CS195: Computer Vision

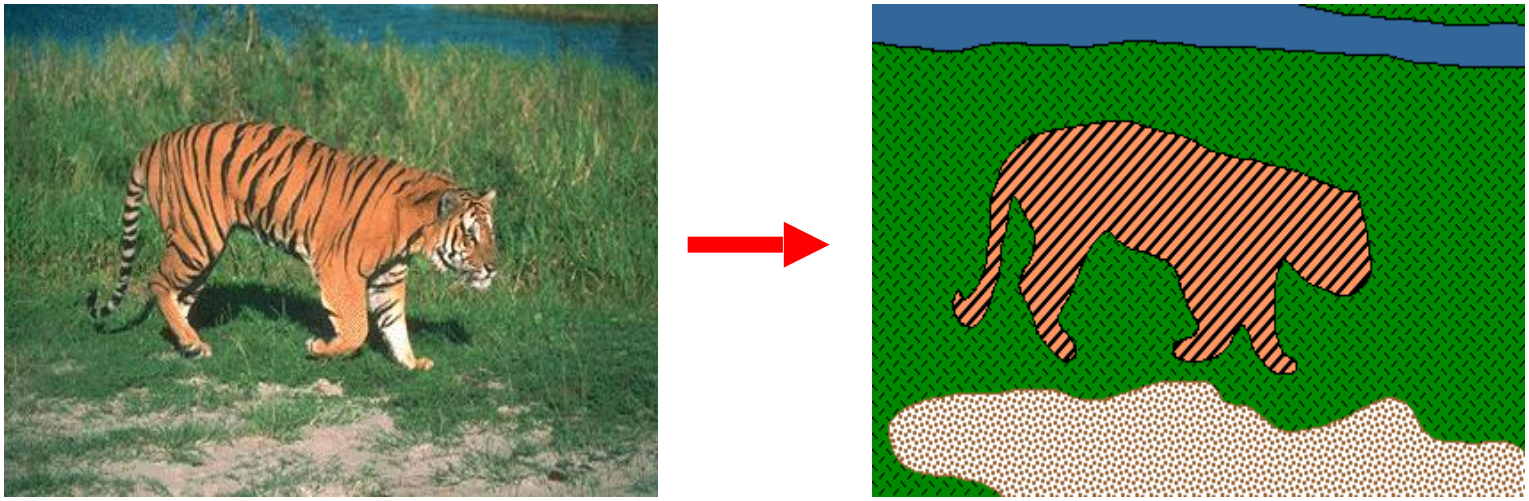
Semantic Segmentation

Wednesday, October 23rd, 2024

Drake UNIVERSITY

Image segmentation

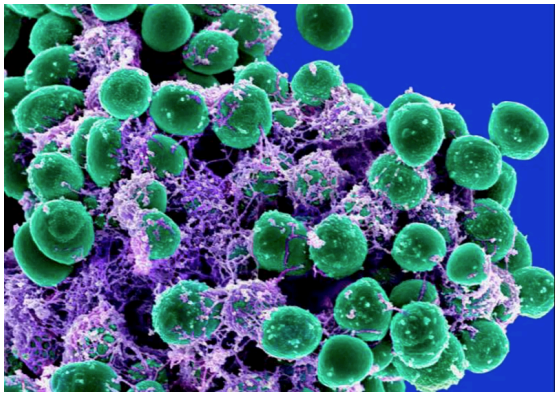
- Goal: break apart an image into simpler components



Segment Anything (SAM)

- Segment Anything (SAM) is a **foundation model** for segmentation

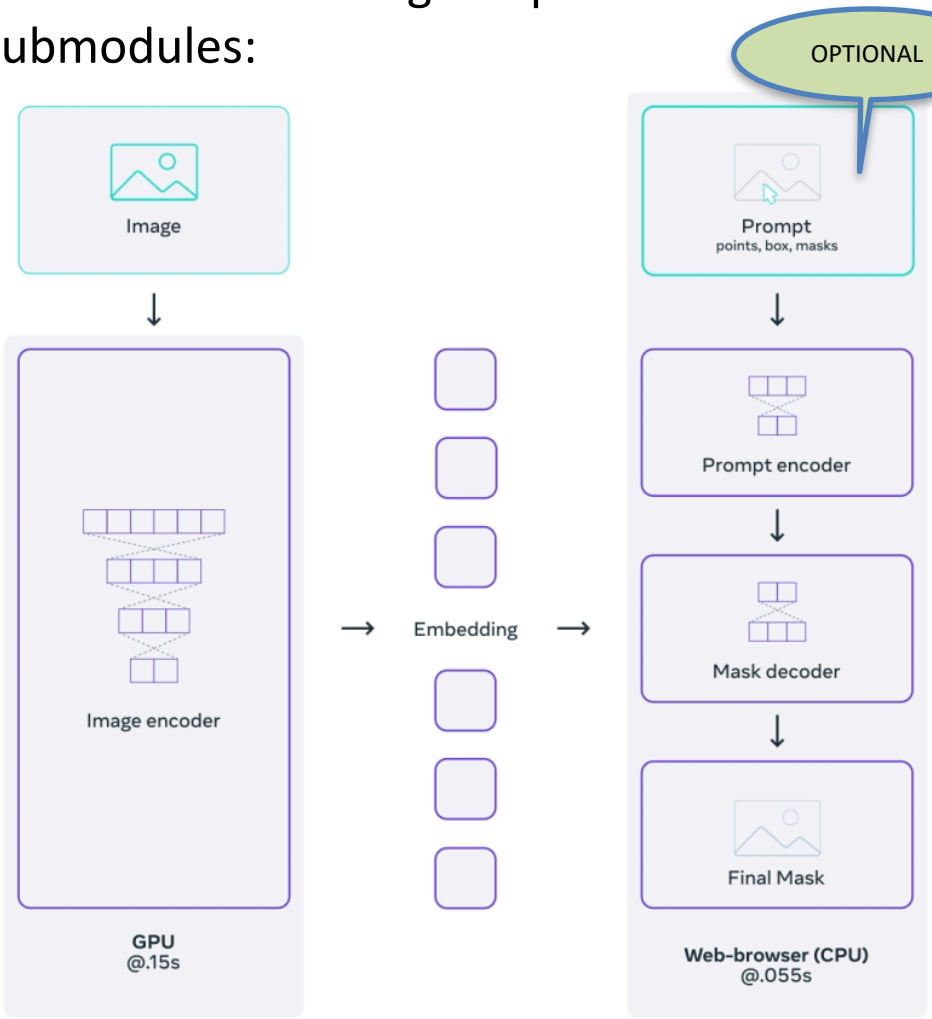
- It can segment unfamiliar objects and never-seen-before image without the need for additional need for training
    - zero-shot generalization property



Segment Anything - arxiv'23

Segment Anything (SAM)

- Segment Anything (SAM) is designed to be efficient enough to power its data engine. Model is decoupled into two submodules:
  - a one-time image encoder and
  - a lightweight mask decoder (that can run in a web-browser in just a few milliseconds per prompt)



- SAM is trained on more than 1.1 billion segmentation masks collected on 11 million images
  - authors used SAM and data to interactively annotate images and update the model. This cycle was repeated many times over to improve both the model and the dataset

Segment Anything - arxiv'23

Segment Anything (SAM)

- Segment Anything (SAM) can also segment images based on a variety of input prompts

Segment based on prompt with interactive points and boxes

Generate multiple valid masks for ambiguous prompts

Segment Anything - arxiv'23

Segment Anything (SAM)

- Segment Anything (SAM) can also segment images based on a variety of input prompts

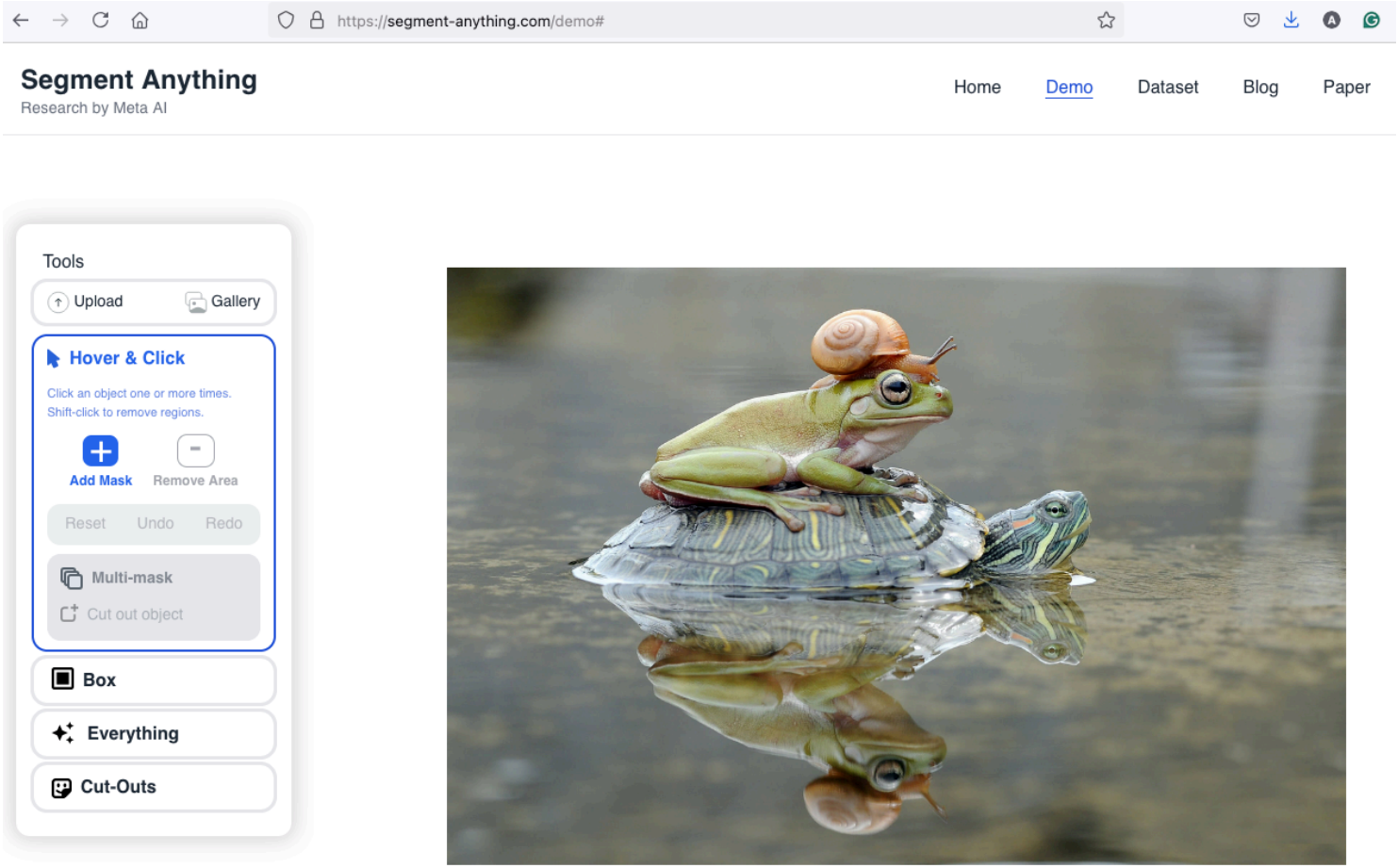
Cons: it may not work for a uncommon object such as sea-lion which remains In underwater environment

