Computer Vision

Semantic segmentation

Tae-Hyun Oh (오태현)
전자전기공학과
POSTECH

Slide by Sungbin Kim (김성빈)
TAs: {Dongmin Choi , Jongha Kim, Juyong Lee, Sungbin Kim} (in alphabetic order)

boostcamp AI Tech

1. Semantic segmentation

1.1 What is semantic segmentation?

1.2 Where can semantic segmentation be applied to?

2. Semantic segmentation architectures

2.1 Fully Convolutional Networks (FCN)

2.2 Hypercolumns for object segmentation

2.3 U-Net

2.4 DeepLab

1.

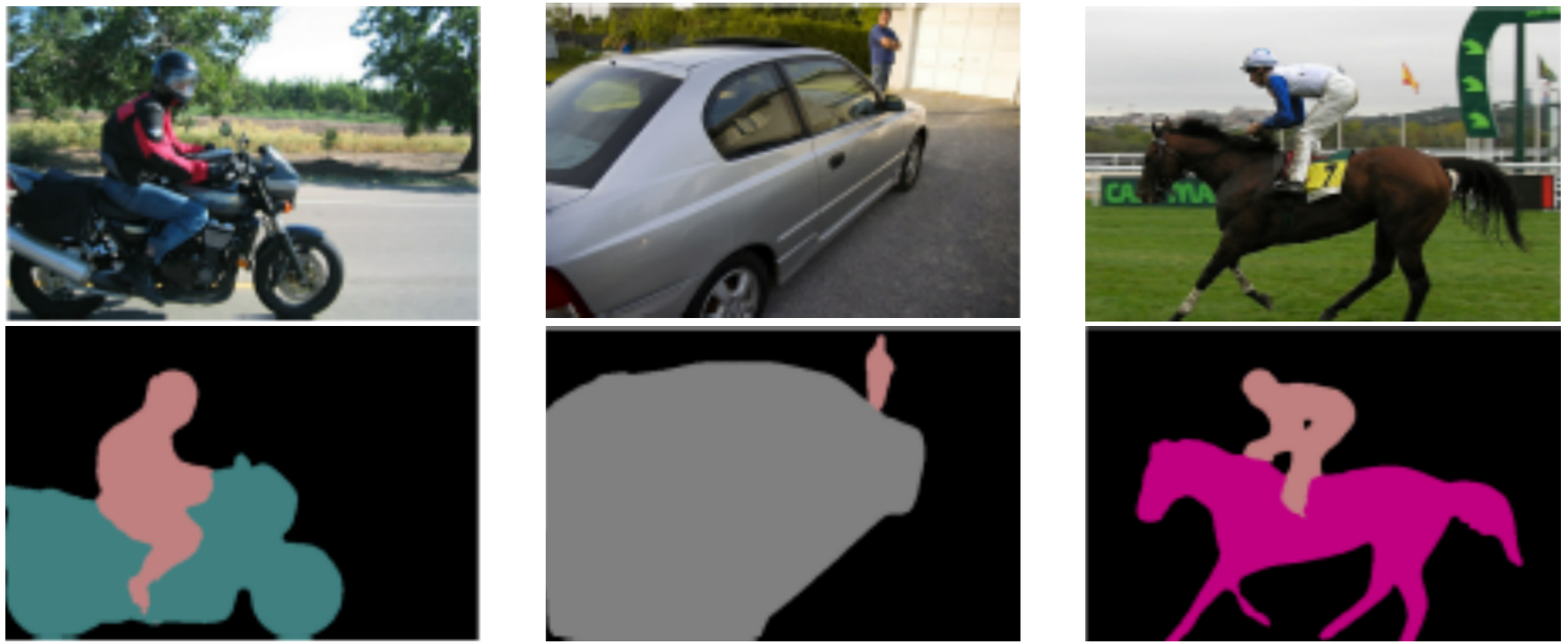
Semantic segmentation

1.1 What is semantic segmentation?

Semantic segmentation

[Chen et al., arXiv 2017]

- Classify each pixel of an image into a category
- Don't care about instances. Only care about semantic category

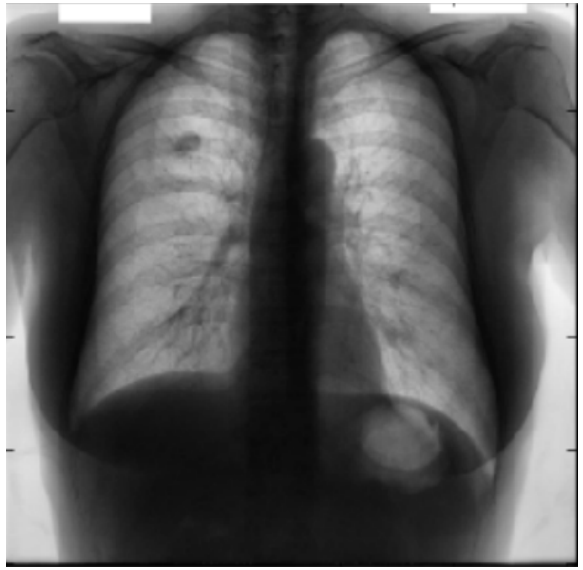


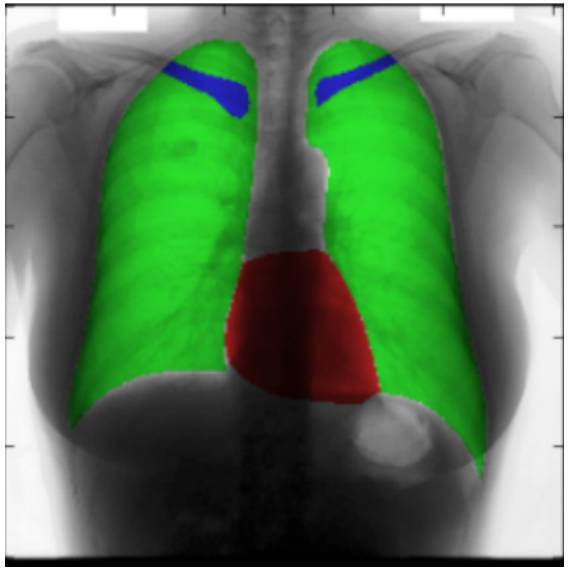
1.2 Where can semantic segmentation be applied to?

Applications

[Novikov et al., IEEE T-MI 2016]

- Medical images
- Autonomous driving
- Computational photography (next slide)
- ...





[https://thegradient.pub/semantic-segmentation/]


[Aksoy et al., SIGGRAPH 2018]


[Aksoy et al., SIGGRAPH 2018]


[Aksoy et al., SIGGRAPH 2018]

2.

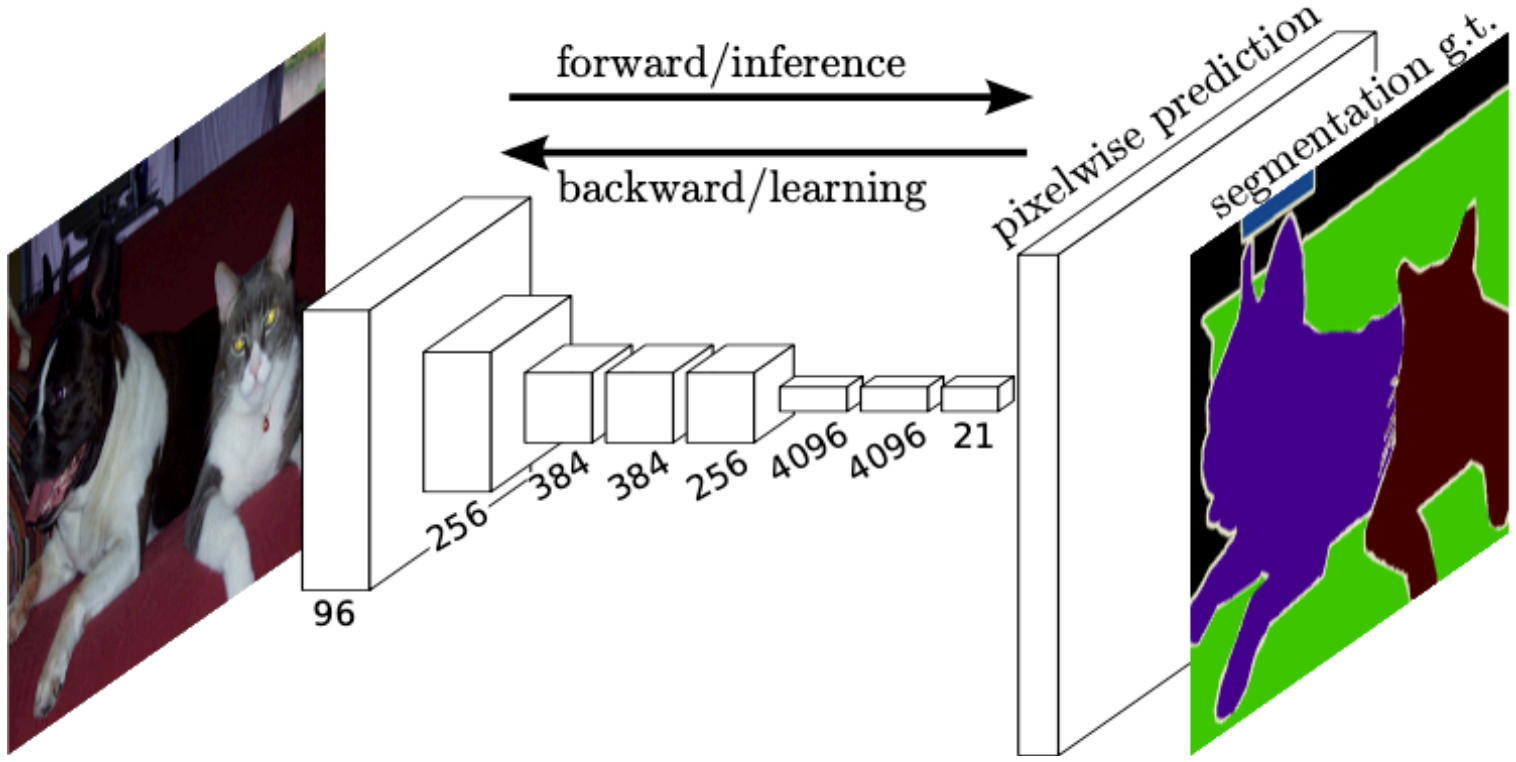
Semantic segmentation architectures

2.1 Fully Convolutional Networks (FCN)

Fully convolutional networks

[Long et al., CVPR 2015]

- The first end-to-end architecture for semantic segmentation
- Take an image of an arbitrary size as input,
  and output a segmentation map of the corresponding size to the input



