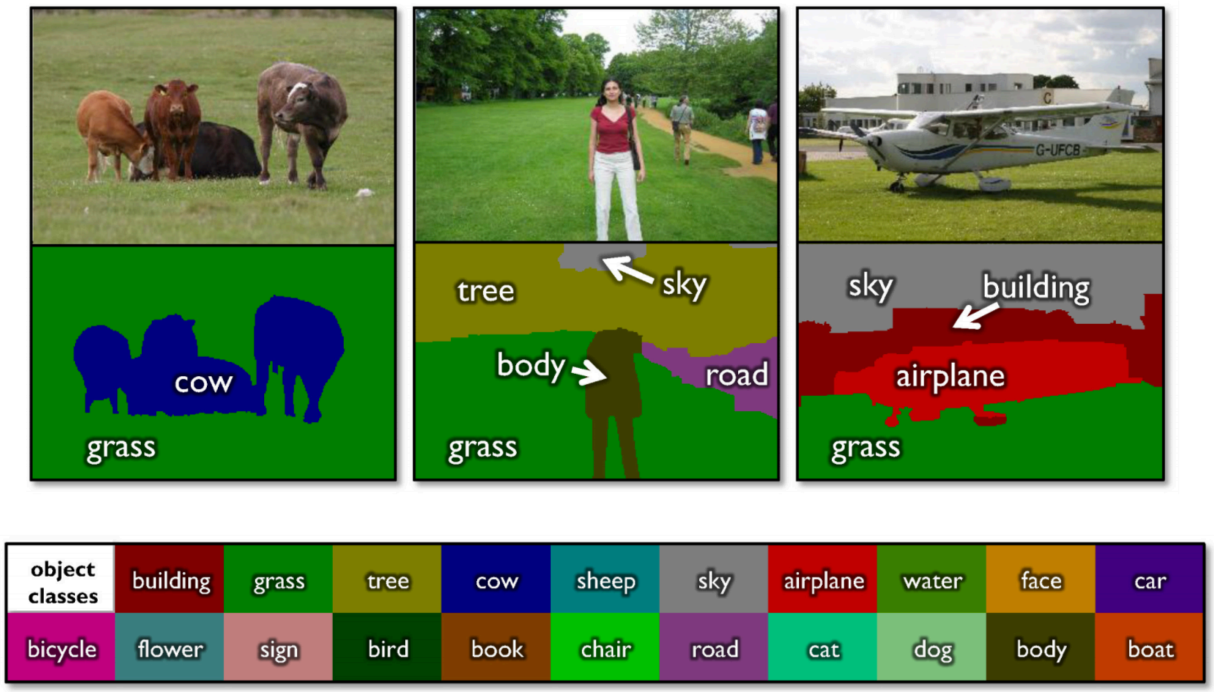
Semantic Segmentation, Dense Labeling

Liwei Wang

CS@UIUC

Semantic segmentation



From CS543 LAZ

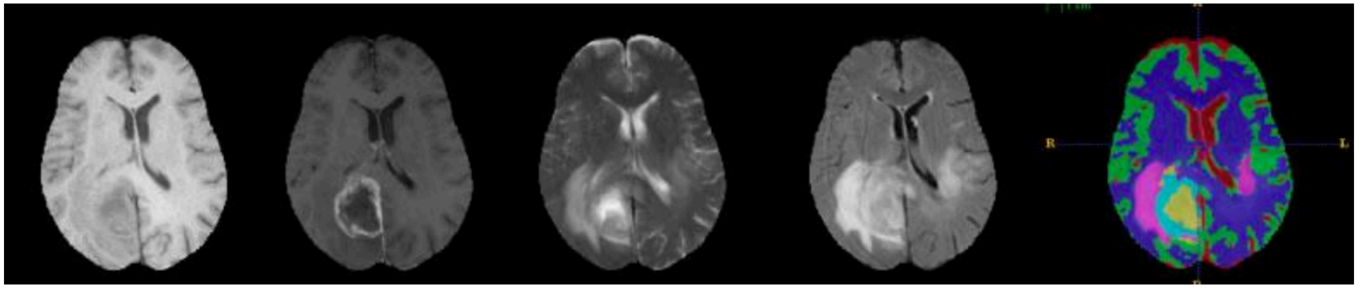
Why Semantic Segmentation ?

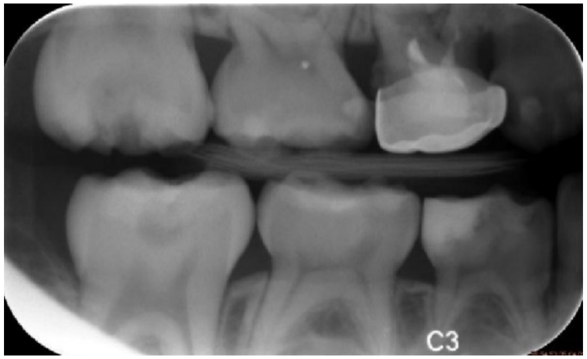
- Road Scene Understanding
- Useful for Self-Driving Car and autonomous drones

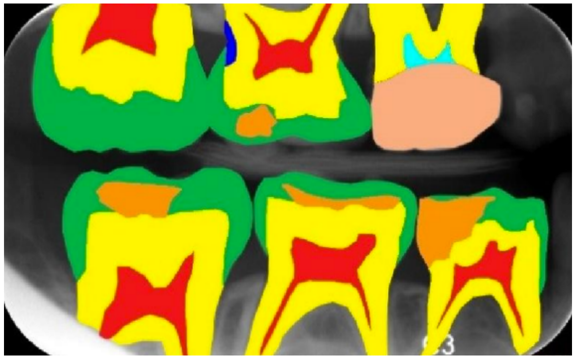
From cityscape dataset

Why Semantic Segmentation ?

- Medical Image Analysis







From web

Very Challenging Problem

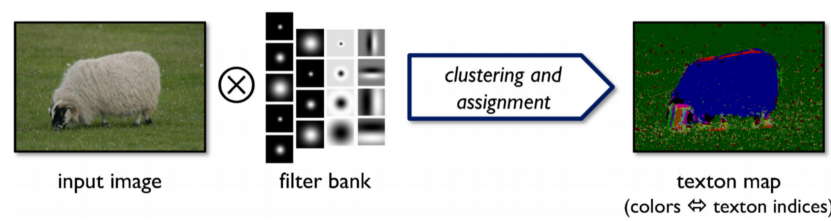
History

- Problem: label each pixel by one of C classes
- Define an energy function where unaries correspond to local classifier responses and smoothing potentials correspond to contextual terms
- Solve a multi-class graph cut problem

$$\log P(\mathbf{c}|\mathbf{x}, \boldsymbol{\theta}) =$$

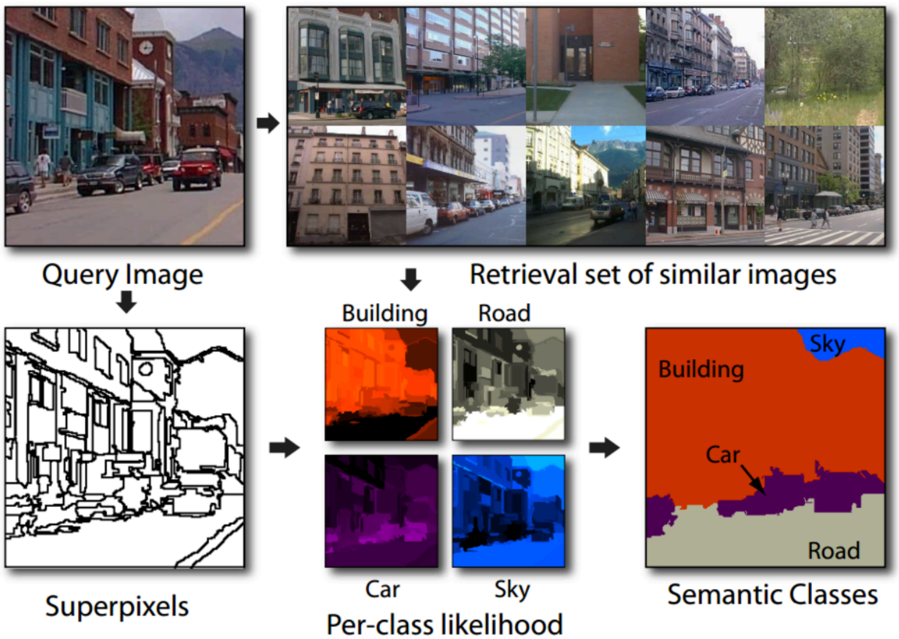
$$\sum_i \overbrace{\psi_i(c_i, \mathbf{x}; \boldsymbol{\theta}_\psi)}^{\text{texture-layout}} + \overbrace{\pi(c_i, x_i; \boldsymbol{\theta}_\pi)}^{\text{color}} + \overbrace{\lambda(c_i, i; \boldsymbol{\theta}_\lambda)}^{\text{location}}$$

$$+ \sum_{(i,j)\in\mathcal{E}} \overbrace{\phi(c_i, c_j, \mathbf{g}_{ij}(\mathbf{x}); \boldsymbol{\theta}_\phi)}^{\text{edge}} - \log Z(\boldsymbol{\theta}, \mathbf{x}) \quad (1)$$



From TextonBoost, ECCV 2006

History



CRF energy function is defined on super-pixels:

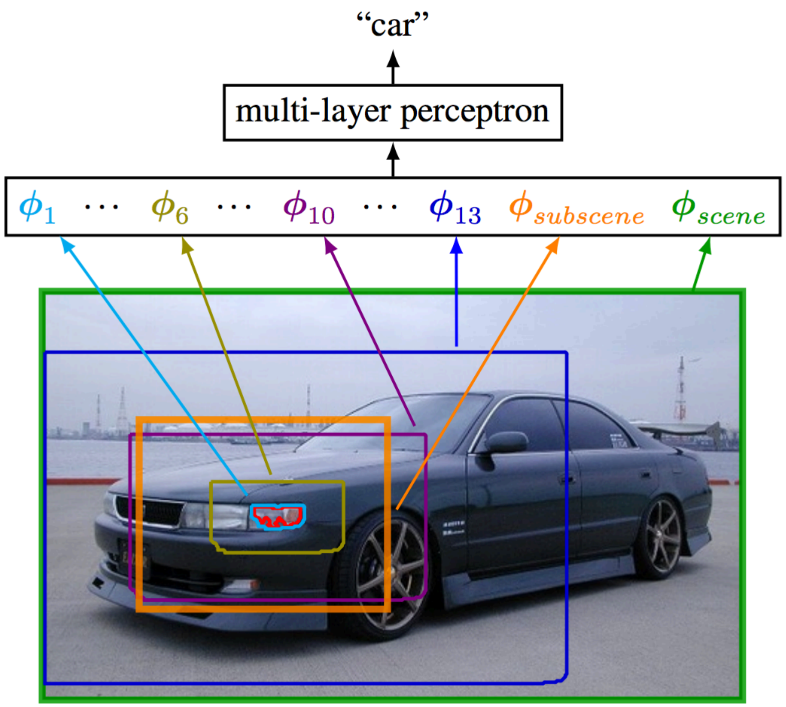
- Unaries are based on nearest neighbor retrieval
- Pairwise potentials capture class co-occurrence statistics

J. Tighe and S. Lazebnik, SuperParsing, ECCV 2010

Now, what is happening

- How deep neural networks can be used for Semantic Segmentation ?
- How to model local and contextual information with Deep Nets ?
- Differences and Similarities among methods ?

