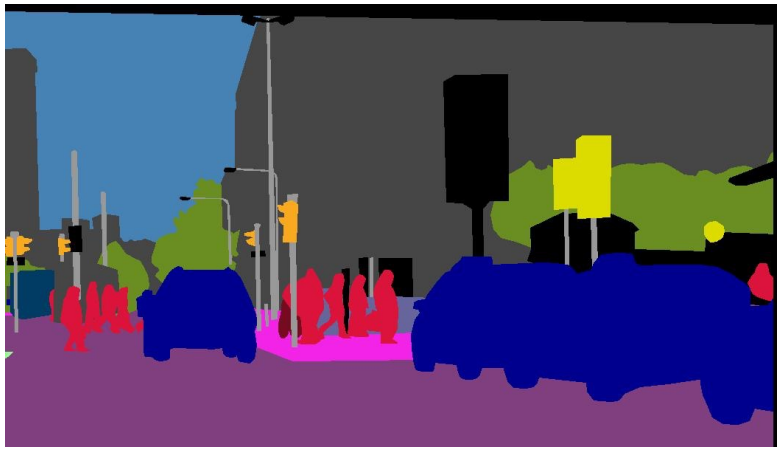
Image Segmentation

IN5400 Spring 2022

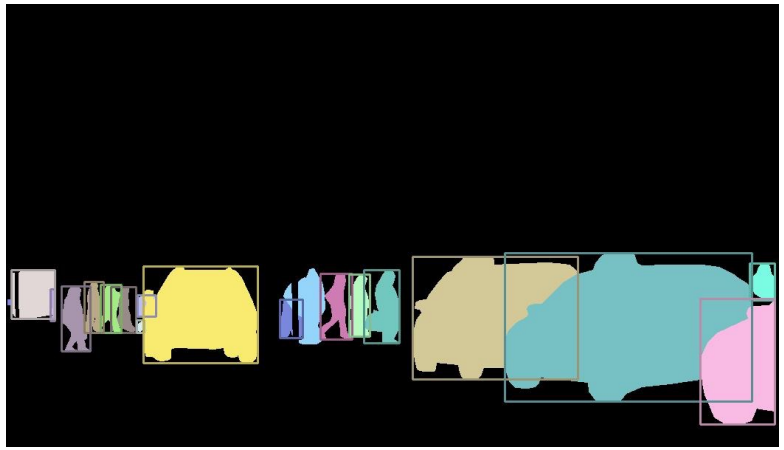
Image Segmentation



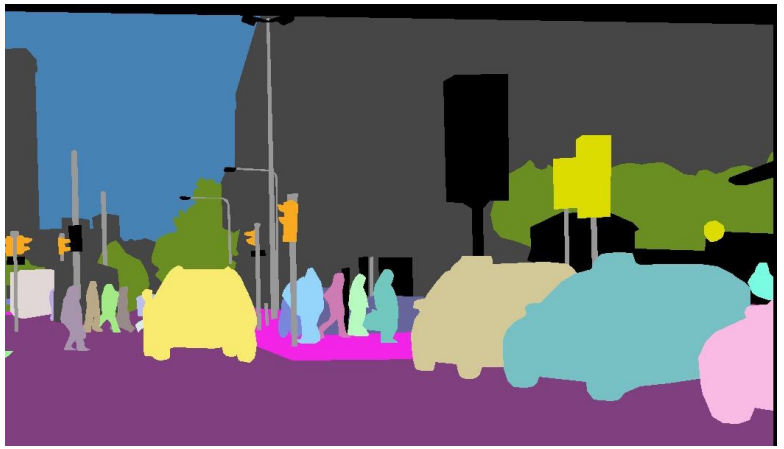
Image to be segmented



Semantic Segmentation



Instance Segmentation



Panoptic Segmentation

Semantic Segmentation:
- Object classes
- Background classes

Instance Segmentation:
- Object classes
- Object instances

Panoptic Segmentation:
- Semantic segmentation *AND* Instance segmentation

Images' source: "Panoptic Segmentation" Kirillov, A., et al.

Semantic Segmentation

- Classify every pixel in an image
- Differentiate between classes
- Do not differentiate between multiple instances of the same class



Source: ICNet, Zhao *et al.* 2017: semantic segmentation demo video

Naïve approach #1: Sliding Window

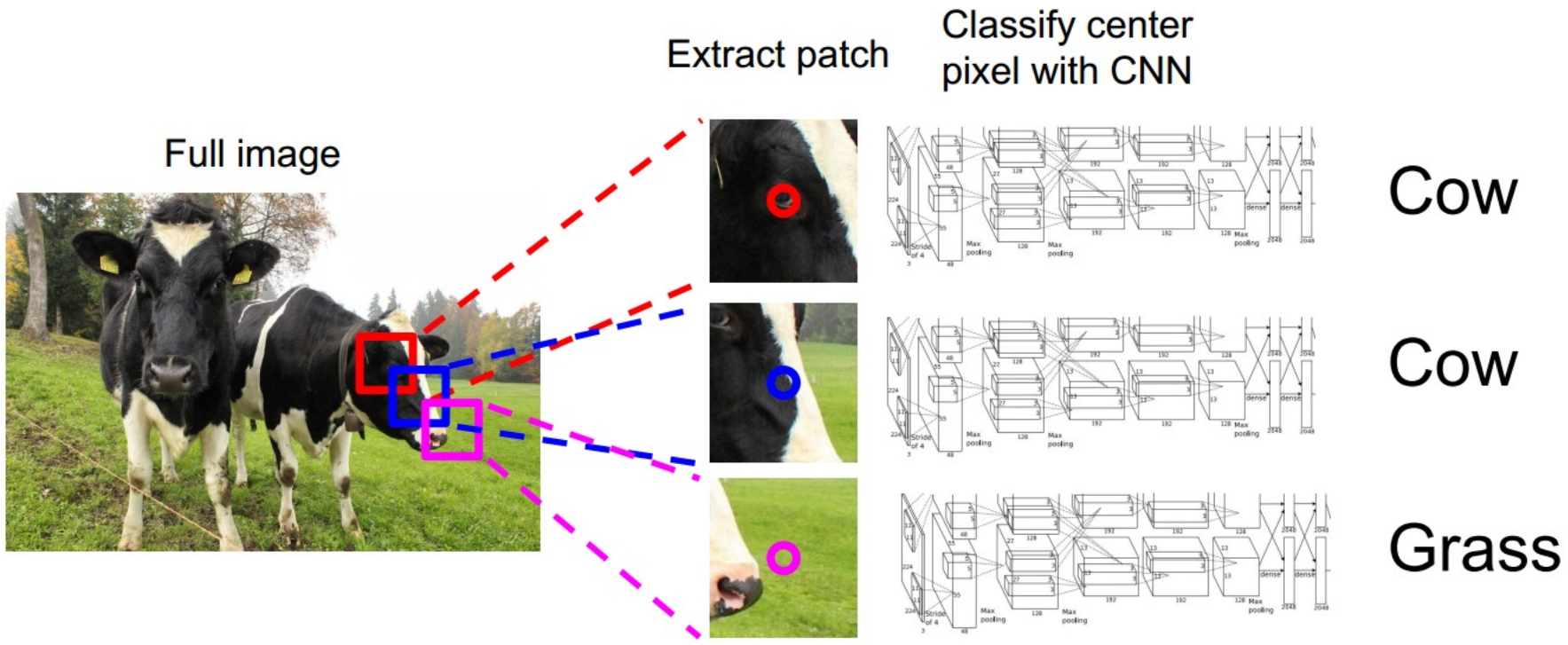


Image source: UMich EECS 498-007 / 598-005: Deep Learning for Computer Vision slides