Bounding box prompts from an object detector can enable text-to-object segmentation (may work for popular objects)

Segment Anything - arxiv'23

Group Activity#1: Try Segment Anything Demo

- Try zero-shot generalization capability
- Try to segment images based on a variety of input prompts



https://segment-anything.com/demo#

Group Activity#2: Try Segment Anything Demo

- Try zero-shot generalization capability using PyTorch code

https://github.com/alimoorreza/cs195-fall24-notes/blob/main/cs195_segment_anything_SAM_i

cs195-fall24-notes / cs195_segment_anything_SAM_inference.ipynb

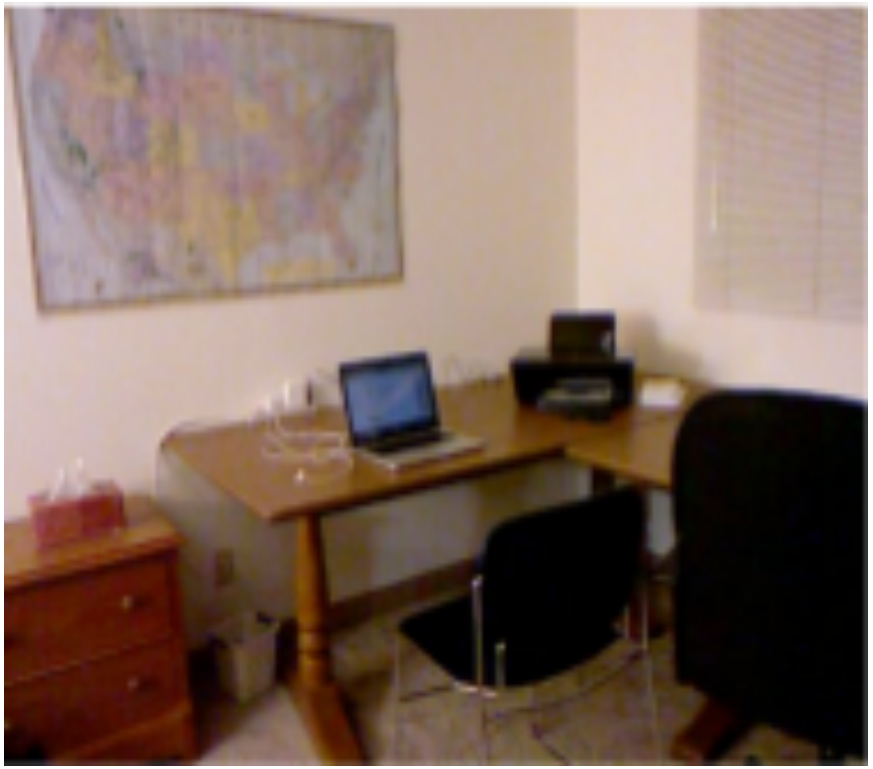
Blame 749 lines (749 loc) · 1.87 MB

```
cur_image = 'Crocodile_27.png'
#cur_image   = 'gridline.png'

image       = cv2.imread(root_dir + cur_image)
image       = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
masks2      = mask_generator_2.generate(image)

if is_display_enabled == True:
  plt.figure(figsize=(10,10))
  plt.imshow(image)
  show_anns(masks2)
  plt.axis('off')
  plt.show()


# ---- save the mask ----
# make a directory for each animal
cur_animal    = cur_image.split("/")[0]
dest_dir            = root_dir + "/"

# save the masks + image as pickle file
f1      = open(dest_dir + cur_image[0:-4] + '.pkl', 'wb')
my_dict = {'image': image, 'masks': masks2}
pickle.dump(my_dict, f1)
f1.close()
```

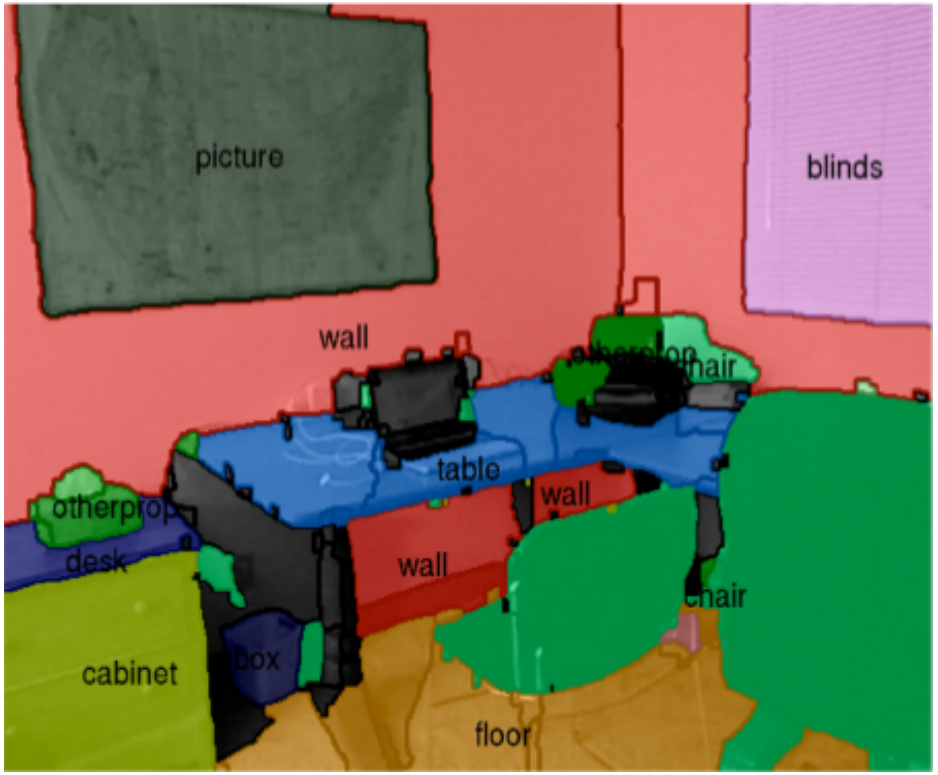
https://github.com/alimoorreza/cs195-fall24-notes/blob/main/cs195_segment_anything_SAM_inference.ipynb

Semantic Segmentation

- Semantic segmentation is the task of automatically labeling every pixel in an input image



RGB image



Corresponding semantic segmentation into various indoor object classes

Semantic Segmentation

- Semantic segmentation is the task of automatically labeling every pixel in an input image