Zoom-out Features

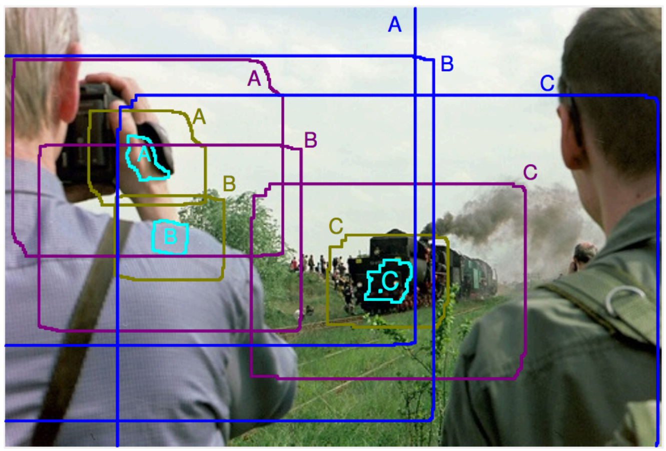


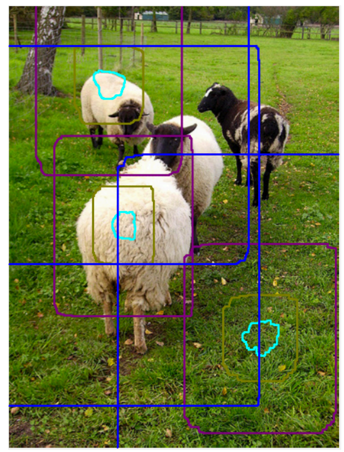
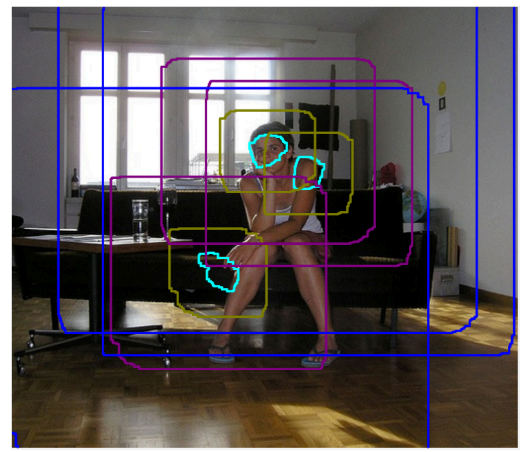
- A simple Feed-Forward Network
- Feature Concatenation from Different Scales
- Strong features + Softmax Classification

Zoom-out Features

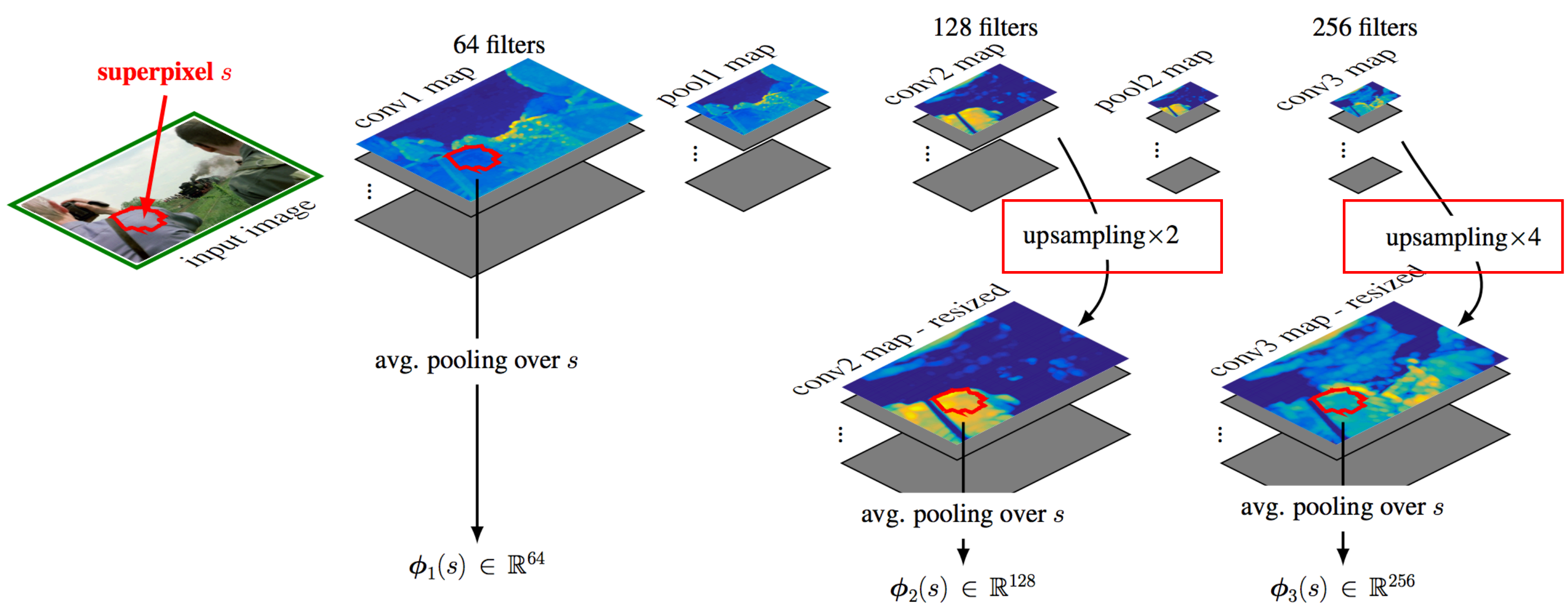
- Sub-scene Level Features
  - Bounding box of superpixels within radius three from the superpixel at hand
  - Warp bounding box to 256 x 256 pixels
  - Activations of the last fully connected layer
- Scene Level Features
  - Warp image to 256 x 256 pixels
  - Activations of the last fully connected layer

Zoom-out Features



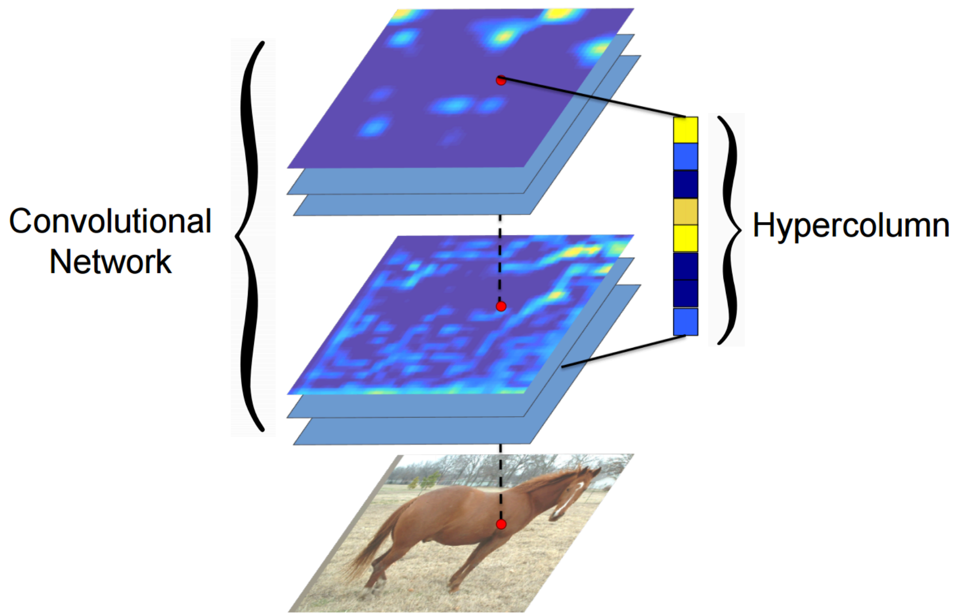



Zoom-out Features



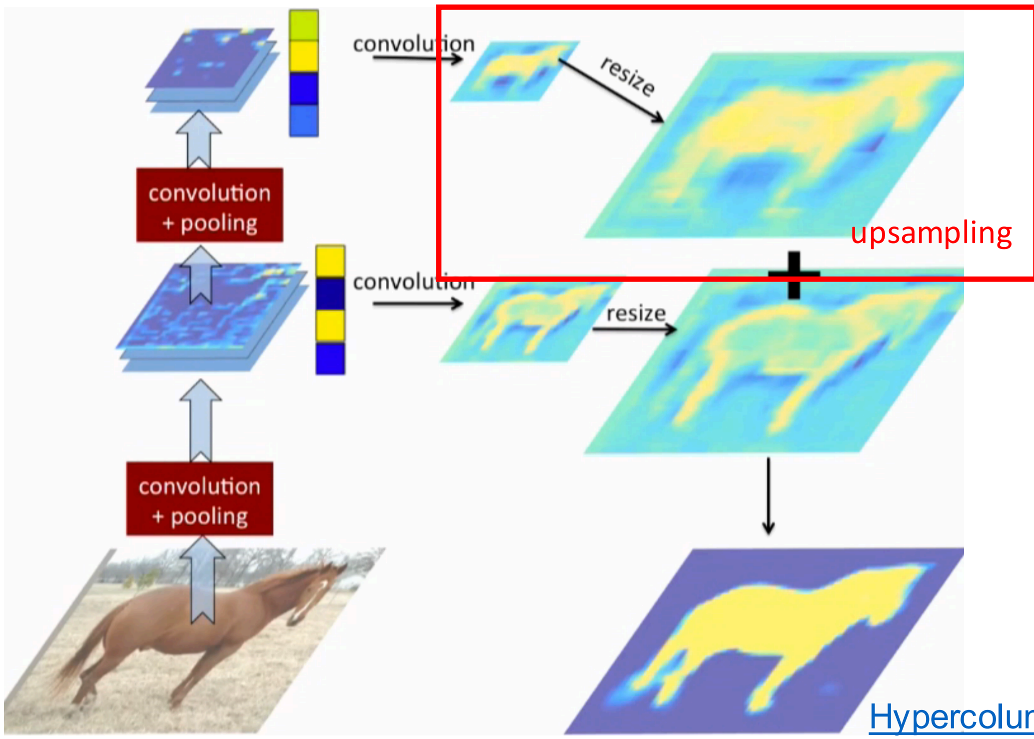
[Feedforward Semantic Segmentation With Zoom-Out Features, CVPR 2015]

Hypercolumns Representation



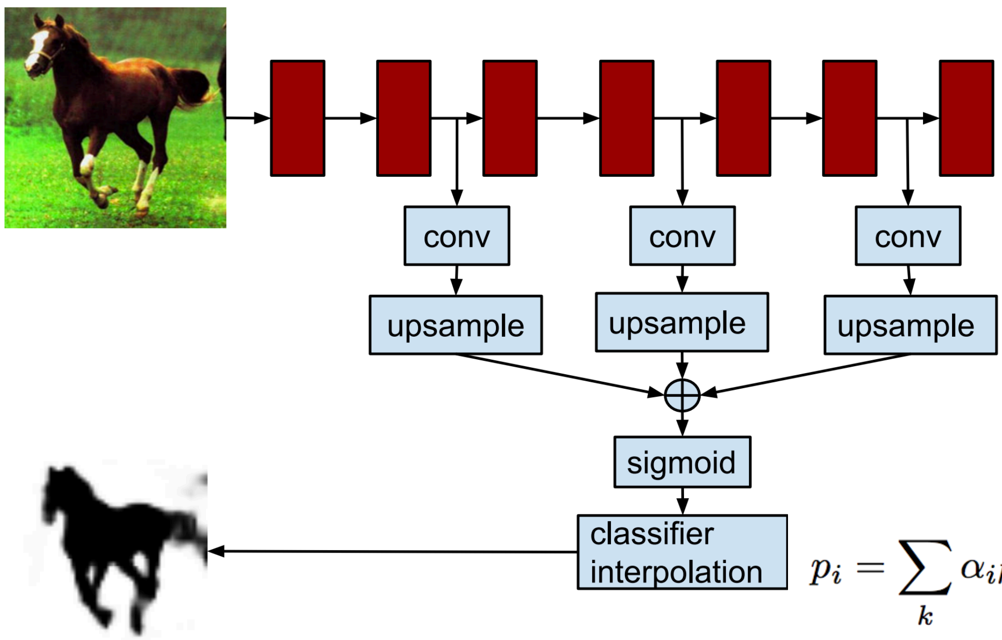
[Hypercolumns for object segmentation and fine-grained localization](#) CVPR 2015

