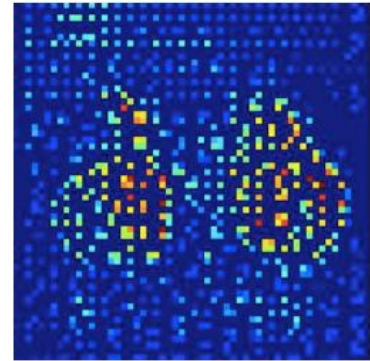
14x14 deconv



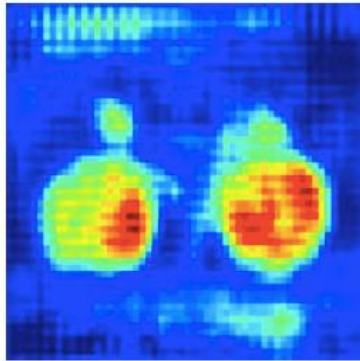
28x28 unpooling



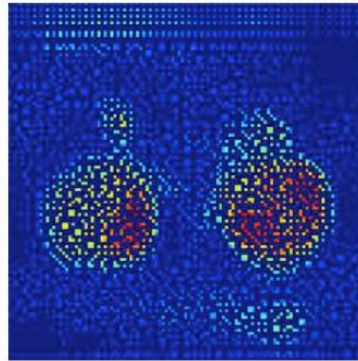
28x28 deconv



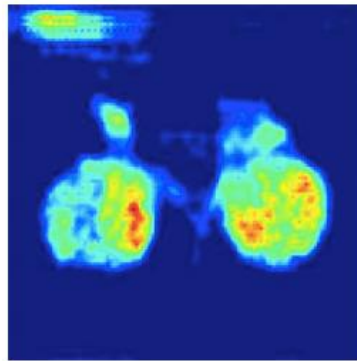
54x54 unpooling



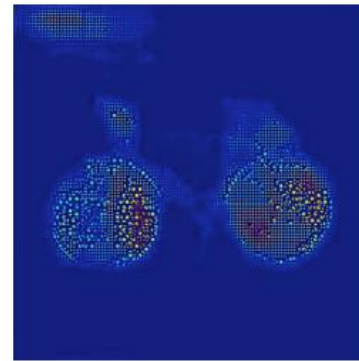
54x54 deconv



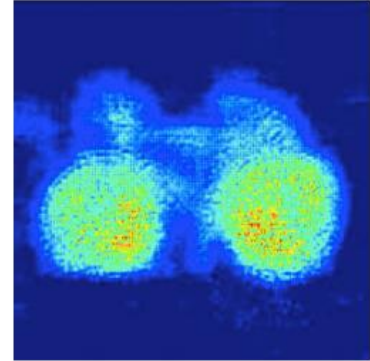
112x112 unpooling



112x112 deconv



224x224 unpooling



224x224 deconv

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

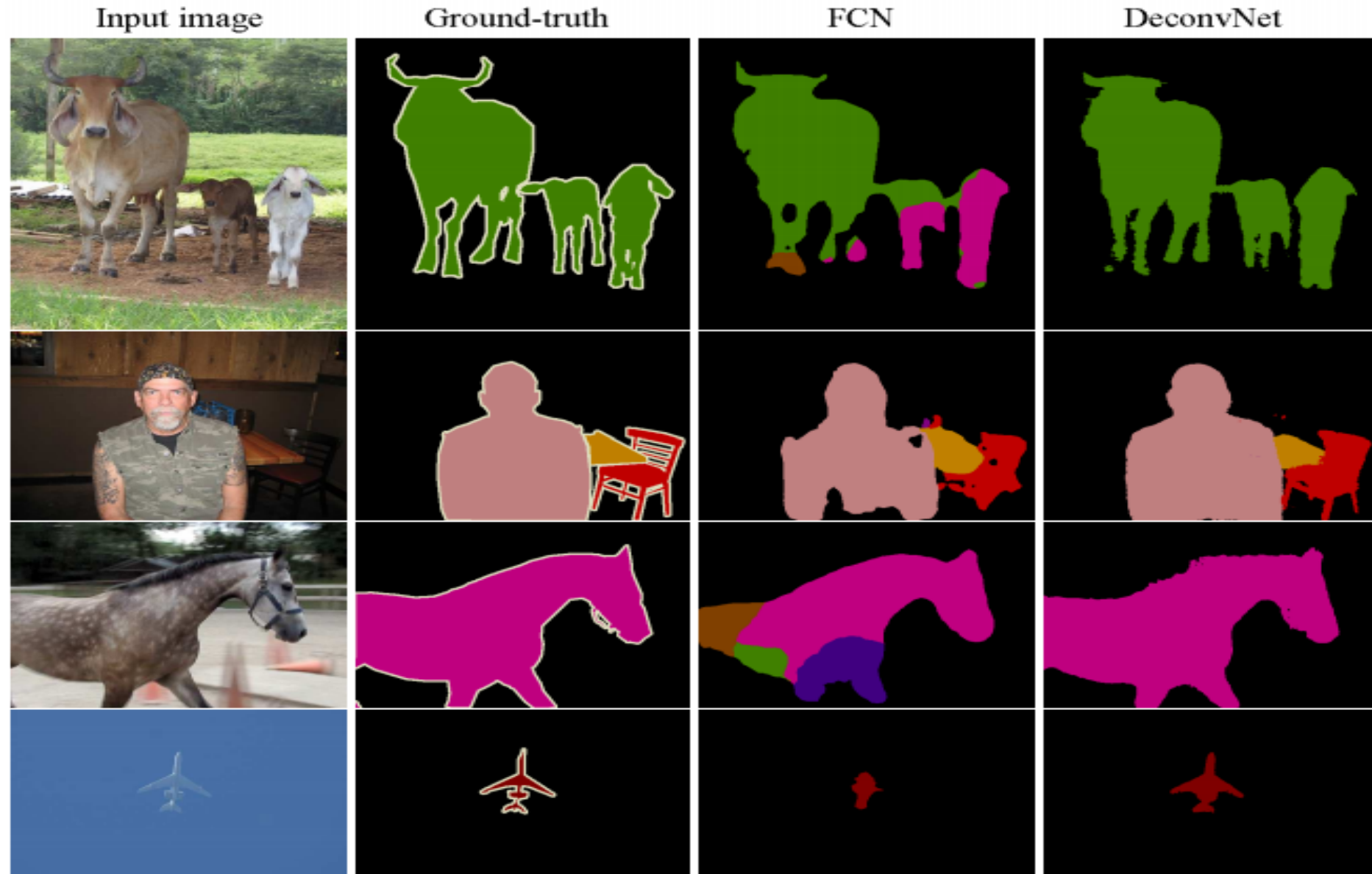


DECONVNET RESULTS

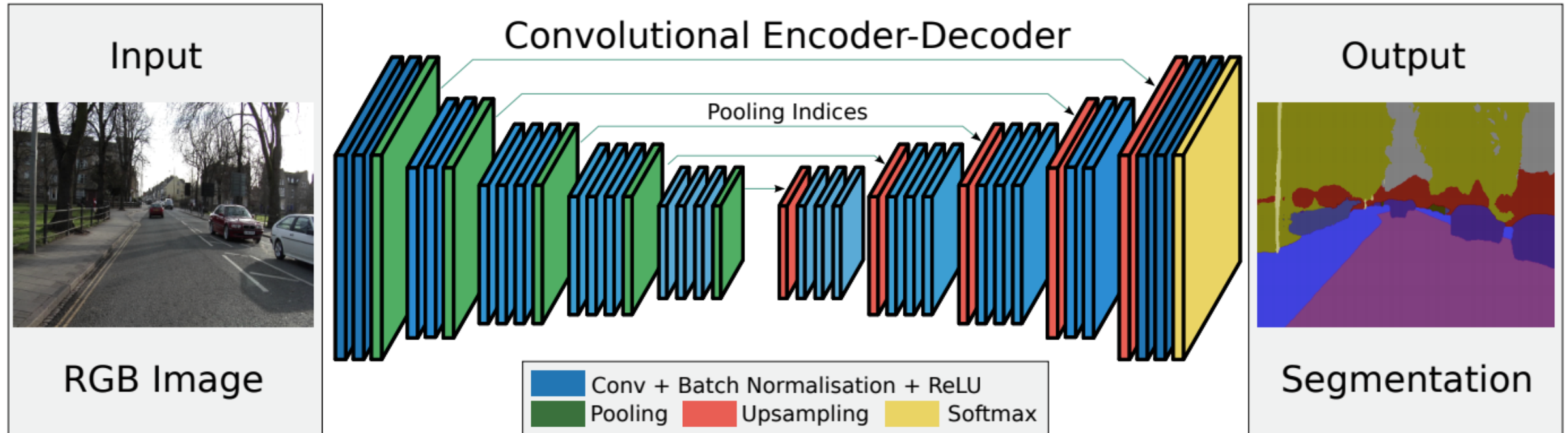
PASCAL VOC 2012	mIoU
FCN-8	62.2
DeconvNet	69.6
Ensemble of DeconvNet and FCN	71.7