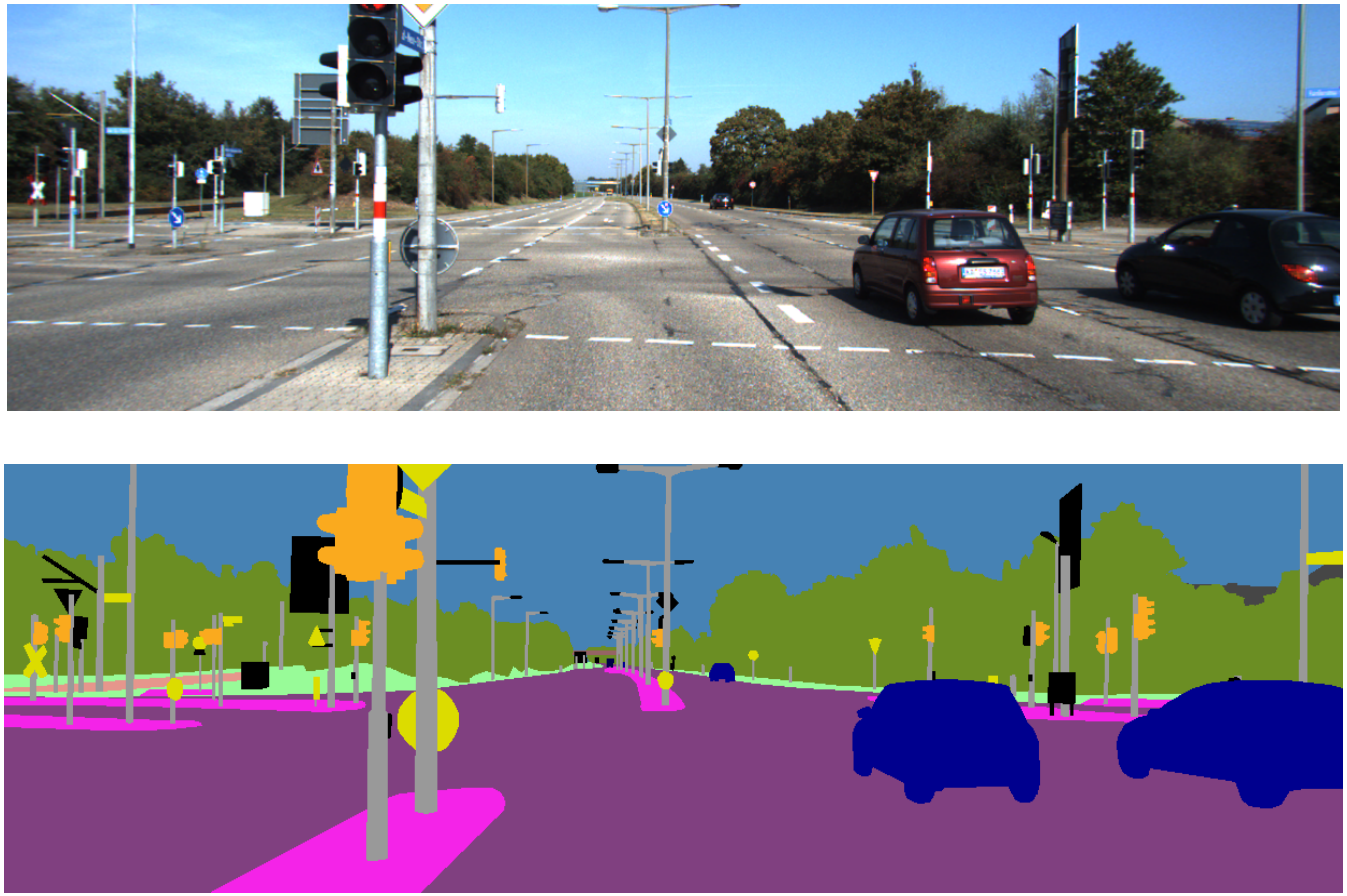


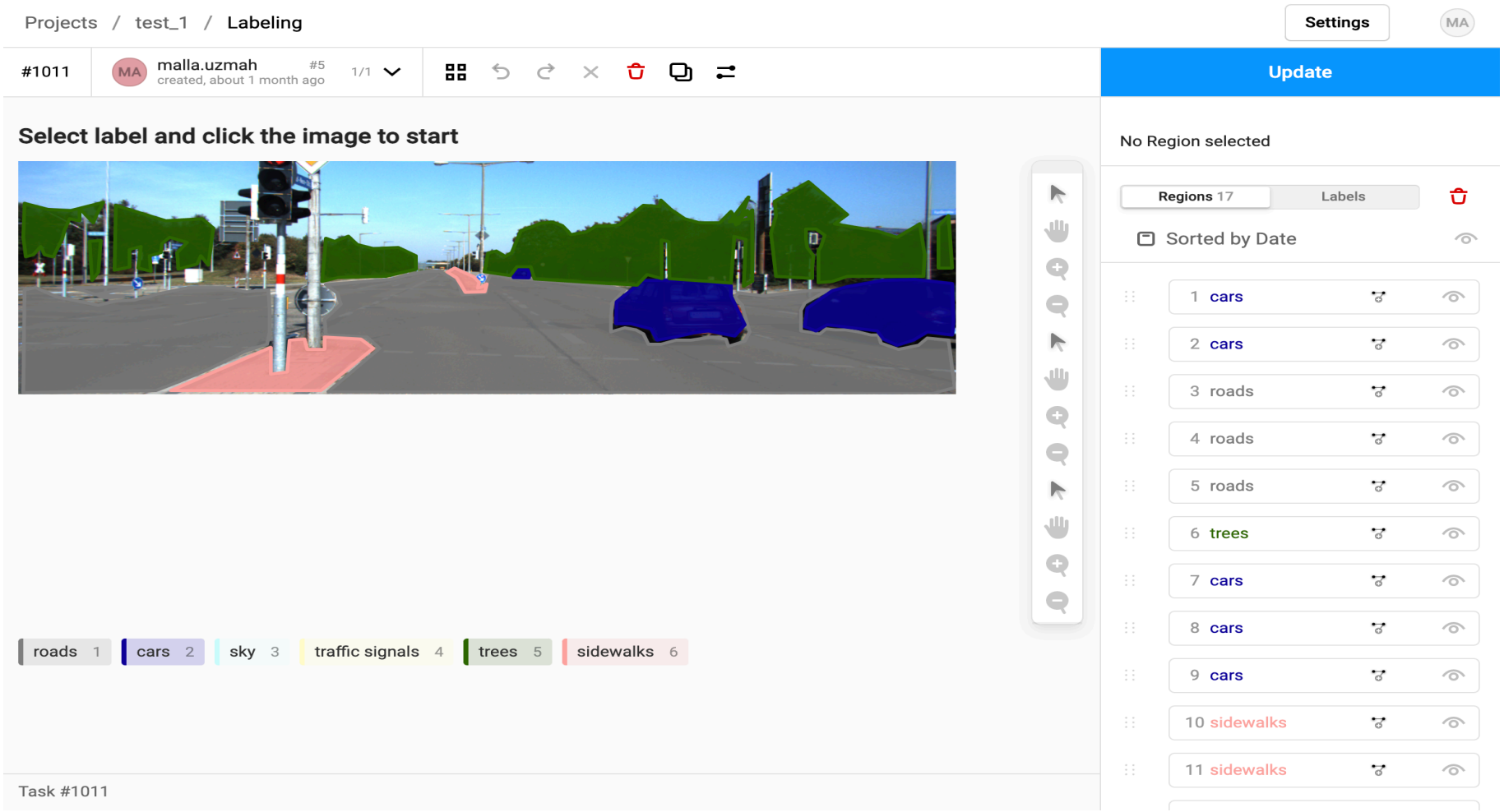
Figure: Corresponding semantic segmentation into various **outdoor** object classes

Semantic Segmentation

- How do you get the ground truth masks for semantic segmentation at the first place?

The masks for various **outdoor** object classes needs to be annotated first before you can teach a machine to learn to predict similar objects in the future

Annotation tool for Semantic Segmentation



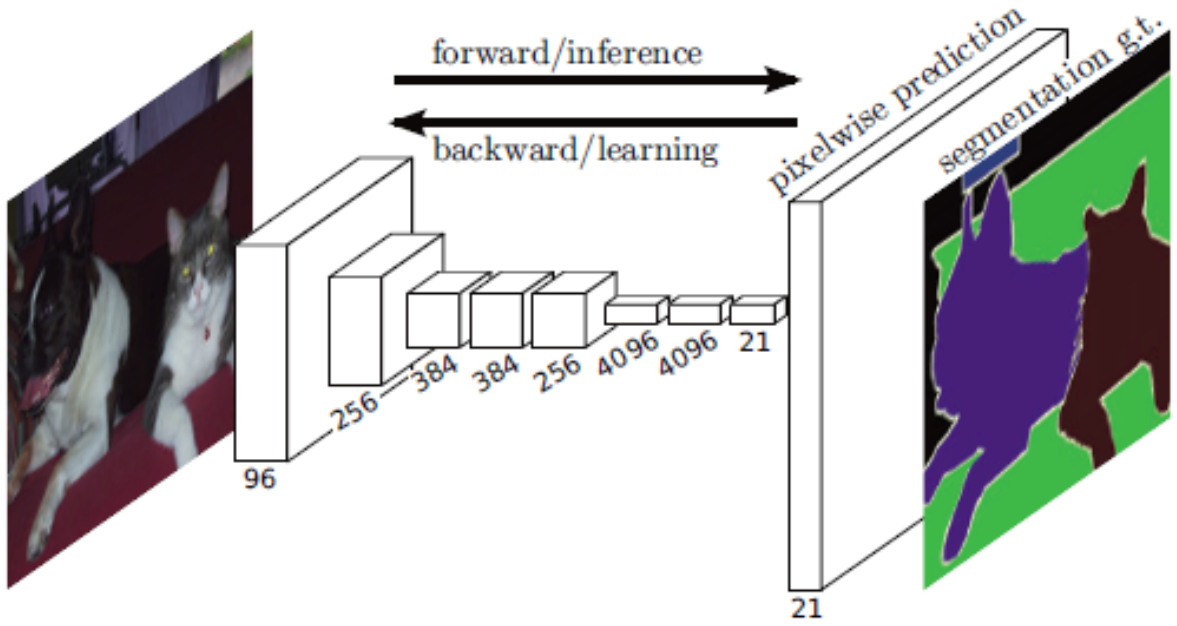
Label Studio: An Open Source Annotation Tool https://labelstudio.io

Popular CNNs for Semantic Segmentation

- FCN: Fully Convolutional Network - CVPR'15
- SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation - IEEE PAMI'17
- U-Net: Convolutional Networks for Biomedical Image Segmentation - MICCAI'15
- PSPNet: Pyramid Scene Parsing Network - CVPR'17

FCN: Fully Convolutional Network

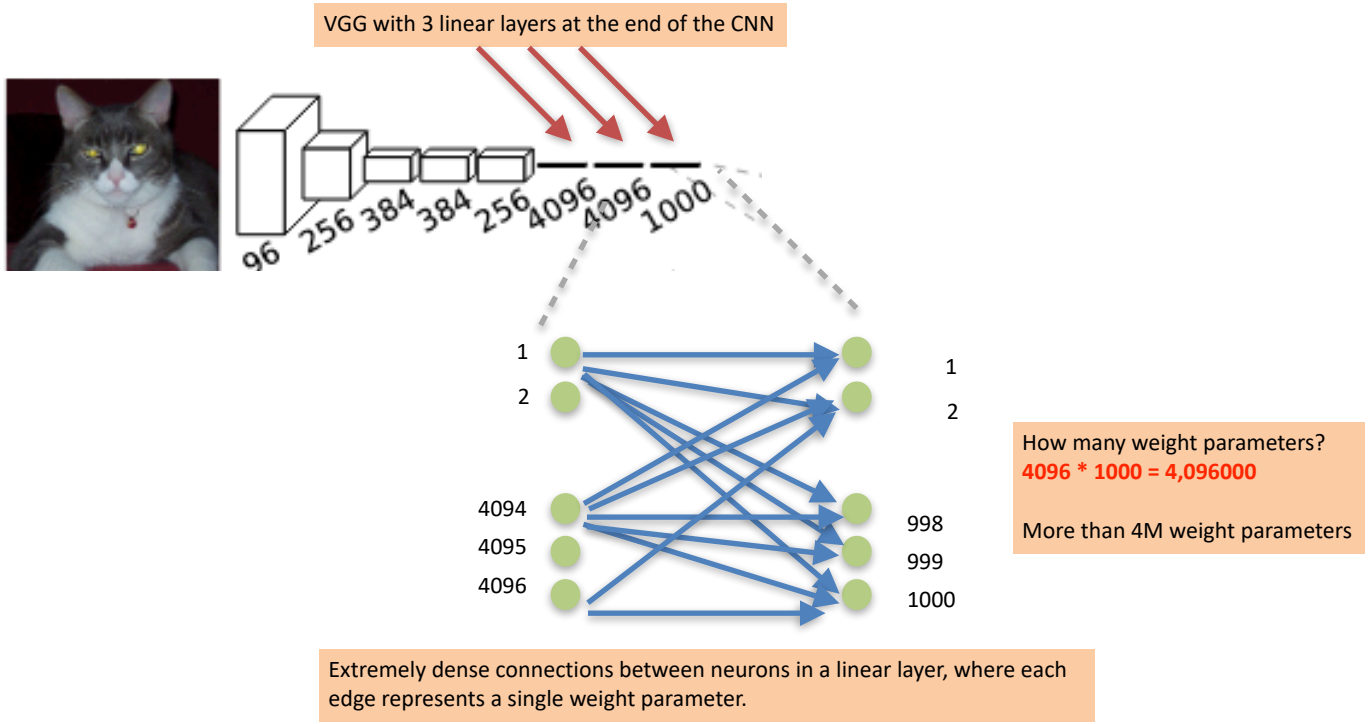
- Semantic segmentation is the task of automatically labeling every pixel in an input image