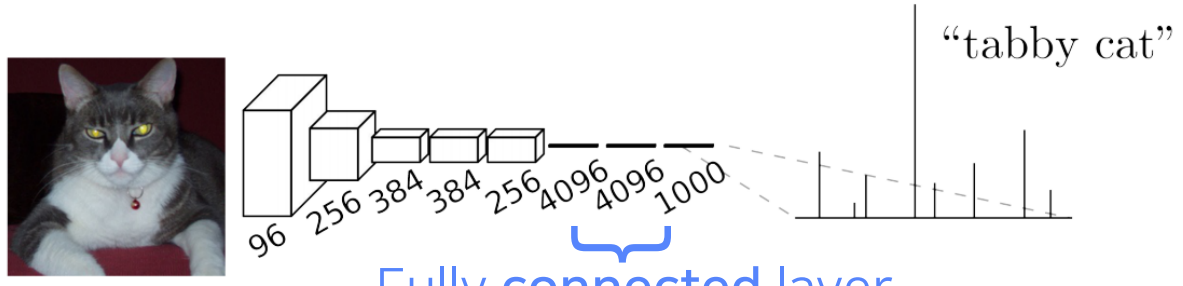
2.1 Fully Convolutional Networks (FCN)

Fully connected vs Fully convolutional

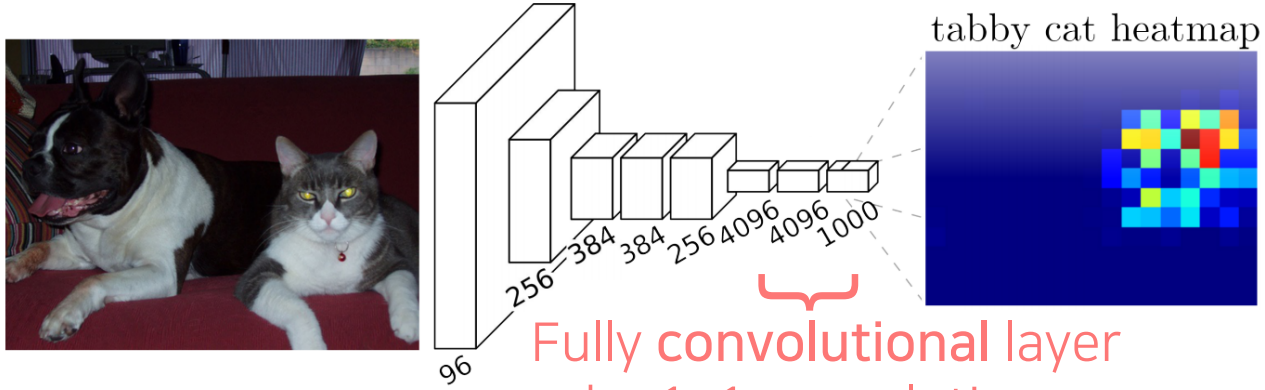
[Long et al., CVPR 2015]

- Fully connected layer: Output a fixed dimensional vector and discard spatial coordinates
- Fully convolutional layer: Output a classification map which has spatial coordinates

Image classification



Semantic segmentation

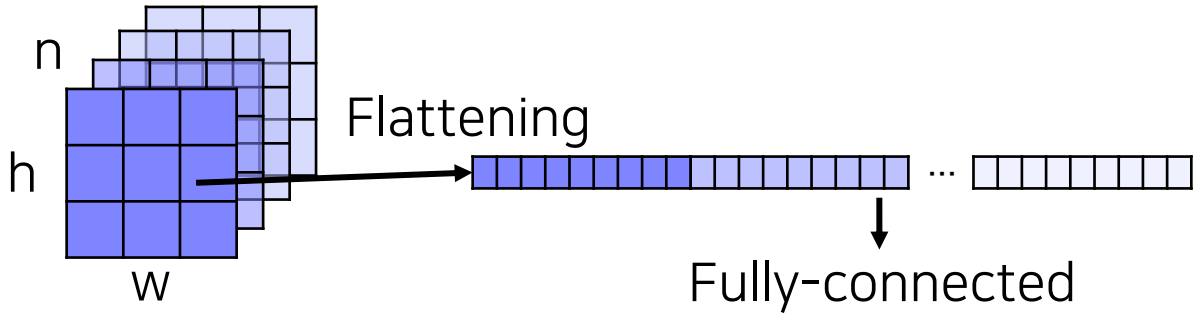


2.1 Fully Convolutional Networks (FCN)

Interpreting fully connected layers as 1x1 convolutions

[Long et al., CVPR 2015]

- A fully connected layer classifies a single feature vector



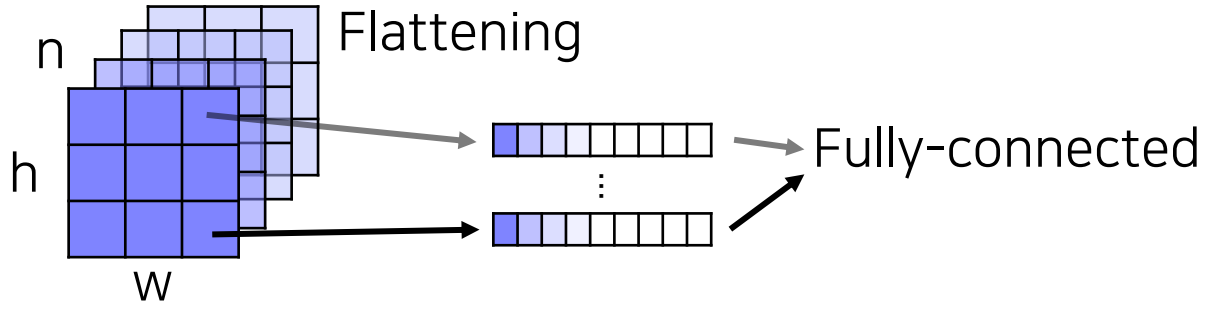
- A 1x1 convolution layer classifies every feature vector of the convolutional feature map

2.1 Fully Convolutional Networks (FCN)

Interpreting fully connected layers as 1x1 convolutions

[Long et al., CVPR 2015]

- A fully connected layer classifies a single feature vector



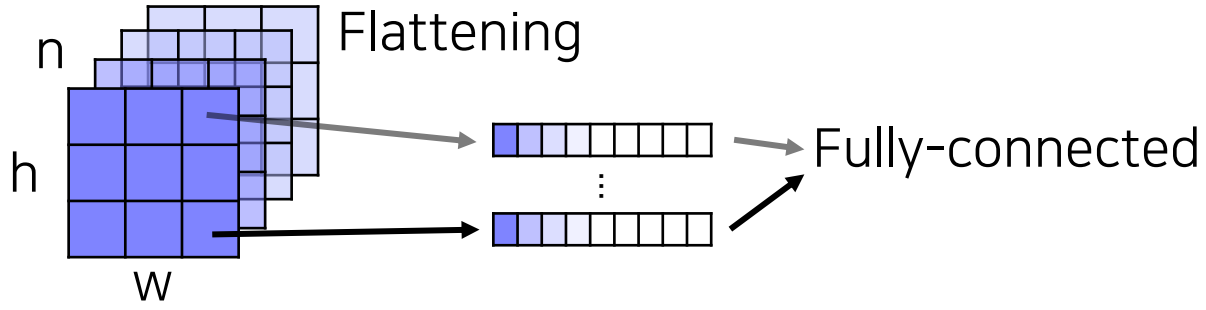
- A 1x1 convolution layer classifies every feature vector of the convolutional feature map

2.1 Fully Convolutional Networks (FCN)

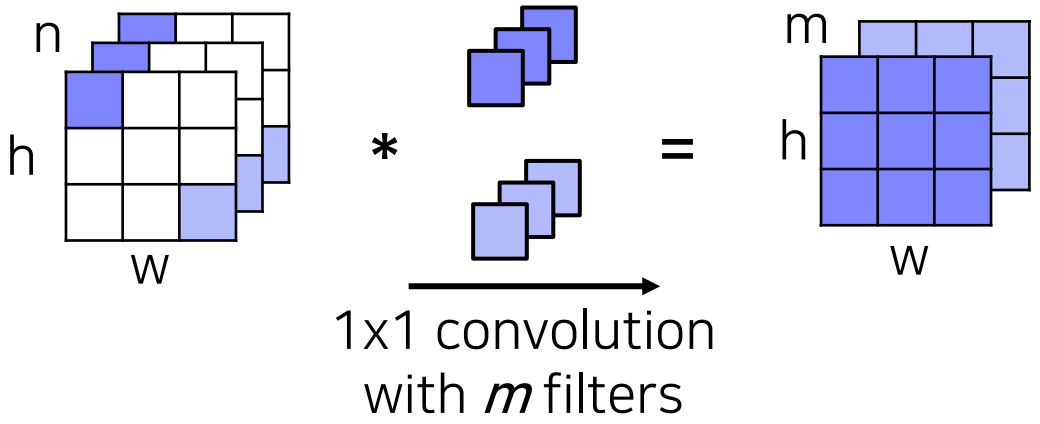
Interpreting fully connected layers as 1x1 convolutions

[Long et al., CVPR 2015]

- A fully connected layer classifies a single feature vector



- A 1x1 convolution layer classifies every feature vector of the convolutional feature map

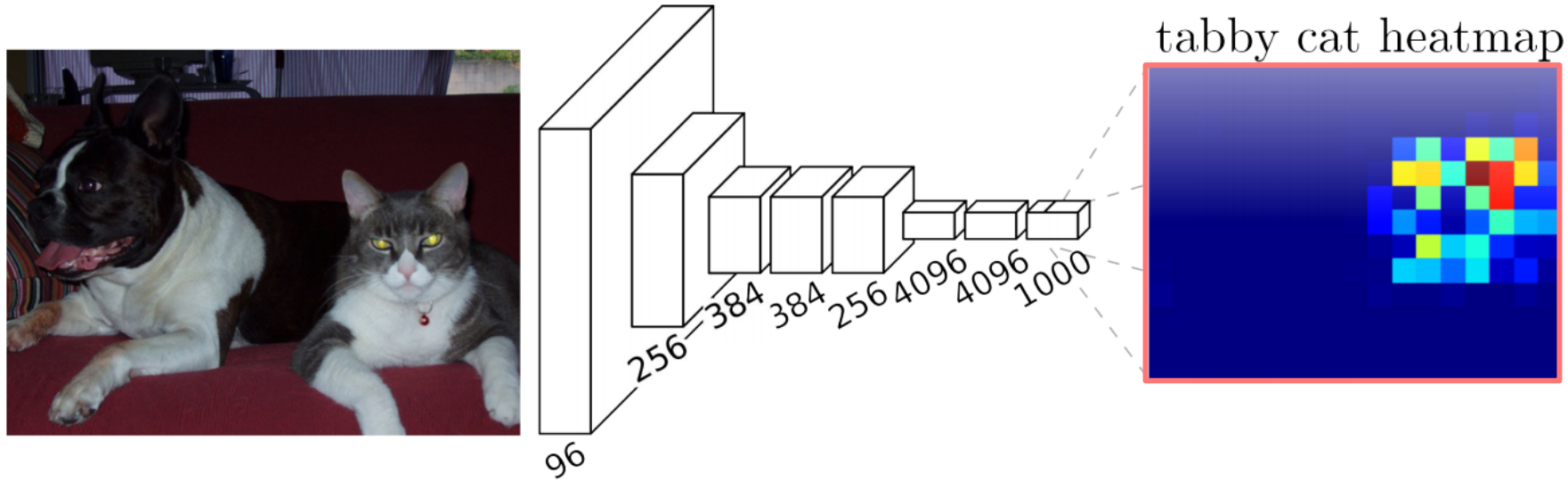


2.1 Fully Convolutional Networks (FCN)

Interpreting fully connected layers as 1x1 convolutions

[Long et al., CVPR 2015]

- A 1x1 convolution layer classifies every feature vector of the convolutional feature map
- Limitation: Predicted score map is in a very low-resolution
- Why?
  - For having a large receptive field, several spatial pooling layers are deployed
- Solution: Enlarge the score map by upsampling!



