Image Segmentation

Computer Vision Fall 2022
Lecture 18 & 19


APPAREL MART BUILDING
LIGHT UP BROADWAY
IGNITE
DEPART. ELEVATE. ARRIVE.
ROCKET DOG
PARKING
Metro
Metro Local
Count your chickens
Local

torchvision

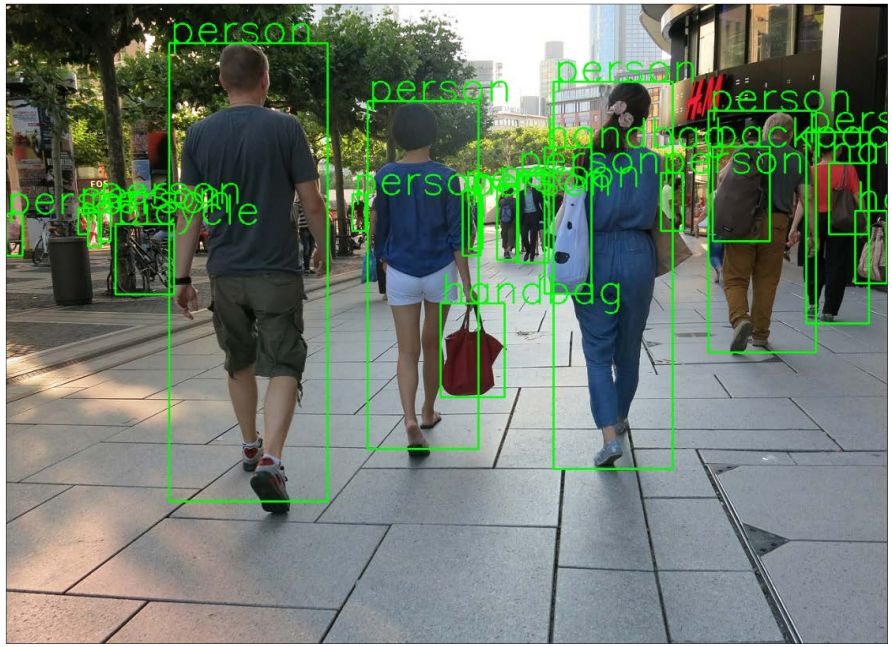
```python
# load model
model = torchvision.models.detection.fasterrcnn_resnet50_fpn(pretrained=True)
# set to evaluation mode
model.eval()
```

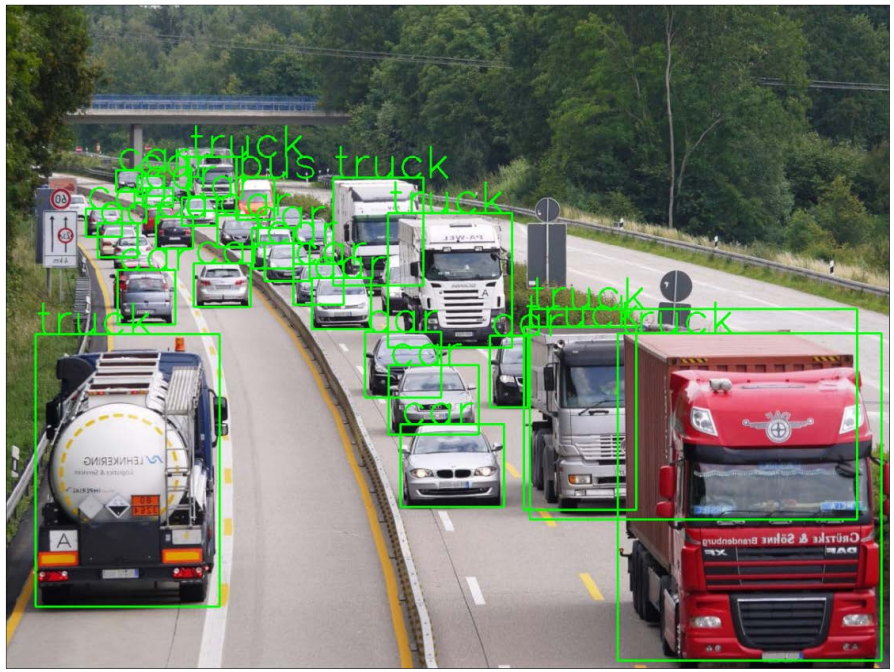
```python
import torch
import torchvision.transforms as T
import torchvision
dir(torchvision.models.detection)
```

✓ 0.1s

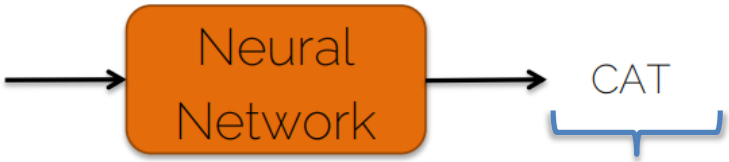
Output exceeds the size limit. Open the full output

```
['FCOS',
 'FCOS_ResNet50_FPN_Weights',
 'FasterRCNN',
 'FasterRCNN_MobileNet_V3_Large_320_FPN_Weights',
 'FasterRCNN_MobileNet_V3_Large_FPN_Weights',
 'FasterRCNN_ResNet50_FPN_V2_Weights',
 'FasterRCNN_ResNet50_FPN_Weights',
 'KeypointRCNN',
 'KeypointRCNN_ResNet50_FPN_Weights',
 'MaskRCNN',
 'MaskRCNN_ResNet50_FPN_V2_Weights',
 'MaskRCNN_ResNet50_FPN_Weights',
 'RetinaNet',
 'RetinaNet_ResNet50_FPN_V2_Weights',
 'RetinaNet_ResNet50_FPN_Weights',
 'SSD300_VGG16_Weights',
 'SSDLite320_MobileNet_V3_Large_Weights',
```

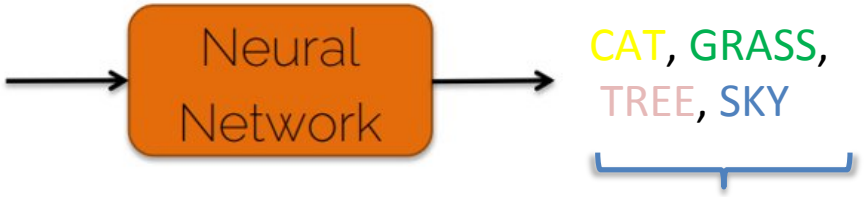




Task definition: semantic segmentation

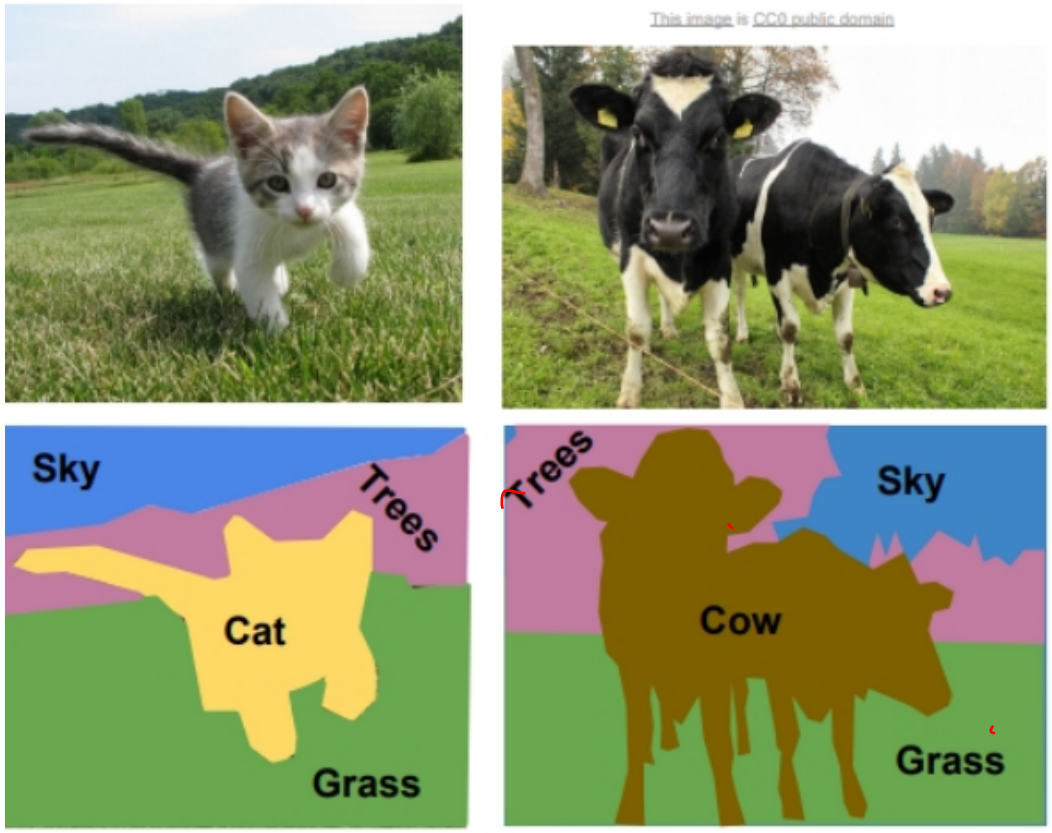


Classify the main object in the image.



No objects, just classify each pixel.

Semantic Segmentation



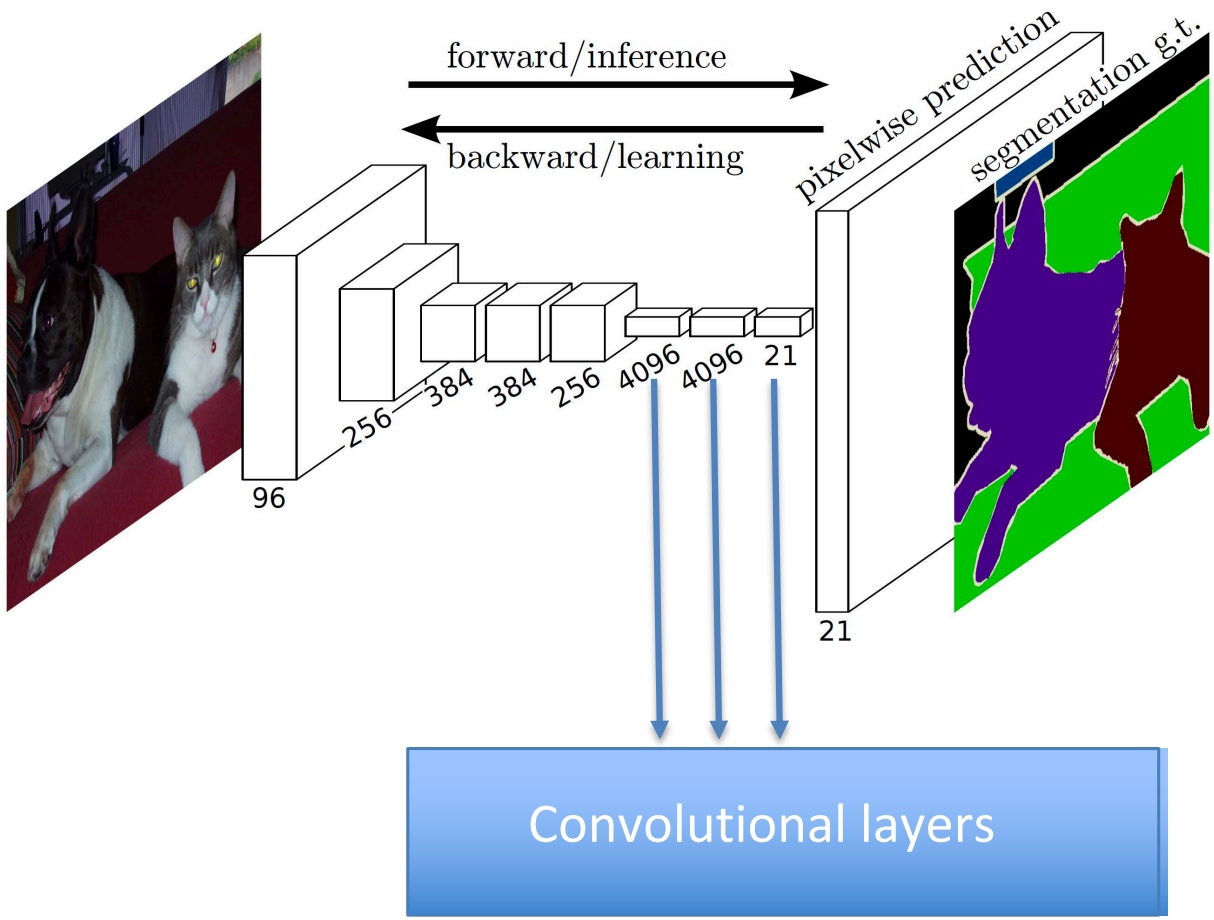
- Every pixel in the image needs to be labelled with a category label.

- Do not differentiate between the instances (see how we do not differentiate between pixels coming from different cows).

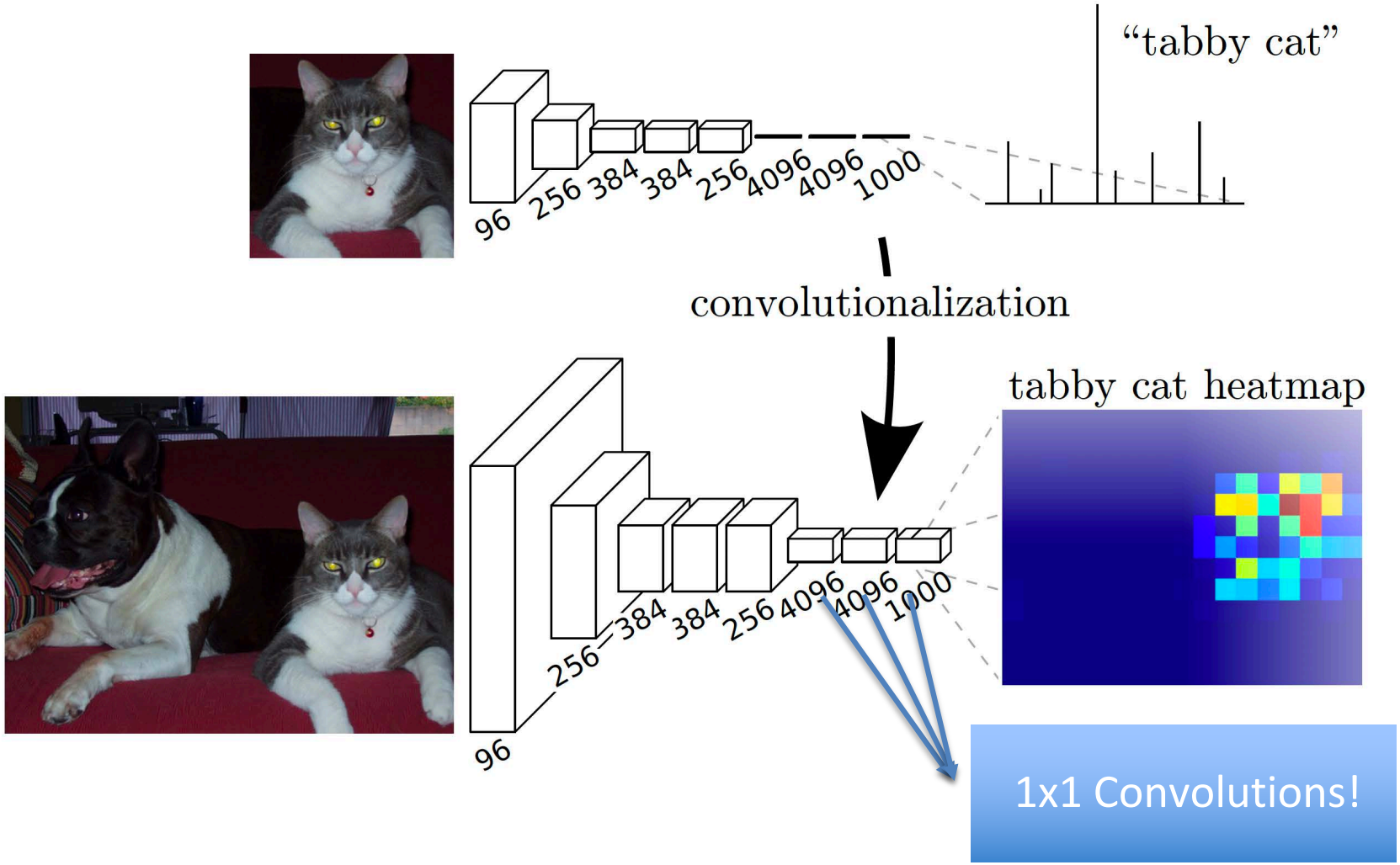
Fully Convolutional Networks

Fully convolutional neural networks

1. Replace FC layers with convolutional layers.
2. Convert the last layer output to the original resolution.
3. Do softmax-cross entropy between the pixelwise predictions and segmentaion ground truth.
4. Backprop and SGD

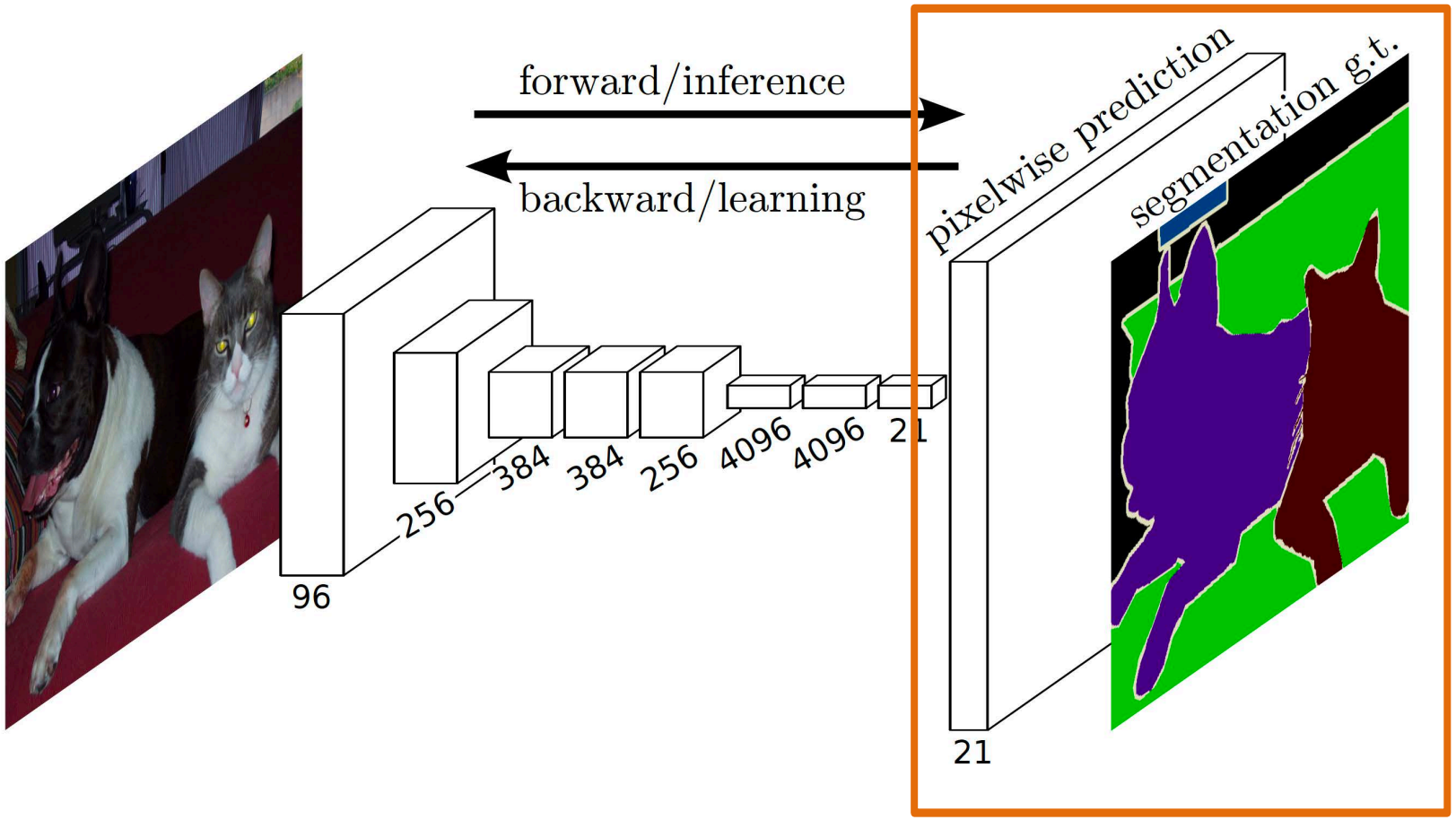


“Convolutionalization”



Semantic Segmentation (FCN)

- Fully Convolutional Networks for Semantic Segmentation



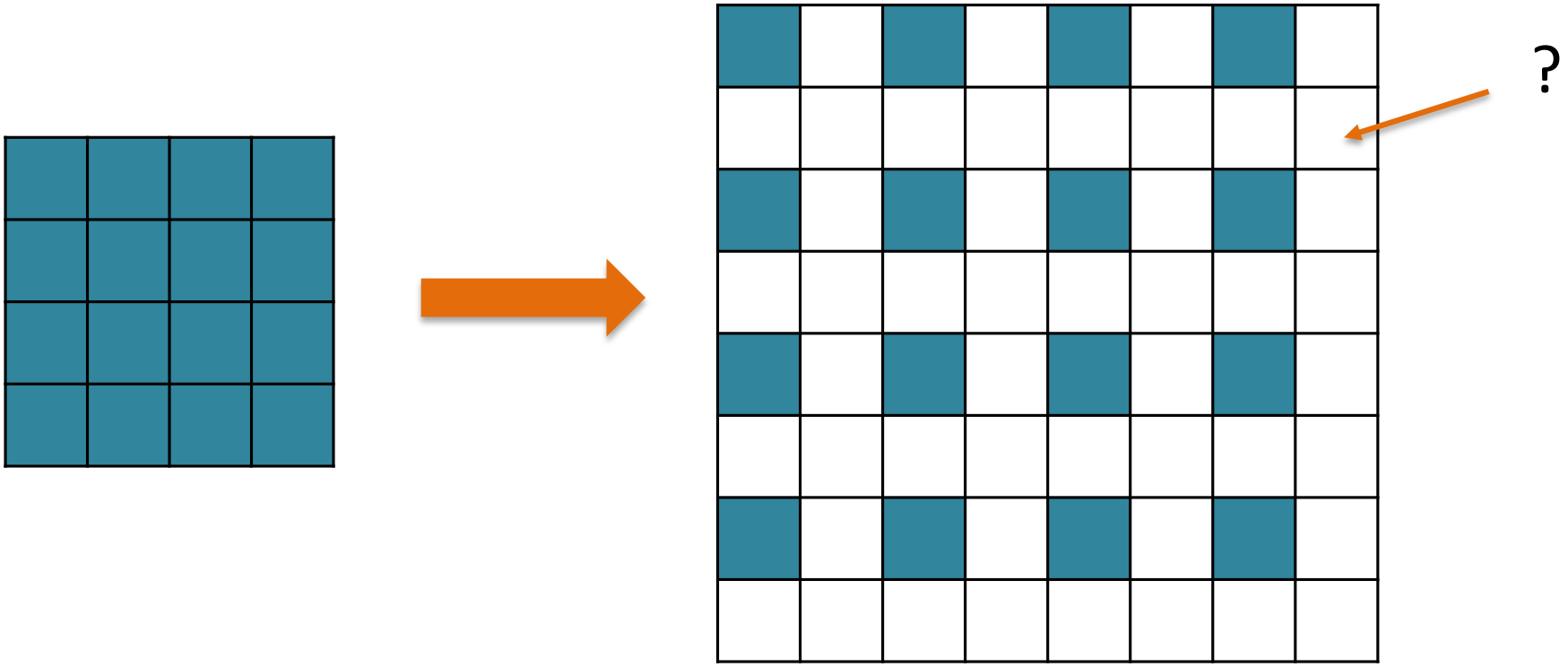
How do we upsample?