FCN: Fully Convolutional Network - CVPR'15

FCN: Fully Convolutional Network

- Why convolutional only operations?
  - The fully connected layer has been removed from the CNN, eliminating the dense connections at the end of the network. This is beneficial as it reduces the number of weight parameters that need to be learned.
  - Dense connections in a fully connected (or linear) layer require a large number of weight parameters. (BAD)



FCN: Fully Convolutional Network

- Main idea: do convolutional operation instead of the fully connected operation (or linear layer) towards the end of the CNN

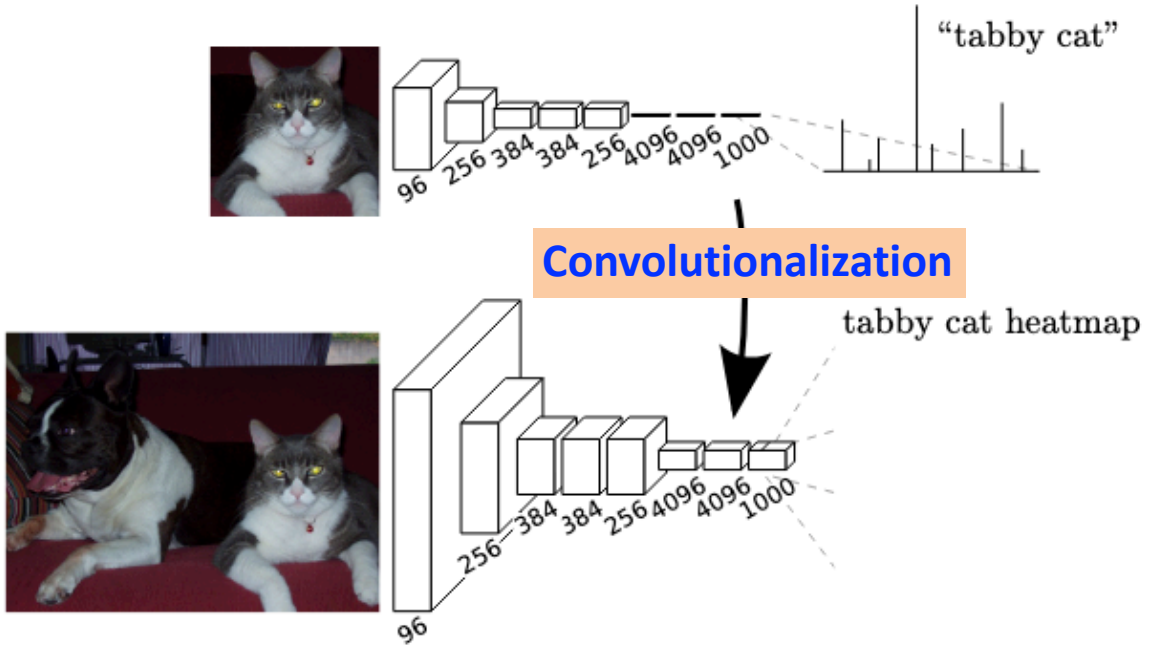
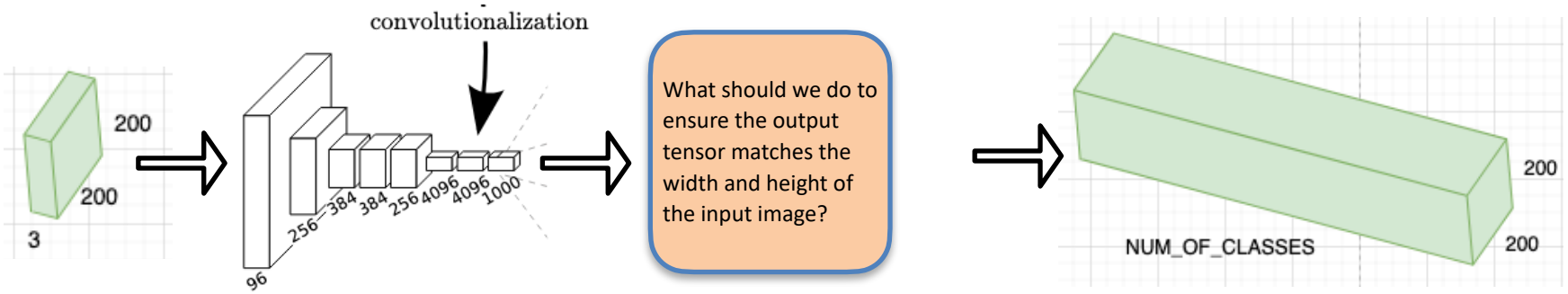


Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.

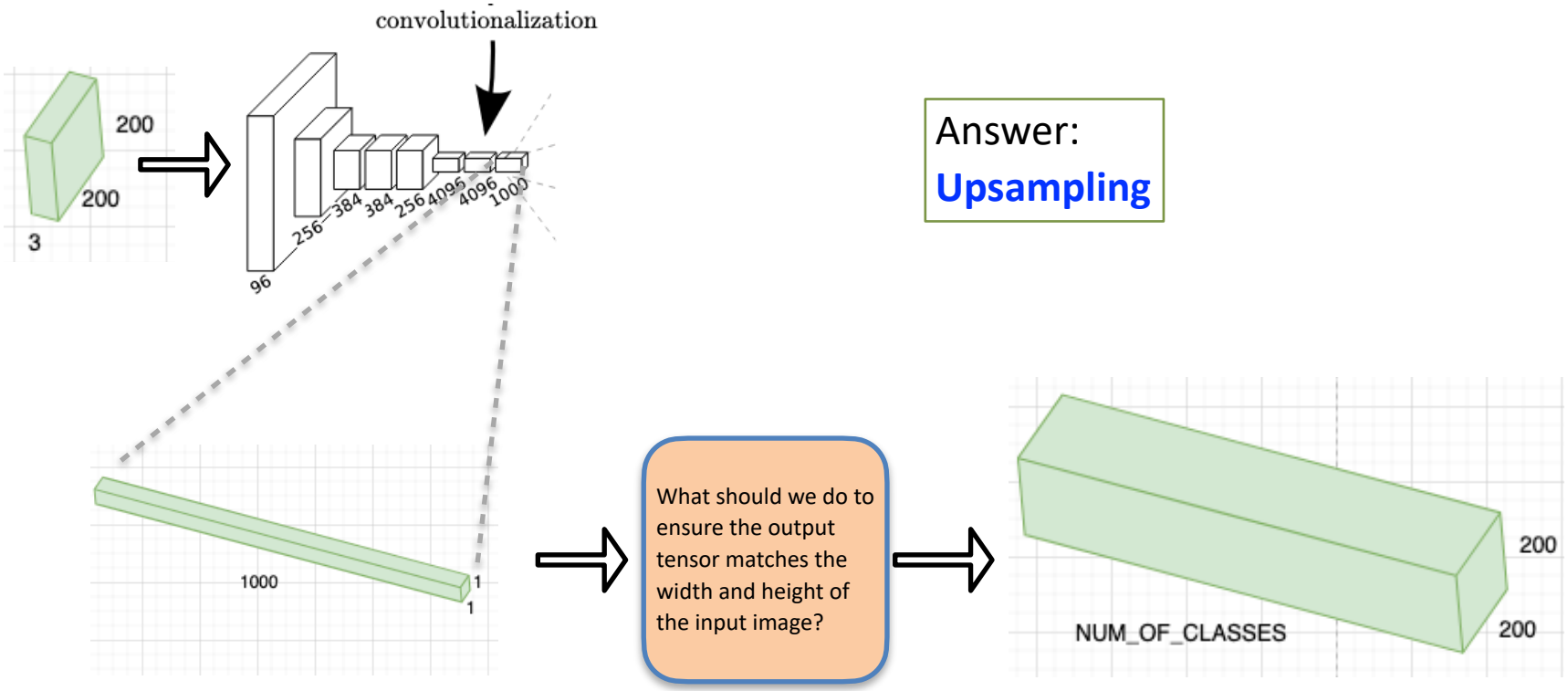
FCN: Fully Convolutional Network

- Next challenge:
  - How can we match the output size to the input resolution? For example, if the input is a tensor of size **200x200**x3, how can we obtain an output tensor of size **200x200**xNUM_OF_CLASSES?



FCN: Fully Convolutional Network

- **Next challenge:** How can we match the output size to the input resolution? For example, if the input is a tensor of size **200x200**x3, how can we obtain an output tensor of size **200x200**xNUM_OF_CLASSES?



