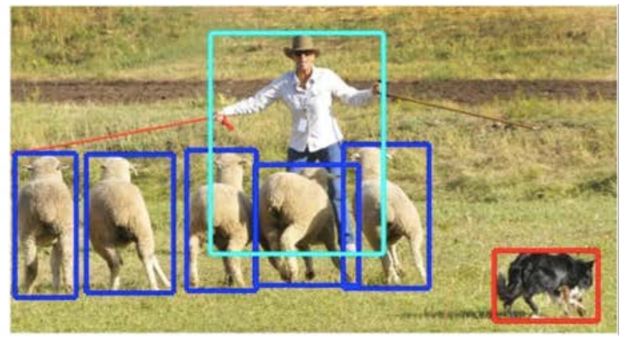
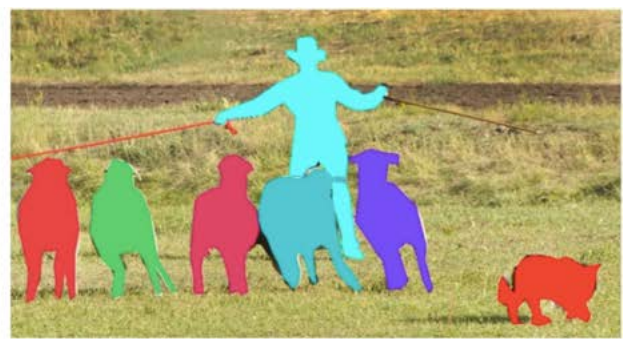
CNNs for dense image labeling

image classification



object detection

semantic segmentation



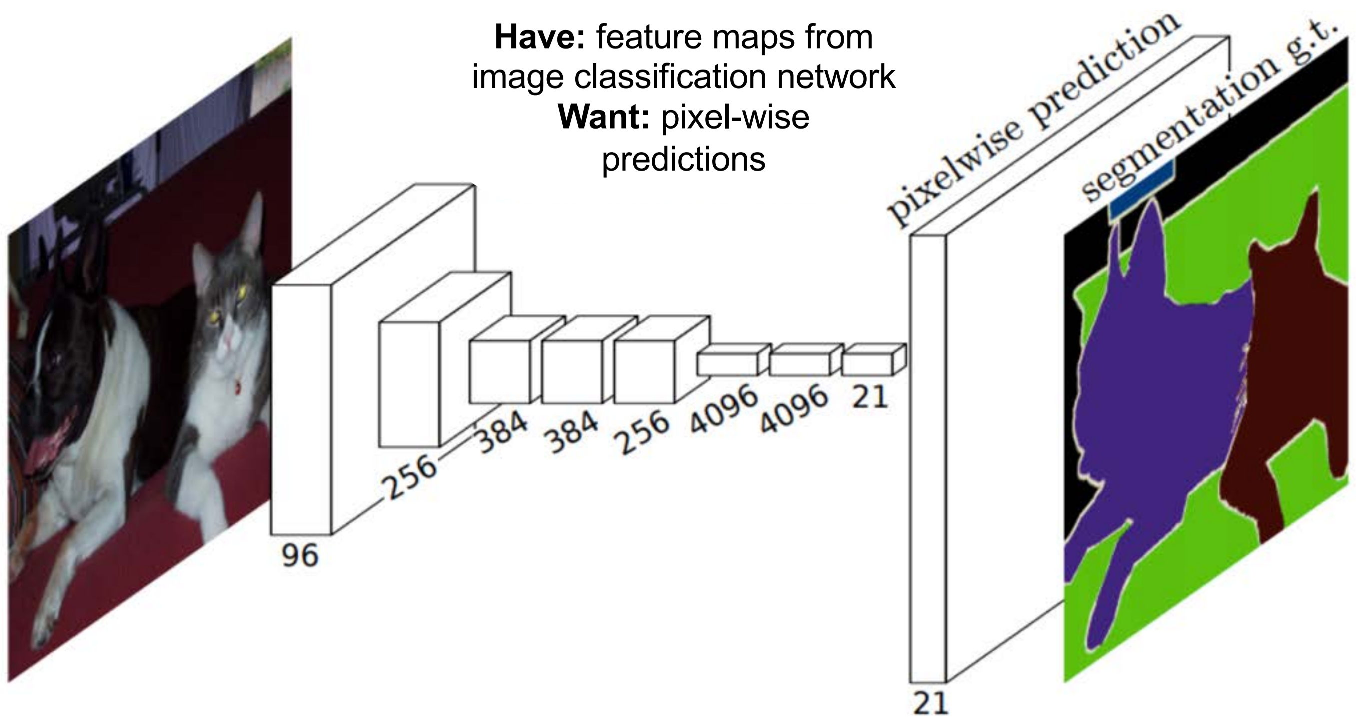
instance segmentation

Outline

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

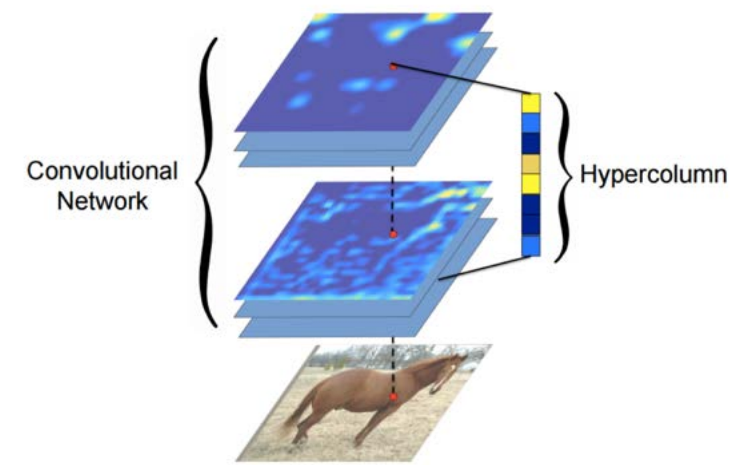
Early "hacks"

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

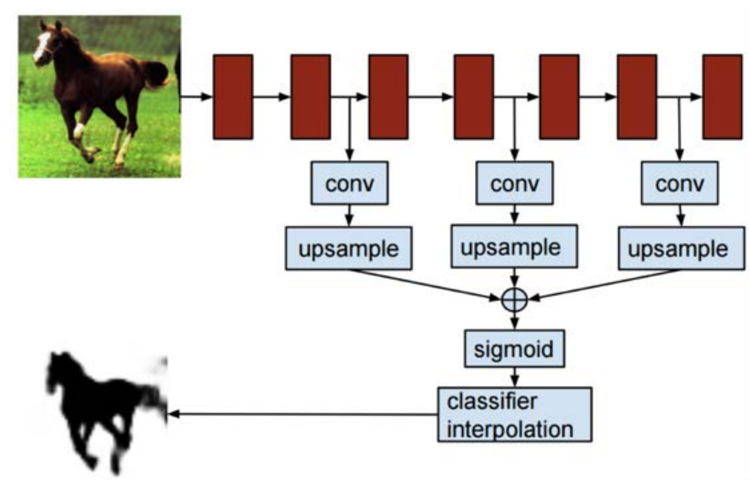


Hypercolumns

- Idea: to obtain a feature representation for an individual pixel, upsample all feature maps to original image resolution and concatenate values from feature maps "above" that pixel

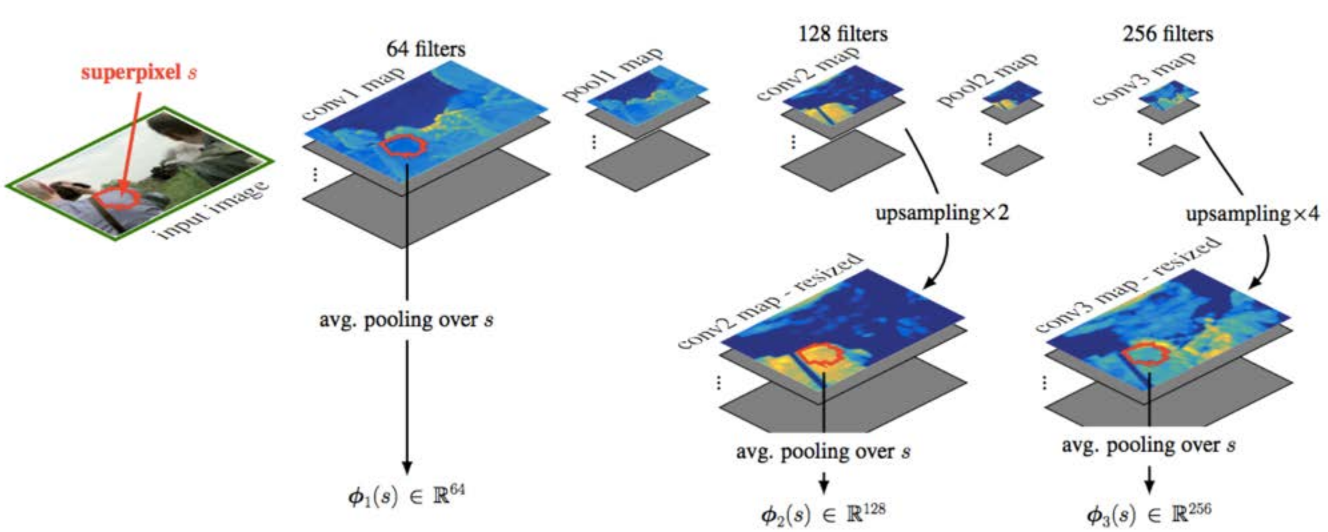


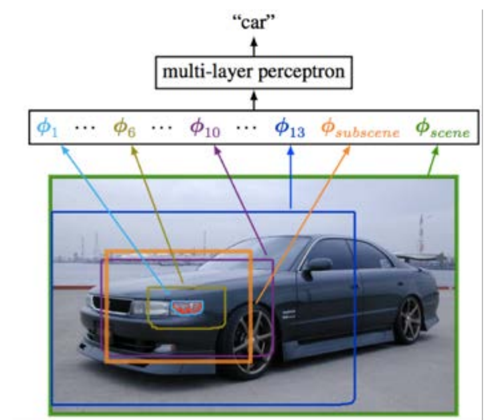
Representation as an end-to-end network:



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

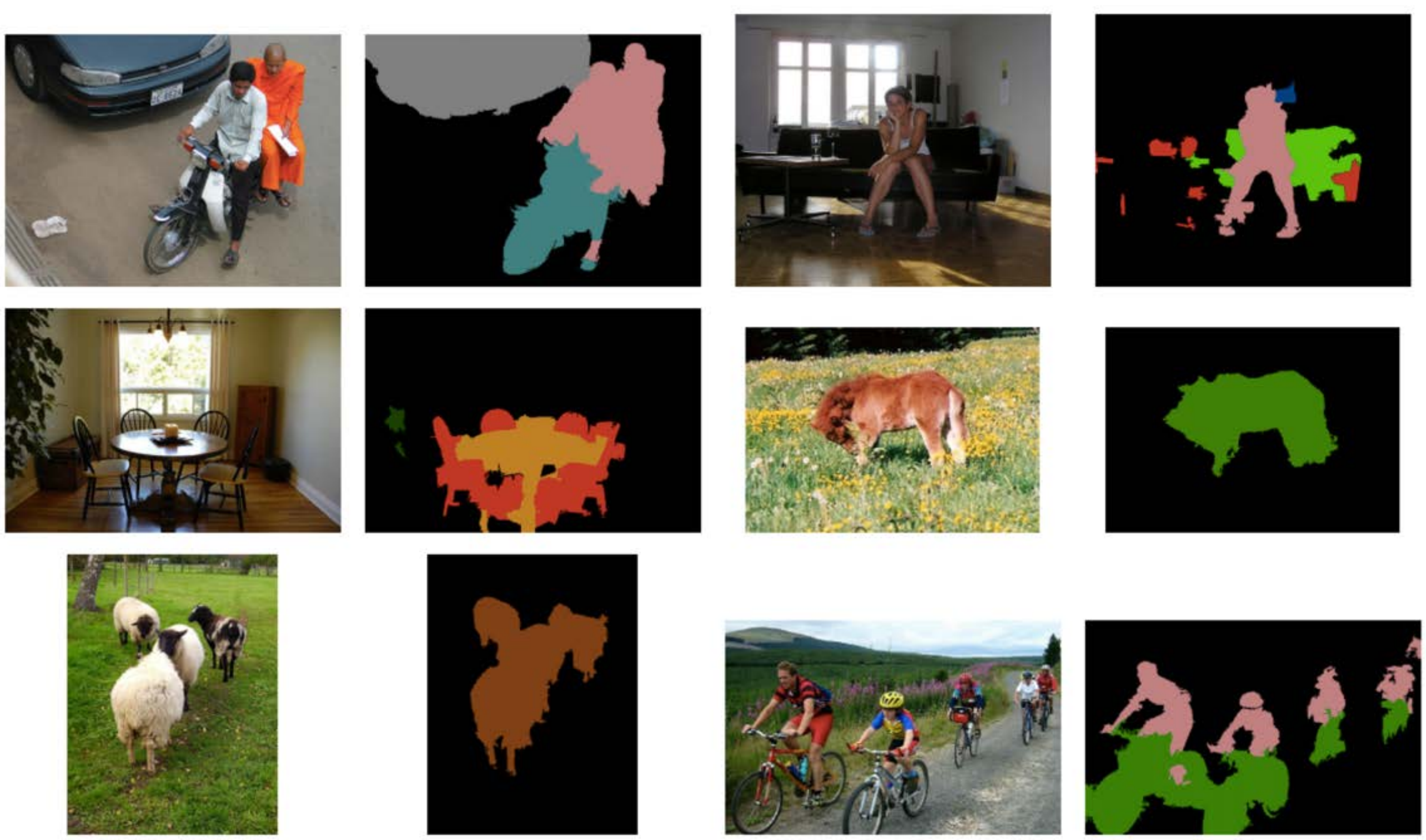
Zoom-out features





M. Mostajabi, P. Yadollahpour and G. Shakhnarovich,
[Feedforward semantic segmentation with zoom-out features](),
CVPR 2015

Zoom-out features: Example results



Zoom-out features: Evaluation

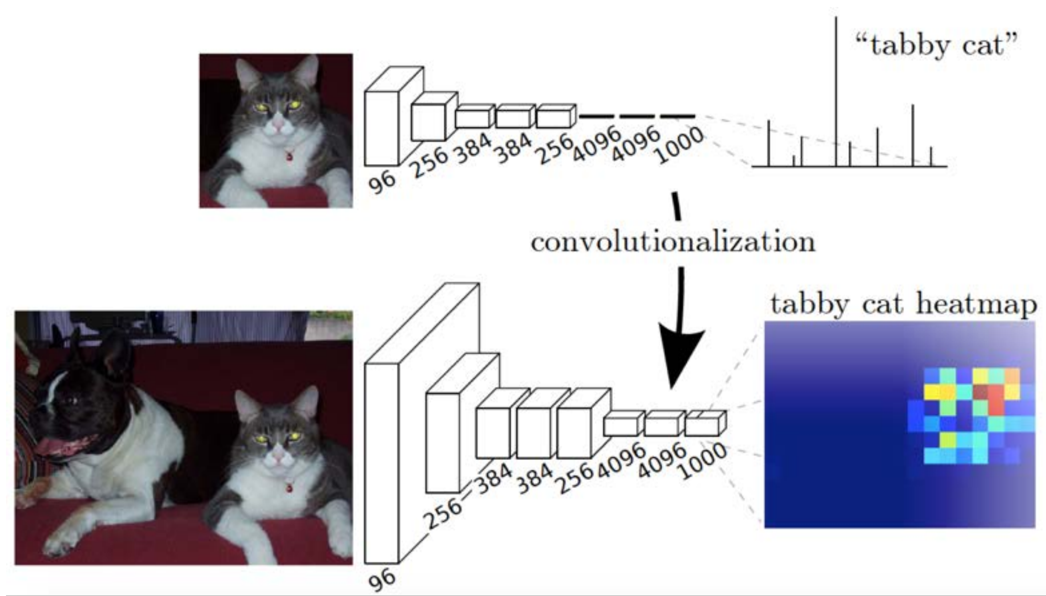
- **Metric: mean IoU**
  - Intersection over union of predicted and ground truth pixels for each class, averaged over classes