DECONVNET RESULTS



SIMILAR ARCHITECTURE: SEGNET



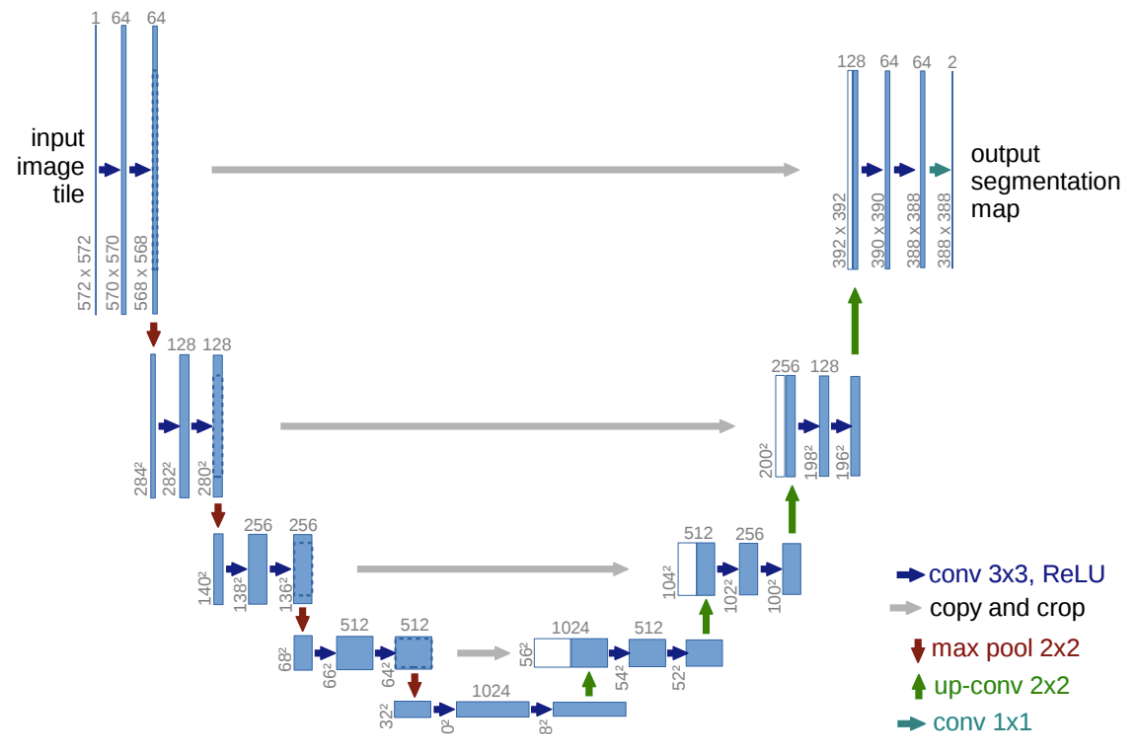
Drop the FC layers,
get better results

V. Badrinarayanan, A. Kendall and R. Cipolla, [SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation](#), PAMI 2017



U-NET

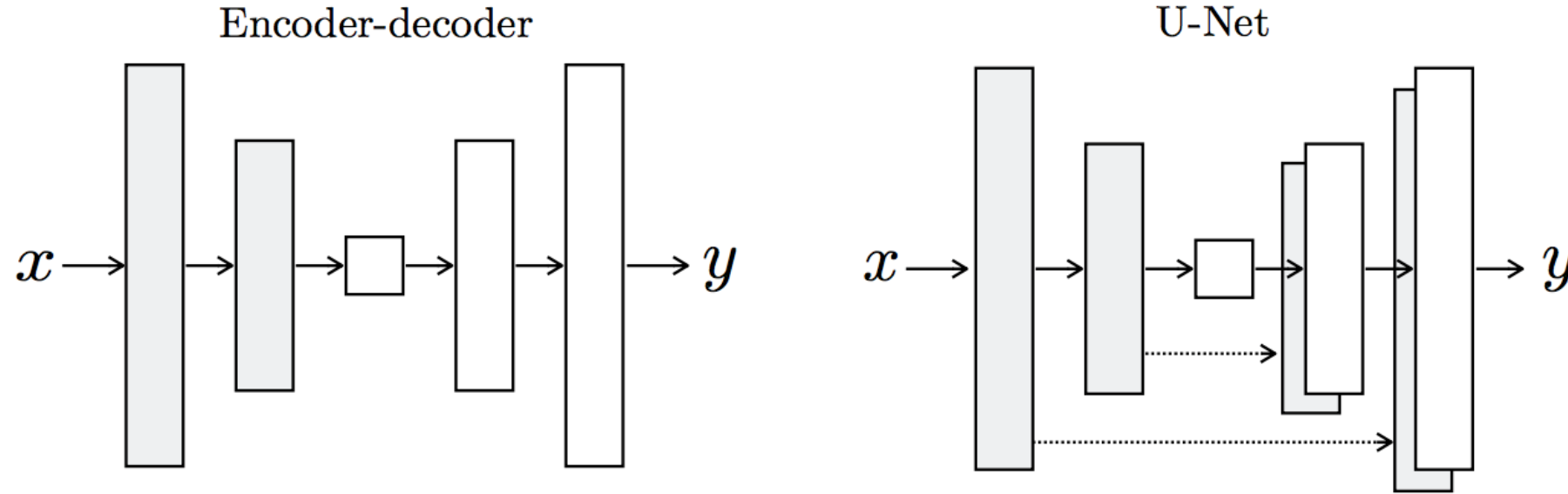
- Like FCN, fuse upsampled higher-level feature maps with higher-resolution, lower-level feature maps
- Unlike FCN, fuse by concatenation, predict at the end



O. Ronneberger, P. Fischer, T. Brox [U-Net: Convolutional Networks for Biomedical Image Segmentation](#), MICCAI 2015



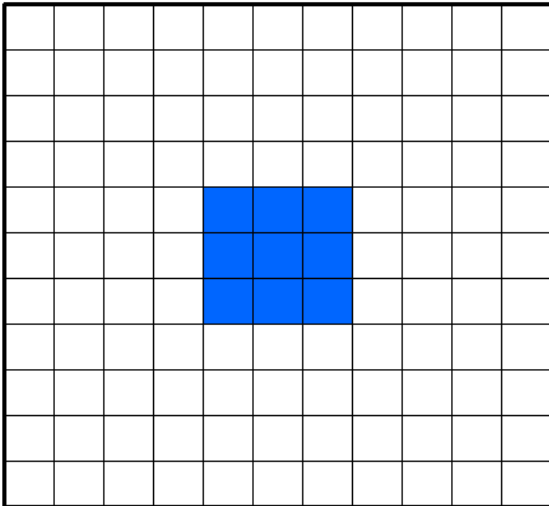
SUMMARY OF UPSAMPLING ARCHITECTURES



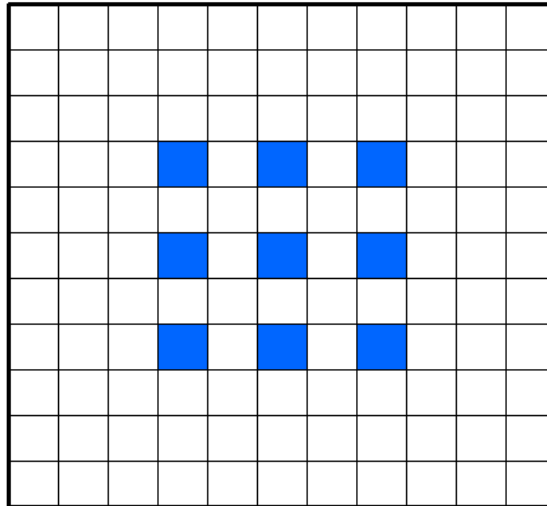
DILATED CONVOLUTIONS

- Idea: instead of reducing spatial resolution of feature maps, use a large sparse filter
- Also known as *à trous* convolution