Hypercolumn Representation



Hypercolumns for object segmentation and fine-grained localization CVPR 2015

Hypercolumn Representation



$$p_i = \sum_k \alpha_{ik} g_k(\mathbf{f}_i) = \sum_k \alpha_{ik} p_{ik}$$

Hypercolumns for object segmentation and fine-grained localization CVPR 2015

Evaluation

- Mean IoU

  Per-class evaluation: an intersection of the predicted and true sets of pixels for a given class, divided by their union (IoU)

$$\text{seg. accuracy} = \frac{\text{true pos.}}{\text{true pos.} + \text{false pos.} + \text{false neg.}}$$

	VOC 2012
Zoom-out	69.6
Hypercolumn	62.6

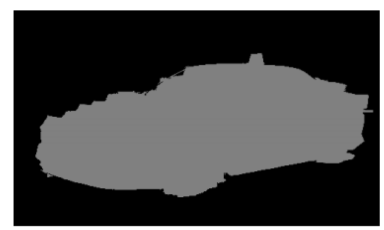
Evaluation

Zoom-out Method on Pascal VOC2012

class	mean	bg																				
acc	69.6	91.9	85.6	37.3	83.2	62.5	66	85.1	80.7	84.9	27.2	73.3	57.5	78.1	79.2	81.1	77.1	53.6	74	49.2	71.7	63.3





Hypercolumn and Zoom out

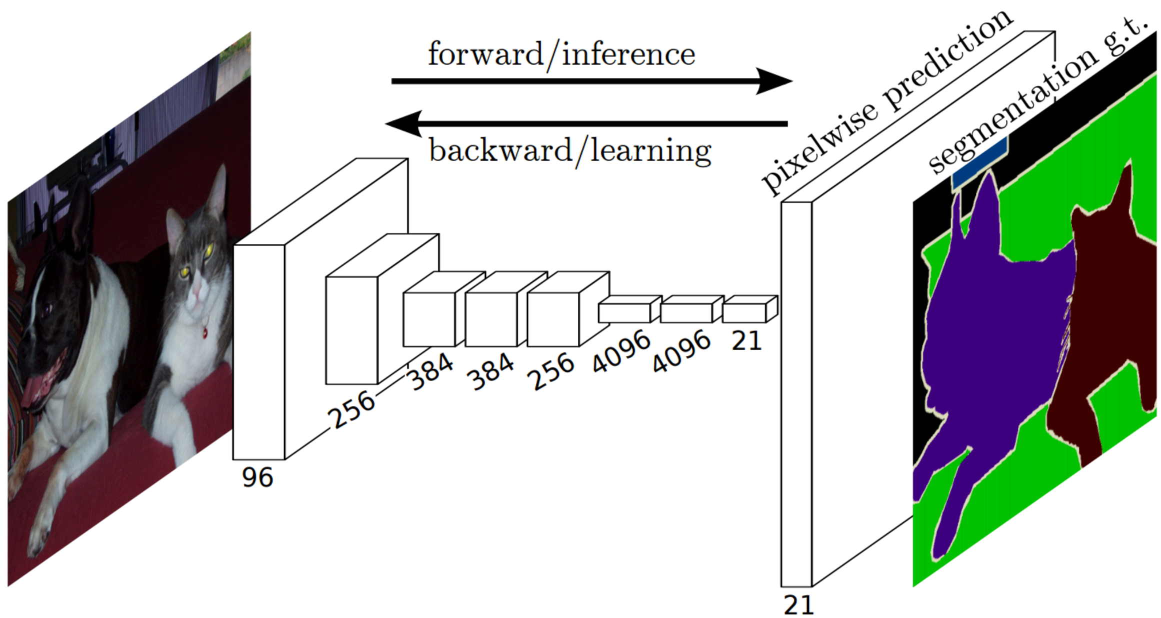
- Both uses the multi-scale features from intermediate layers in CNN

- Both use upsampling operations for each scale

Any Pixel to Pixel ways ?

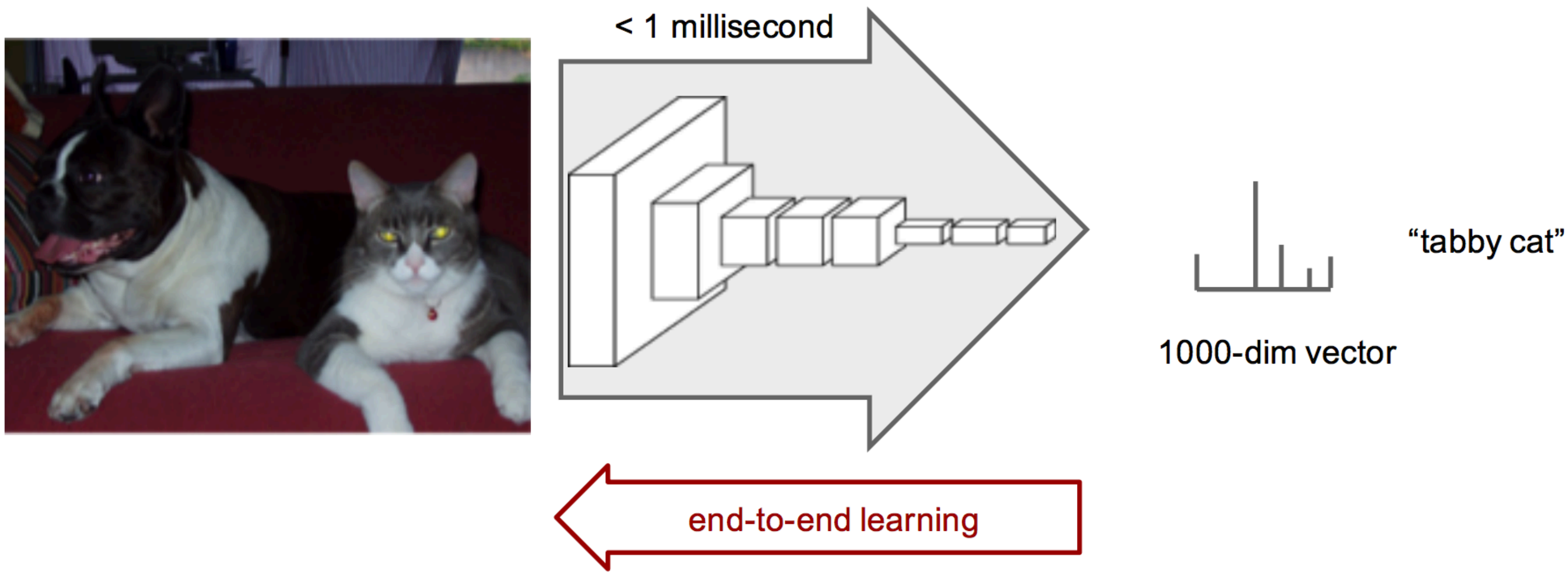
Can upsampling be learned ?

FCN for Semantic Segmentation



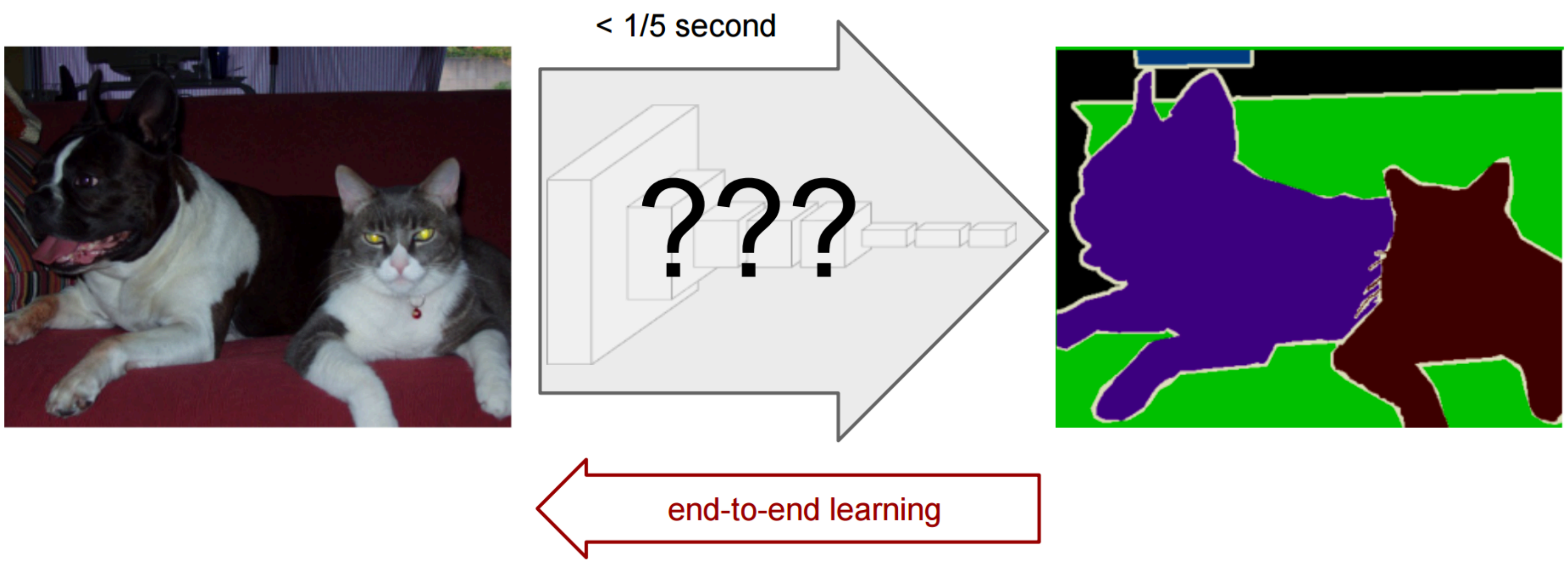
Fully convolutional Networks for Semantic Segmentation, CVPR 2015