Method	VOC2010	VOC2011	VOC2012
zoom-out (ours)	**69.9**	**69.4**	**69.6**
Hypercolumns [13]	–	–	62.6
FCN-8s [26]	–	62.7	62.2
DivMbest+convnet [8]	–	–	52.2
SDS [15]	–	52.6	51.6
DivMbest+rerank [39]	–	–	48.1
Codemaps [24]	–	–	48.3
O2P [4]	–	47.6	47.8
Regions & parts[2]	–	40.8	–
D-sampling [27]	33.5	–	–
Harmony potentials [3]	40.1	–	–

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

Fully convolutional networks

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



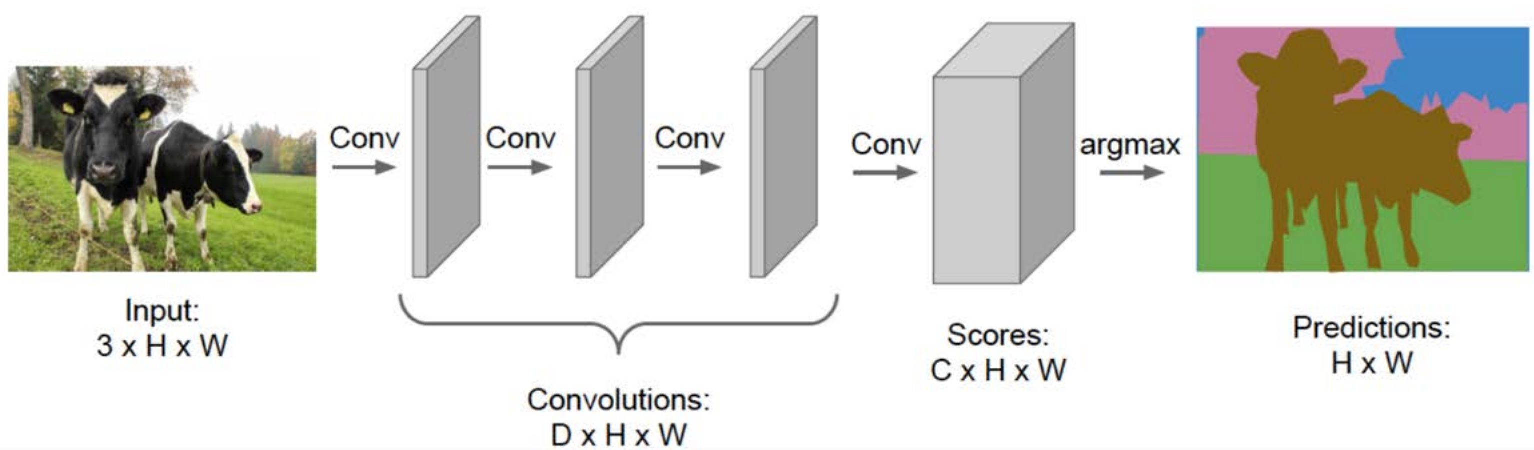
J. Long, E. Shelhamer, and T. Darrell, [Fully Convolutional Networks for Semantic Segmentation](), CVPR 2015

Dense prediction: Outline

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

Fully convolutional networks

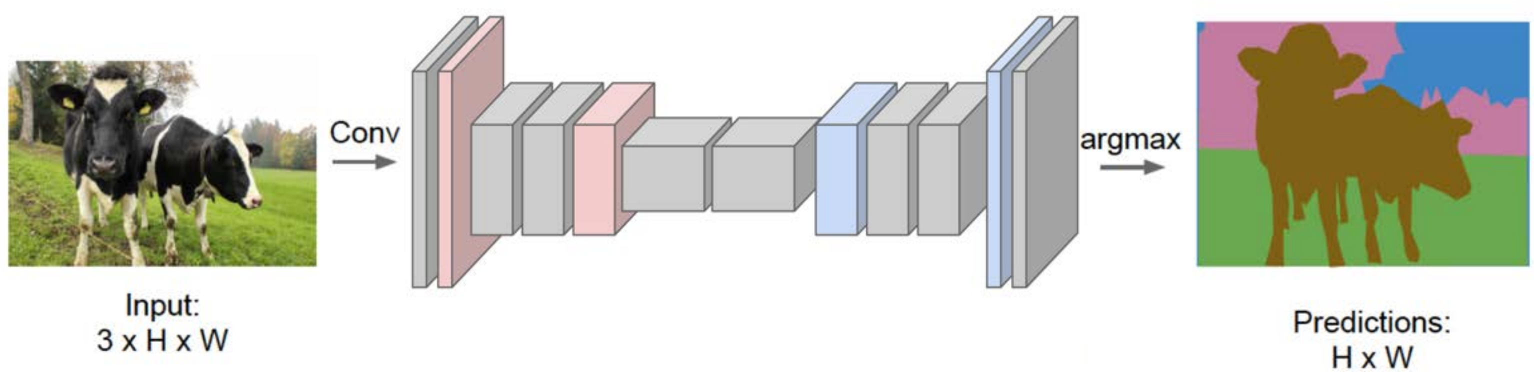
- Design a network with only convolutional layers, make predictions for all pixels at once
- Can the network operate at full image resolution?



Source: [Stanford CS231n]()

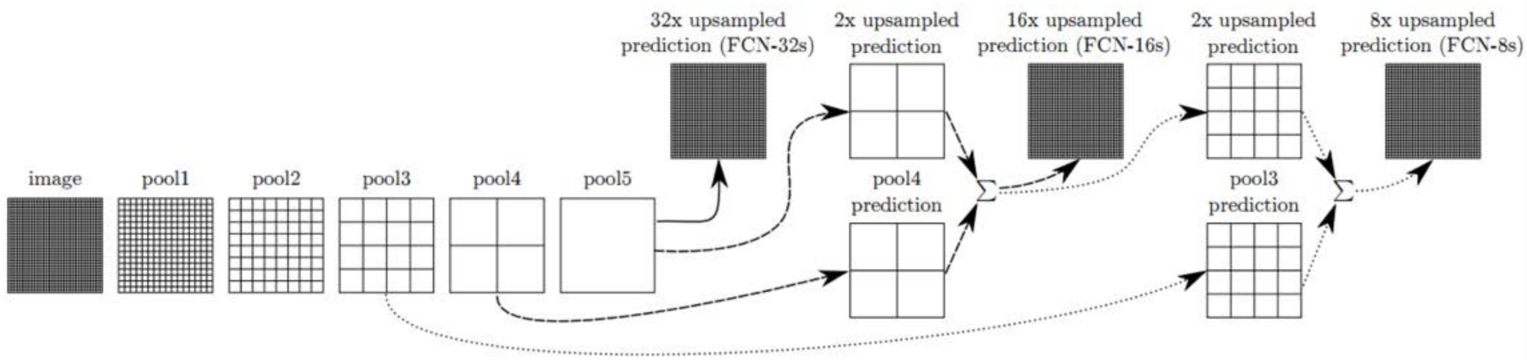
Fully convolutional networks

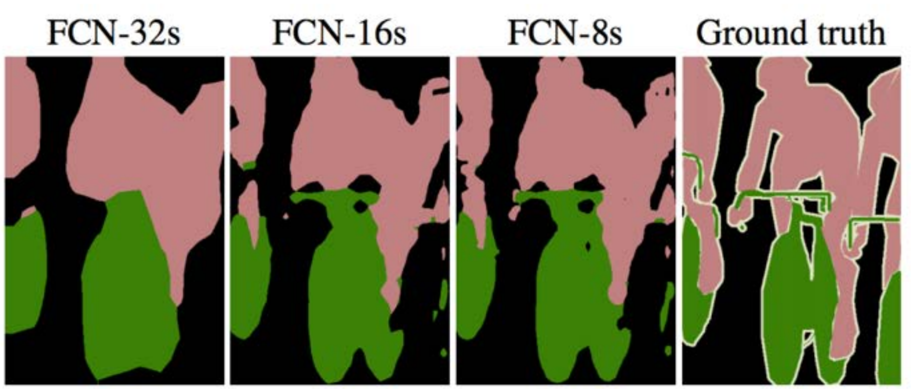
- Design a network with only convolutional layers, make predictions for all pixels at once
- Can the network operate at full image resolution?
- Practical solution: first downsample, then upsample



Source: Stanford CS231n

Fully convolutional networks (FCN)





Comparison on a subset of PASCAL 2011 validation data:

	pixel acc.	mean acc.	mean IU
FCN-32s-fixed	83.0	59.7	45.4
FCN-32s	89.1	73.3	59.4
FCN-16s	90.0	75.7	62.4
FCN-8s	**90.3**	**75.9**	**62.7**

J. Long, E. Shelhamer, and T. Darrell, Fully Convolutional Networks for Semantic Segmentation, CVPR 2015

Outline

- Early "hacks"
  - Hypercolumns
  - Zoom-out features
  - Fully convolutional networks
- Deep network operations for dense prediction
  - Transposed convolutions
  - Unpooling
  - Dilated convolutions

Upsampling in a deep network

- Regular convolution (stride 1, pad 0)

$$\begin{array}{|c|c|c|c|}\hline x_{11} & x_{12} & x_{13} & x_{14} \\ \hline x_{21} & x_{22} & x_{23} & x_{24} \\ \hline x_{31} & x_{32} & x_{33} & x_{34} \\ \hline x_{41} & x_{42} & x_{43} & x_{44} \\ \hline \end{array} * \begin{array}{|c|c|c|}\hline w_{11} & w_{12} & w_{13} \\ \hline w_{21} & w_{22} & w_{23} \\ \hline w_{31} & w_{32} & w_{33} \\ \hline \end{array} = \begin{array}{|c|c|}\hline z_{11} & z_{12} \\ \hline z_{21} & z_{22} \\ \hline \end{array}$$

- Matrix-vector form:

$$\begin{pmatrix} w_{11} & w_{12} & w_{13} & 0 & w_{21} & w_{22} & w_{23} & 0 & w_{31} & w_{32} & w_{33} & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{11} & w_{12} & w_{13} & 0 & w_{21} & w_{22} & w_{23} & 0 & w_{31} & w_{32} & w_{33} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_{11} & w_{12} & w_{13} & 0 & w_{21} & w_{22} & w_{23} & 0 & w_{31} & w_{32} & w_{33} & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{11} & w_{12} & w_{13} & 0 & w_{21} & w_{22} & w_{23} & 0 & w_{31} & w_{32} & w_{33} \end{pmatrix} \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ \vdots \\ x_{44} \end{pmatrix} = \begin{pmatrix} z_{11} \\ z_{12} \\ z_{21} \\ z_{22} \end{pmatrix}$$

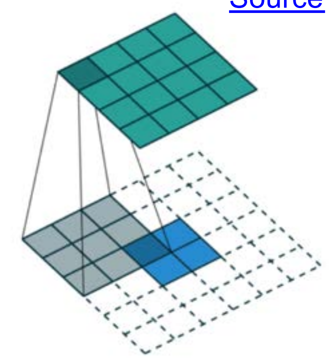
4x4 input, 2x2 output

Upsampling in a deep network

- Transposed convolution

$$\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} *^T \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}$$

[Source]



$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{bmatrix} z_{11} \\ z_{12} \\ z_{21} \\ z_{22} \end{bmatrix} = \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{bmatrix}$$

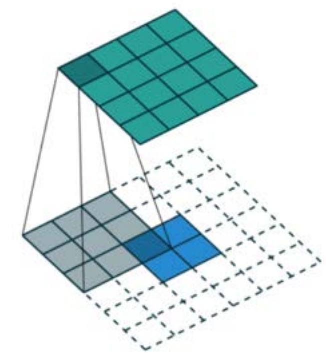
2x2 input, 4x4 output

Not an inverse of the original convolution operation, simply reverses dimension change!

Upsampling in a deep network

- Transposed convolution

$$\begin{array}{|c|c|}\hline z_{11} & z_{12} \\ \hline z_{21} & z_{22} \\ \hline\end{array} \;*^T\; \begin{array}{|c|c|c|}\hline w_{11} & w_{12} & w_{13} \\ \hline w_{21} & w_{22} & w_{23} \\ \hline w_{31} & w_{32} & w_{33} \\ \hline\end{array} \;=\; \begin{array}{|c|c|c|c|}\hline x_{11} & x_{12} & x_{13} & x_{14} \\ \hline x_{21} & x_{22} & x_{23} & x_{24} \\ \hline x_{31} & x_{32} & x_{33} & x_{34} \\ \hline x_{41} & x_{42} & x_{43} & x_{44} \\ \hline\end{array}$$



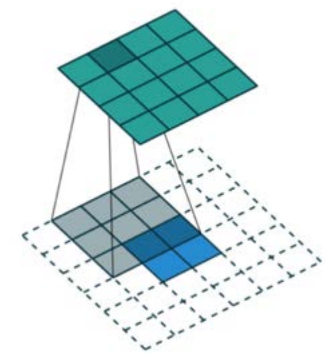
$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{pmatrix} z_{11} \\ z_{12} \\ z_{21} \\ z_{22} \end{pmatrix} = \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}$$

$$x_{11} = w_{11} z_{11}$$

Upsampling in a deep network

- Transposed convolution

$$\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} *^{T} \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}$$



Convolve input with *flipped* filter

$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{pmatrix} z_{11} \\ z_{12} \\ z_{21} \\ z_{22} \end{pmatrix} = \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}$$

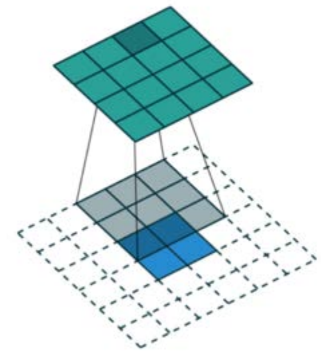
$$x_{12} = w_{12} z_{11} + w_{11} z_{12}$$

Upsampling in a deep network

- Transposed convolution

$$\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} *^{T} \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}$$

Convolve input with *flipped* filter



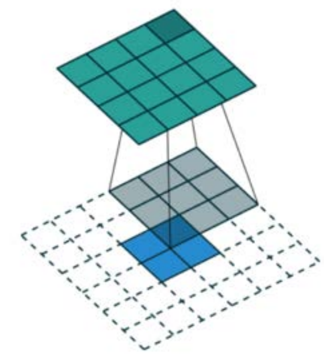
$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{pmatrix} z_{11} \\ z_{12} \\ z_{21} \\ z_{22} \end{pmatrix} = \begin{pmatrix} x_{11} \\ x_{12} \\ \boxed{x_{13}} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}$$

$$x_{13} = w_{13} z_{11} + w_{12} z_{12}$$

Upsampling in a deep network

- Transposed convolution

$$\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} *^T \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}$$



Convolve input with *flipped* filter

$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{pmatrix} z_{11} \\ z_{12} \\ z_{21} \\ z_{22} \end{pmatrix} = \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}$$

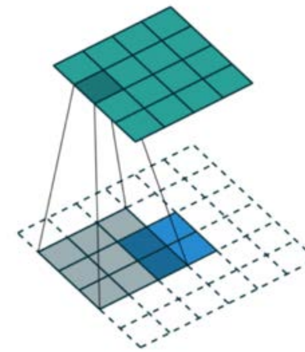
$$x_{14} = w_{13} z_{12}$$

Upsampling in a deep network

- Transposed convolution

$$\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} *^T \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}$$

Convolve input with *flipped* filter



$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ \boxed{w_{21} \quad 0 \quad w_{11} \quad 0} & & & \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{pmatrix} z_{11} \\ z_{12} \\ z_{21} \\ z_{22} \end{pmatrix} = \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ \boxed{x_{21}} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}$$

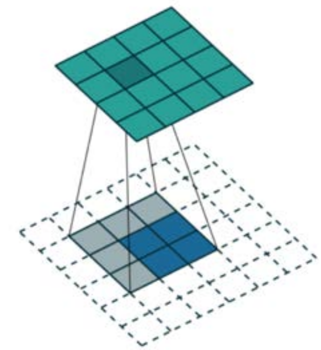
$$x_{21} = w_{21} z_{11} + w_{11} z_{21}$$

Upsampling in a deep network

- Transposed convolution

$$\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} *^T \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}$$