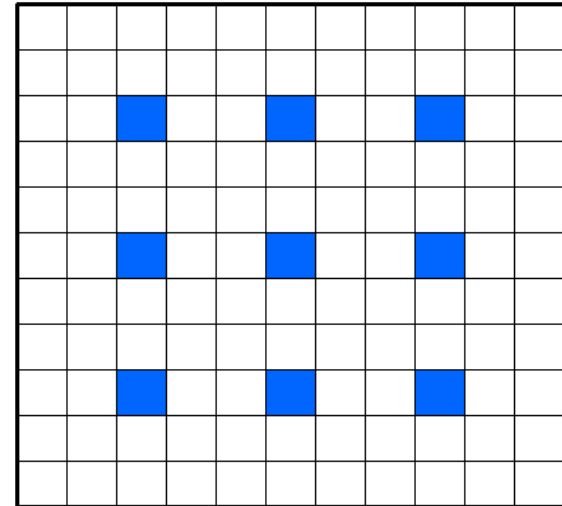
Dilation factor 1



Dilation factor 2

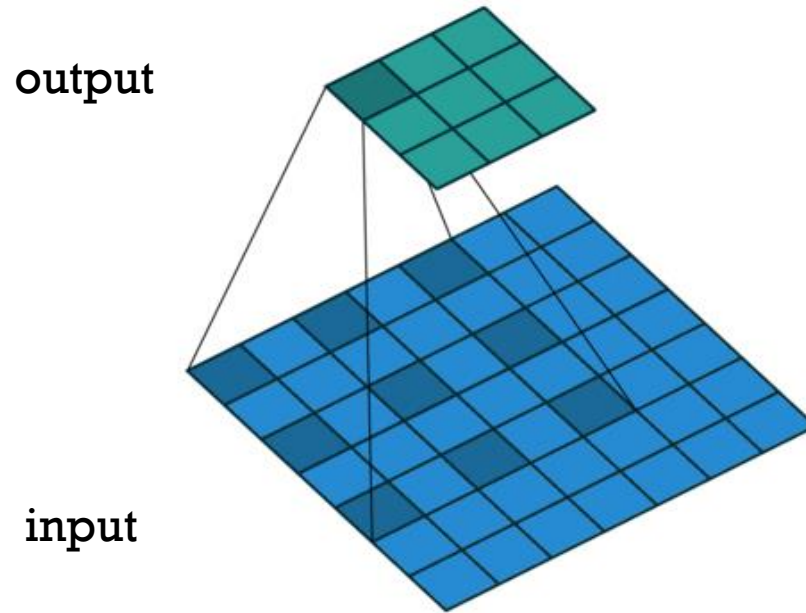


Dilation factor 3



DILATED CONVOLUTIONS

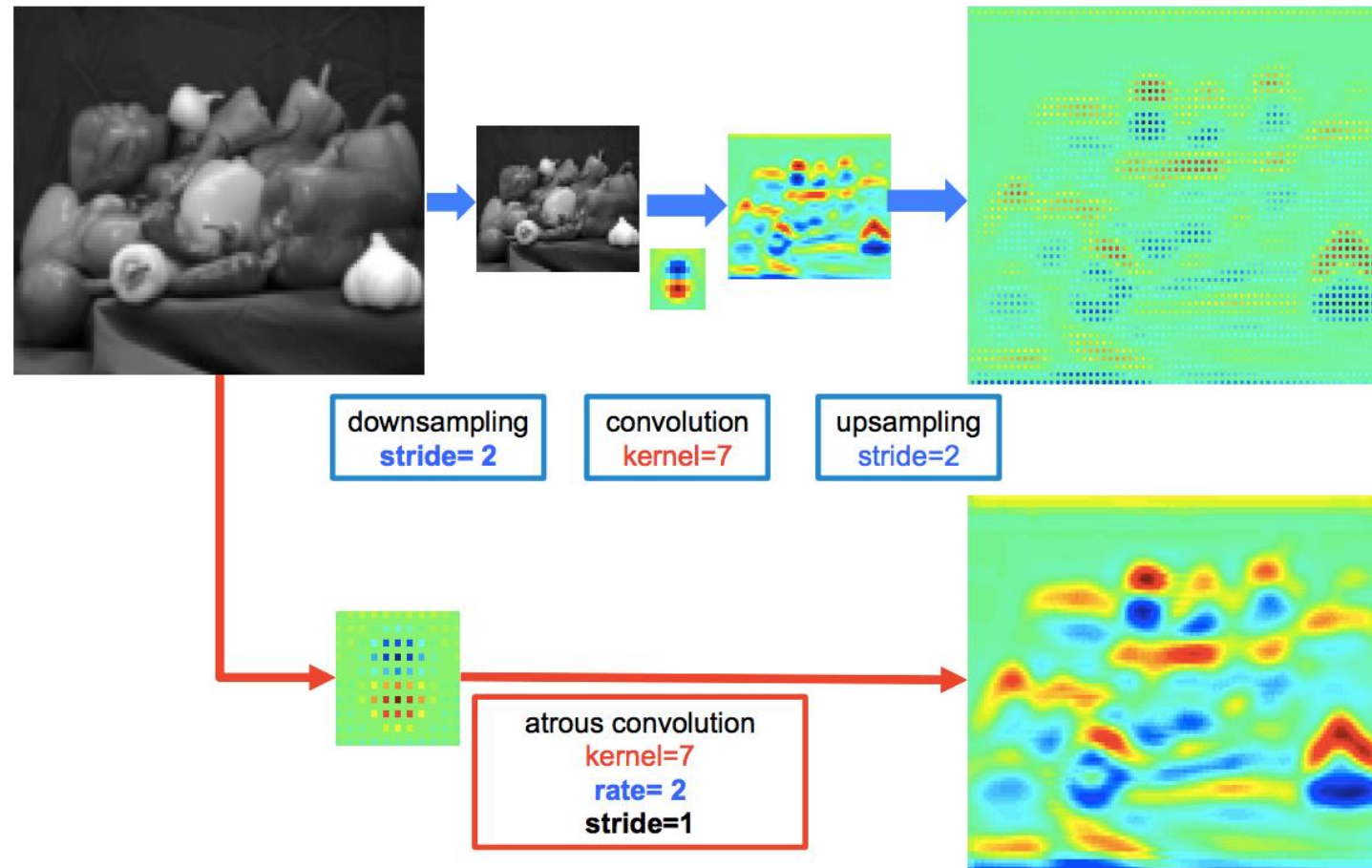
- Idea: instead of reducing spatial resolution of feature maps, use a large sparse filter



Like 2x downsampling
followed by 3x3
convolution followed by
2x upsampling



DILATED CONVOLUTIONS



DILATED CONVOLUTIONS

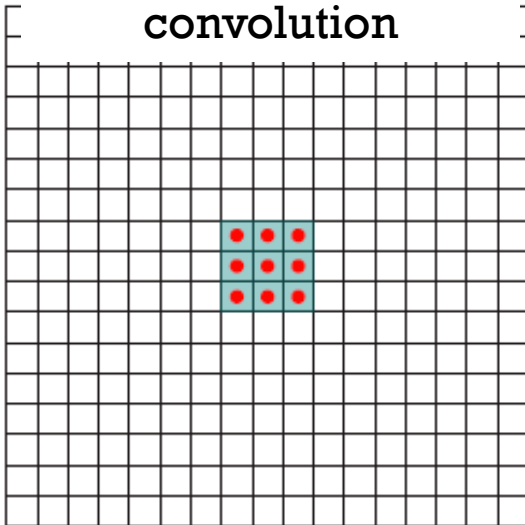
- Can be used in FCN to remove downsampling: change stride of max pooling layer from 2 to 1, dilate subsequent convolutions by factor of 2.



