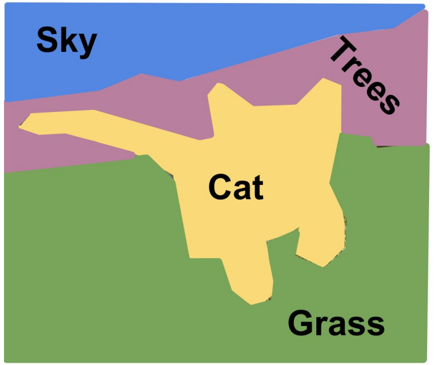
Segmentation

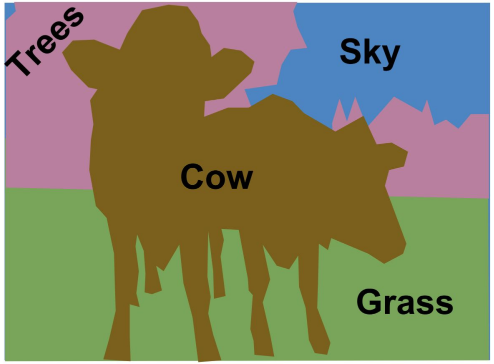
Vladislav Goncharenko
Spring 2019
MIPT, Moscow

Outline

- Segmentation task
- Simple solutions
- Upsampling methods
  - Unpooling
  - Transposed convolution
- FCN
- DeconvNet
- SegNet
- U-Net
- Mask R-CNN

Semantic Segmentation





Input image (C x H x W)

Output

mask of classes (1 x H x W)

Assumption

just class label for each pixel

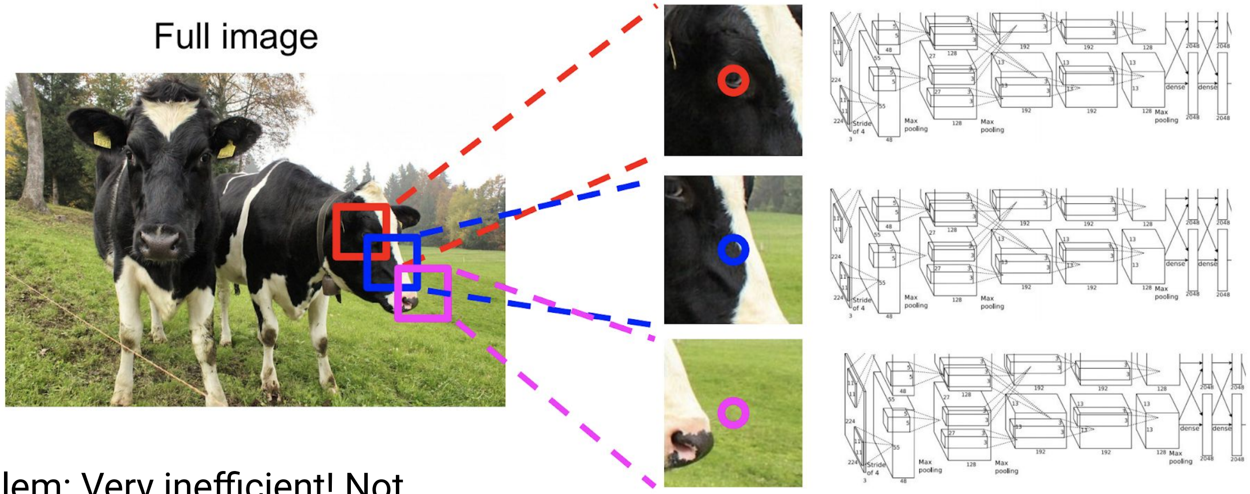
Naive solution

Problems?
comp. heavy

Extract patch

Classify center pixel with CNN

Full image



Cow

Cow

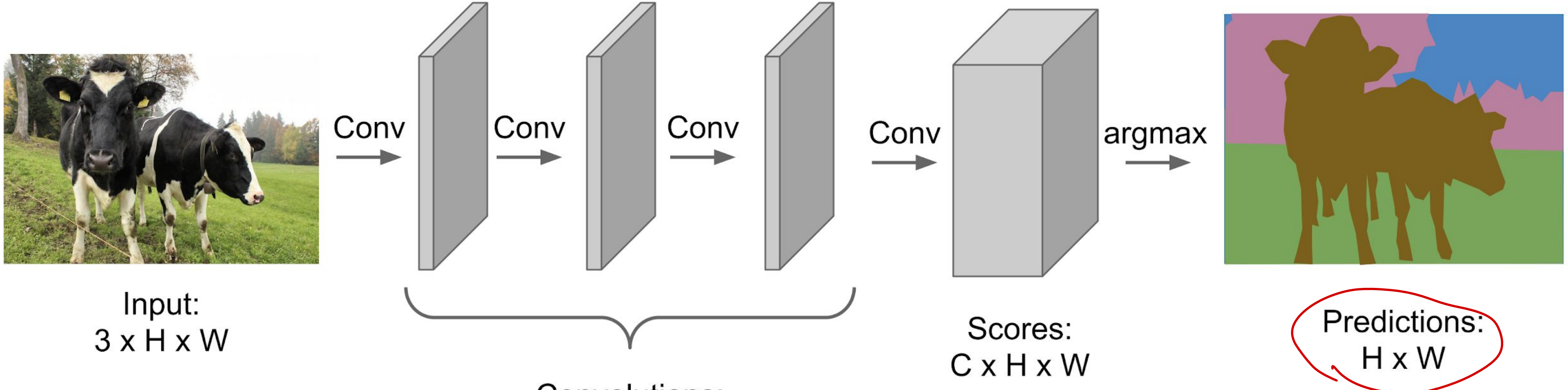
Grass

Problem: Very inefficient! Not reusing shared features between overlapping patches (same as R-CNN)

Farabet et al, “Learning Hierarchical Features for Scene Labeling,” TPAMI 2013
Pinheiro and Collobert, “Recurrent Convolutional Neural Networks for Scene Labeling”, ICML 2014

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
(remember VGG architecture)

conv (3×3) stride = 1

pad = 1

con(3,3)

conv(3,3)

⋮

conv(3,3) → #C = # labels(categ)

↓

Softmax

↓

argmax

⇓

categ.

Loss func:

Per-pixel cross-entropy

Problems:

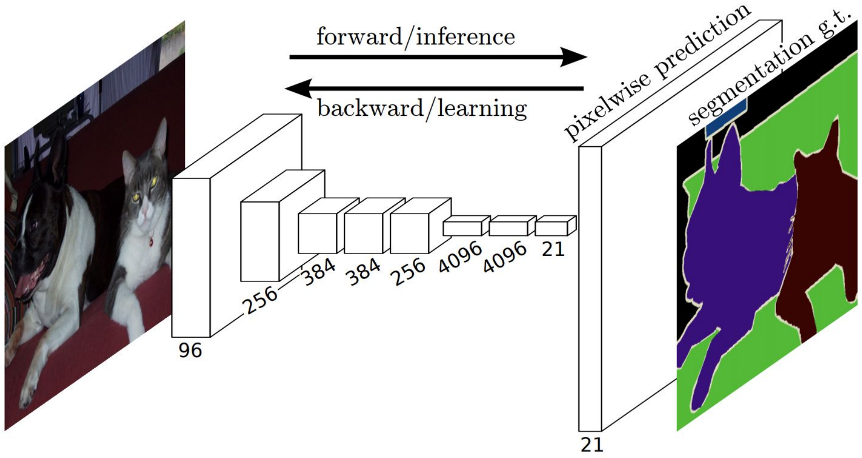
1) Receptive Field

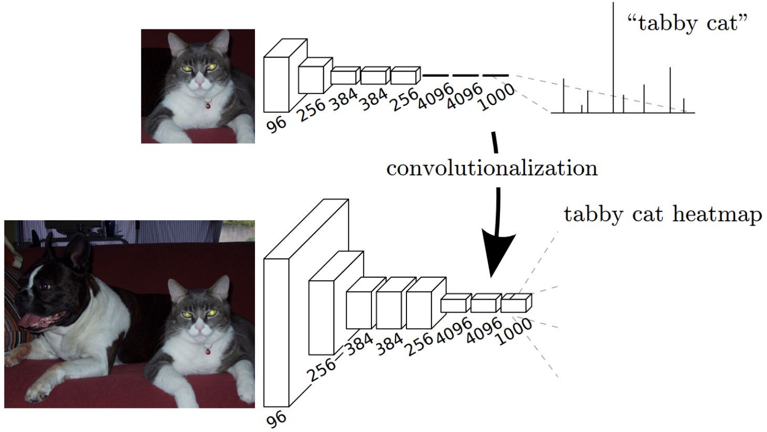
conv(3x3, 1, 1)
conv(3x3, 1, 1) $\Rightarrow$ Rec. Field size = 5x5