Problems: Very expensive!
Global context missing

- A paper using this approach "Deep neural networks segment neuronal membranes in electron microscopy images." Ciresan, Dan, et al. 2012

Naïve approach #2: CNN without downsampling

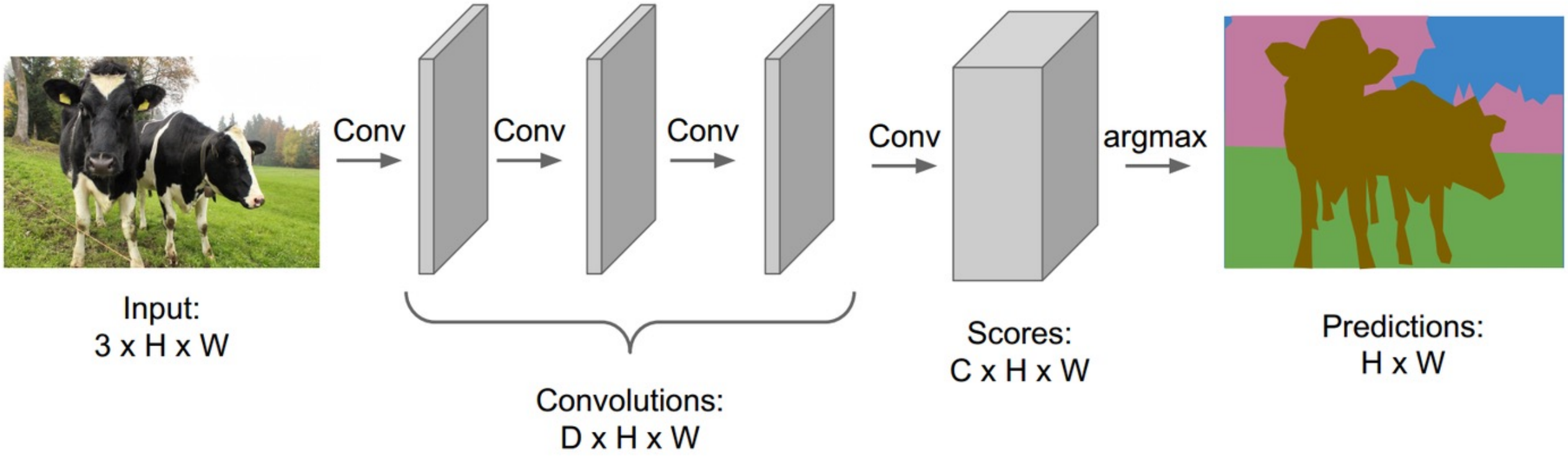


Image source: UMich EECS 498-007 / 598-005: Deep Learning for Computer Vision slides

Some Problems: Small receptive field, Expensive for large images, Can’t use existing pre-trained networks

Basic sketch for semantic segmentation

- Have large receptive fields (using downsampling for instance) to capture large spatial areas for global context.
  - But feature maps with large receptive fields have low resolution.
  - Find a way to go from low resolution to image-level resolution.
- Use information from early low resolution layers to capture finer details (boundary of the segmentation mask).

Problem #1: How to capture global context?

Global context: Information from the entire image

We will look at two common solutions.

Solution #1: (you already know this)

- Downsample feature maps using max/avg pooling or convolution with stride > 1.

Solution #2: (discussed later)

- Use “dilated” convolution. Also called atrous convolution (from French à trous).

Solution #1: A simple architecture

- Encode the input to a vector using only convolutions
- “Upsample” the encoded vector to image-size resolution.

Note:

- This architecture makes a big jump in the end from low resolution to high resolution in a single step.
- It might be better to do it gradually using more parameters (next slide).

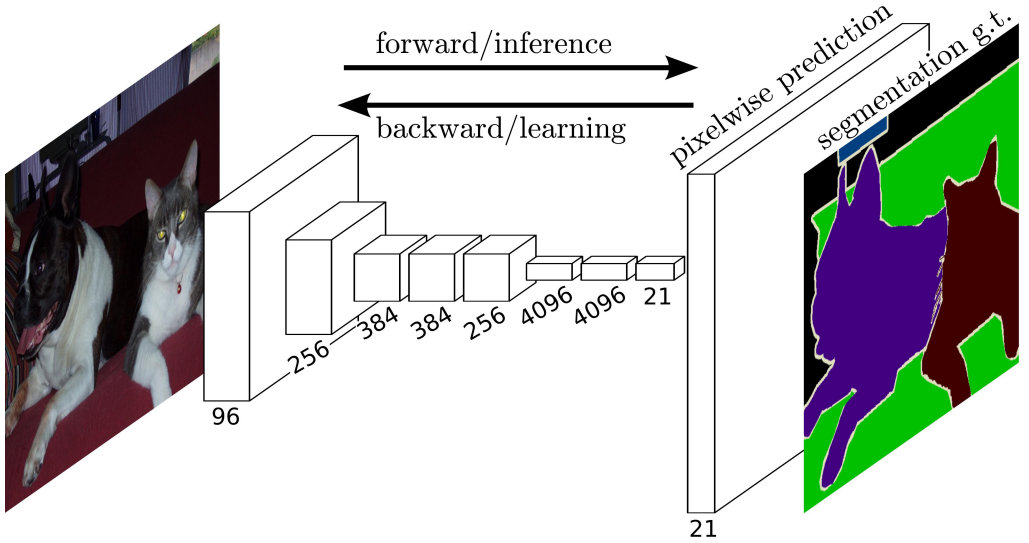


Image source: “Fully Convolutional Networks for Semantic Segmentation” Long, Jonathan, et al.

Solution #1: Encoder-Decoder architecture

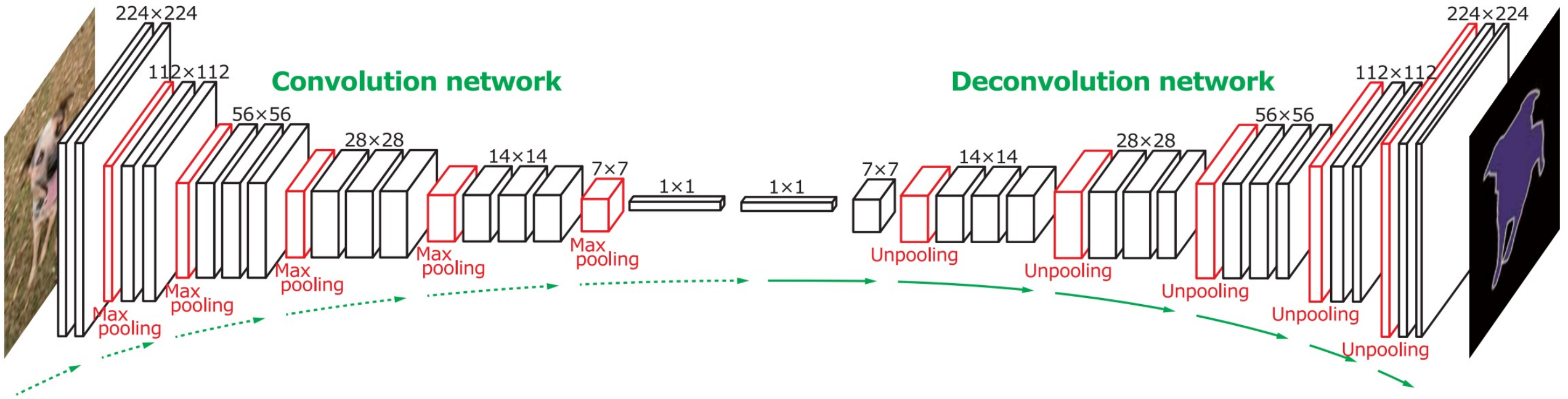


Image source: "Learning Deconvolution Network for Semantic Segmentation" Noh, Hyeonwoo, et al.