Convnets for Classification



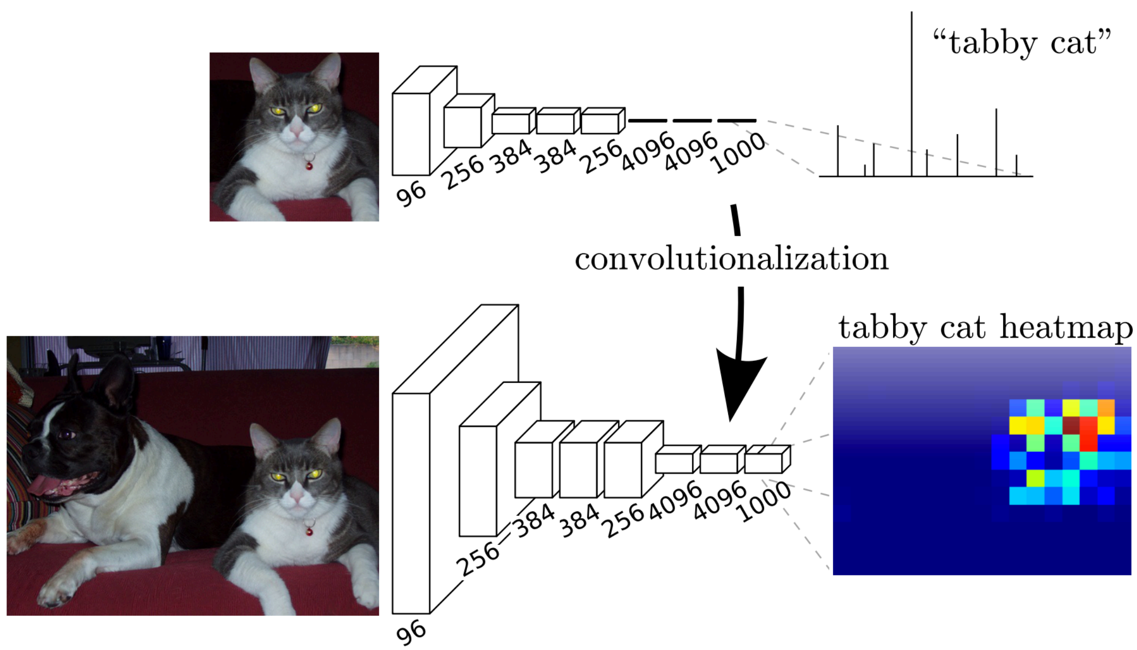
Fully convolutional Networks for Semantic Segmentation, CVPR 2015

Convnets for Segmentation ?



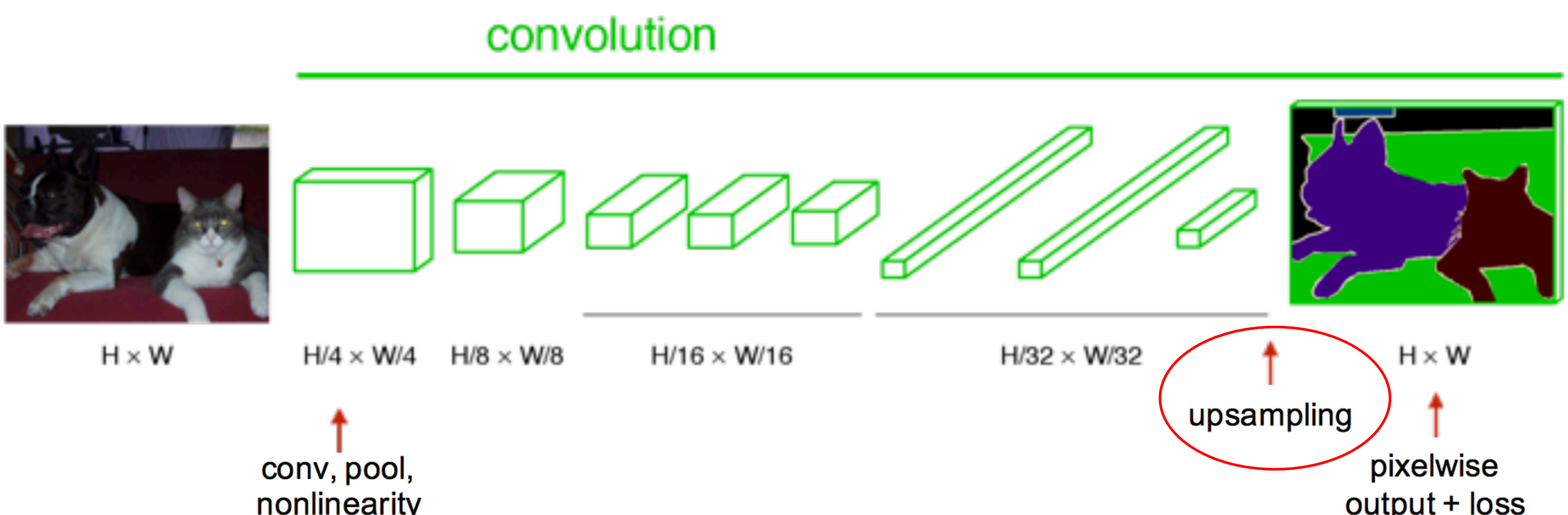
Fully convolutional Networks for Semantic Segmentation, CVPR 2015

From Convnets to FCN



Fully convolutional Networks for Semantic Segmentation, CVPR 2015

Pixel in , Pixel out

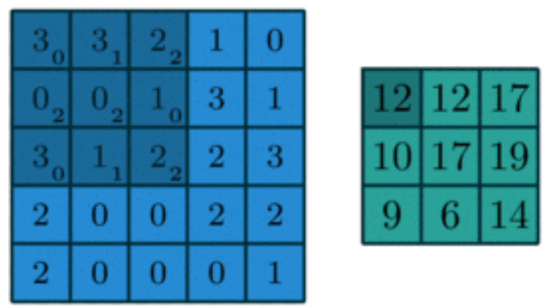


Upsampling is backwards strided convolution

How does convolution works ?

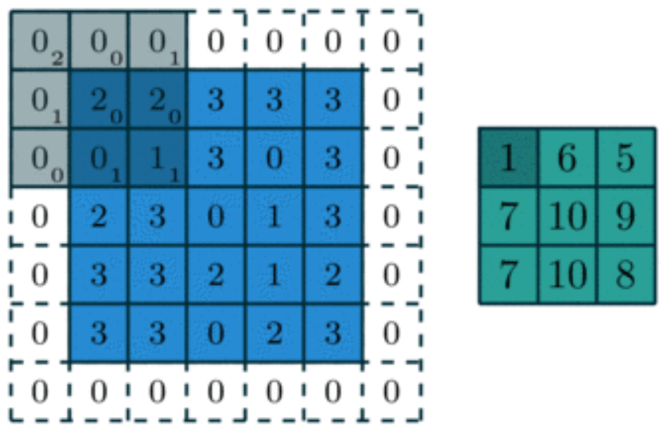
Kernel/filter:

$$\begin{pmatrix} 0 & 1 & 2 \\ 2 & 2 & 0 \\ 0 & 1 & 2 \end{pmatrix}$$

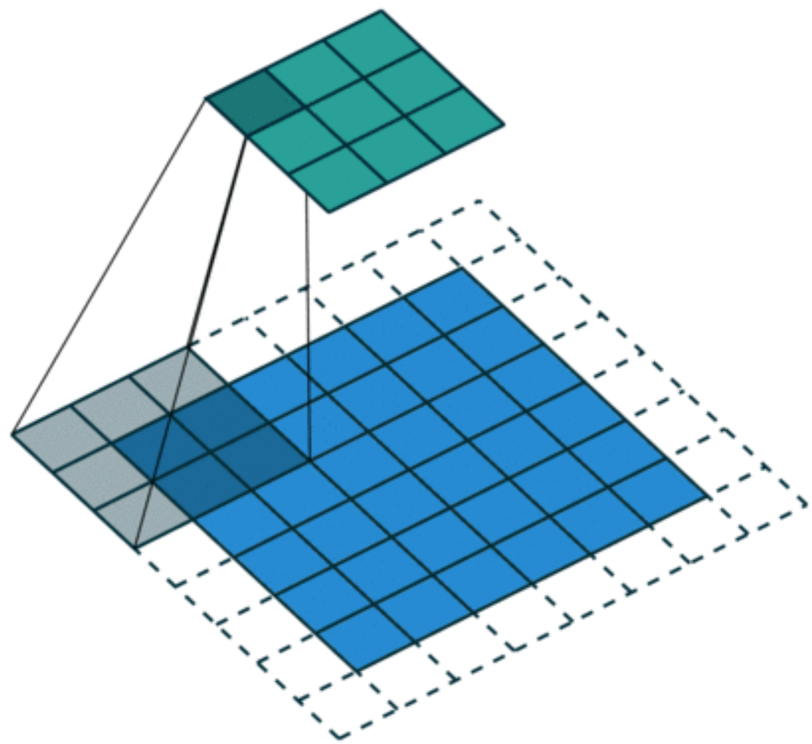


How does convolution works ?

Kernel/filter:

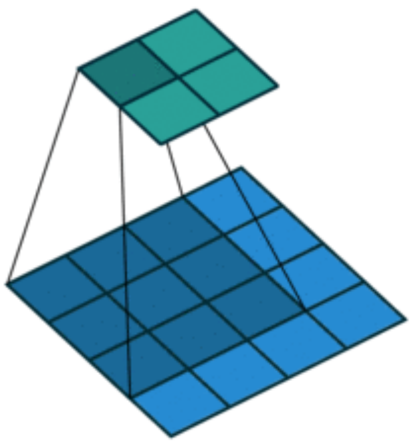


How does convolution works ?



How does convolution works ?

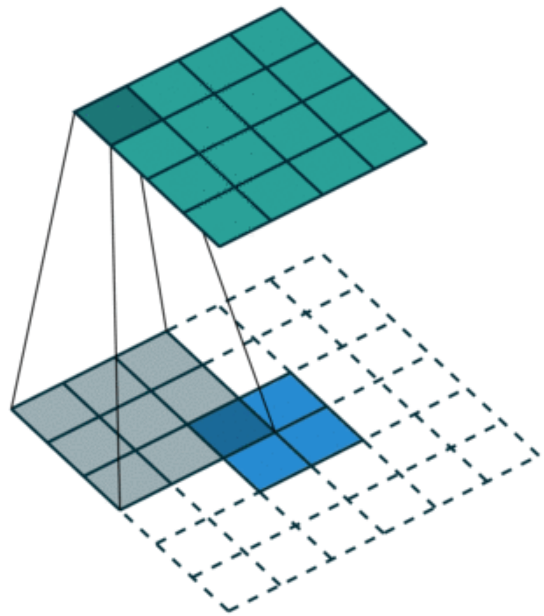
Convolution as a matrix operation



$$
\begin{pmatrix}
w_{0,0} & 0 & 0 & 0 \\
w_{0,1} & w_{0,0} & 0 & 0 \\
w_{0,2} & w_{0,1} & 0 & 0 \\
0 & w_{0,2} & 0 & 0 \\
w_{1,0} & 0 & w_{0,0} & 0 \\
w_{1,1} & w_{1,0} & w_{0,1} & w_{0,0} \\
w_{1,2} & w_{1,1} & w_{0,2} & w_{0,1} \\
0 & w_{1,2} & 0 & w_{0,2} \\
w_{2,0} & 0 & w_{1,0} & 0 \\
w_{2,1} & w_{2,0} & w_{1,1} & w_{1,0} \\
w_{2,2} & w_{2,1} & w_{1,2} & w_{1,1} \\
0 & w_{2,2} & 0 & w_{1,2} \\
0 & 0 & w_{2,0} & 0 \\
0 & 0 & w_{2,1} & w_{2,0} \\
0 & 0 & w_{2,2} & w_{2,1} \\
0 & 0 & 0 & w_{2,2}
\end{pmatrix}^T
$$