$$RFS = N \times N \, , \quad \frac{N-1}{2}$$

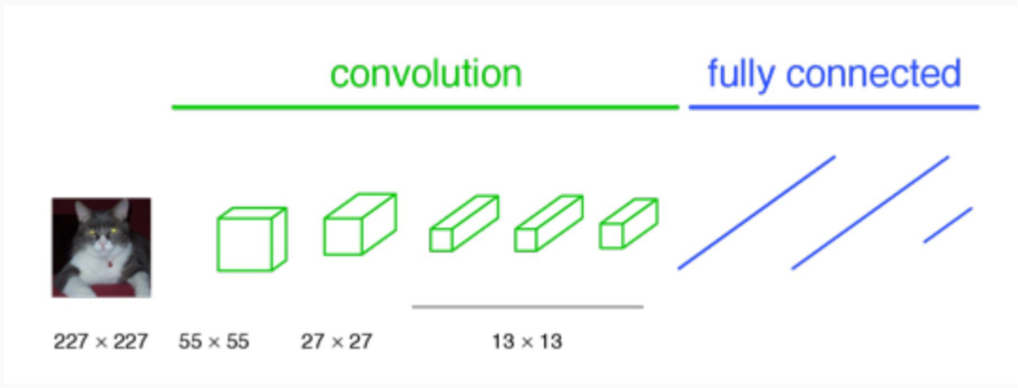
2) High-Res img is expensive to Conv!