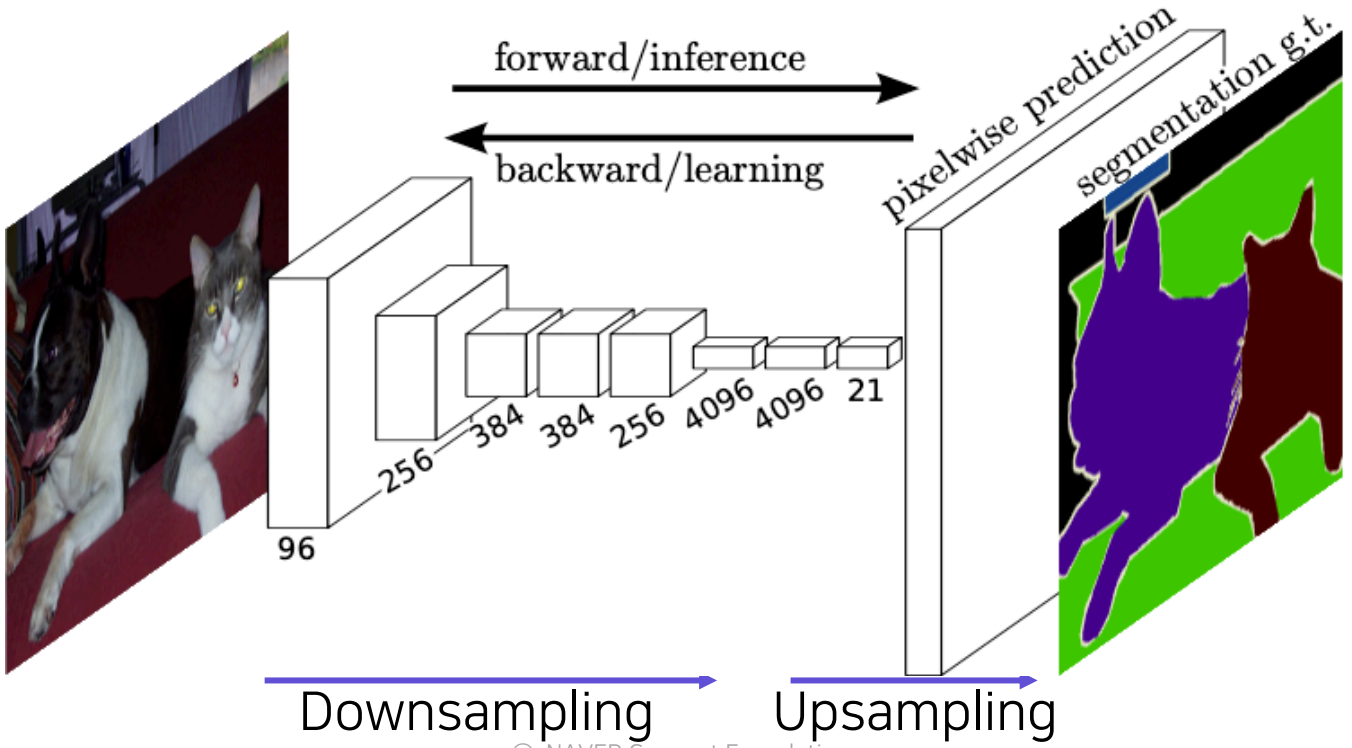
What is upsampling?

2.1 Fully Convolutional Networks (FCN)

Upsampling

[Long et al., CVPR 2015]

- The size of the input image is reduced to a smaller feature map
- Upsample to the size of input image

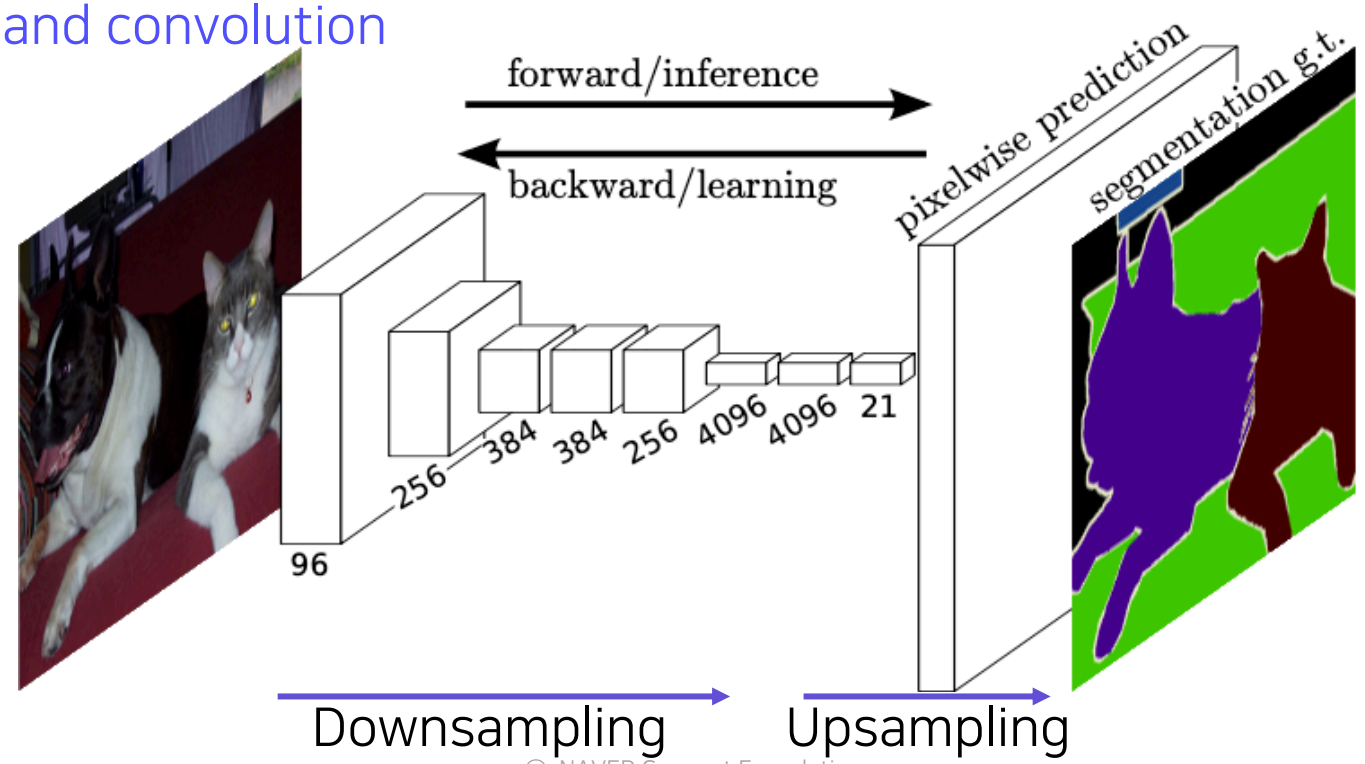


2.1 Fully Convolutional Networks (FCN)

Upsampling

[Long et al., CVPR 2015]

- Upsampling is used to resize a small activation map to the size of the input image
  - Unpooling
  - Transposed convolution
  - Upsample and convolution

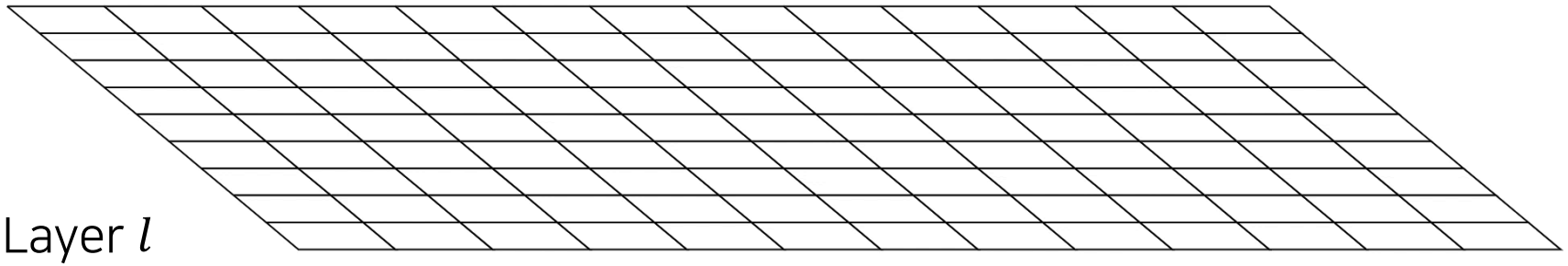


2.1 Fully Convolutional Networks (FCN)

Transposed convolution

- Transposed convolutions work by swapping the forward and backward passes of convolution

Layer $l-1$



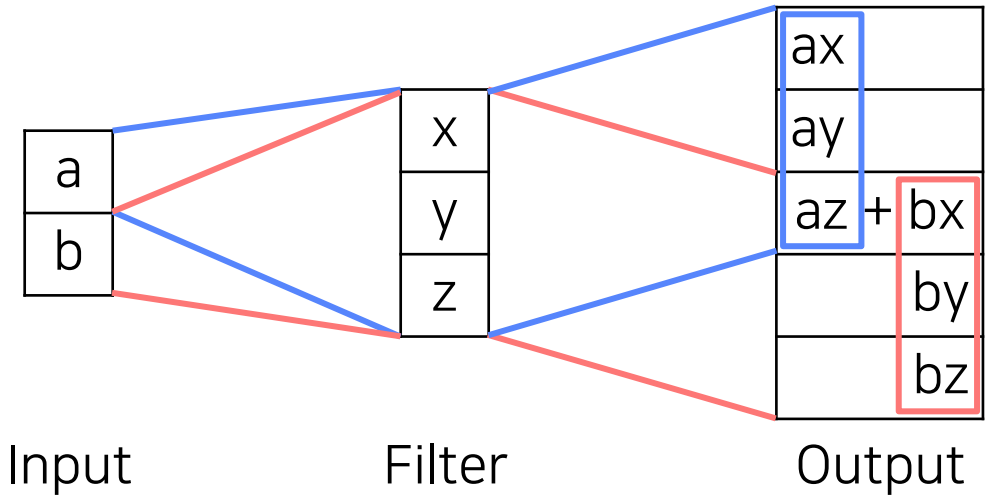
Layer l

[https://distill.pub/2016/deconv-checkerboard/]