From Theano Document Website

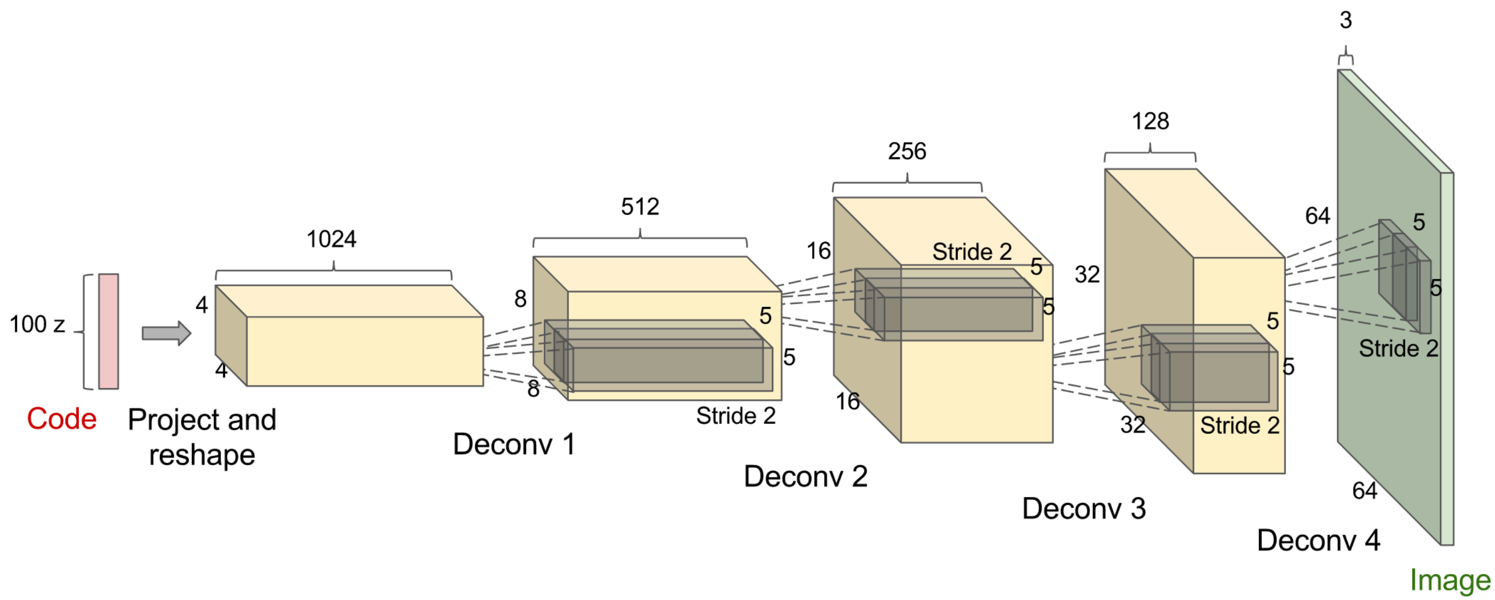
Upsampling is backwards strided convolution



Transposed convolution

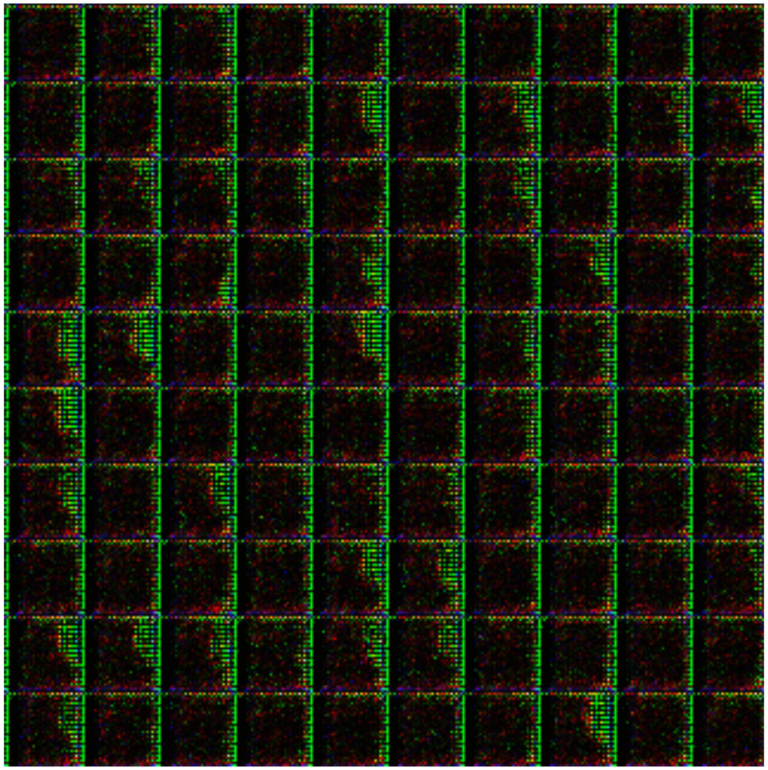
From Theano Document Website

Actually, we can build deconv networks…

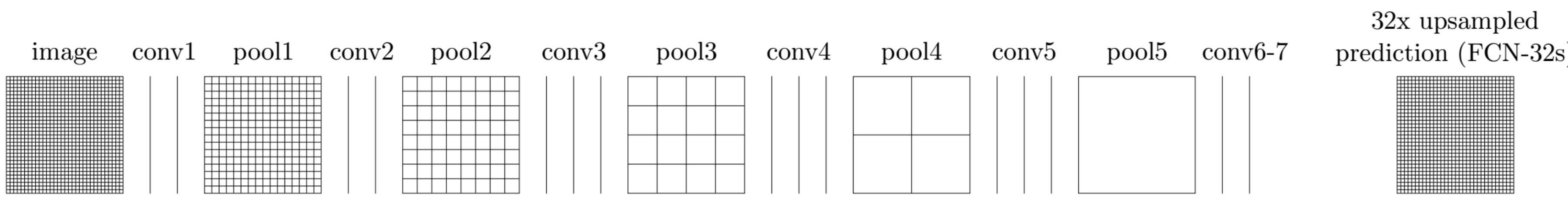


DCGAN

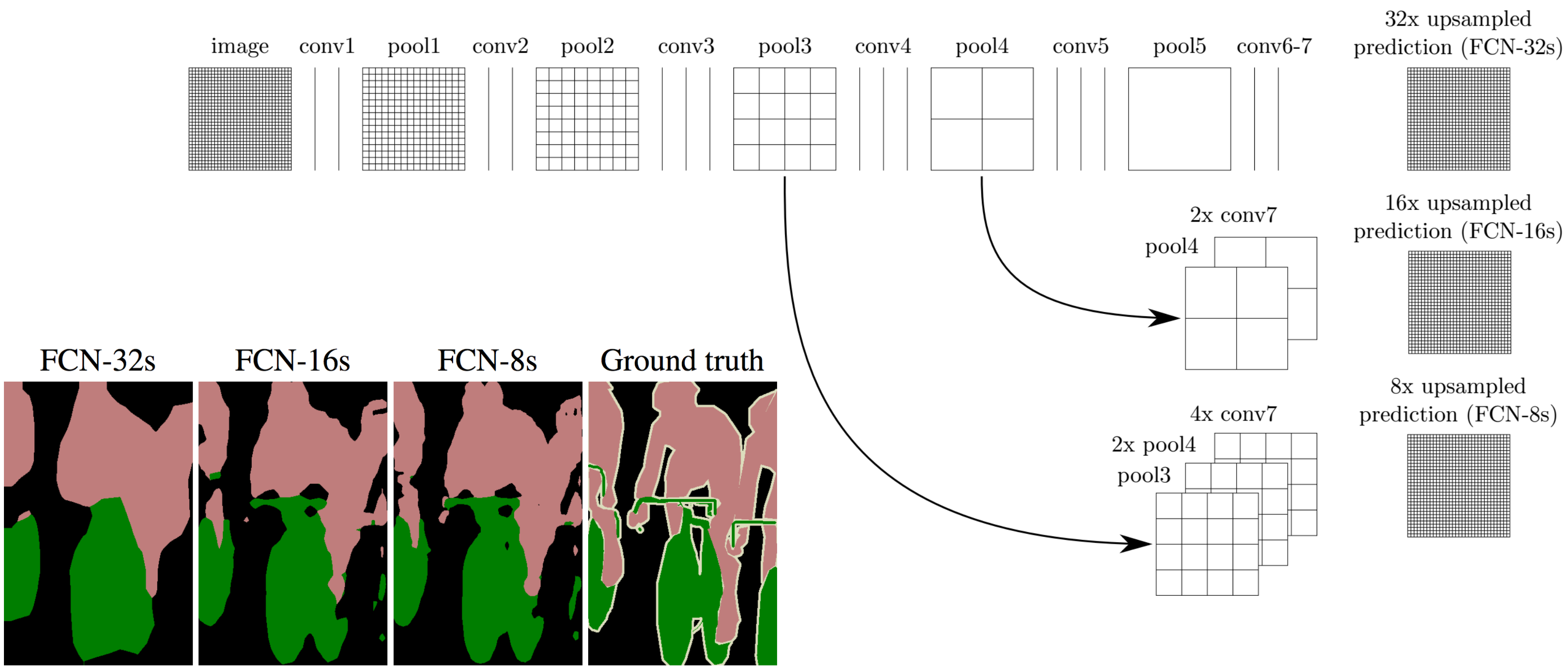
Deconv Layers for generating image !