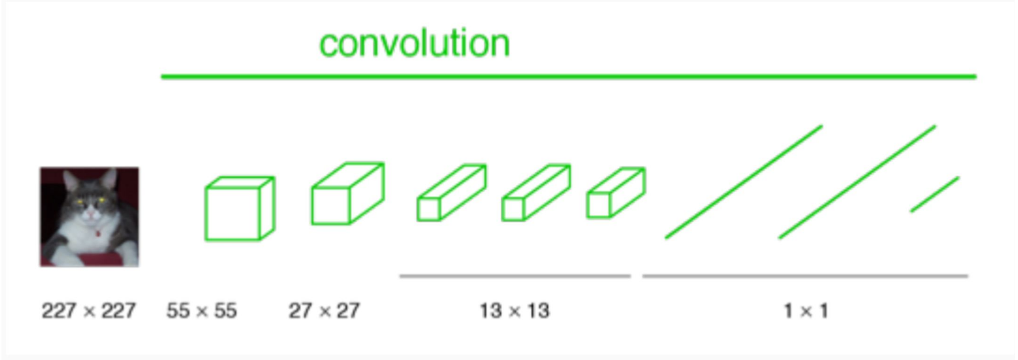
FCN – Fully Convolutional Network



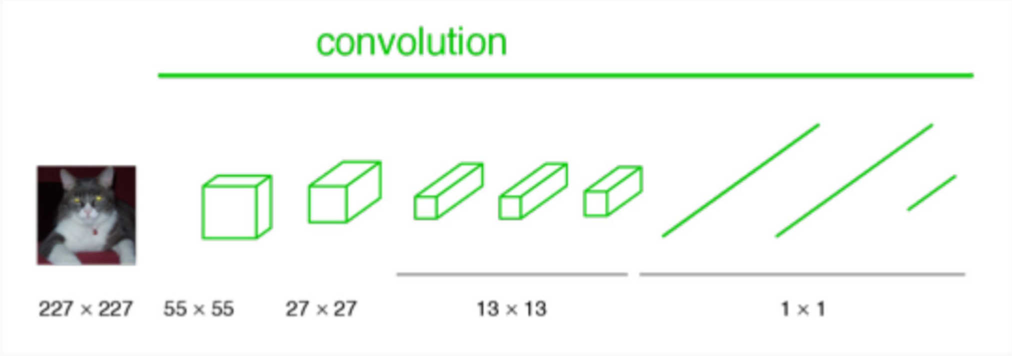


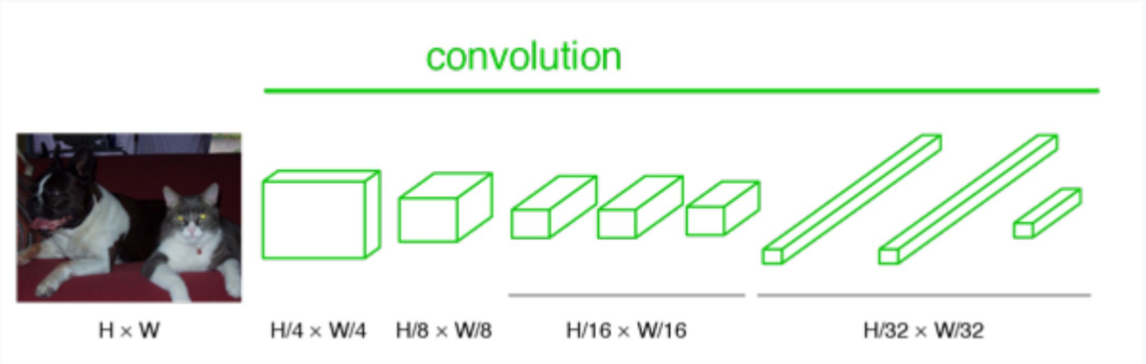
Regular CNN to FCN



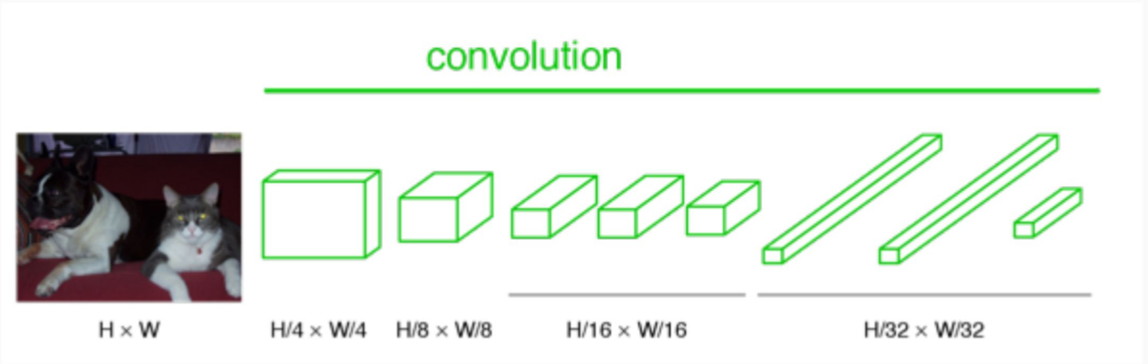


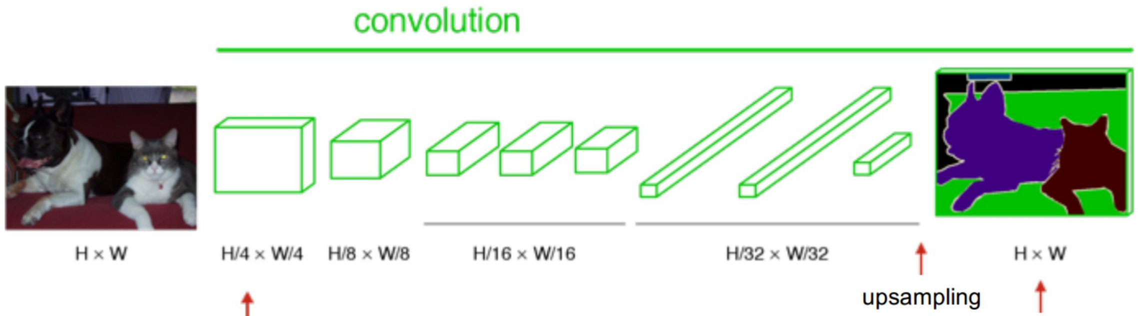
Regular CNN to FCN





Regular CNN to FCN

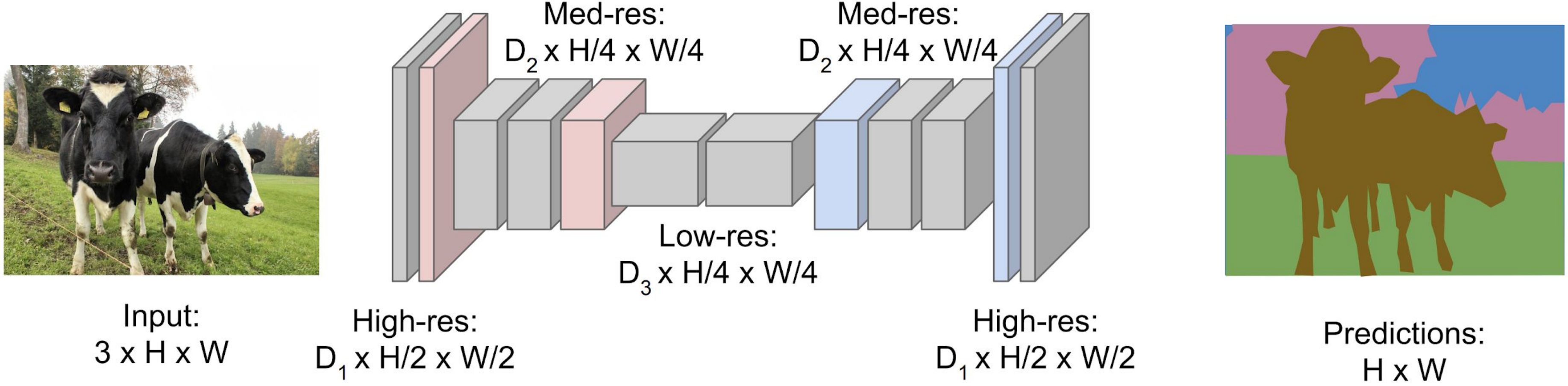




Problem: we lose resolution

Downsampling and Upsampling

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



Simple unpooling

Avg - pooling

Nearest Neighbor

1	2
3	4

Input: 2 x 2

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

Output: 4 x 4

"Bed of Nails"

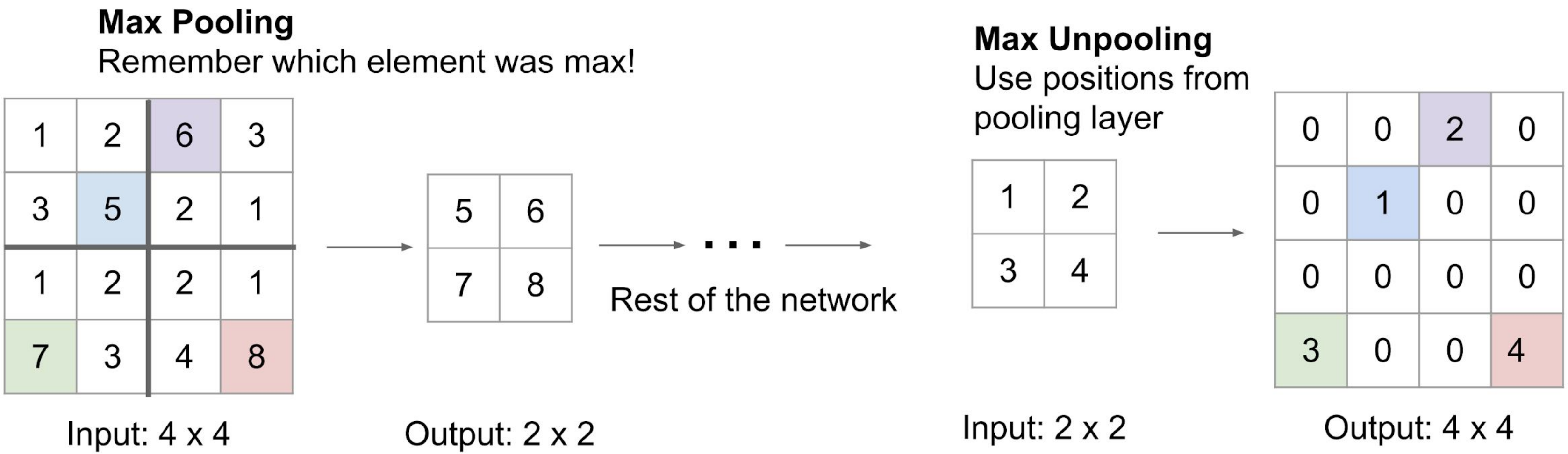
1	2
3	4

Input: 2 x 2

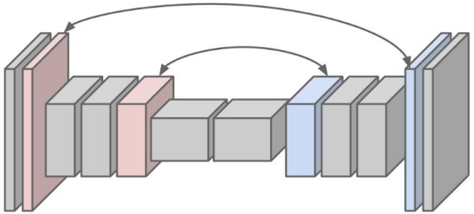
1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Output: 4 x 4

Max Unpooling



Corresponding pairs of downsampling and upsampling layers



Convolution as Matrix Multiplication (1D)

We can express convolution in terms of a matrix multiplication

$$\vec{x} * \vec{a} = X\vec{a}$$

$$\underbrace{\begin{bmatrix} x & y & z & 0 & 0 & 0 \\ 0 & x & y & z & 0 & 0 \\ 0 & 0 & x & y & z & 0 \\ 0 & 0 & 0 & x & y & z \end{bmatrix}}_{\text{conv matrix}} \begin{bmatrix} 0 \\ a \\ b \\ c \\ d \\ 0 \end{bmatrix} = \begin{bmatrix} ay + bz \\ ax + by + cz \\ bx + cy + dz \\ cx + dy \end{bmatrix}$$

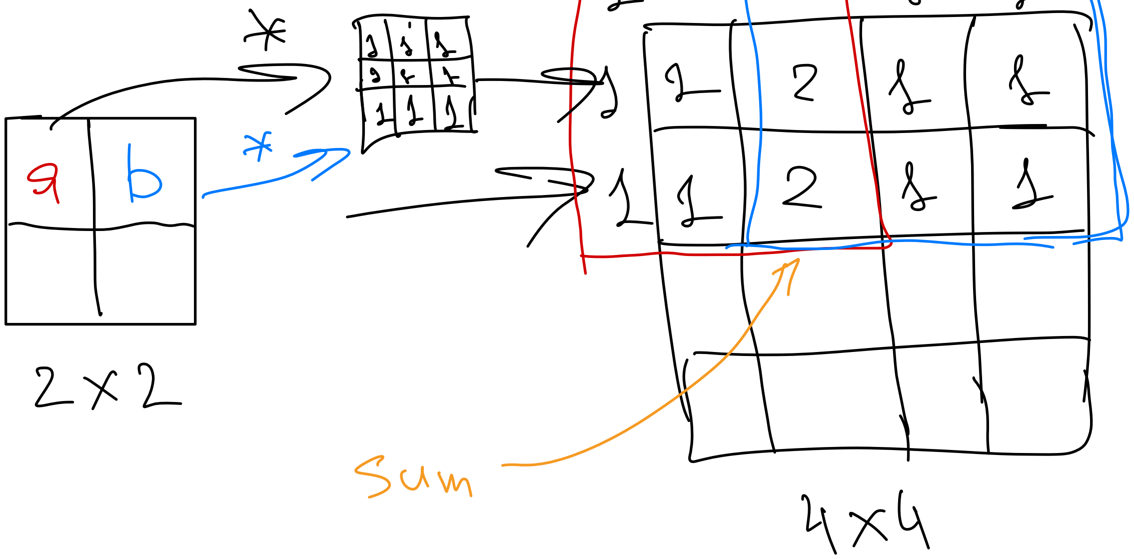
Example: 1D conv, kernel size=3, stride=1, padding=1