- But what is this "unpooling" and "deconvolution" you speak of?

Problem #2: How do we upsample features?

- Downsampling is part of all CNN architectures.
  - Helps cover large spatial areas of image.
  - Deeper low resolution maps tell us “what” is in the image.
- But image segmentation = pixel-level labels.
- How do we get back to image-level resolution?
- There are many solutions:
  - Nearest neighbour, bilinear, bicubic upsampling techniques
  - Unpooling
  - Transposed convolution OR deconvolution

Nearest neighbour upsampling

1	2
3	4

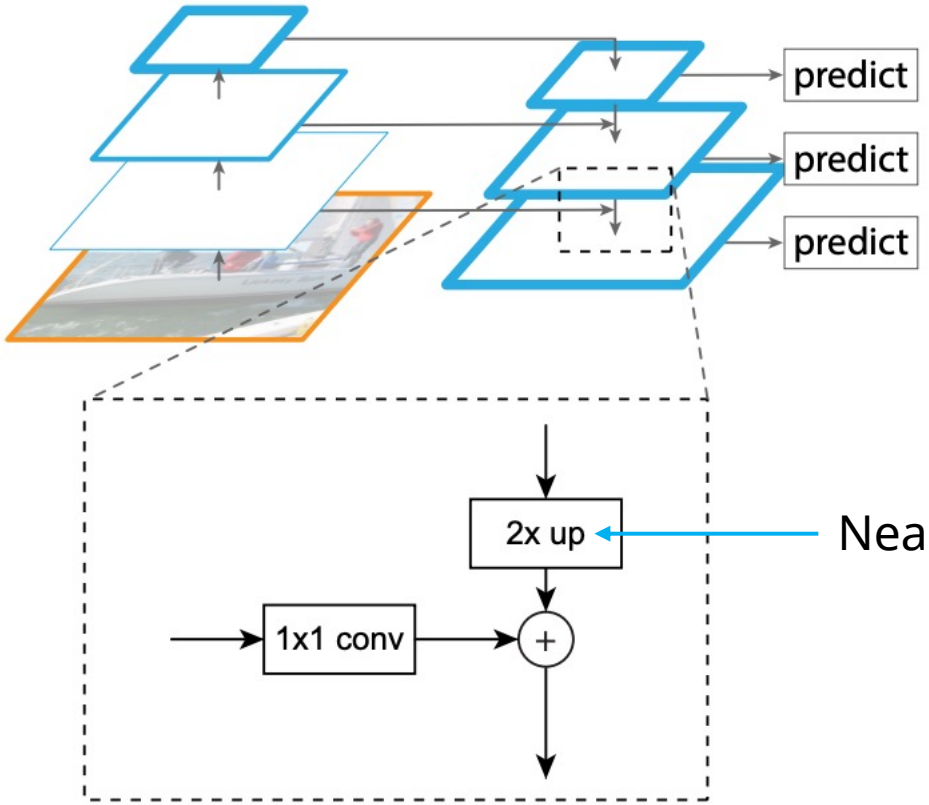


1	1	1	2	2	2
1	1	1	2	2	2
1	1	1	2	2	2
3	3	3	4	4	4
3	3	3	4	4	4
3	3	3	4	4	4

Input: C x 2 x 2

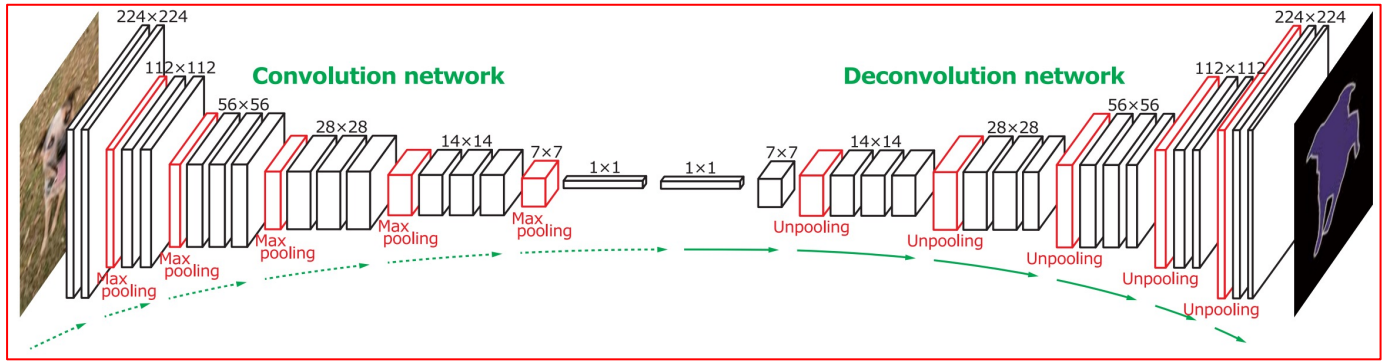
Output: C x 6 x 6

Nearest neighbour upsampling in FPN



Nearest neighbour upsampling

Unpooling



Rest of the network

Store the indices while max-pooling

5	2	4	6
3	7	2	5
8	9	4	2
7	2	3	5

7	6
9	5

Max pooling result

2	1
3	7

Feature map to unpool

(1,1)	(0, 3)
(2, 1)	(3, 3)

Max pooling indices

0	0	0	1
0	2	0	0
0	3	0	2
0	0	0	7

Unpooled output

Cf. Slide 9 or the next slide

Unpooling in SegNet

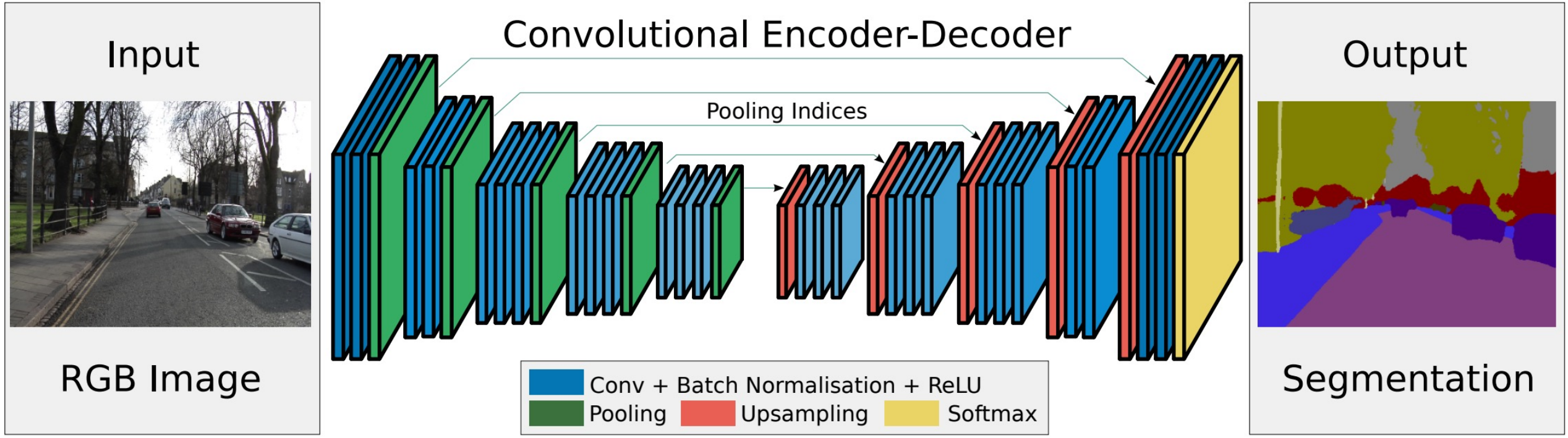


Image source: “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation” Badrinarayanan, V., et al.