Long, Shelhamer, Darrell - Fully Convolutional Networks for Semantic Segmentation, CVPR 2015, PAMI 2016

Upsampling

Types of upsamplings

- 1. Interpolation



Types of upsamplings

- 1. Interpolation

Original image  x 10

Nearest neighbor interpolation

Bilinear interpolation

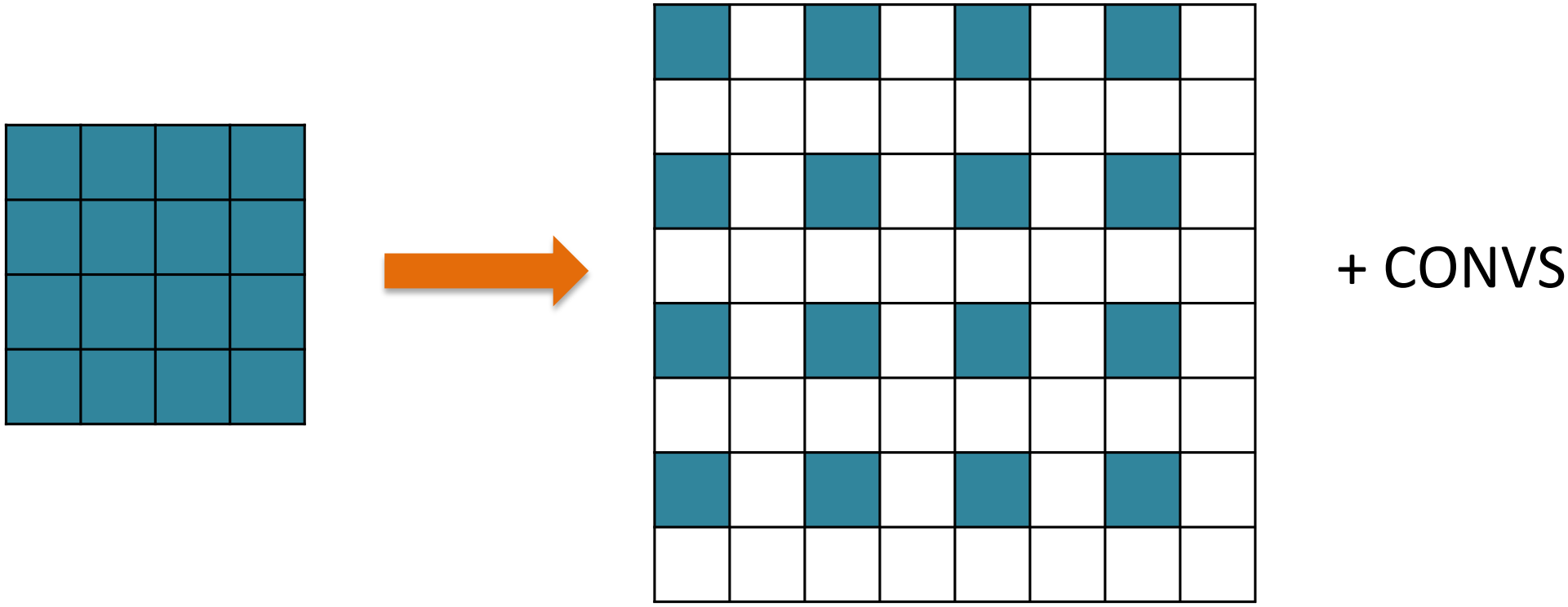
Bicubic interpolation

Image: Michael Guerzhoy

Types of upsamplings

- 2. Fixed unpooling

efficient



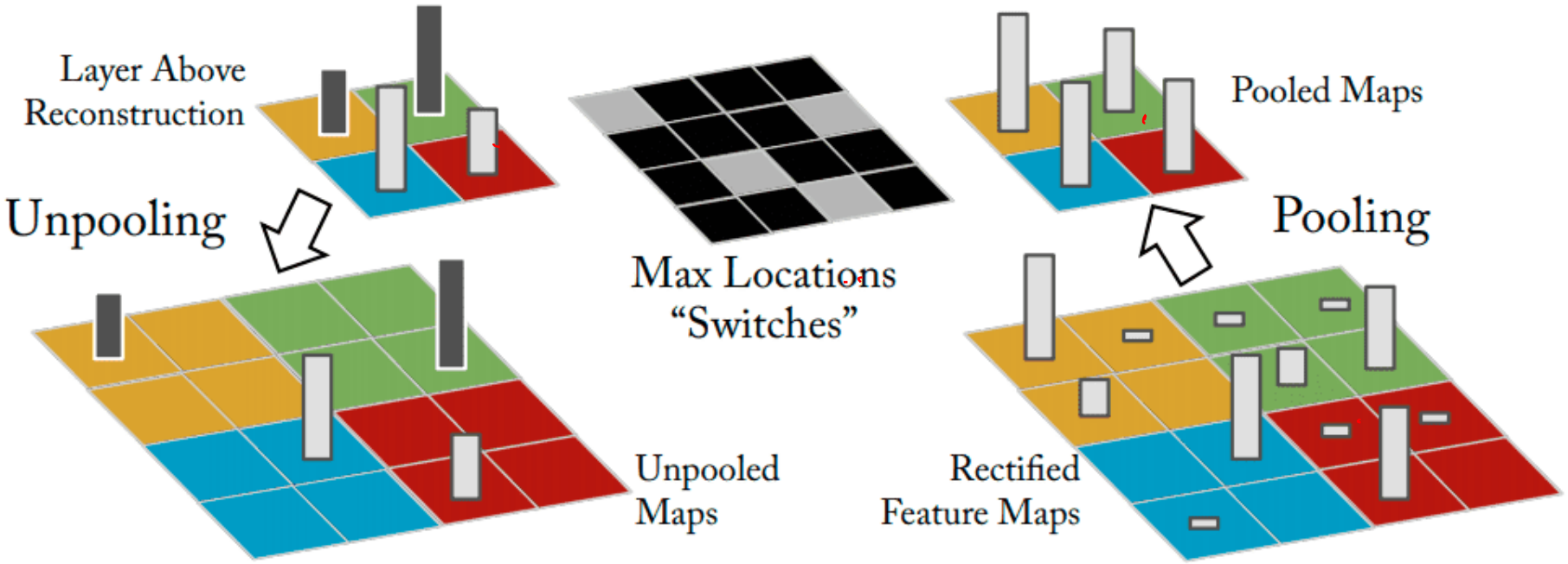
+ CONVS

A. Dosovitskiy, “Learning to Generate Chairs, Tables and Cars with Convolutional Networks“. TPAMI 2017

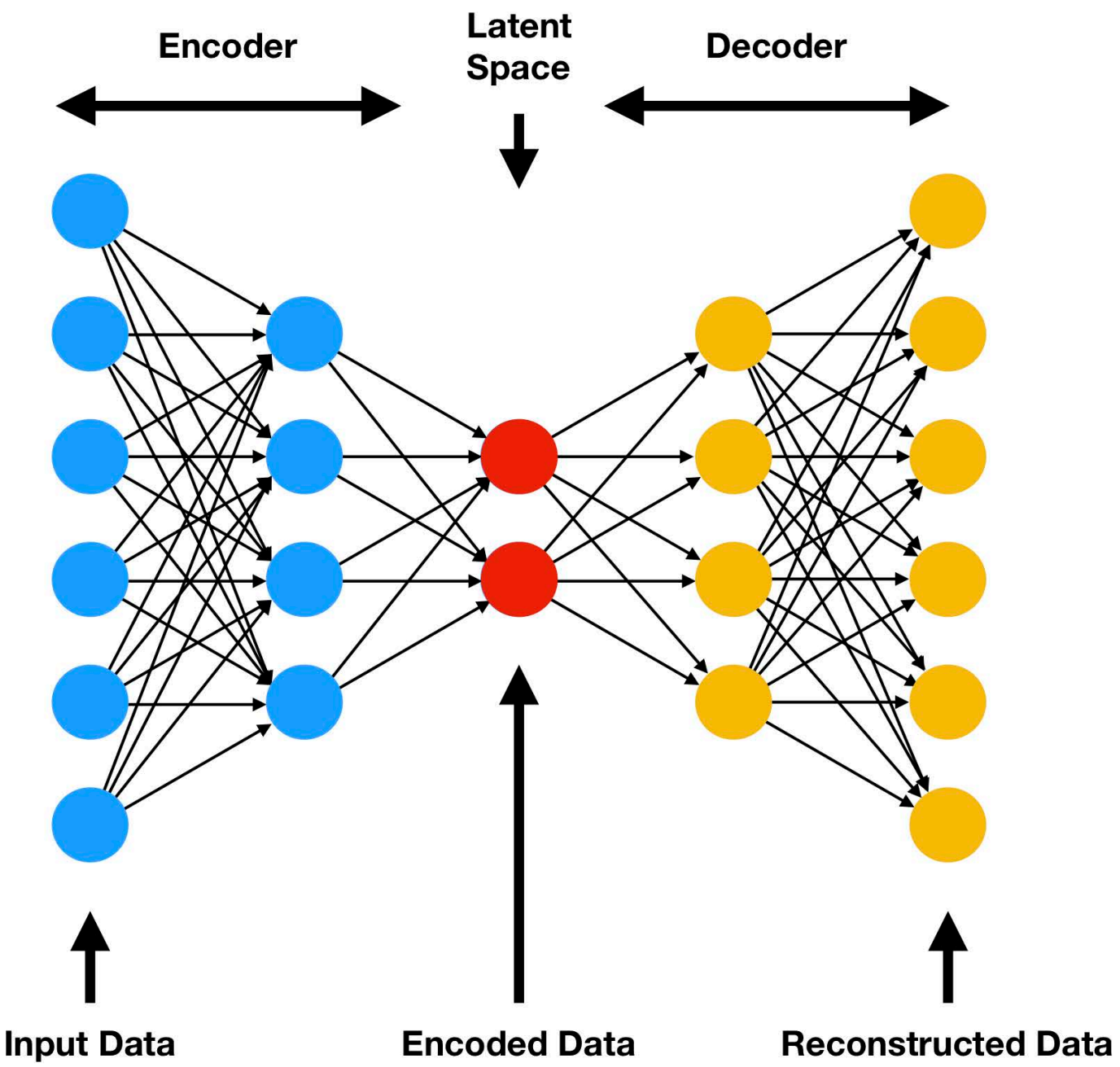
Types of upsamplings

- 3. Unpooling: “DeconvNet”

  Keep the locations where the max came from

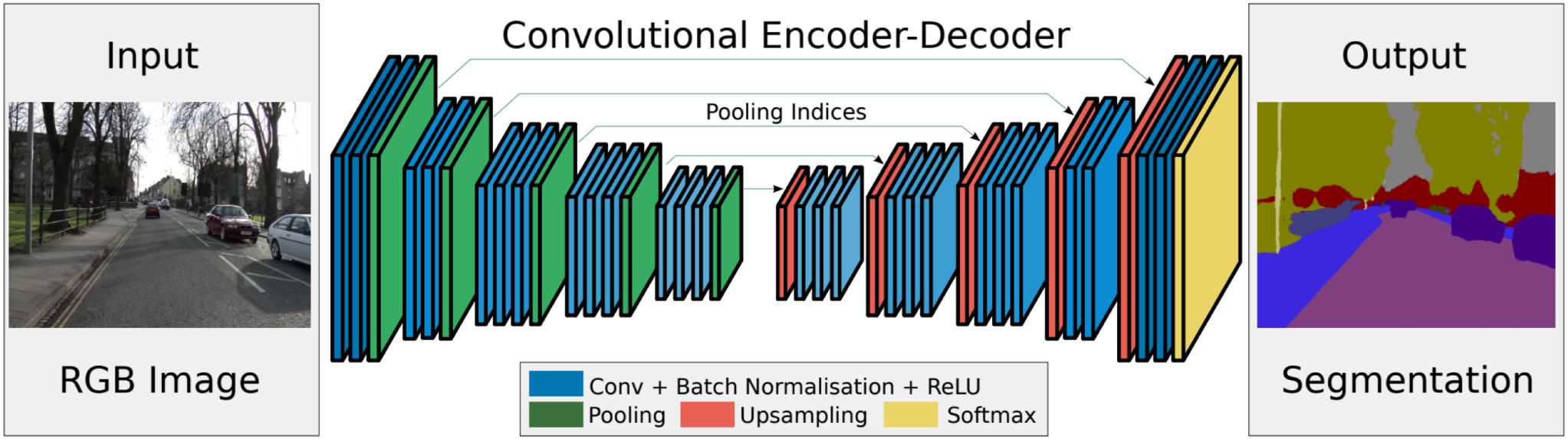


Autoencoder-style architecture


Encoder
Latent Space
Decoder
Input Data
Encoded Data
Reconstructed Data

SegNet

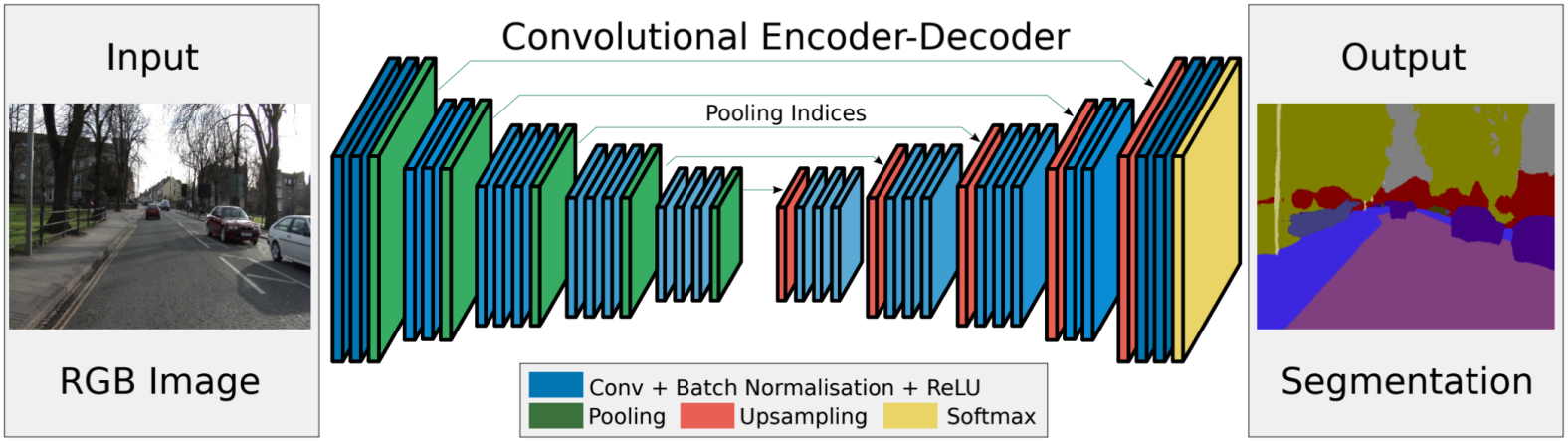
- Step-wise upsampling



Badrinarayanan et al. “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation“. TPAMI 2016

SegNet

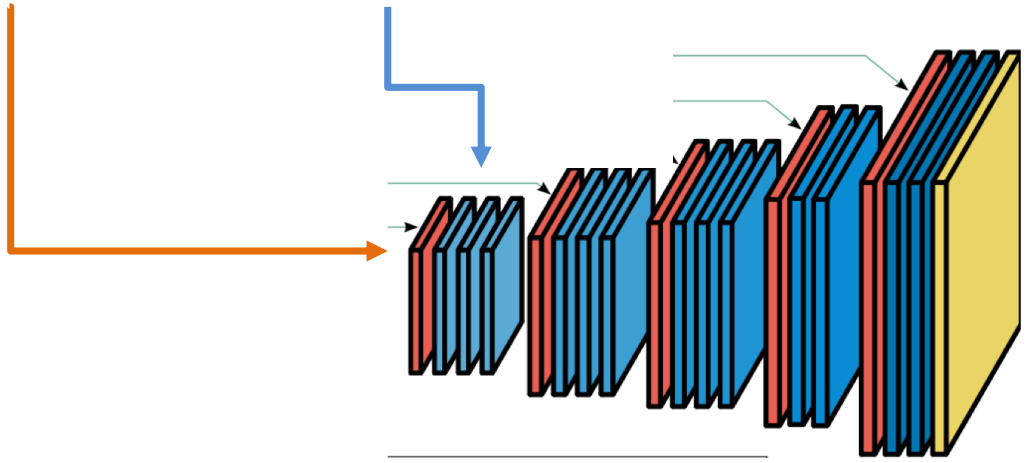
- **Encoder**: normal convolutional filters + pooling
- **Decoder**: Upsampling + convolutional filters



Badrinarayanan et al. “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation“. TPAMI 2016

SegNet

- **Encoder**: normal convolutional filters + pooling
- **Decoder**: Upsampling + convolutional filters



Badrinarayanan et al. ”SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation“. TPAMI 2016

SegNet

- **Encoder**: normal convolutional filters + pooling
- **Decoder**: Upsampling + convolutional filters
- The convolutional filters in the decoder are learned using backprop and their goal is to refine the upsampling

Badrinarayanan et al. “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation“. TPAMI 2016

SegNet

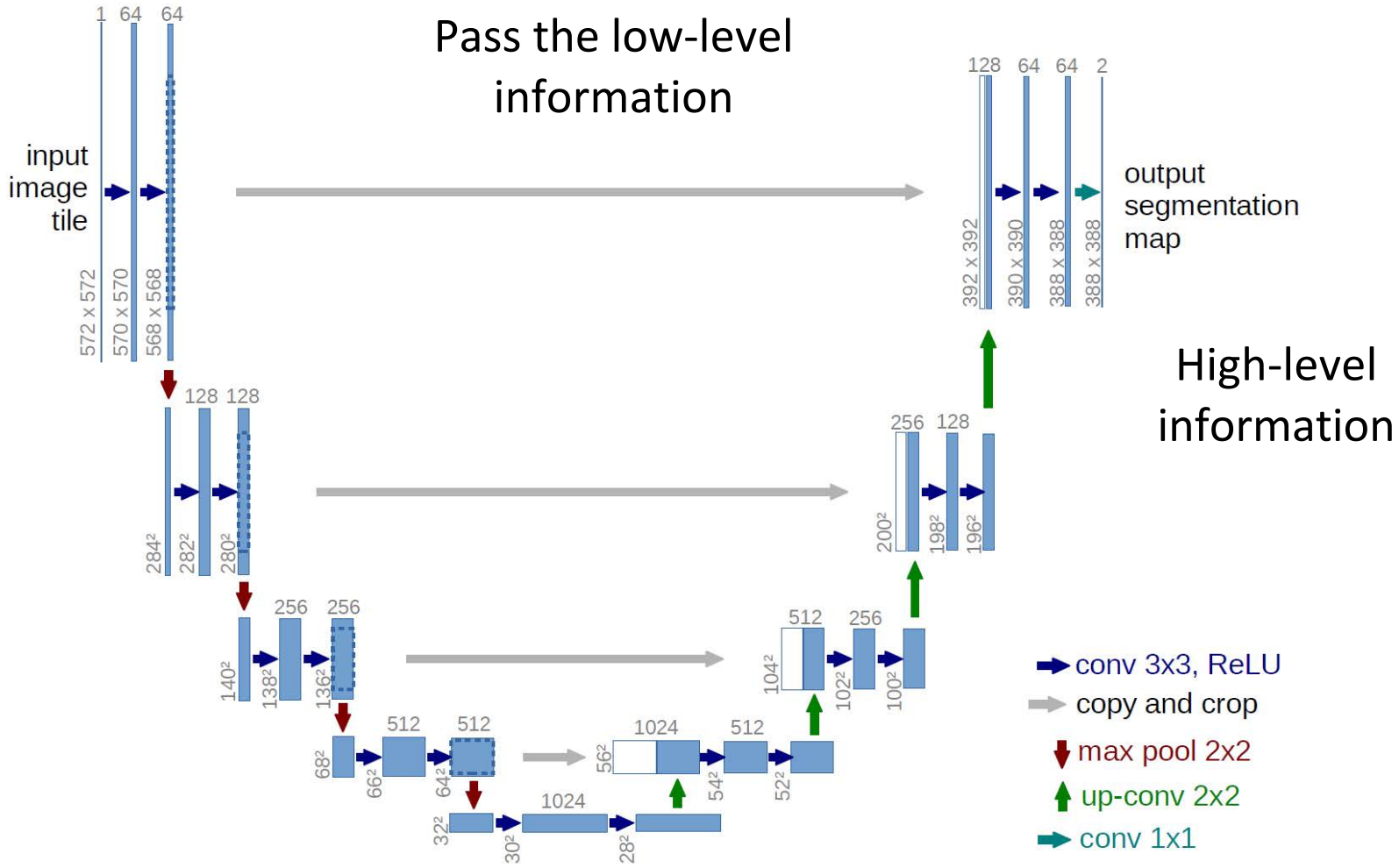
- **Encoder**: normal convolutional filters + pooling
- **Decoder**: Upsampling + convolutional filters
- **Softmax** layer: The output of the soft-max classifier is a K channel image of probabilities where K is the number of classes.