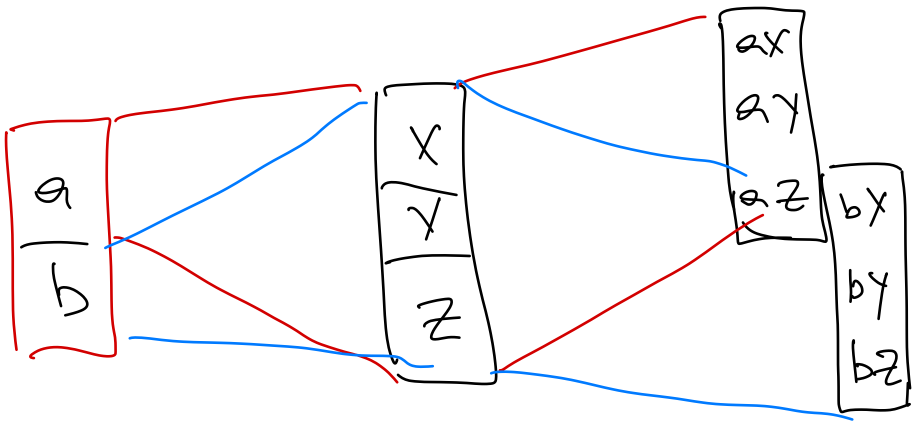
Convolution transpose multiplies by the transpose of the same matrix:

$$\vec{x} *^T \vec{a} = X^T\vec{a}$$

$$\begin{bmatrix} x & 0 & 0 & 0 \\ y & x & 0 & 0 \\ z & y & x & 0 \\ 0 & z & y & x \\ 0 & 0 & z & y \\ 0 & 0 & 0 & z \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} ax \\ ay + bx \\ az + by + cx \\ bz + cy + dx \\ cz + dy \\ dz \end{bmatrix}$$

When stride=1, convolution transpose is just a regular convolution (with different padding rules) stride = 2 3x3 conv





Terms

Some researchers denote Transposed convolution with term “Deconvolution”. However:

deconvolution is an algorithm-based process used to reverse the effects of convolution on recorded data [Wikipedia]

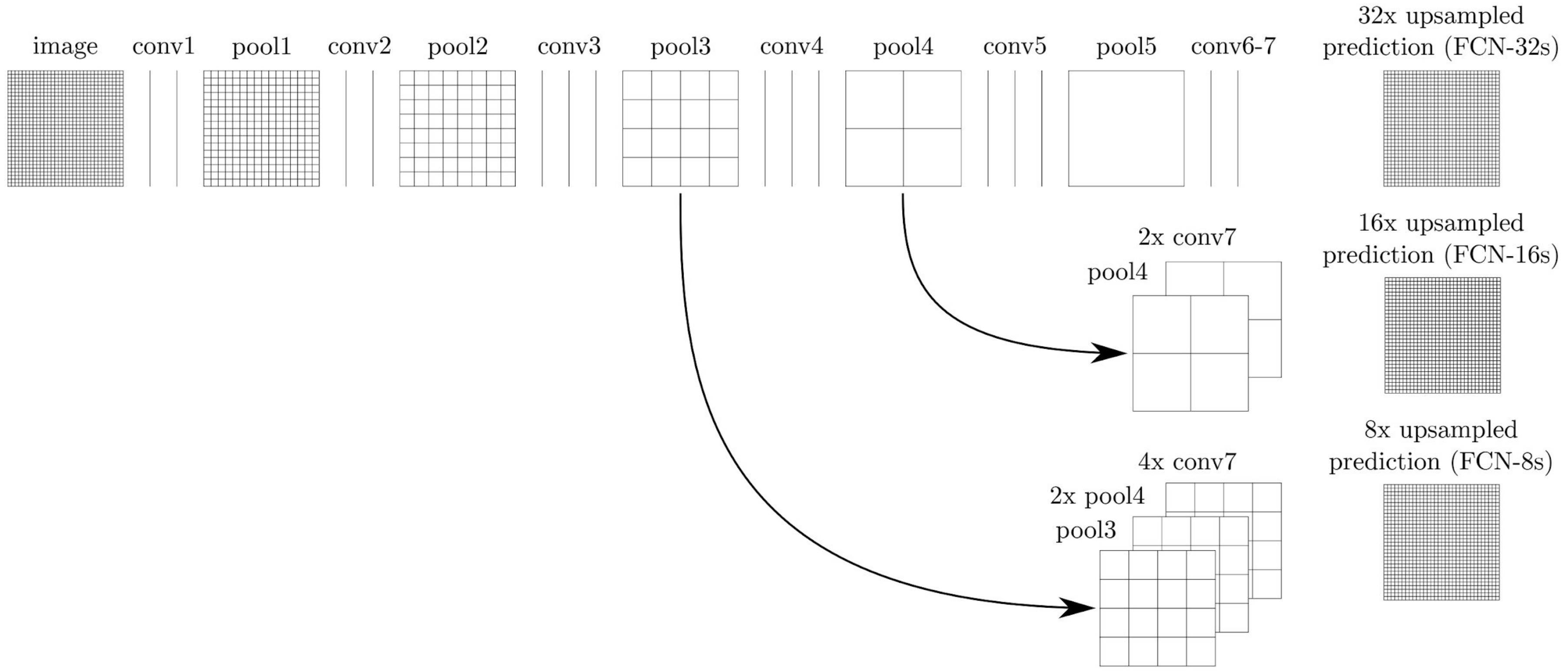
What we do is not a deconvolution.

Please be precise in terms

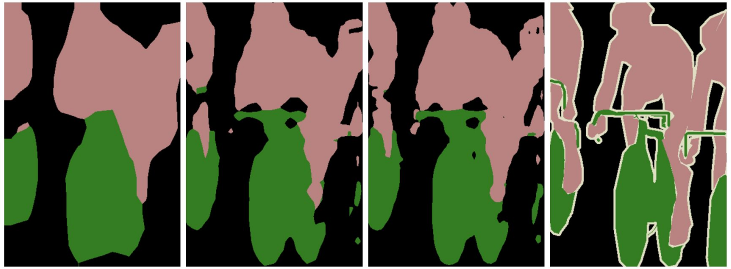
Right terms:

- Transposed convolution
- Upconvolution

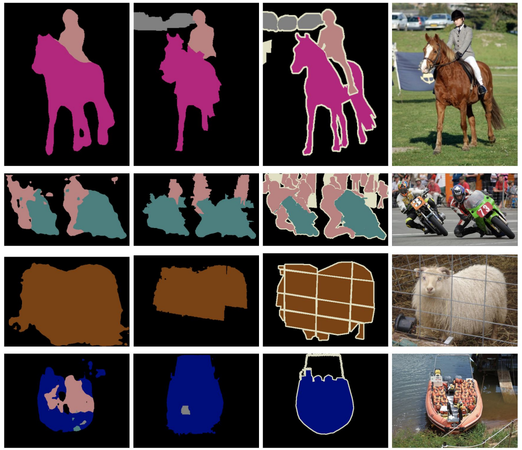
FCN: Introduce skip connections



FCN: Results

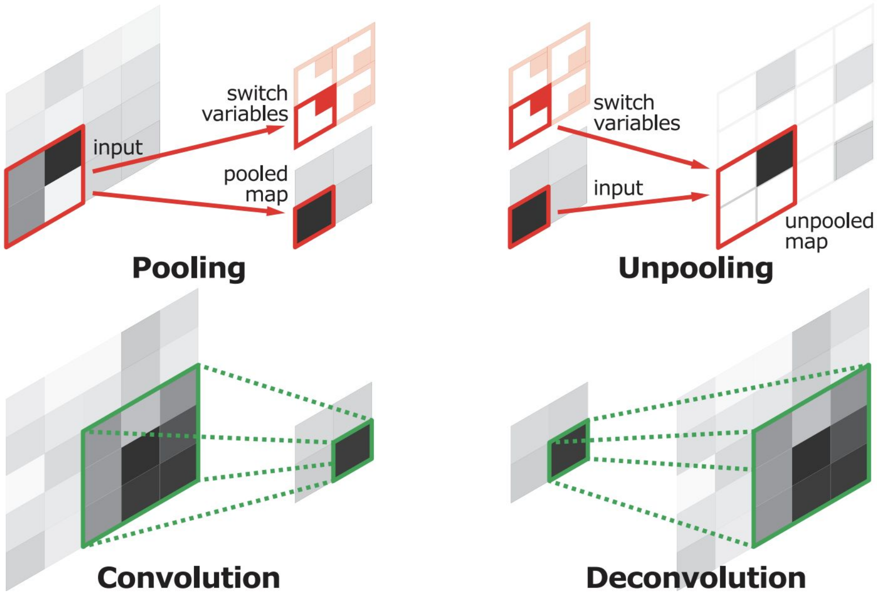


	pixel acc.	mean acc.	mean IU	f.w. IU
FCN-32s-fixed	83.0	59.7	45.4	72.0
FCN-32s	89.1	73.3	59.4	81.4
FCN-16s	90.0	75.7	62.4	83.0
FCN-8s	**90.3**	**75.9**	**62.7**	**83.2**



	FCN-AlexNet	FCN-VGG16	FCN-GoogLeNet[4]
mean IU	39.8	**56.0**	42.5
forward time	50 ms	210 ms	59 ms
conv. layers	8	16	22
parameters	57M	134M	6M
rf size	355	404	907
max stride	32	32	32

DeconvNet: use Max Unpooling



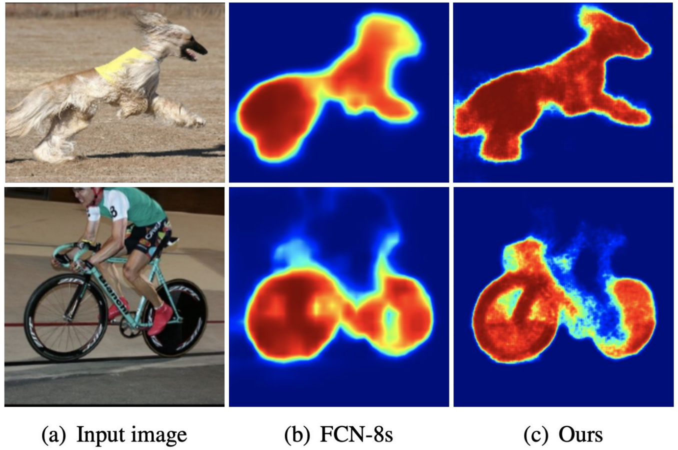


Figure 5. Comparison of class conditional probability maps from FCN and our network (top: dog, bottom: bicycle).

DeconvNet: use Max Unpooling

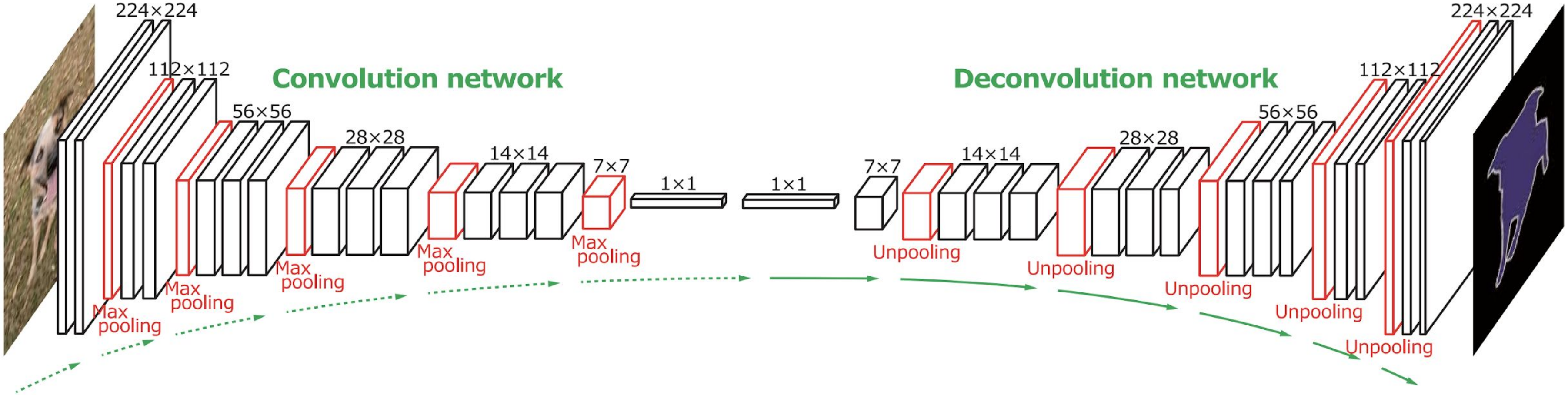
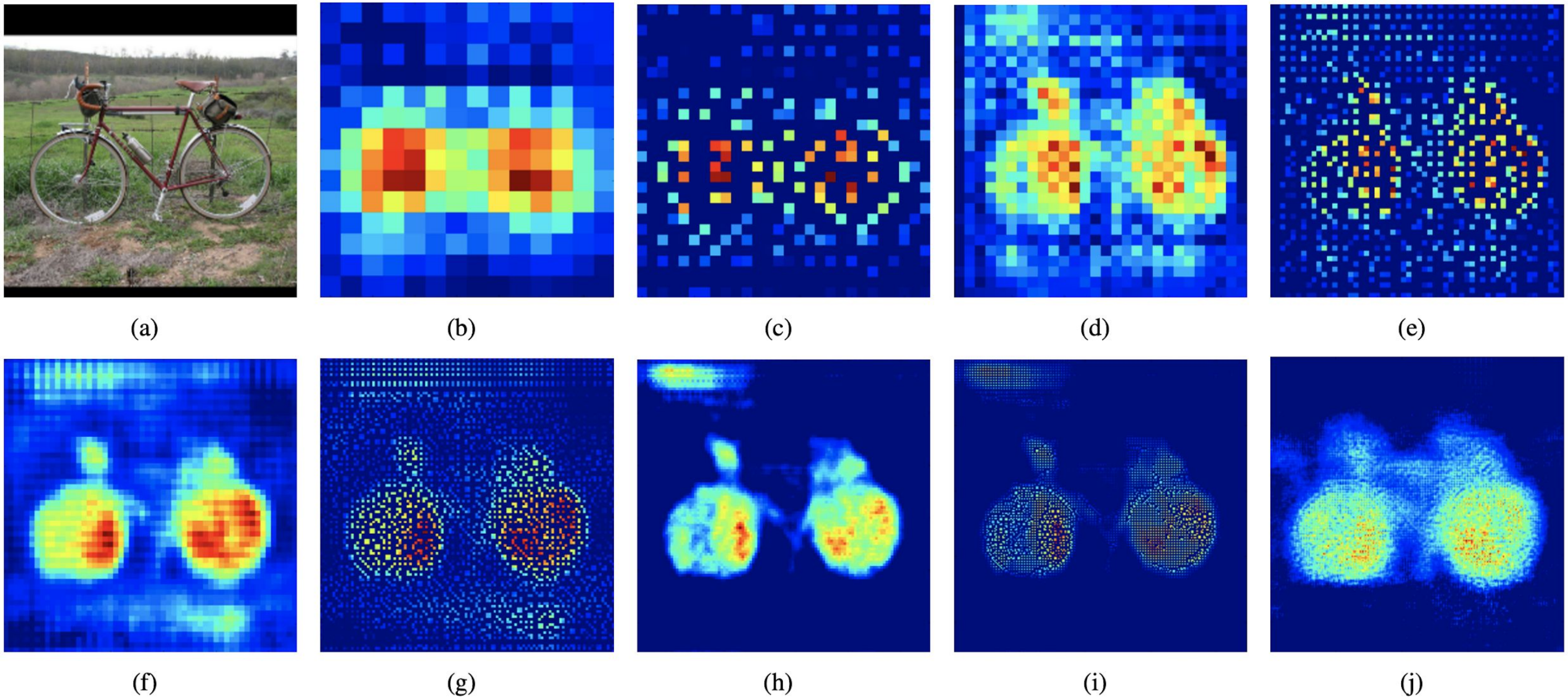