Convolve input with *flipped* filter



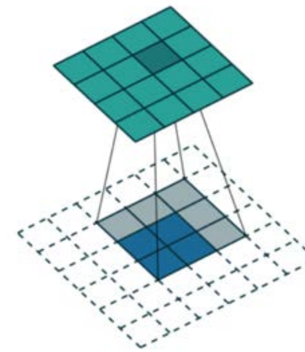
$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{pmatrix} z_{11} \\ z_{12} \\ z_{21} \\ z_{22} \end{pmatrix} = \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}$$

$$x_{22} = w_{22}z_{11} + w_{21}z_{12} + w_{12}z_{21} + w_{11}z_{22}$$

Upsampling in a deep network

- Transposed convolution

$$\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} *^{T} \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}$$



Convolve input with *flipped* filter

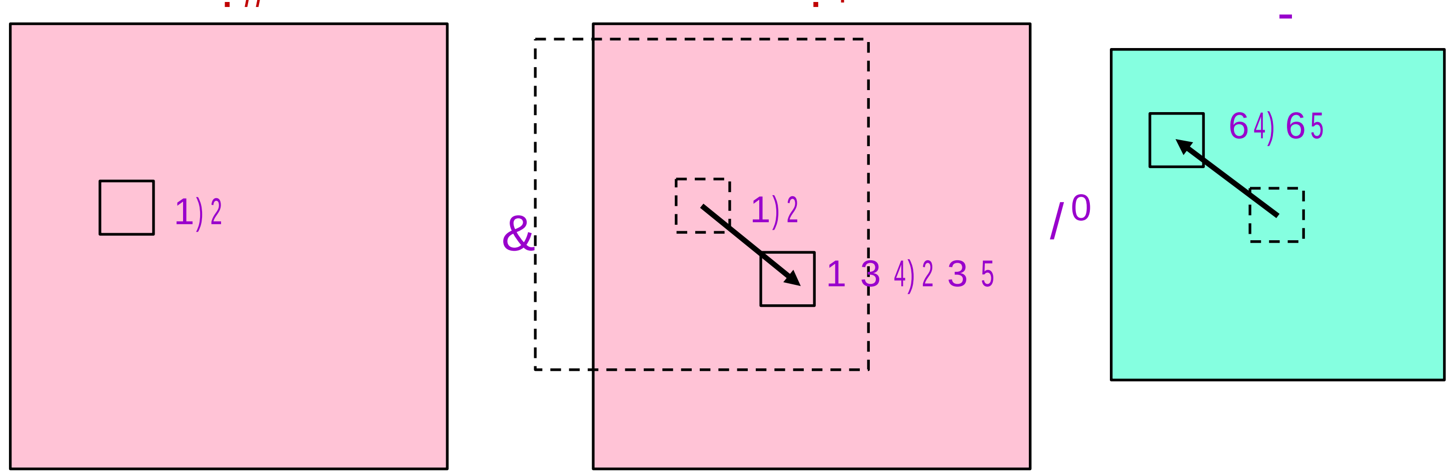
$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{bmatrix} z_{11} \\ z_{12} \\ z_{21} \\ z_{22} \end{bmatrix} = \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ \boxed{x_{23}} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{bmatrix}$$

$$x_{23} = w_{23}z_{11} + w_{22}z_{12} + w_{13}z_{21} + w_{12}z_{22}$$

! "#$%&' ($#) *$+ , '-$. . '/0+'#012'%$3"+

$$\frac{!\,"}{!\,\#_{\$\%}} \;\&\; '_{()*} \frac{!\,"}{!\,+_{\$\,,\,()\%\,,\,*}} \frac{!\,+_{\$\,,\,()\%\,,\,*}}{!\,\#_{\$\%}} \;\&\; '_{()*} \frac{!\,"}{!\,+_{\$\,,\,()\%\,,\,*}} \;-_{.\,()\,.\,*}$$

$$\frac{!\,"}{!\,\#} \qquad \frac{!\,"}{!\,+} \qquad -$$



4 5+$1.-0." ,'#0120%67801'8.'79"'.$: "'0-"+$7801'$.'
;$#) *$+, .'-$. .'/0+'+"<6%$+'#0120%67801

Upsampling in a deep network

- Transposed convolution

$$\begin{array}{|c|c|}\hline z_{11} & z_{12} \\ \hline z_{21} & z_{22} \\ \hline\end{array} \;*^{T}\; \begin{array}{|c|c|c|}\hline w_{11} & w_{12} & w_{13} \\ \hline w_{21} & w_{22} & w_{23} \\ \hline w_{31} & w_{32} & w_{33} \\ \hline\end{array} \;=\; \begin{array}{|c|c|c|c|}\hline x_{11} & x_{12} & x_{13} & x_{14} \\ \hline x_{21} & x_{22} & x_{23} & x_{24} \\ \hline x_{31} & x_{32} & x_{33} & x_{34} \\ \hline x_{41} & x_{42} & x_{43} & x_{44} \\ \hline\end{array}$$

$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{bmatrix} z_{11} \\ z_{12} \\ x_{21} \\ z_{22} \end{bmatrix} = \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{bmatrix}$$

Upsampling in a deep network

- Transposed convolution

$$\begin{array}{|c|c|}\hline z_{11} & z_{12} \\ \hline z_{21} & z_{22} \\ \hline\end{array} \;*^{T}\; \begin{array}{|c|c|c|}\hline w_{11} & w_{12} & w_{13} \\ \hline w_{21} & w_{22} & w_{23} \\ \hline w_{31} & w_{32} & w_{33} \\ \hline\end{array} \;=\; \begin{array}{|c|c|c|c|}\hline x_{11} & x_{12} & x_{13} & x_{14} \\ \hline x_{21} & x_{22} & x_{23} & x_{24} \\ \hline x_{31} & x_{32} & x_{33} & x_{34} \\ \hline x_{41} & x_{42} & x_{43} & x_{44} \\ \hline\end{array}$$

$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{pmatrix} z_{11} \\ z_{12} \\ x_{21} \\ z_{22} \end{pmatrix} = \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}$$

Alternate view:

- Place copies of the filter on the output, weighted by entries of the input

Upsampling in a deep network

- Transposed convolution

$$\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} *^T \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}$$

$$\begin{pmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{11} & 0 & 0 \\ w_{13} & w_{12} & 0 & 0 \\ 0 & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{11} & 0 \\ w_{22} & w_{21} & w_{12} & w_{11} \\ w_{23} & w_{22} & w_{13} & w_{12} \\ 0 & w_{23} & 0 & w_{13} \\ w_{31} & 0 & w_{21} & 0 \\ w_{32} & w_{31} & w_{22} & w_{21} \\ w_{33} & w_{32} & w_{23} & w_{22} \\ 0 & w_{33} & 0 & w_{23} \\ 0 & 0 & w_{31} & 0 \\ 0 & 0 & w_{32} & w_{31} \\ 0 & 0 & w_{33} & w_{32} \\ 0 & 0 & 0 & w_{33} \end{pmatrix} \begin{pmatrix} z_{11} \\ \boxed{z_{12}} \\ x_{21} \\ z_{22} \end{pmatrix} = \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}$$

Alternate view:

- Place copies of the filter on the output, weighted by entries of the input
- Sum where copies of the filter overlap

Upsampling in a deep network

- Transposed convolution

$$
\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} *^{T} \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}
$$

$$
\begin{pmatrix}
w_{11} & 0 & 0 & 0 \\
w_{12} & w_{11} & 0 & 0 \\
w_{13} & w_{12} & 0 & 0 \\
0 & w_{13} & 0 & 0 \\
w_{21} & 0 & w_{11} & 0 \\
w_{22} & w_{21} & w_{12} & w_{11} \\
w_{23} & w_{22} & w_{13} & w_{12} \\
0 & w_{23} & 0 & w_{13} \\
w_{31} & 0 & w_{21} & 0 \\
w_{32} & w_{31} & w_{22} & w_{21} \\
w_{33} & w_{32} & w_{23} & w_{22} \\
0 & w_{33} & 0 & w_{23} \\
0 & 0 & w_{31} & 0 \\
0 & 0 & w_{32} & w_{31} \\
0 & 0 & w_{33} & w_{32} \\
0 & 0 & 0 & w_{33}
\end{pmatrix}
\begin{pmatrix} z_{11} \\ z_{12} \\ \boxed{x_{21}} \\ z_{22} \end{pmatrix}
=
\begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}
$$

Alternate view:

- Place copies of the filter on the output, weighted by entries of the input
- Sum where copies of the filter overlap

Upsampling in a deep network

- Transposed convolution

$$
\begin{array}{|c|c|}\hline z_{11} & z_{12} \\ \hline z_{21} & z_{22} \\ \hline \end{array}
\;*^T\;
\begin{array}{|c|c|c|}\hline w_{11} & w_{12} & w_{13} \\ \hline w_{21} & w_{22} & w_{23} \\ \hline w_{31} & w_{32} & w_{33} \\ \hline \end{array}
\;=\;
\begin{array}{|c|c|c|c|}\hline x_{11} & x_{12} & x_{13} & x_{14} \\ \hline x_{21} & x_{22} & x_{23} & x_{24} \\ \hline x_{31} & x_{32} & x_{33} & x_{34} \\ \hline x_{41} & x_{42} & x_{43} & x_{44} \\ \hline \end{array}
$$

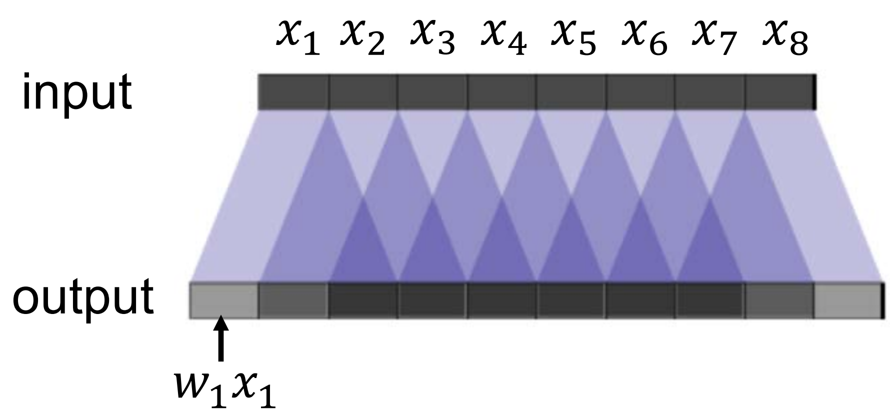
$$
\begin{pmatrix}
w_{11} & 0 & 0 & 0 \\
w_{12} & w_{11} & 0 & 0 \\
w_{13} & w_{12} & 0 & 0 \\
0 & w_{13} & 0 & 0 \\
w_{21} & 0 & w_{11} & 0 \\
w_{22} & w_{21} & w_{12} & w_{11} \\
w_{23} & w_{22} & w_{13} & w_{12} \\
0 & w_{23} & 0 & w_{13} \\
w_{31} & 0 & w_{21} & 0 \\
w_{32} & w_{31} & w_{22} & w_{21} \\
w_{33} & w_{32} & w_{23} & w_{22} \\
0 & w_{33} & 0 & w_{23} \\
0 & 0 & w_{31} & 0 \\
0 & 0 & w_{32} & w_{31} \\
0 & 0 & w_{33} & w_{32} \\
0 & 0 & 0 & w_{33}
\end{pmatrix}
\begin{pmatrix} z_{11} \\ z_{12} \\ x_{21} \\ \boxed{z_{22}} \end{pmatrix}
=
\begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{pmatrix}
$$

Alternate view:

- Place copies of the filter on the output, weighted by entries of the input
- Sum where copies of the filter overlap