Transposed convolution

- Learns parameters (of kernel K) to enlarge the input.
- Stride(*s*) and padding(*p*) are defined such that
  - Result of [applying K with (*s* and *p*) on the output] has the same shape as the input
  - i.e., if you take the output and apply a standard convolution with the given s and p, you would something that has same dimensions as the input.
- Also (incorrectly) called “deconvolution”.
- **`nn.ConvTranspose2d`** in PyTorch.

Transposed convolution

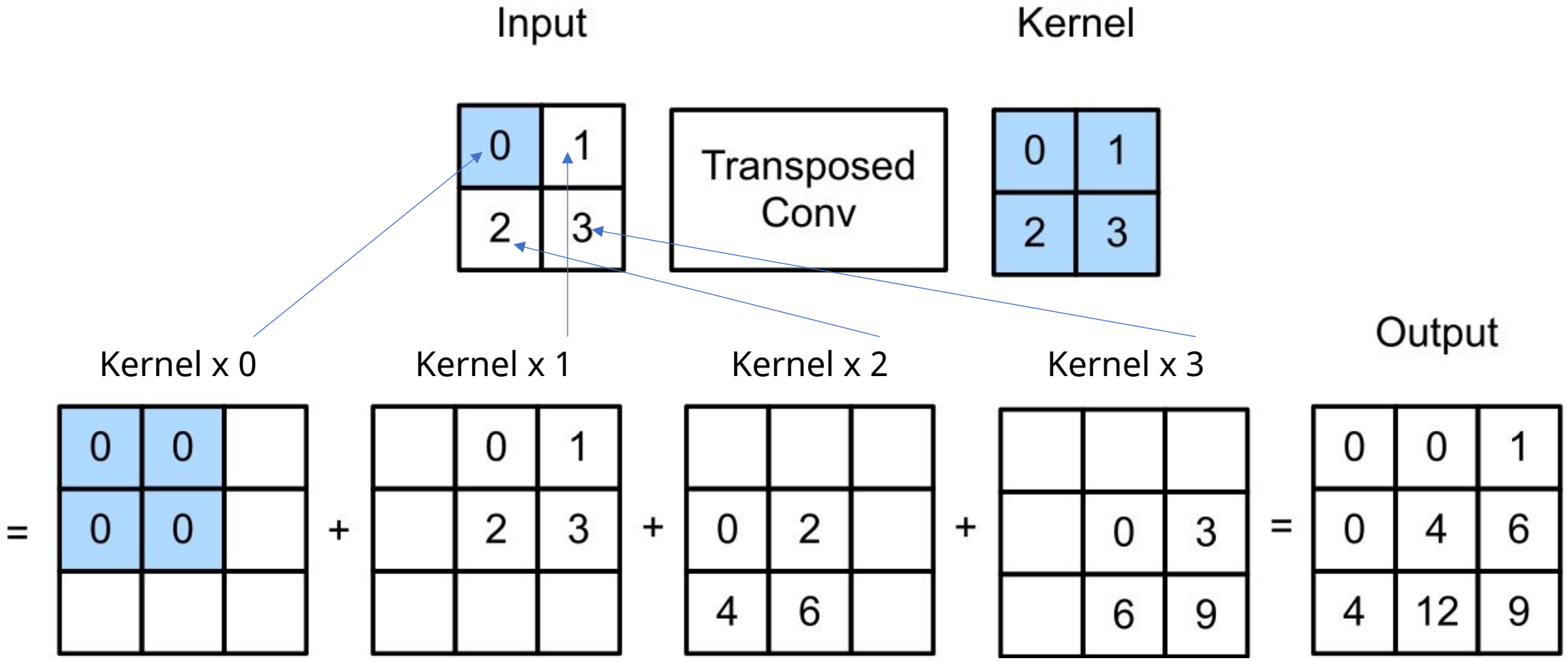


Image source: d2l.ai chapter on transposed convolution

Transposed convolution – stride 2

Stride is applied on the intermediate results

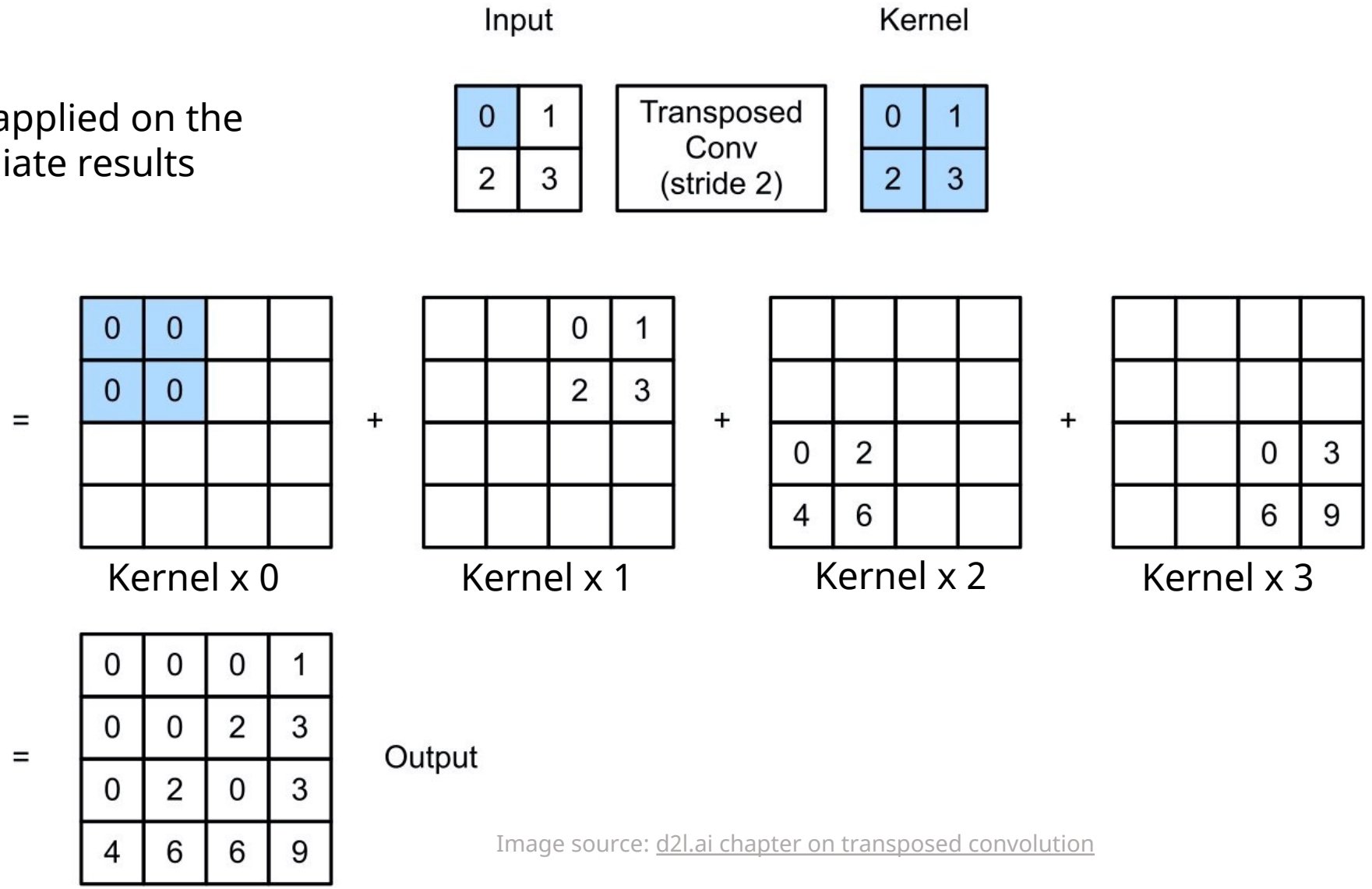


Image source: d2l.ai chapter on transposed convolution

Problem #3: How to fetch precise boundary locations?