Badrinarayanan et al. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation“. TPAMI 2016

Skip connections (U-Net)

Skip Connections

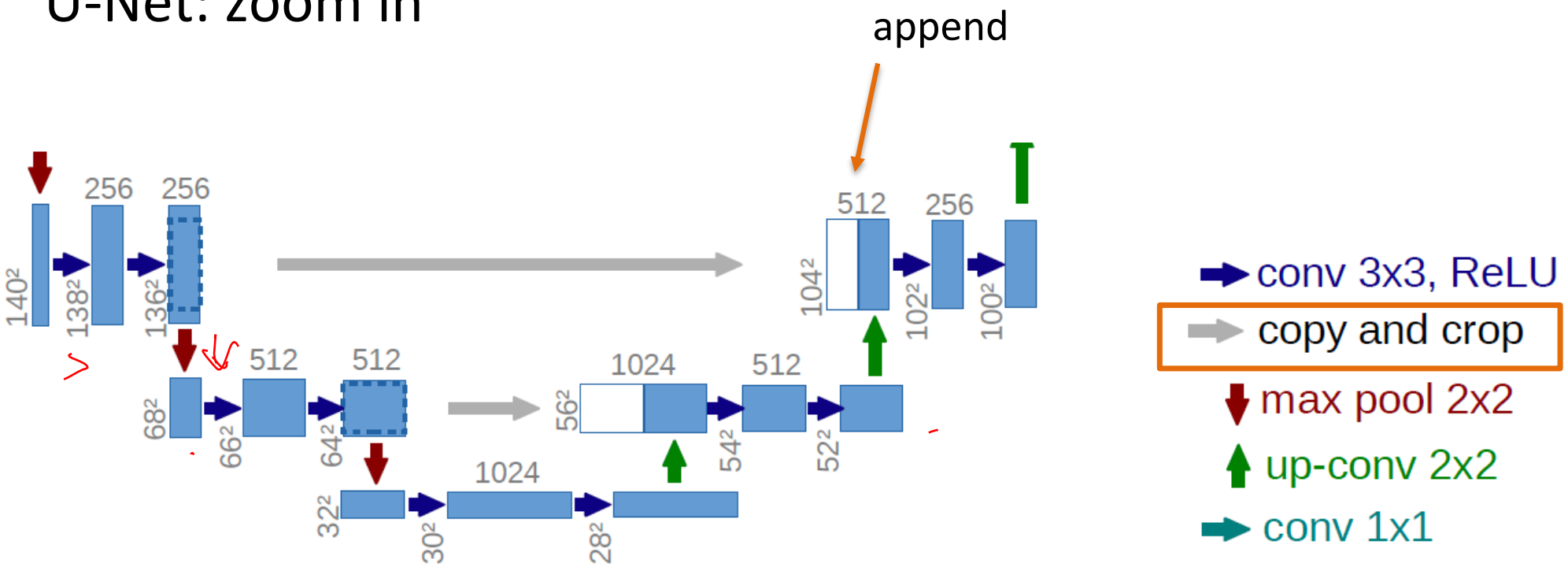
- U-Net



O. Ronneberger et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. MICCAI 2015

Skip Connections

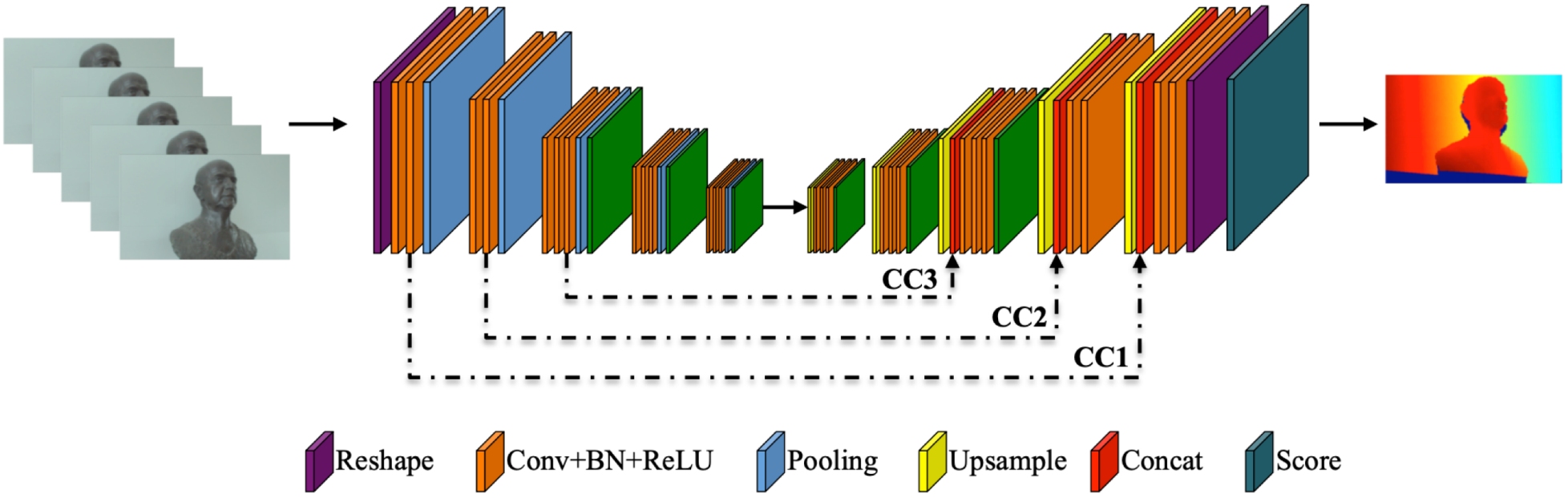
- U-Net: zoom in



O. Ronneberger et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. MICCAI 2015

Skip Connections

- Concatenation connections



C. Hazirbas et al. “deep depth from focus”. ACCV 2018

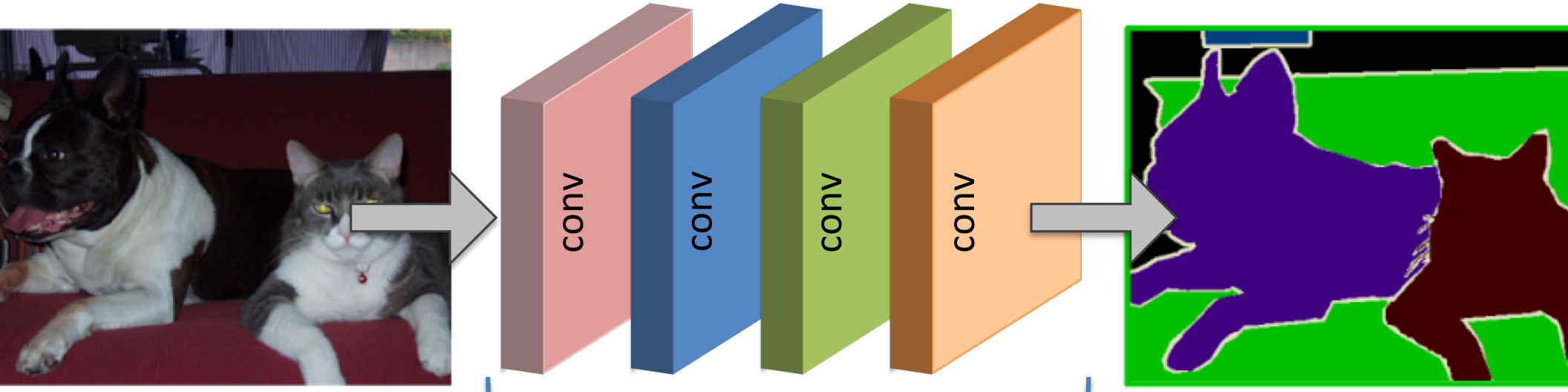
Semantic Segmentation: 3 challenges

- Reduced feature resolution
  - Proposed solution: Atrous convolutions
- Objects exist at multiple scales
  - Proposed solution: Pyramid pooling, as in detection.
- Poor localization of the edges
  - Proposed solution: Refinement with Conditional Random Field (CRF)

Semantic Segmentation: 3 challenges

- Reduced feature resolution
  - Proposed solution: Atrous convolutions
- Objects exist at multiple scales
  - Proposed solution: Pyramid pooling, as in detection.
- Poor localization of the edges
  - Proposed solution: Refinement with Conditional Random Field (CRF)

Wish: no reduced feature resolution



pixels in
width x height x RGB

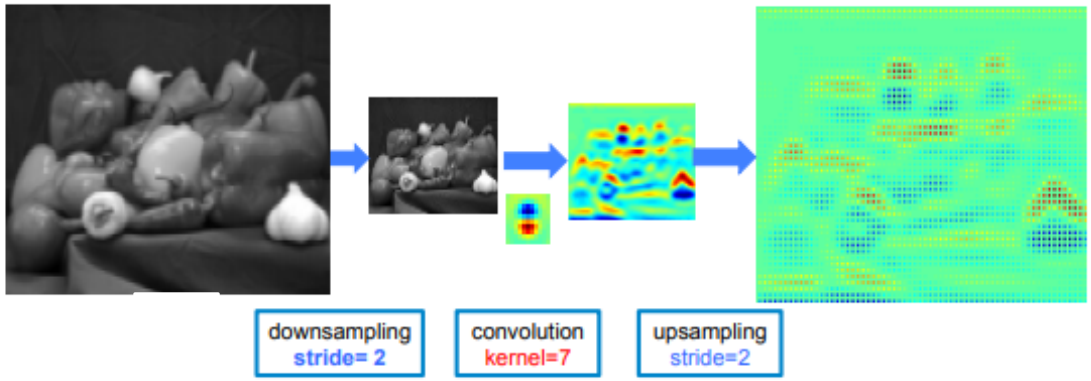
Just convs & activations

pixels out
width x height x classes

Fully Convolutional Network

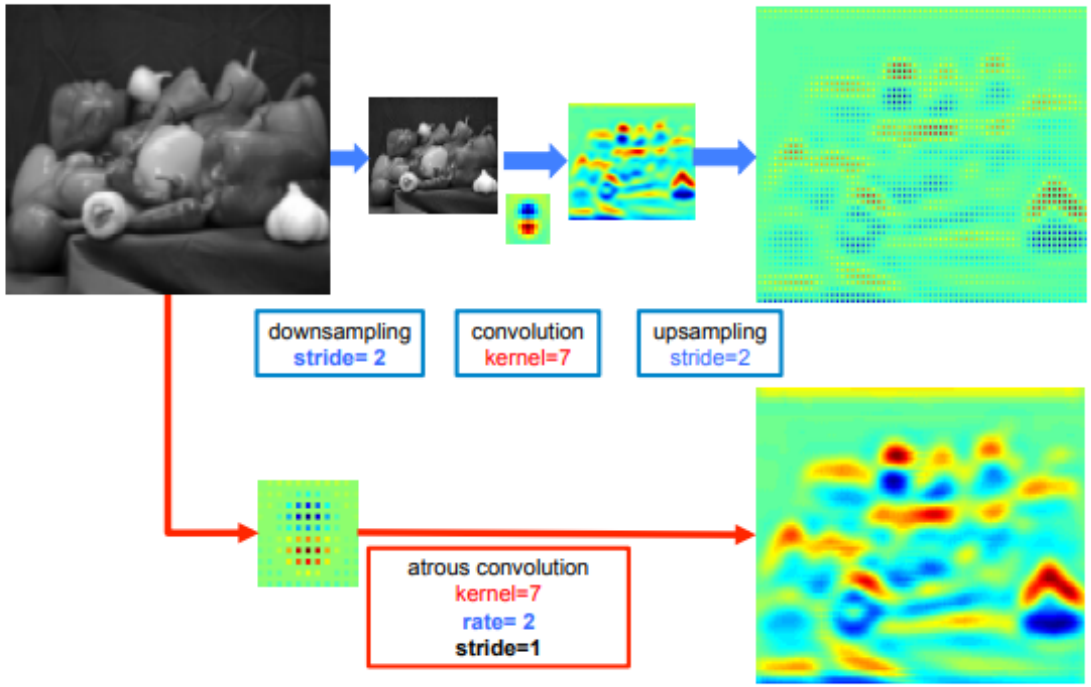
Super expensive!

Alternative: Dilated (atrous) convolutions



Sparse feature extraction with standard convolution on a low resolution input feature map.

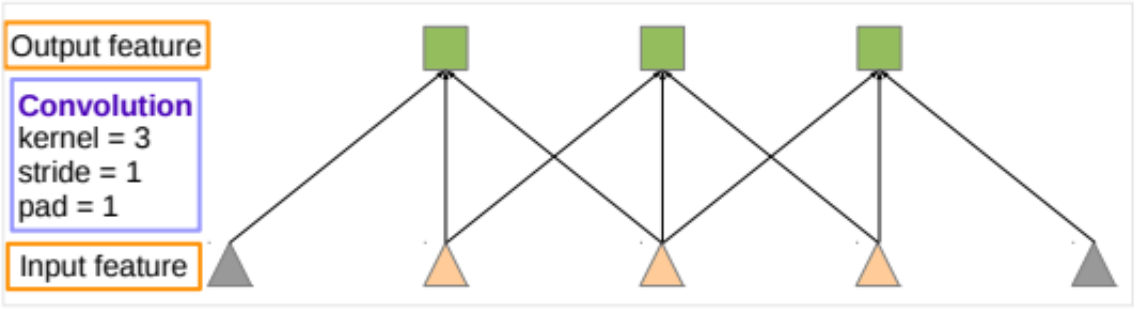
Alternative: Dilated (atrous) convolutions



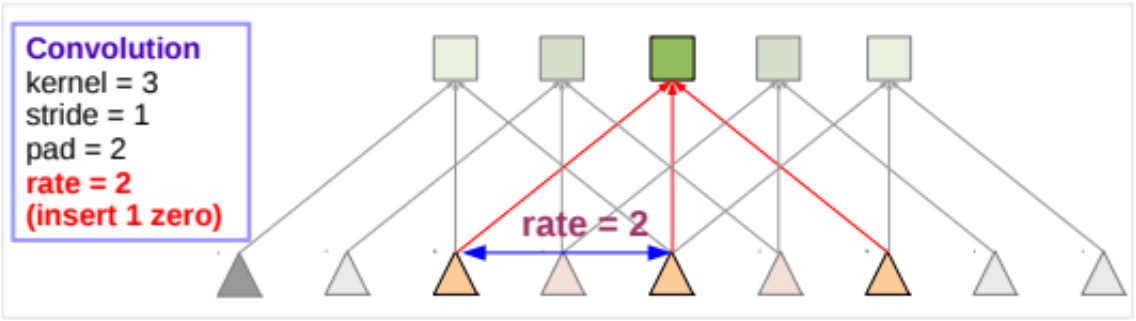
Sparse feature extraction with standard convolution on a low resolution input feature map.

Dense feature extraction with atrous convolution with rate r=2, applied on a high resolution input feature map.

Dilated (atrous) convolutions 1D



(a) Sparse feature extraction

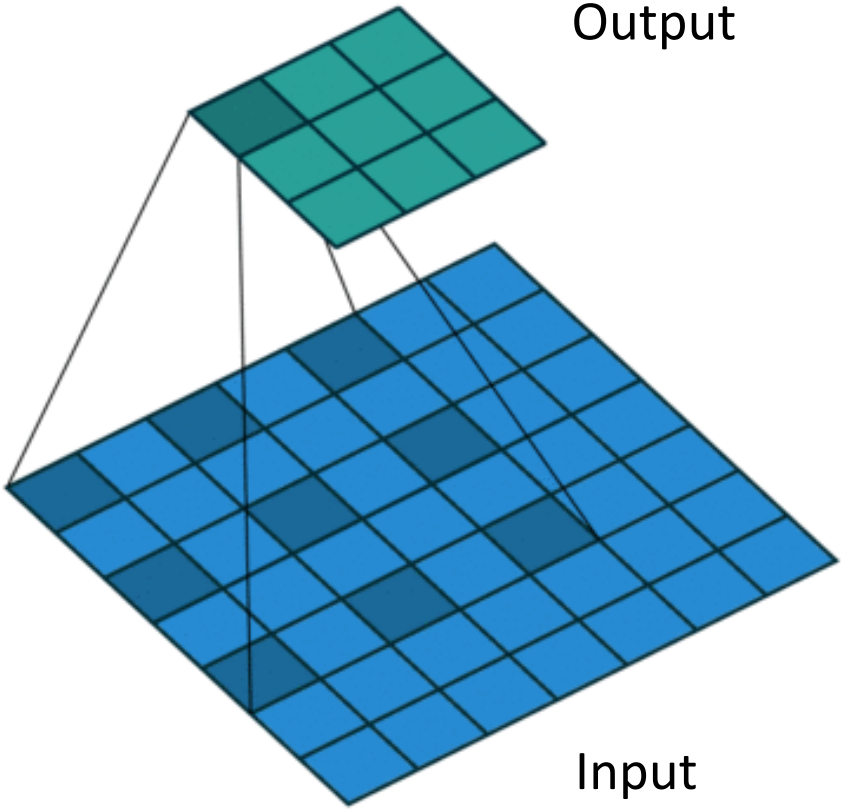


(b) Dense feature extraction

(a) Sparse feature extraction with standard convolution on a low resolution input feature map.

(b) Dense feature extraction with atrous convolution with rate r = 2, applied on a high resolution input feature map.

Dilated (atrous) convolutions in 2D



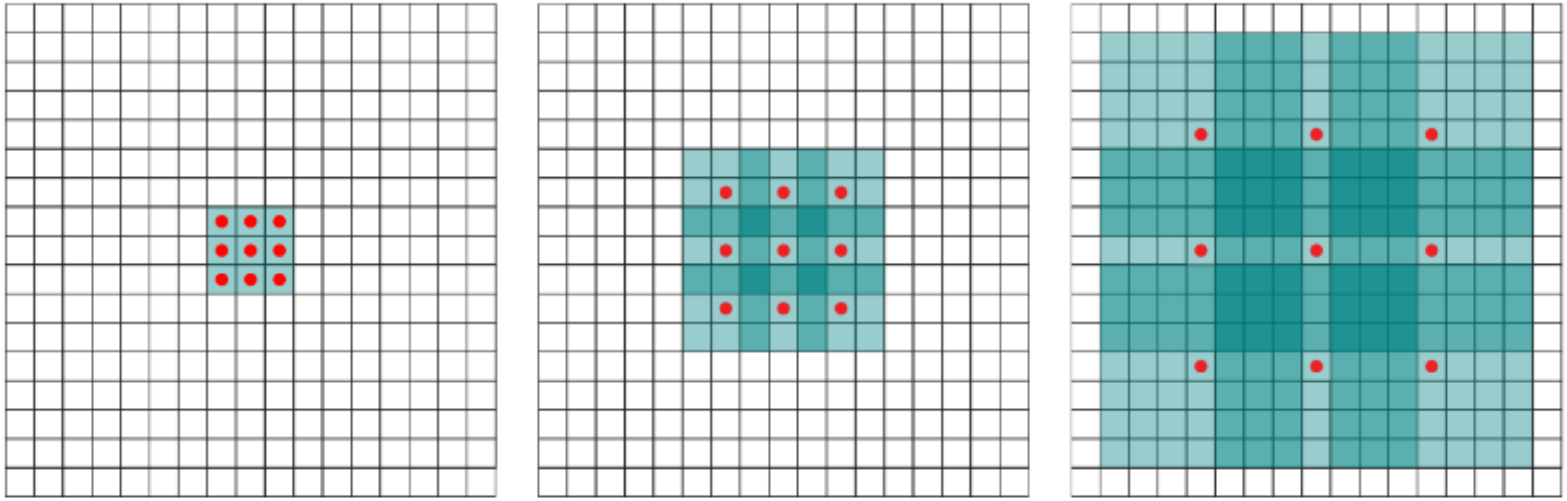
Standard convolution has dilation 1

An analogy for dilated conv is a conv filter with holes

class **torch.nn.Conv2d** (***in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=2***)

class **torch.nn.ConvTranspose2d** (***in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=2***)

Dilated (atrous) convolutions 2D



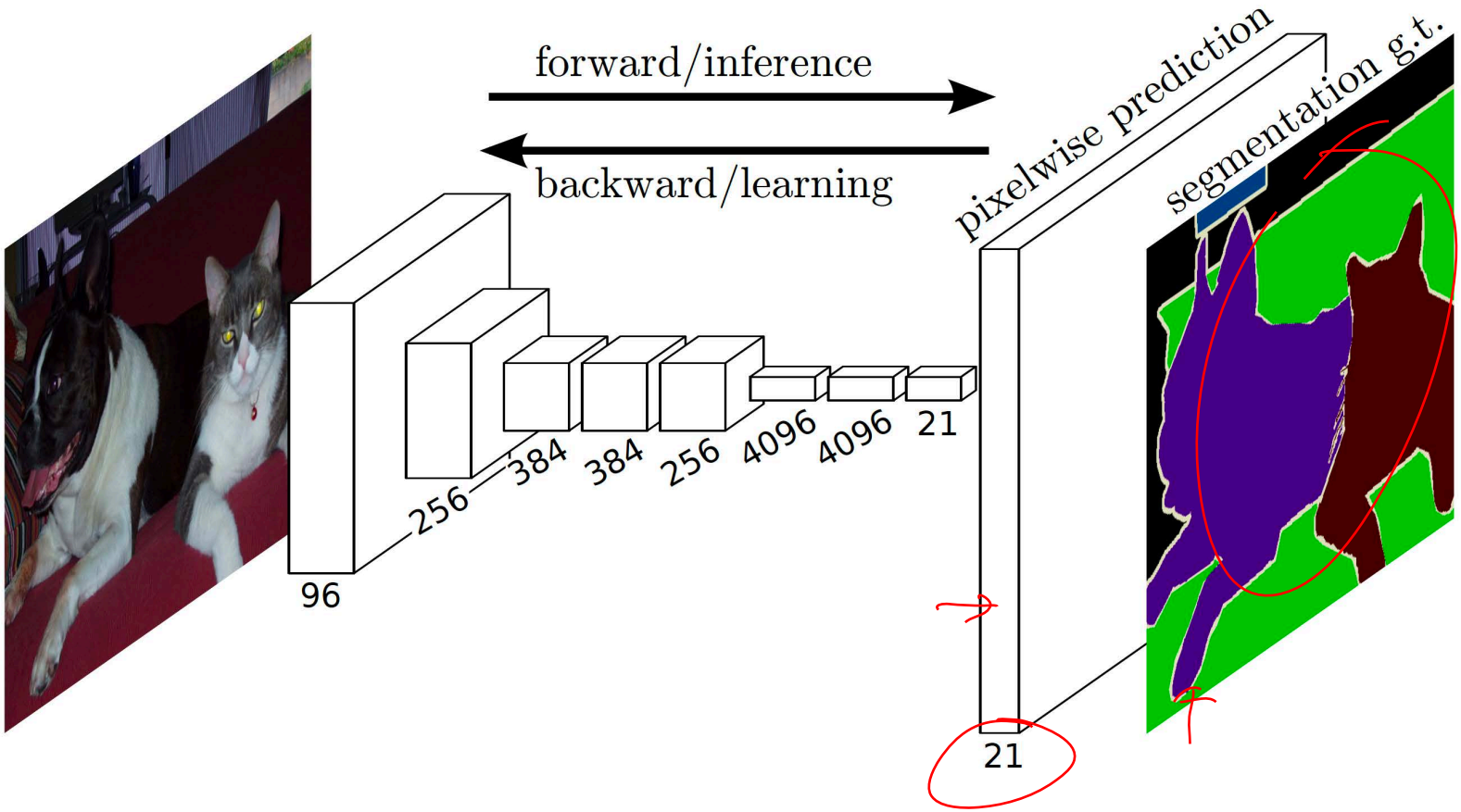
Each layer has the same number of parameters, but the receptive field grows exponentially while the number of parameters grows linearly.