Upsampling in a deep network

- 1D example

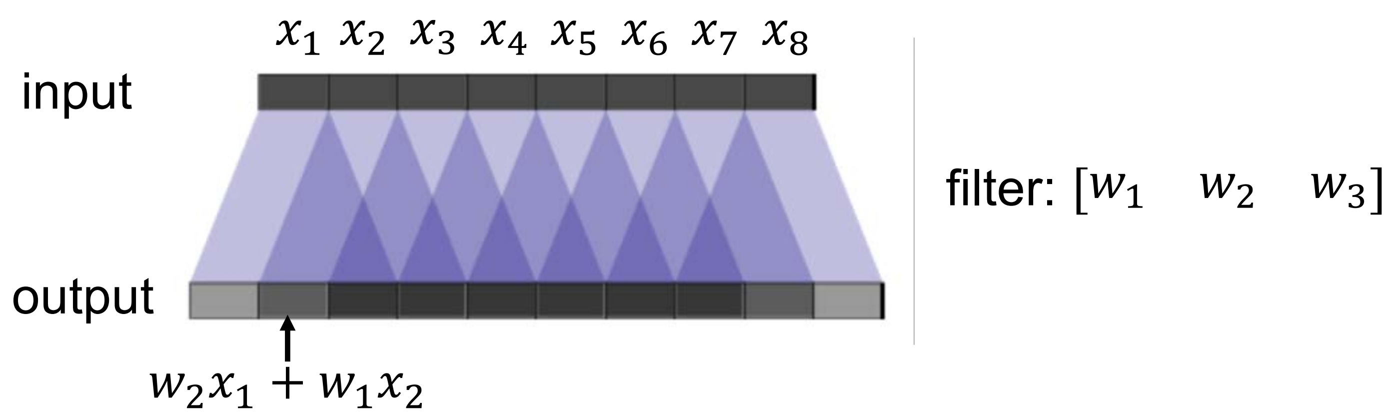


filter: $[w_1 \quad w_2 \quad w_3]$

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

Upsampling in a deep network

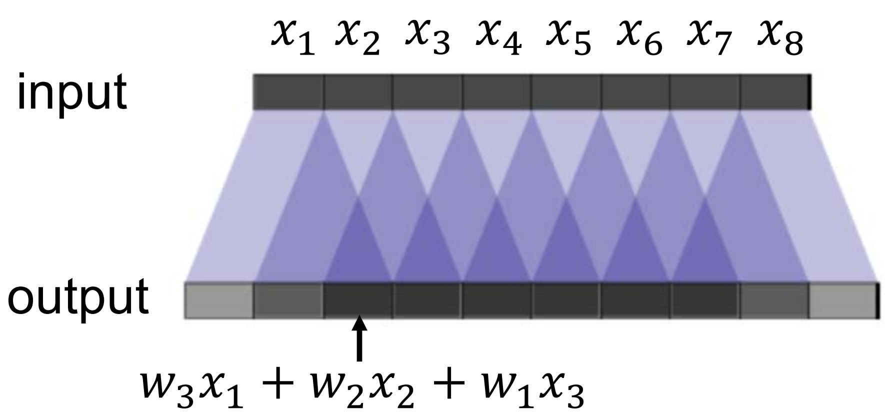
- 1D example



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

Upsampling in a deep network

- 1D example

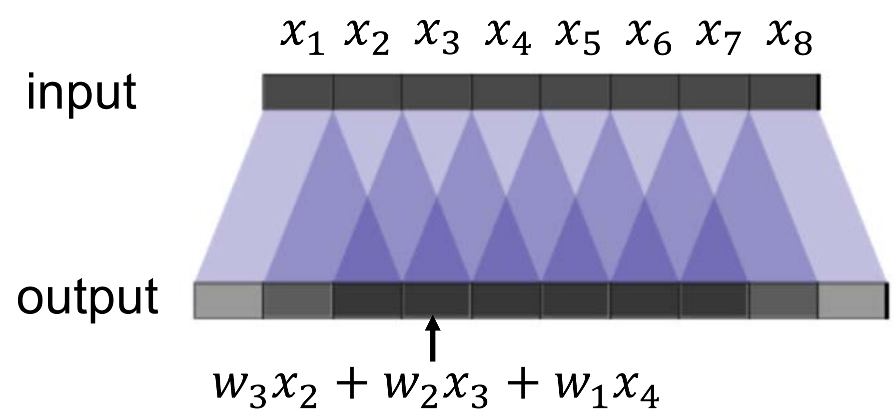


filter: $[w_1 \quad w_2 \quad w_3]$

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

Upsampling in a deep network

- 1D example

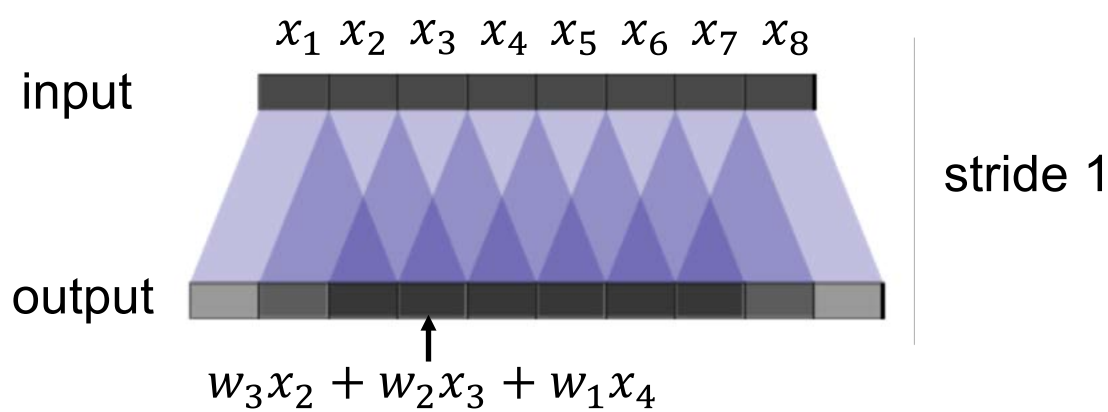


filter: $[w_1 \quad w_2 \quad w_3]$

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

Upsampling in a deep network

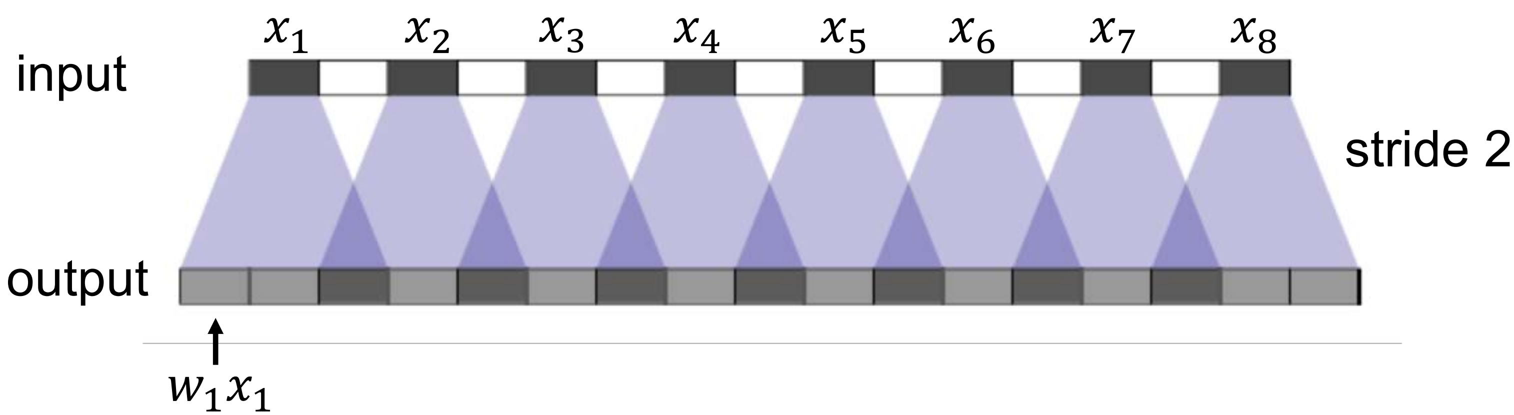
- *Backwards-strided convolution*: to increase resolution, use *output stride* > 1



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

Upsampling in a deep network

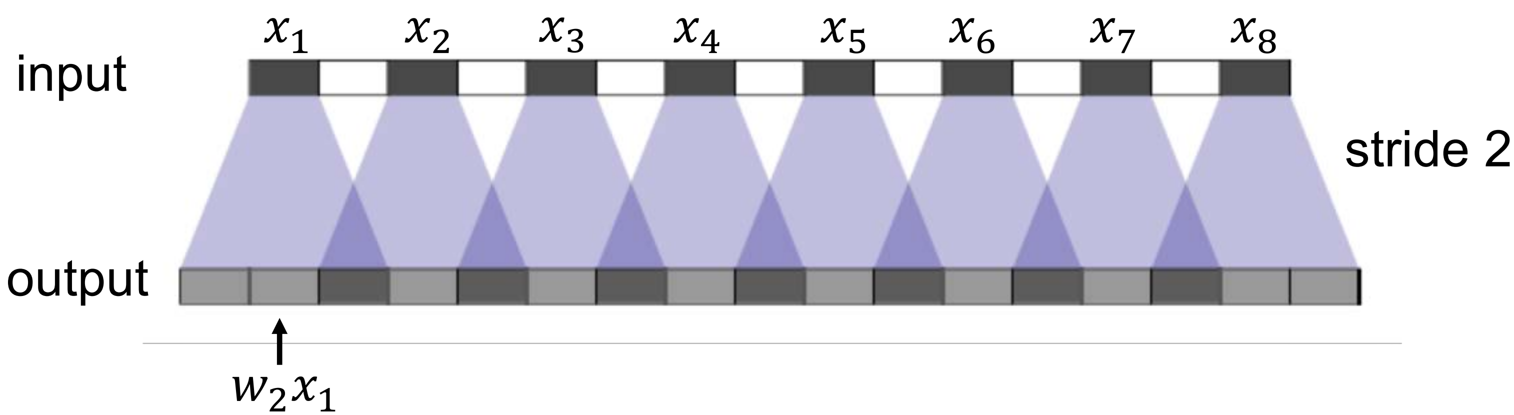
- *Backwards-strided convolution*: to increase resolution, use *output stride* > 1



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

Upsampling in a deep network

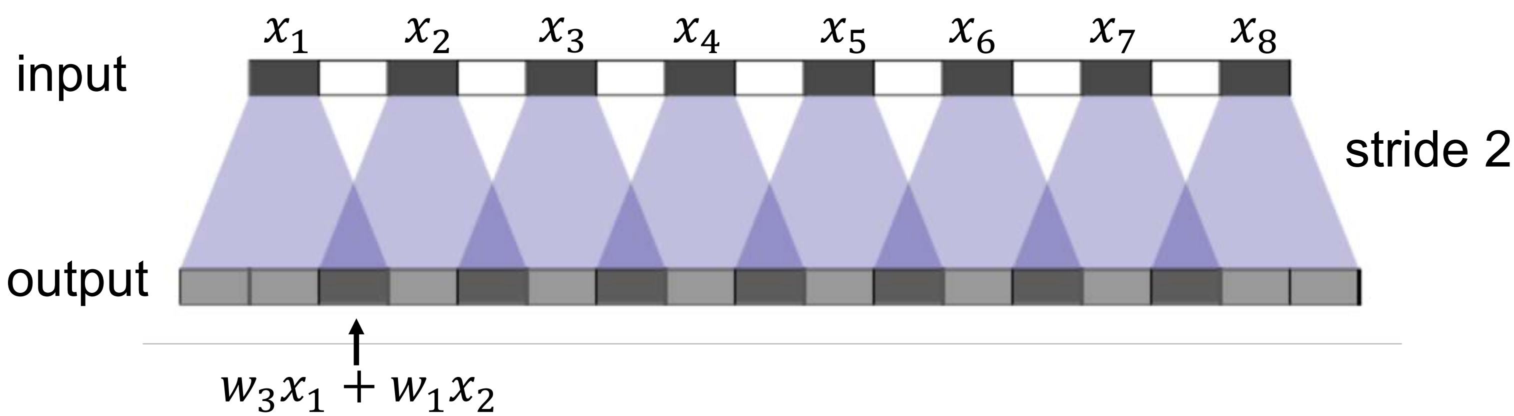
- *Backwards-strided convolution*: to increase resolution, use *output stride* > 1



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

Upsampling in a deep network

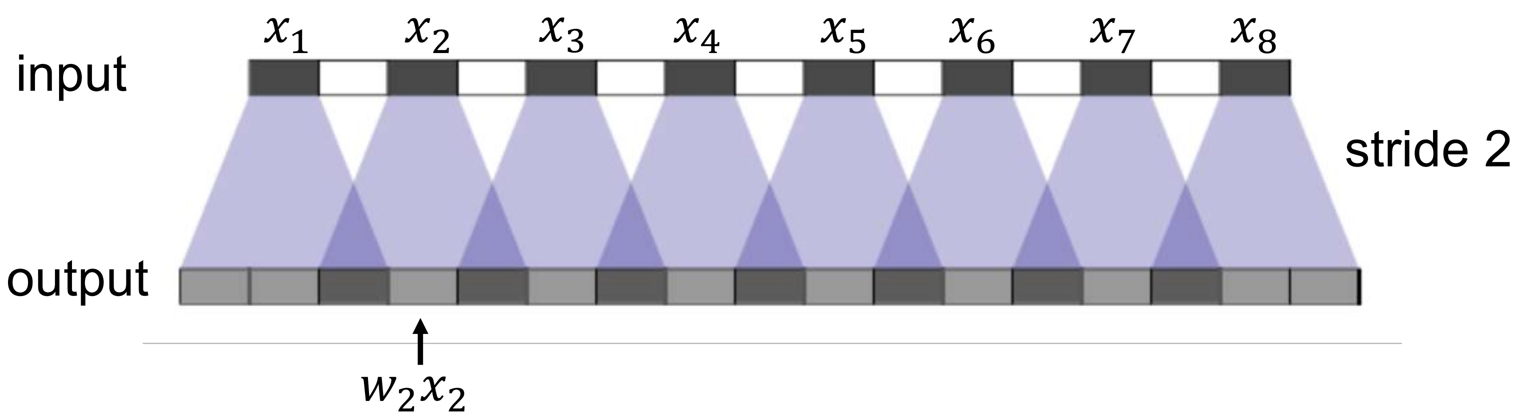
- *Backwards-strided convolution*: to increase resolution, use *output stride* > 1



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

Upsampling in a deep network

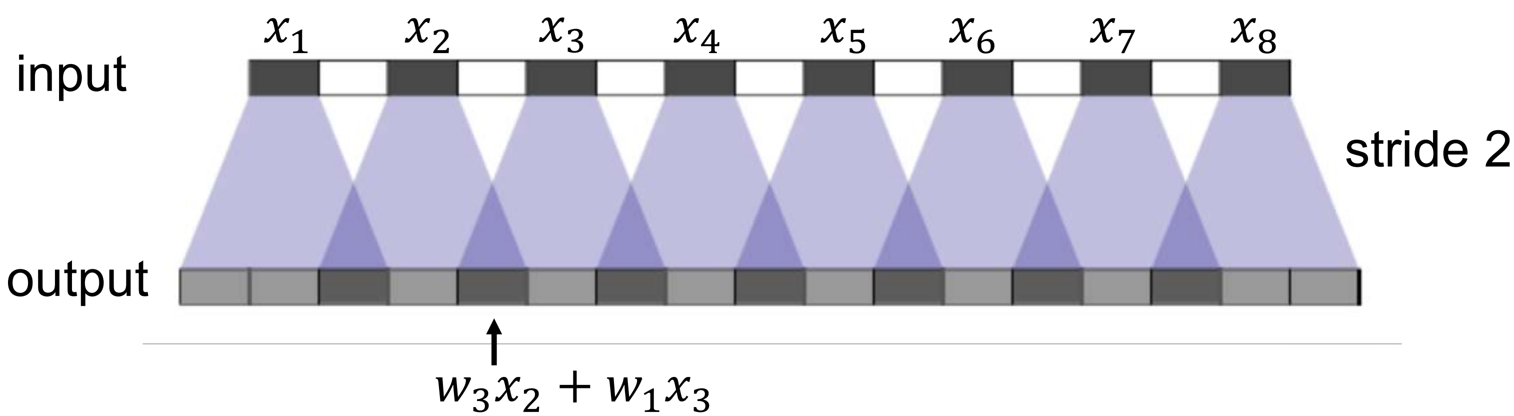
- *Backwards-strided convolution*: to increase resolution, use *output stride* > 1



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

Upsampling in a deep network

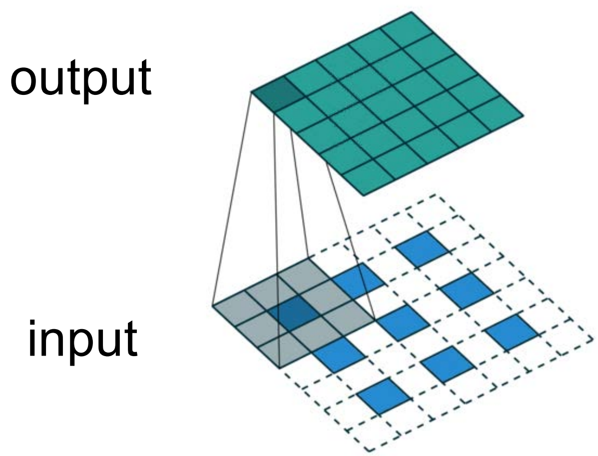
- *Backwards-strided convolution*: to increase resolution, use *output stride* > 1



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

Upsampling in a deep network

- *Backwards-strided convolution*: to increase resolution, use *output stride* > 1
  - 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



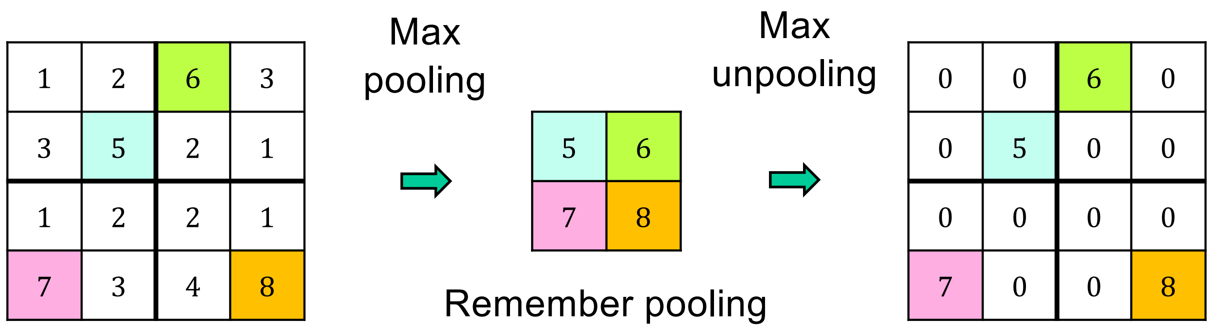
Q: What 3x3 filter would correspond to bilinear upsampling?