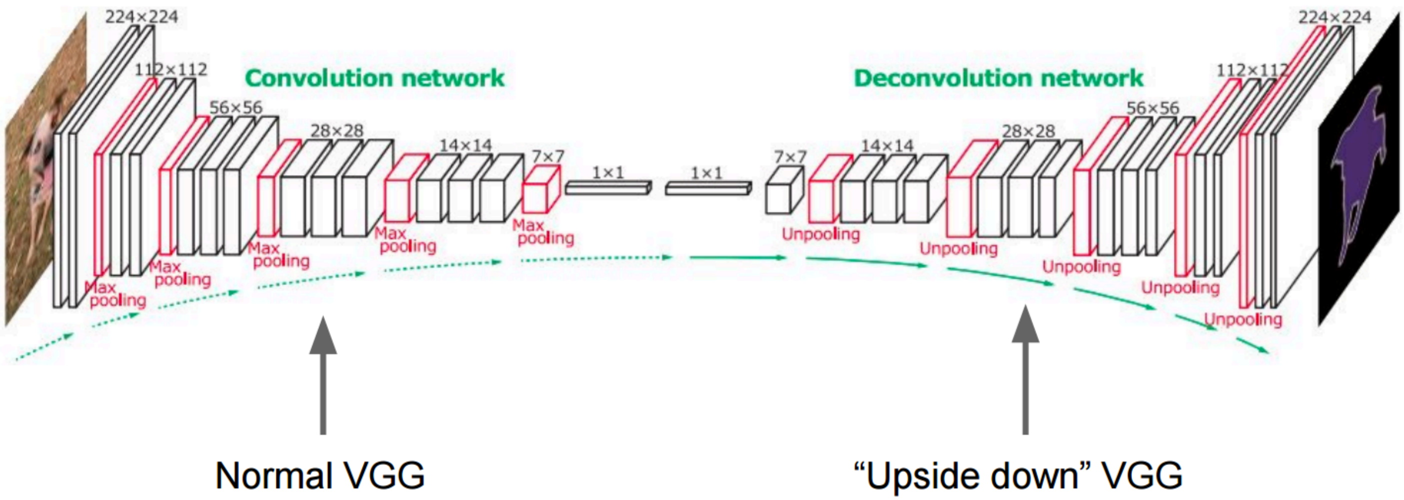
Framework of FCN



Framework of FCN

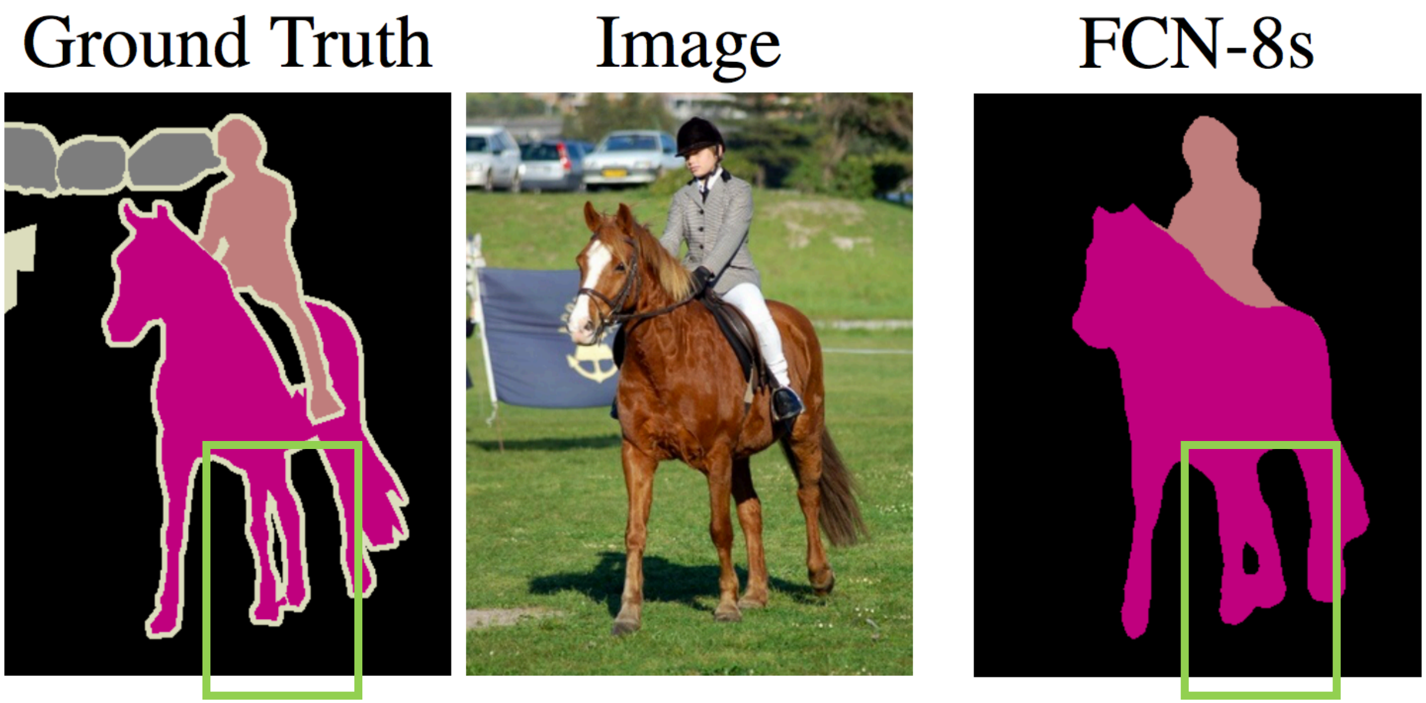


More than one upsampling layer



Noh et al, “Learning Deconvolution Network for Semantic Segmentation”, ICCV 2015

FCN is still not good ?



Very coarse feature maps --- FCN-8s is still very coarse

Dilated Convolution

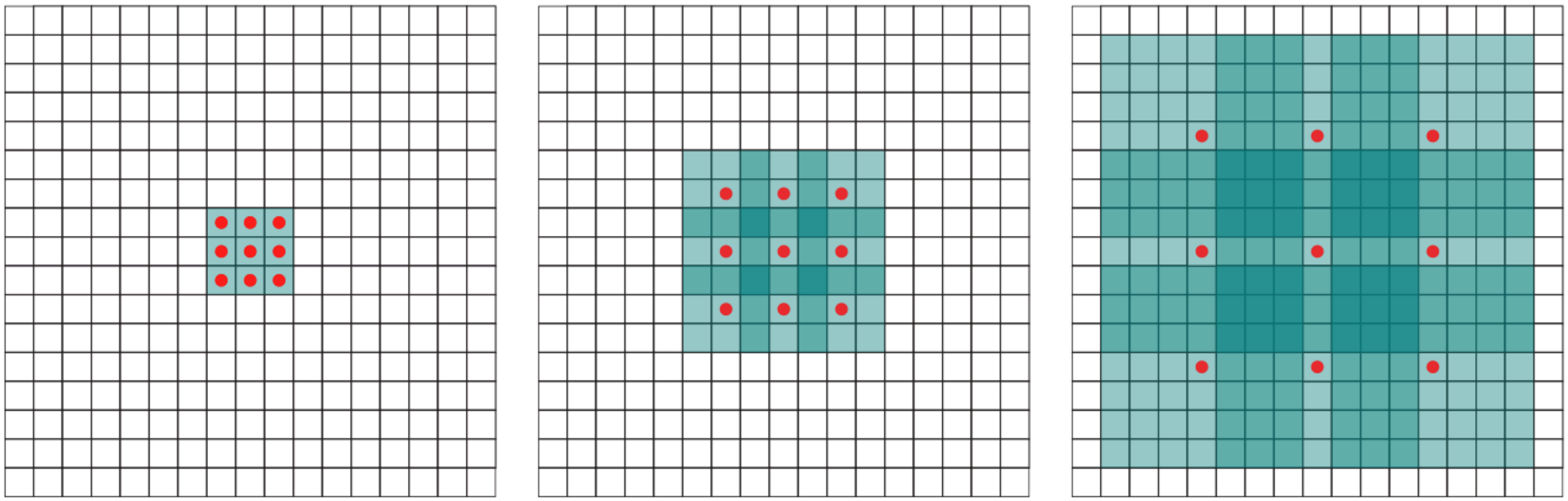
Discrete Convolution Operation:

$$(F * k)(\mathbf{p}) = \sum_{\mathbf{s}+\mathbf{t}=\mathbf{p}} F(\mathbf{s})\, k(\mathbf{t})$$

Dilated Convolution Operation:

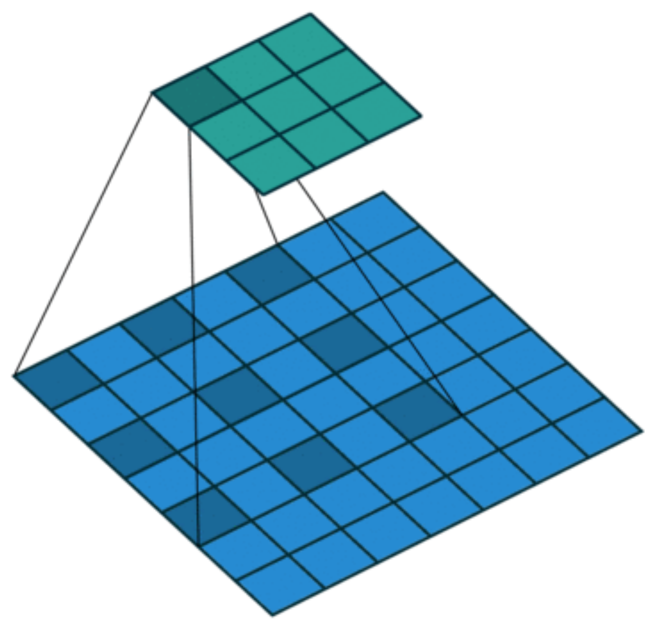
$$(F *_l k)(\mathbf{p}) = \sum_{\mathbf{s}+l\mathbf{t}=\mathbf{p}} F(\mathbf{s})\, k(\mathbf{t}).$$

Dilated Convolution



(a) (b) (c)

Dilated Convolution



Network with Dilated Convolution

- Following FCN structure
- Using VGG-16 networks with modifications
- Called Front End and greatly improve performance

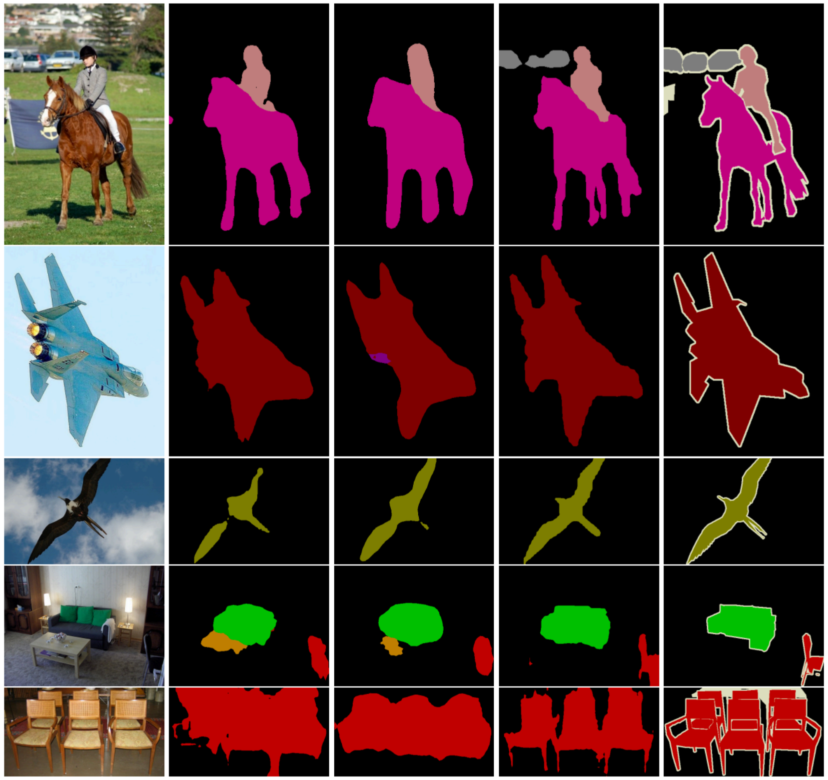
Network with Dilated Convolution

```python
print 'VGG-16'
network = [
    {'k': 3, 'p': 1}, 'conv1_1',
    {'k': 3, 'p': 1}, 'conv1_2',
    {'k': 2, 's': 2}, 'pool1',
    {'k': 3, 'p': 1}, 'conv2_1',
    {'k': 3, 'p': 1}, 'conv2_2',
    {'k': 2, 's': 2}, 'pool2',
    {'k': 3, 'p': 1}, 'conv3_1',
    {'k': 3, 'p': 1}, 'conv3_2',
    {'k': 3, 'p': 1}, 'conv3_3',
    {'k': 2, 's': 2}, 'pool3',
    {'k': 3, 'p': 1}, 'conv4_1',
    {'k': 3, 'p': 1}, 'conv4_2',
    {'k': 3, 'p': 1}, 'conv4_3',
    {'k': 2, 's': 2}, 'pool4',
    {'k': 3, 'p': 1}, 'conv5_1',
    {'k': 3, 'p': 1}, 'conv5_2',
    {'k': 3, 'p': 1}, 'conv5_3',
    {'k': 2, 's': 2}, 'pool5',
    {'k': 7}, 'fc6',
    {'k': 1}, 'fc7',
    {'k': 1}, 'fc8',
]
```

```python
print 'VGG-16 with Dilated Convs for Dense Prediction\n'
print 'Pascal VOC front end'
network = [
    {'k': 3}, 'conv1_1',
    {'k': 3}, 'conv1_2',
    {'k': 2, 's': 2}, 'pool1',
    {'k': 3}, 'conv2_1',
    {'k': 3}, 'conv2_2',
    {'k': 2, 's': 2}, 'pool2',
    {'k': 3}, 'conv3_1',
    {'k': 3}, 'conv3_2',
    {'k': 3}, 'conv3_3',
    {'k': 2, 's': 2}, 'pool3',
    {'k': 3}, 'conv4_1',
    {'k': 3}, 'conv4_2',
    {'k': 3}, 'conv4_3',
    {'k': 3, 'd': 2}, 'conv5_1',
    {'k': 3, 'd': 2}, 'conv5_2',
    {'k': 3, 'd': 2}, 'conv5_3',
    {'k': 7, 'd': 4}, 'fc6',
    {'k': 1}, 'fc7',
    {'k': 1}, 'fc-final',
]
```