- Earlier high resolution layers have more of the “**where**” information and deeper low resolution layers have more of the “**what**” information.
- Decoder could benefit from information in the high-resolution layers. They could help improve finer details.
- We’ve already looked at the solution in the last lecture when we looked at the Feature Pyramid Network (FPN).

U-net

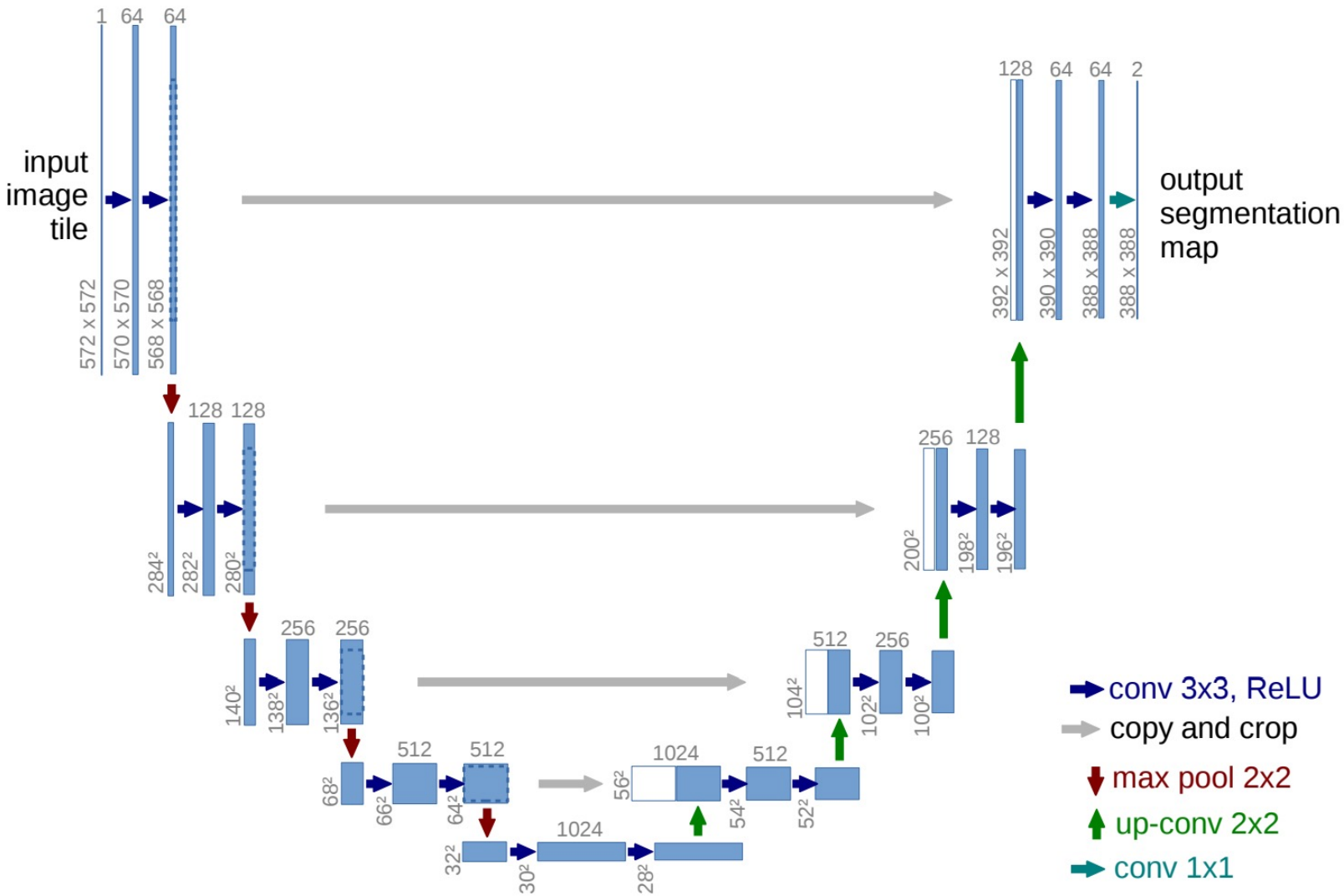


Image source: "U-Net: Convolutional Networks for Biomedical Image Segmentation" Ronneberger, O., et al.

U-net

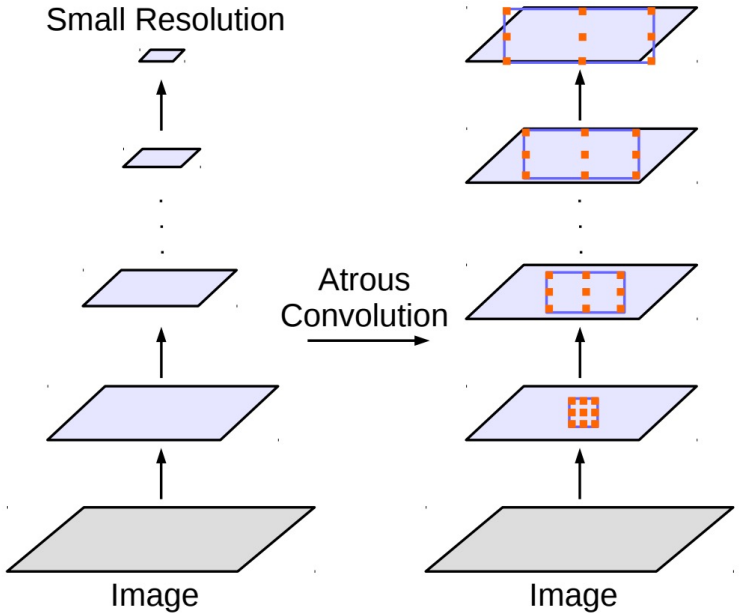
- Originally developed for medical image segmentation.
- Very popular architecture / often used as a baseline.
- Simple yet powerful.
- Performs well on many different kinds of data. For example, Chargrid uses U-net to process documents.
- Has many extensions like UNet++, Attention U-Net and so on

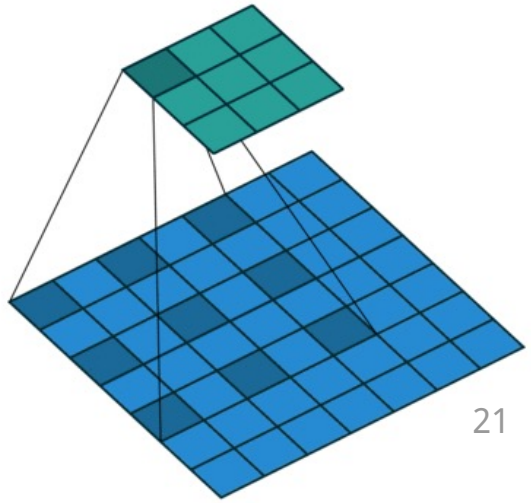
Dilated / Atrous convolution

$$y[i] = \sum_k x[i + r.k]w[k]$$

where **r** is the dilation rate (or simply rate or dilation), **x** is the input, **y** is the output and **w** is the kernel.

- Standard convolution is a special case of atrous convolution with r=1.
- Atrous convolution replaces downsampling layers. Double the dilation rate to replace the downsampling layer.
- The very popular (and close to the State of The Art) DeepLabv3+ (image on next slide) applies atrous convolutions with different rates in parallel on the last feature map and concatenates them before upsampling and making pixel-level predictions.