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

V. Dumoulin and F. Visin, A guide to convolution arithmetic for deep learning, arXiv 2018

Upsampling in a deep network

- Alternative to transposed convolution:
  max unpooling

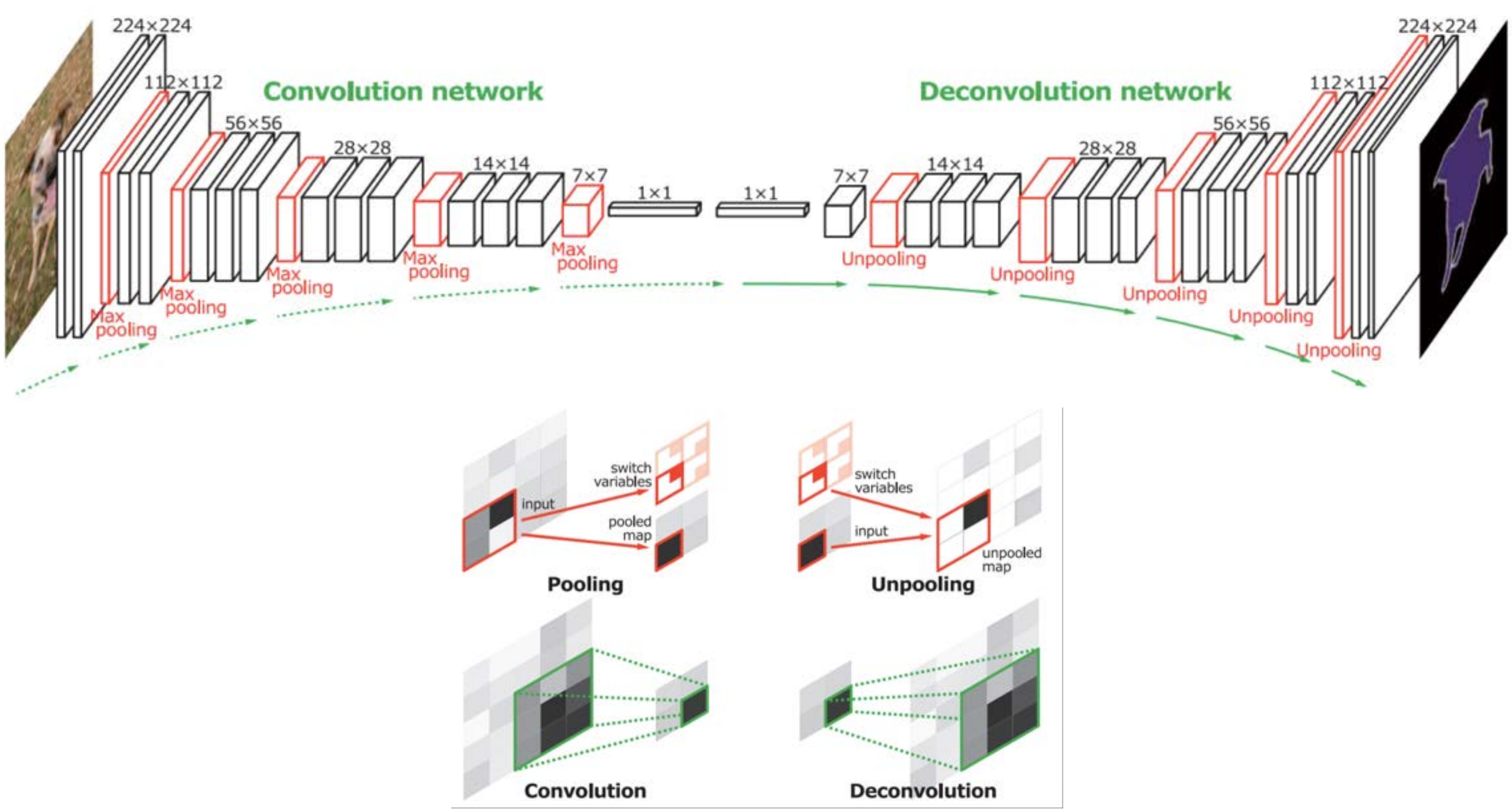


Remember pooling indices (which element was max)

Output is sparse, so need to follow this with a transposed convolution layer

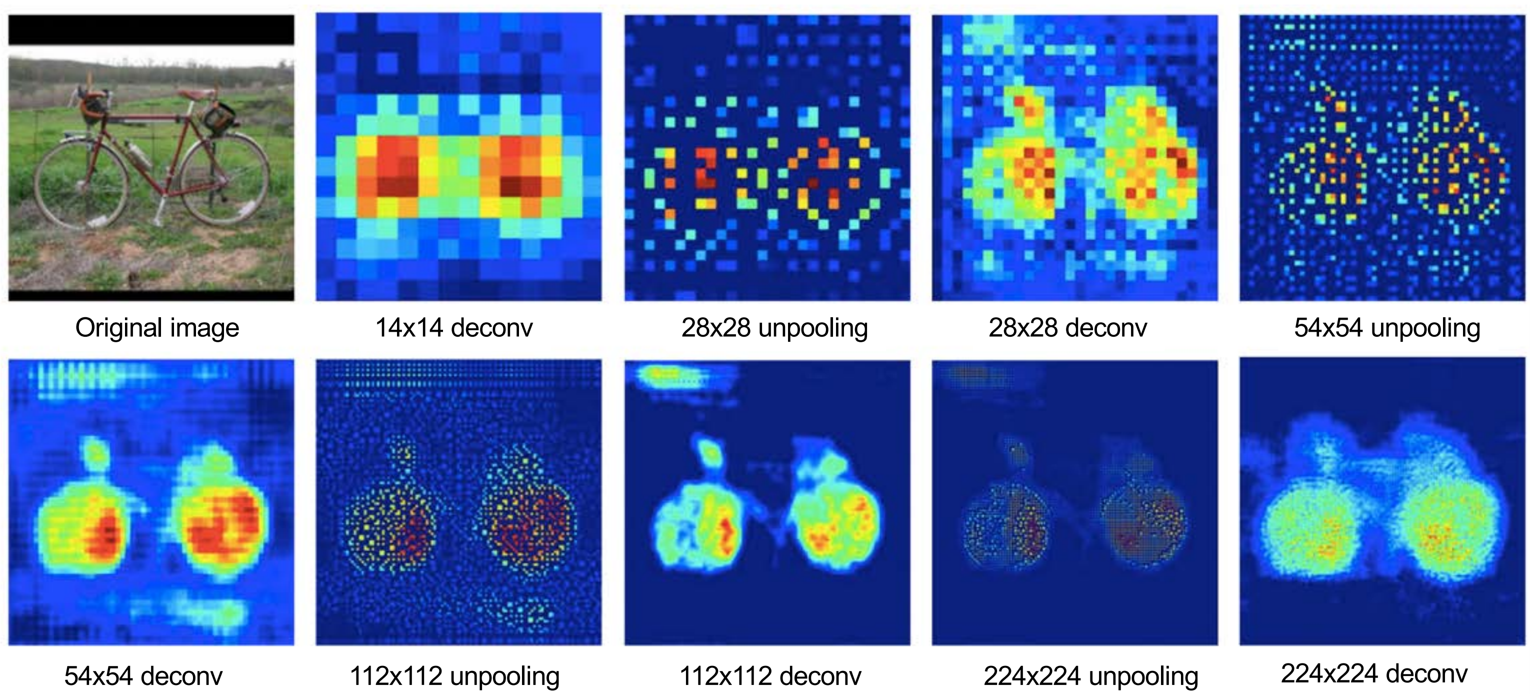
(sometimes called deconvolution instead of transposed convolution, but this is not accurate)

DeconvNet



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

DeconvNet

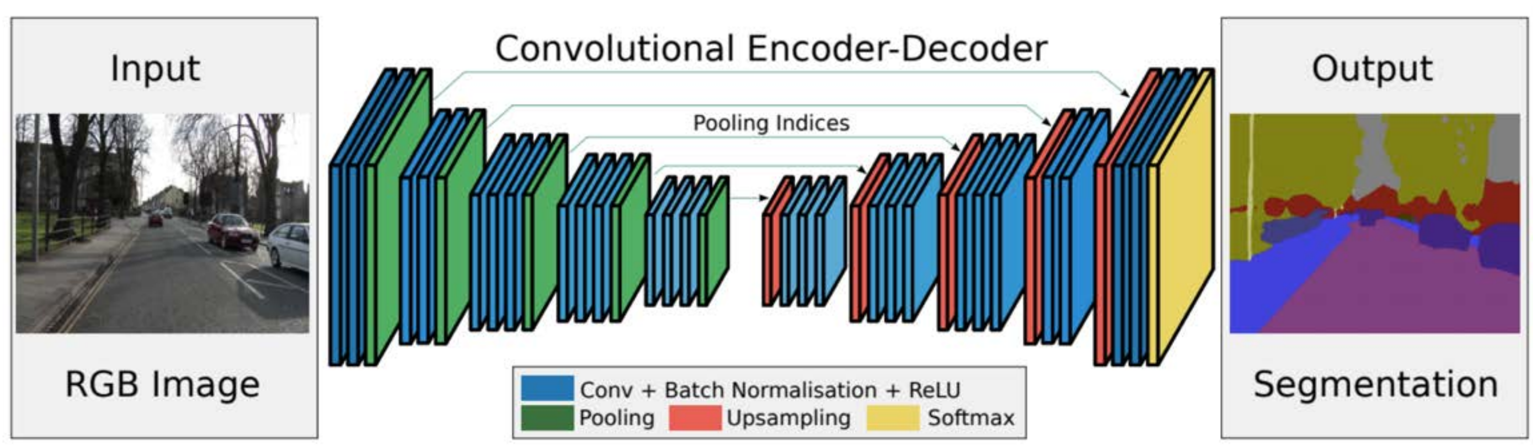


H. Noh, S. Hong, and B. Han, [Learning Deconvolution Network for Semantic Segmentation](), ICCV 2015

DeconvNet results

PASCAL VOC 2012	mIoU
Hypercolumns	59.2
ZoomOut	64.4
FCN-8	62.2
DeconvNet	69.6
Ensemble of DeconvNet and FCN	71.7

Similar architecture: SegNet

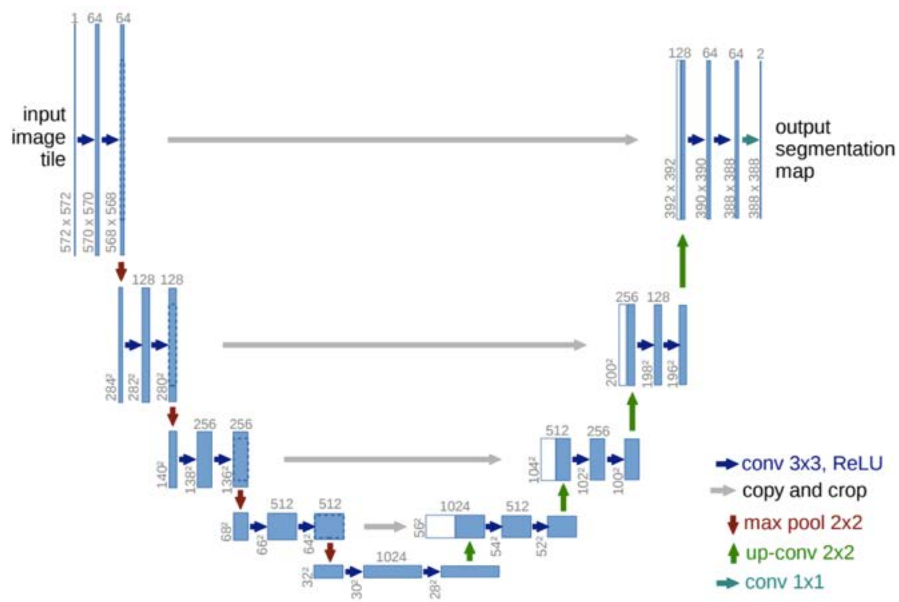


Drop the FC layers,
get better results

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-res, 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

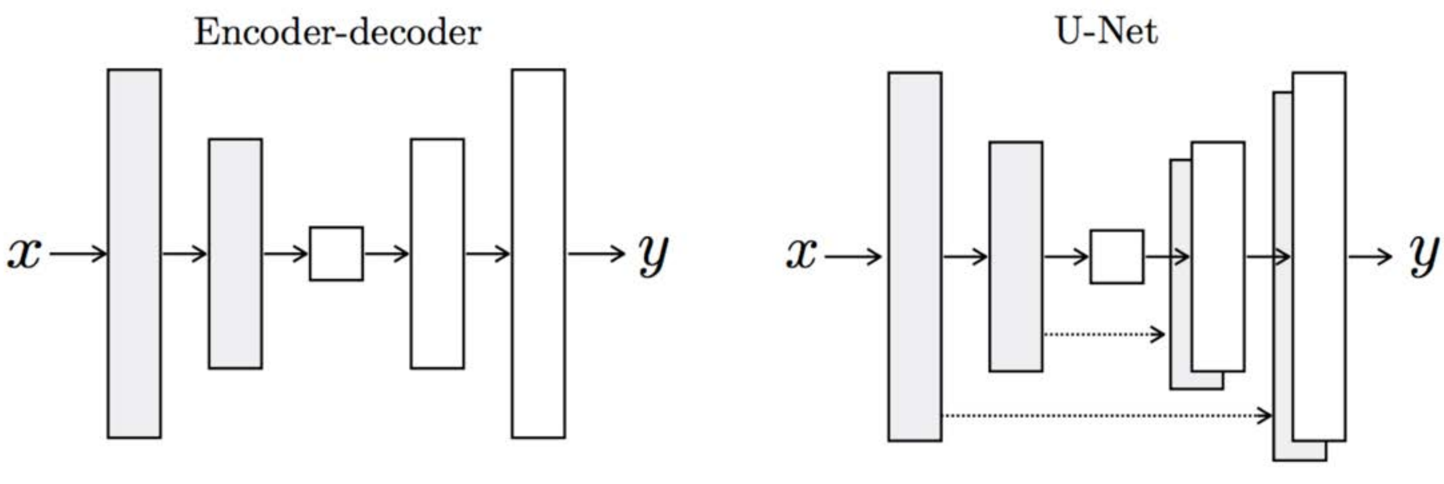
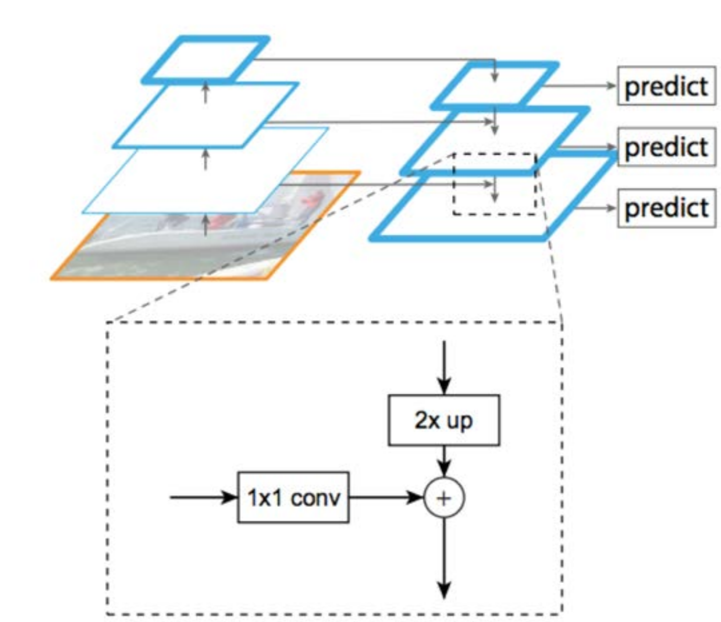


Figure source

Recall: Feature pyramid networks

- Improve predictive power of lower-level feature maps by adding contextual information from higher-level feature maps
- Predict different sizes of bounding boxes from different levels of the pyramid (but share parameters of predictors)



T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, [Feature pyramid networks for object detection](), CVPR 2017

Outline

- Early “hacks”
  - Hypercolumns
  - Zoom-out features
  - Fully convolutional networks
- Deep network operations for dense prediction
  - Transposed convolutions
  - Unpooling
  - Dilated convolutions

Dilated convolutions

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

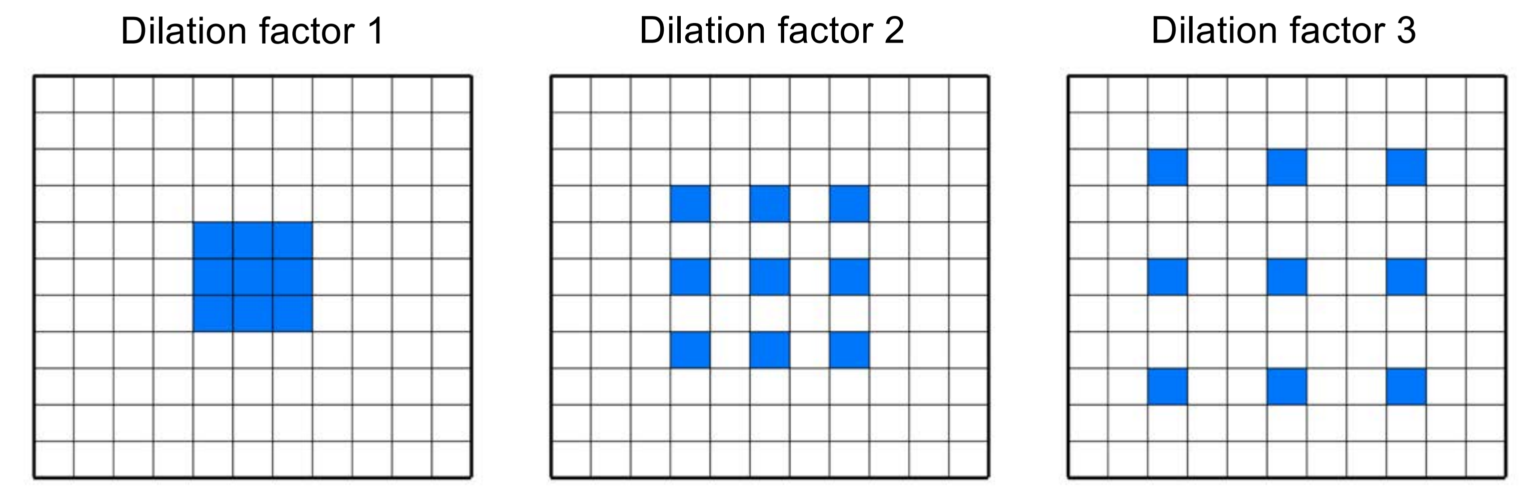
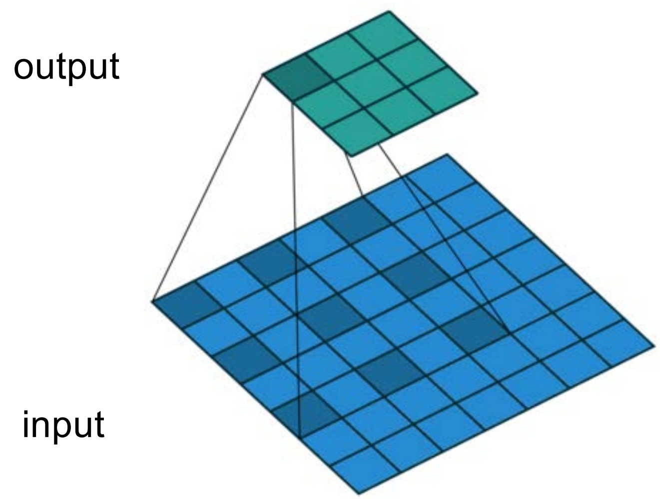