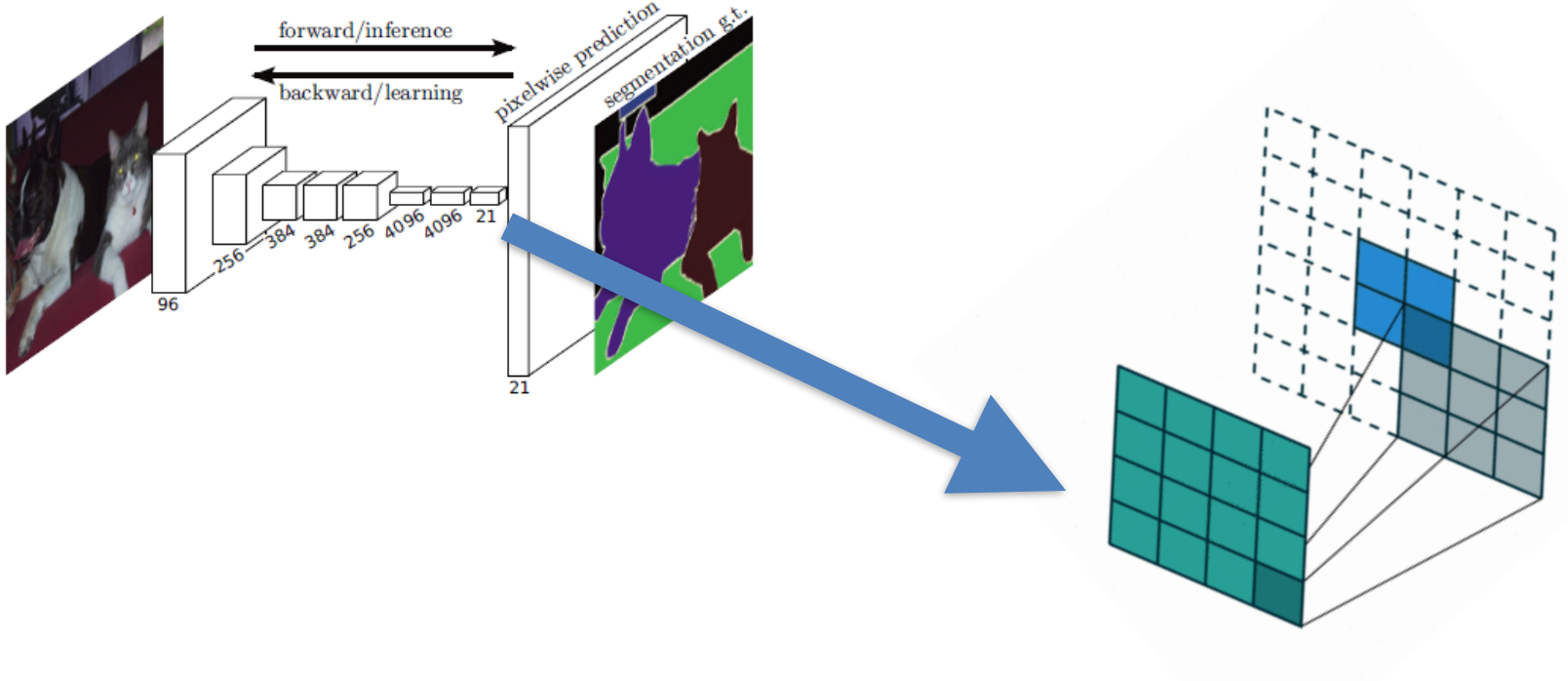
Answer:
Upsampling

FCN: Upsampling via Transposed Convolution Operation

- Spatial dimensions (width and height) of the feature maps shrink due to repeated convolution and pooling operations

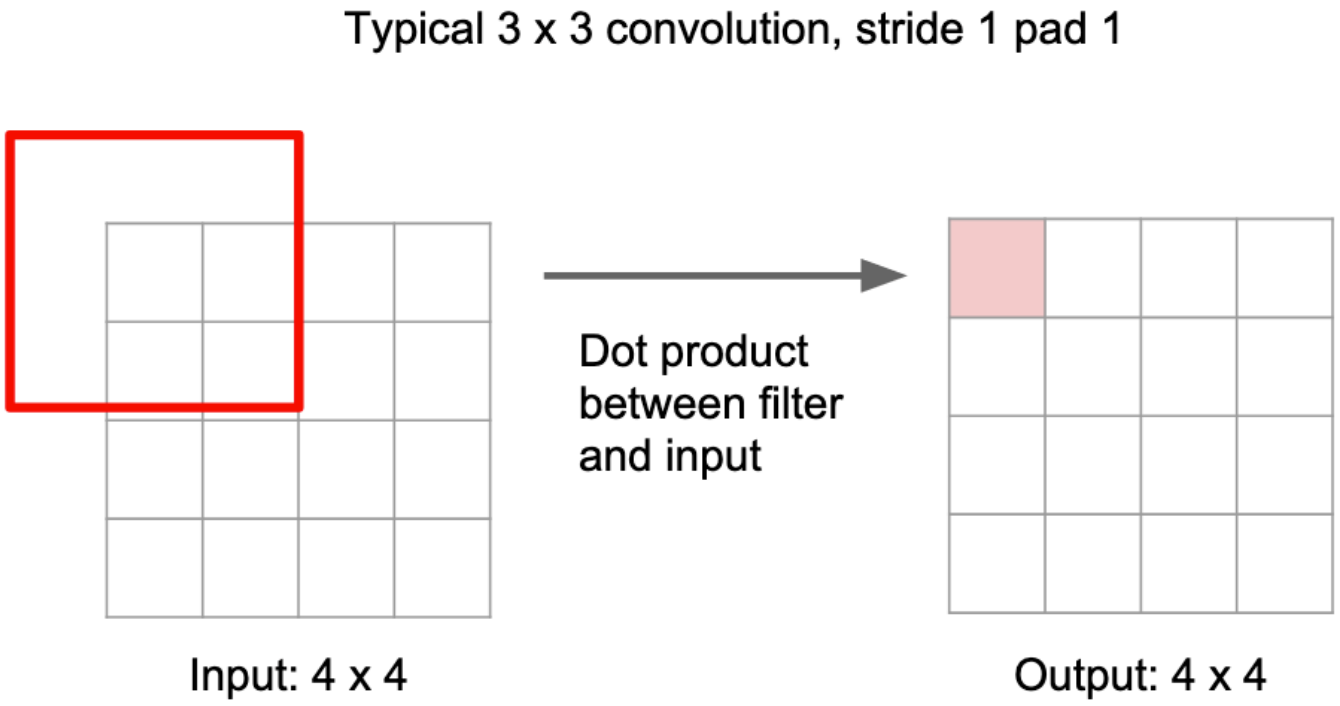
Deconvolution is an incorrect name, but people sometimes refer to this process using this term.

- FCN applies upsampling called **transposed convolution** or **fractional-strided convolution**



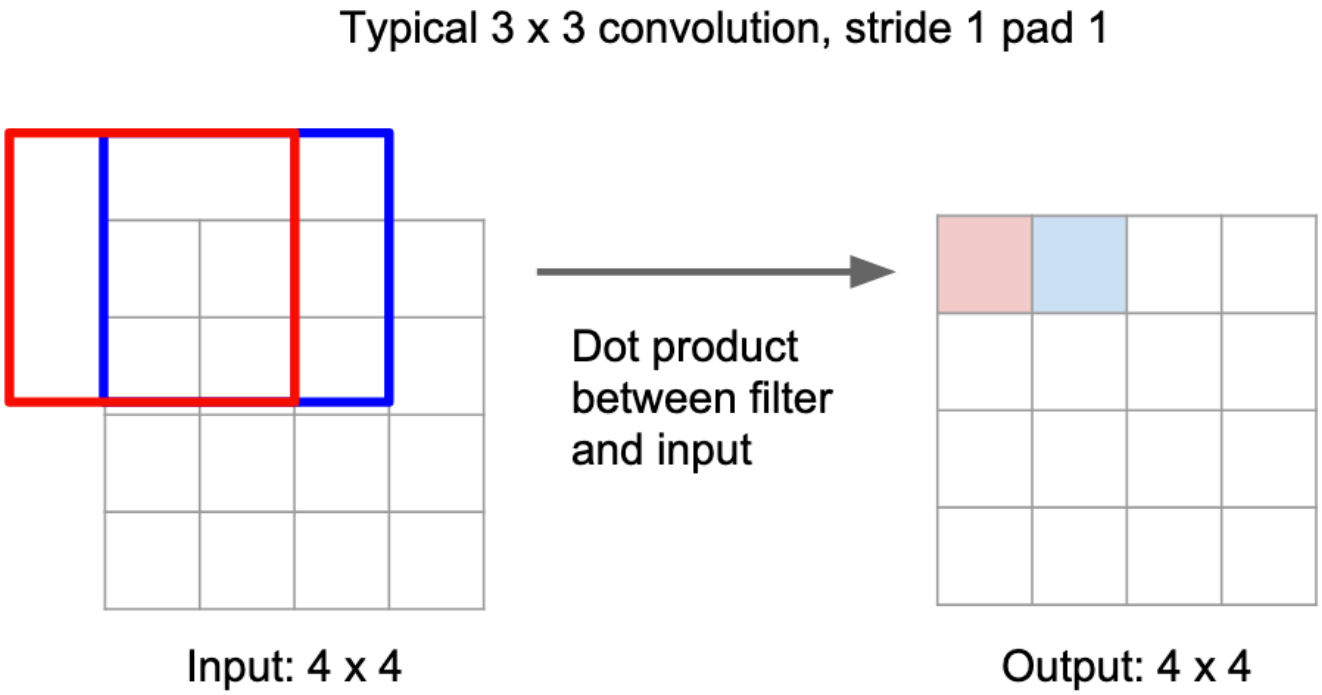
FCN: Upsampling via Transposed Convolution

- Let's look into the traditional **convolution** operation



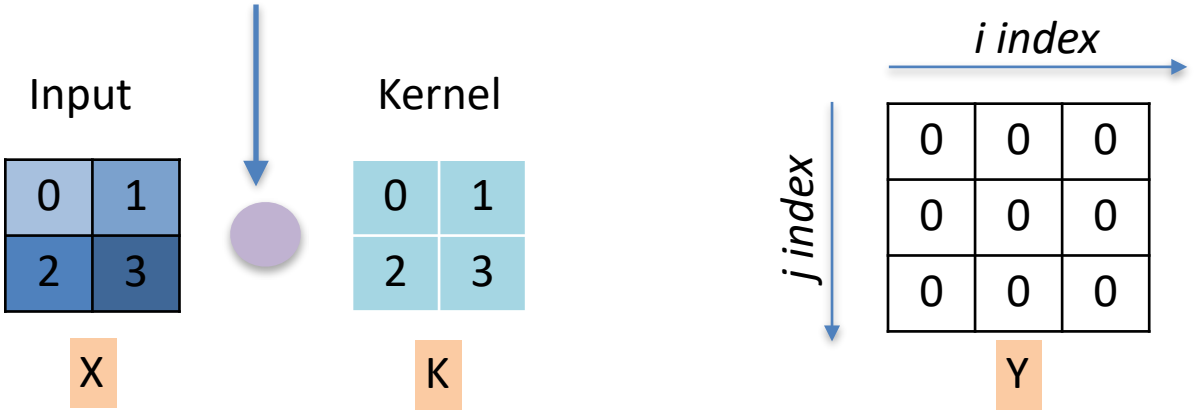
FCN: Upsampling via Transposed Convolution