Figure 2. Overall architecture of the proposed network. On top of the convolution network based on VGG 16-layer net, we put a multi-layer deconvolution network to generate the accurate segmentation map of an input proposal. Given a feature representation obtained from the convolution network, dense pixel-wise class prediction map is constructed through multiple series of unpooling, deconvolution and rectification operations.

DeconvNet: use Max Unpooling



SegNet: Further development

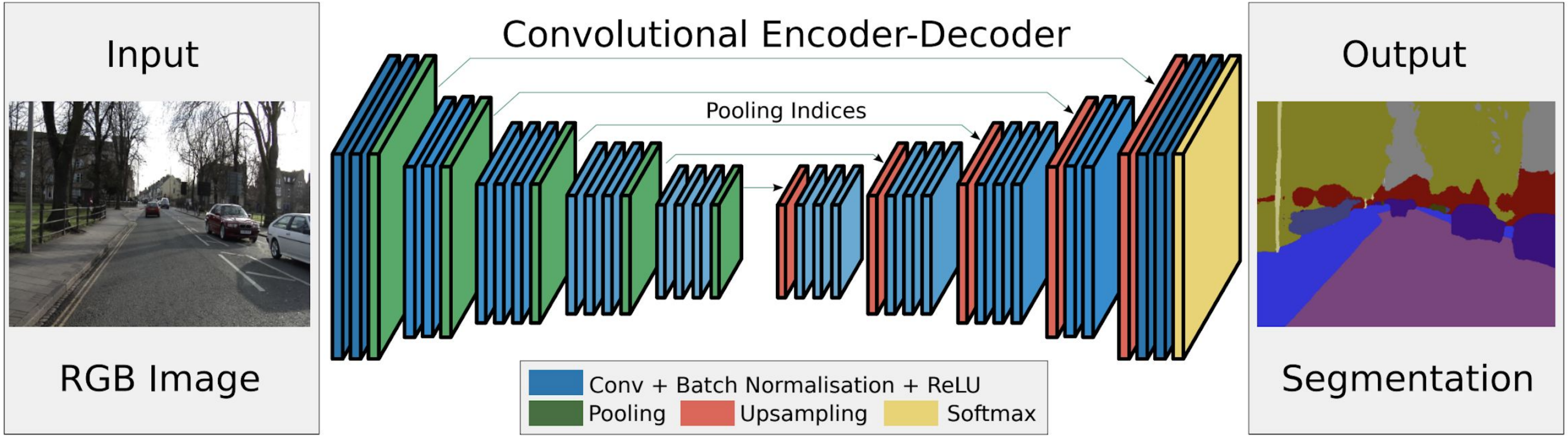
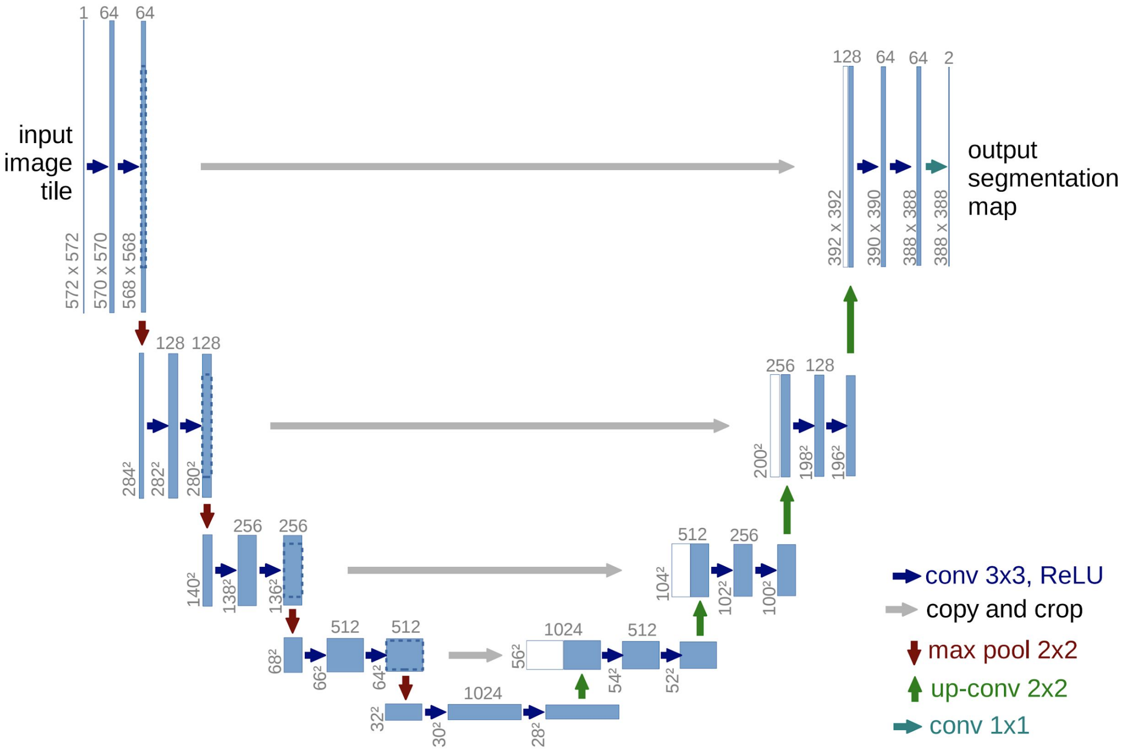


Fig. 2. An illustration of the SegNet architecture. There are no fully connected layers and hence it is only convolutional. A decoder upsamples its input using the transferred pool indices from its encoder to produce a sparse feature map(s). It then performs convolution with a trainable filter bank to densify the feature map. The final decoder output feature maps are fed to a soft-max classifier for pixel-wise classification.