DILATED CONVOLUTIONS

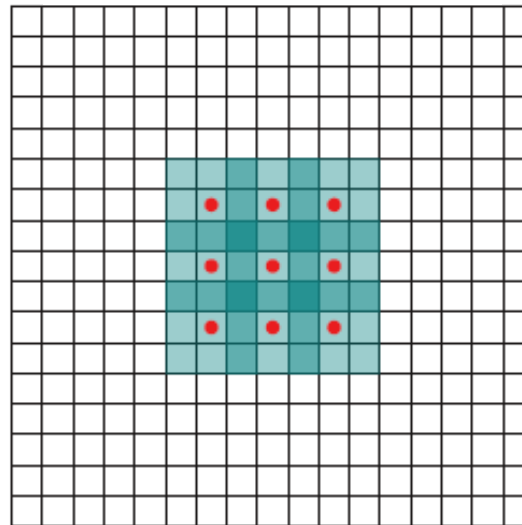
- Can increase receptive field size exponentially with a linear growth in the number of parameters

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



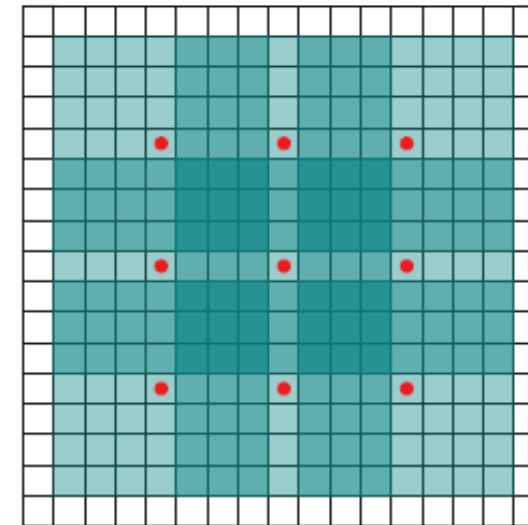
Receptive field: 3x3

F2 produced from
F1 by 2-dilated
convolution



Receptive field: 7x7

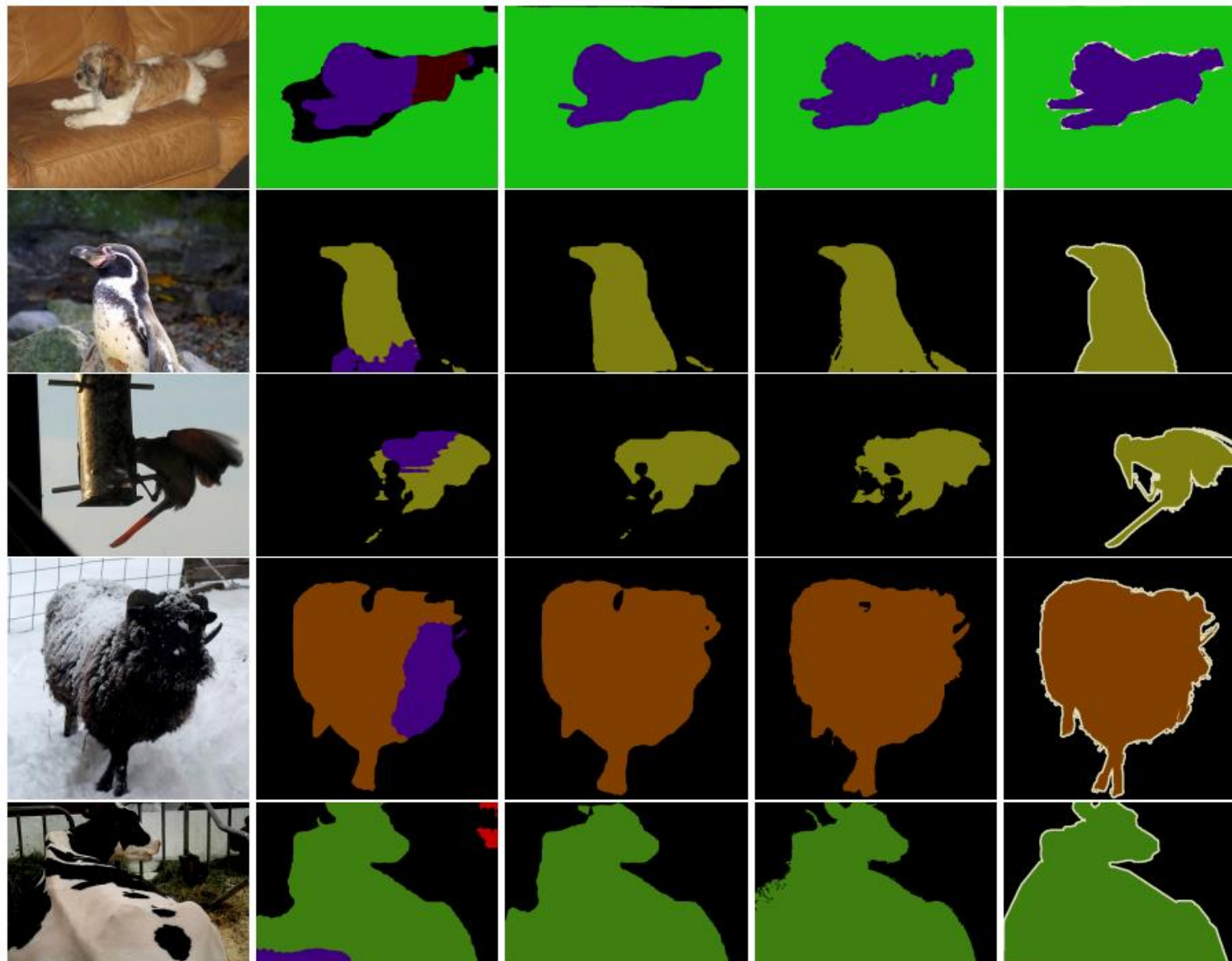
F3 produced from
F2 by 4-dilated
convolution