2.1 Fully Convolutional Networks (FCN)

Transposed convolution

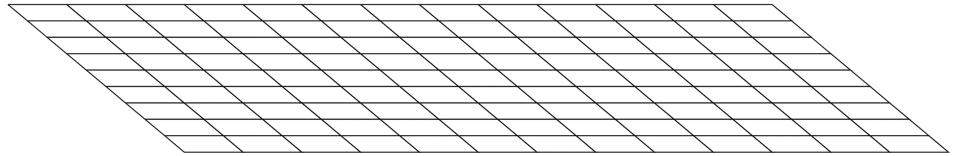
- Transposed convolutions work by swapping the forward and backward passes of convolution

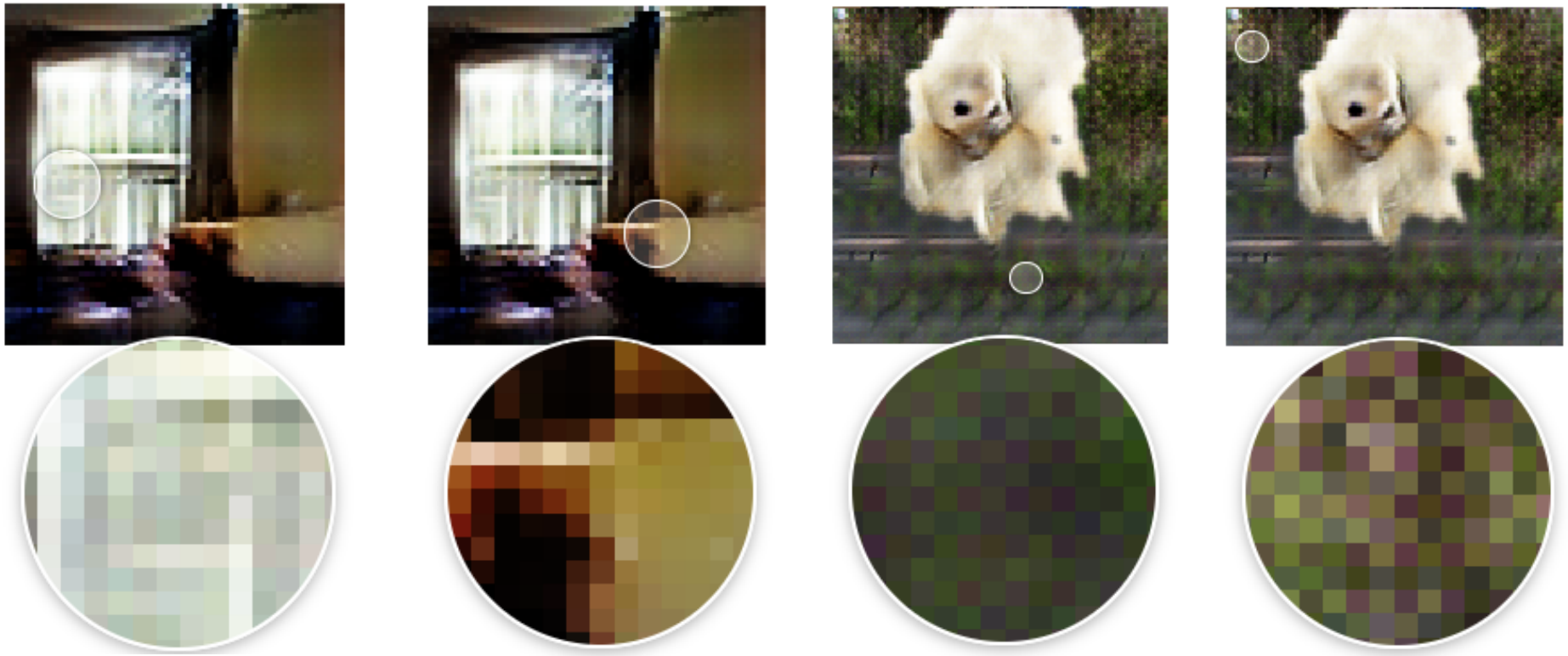


2.1 Fully Convolutional Networks (FCN)

Problems with transposed convolution

- Checkerboard artifacts due to uneven overlaps



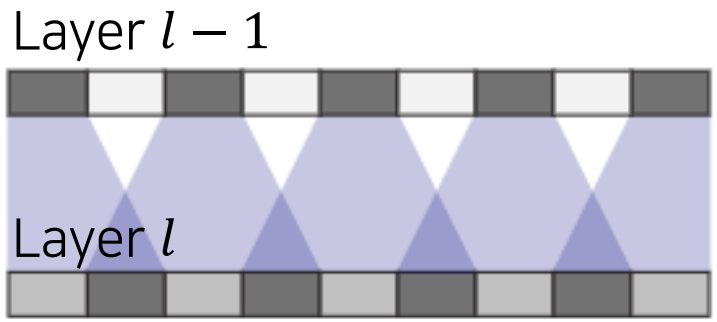


[https://distill.pub/2016/deconv-checkerboard/]

2.1 Fully Convolutional Networks (FCN)

Better approaches for upsampling

- **Avoid overlap issues** in transposed convolution



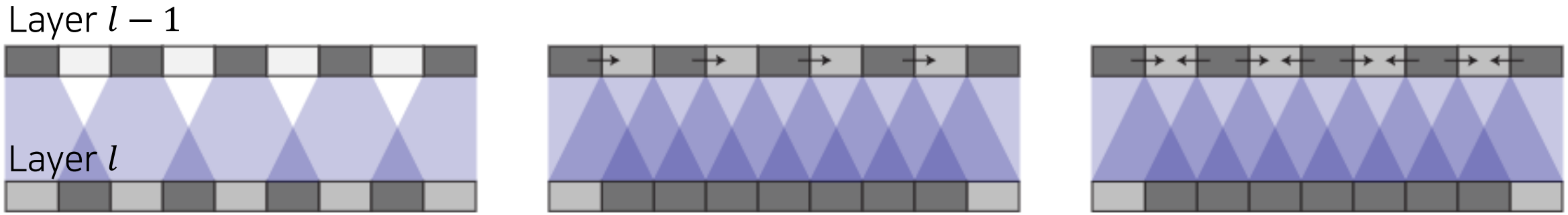
Transposed convolution

[https://distill.pub/2016/deconv-checkerboard/]

2.1 Fully Convolutional Networks (FCN)

Better approaches for upsampling

- **Avoid overlap issues** in transposed convolution
- Decompose into spatial upsampling and feature convolution
  - **{Nearest-neighbor (NN), Bilinear}** interpolation followed by **convolution**



Transposed convolution | NN-resize convolution | Bilinear-resize convolution

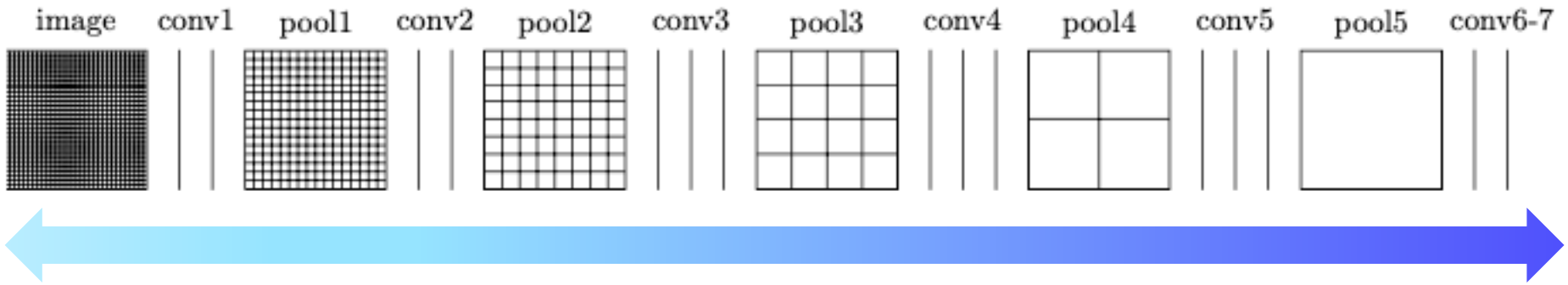
[https://distill.pub/2016/deconv-checkerboard/]

Back to FCN

2.1 Fully Convolutional Networks (FCN)

Adding skip connections for enlarging the score map

[Long et al., CVPR 2015]



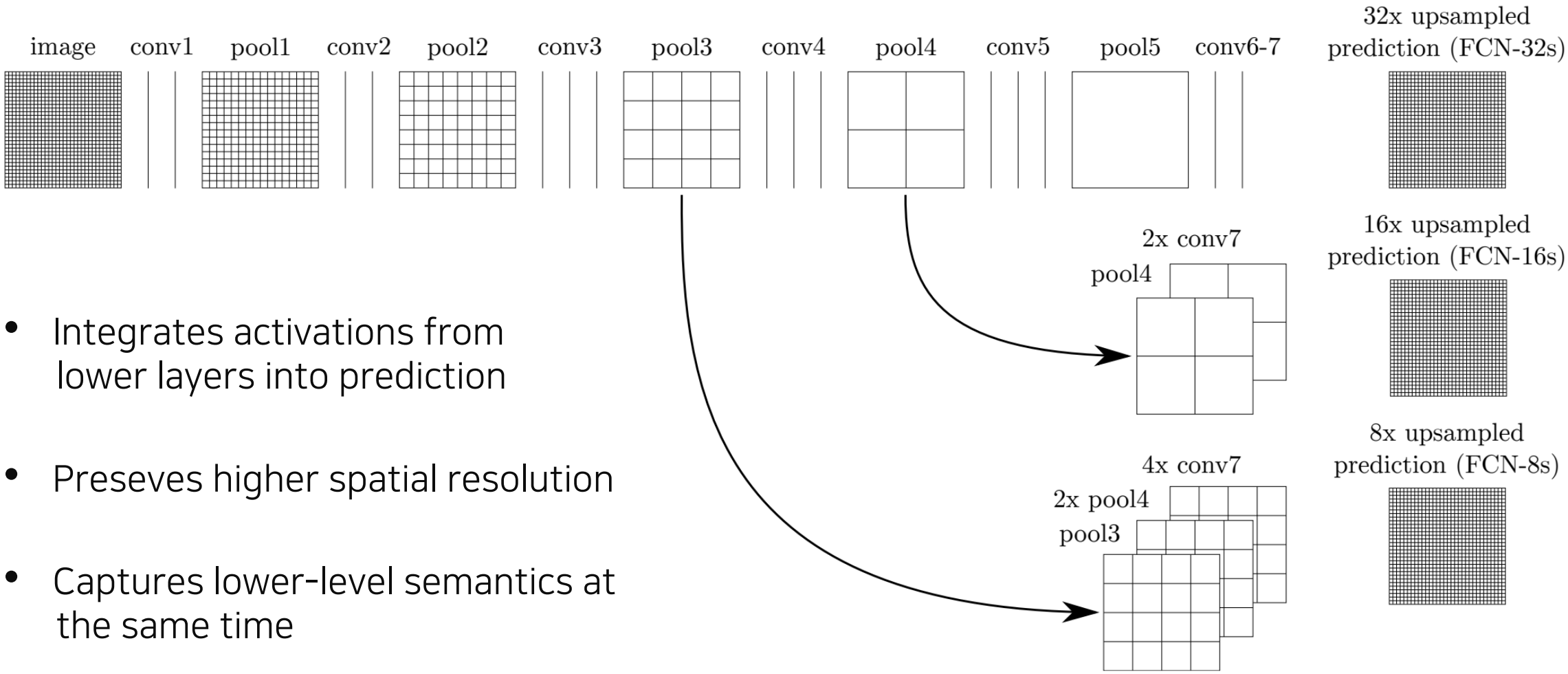
Fine
Low-level
Detail
Local

Coarse
Semantic
Holistic
Global

2.1 Fully Convolutional Networks (FCN)

Adding skip connections for enlarging the score map

[Long et al., CVPR 2015]