Image source

Dilated convolutions

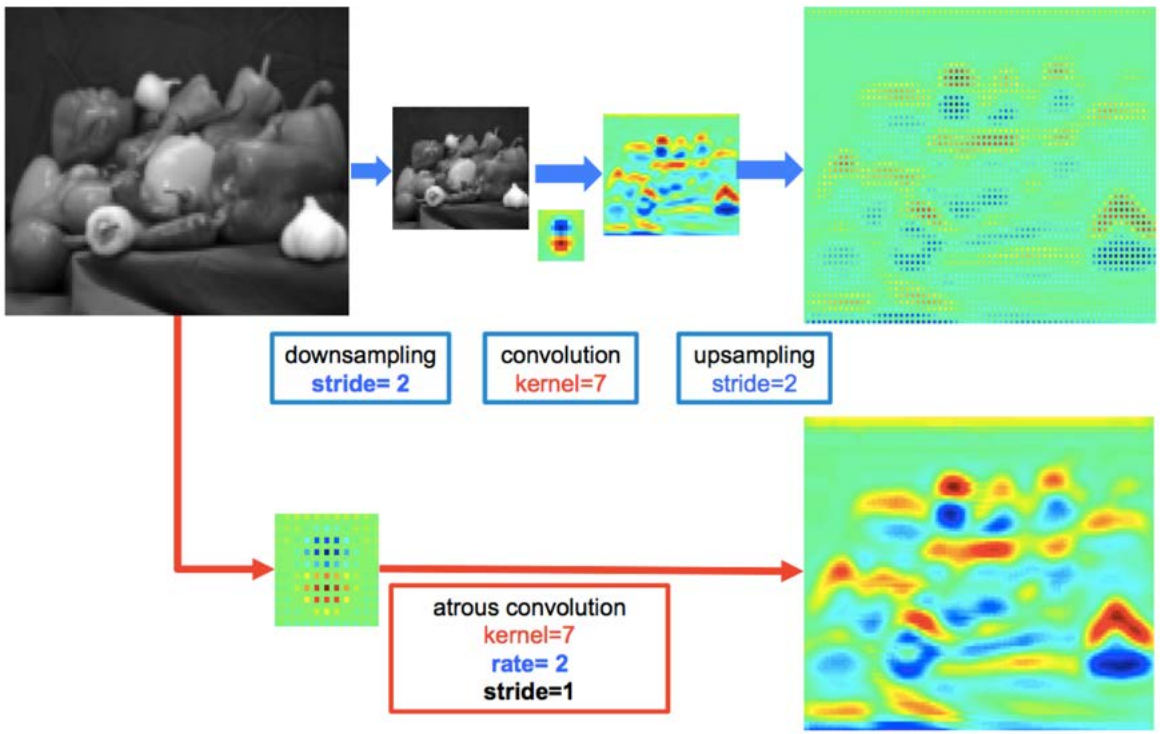
- Idea: instead of reducing spatial resolution of feature maps, use a large sparse filter



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

V. Dumoulin and F. Visin, [A guide to convolution arithmetic for deep learning](), arXiv 2018

Dilated convolutions



L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. Yuille, DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs, PAMI 2017

Dilated convolutions

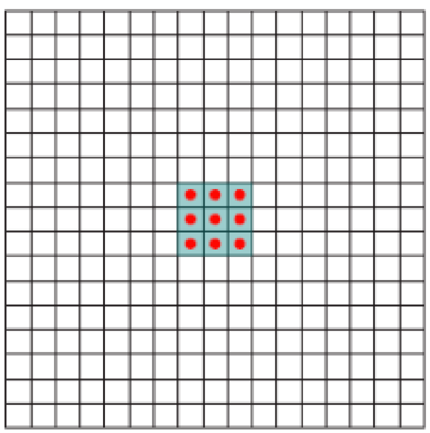
- Can be used in FCN to remove downsampling: change stride of max pooling layer from 2 to 1, dilate subsequent convolutions by factor of 2 (in theory, can be done without re-training any parameters)

L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. Yuille, DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs, PAMI 2017

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 Receptive field: 7x7 Receptive field: 15x15

F. Yu and V. Koltun, Multi-scale context aggregation by dilated convolutions, ICLR 2016

Dilated convolutions

- Context module with dilation
  - Returns same number of feature maps at the same resolution as the input, so can be plugged in to replace components of existing dense prediction architectures
  - Requires identity initialization

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

F. Yu and V. Koltun, Multi-scale context aggregation by dilated convolutions, ICLR 2016

Dilated convolutions: Evaluation

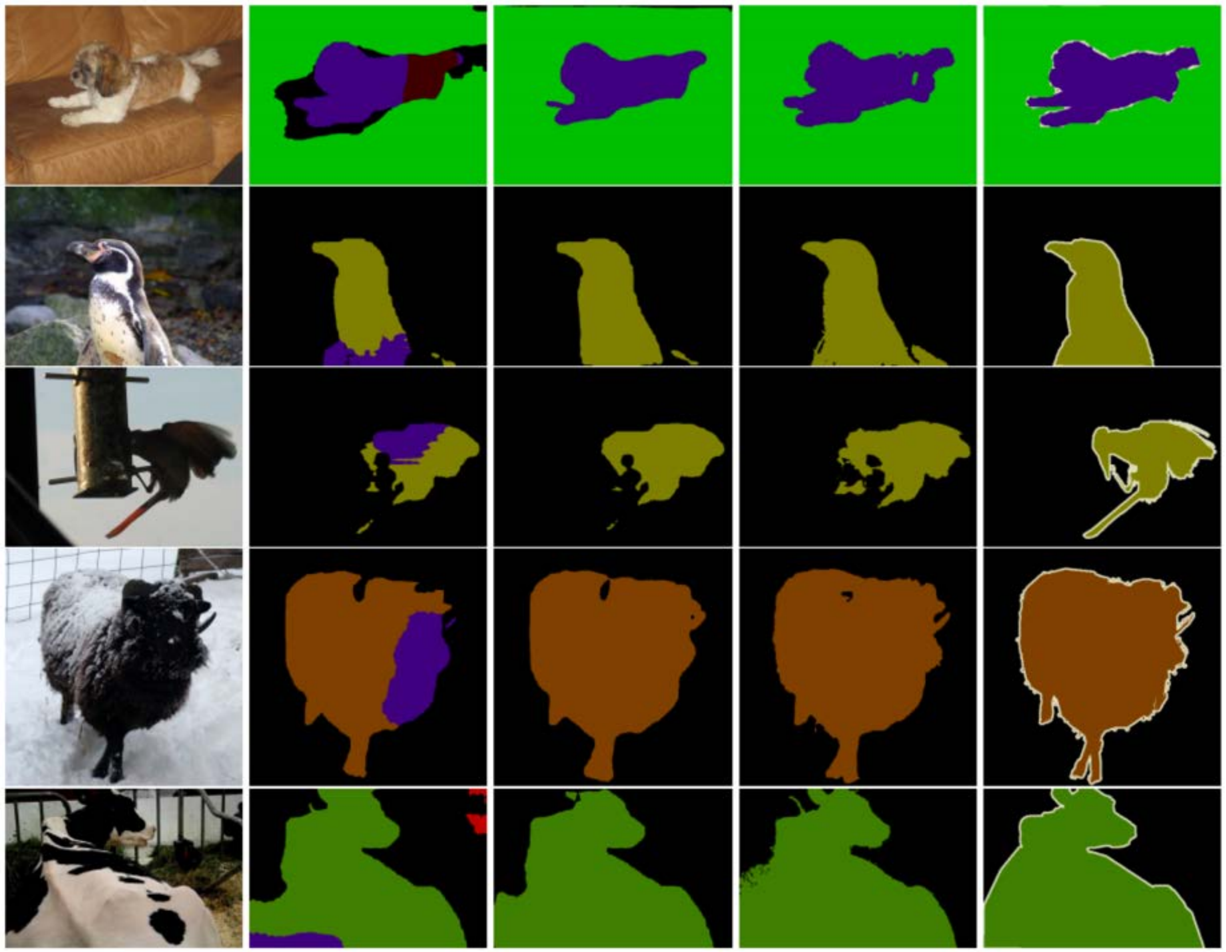
Results on VOC 2012

	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: re-implementation of FCN-8 with last two pooling layers dropped (5% better than original FCN-8)

F. Yu and V. Koltun, Multi-scale context aggregation by dilated convolutions, ICLR 2016

Dilated convolutions: Evaluation

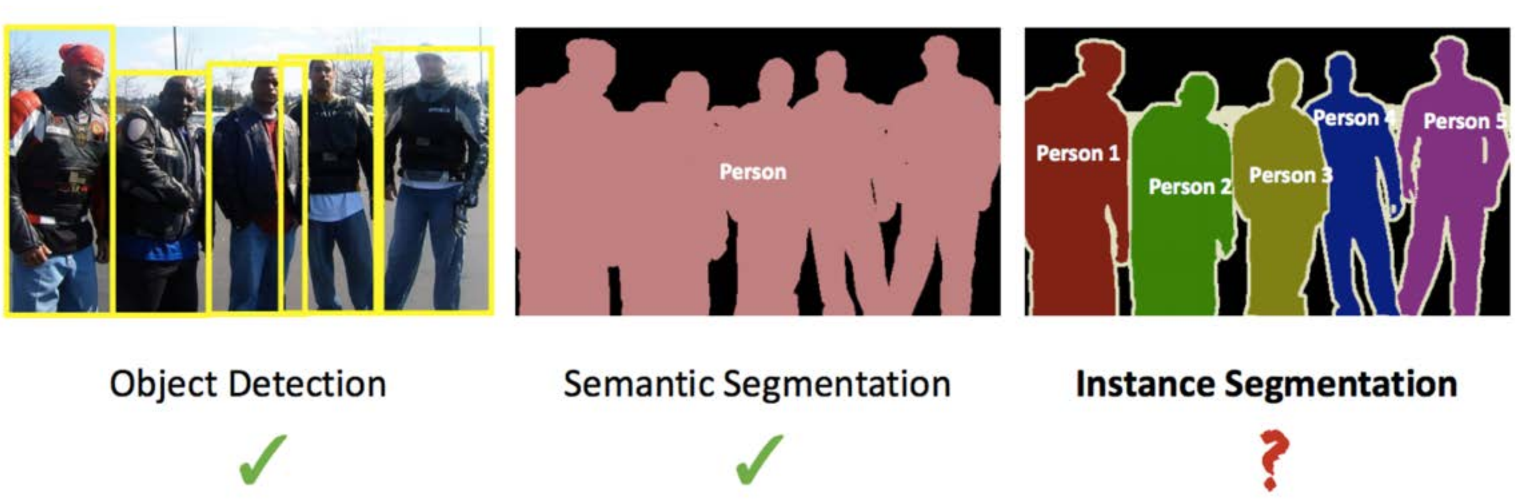


(a) Image (b) Front end (c) + Context (d) + CRF-RNN (e) Ground truth

Outline

- Early “hacks”
  - Hypercolumns
  - Zoom-out features
  - Fully convolutional networks
- Deep network operations for dense prediction
  - Transposed convolutions
  - Unpooling
  - Dilated convolutions
- Instance segmentation
  - Mask R-CNN

Instance segmentation

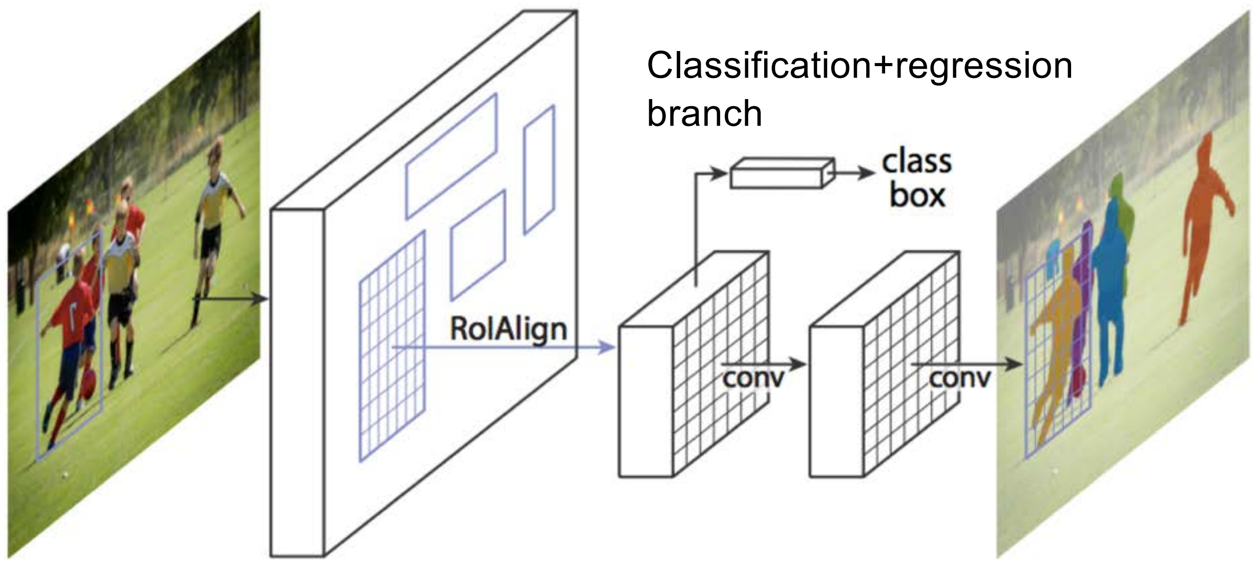


Source: [Kaiming He]