Receptive field:
15x15



DILATED CONVOLUTIONS: EVALUATION



(a) Image

(b) Front end

(c) + Context

(d) + CRF-RNN

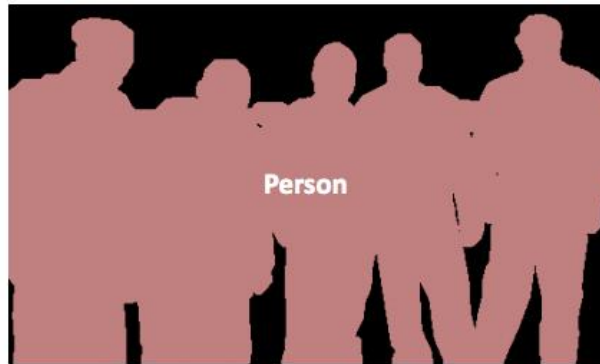
(e) Ground truth



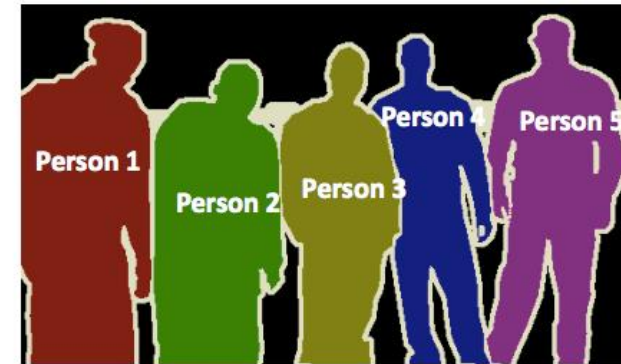
INSTANCE SEGMENTATION



Object Detection



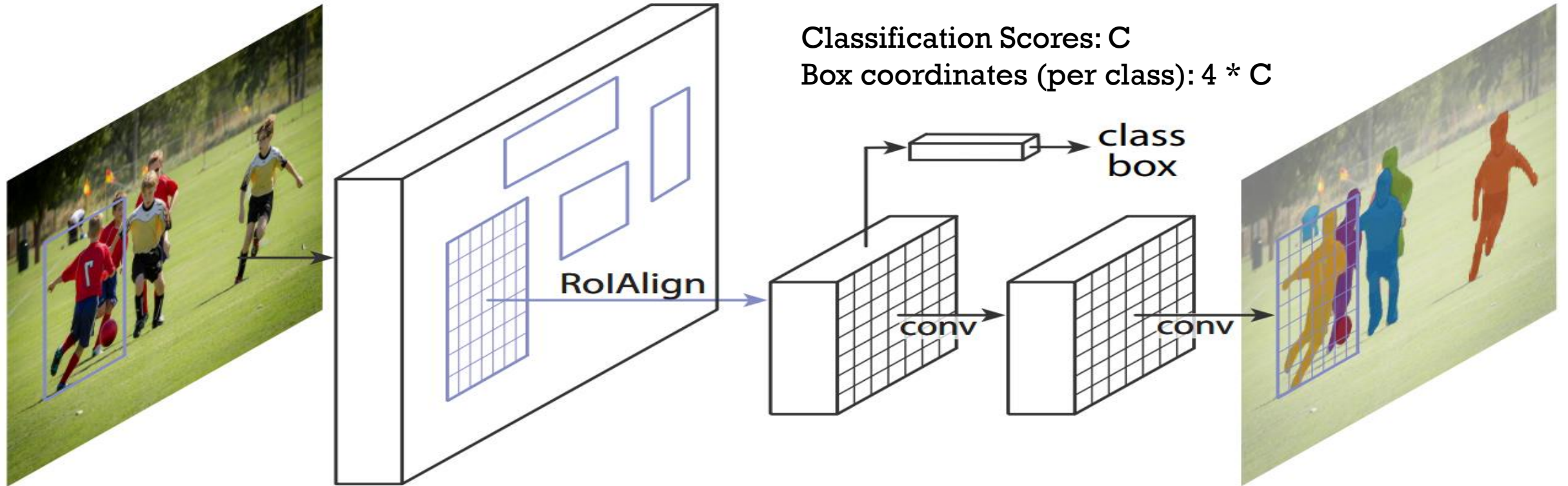
Semantic Segmentation



Instance Segmentation



Instance Segmentation: Mask R-CNN (He et al. ICCV 2017)

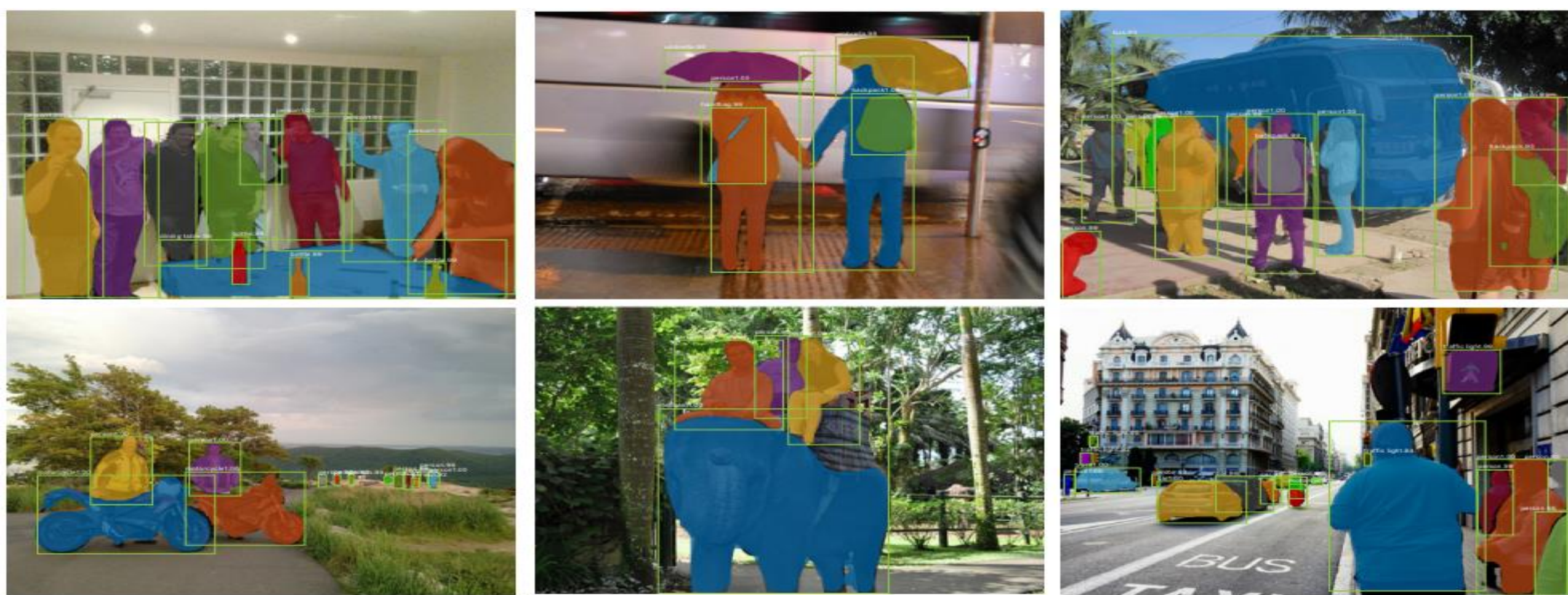


Mask R-CNN - extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition.

Mask R-CNN is simple to train and adds only a small overhead to Faster R-CNN, running at 5 fps.



Instance Segmentation: Mask R-CNN (He et al. ICCV 2017)



Mask R-CNN results on the COCO test set. These results are based on ResNet-101, achieving a mask AP of 35.7 and running at 5 fps. Masks are shown in color, and bounding box, category, and confidences are also shown.

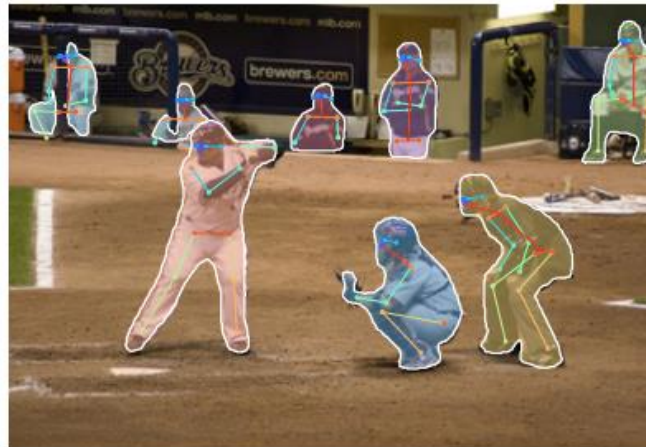
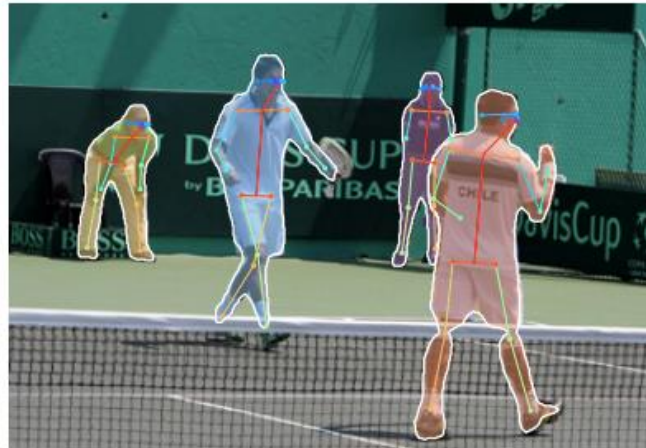