U-Net



1 64 64
input
image
tile
572 x 572
570 x 570
568 x 568
128 128
284²
282²
280²
256 256
140²
138²
136²
512 512
68²
66²
64²
1024
32²
30²
28²
1024 512
56²
54²
52²
512 256
104²
102²
100²
256 128
200²
198²
196²
128 64 64 2
392 x 392
390 x 390
388 x 388
388 x 388
output
segmentation
map
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

U-Net

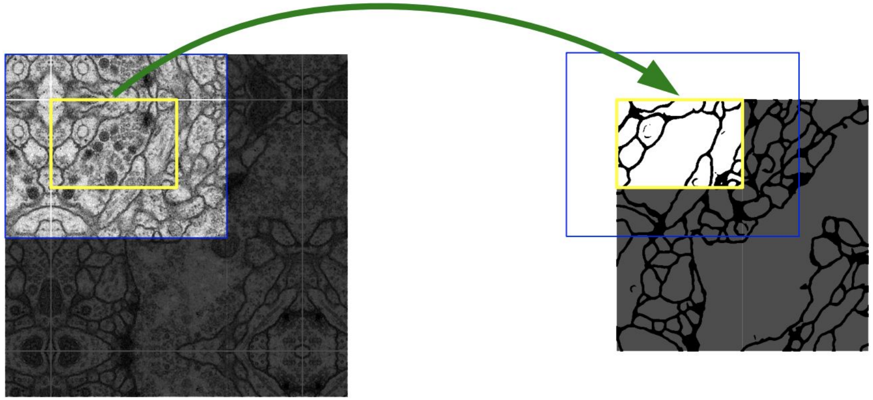
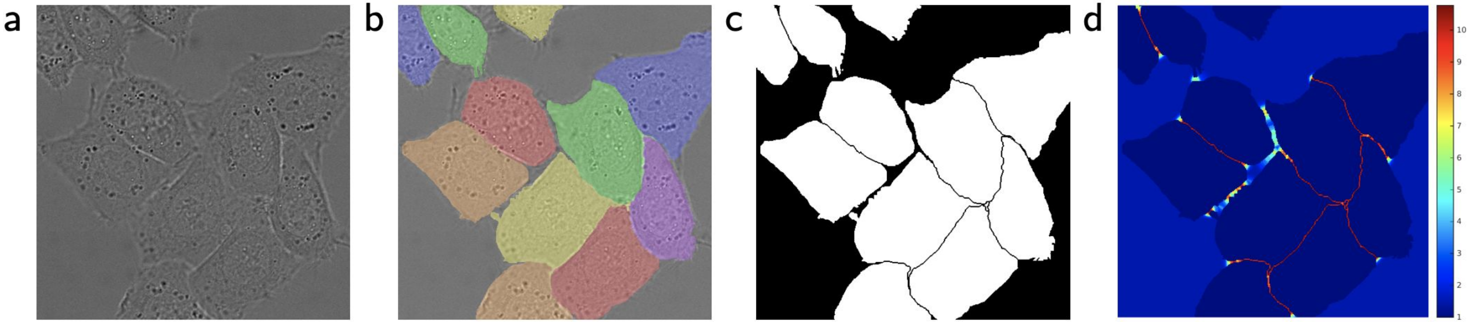
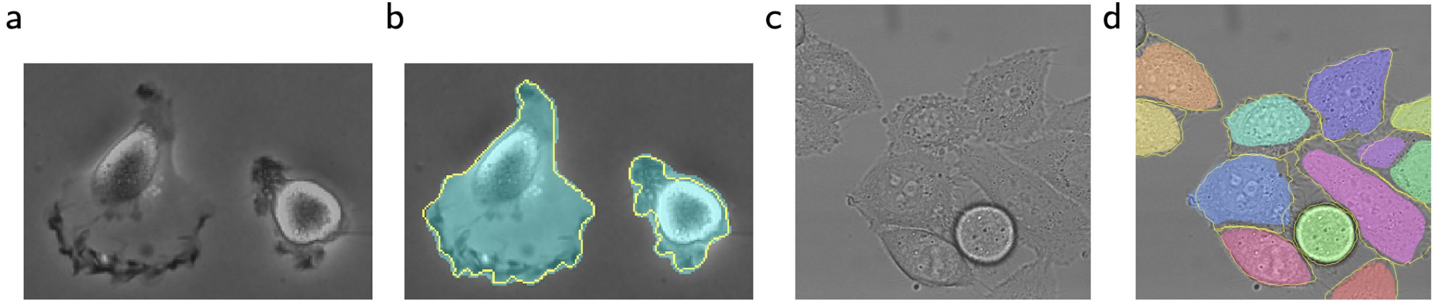


Fig. 2. Overlap-tile strategy for seamless segmentation of arbitrary large images (here segmentation of neuronal structures in EM stacks). Prediction of the segmentation in the yellow area, requires image data within the blue area as input. Missing input data is extrapolated by mirroring

U-Net





Instance Segmentation

Input image (C x H x W)

Output

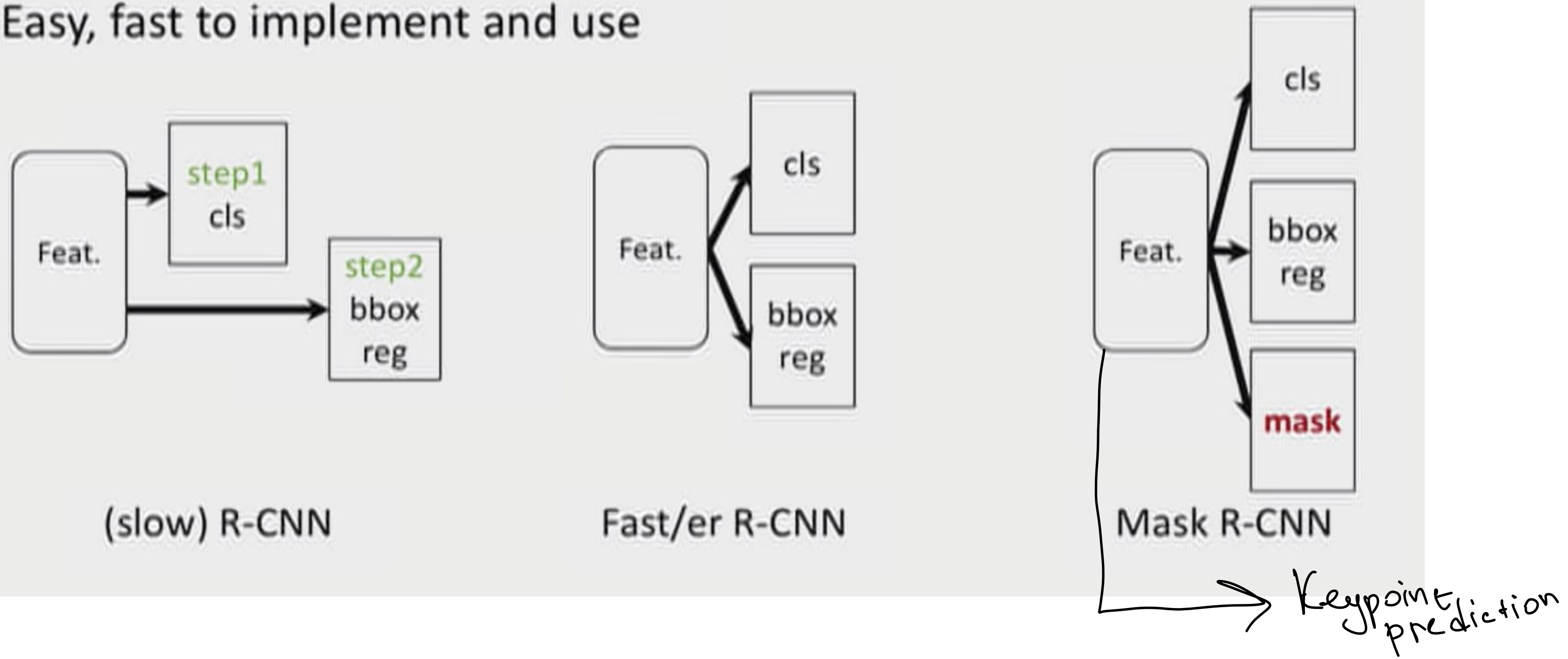
mask of class instances (detection + 1 x H x W)