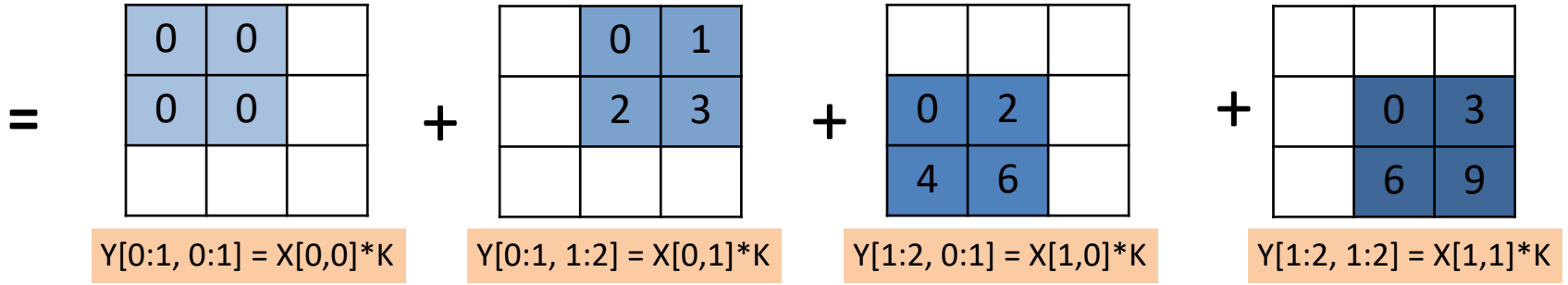
- Let's look into the traditional **convolution** operation



FCN: Upsampling via Transposed Convolution

- Now let's look into the **transposed convolution** operation



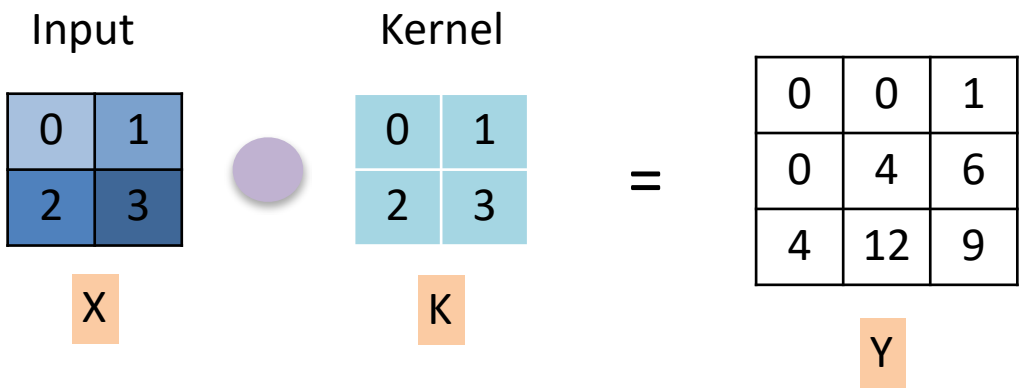


=

0	0	1
0	4	6
4	12	9

FCN: Upsampling via Transposed Convolution

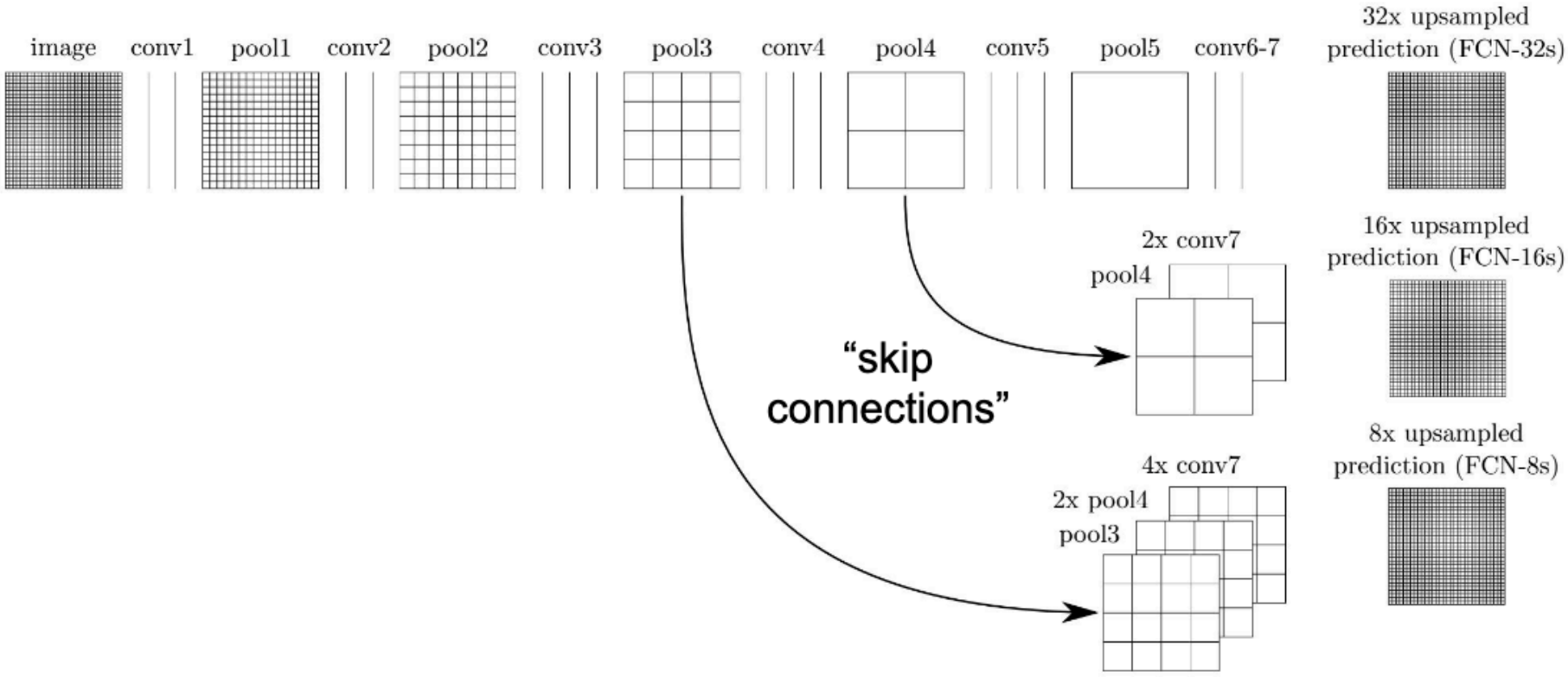
- So input X **transposed convolution** operation with K will produce Y




```
def transpose_convolution(X, K):
    h, w = K.shape
    Y = zeros((X.shape[0] + h -1, X.shape[1] + w -1))
    for i in range(X.shape[0]):
        for j in range(X.shape[1]):
            Y[i:i+h, j:j+w] += X[i,j]*K
```

FCN: Upsampling via Transposed Convolution

- FCN applies a learnable upsampling called **transposed convolution**



FCN semantic segmentation results

- FCN applies

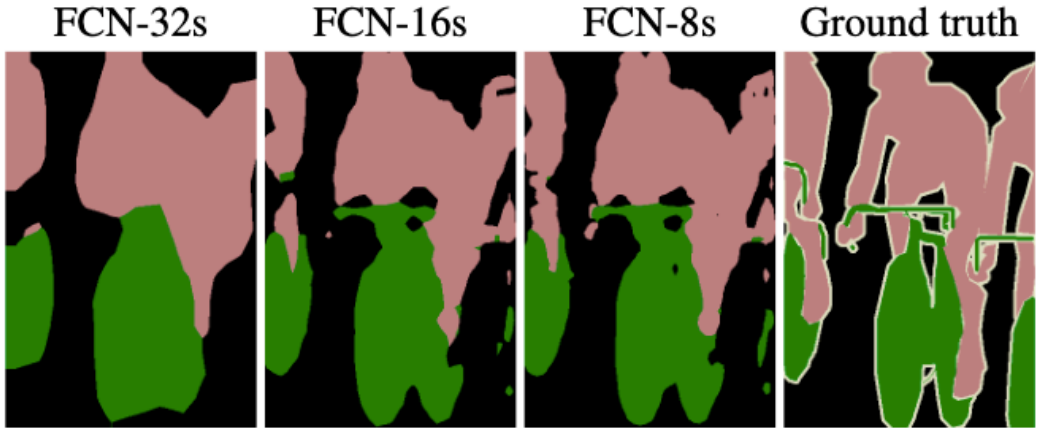


Figure 4. Refining fully convolutional nets by fusing information from layers with different strides improves segmentation detail. The first three images show the output from our 32, 16, and 8 pixel stride nets (see Figure 3).

Table 2. Comparison of skip FCNs on a subset of PASCAL VOC2011 validation[7]. Learning is end-to-end, except for FCN-32s-fixed, where only the last layer is fine-tuned. Note that FCN-32s is FCN-VGG16, renamed to highlight stride.

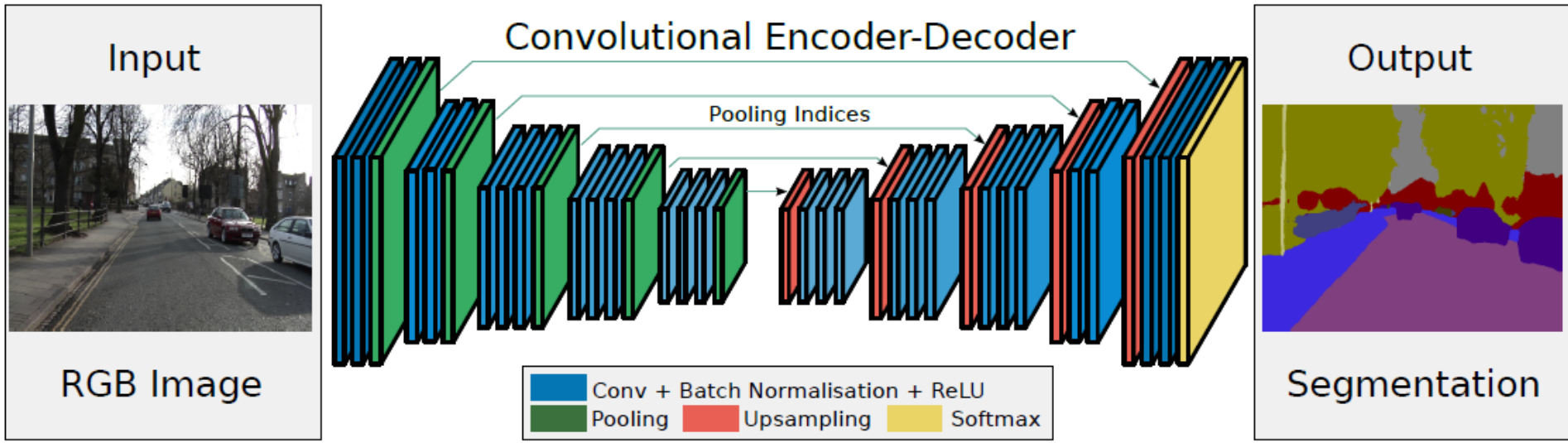
	pixel acc.	mean acc.	mean IU	f.w. IU
FCN-32s-fixed	83.0	59.7	45.4	72.0
FCN-32s	89.1	73.3	59.4	81.4
FCN-16s	90.0	75.7	62.4	83.0
FCN-8s	**90.3**	**75.9**	**62.7**	**83.2**