Other dense prediction problems



KEYPOINT PREDICTION

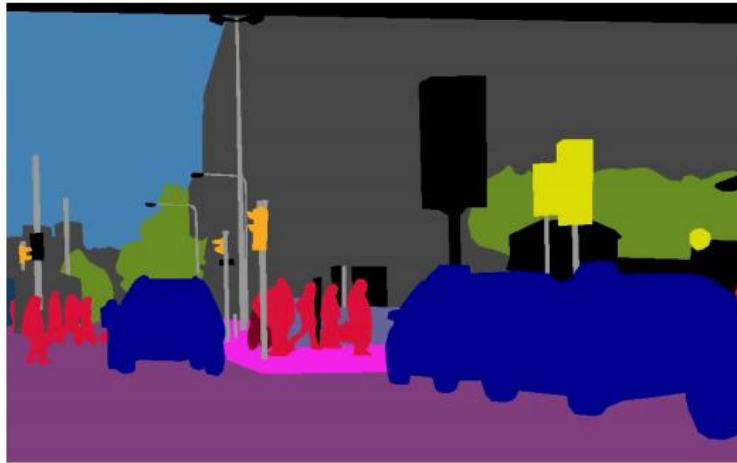
- Given K keypoints, train model to predict K $m \times m$ one-hot maps with cross-entropy losses over m^2 outputs



PANOPTIC SEGMENTATION



(a) image



(b) semantic segmentation



(c) instance segmentation



(d) panoptic segmentation



PANOPTIC SEGMENTATION



Figure 2: Panoptic FPN results on COCO (top) and Cityscapes (bottom) using a single ResNet-101-FPN network.