Mask R-CNN

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

Classification+regression branch

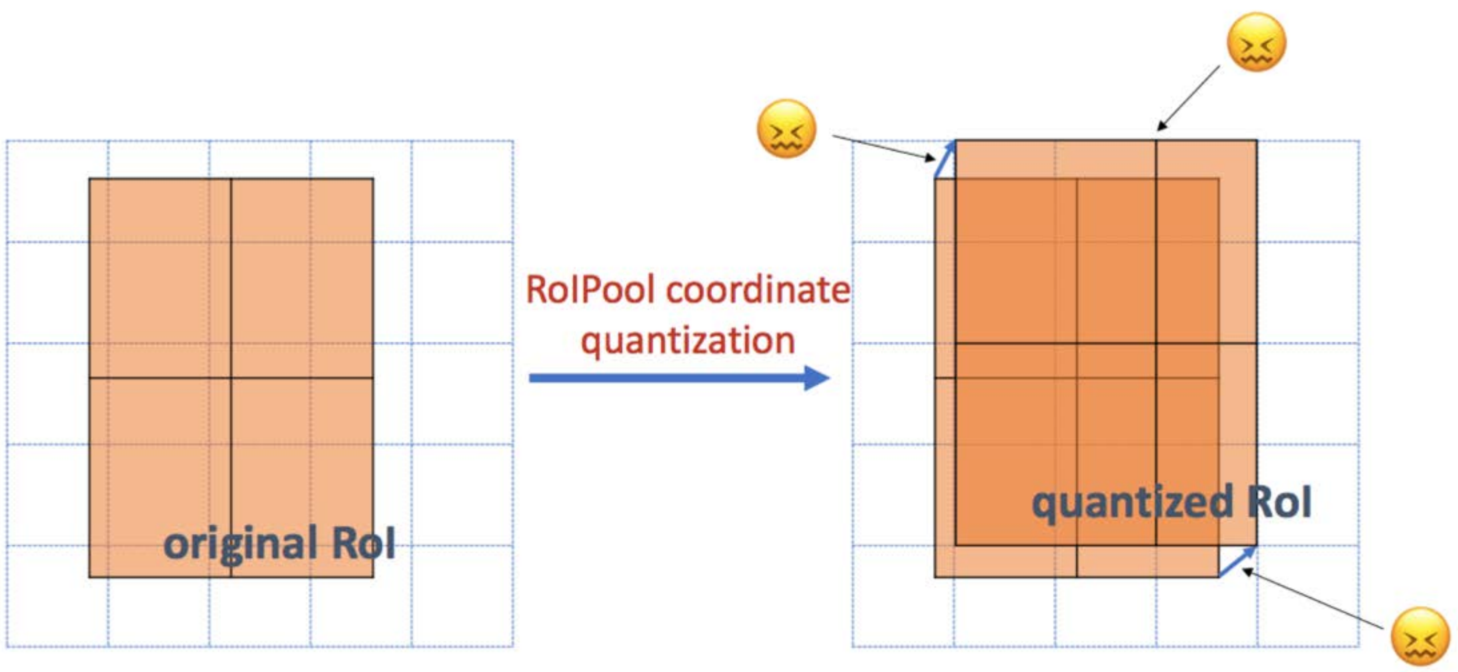


Mask branch: separately predict segmentation for each possible class

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

RoIAlign vs. RoIPool

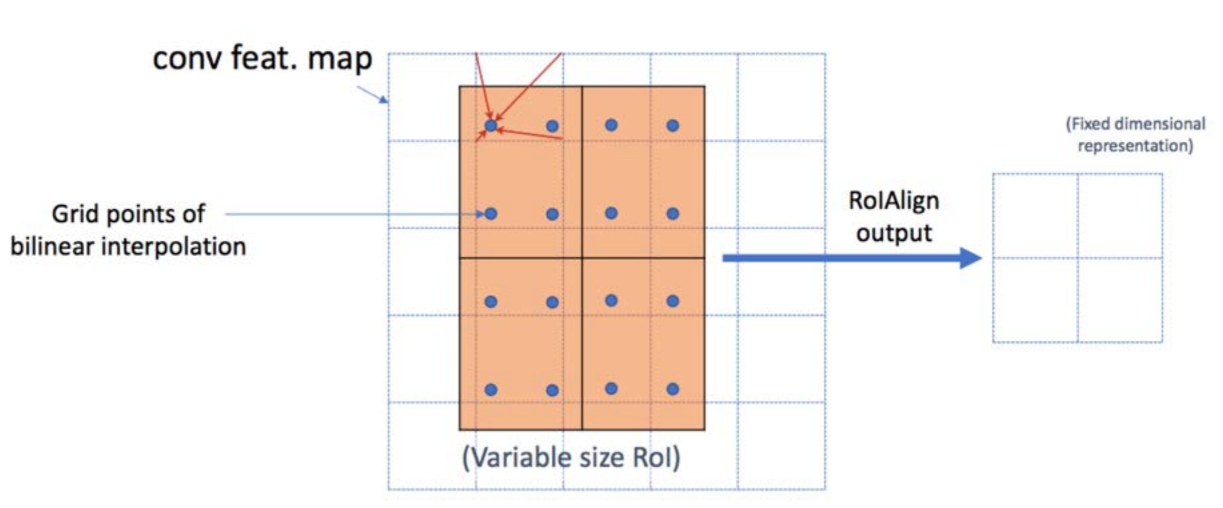
- RoIPool: nearest neighbor quantization



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

RoIAlign vs. RoIPool

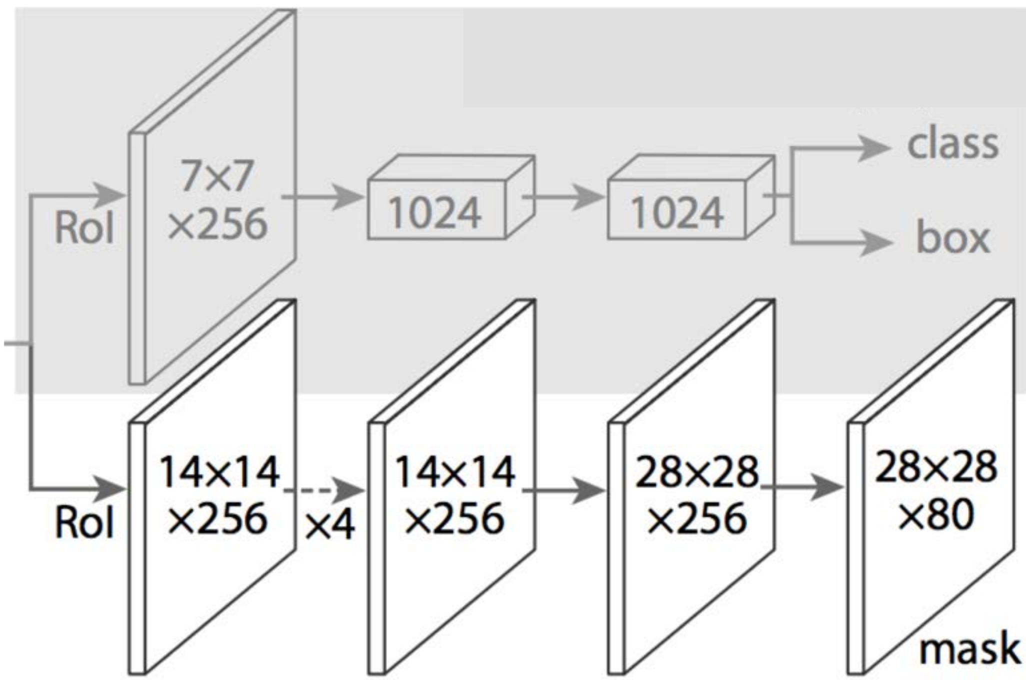
- RoIPool: nearest neighbor quantization
- RoIAlign: bilinear interpolation



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



Classification/regression head from an established object detector (e.g., FPN)

Separately predict binary mask for each class with per-pixel sigmoids, use average binary cross-entropy loss

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

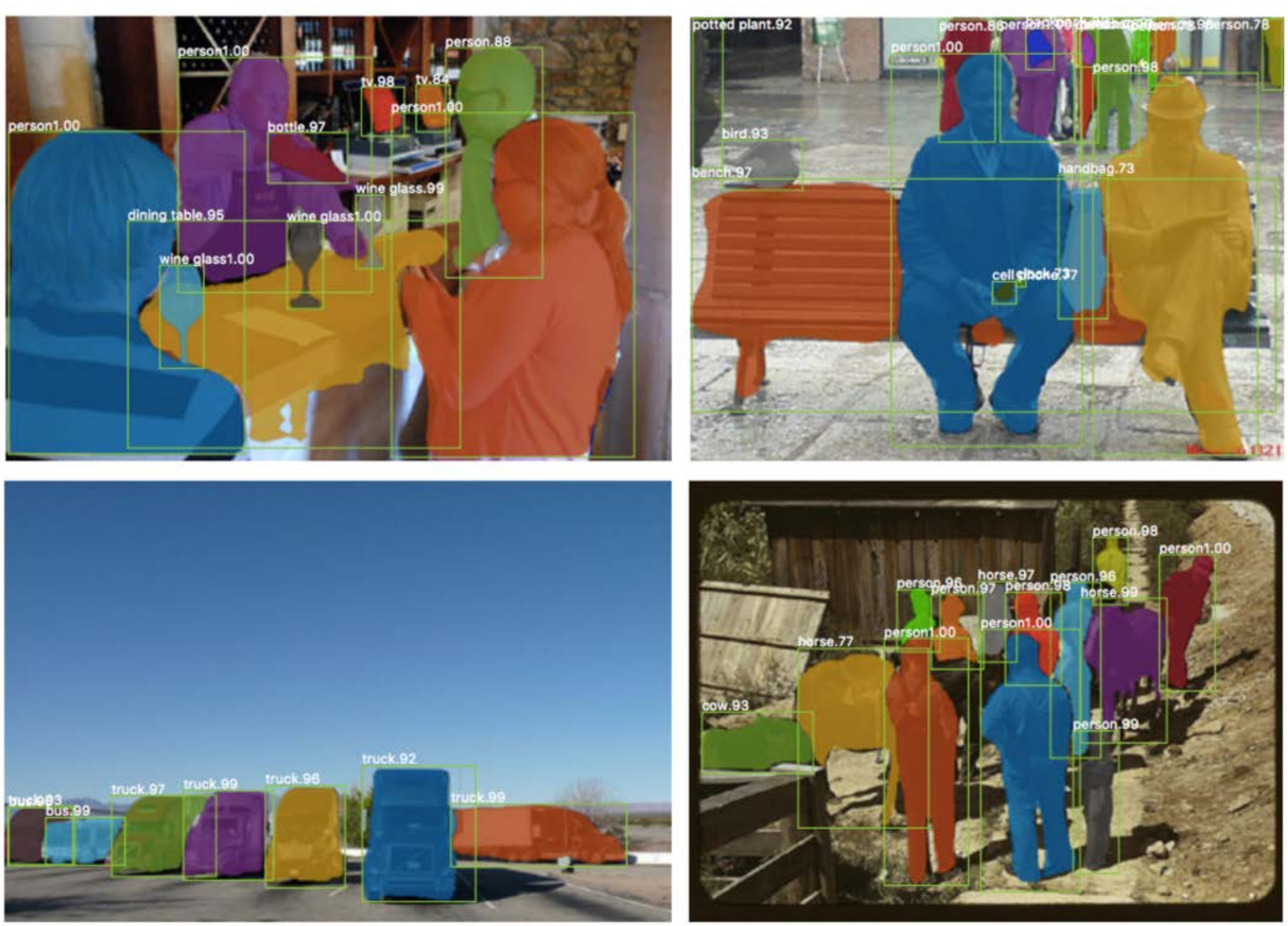
Mask R-CNN

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_{50}	AP_{75}	AP_S	AP_M	AP_L
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	-	-	-	-
Mask R-CNN	ResNet-101-C4	33.1	54.9	34.8	12.1	35.6	51.1
Mask R-CNN	ResNet-101-FPN	35.7	58.0	37.8	15.5	38.1	52.4
Mask R-CNN	ResNeXt-101-FPN	**37.1**	**60.0**	**39.4**	**16.9**	**39.9**	**53.5**

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)