DeepLabv3+

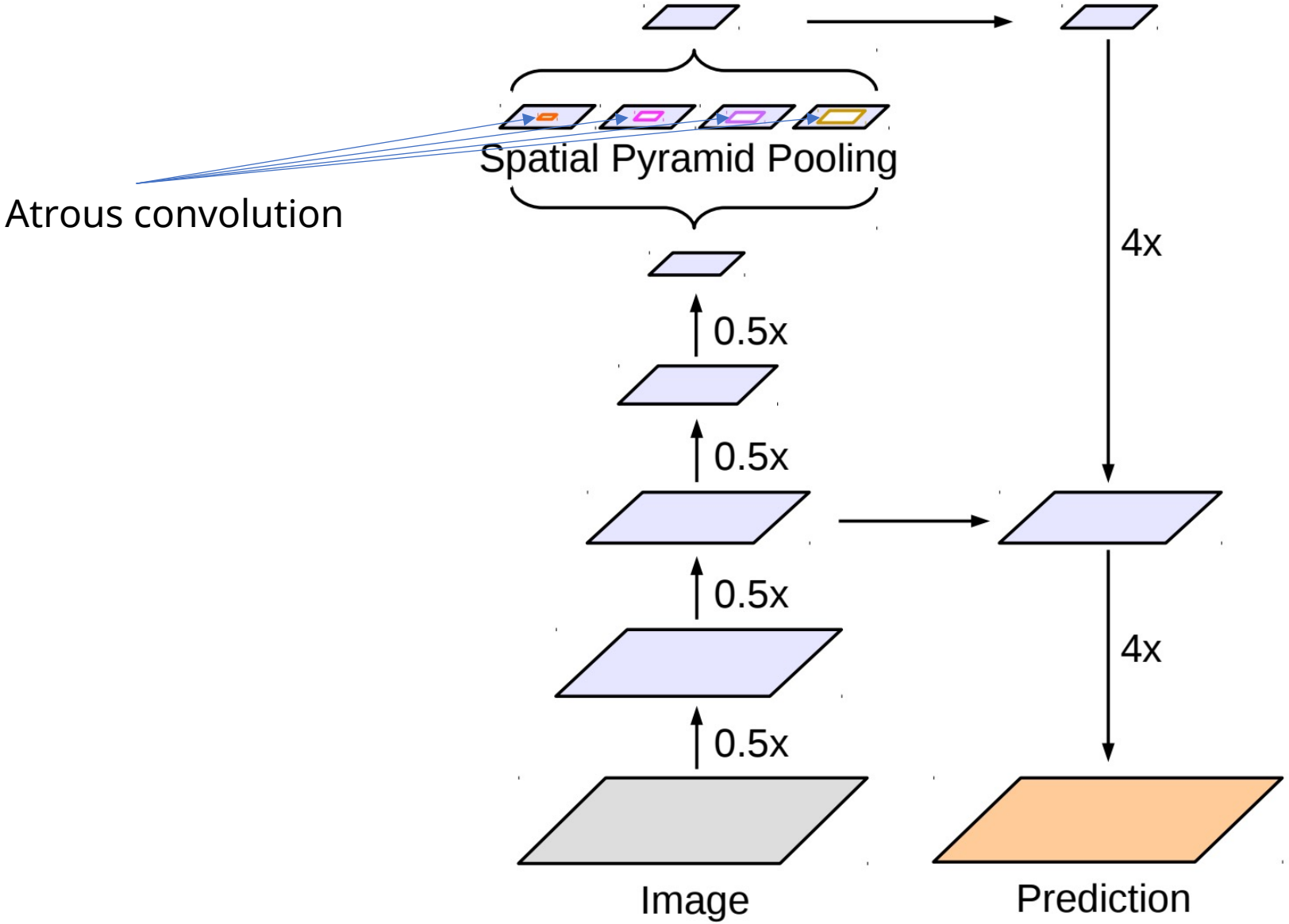
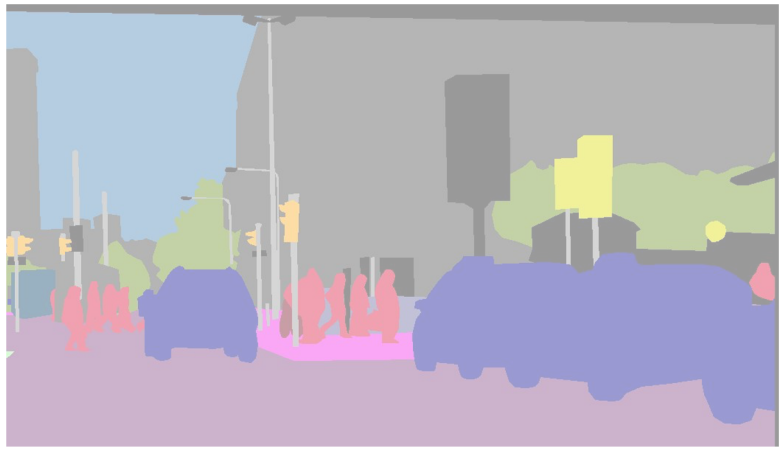


Image source: DeepLabv3+ paper

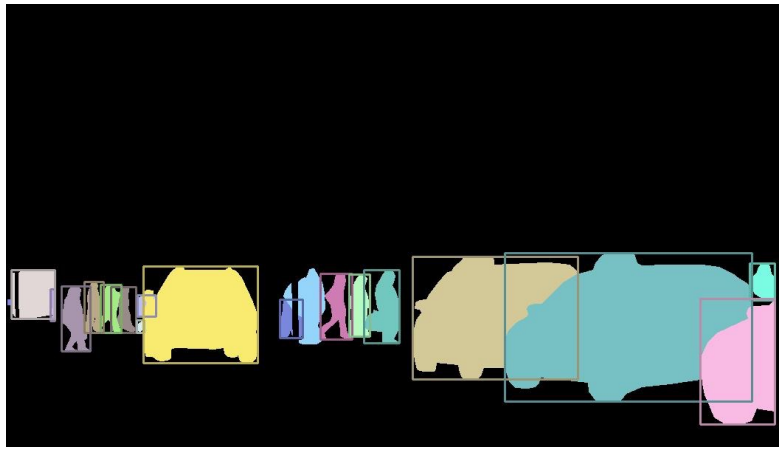
Image Segmentation



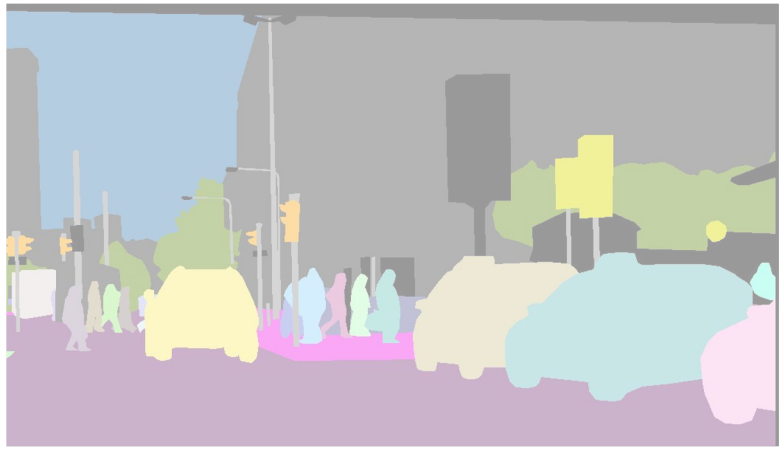
Image to be segmented



Semantic Segmentation



Instance Segmentation



Panoptic Segmentation

Semantic Segmentation:
- Object classes
- Background classes

Instance Segmentation:
- Object classes
- Object instances

Panoptic Segmentation:
- Semantic segmentation *AND* Instance segmentation

Images' source: "Panoptic Segmentation" Kirillov, A., et al.