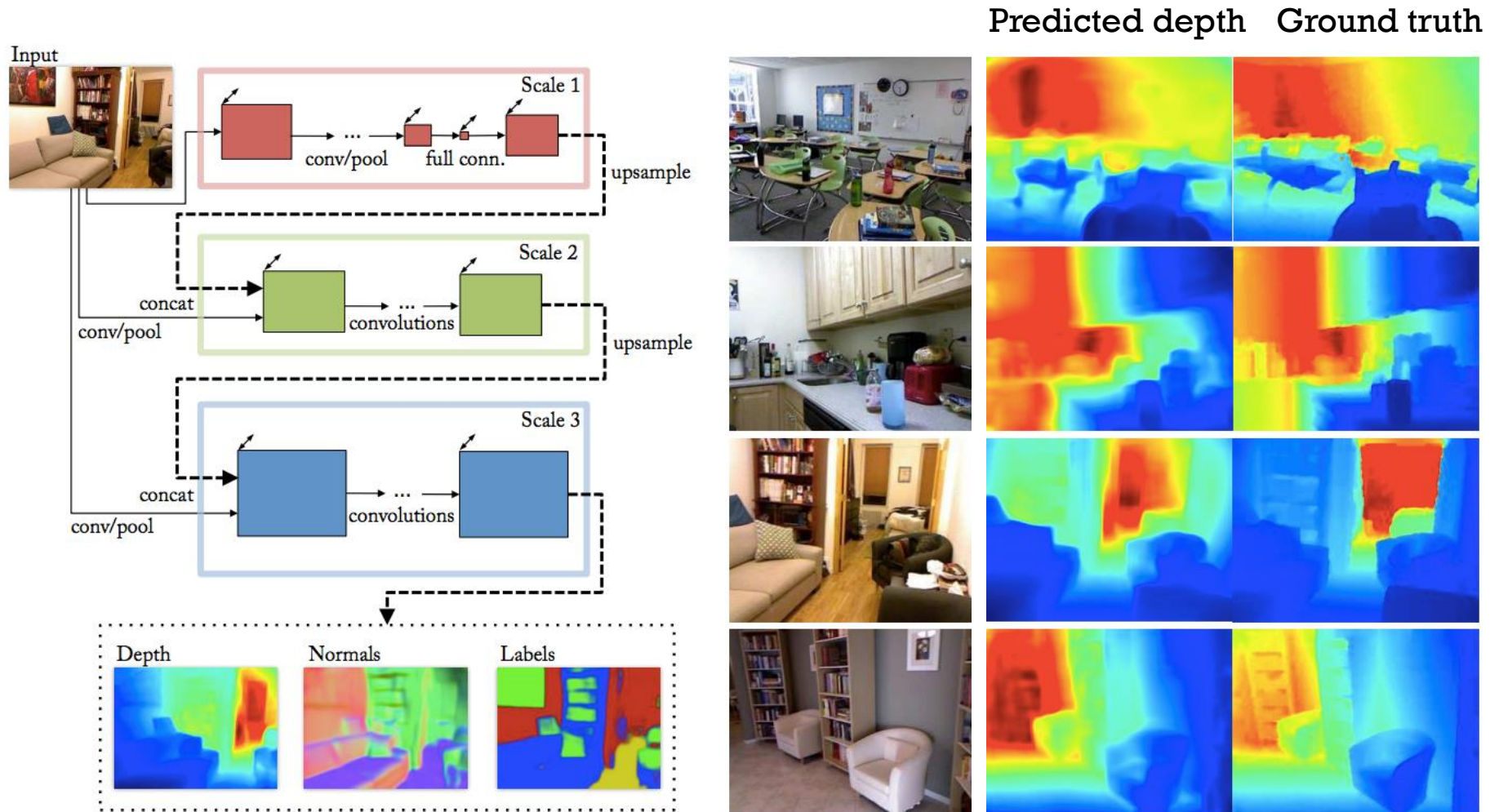


EVEN MORE DENSE PREDICTION PROBLEMS

- Depth estimation
- Surface normal estimation
- Colorization
-



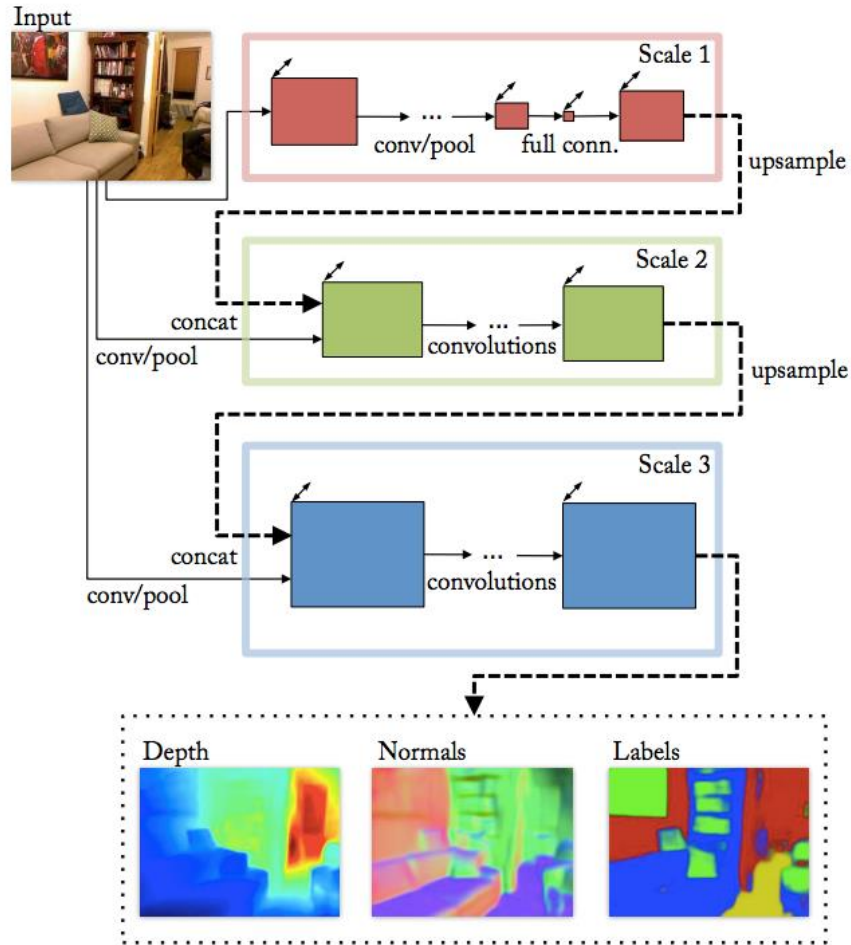
DEPTH AND NORMAL ESTIMATION



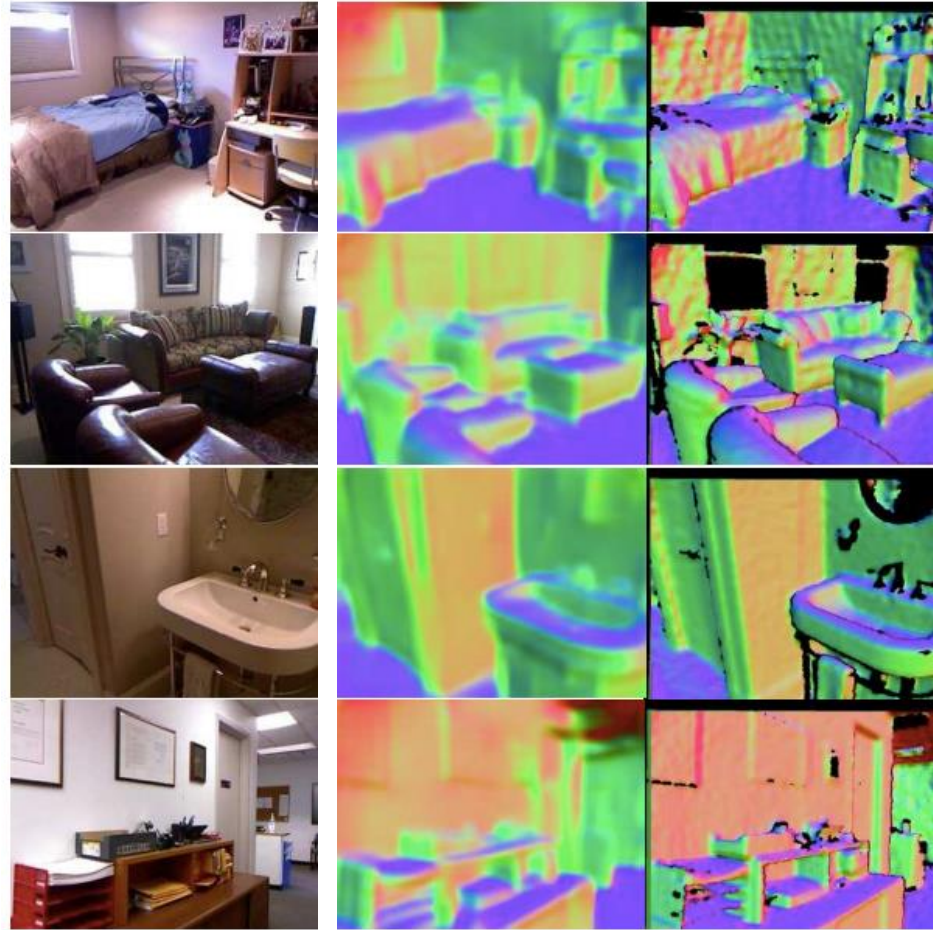
D. Eigen and R. Fergus, Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture, ICCV 2015