Assumption

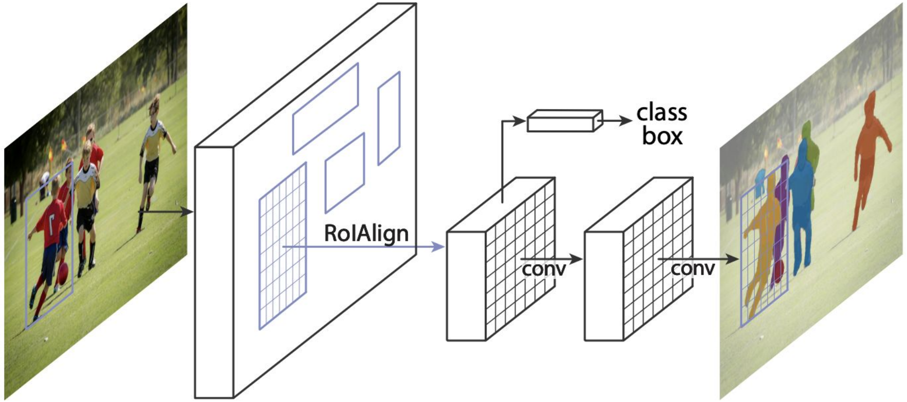
separated masks for each distinct object

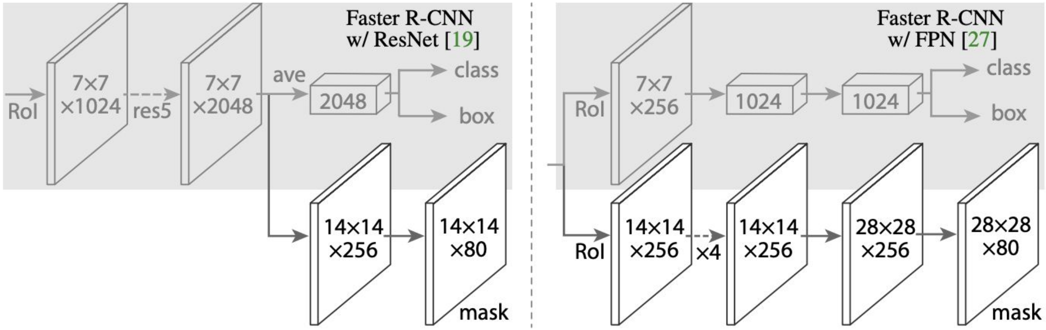
Mask R-CNN: Instance segmentation

- Easy, fast to implement and use

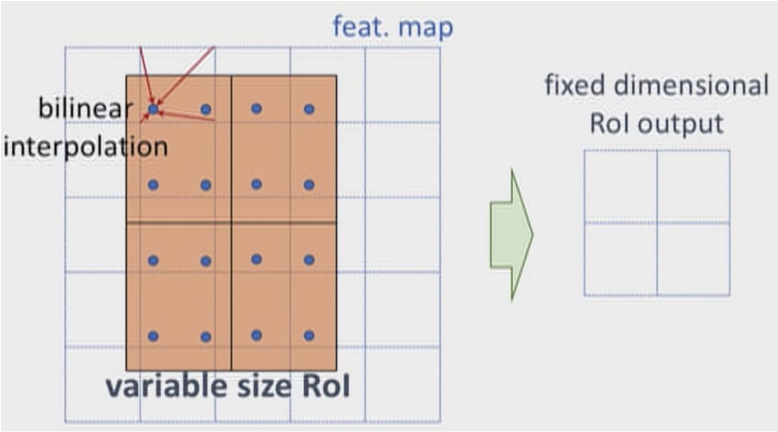


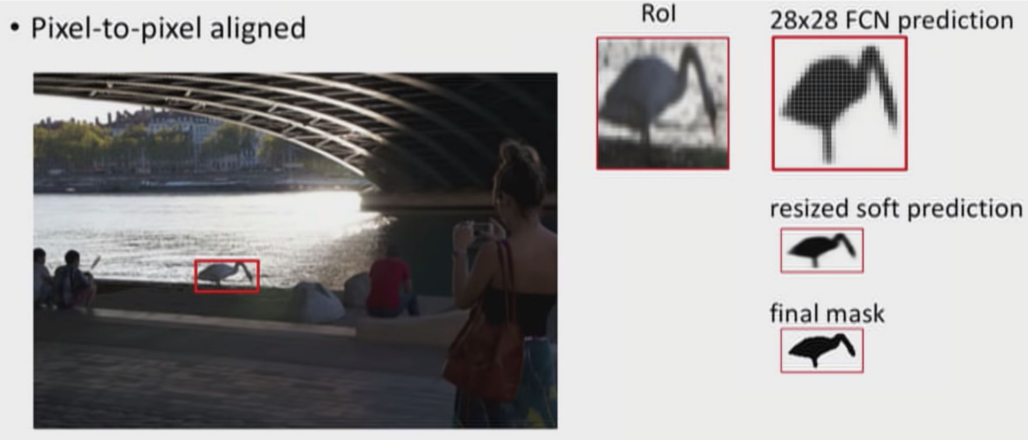
Mask R-CNN


class
box
RoIAlign
conv
conv


Faster R-CNN
w/ ResNet [19]
RoI
7×7
×1024
res5
7×7
×2048
ave
2048
class
box
14×14
×256
14×14
×80
mask
Faster R-CNN
w/ FPN [27]
RoI
7×7
×256
1024
1024
class
box
RoI
14×14
×256
×4
14×14
×256
28×28
×256
28×28
×80
mask

Mask R-CNN: RoI Align



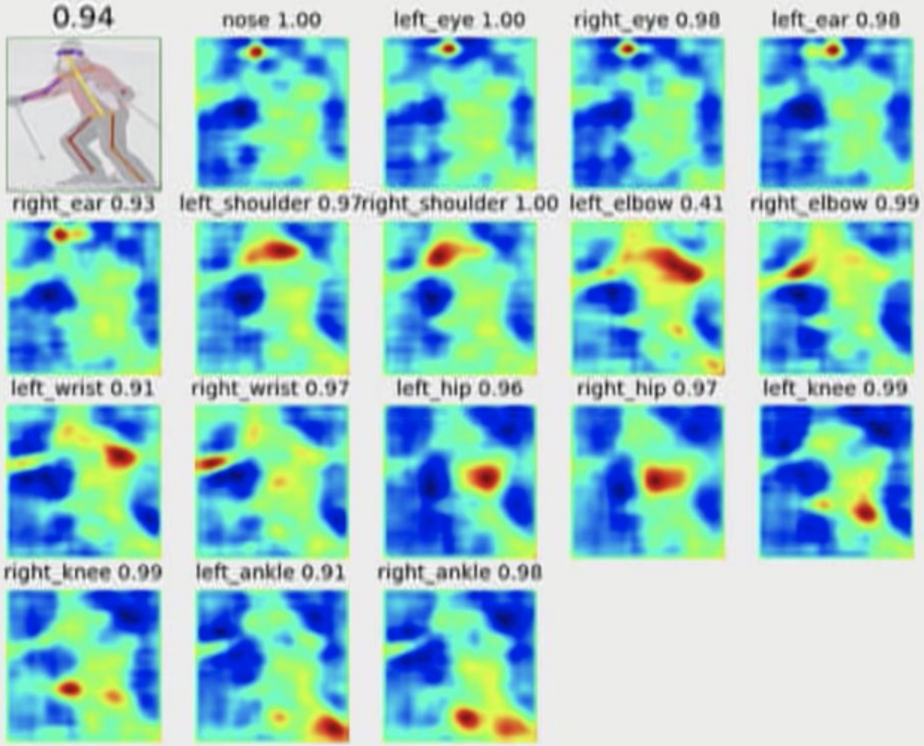


Mask R-CNN: Results

Bonus: Pose estimation

- keypoint = 1-hot mask
- human pose = 17 masks

- One framework for
  - ✓bbox
  - ✓mask
  - ✓keypoint



Bonus: Pose estimation

Revise

- Segmentation task
- Simple solutions
- Upsampling methods
  - Unpooling
  - Transposed convolution
- FCN
- DeconvNet
- SegNet
- U-Net
- Mask R-CNN

Next time

- Variational AutoEncoders
- GANs

Interactive playground for image generation