Instance Segmentation

- Only label pixels belonging to objects.
- Different label for different instances of the same class.
- Common solution:
  - Perform object detection
  - Generate a mask for each object in parallel
  - Mask R-CNN: based on Faster R-CNN
    - Check our previous lecture

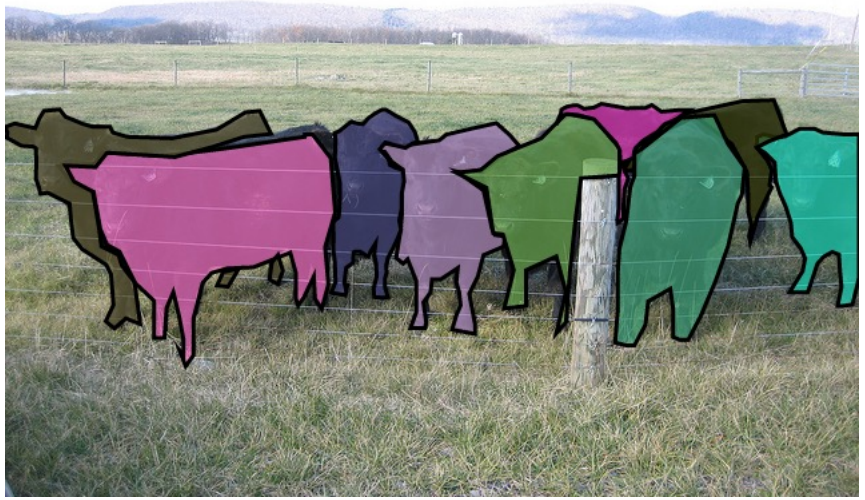


Image source: COCO dataset

Mask R-CNN

- A commonly used framework.
- Works well in practice.
- Extends Faster R-CNN by predicting mask in parallel to box offset and class.
  - A mask is predicted for each class for each ROI.
- Has many extensions as usual. For e.g., Cascade Mask R-CNN

Mask R-CNN

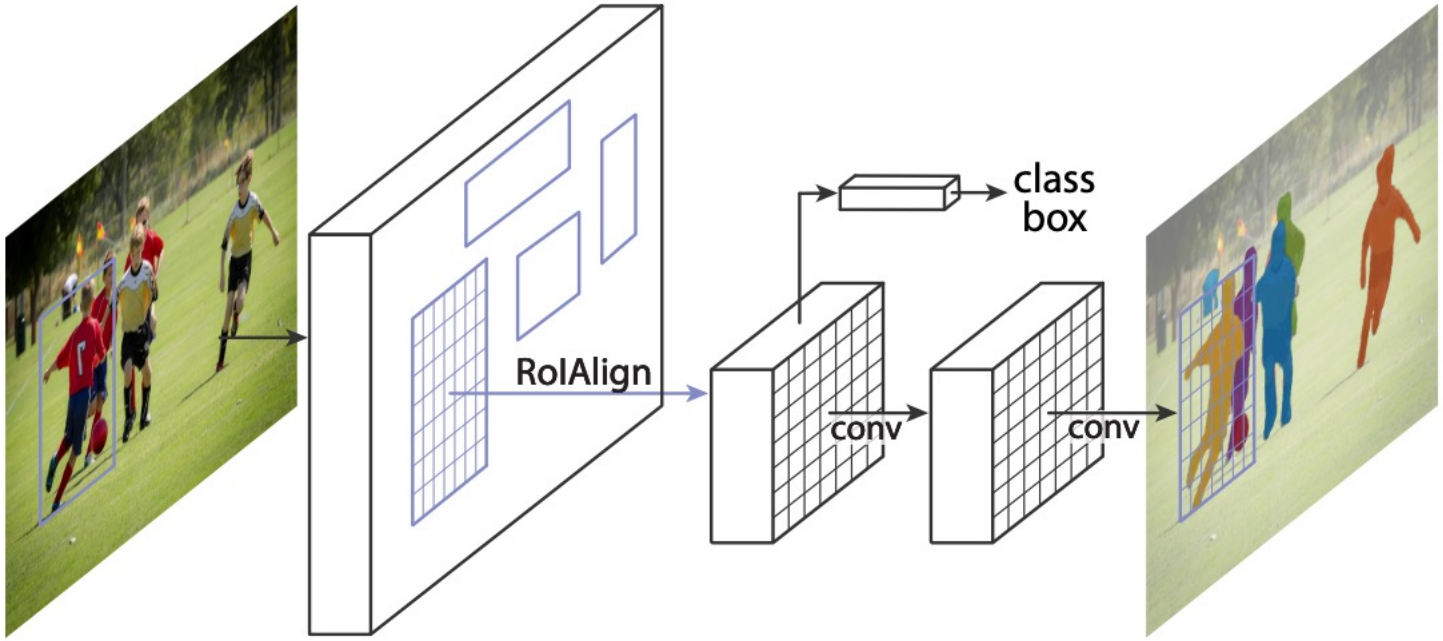


Image source: ”Mask R-CNN”

Main performance metrics

- **Mean IoU**
  - **THE** most popular metric that everyone reports on.
  - It is defined as the average IoU over all classes.
  - Mainly used for semantic segmentation
- **Mask mAP**
  - mAP from the previous lecture but IoU is calculated for mask (and not box).
  - Mainly used for instance segmentation

Other performance metrics

- **Pixel accuracy**
  - Fraction of pixels correctly classified.
- **Mean pixel accuracy**
  - Pixel accuracy averaged over classes.
- **Dice coefficient**
  - Dice = $\frac{2|A \cap B|}{|A|+|B|}$
  - Similar to IoU, as is evident from the equation above.
  - Same as F1 score for binary segmentation (like foreground vs. background).
  - More popular in medical image segmentation domain. Used by UNet++ as loss.
- Many other metrics exist

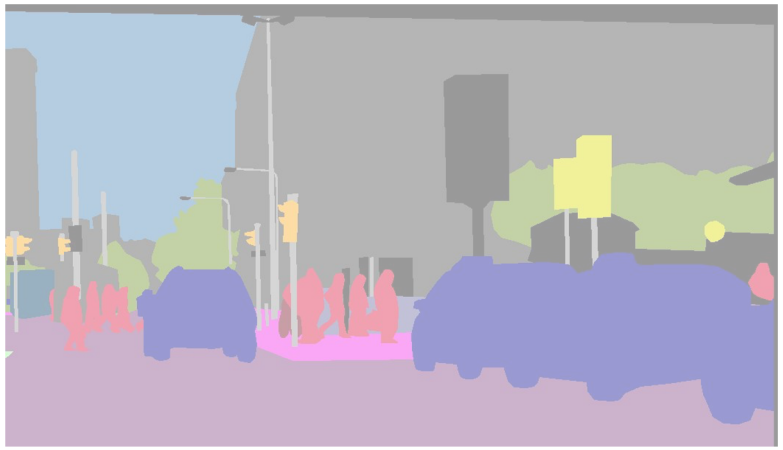
Basic loss functions

- Semantic segmentation
  - Calculate cross entropy for each pixel.
  - Sum the losses of all the pixels.
  - Balance for class if required.
- Instance segmentation
  - Since it’s typically trained jointly with object detection, the loss is the sum of all the three losses: classification loss, box regression loss, mask loss
  - The mask loss is calculated just like for semantic segmentation
- Loss functions are typically modified as per the need/problem.
- There are many application/dataset-specific loss functions out there.