Image Segmentation

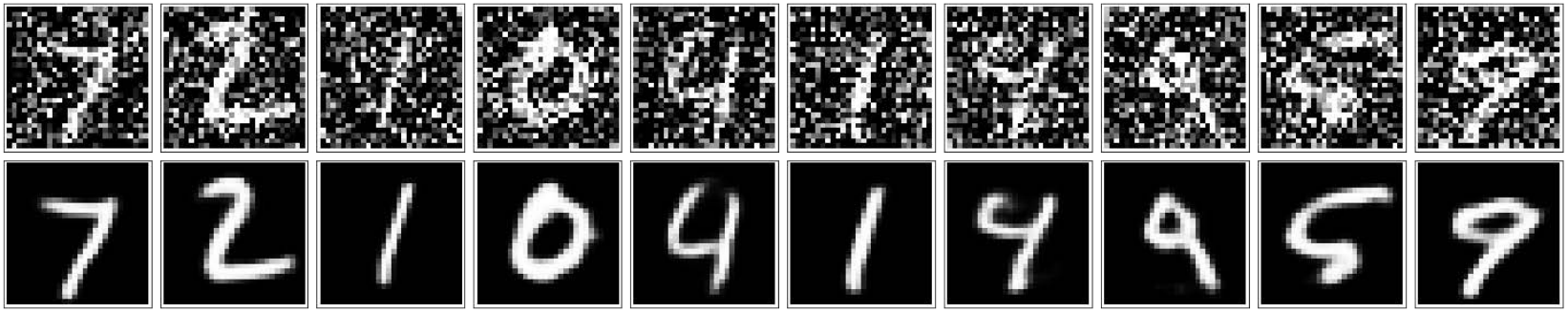
Computer Vision Fall 2022
Lecture 18 & 19

Fully convolutional neural networks

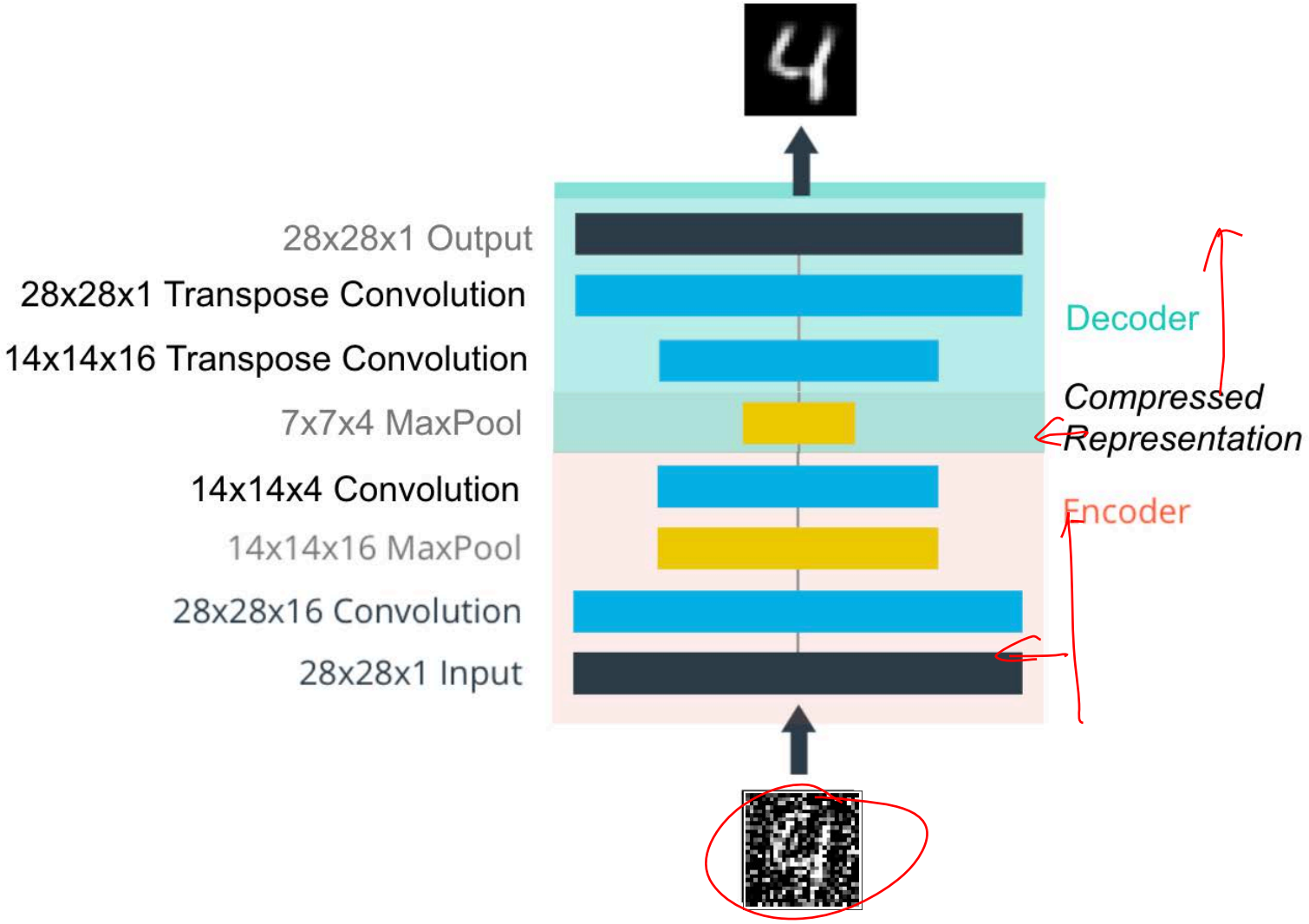
- A FCN is able to deal with any input/output size



Long, Shelhamer, Darrell - Fully Convolutional Networks for Semantic Segmentation, CVPR 2015, PAMI 2016



Convolutional Auto-Encoder



```python
import torch.nn as nn
import torch.nn.functional as F

# define the NN architecture
class ConvAutoencoder(nn.Module):
    def __init__(self):
        super(ConvAutoencoder, self).__init__()
        ## encoder layers ##
        # conv layer (depth from 1 --> 16), 3x3 kernels
        self.conv1 = nn.Conv2d(1, 16, 3, padding=1)
        # conv layer (depth from 16 --> 8), 3x3 kernels
        self.conv2 = nn.Conv2d(16, 4, 3, padding=1)
        # pooling layer to reduce x-y dims by two; kernel and stride of 2
        self.pool = nn.MaxPool2d(2, 2)

        ## decoder layers ##
        self.conv4 = nn.Conv2d(4, 16, 3, padding=1)
        self.conv5 = nn.Conv2d(16, 1, 3, padding=1)
```

```python
    def forward(self, x):
        # add layer, with relu activation function
        # and maxpooling after
        x = F.relu(self.conv1(x))
        x = self.pool(x)
        # add hidden layer, with relu activation function
        x = F.relu(self.conv2(x))
        x = self.pool(x)  # compressed representation

        ## decoder
        # upsample, followed by a conv layer, with relu activation function
        # this function is called `interpolate` in some PyTorch versions
        x = F.upsample(x, scale_factor=2, mode='nearest')
        x = F.relu(self.conv4(x))
        # upsample again, output should have a sigmoid applied
        x = F.upsample(x, scale_factor=2, mode='nearest')
        x = F.sigmoid(self.conv5(x))

        return x
```

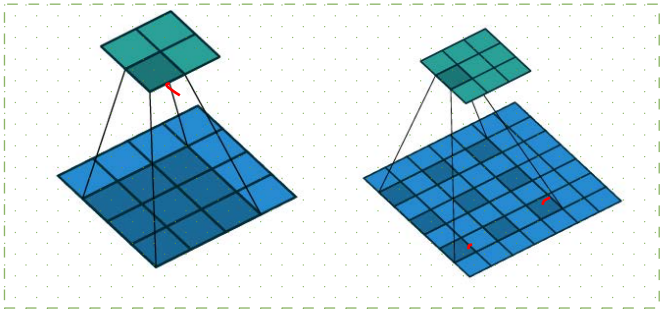
```python
import torch.nn as nn
import torch.nn.functional as F

# define the NN architecture
class ConvAutoencoder(nn.Module):
    def __init__(self):
        super(ConvAutoencoder, self).__init__()
        ## encoder layers ##
        # conv layer (depth from 1 --> 16), 3x3 kernels
        self.conv1 = nn.Conv2d(1, 16, 3, padding=1)
        # conv layer (depth from 16 --> 8), 3x3 kernels
        self.conv2 = nn.Conv2d(16, 4, 3, padding=1)
        # pooling layer to reduce x-y dims by two; kernel and stride of 2
        self.pool = nn.MaxPool2d(2, 2)

        ## decoder layers ##
        self.conv4 = nn.Conv2d(4, 16, 3, padding=1)
        self.conv5 = nn.Conv2d(16, 1, 3, padding=1)
```

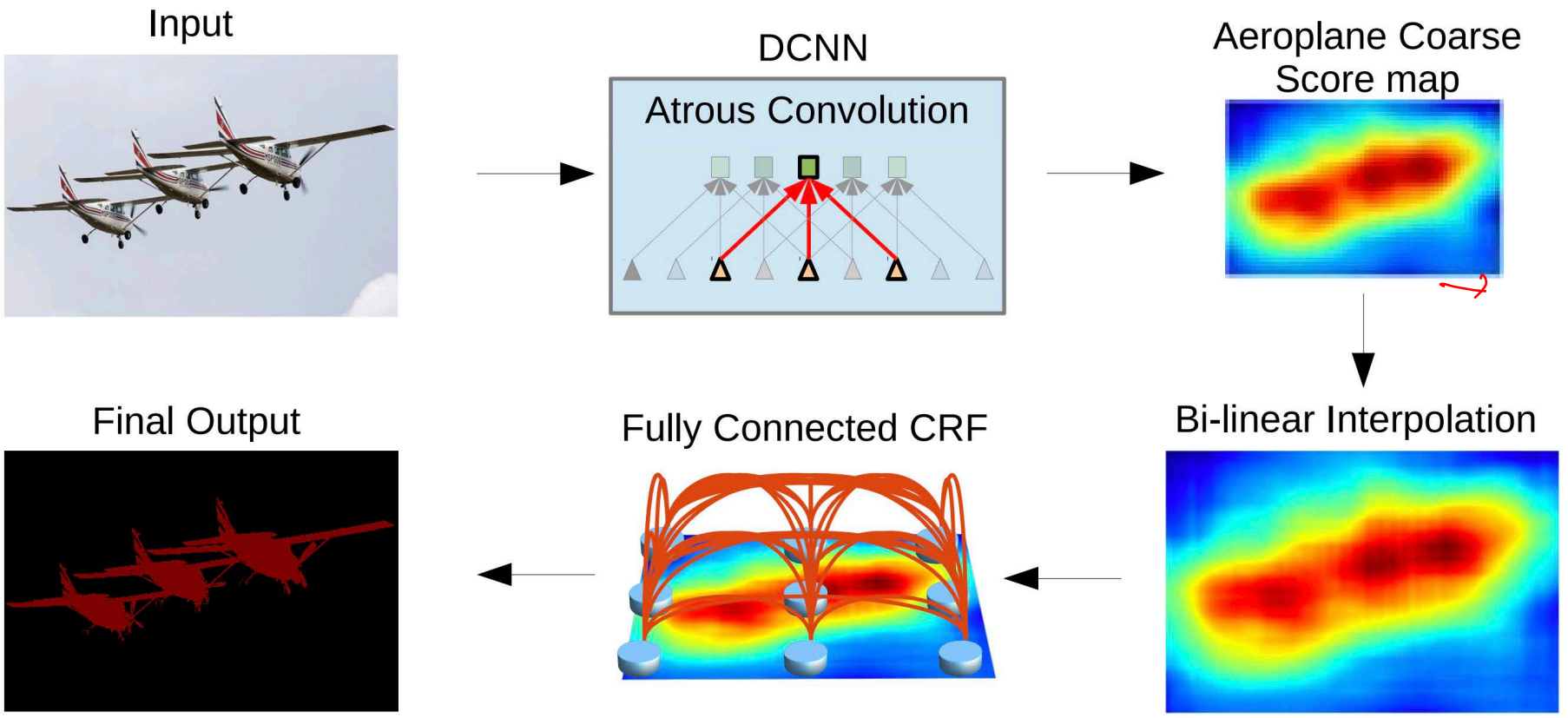
```
downsample = nn.Conv2d(16, 16, 3, stride=2, padding=1)

upsample = nn.ConvTranspose2d(16, 16, 3, stride=2, padding=1)
```

```python
    def forward(self, x):
        # add layer, with relu activation function
        # and maxpooling after
        x = F.relu(self.conv1(x))
        x = self.pool(x)
        # add hidden layer, with relu activation function
        x = F.relu(self.conv2(x))
        x = self.pool(x)  # compressed representation

        ## decoder
        # upsample, followed by a conv layer, with relu activa
        # this function is called `interpolate` in some PyTorc
        x = F.upsample(x, scale_factor=2, mode='nearest')
        x = F.relu(self.conv4(x))
        # upsample again, output should have a sigmoid applied
        x = F.upsample(x, scale_factor=2, mode='nearest')
        x = F.sigmoid(self.conv5(x))

        return x
```

mode (string)
– algorithm used for **upsampling**: 'nearest' | 'linear' | 'bilinear' | 'bicubic' | 'trilinear'.
Default: 'nearest'

DeepLab



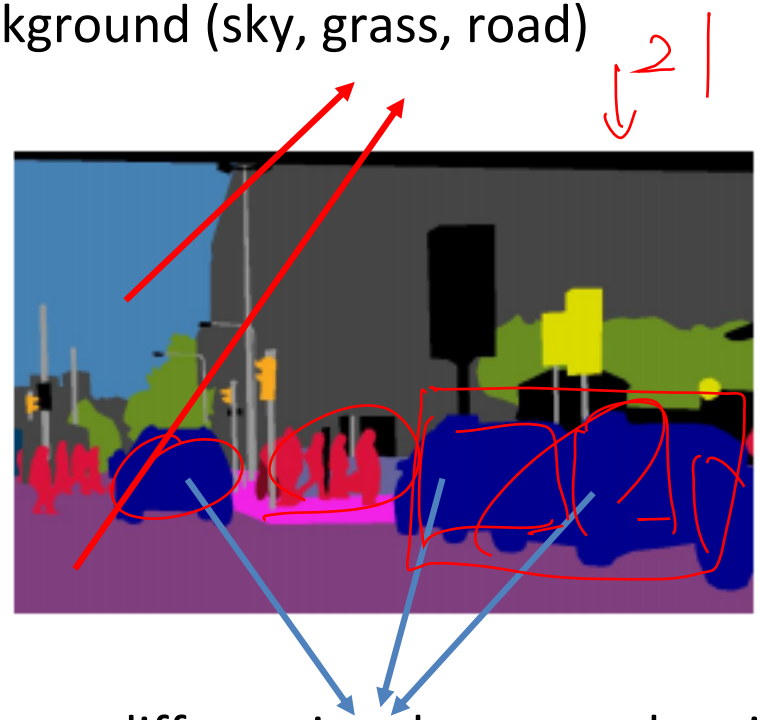


Chen, Liang-Chieh, et al. "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs." IEEE transactions on pattern analysis and machine intelligence 40.4 (2017): 834-848.

Instance segmentation

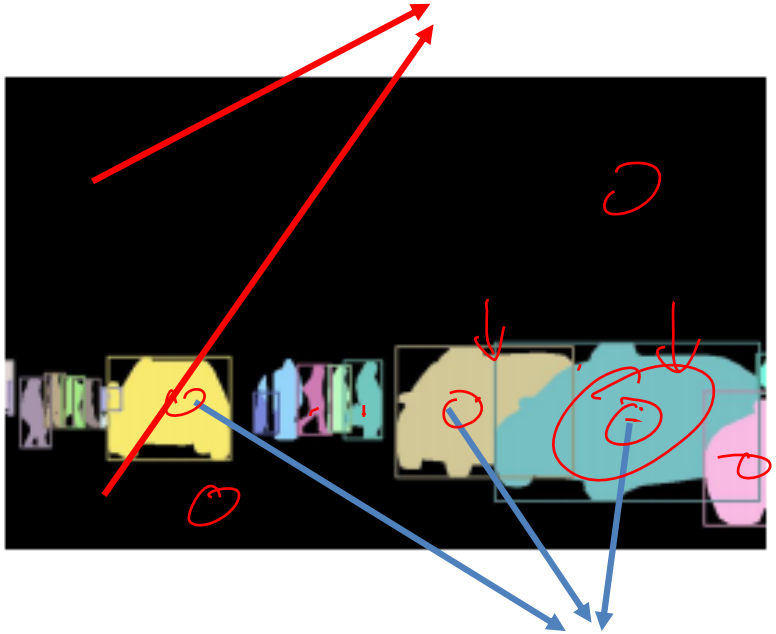
Instance Segmentation

Label every pixel, including the background (sky, grass, road)



Do not differentiate between the pixels coming from instances of the same class

Do not label pixels coming from uncountable objects (sky, grass, road)



Differentiate between the pixels coming from instances of the same class

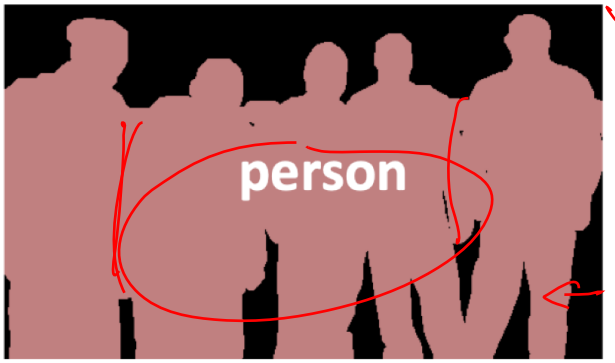
Instance segmentation methods

Proposal-based

FCN-based

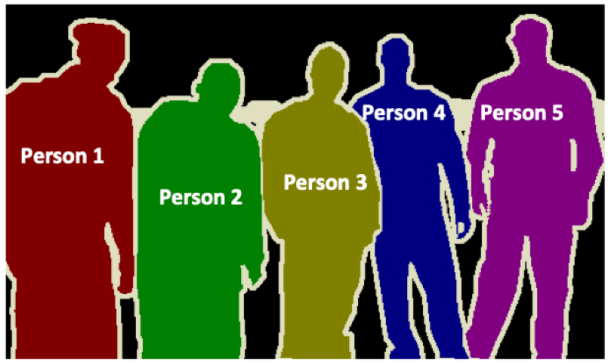
1. Proposals

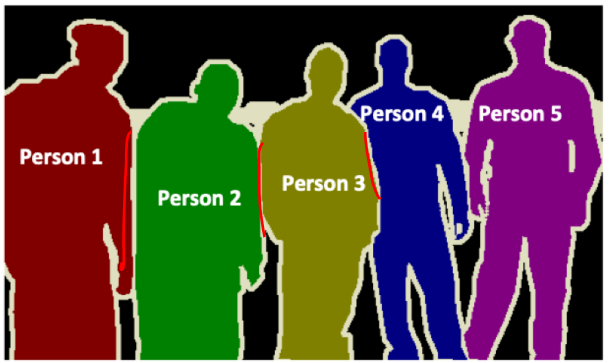
vs.



1. Semantic segmentation

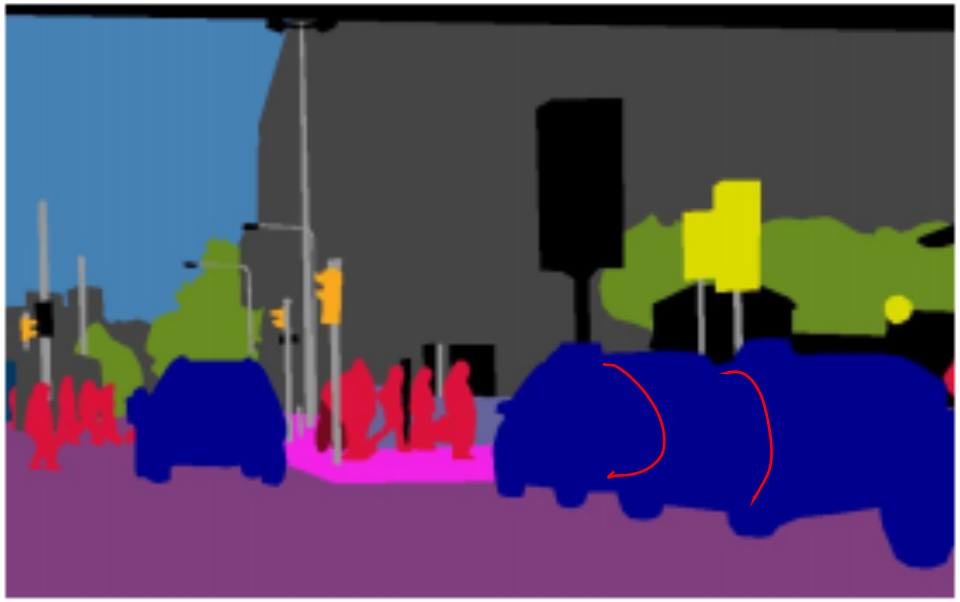
2. Assign a class





2. Find instances

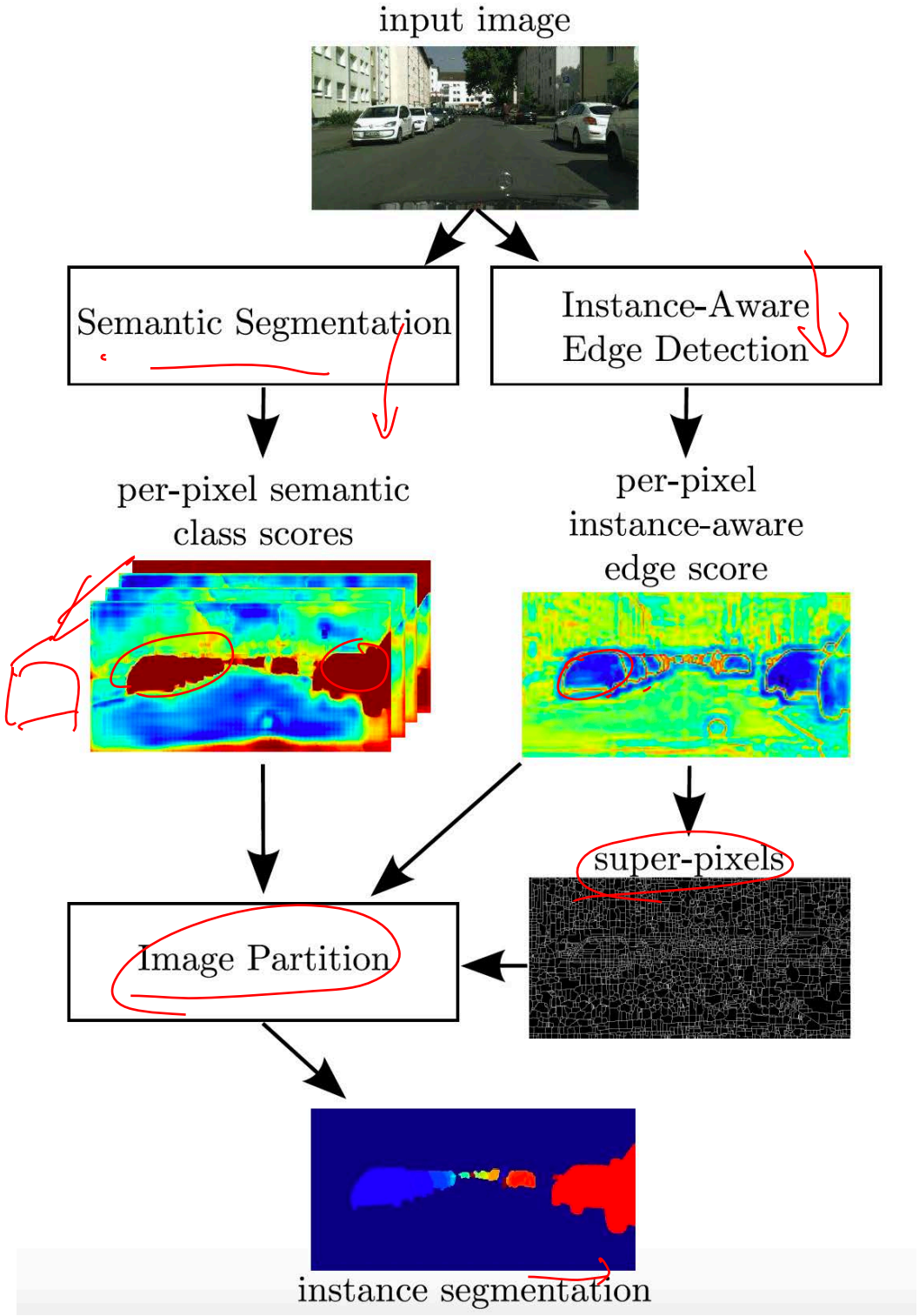
FCN-based methods



A semantic map…

We already know how to obtain this!

Instancecut



Kirillov, Alexander, et al. "Instancecut: from edges to instances with multicut." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

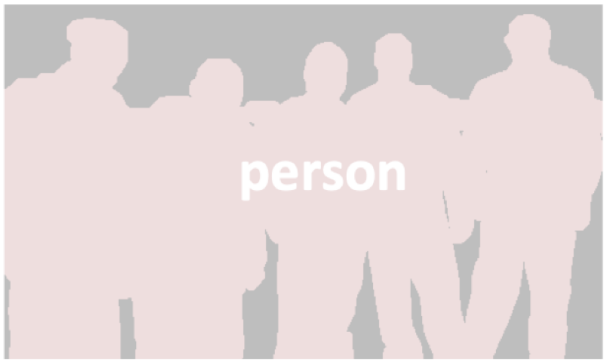
Instance segmentation methods

Proposal-based

FCN-based

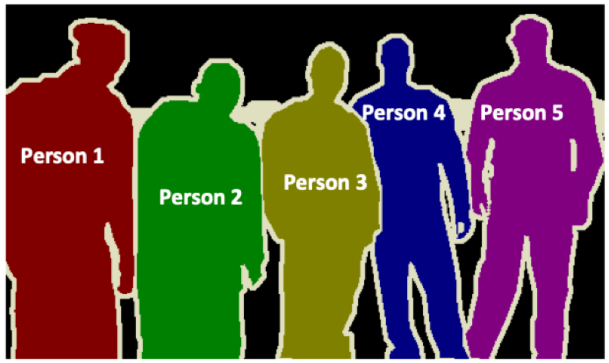
1. Proposals

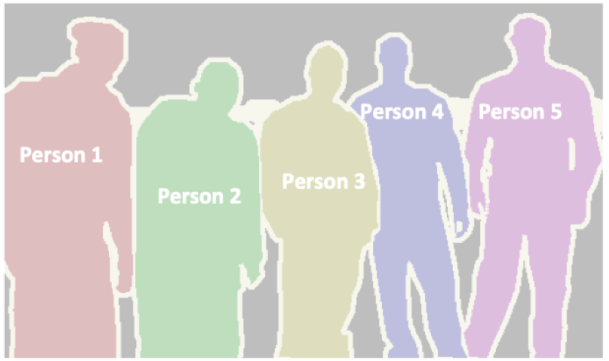
vs.