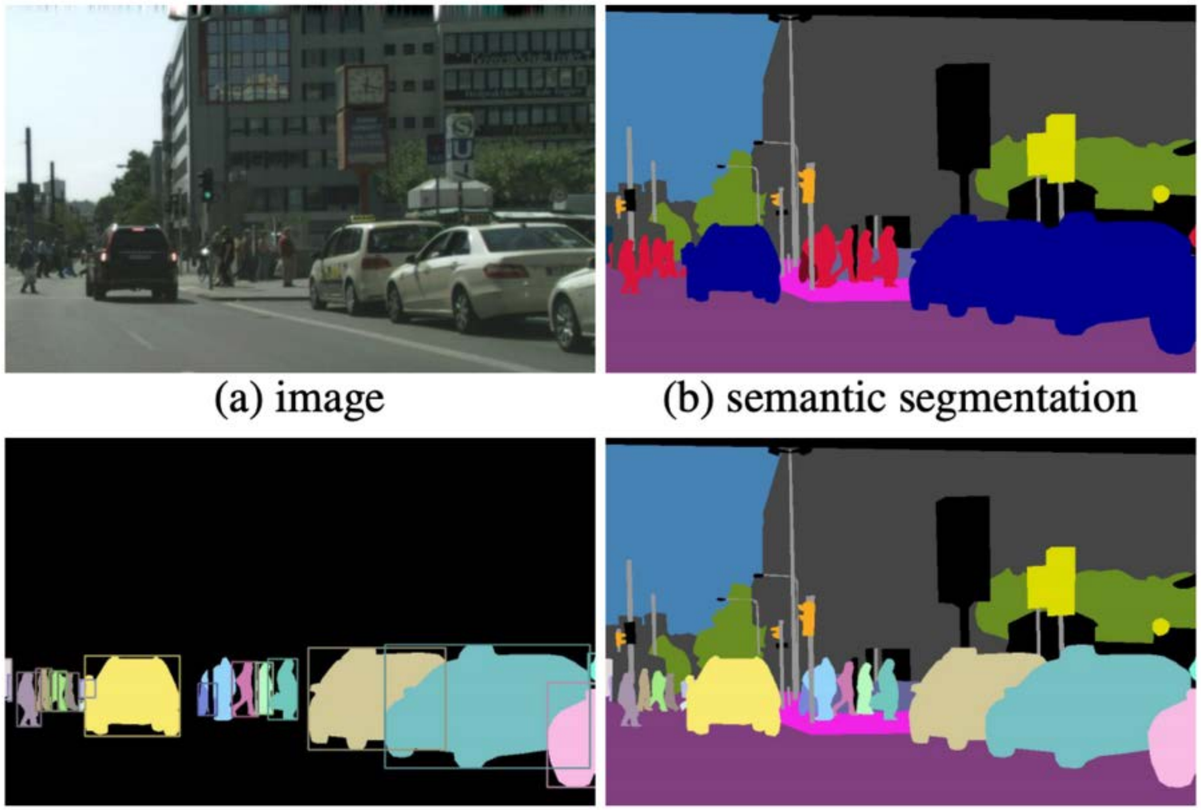
Keypoint prediction

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

Outline

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

Recently proposed task: Panoptic segmentation

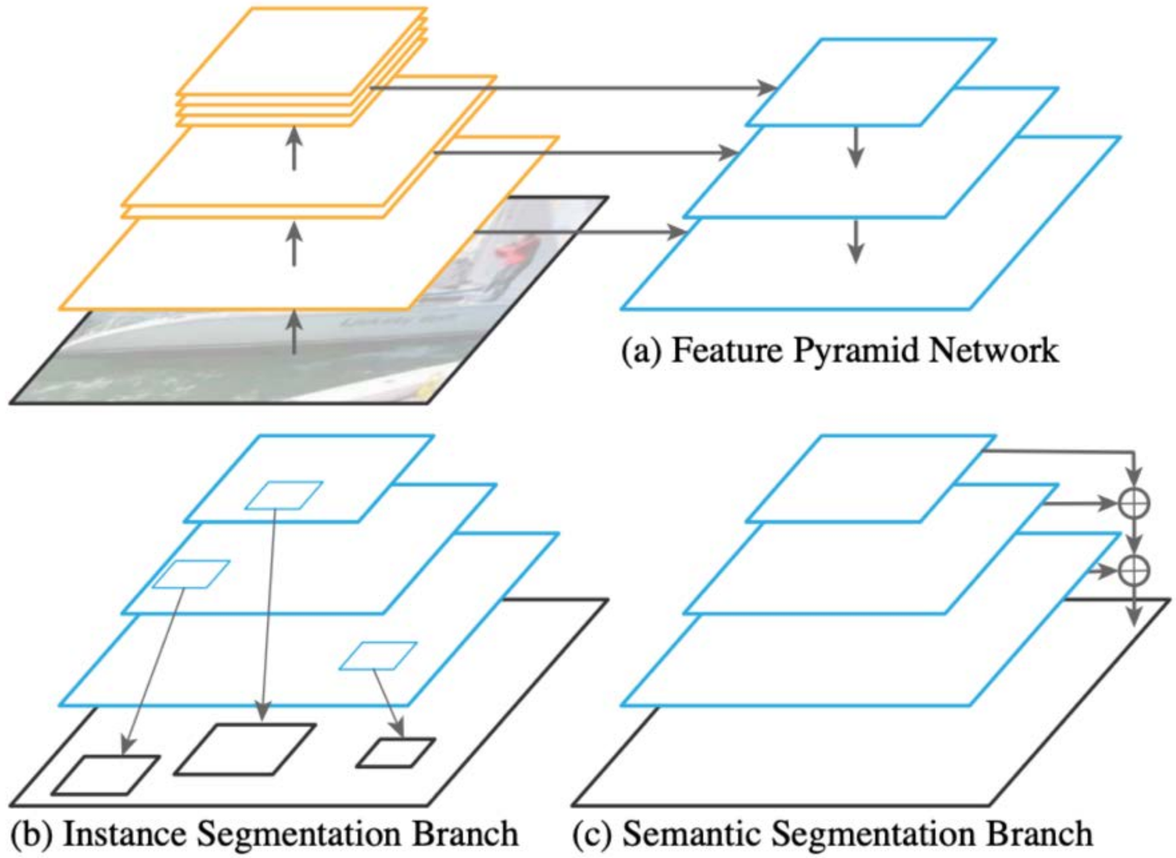


(a) image (b) semantic segmentation

(c) instance segmentation (d) panoptic segmentation

A. Kirillov et al. [Panoptic segmentation](). CVPR 2019

Panoptic feature pyramid networks



A. Kirillov et al. [Panoptic feature pyramid networks](). CVPR 2019

Panoptic feature pyramid networks

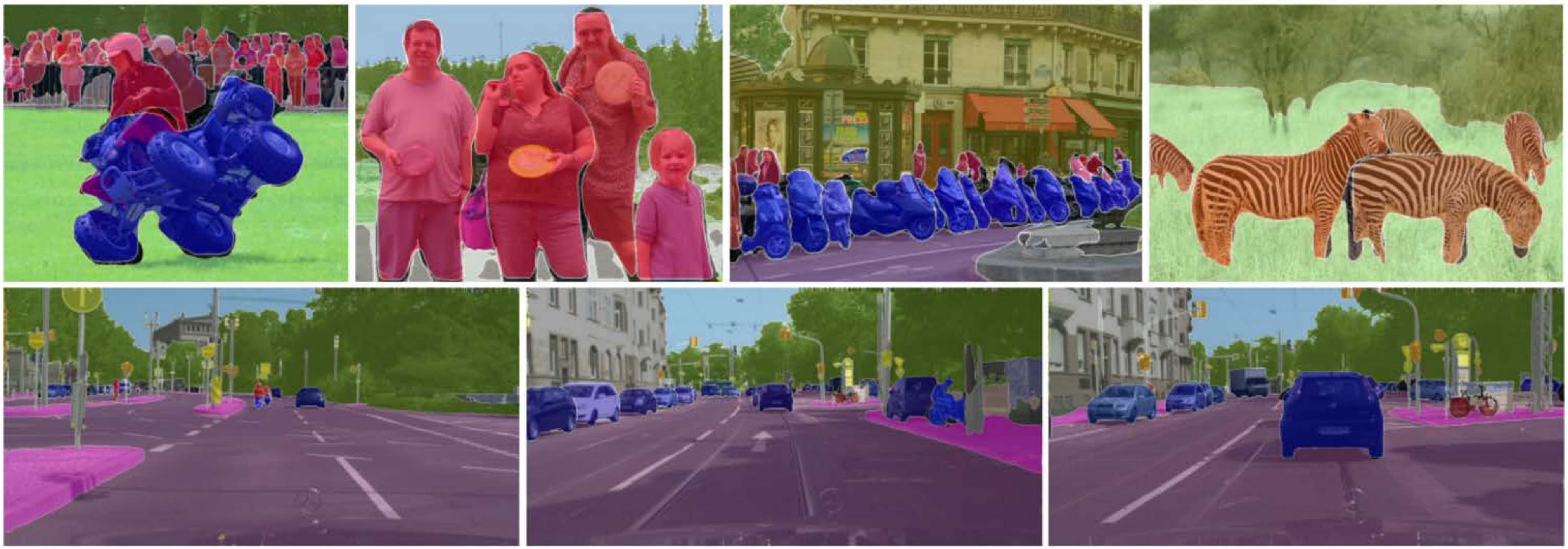


Figure 2: Panoptic FPN results on COCO (top) and Cityscapes (bottom) using a single ResNet-101-FPN network.

A. Kirillov et al. Panoptic feature pyramid networks. CVPR 2019

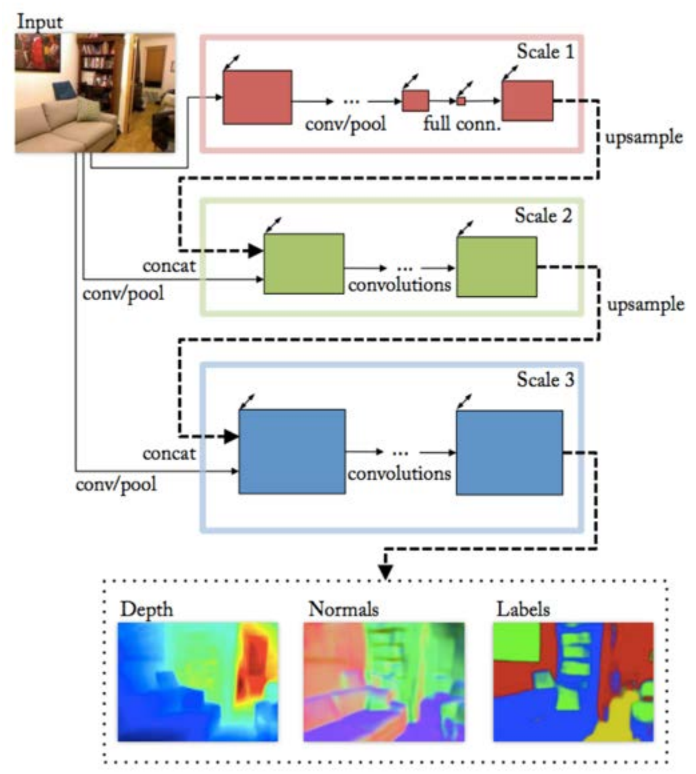
Another recent task: Amodal instance segmentation

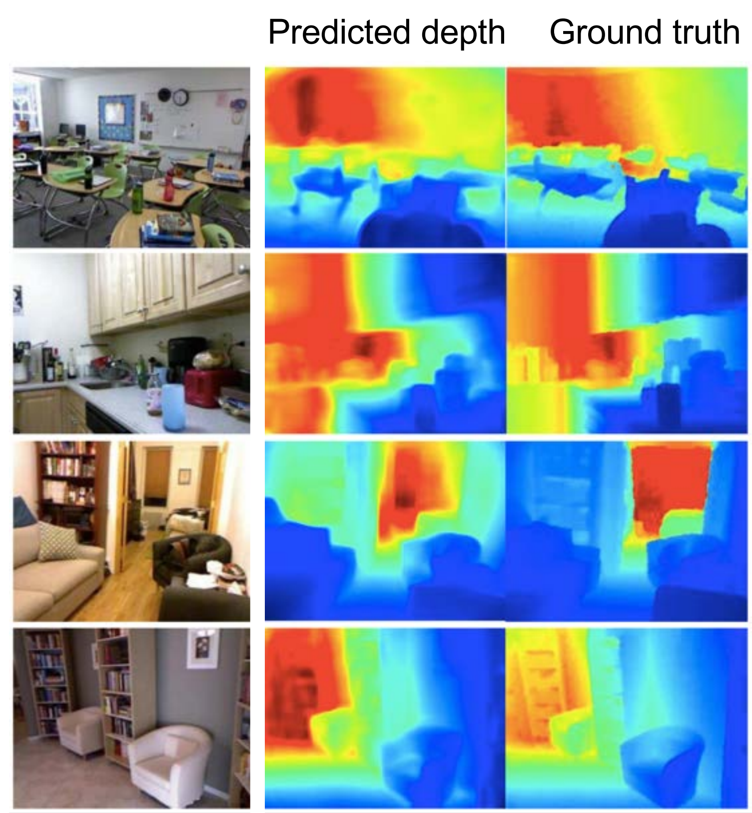
K. Li and J. Malik. [Amodal instance segmentation.]() ECCV 2016

Even more dense prediction problems

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

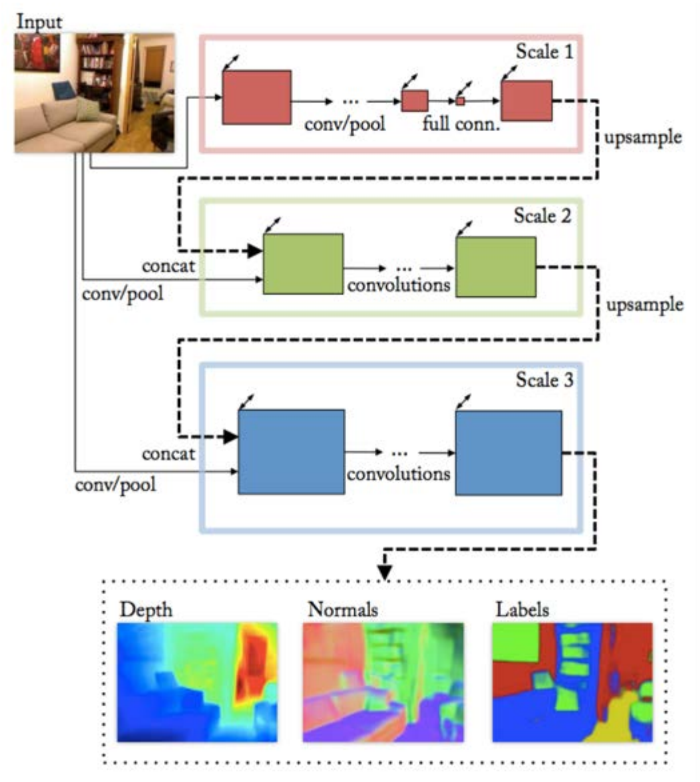
Depth and normal estimation

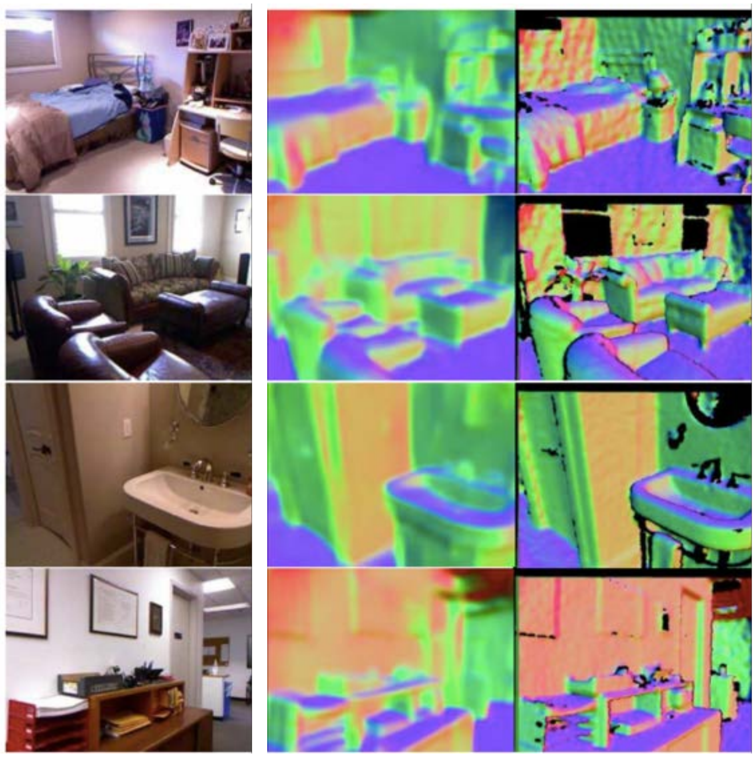




D. Eigen and R. Fergus, [Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture](), ICCV 2015

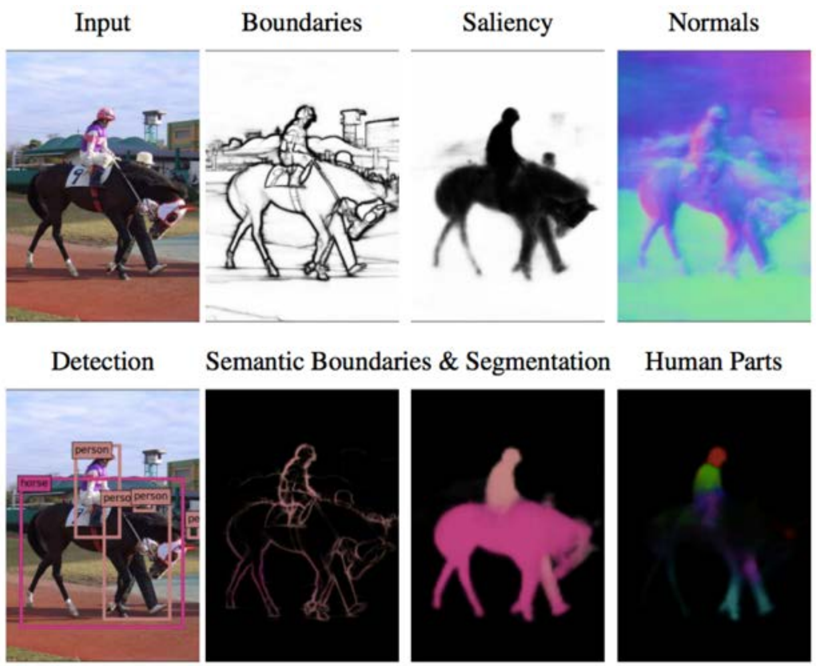
Depth and normal estimation





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



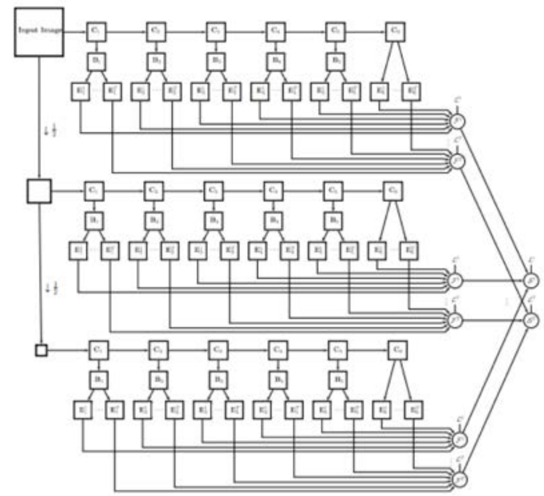
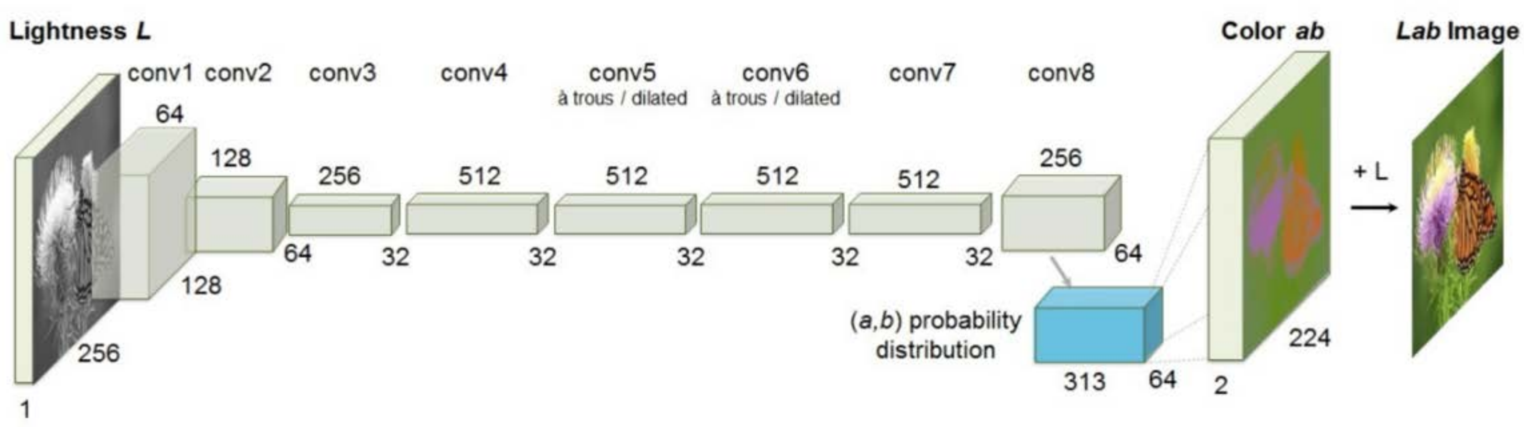


Figure 2: UberNet architecture: an image pyramid is formed by successive down-sampling operations, and each image is processed by a CNN with tied weights; the responses of the network at consecutive layers ($\mathbf{C}_i$) are processed with Batch Normalization ($\mathbf{B}_i$) and then fed to task-specific skip layers ($\mathbf{E}_i^t$); 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



R. Zhang, P. Isola, and A. Efros, [Colorful Image Colorization](), ECCV 2016