Image Segmentation



Image to be segmented



Semantic Segmentation



Instance Segmentation



Panoptic Segmentation

Semantic Segmentation:
- Object classes
- Background classes

Instance Segmentation:
- Object classes
- Object instances

Panoptic Segmentation:
- Semantic segmentation *AND* Instance segmentation

Images' source: "Panoptic Segmentation" Kirillov, A., et al.

Panoptic segmentation

- Combines semantic segmentation and instance segmentation.
- Each pixel is labelled and different instances of an object class have different labels.
- Terminology:
  - **Thing**: an object in foreground
  - **Stuff**: background like grass, mountains, etc.
- Uses a new metric: Panoptic Quality (PQ).

Panoptic segmentation approaches

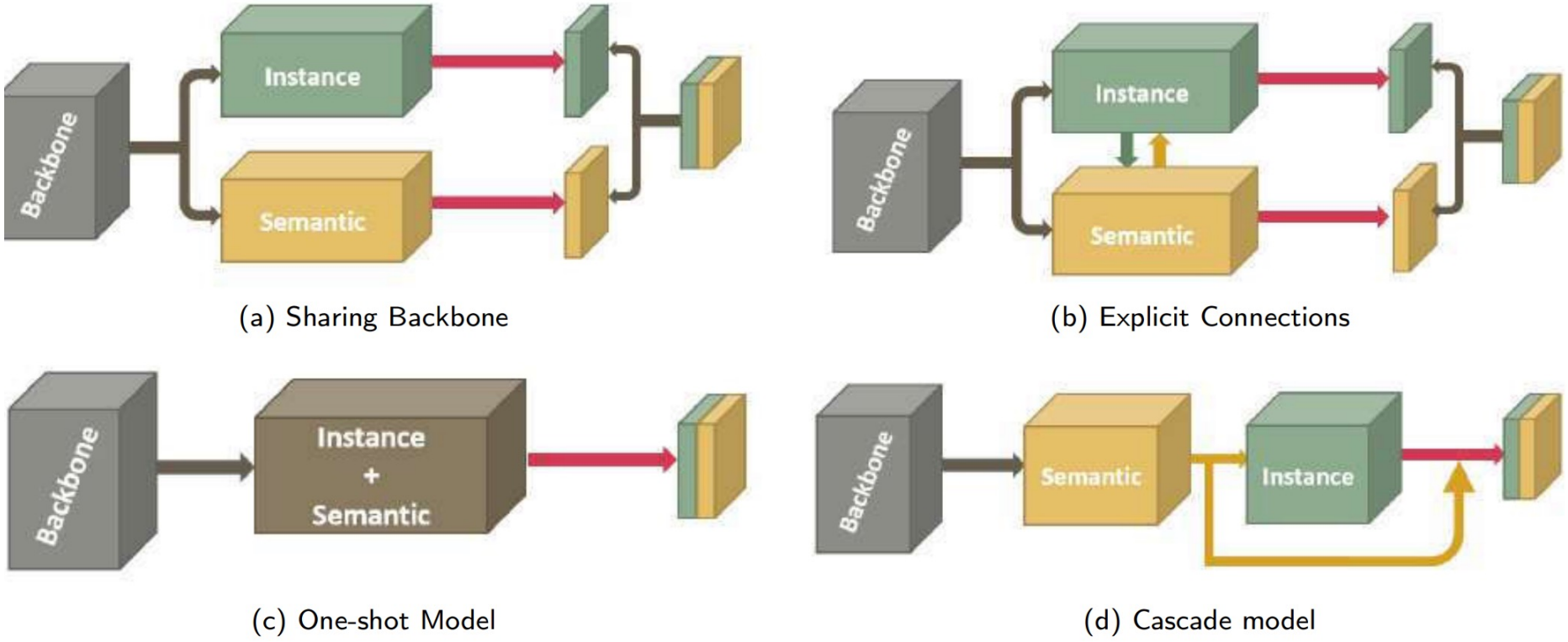


Image source: "Panoptic segmentation: A Review" Elharrouss, Omar, et al.

Panoptic DeepLab

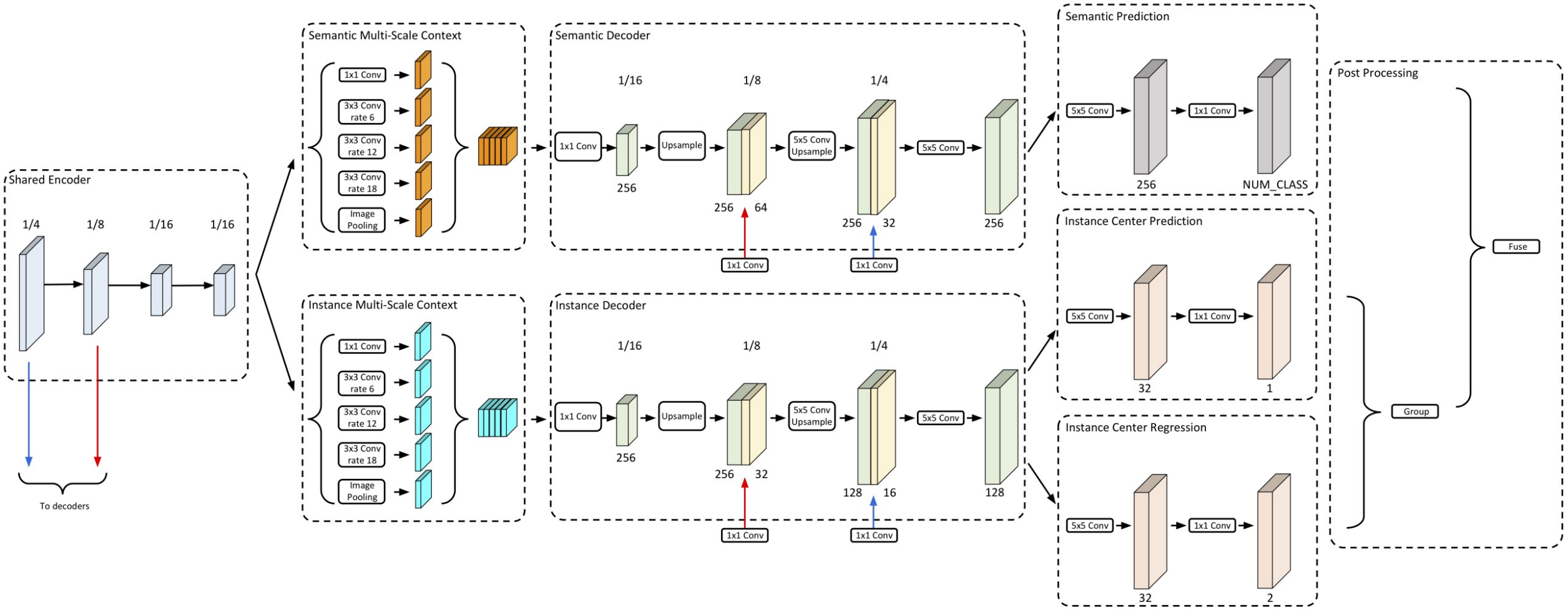


Image source: Panoptic DeepLab paper

Suggested resources

- 2nd half of this lecture from UMich is on image segmentation: https://www.youtube.com/watch?v=9AyMR4IhSWQ
- Transformers are now SOTA (state of the art) for many (if not all) image segmentation tasks on the popular datasets. If you'd like to get a basic idea about transformers then:
  - This lecture, again from UMich, introduces transformers towards the end after talking about attention mechanism: https://www.youtube.com/watch?v=YAgjfMR9R_M
  - The awesome blog post that everyone refers to: https://jalammar.github.io/illustrated-transformer/