1. Semantic segmentatio

2. Assign a class



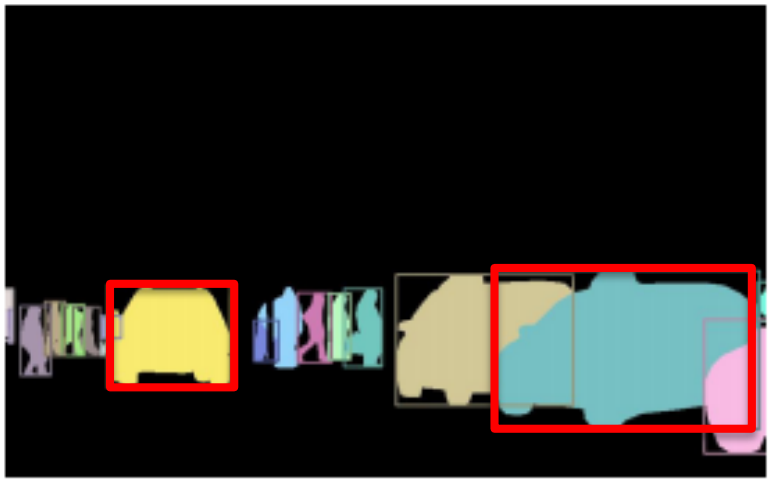


2. Find instances

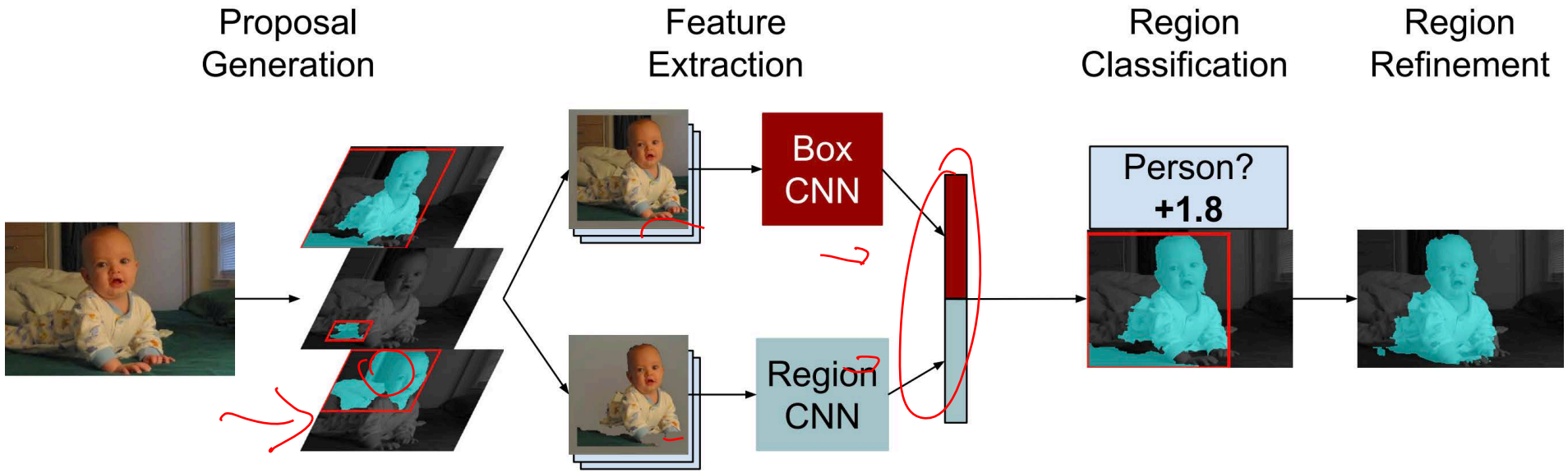
Proposal-based methods

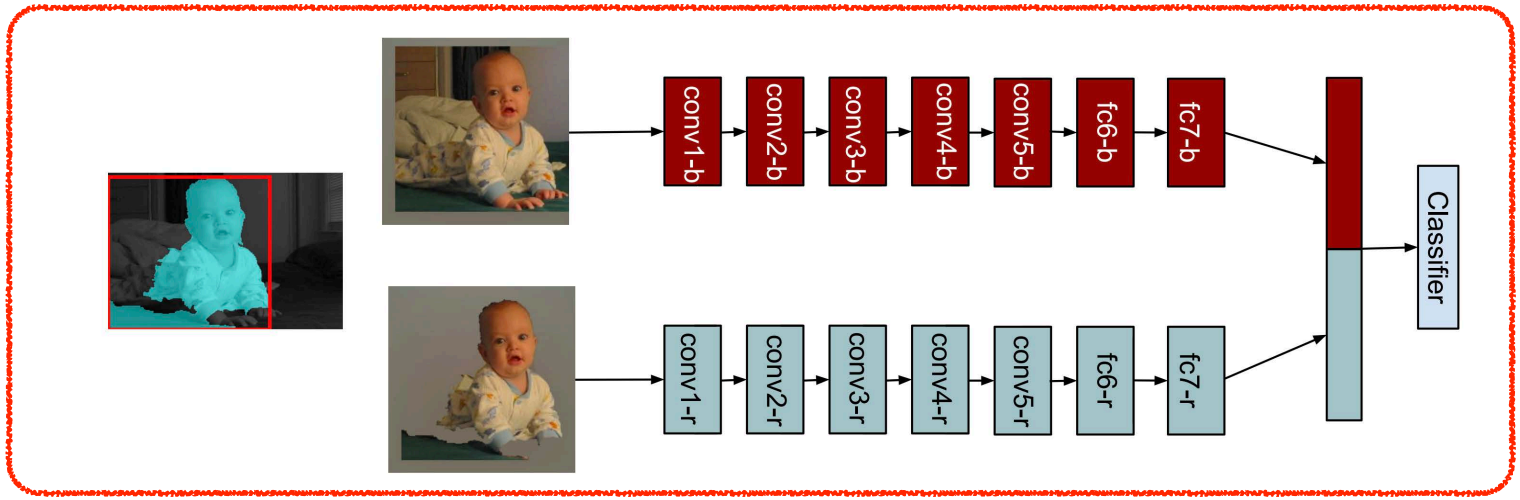
Bounding boxes…..

We already know how to obtain those!



SDS





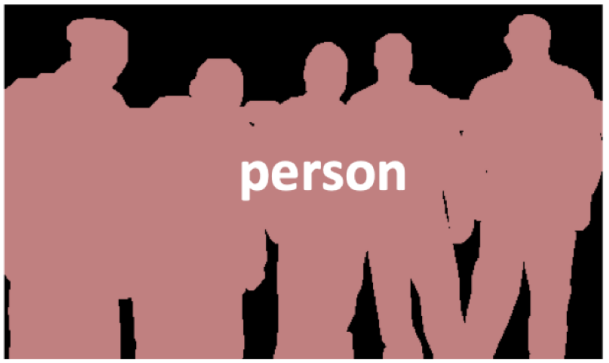
Hariharan et al. “Simultaneous Detection and Segmentation“. ECCV 2014

The best of both worlds

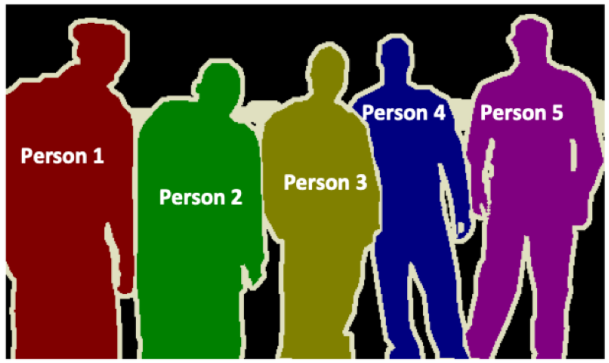
Proposal-based

FCN-based

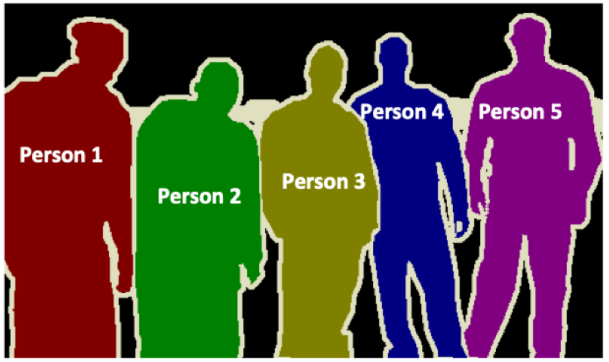
1. Proposals



1. Semantic segmentatio



2. Assign a class



2. Find instances

Mask R-CNN

Mask r-cnn

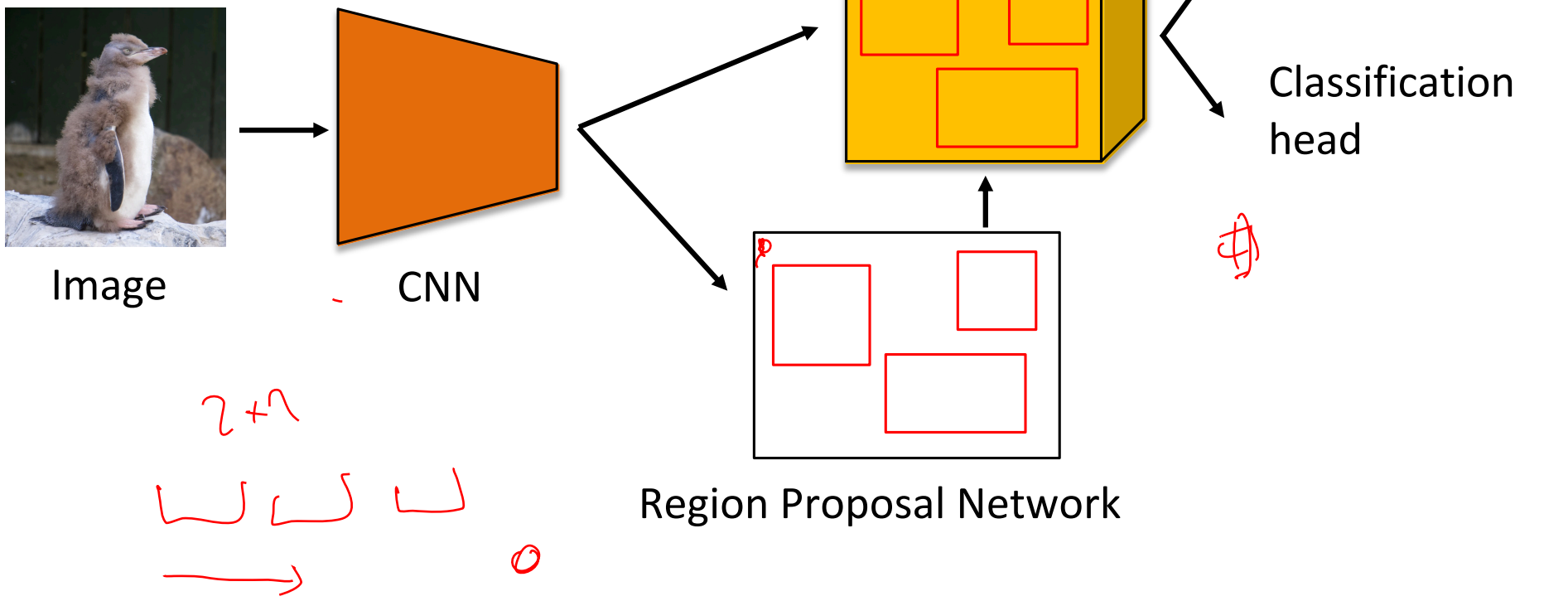
K He, G Gkioxari, P Dollár… - Proceedings of the IEEE …, 2017 - openaccess.thecvf.com

… **mask**, with minimal modification **Mask R-CNN** can be applied to detect instance-specific poses. Without tricks, **Mask R-CNN** … **masks** on the top 100 detection boxes, **Mask R-CNN** adds a …

☆ Save Cite Cited by 21993 Related articles All 20 versions »

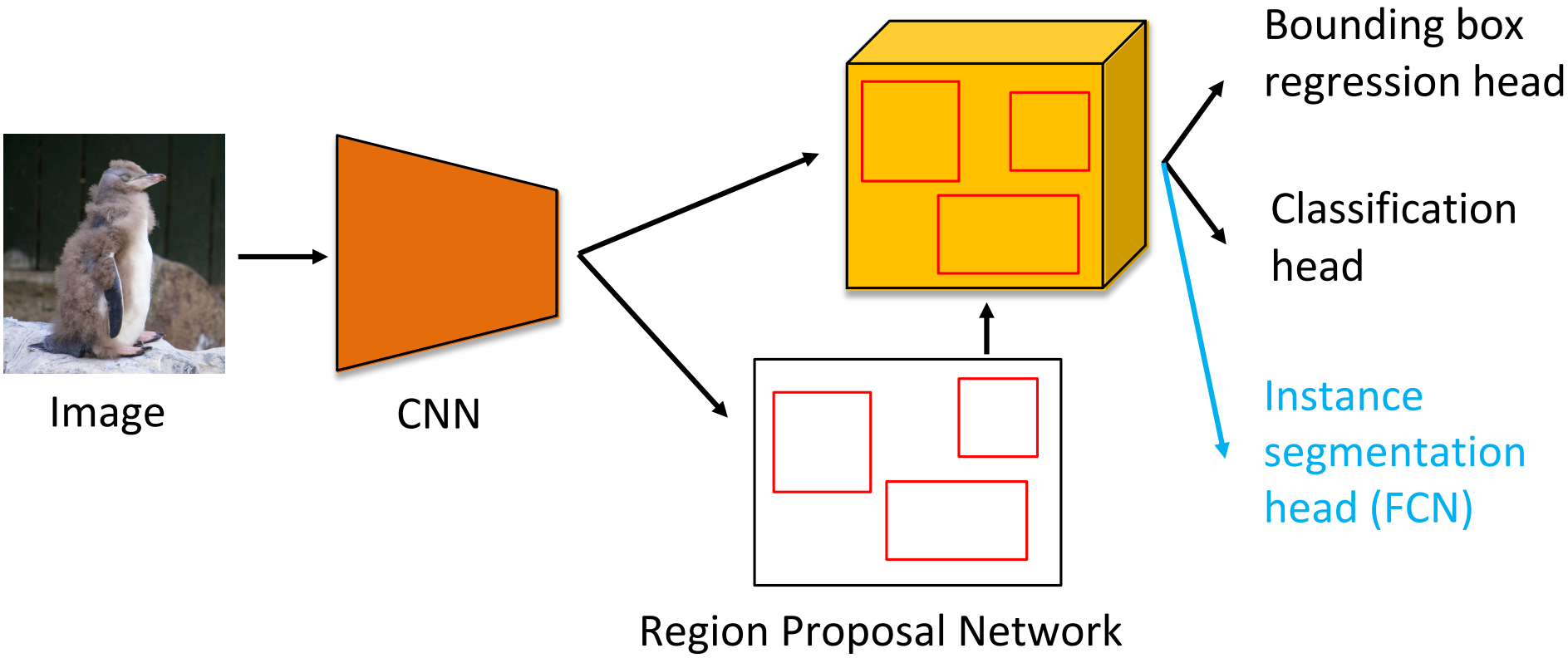
What is Mask-RCNN?

- Starting from the Faster R-CNN architecture



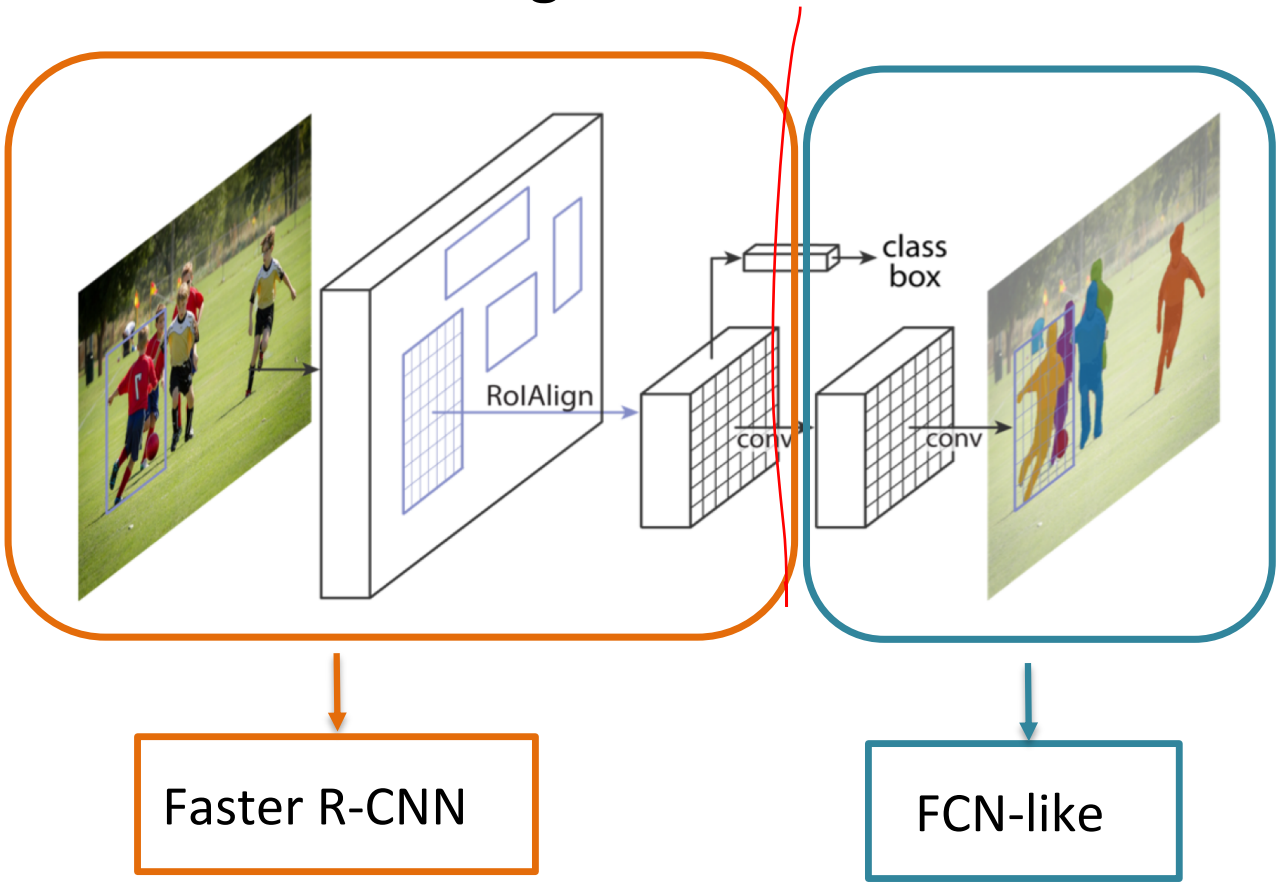
What is Mask-RCNN?

- Faster R-CNN + FCN for segmentation



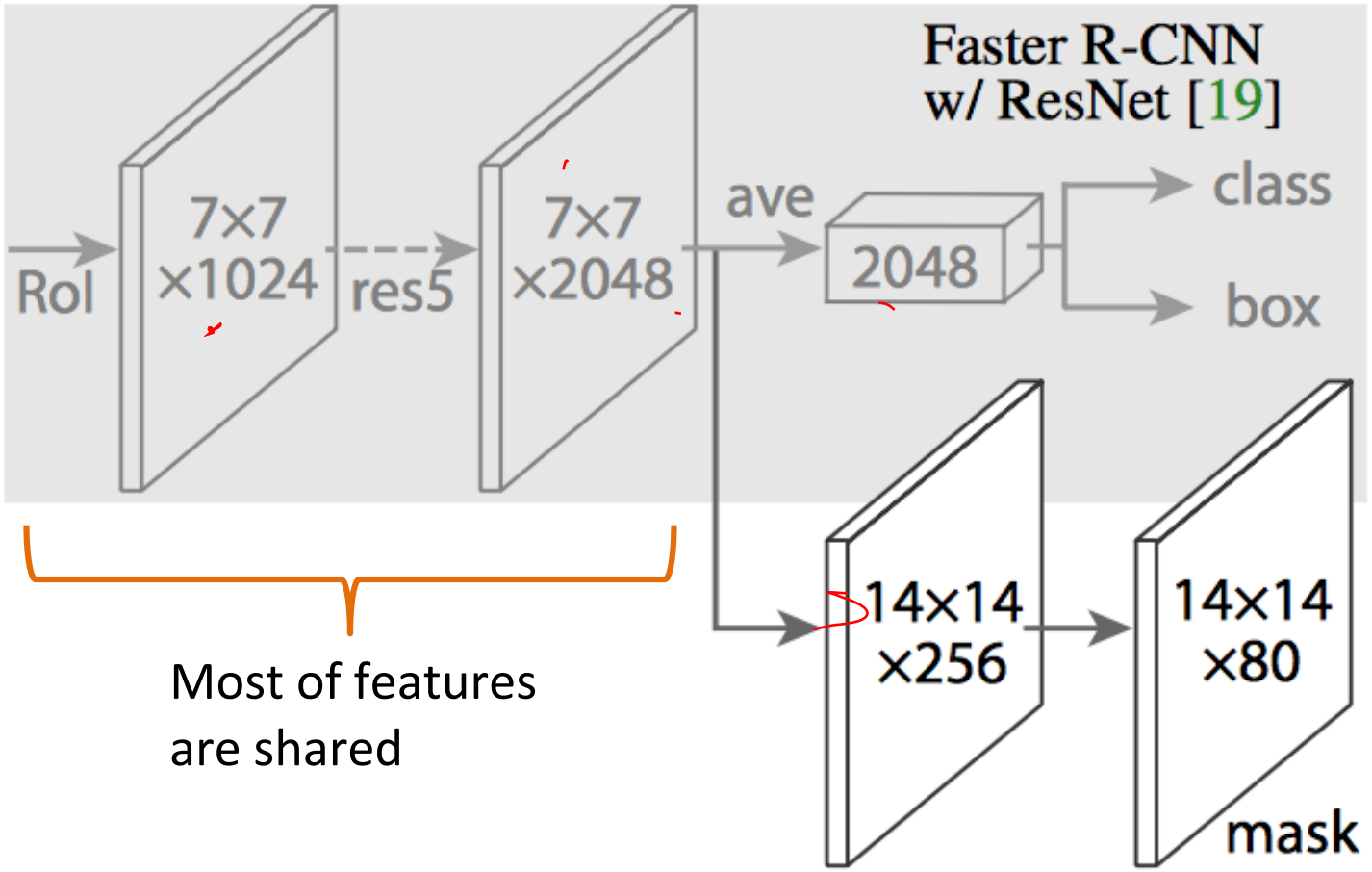
What is Mask-RCNN?