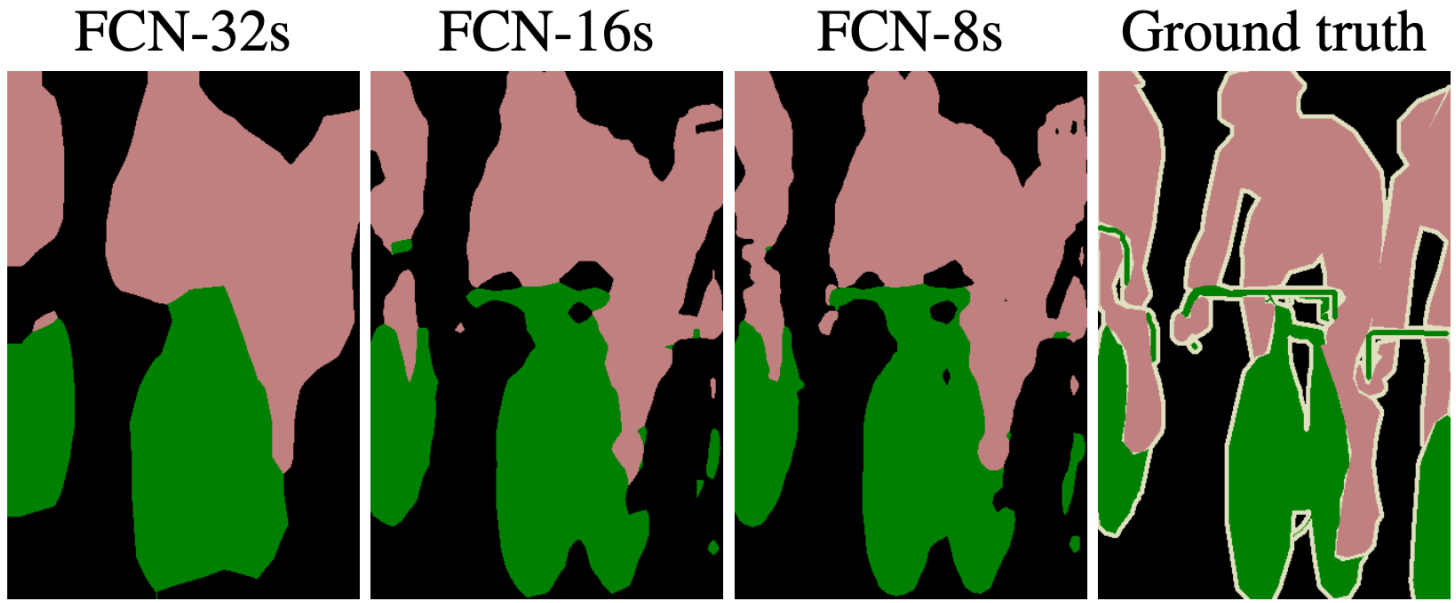
- Integrates activations from lower layers into prediction
- Preseves higher spatial resolution
- Captures lower-level semantics at the same time

2.1 Fully Convolutional Networks (FCN)

Features of FCN

[Long et al., CVPR 2015]

- Faster
  - The end-to-end architecture that does not depend on other hand-crafted components
- Accurate
  - Feature representation and classifiers are jointly optimized

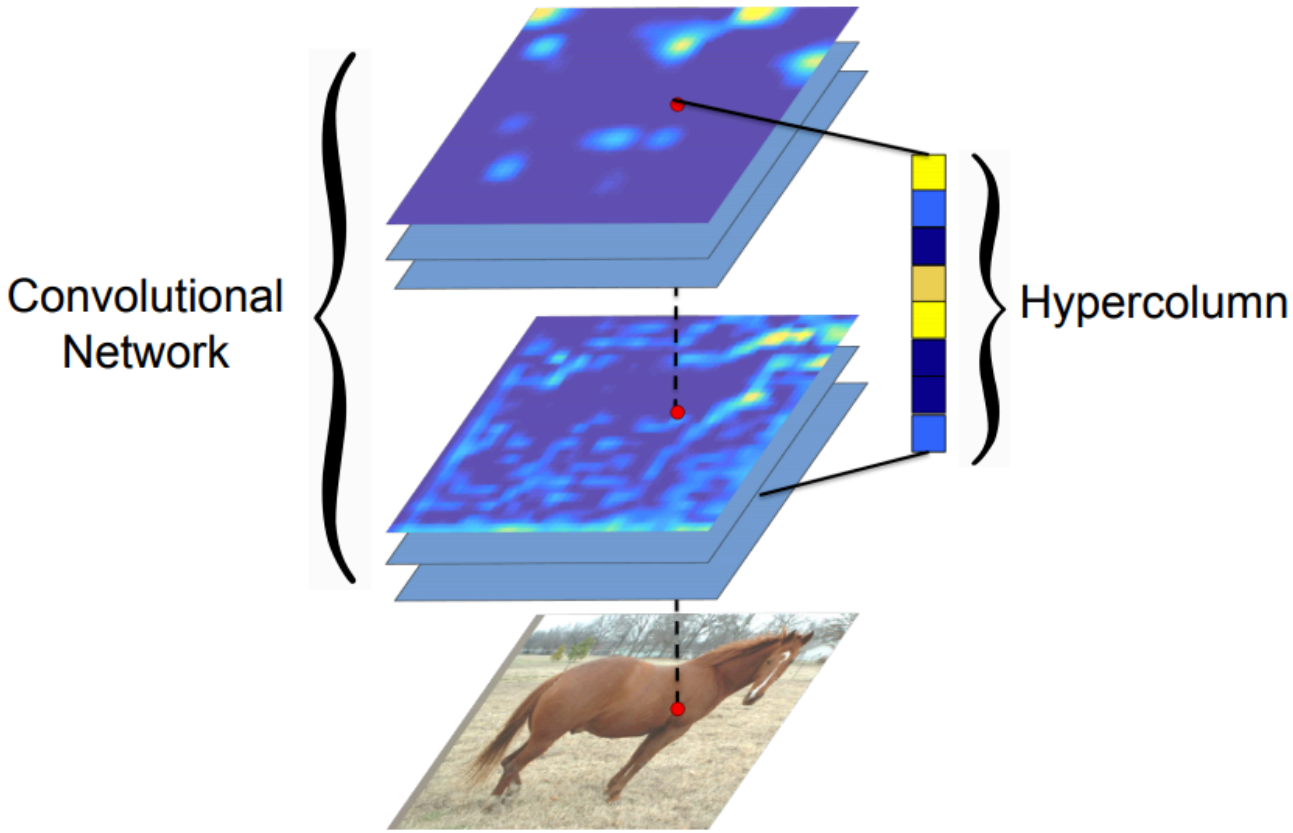


2.2 Hypercolumns for object segmentation

Fully convolutional networks

[Hariharan et al., CVPR 2015]

- CNN layers typically use the output of the last layer as feature representation
  - Too coarse spatially



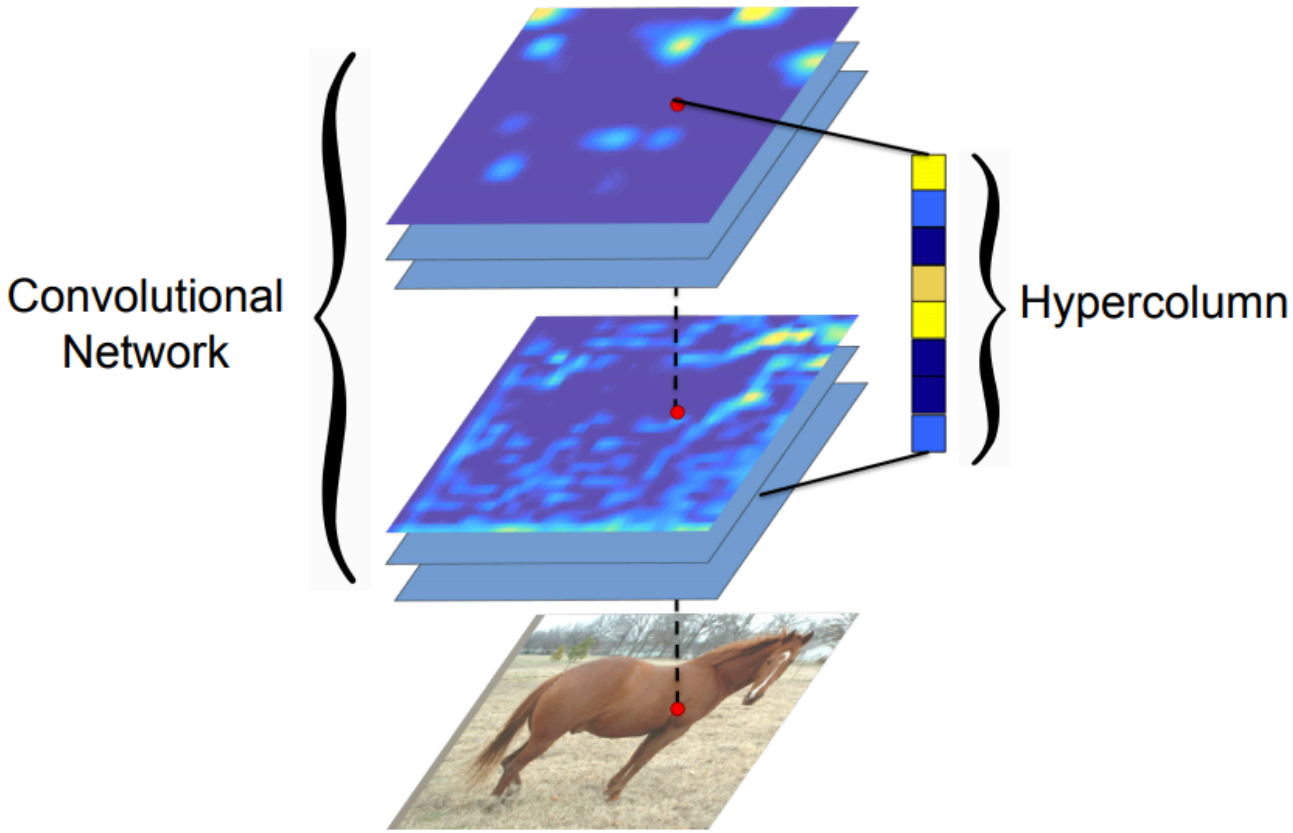
2.2 Hypercolumns for object segmentation