From arunmallya

Network with Dilated Convolution



(a) Image (b) FCN-8s (c) DeepLab (d) Our front end (e) Ground truth

Network with Dilated Convolution

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean IoU
FCN-8s	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1	62.2
DeepLab	72	31	71.2	53.7	60.5	77	71.9	73.1	25.2	62.6	49.1	68.7	63.3	73.9	73.6	50.8	72.3	42.1	67.9	52.6	62.1
DeepLab-Msc	74.9	34.1	72.6	52.9	61.0	77.9	73.0	73.7	26.4	62.2	49.3	68.4	64.1	74.0	75.0	51.7	72.7	42.5	67.2	55.7	62.9
Our front end	**82.2**	**37.4**	**72.7**	**57.1**	**62.7**	**82.8**	**77.8**	**78.9**	**28**	**70**	**51.6**	**73.1**	**72.8**	**81.5**	**79.1**	**56.6**	**77.1**	**49.9**	**75.3**	**60.9**	**67.6**

Multi-scale Context Aggregation

Layer	1	2	3	4	5	6	7	8
Convolution	3×3	3×3	3×3	3×3	3×3	3×3	3×3	1×1
Dilation	1	1	2	4	8	16	1	1
Truncation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Receptive field	3×3	5×5	9×9	17×17	33×33	65×65	67×67	67×67
Output channels								
Basic	C	C	C	C	C	C	C	C
Large	$2C$	$2C$	$4C$	$8C$	$16C$	$32C$	$32C$	C

Multi-scale Context Aggregation

Context Aggregation Module

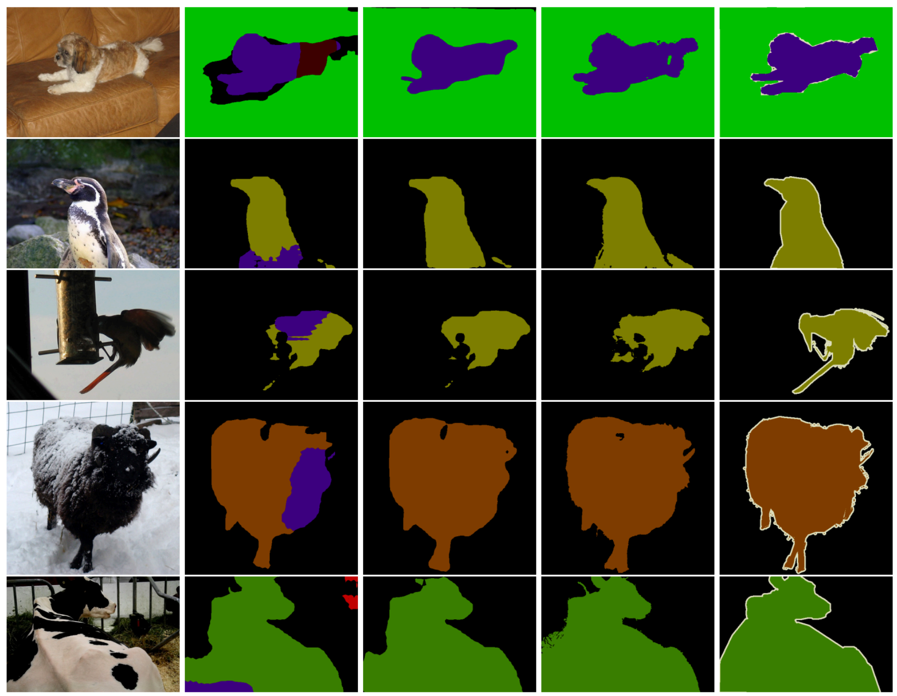
Layer Name	Receptive Field	Effective Stride	Output Size
Input	--	--	66
ct_conv1_1	3	1	130
ct_conv1_2	5	1	128
ct_conv2_1	9	1	124
ct_conv3_1	17	1	116
ct_conv4_1	33	1	100
ct_conv5_1	65	1	68
ct_fc1	67	1	66
ct_final	67	1	66

From arunmallya

Front End + Context Net

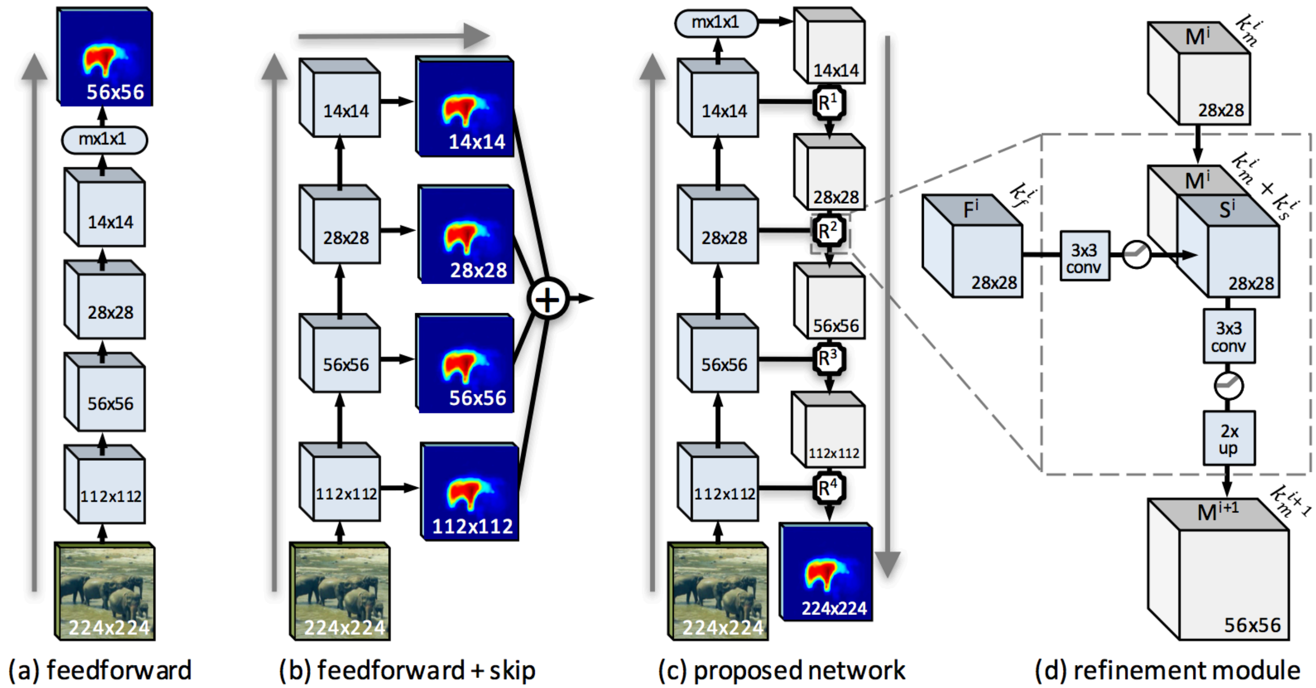
	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean IoU
Front end	86.3	38.2	76.8	**66.8**	63.2	87.3	78.7	82	33.7	76.7	53.5	73.7	76	76.6	83	**51.9**	77.8	44	79.9	**66.3**	69.8
Front + Basic	86.4	37.6	78.5	66.3	64.1	89.9	79.9	84.9	**36.1**	79.4	**55.8**	77.6	81.6	79	83.1	51.2	81.3	43.7	82.3	65.7	71.3
Front + Large	**87.3**	**39.2**	**80.3**	65.6	**66.4**	**90.2**	**82.6**	**85.8**	34.8	**81.9**	51.7	**79**	**84.1**	**80.9**	**83.2**	51.2	**83.2**	**44.7**	**83.4**	65.6	**72.1**
Front end + CRF	89.2	38.8	80	**69.8**	63.2	88.8	80	85.2	33.8	80.6	55.5	77.1	80.8	77.3	84.3	**53.1**	80.4	45	80.7	**67.9**	71.6
Front + Basic + CRF	89.1	38.7	81.4	67.4	65	91	81	86.7	**37.5**	81	**57**	79.6	83.6	79.9	**84.6**	52.7	83.3	44.3	82.6	67.2	72.7
Front + Large + CRF	**89.6**	**39.9**	**82.7**	66.7	**67.5**	**91.1**	**83.3**	**87.4**	36	**83.3**	52.5	**80.7**	**85.7**	**81.8**	84.4	52.6	**84.4**	**45.3**	**83.7**	66.7	**73.3**
Front end + RNN	88.8	38.1	80.8	**69.1**	65.6	89.9	79.6	85.7	36.3	83.6	57.3	77.9	83.2	77	**84.6**	**54.7**	82.1	**46.9**	80.9	66.7	72.5
Front + Basic + RNN	89	38.4	82.3	67.9	65.2	91.5	80.4	87.2	**38.4**	82.1	**57.7**	79.9	85	79.6	84.5	53.5	84	45	82.8	66.2	73.1
Front + Large + RNN	**89.3**	**39.2**	**83.6**	67.2	**69**	**92.1**	**83.1**	**88**	**38.4**	**84.8**	55.3	**81.2**	**86.7**	**81.3**	84.3	53.6	**84.4**	45.8	**83.8**	**67**	**73.9**

Front End + Context Net



(a) Image (b) Front end (c) +Context (d) +CRF-RNN (e) Ground truth

Top to Down Refinement



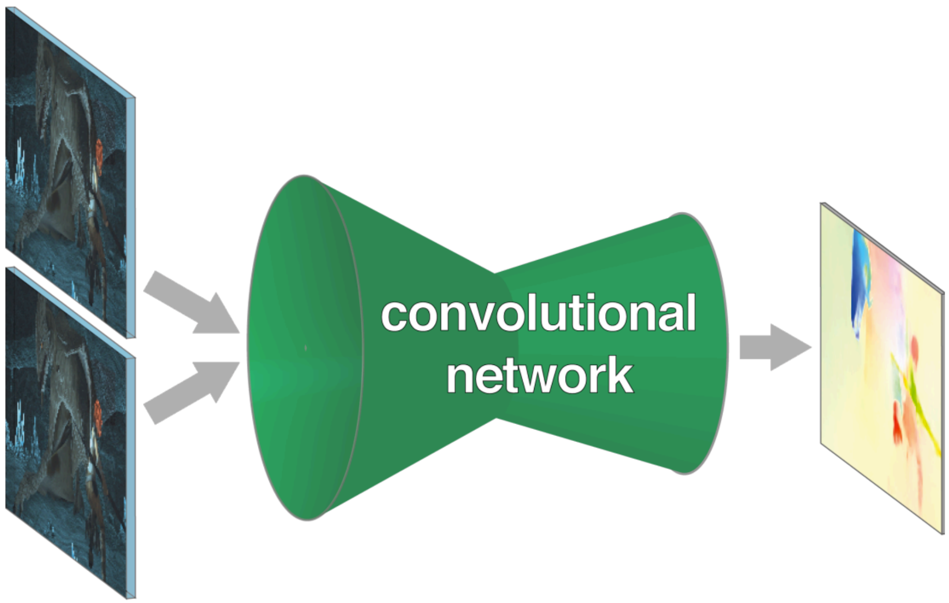
Learning to refine object segments, ECCV 2016