- Faster R-CNN + FCN for segmentation



Mask loss = binary cross entropy per pixel for the k segmentation classes

Mask R-CNN



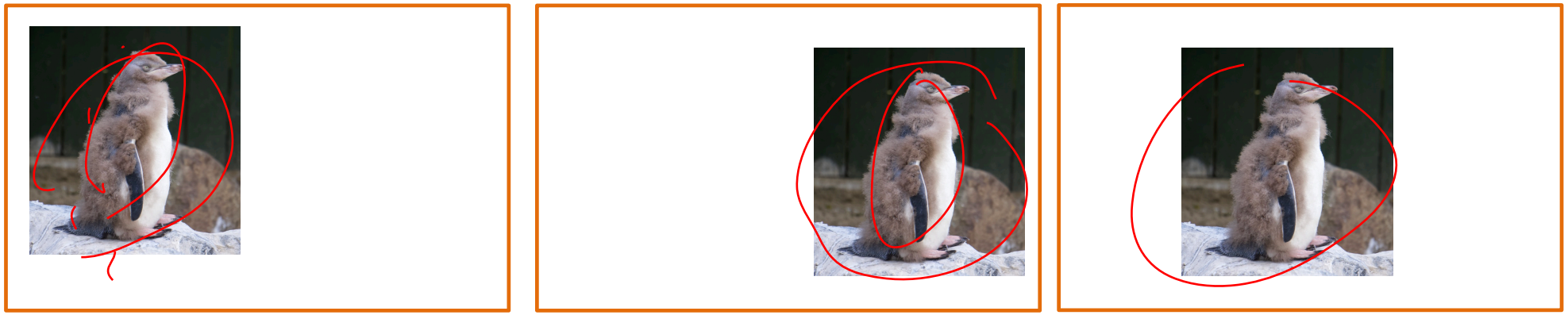
Object recognition head

Segmentation head

He, Kaiming, et al. "Mask r-cnn." Proceedings of the IEEE international conference on computer vision. 2017.

Detection vs. Segmentation

- Detection: for object classification, you require **invariant** representations



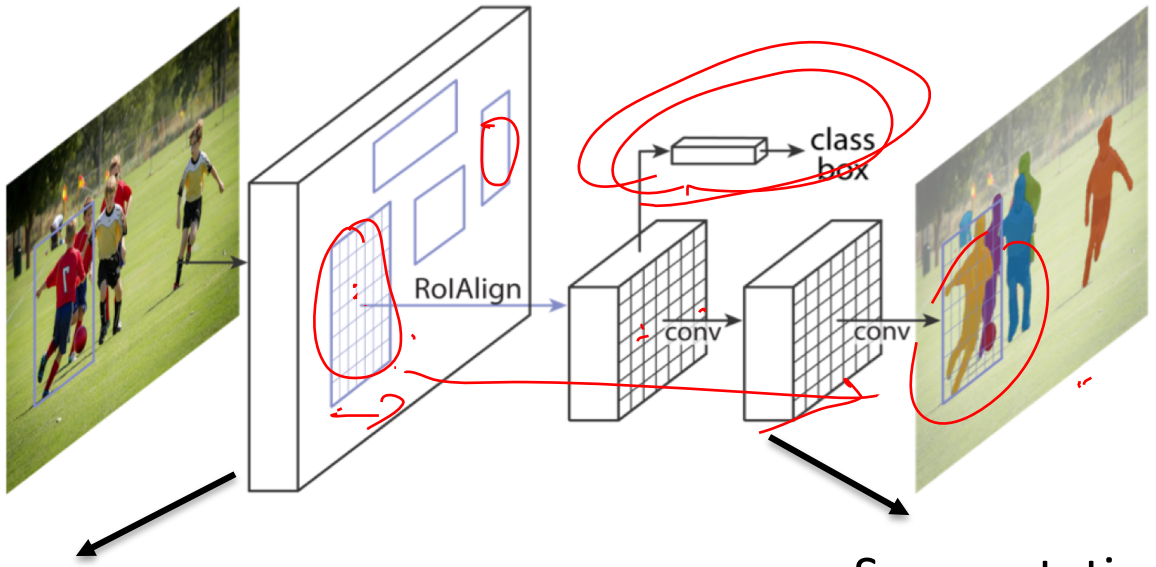
Translation equivariance: wherever the penguin is in the image, I still want to have “penguin” as my classification output

Detection vs. Segmentation

- Detection: for object classification, you require **invariant** representations
- Segmentation: you require **equivariant** representations
  - Translated object: Translated mask
  - Scaled object: scaled mask
  - For semantic segmentation, small objects are less important (less pixels), but for instance segmentation, all objects (no matter the size) are equally important

Mask-RCNN: operations

- What operations are equivariant?



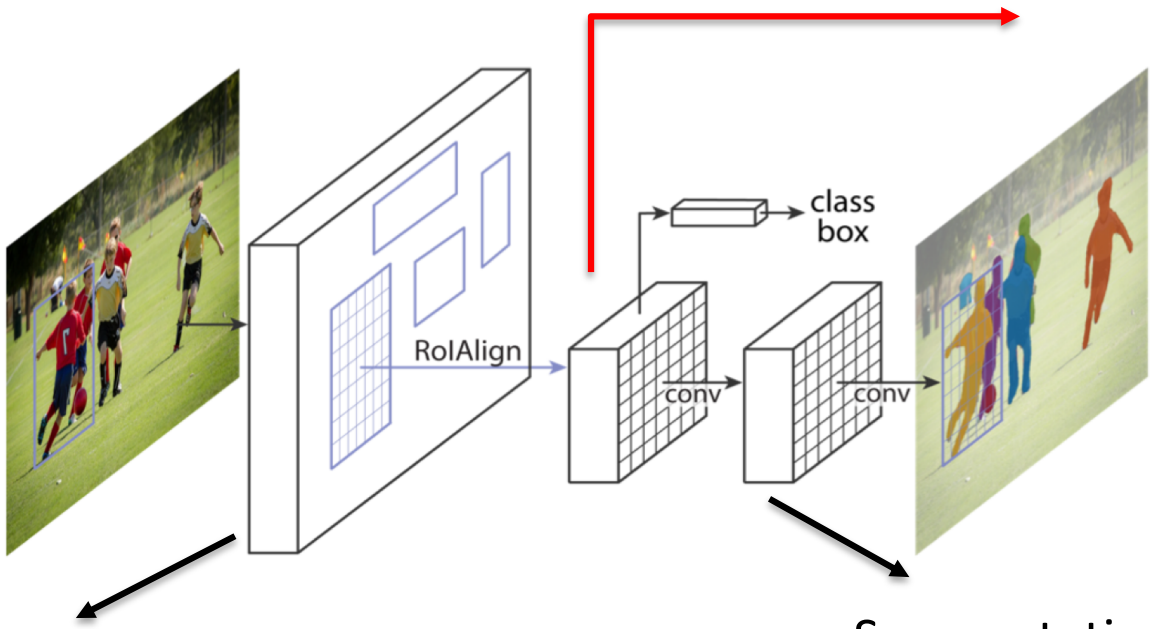
Features extraction = convolutional layers

Segmentation head is a fully convolutional network

Mask-RCNN: operations

- What operations are equivariant?

Fully connected layers and global pooling layers give invariance!

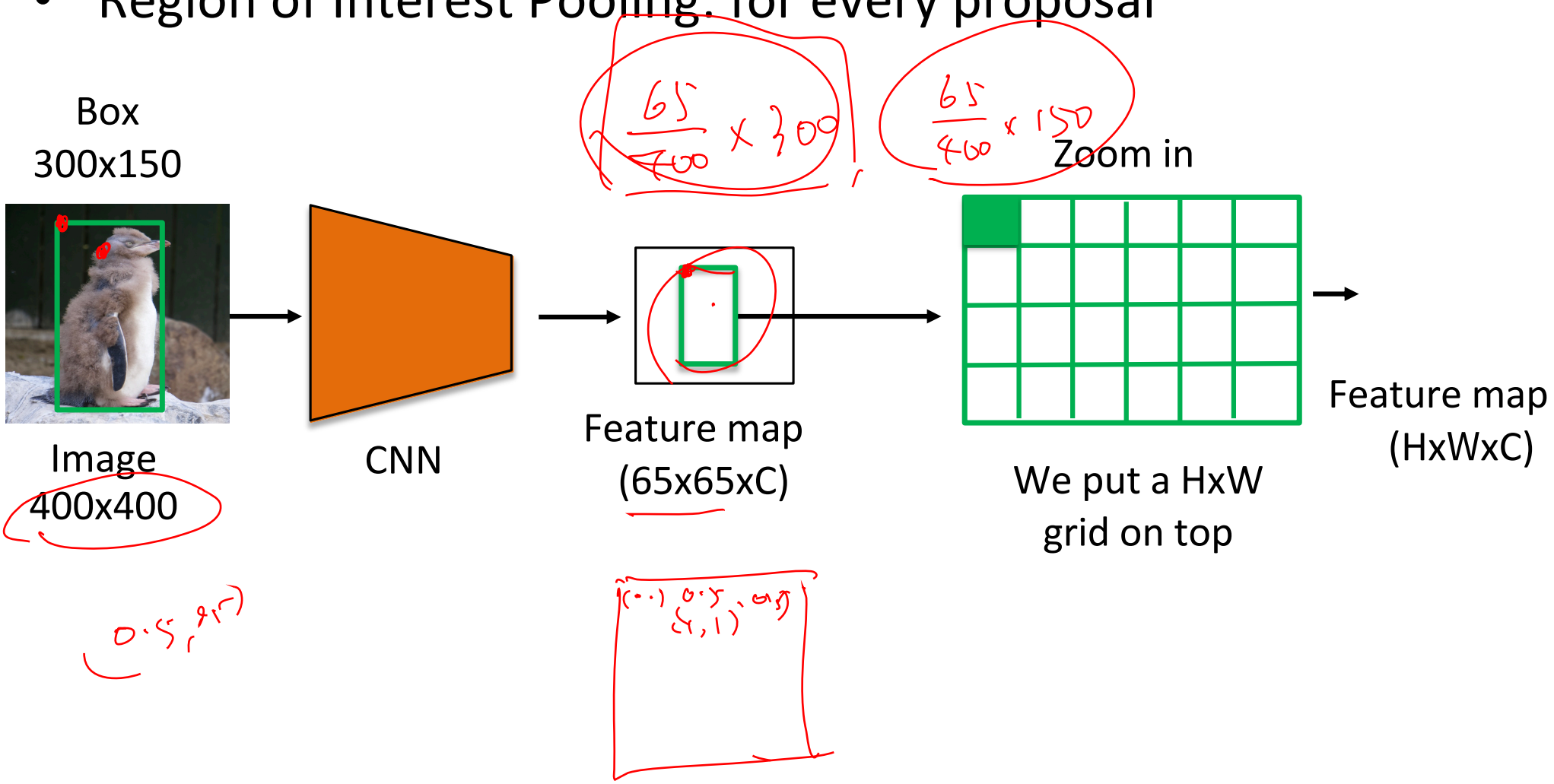


Features extraction = convolutional layers

Segmentation head is a fully convolutional network

Recall: RoI pooling

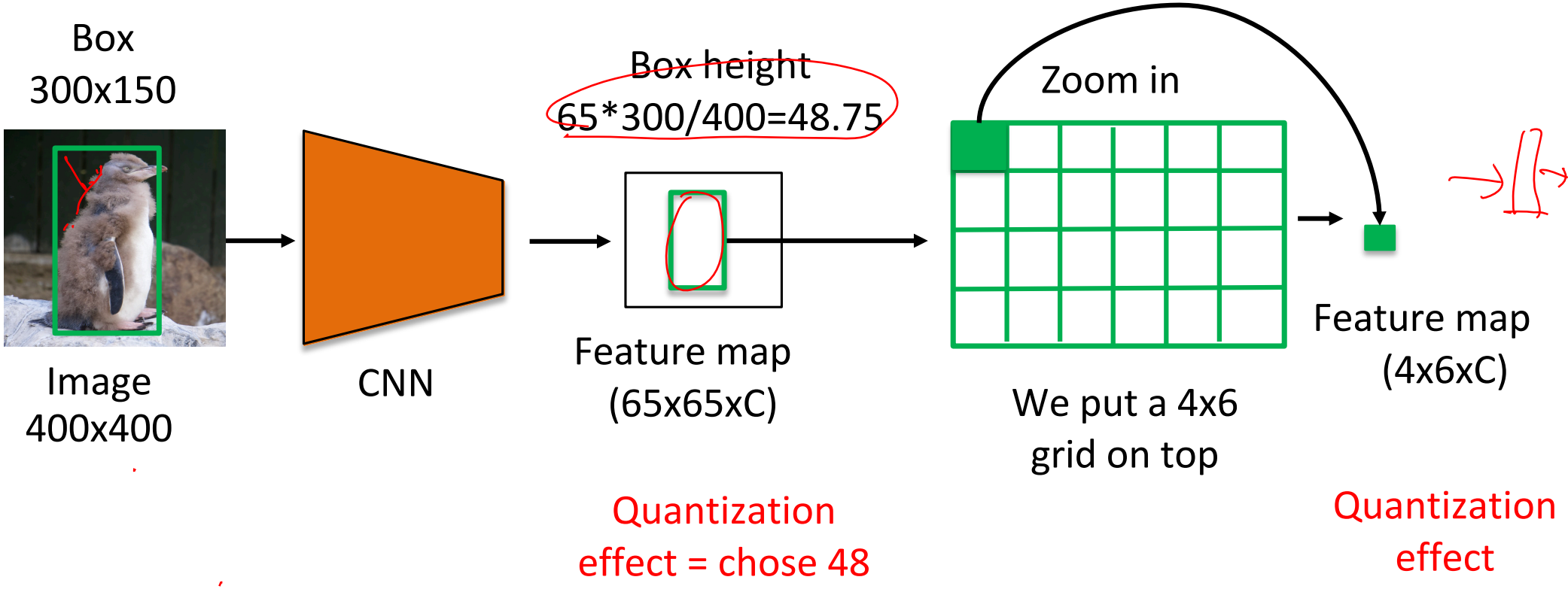
- Region of Interest Pooling: for every proposal



Recall: RoI pooling

Not suitable to extract pixel-wise precise masks

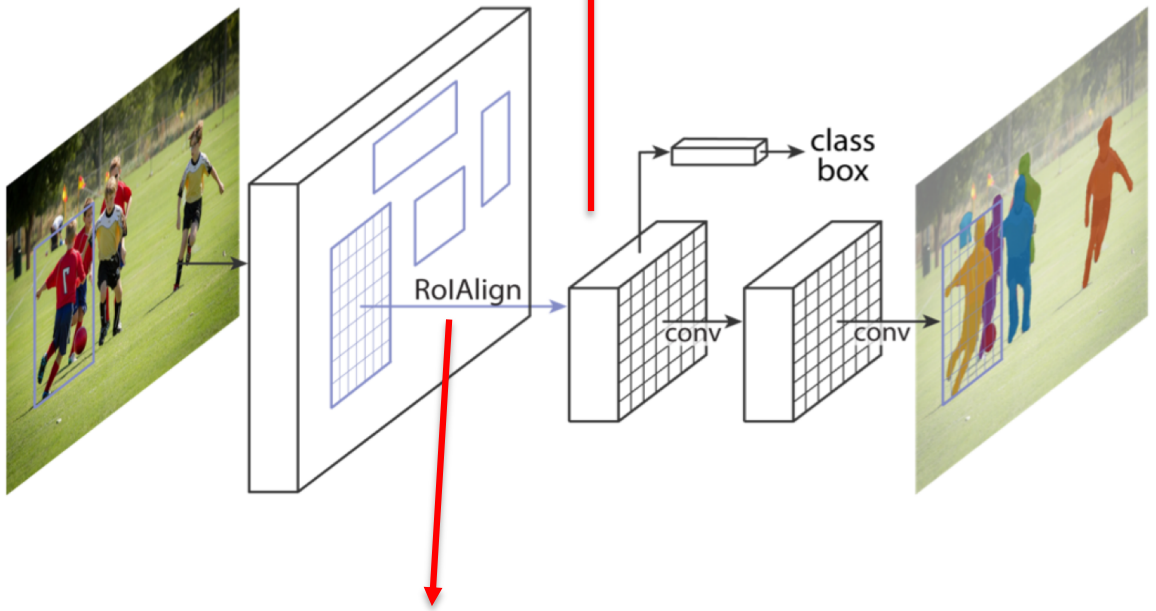
- Let us look at sizes



Mask-RCNN: operations

- Make all operations equivariant

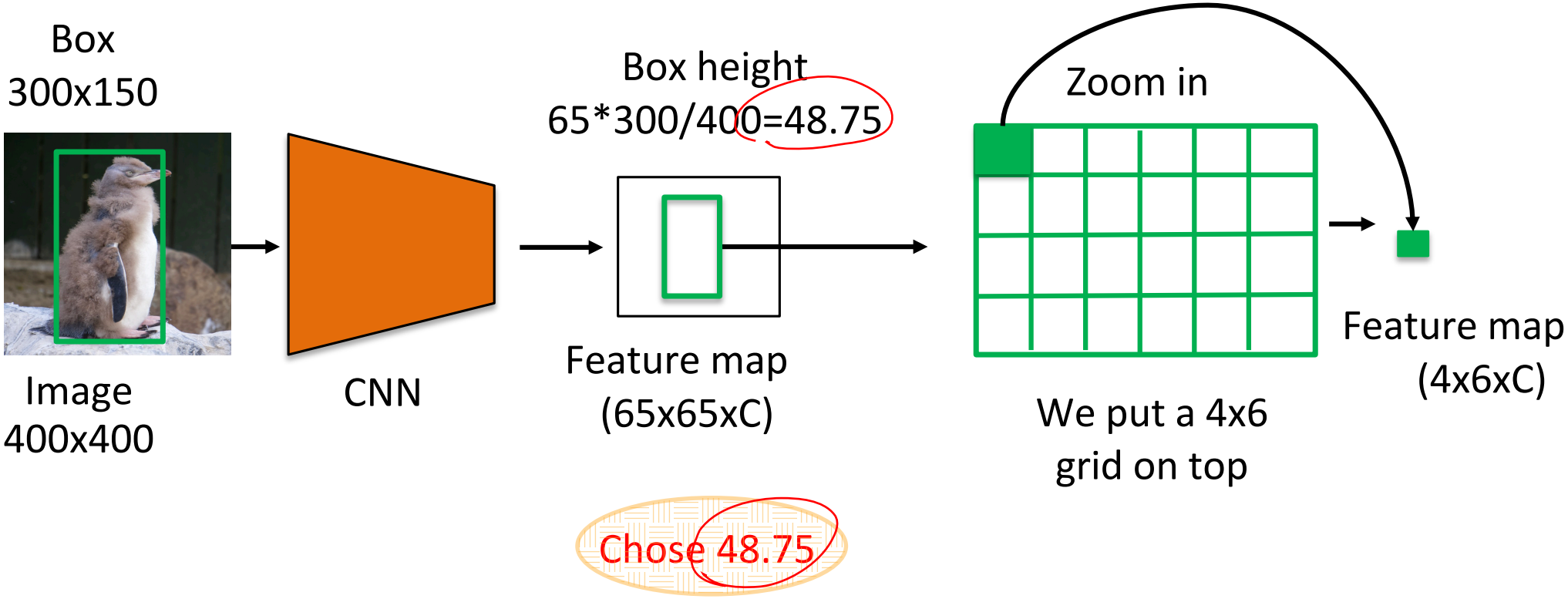
Fully connected layers and global pooling layers give invariance!



Exchange RoI pooling by an equivariant operation = RoI Align

RoIAlign

- Erase quantization effects



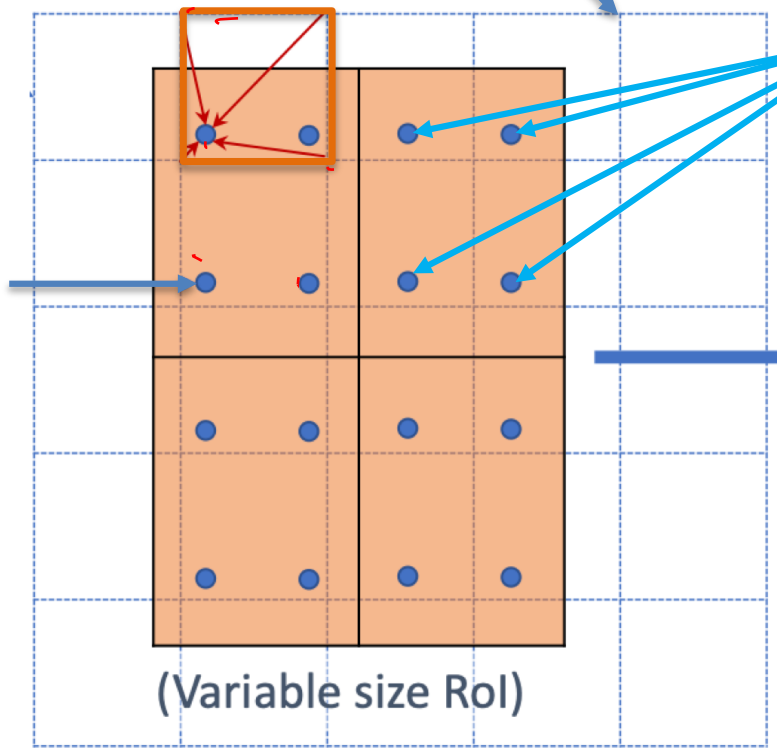
ROIAlign

To obtain the value use bilinear interpolation

Feature map

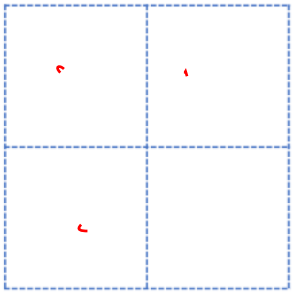
Each unit is sampled 4 times

Grid points for bilinear interpolation



RoIAlign output

(Fixed dimensional representation)