Fully convolutional networks

[Hariharan et al., CVPR 2015]

- CNN layers typically use the output of the last layer as feature representation
  - Too coarse spatially

- Hypercolumn at a pixel is a stacked vector of all CNN units on that pixel
  - Fine localized information is extracted from earlier layers
  - Coarse semantic information is extracted from latter layers

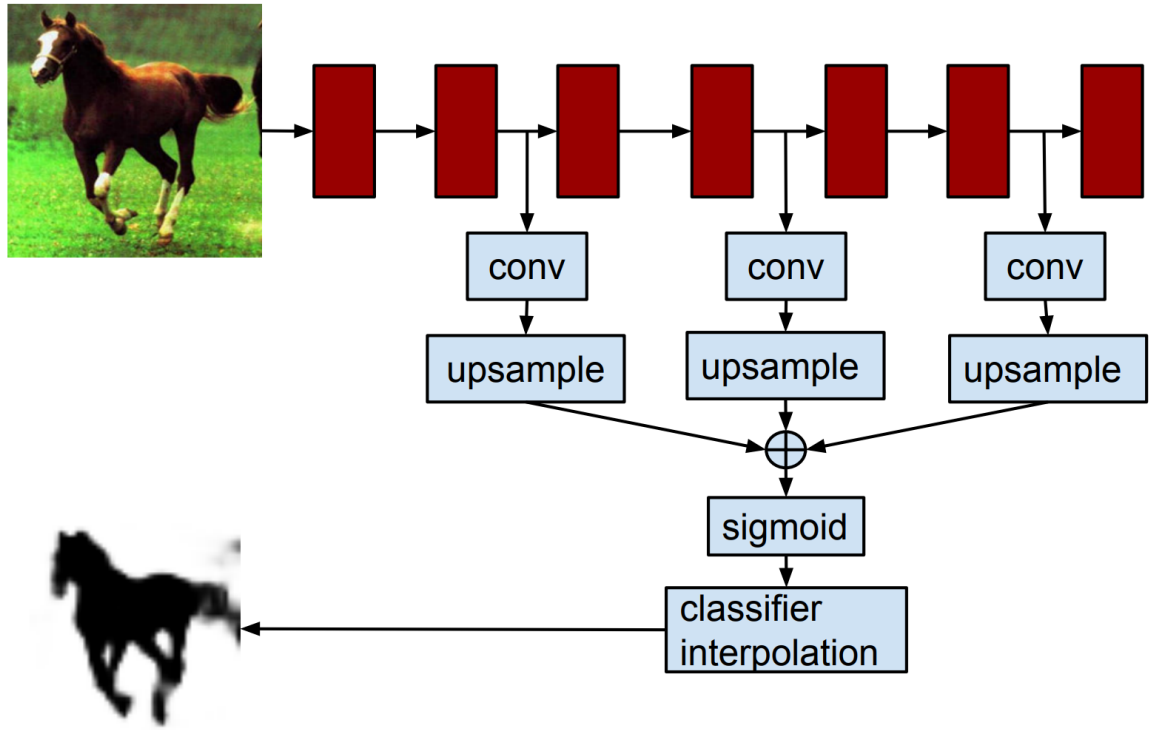


2.2 Hypercolumns for object segmentation

Overall architecture

[Hariharan et al., CVPR 2015]

- (Concurrent work) Very similar to FCN
- Difference: Apply to each bounding box



2.3 U-Net

U-Net

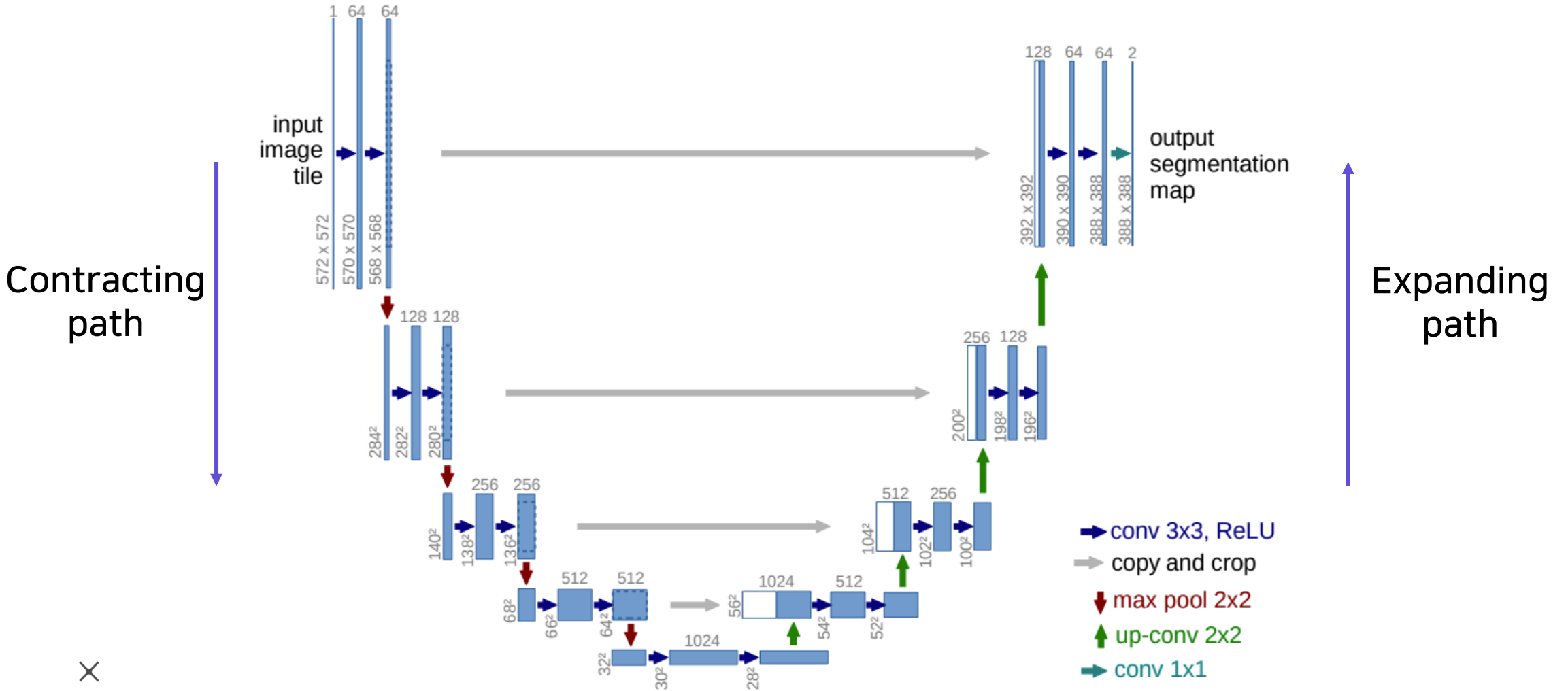
[Ronneberger et al., MICCAI 2015]

- Built upon “fully convolutional networks”
  - Share the same FCN property
- Predict a dense map by concatenating feature maps from contracting path
  - **Similar to skip connections in FCN**
- Yield more precise segmentations

2.3 U-Net

Overall architecture

[Ronneberger et al., MICCAI 2015]

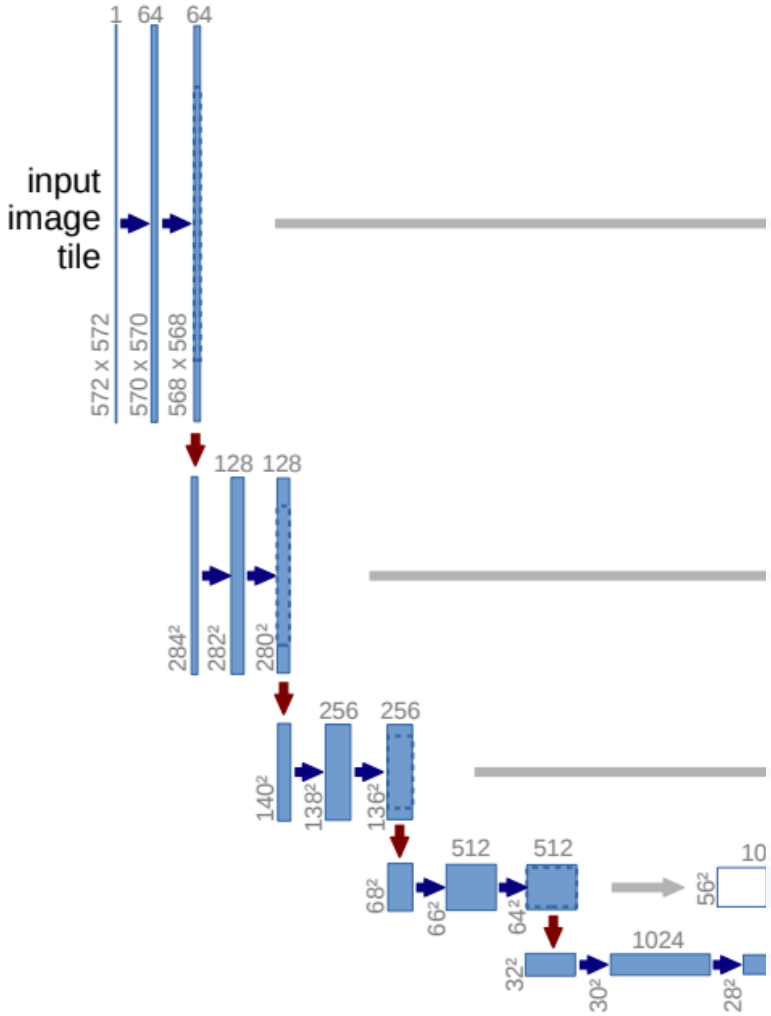


2.3 U-Net

Overall architecture

[Ronneberger et al., MICCAI 2015]