Refine Deep Mask

(a) DeepMask Output (b) SharpMask Output

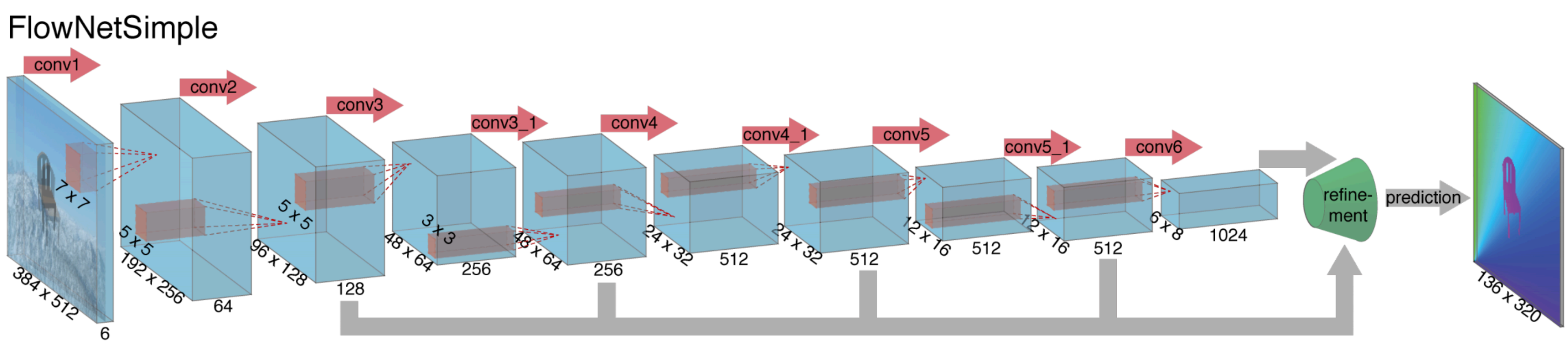
Learning to refine object segments, ECCV 2016

Dense Labeling Task: Learning Optical Flow



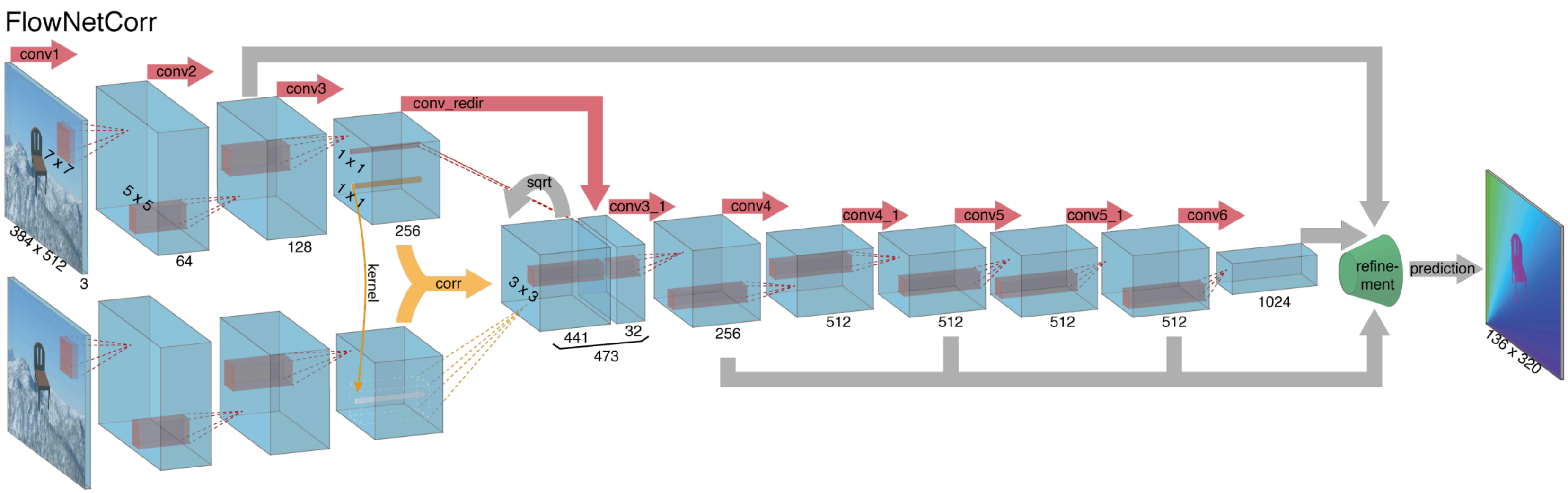
FlowNet: Learning Optical Flow with Convolutional Networks

FlowNetSimple



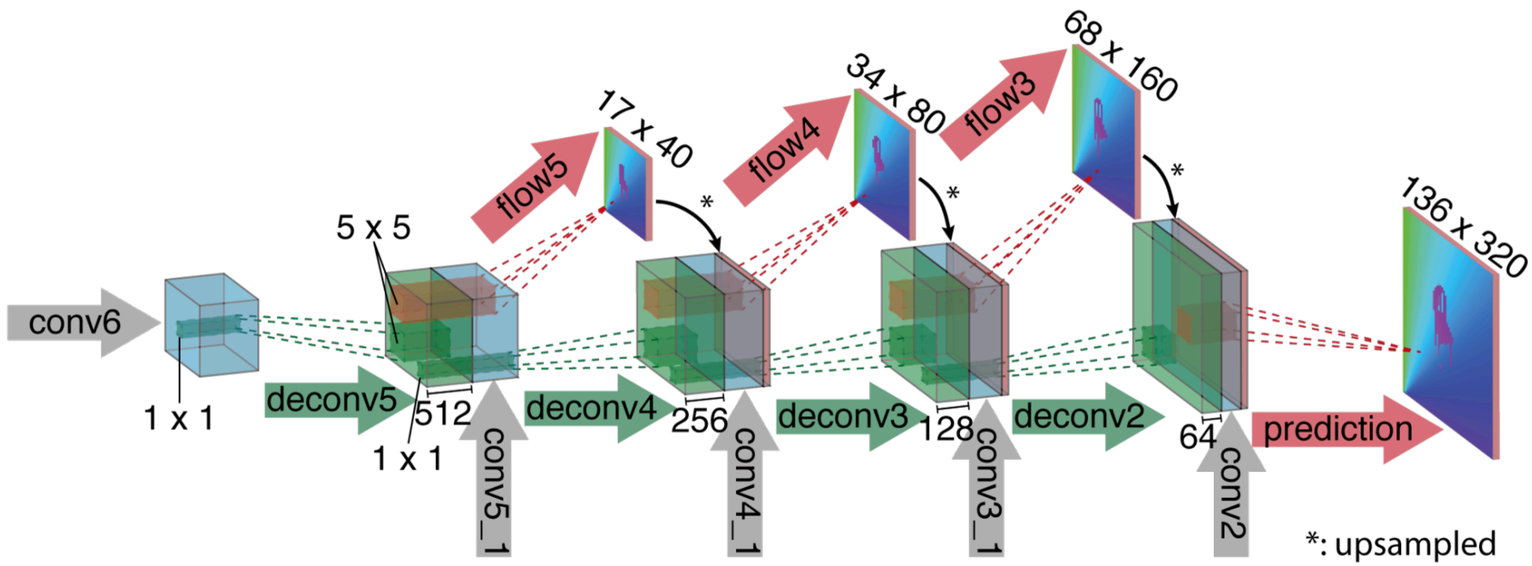
FlowNet: Learning Optical Flow with Convolutional Networks

FlowNetCorr



FlowNet: Learning Optical Flow with Convolutional Networks

Refinement of the coarse feature maps

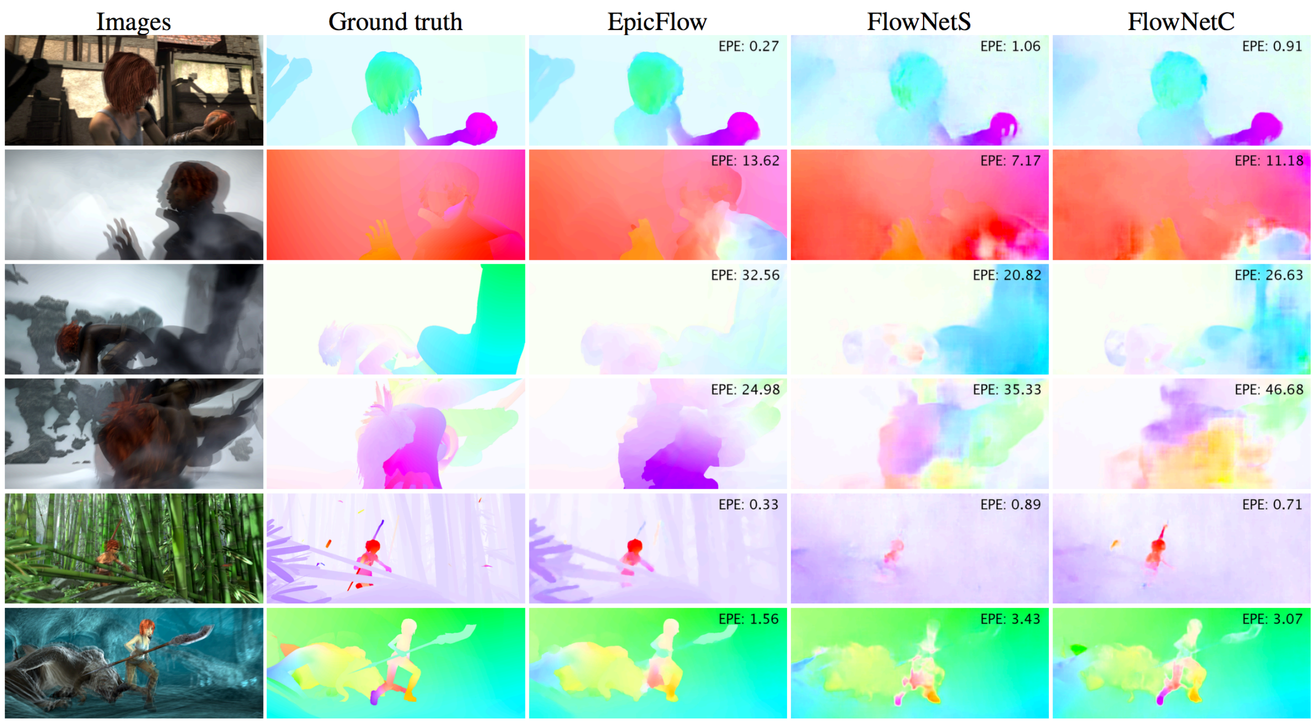


FlowNet: Learning Optical Flow with Convolutional Networks

Examples of Data Pairs

FlowNet: Learning Optical Flow with Convolutional Networks

Results



FlowNet: Learning Optical Flow with Convolutional Networks