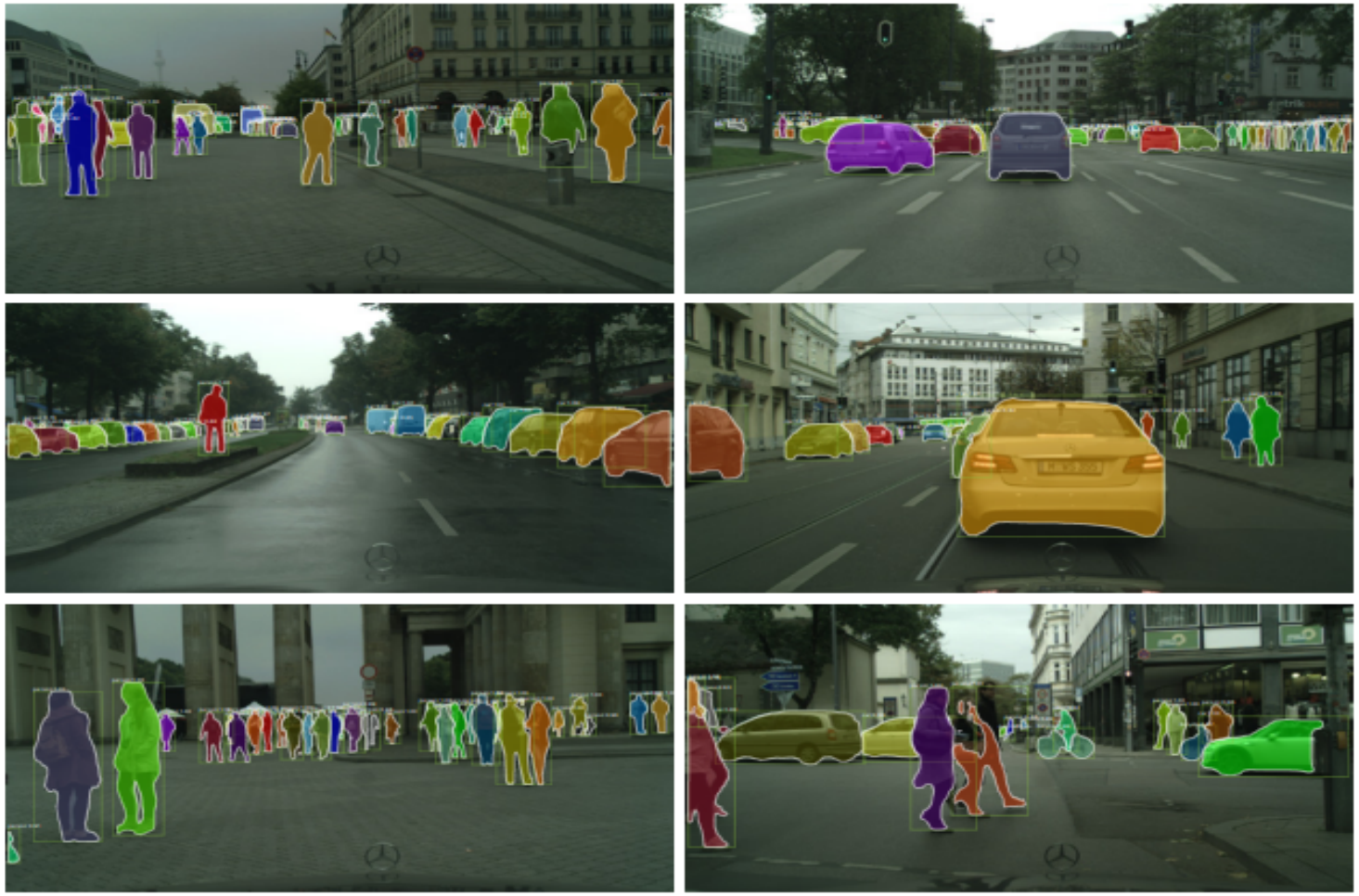
Max pooling on the 4 positions to obtain one output value

Mask R-CNN: qualitative results

Mask R-CNN: qualitative results

Mask R-CNN: qualitative results

Mask R-CNN: qualitative results



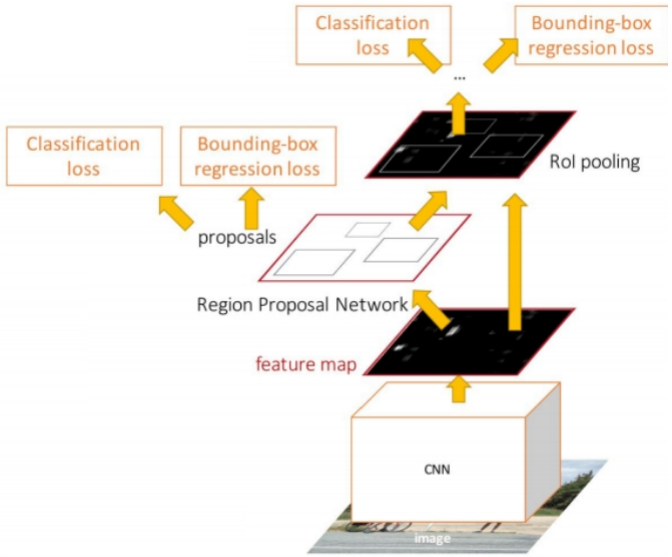
Mask R-CNN: extended for joints

Model a keypoint's location as a one-hot mask, and adopt Mask R-CNN to predict K masks (which are in the end only 1 pixel), one for each of K keypoint types (e.g., left shoulder, right elbow). This demonstrates the flexibility of Mask R-CNN.

Is one-stage vs two-stage also applicable to masks?

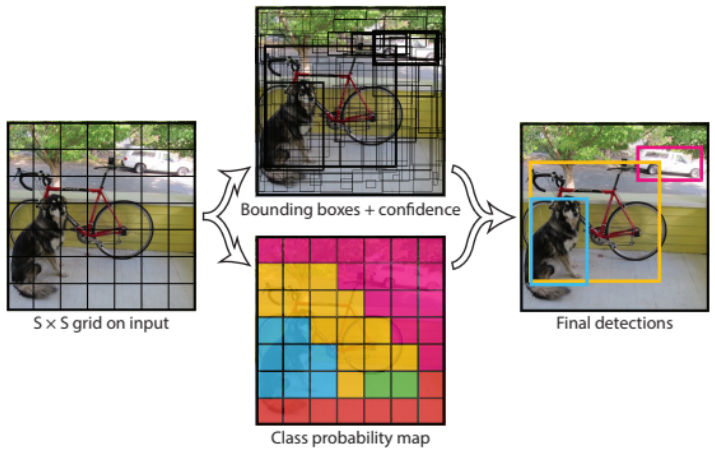
One-stage vs two-stage detectors

Faster R-CNN



Slower, but has higher performance

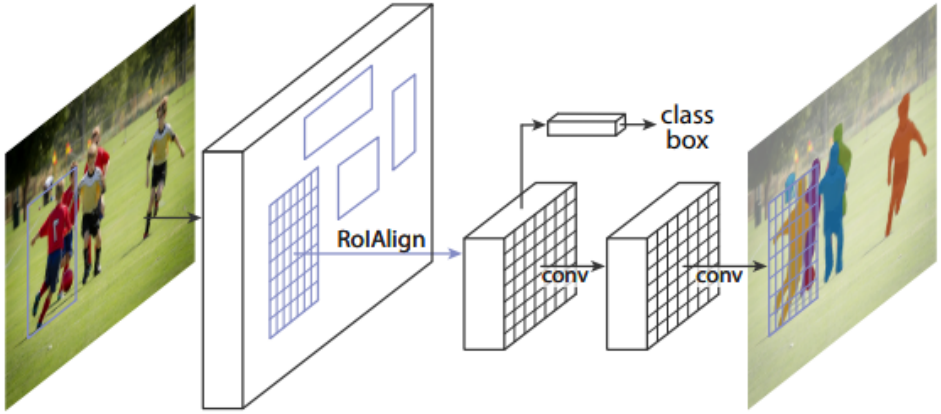
YOLO



Faster, but has lower performance

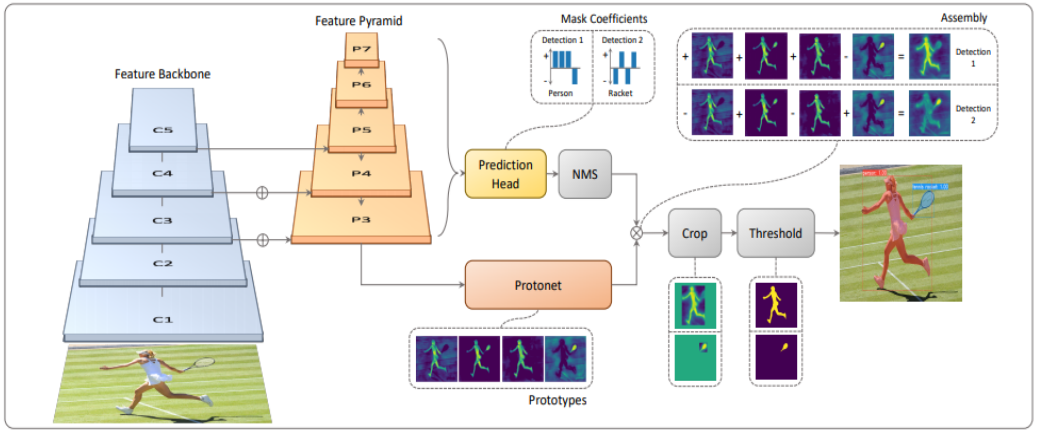
One-stage vs two-stage instance segmenters

Mask R-CNN



Slower, but has
higher performance

YOLACT

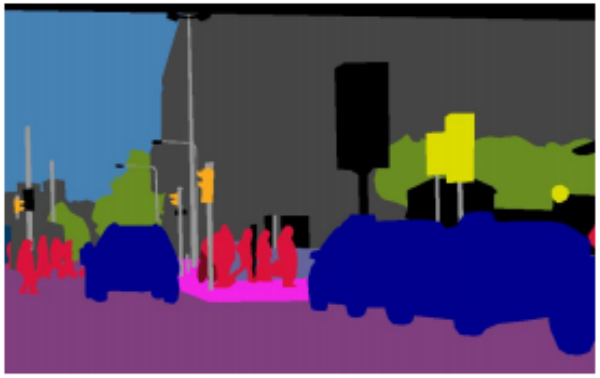


Faster, but has lower
performance

Panoptic Segmentation

Panoptic segmentation

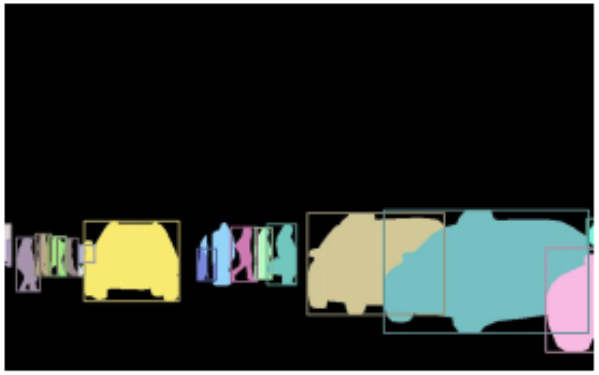
Semantic segmentation



FCN-like

+

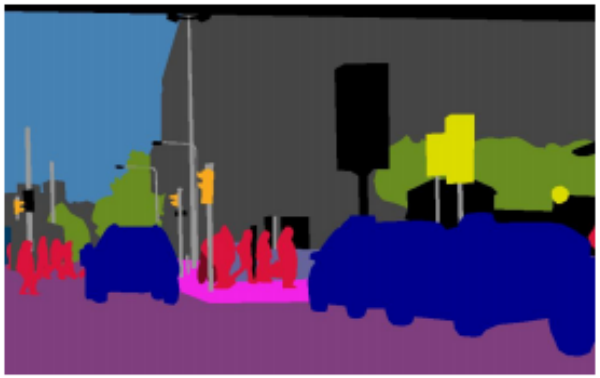
Instance segmentation



Mask R-CNN

Panoptic segmentation

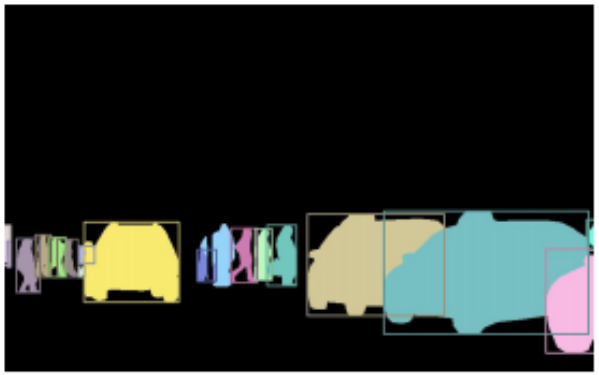
Semantic segmentation



FCN-like

+

Instance segmentation



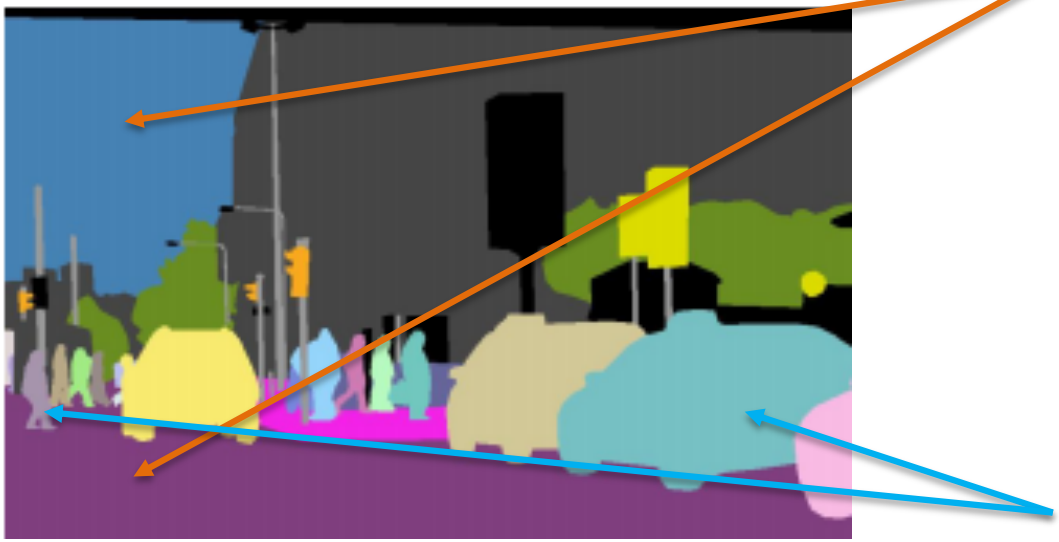
Mask R-CNN

=

Panoptic segmentation

UPSNet

Panoptic segmentation



It gives labels to uncountable objects called "stuff" (sky, road, etc), similar to FCN-like netsorks.

It differentiates between pixels coming from different distances of the same class (countable objects) called "things" (cars, pedestrians, etc).

Panoptic segmentation

Problem: some pixels might get classified as stuff from FCN network, while at the same time being classified as instances of some class from Mask R-CNN (conflicting results)!

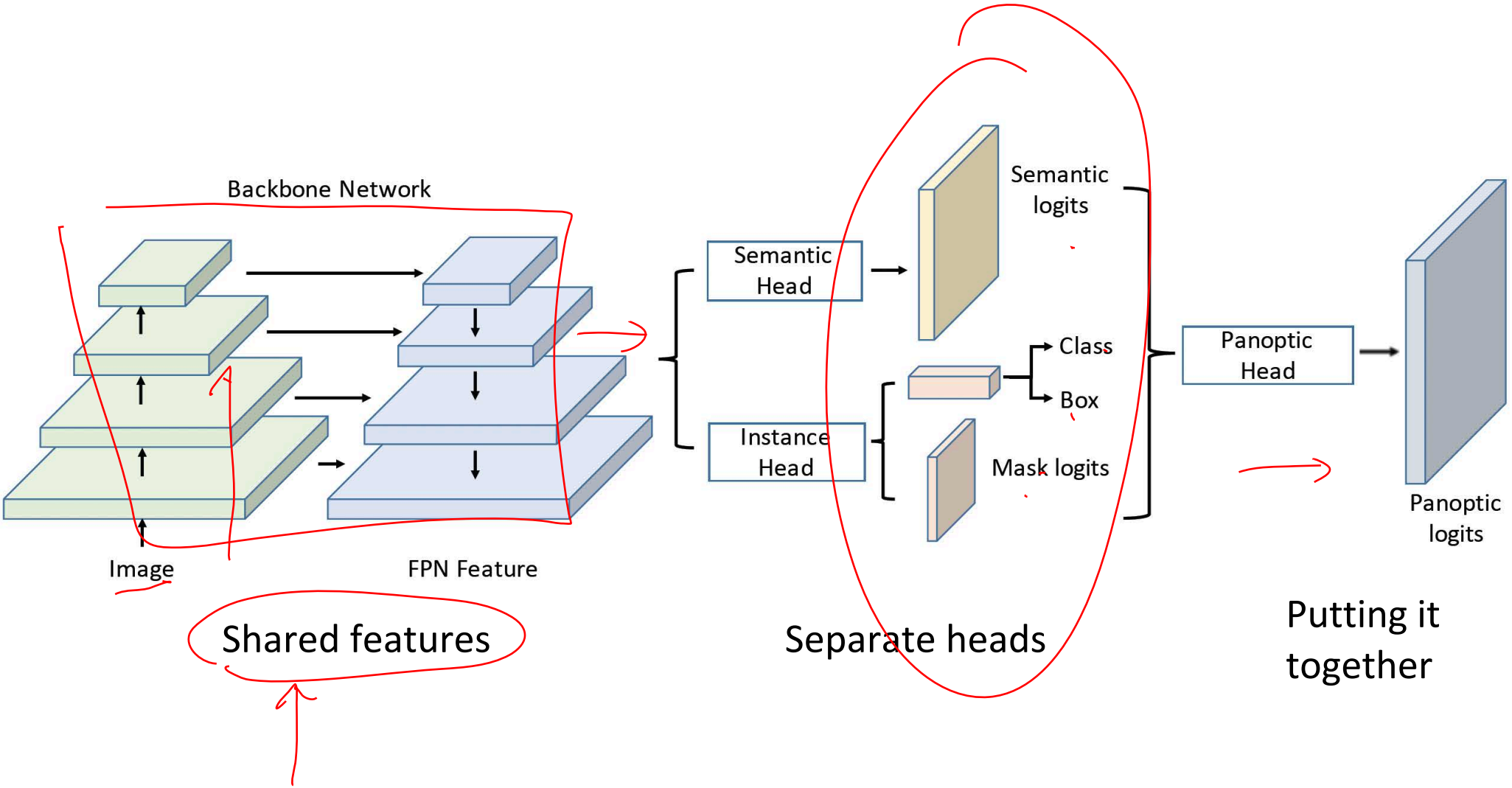
Xiong, Yuwen, et al. "Upsnet: A unified panoptic segmentation network." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

Panoptic segmentation

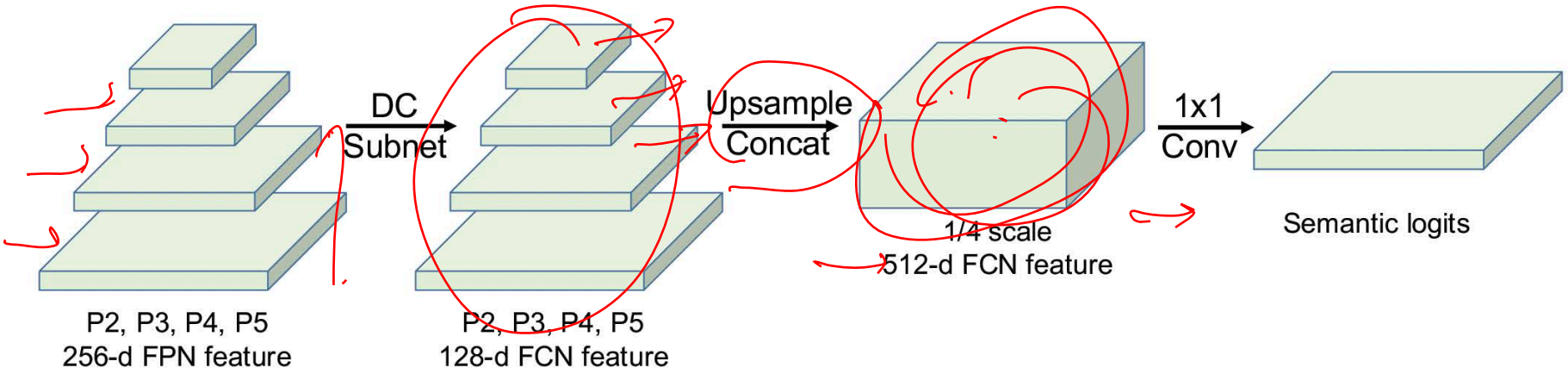
Solution: Parametric-free panoptic head which combines the information from the FCN and Mask R-CNN, giving final predictions.

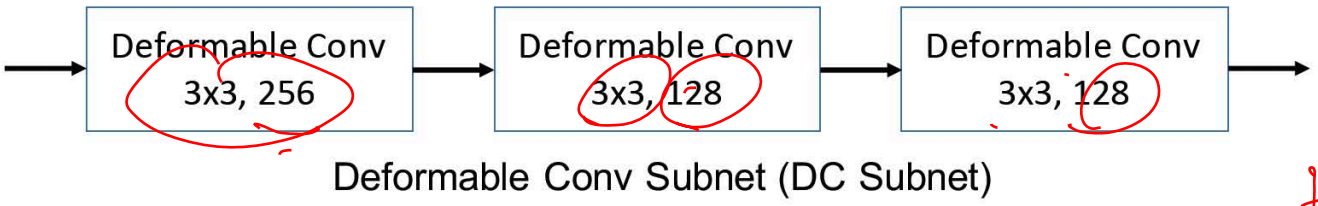
Xiong, Yuwen, et al. "Upsnet: A unified panoptic segmentation network." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

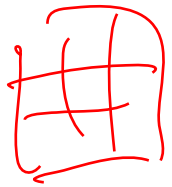
Network architecture



The semantic head

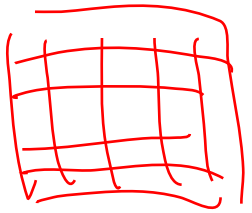






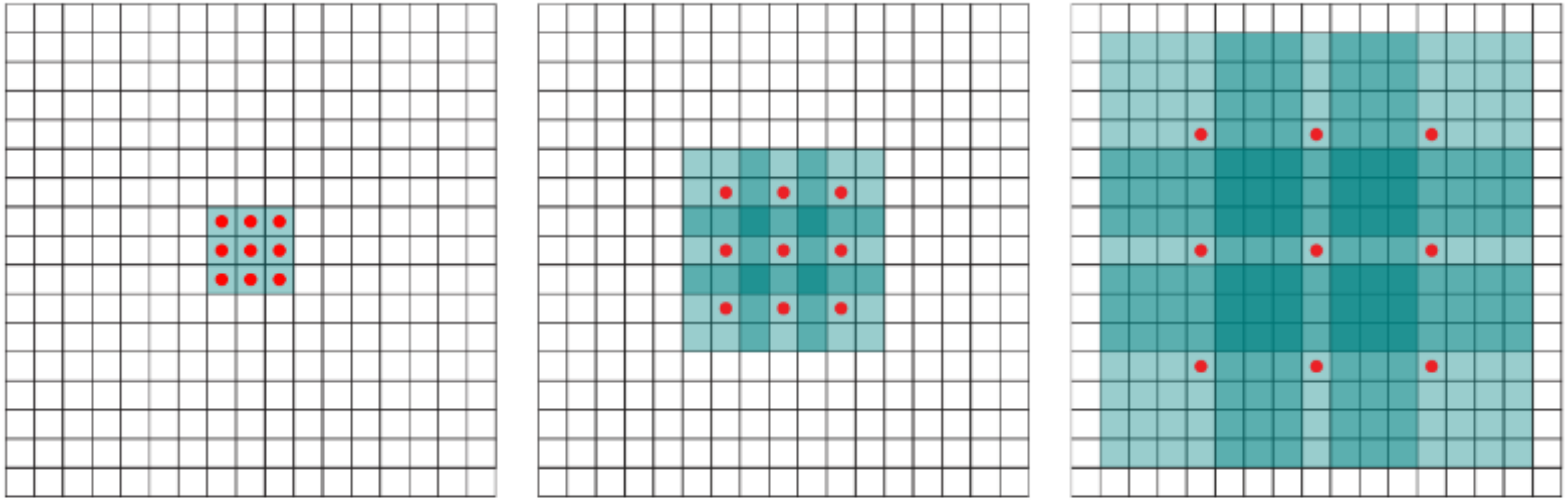
As all semantic heads a fully convolutional network.

New: deformable convolutions!



Dai, Jifeng, et al. "Deformable convolutional networks." ICCV. 2017.