Contracting path



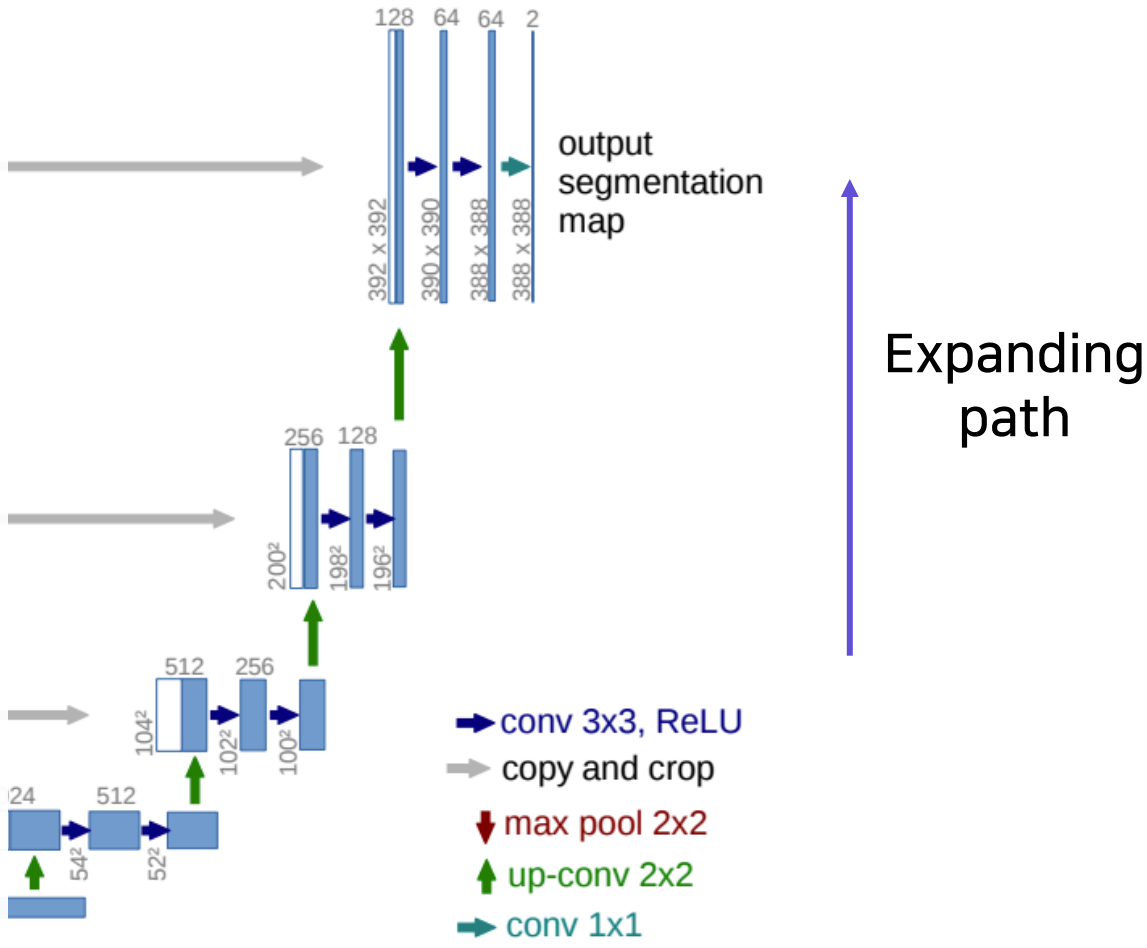
- Contracting Path
  - Repeatedly applying 3x3 convolutions
  - Doubling the number of feature channels
  - Being used to capture holistic context

2.3 U-Net

Overall architecture

[Ronneberger et al., MICCAI 2015]

- Expanding Path
  - Repeatedly applying 2x2 convolutions
  - Halving the number of feature channels
  - Concatenating the corresponding feature maps from the contracting path

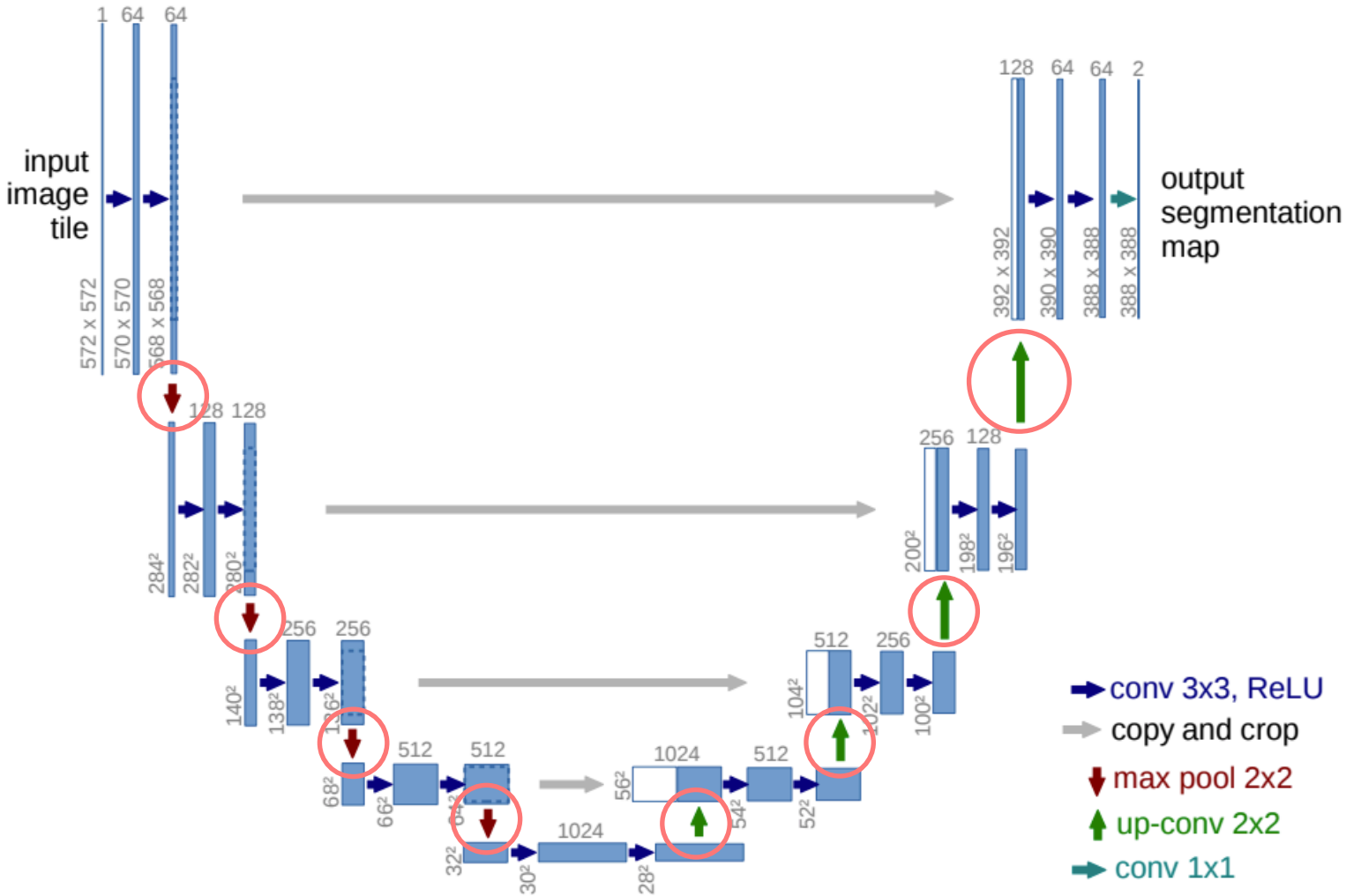


2.3 U-Net

Overall architecture

[Ronneberger et al., MICCAI 2015]

Halve the size of the feature map



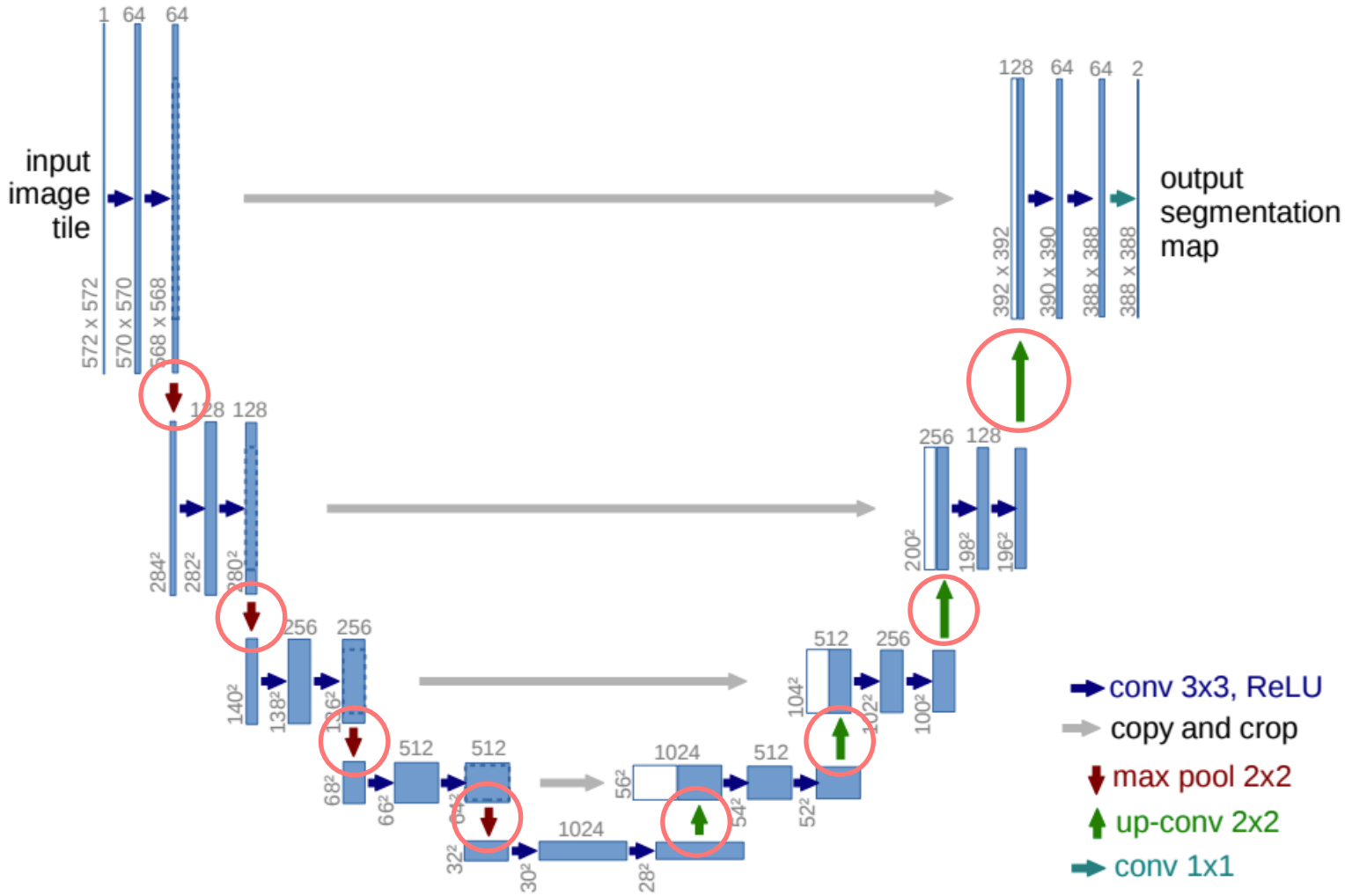
Double the size of the feature map

2.3 U-Net

Overall architecture

[Ronneberger et al., MICCAI 2015]