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



ESTIMATION OF EVERYTHING AT THE SAME TIME

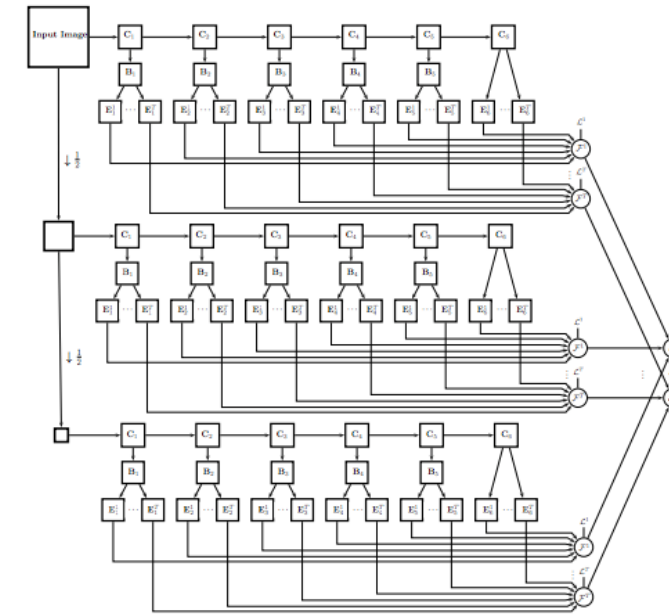
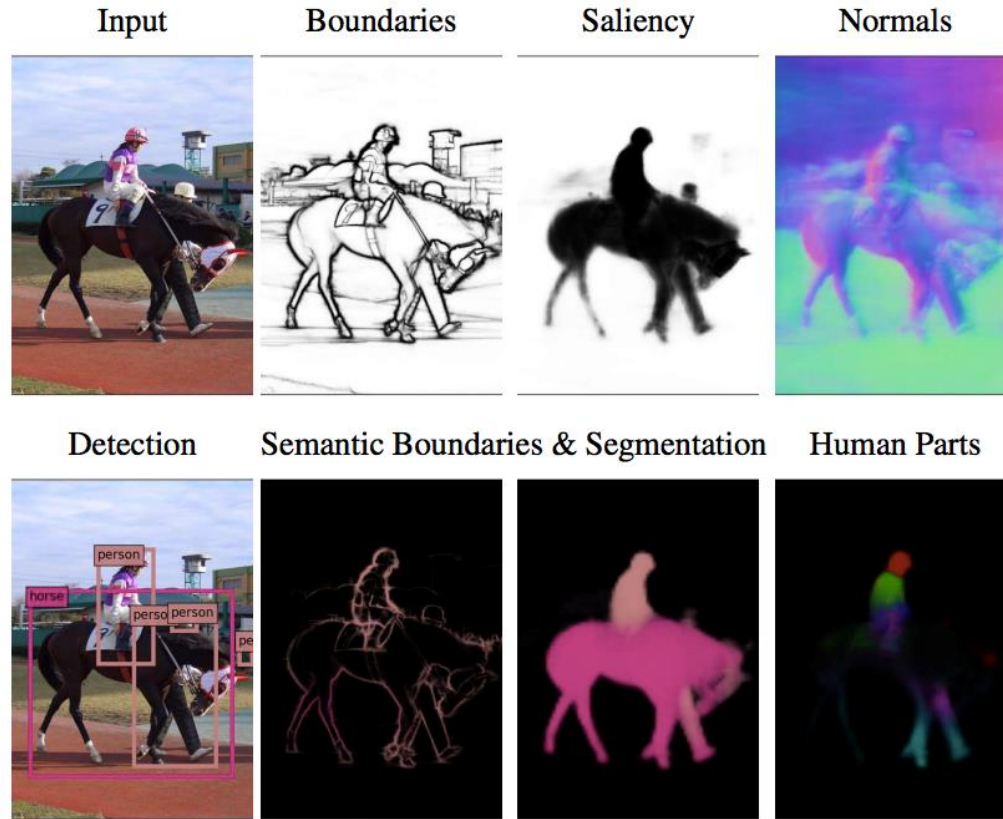
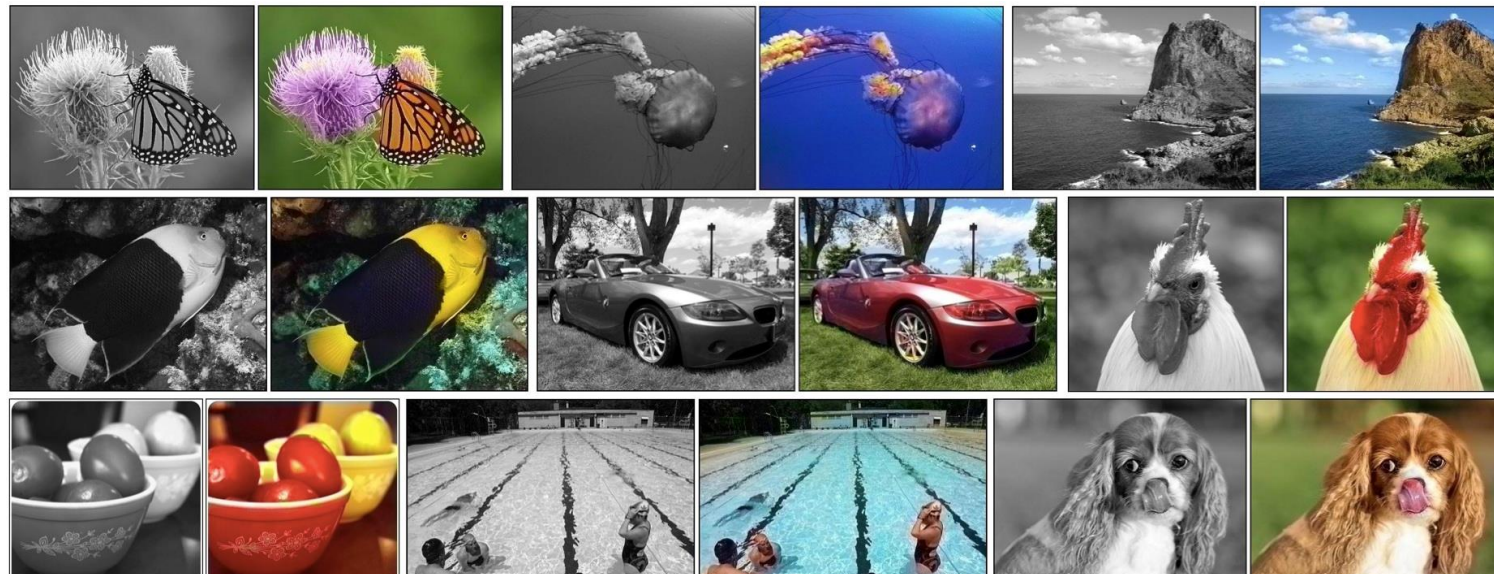
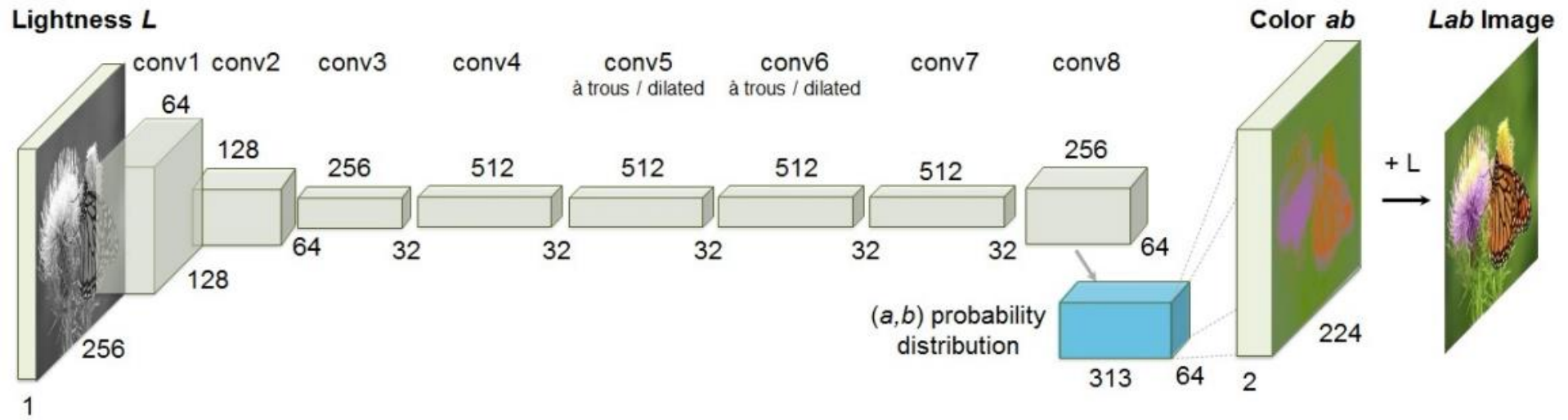


Figure 2: UberNet architecture: an image pyramid is formed by successive down-sampling operations, and each image is processed by a CNN with tied weights; the responses of the network at consecutive layers (C_i) are processed with Batch Normalization (B_i) and then fed to task-specific skip layers (E_i^t); these are combined across network layers ($\mathcal{F}^t$) and resolutions ($\mathcal{S}^t$) and trained using task-specific loss functions ($\mathcal{L}^t$), while the whole architecture is jointly trained end-to-end. For simplicity we omit the interpolation and detection layers mentioned in the text.

I. Kokkinos, UberNet: Training a Universal Convolutional Neural Network for Low-, Mid-, and High-Level Vision using Diverse Datasets and Limited Memory,
ICCV 2017

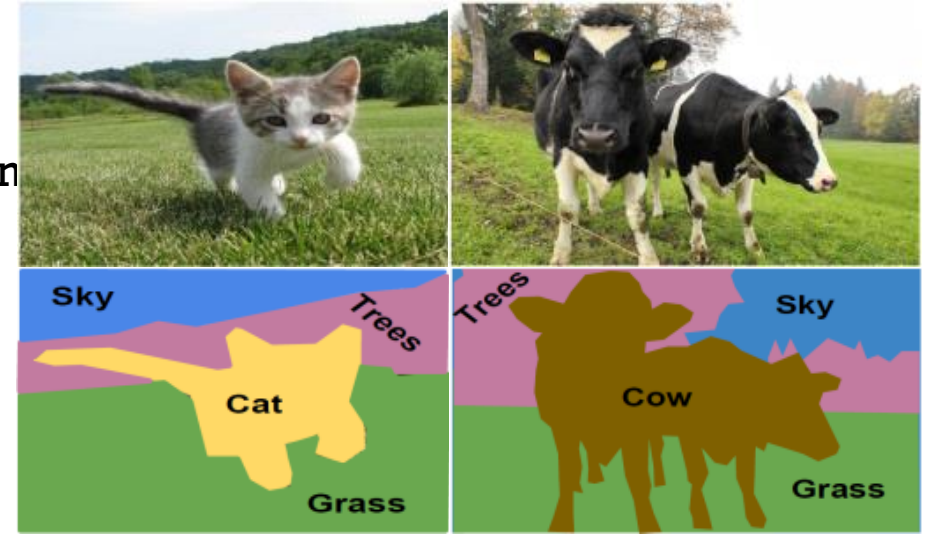


COLORIZATION



THINGS TO REMEMBER

- Semantic Segmentation
 - Sliding window inefficient
 - Fully Convolutional: Convolutions at original image resolution is very expensive
 - Fully Convolutional: Down sampling and Up sampling
 - Unsampling using Unpooling and Transpose Convolution
 - DeconvNet: deeper network
- Instance Segmentation
 - Mask-RCNN: Extension of faster RCNN



ACKNOWLEDGEMENT

- [Deep Learning, Stanford University](#)
- [Introduction to Deep Learning, University of Illinois at Urbana-Champaign](#)
- [Introduction to Deep Learning, Carnegie Mellon University](#)
- [Convolutional Neural Networks for Visual Recognition, Stanford University](#)
- [Natural Language Processing with Deep Learning, Stanford University](#)
- [NVDIEA Deep Learning Teaching Kit](#)
- [And many more...](#)