Recall: Dilated (atrous) convolutions 2D



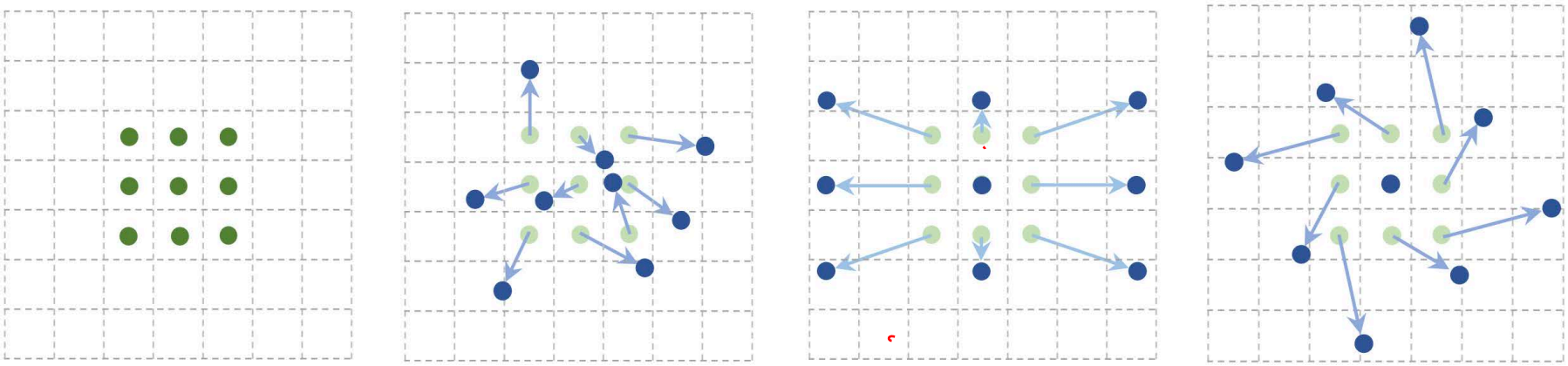
(a) (b) (c)

(a) the dilation parameter is 1, and each element produced by this filter has reception field of 3x3.

(b) the dilation parameter is 2, and each element produced by it has reception field of 7x7.

(c) the dilation parameter is 4, and each element produced by it has reception field of 15x15.

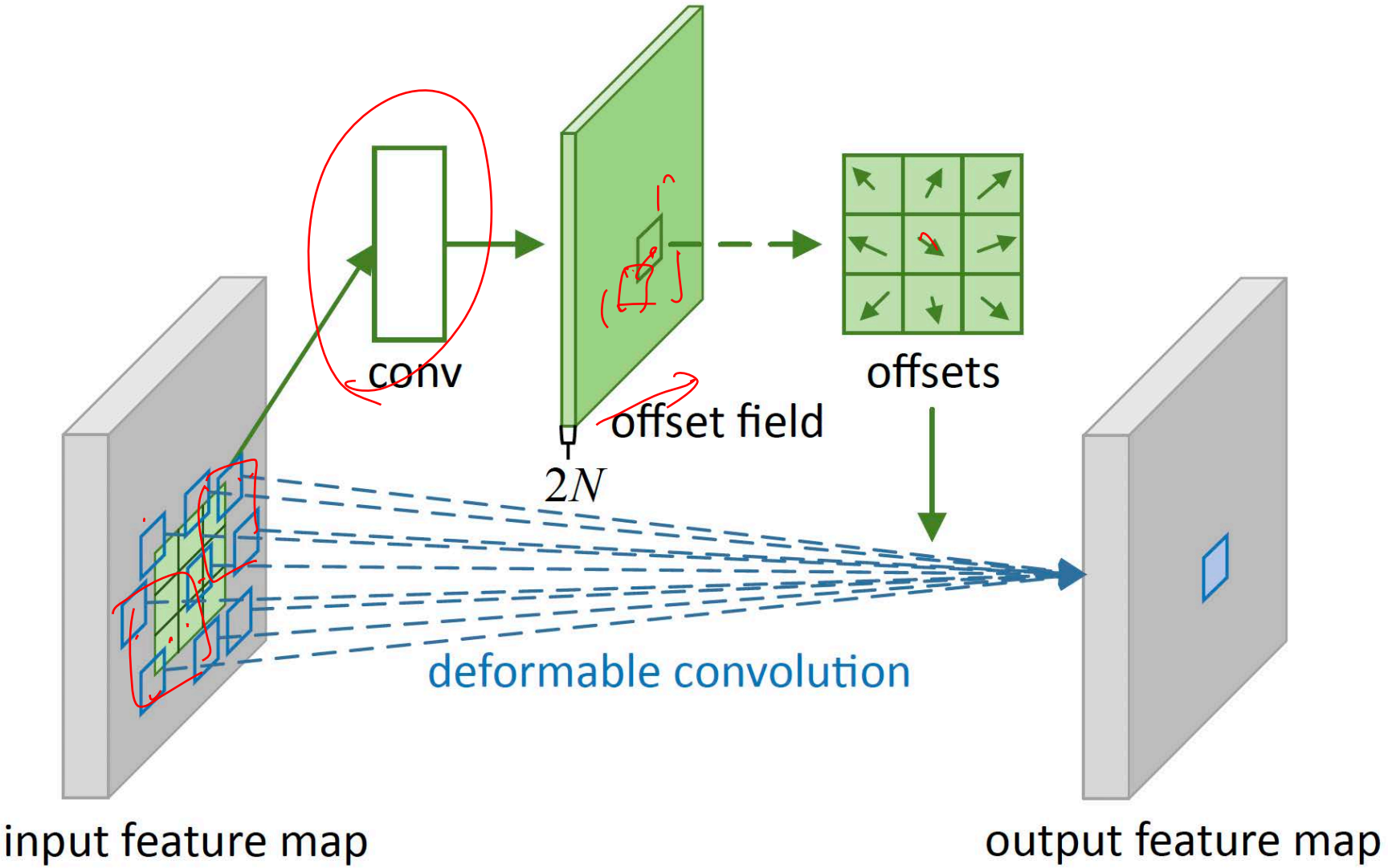
Deformable convolutions



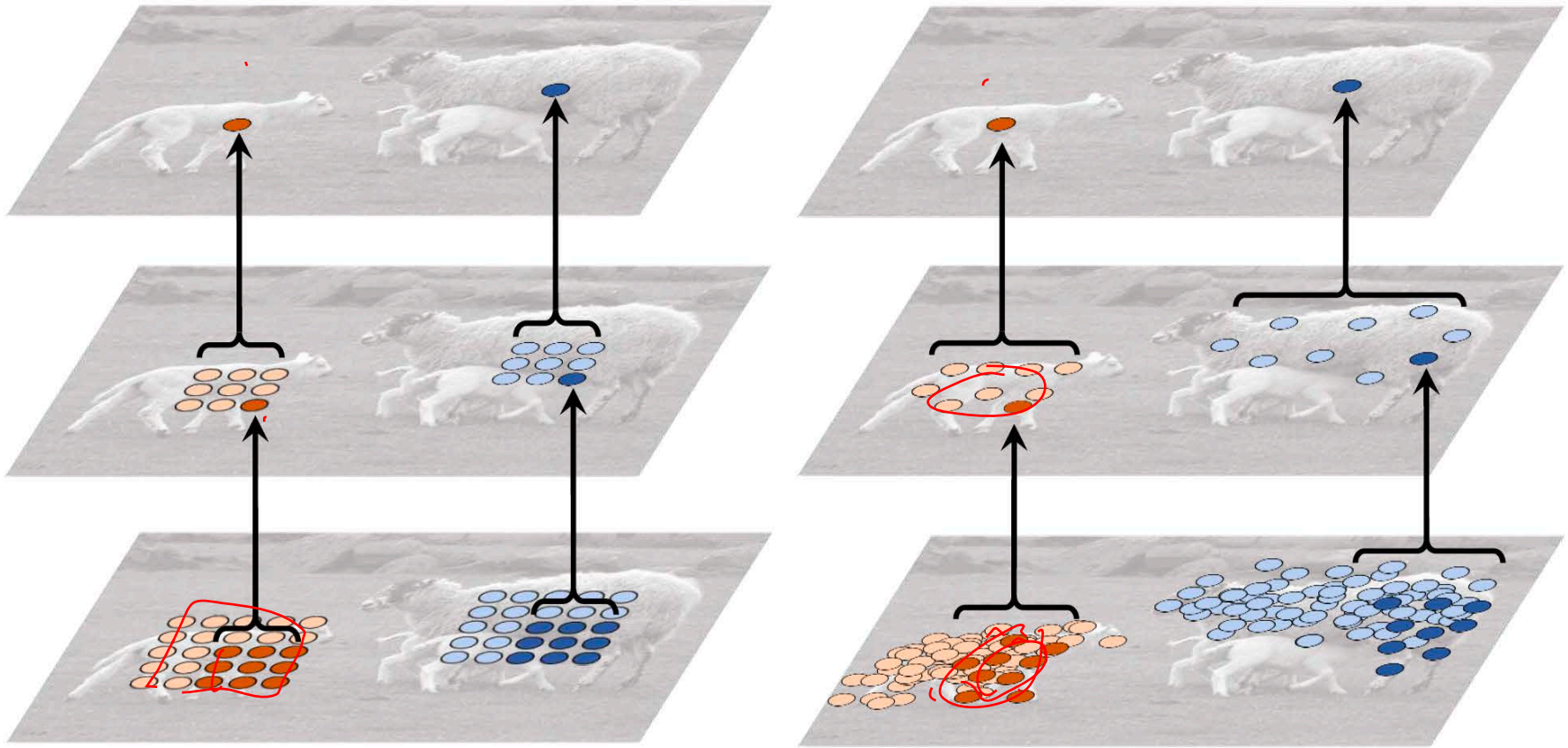
(a) regular sampling grid (green points) of standard convolution. (b) deformed sampling locations (dark blue points) with augmented offsets (light blue arrows) in deformable convolution. (c)(d) are special cases of (b), showing that the deformable convolution generalizes various transformations for scale, (anisotropic) aspect ratio and rotation.

Deformable convolutions: generalization of dilated convolutions when you learn the offset

Deformable convolutions



Deformable convolutions



(a) standard convolution (b) deformable convolution

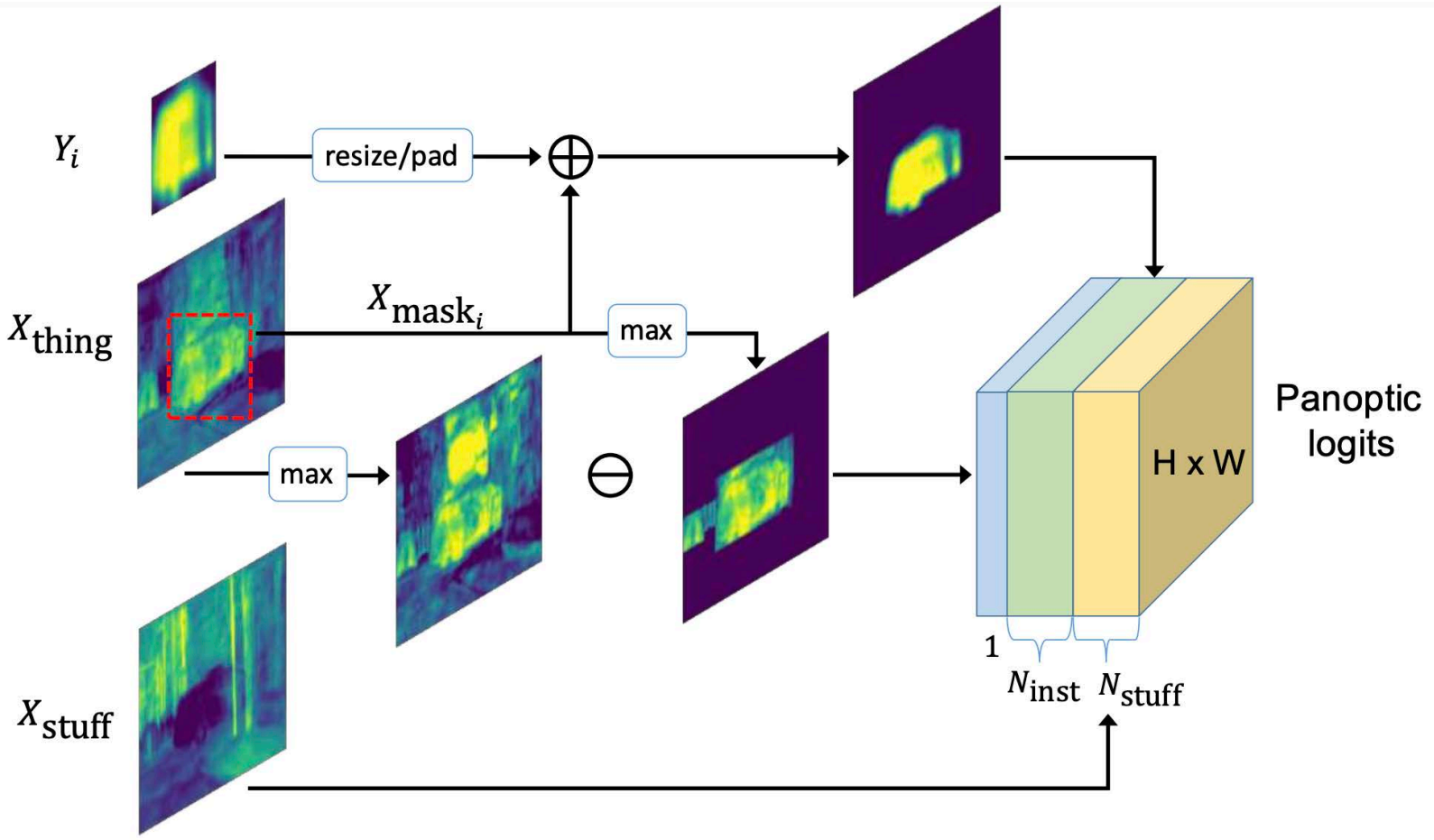
The deformable convolution will pick the values at different locations for convolutions conditioned on the input image of the feature maps.

The Panoptic head

Mask logits from the instance head

Object logits coming from the semantic head (e.g., car)

Stuff logits coming from the semantic head (e.g., sky)

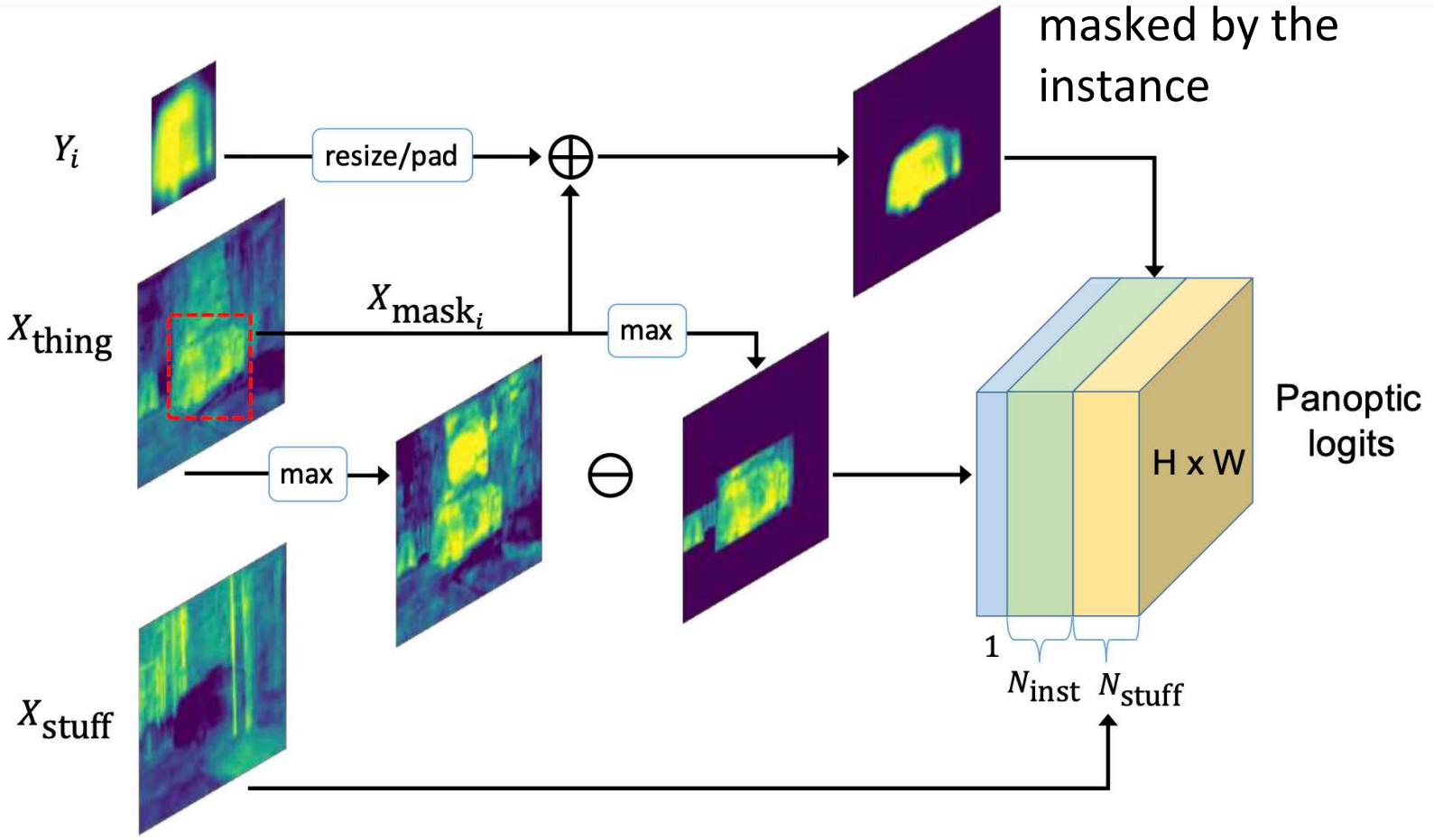


The Panoptic head

Mask logits from the instance head

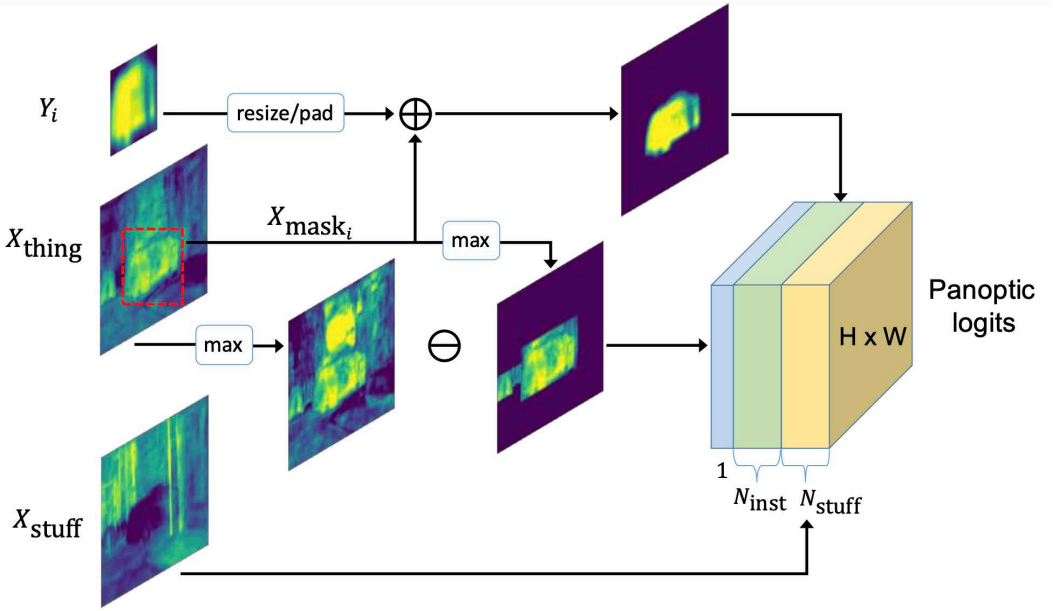
Object logits coming from the semantic head (e.g., car)

Stuff logits coming from the semantic head (e.g., sky)



evaluated directly

The Panoptic head



Perform softmax over the panoptic logits. If the maximum value falls into the first stuff channels, then it belongs to one of the stuff classes. Otherwise the index of the maximum value tells us the instance ID the pixel belongs to.

Read the details on how to use the unknown class (hint: think about the exam…)

Panoptic segmentation: qualitative

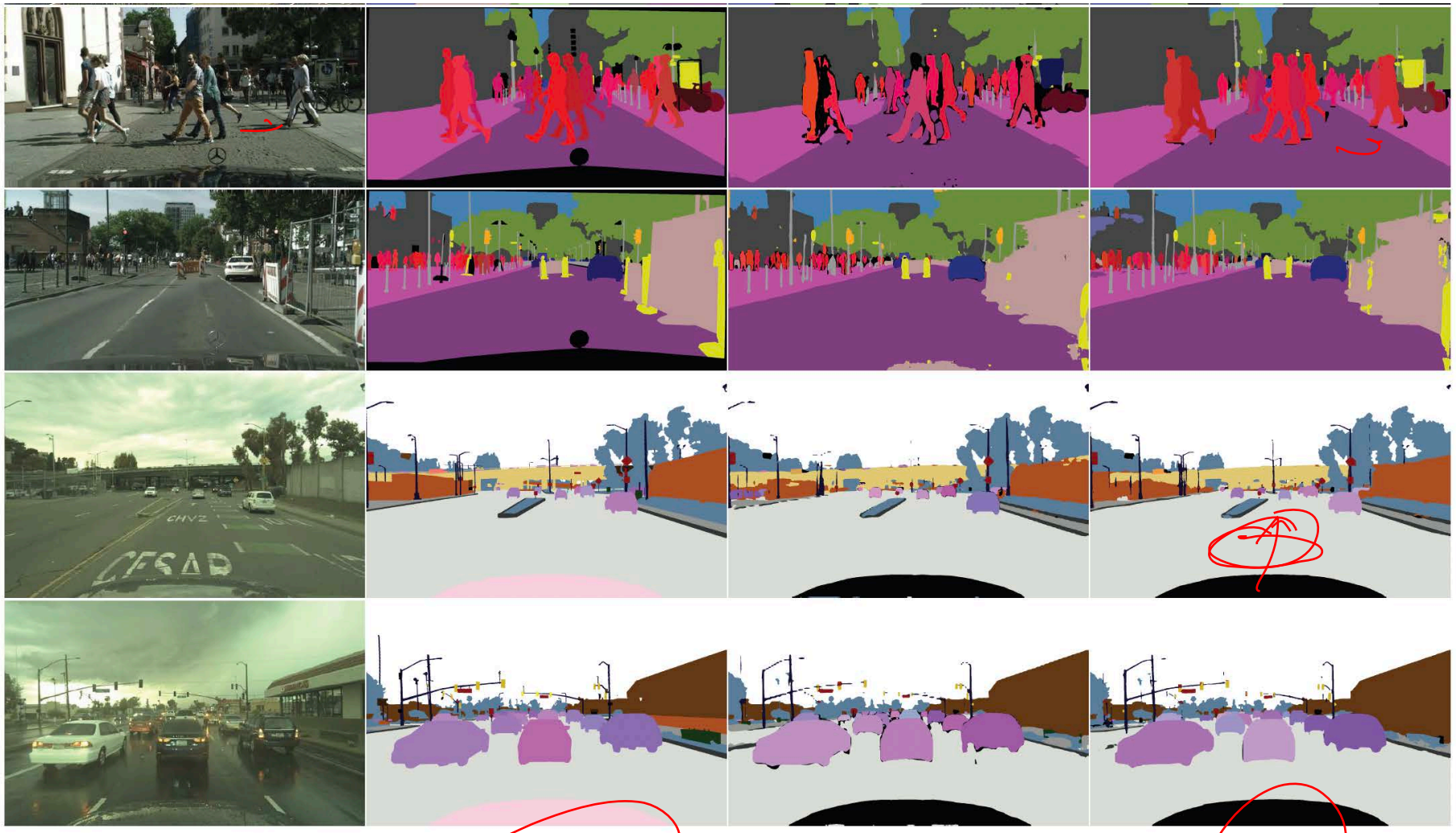


Image | Ground truth | Combined method [17] | Ours