Popular CNNs for Semantic Segmentation

- FCN: Fully Convolutional Network - CVPR'15
- SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation - IEEE PAMI'17
- U-Net: Convolutional Networks for Biomedical Image Segmentation - MICCAI'15
- PSPNet: Pyramid Scene Parsing Network - CVPR'17

SegNet

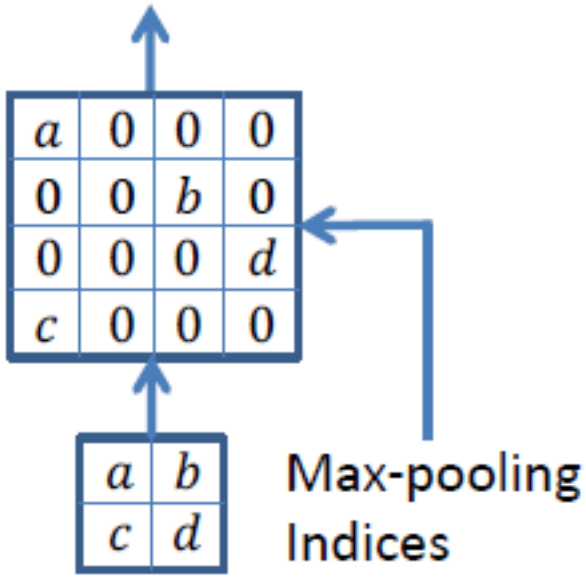
- Semantic segmentation is the task of automatically labeling every pixel in an input image



SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation - IEEE PAMI'17

SegNet

- Upsampling operation is much simpler than transposed convolution
  - Spatial dimensions (width and height) of the feature maps shrink due to repeated max-pooling operations
  - Save the indices of the pooling max-pooling operation
  - Use those indices during upsampling operation



SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation - IEEE PAMI'17

SegNet

- Upsampling operation is much simpler than transposed convolution

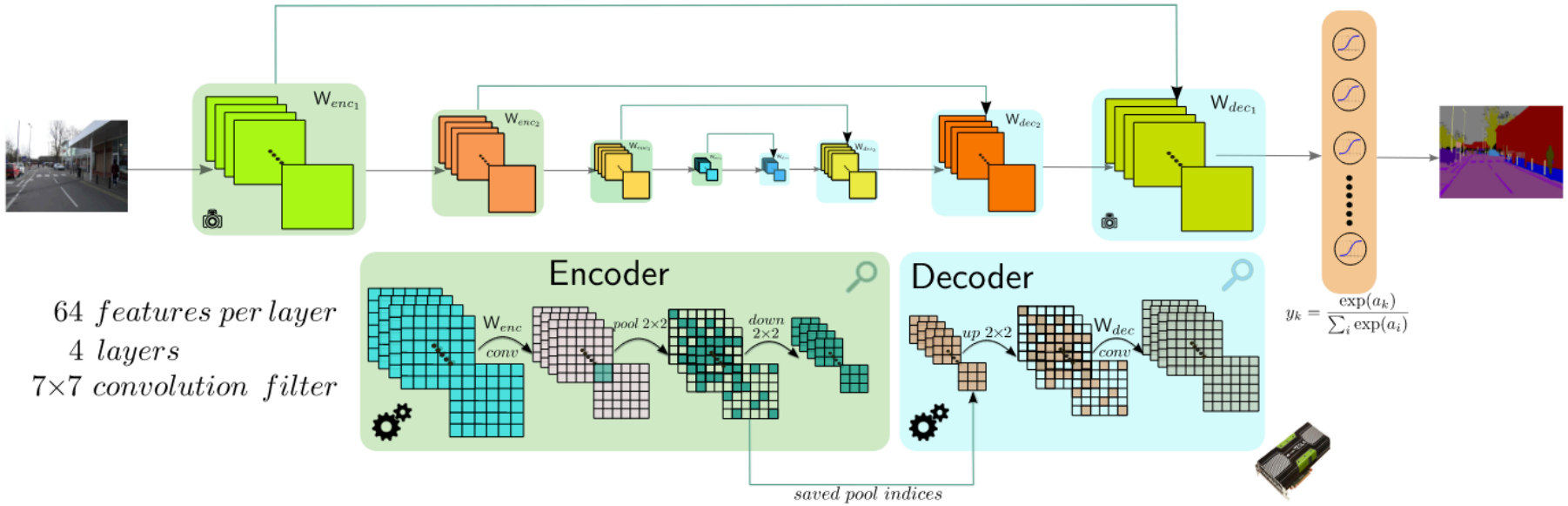
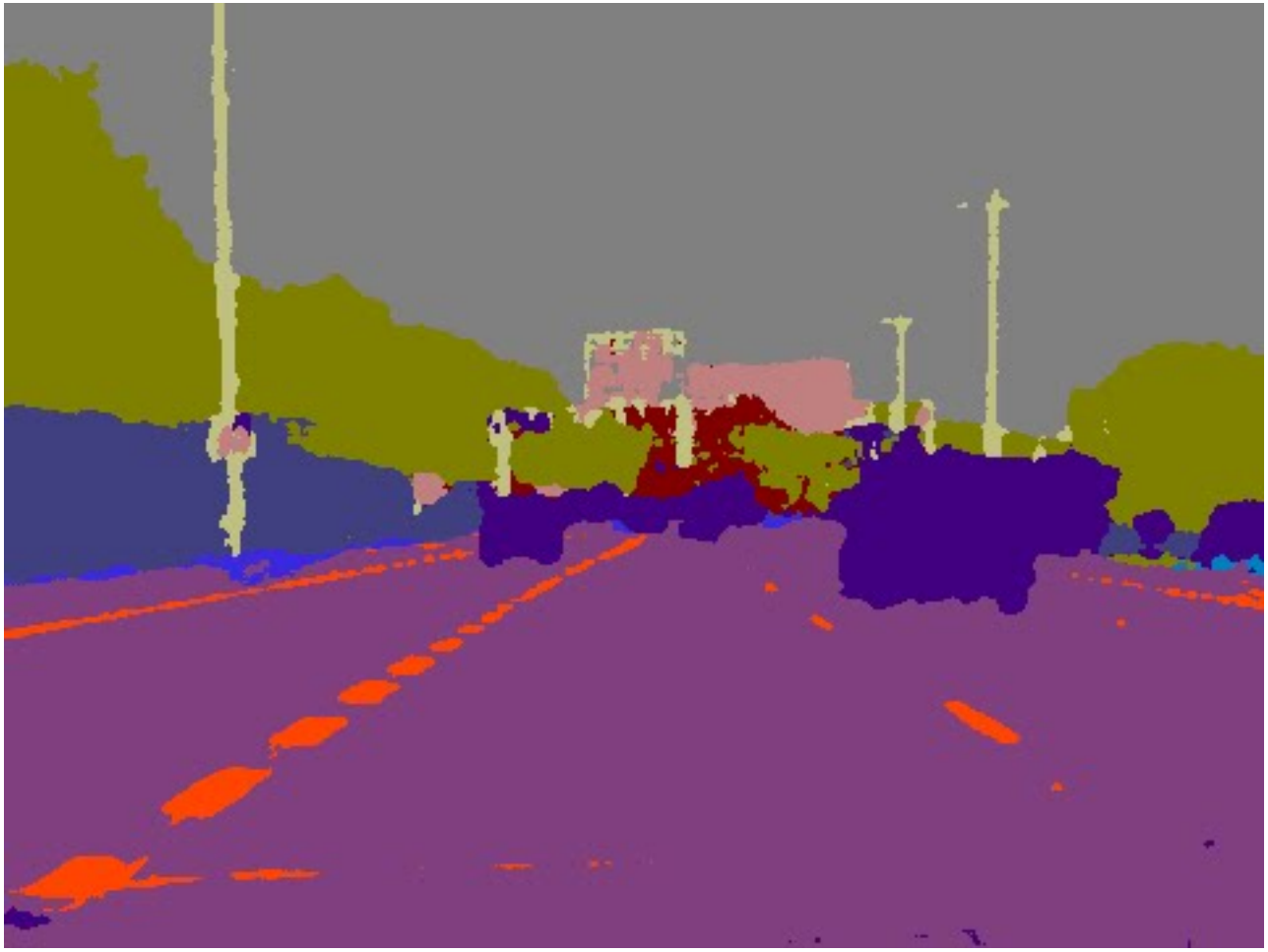


Figure 1. A 4 layer SegNet which takes in an RGB input image and performs *feed-forward* computation to obtain pixel-wise labelling. A stack of feature encoders is followed by a corresponding decoders. The soft-max layer classifies each pixel independently using the features input by the last decoder. An encoder uses the convolution-ReLU-max pooling-subsampling pipeline. A decoder upsamples its input using the transferred pool indices from its encoder. It then performs convolution with a trainable filter bank.

SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation - IEEE PAMI'17

SegNet: Results



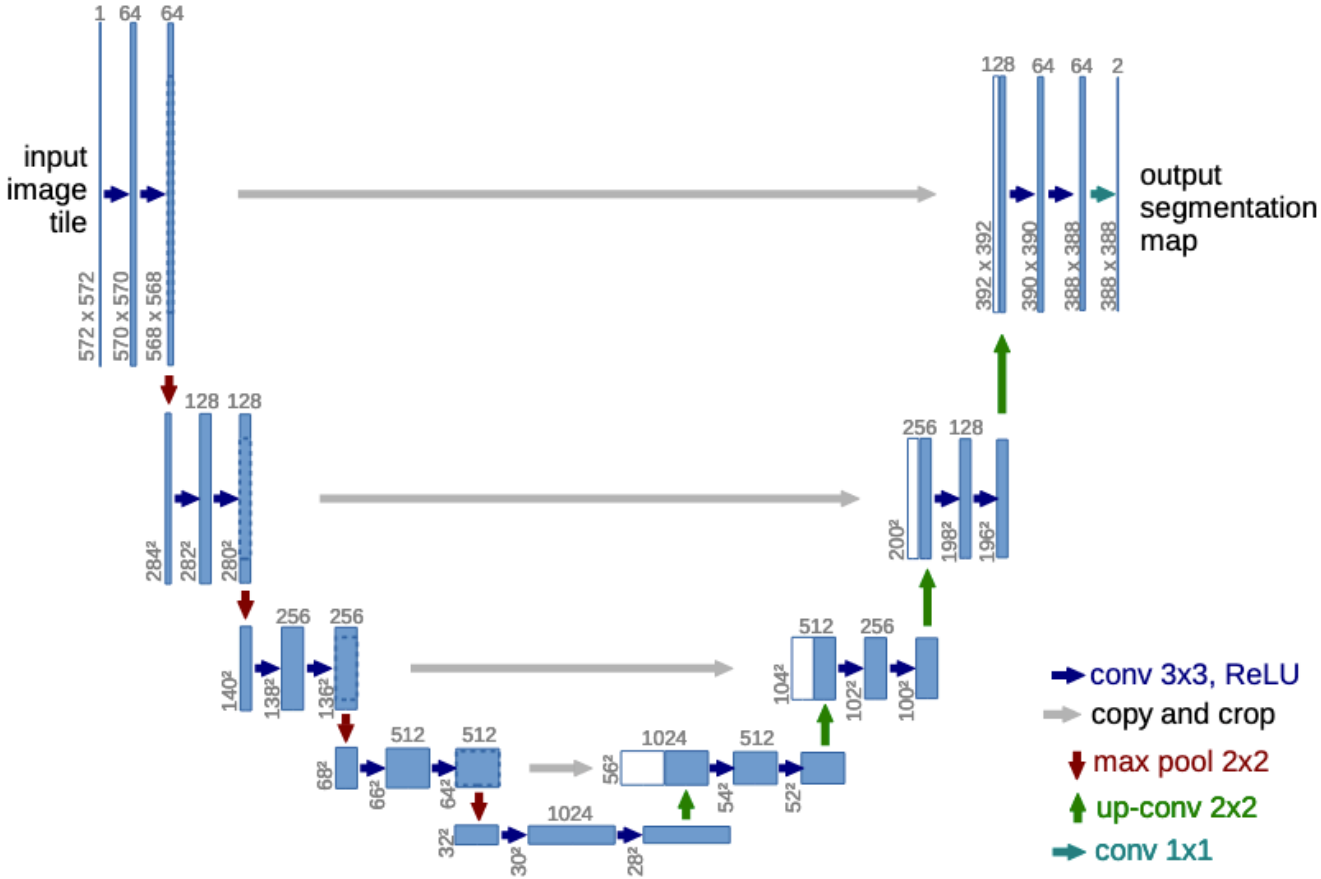
SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation - IEEE PAMI'17

Popular CNNs for Semantic Segmentation

- FCN: Fully Convolutional Network - CVPR'15
- SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation - IEEE PAMI'17
- U-Net: Convolutional Networks for Biomedical Image Segmentation - MICCAI'15
- PSPNet: Pyramid Scene Parsing Network - CVPR'17

U-Net

- Very successful semantic segmentation model for medical image segmentation
- **Main idea:** *concatenate the entire feature map*, rather than saving the pooling indices



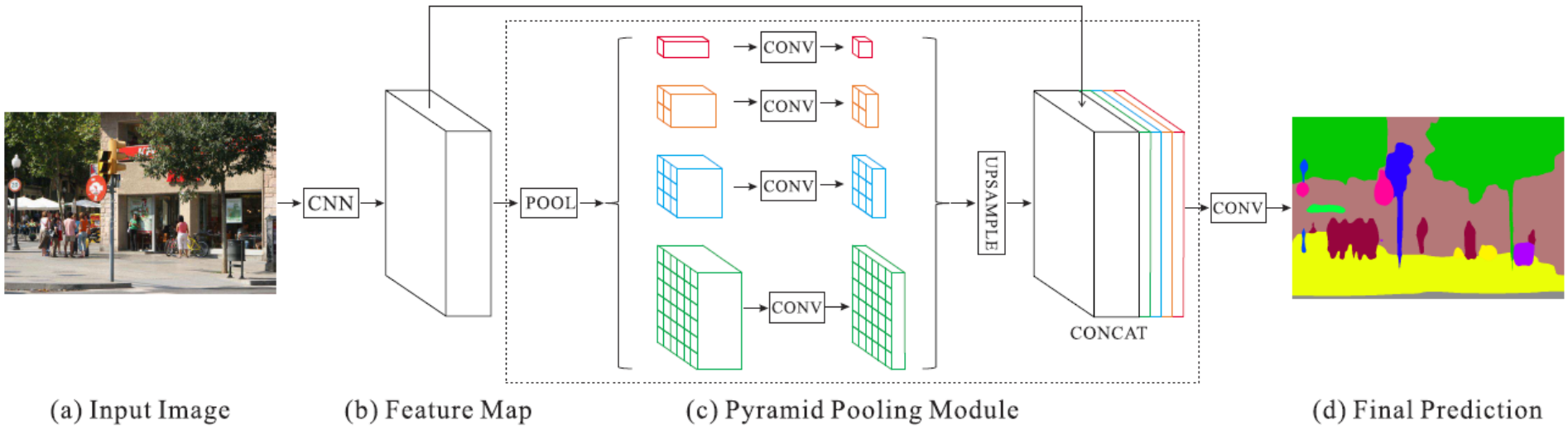
U-Net: Convolutional Networks for Biomedical Image Segmentation - MICCAI'15

Popular CNNs for Semantic Segmentation

- FCN: Fully Convolutional Network - CVPR'15
- SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation - IEEE PAMI'17
- U-Net: Convolutional Networks for Biomedical Image Segmentation - MICCAI'15
- PSPNet: Pyramid Scene Parsing Network - CVPR'17

PSPNet

- **Main idea:** *build feature pyramid with multiple pooling operations*
  - *Produces output maps of 1x1, 2x2, 3x3, 6x6*
  - *Apply upsampling on each (using simple bilinear interpolation)*
  - *Combine the upsampled feature maps using concatenation operation*
- *different pooling outputs preserve different contextual information (global contexts)*



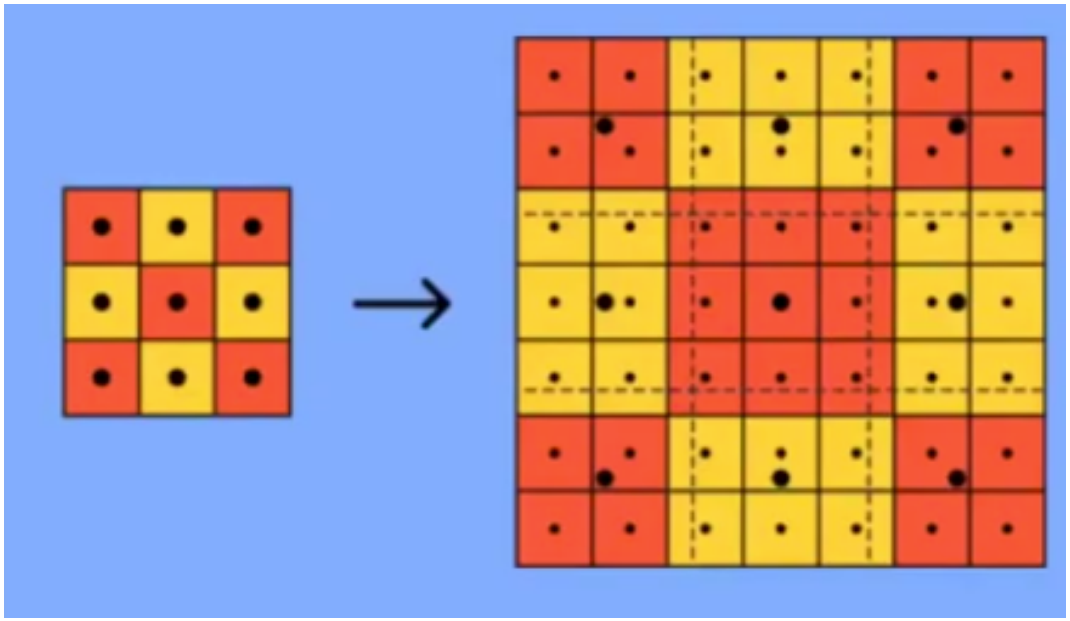
PSPNet: Pyramid Scene Parsing Network - CVPR'17

PSPNet

PSPNet: [Pyramid Scene Parsing Network - CVPR'17]

PSPNet: interpolation

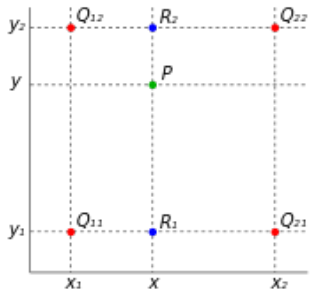
- Nearest Neighbor Interpolation



PSPNet: interpolation

- Bilinear Interpolation

$$f(x, y_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}),$$
$$f(x, y_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}).$$



$$\begin{aligned}
f(x, y) &\approx \frac{y_2 - y}{y_2 - y_1} f(x, y_1) + \frac{y - y_1}{y_2 - y_1} f(x, y_2) \\
&= \frac{y_2 - y}{y_2 - y_1} \left(\frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \right) + \frac{y - y_1}{y_2 - y_1} \left(\frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \right) \\
&= \frac{1}{(x_2 - x_1)(y_2 - y_1)} \left(f(Q_{11})(x_2 - x)(y_2 - y) + f(Q_{21})(x - x_1)(y_2 - y) + f(Q_{12})(x_2 - x)(y - y_1) + f(Q_{22})(x - x_1)(y - y_1) \right) \\
&= \frac{1}{(x_2 - x_1)(y_2 - y_1)} \begin{bmatrix} x_2 - x & x - x_1 \end{bmatrix} \begin{bmatrix} f(Q_{11}) & f(Q_{12}) \\ f(Q_{21}) & f(Q_{22}) \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix}.
\end{aligned}$$