Double the channel of the feature map

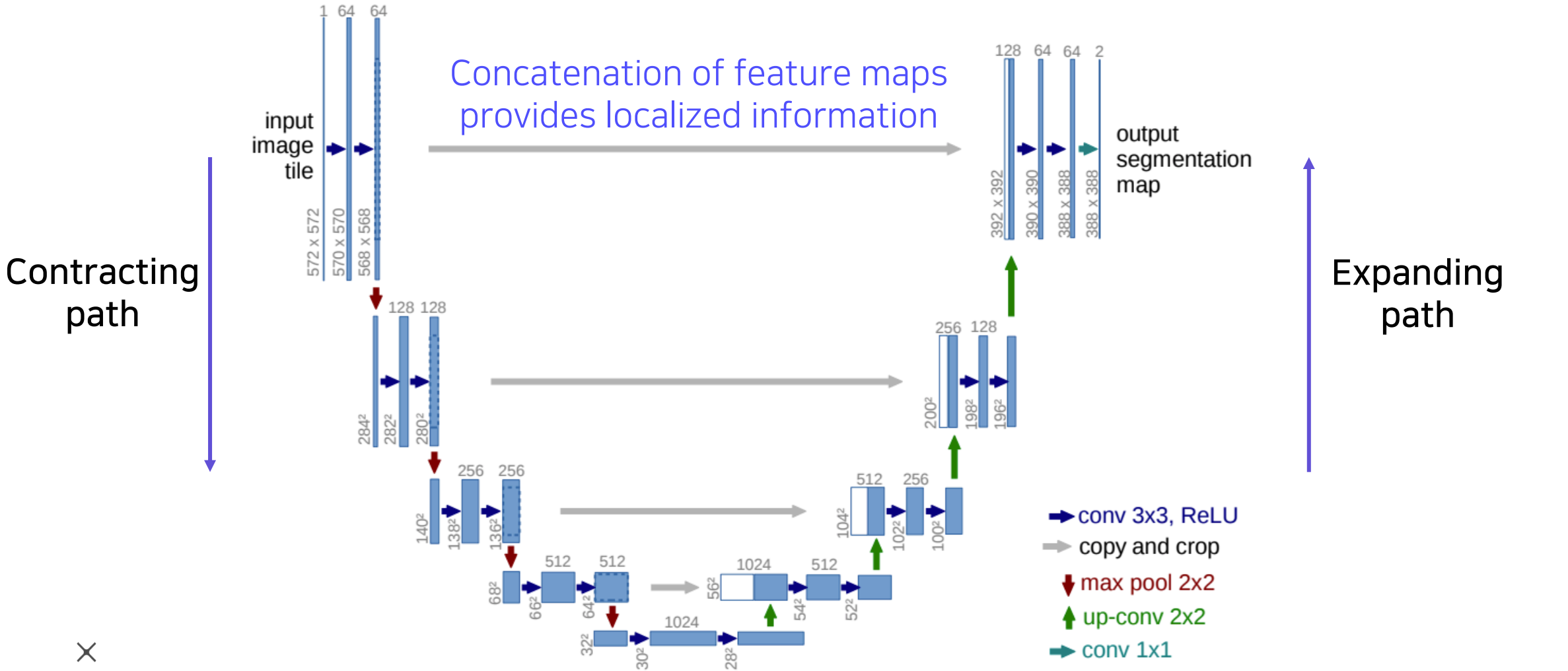


Halve the channel of the feature map

2.3 U-Net

Overall architecture

[Ronneberger et al., MICCAI 2015]

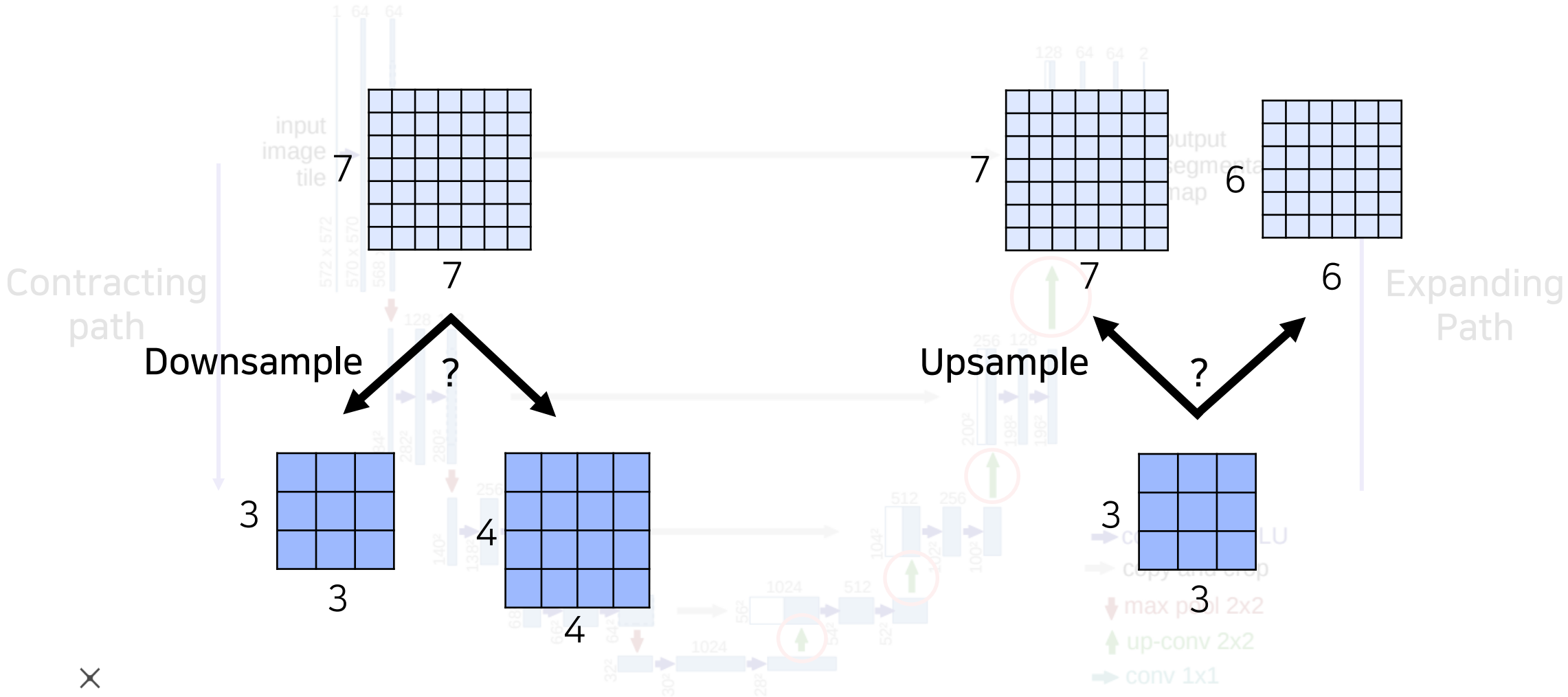


What if the spatial size of the feature map is an odd number?

2.3 U-Net

An even number is required for input and feature sizes

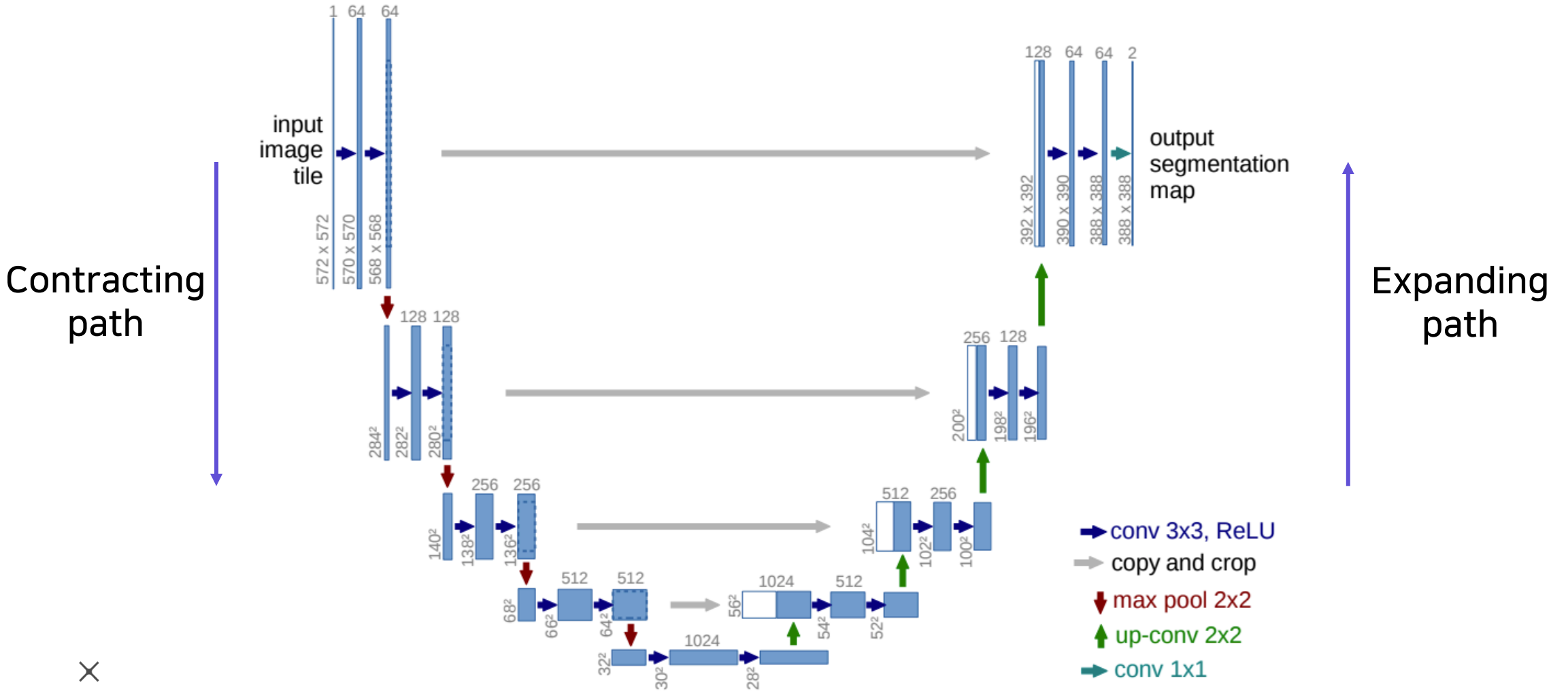
[Ronneberger et al., MICCAI 2015]



2.3 U-Net

PyTorch code for U-Net

[Ronneberger et al., MICCAI 2015]

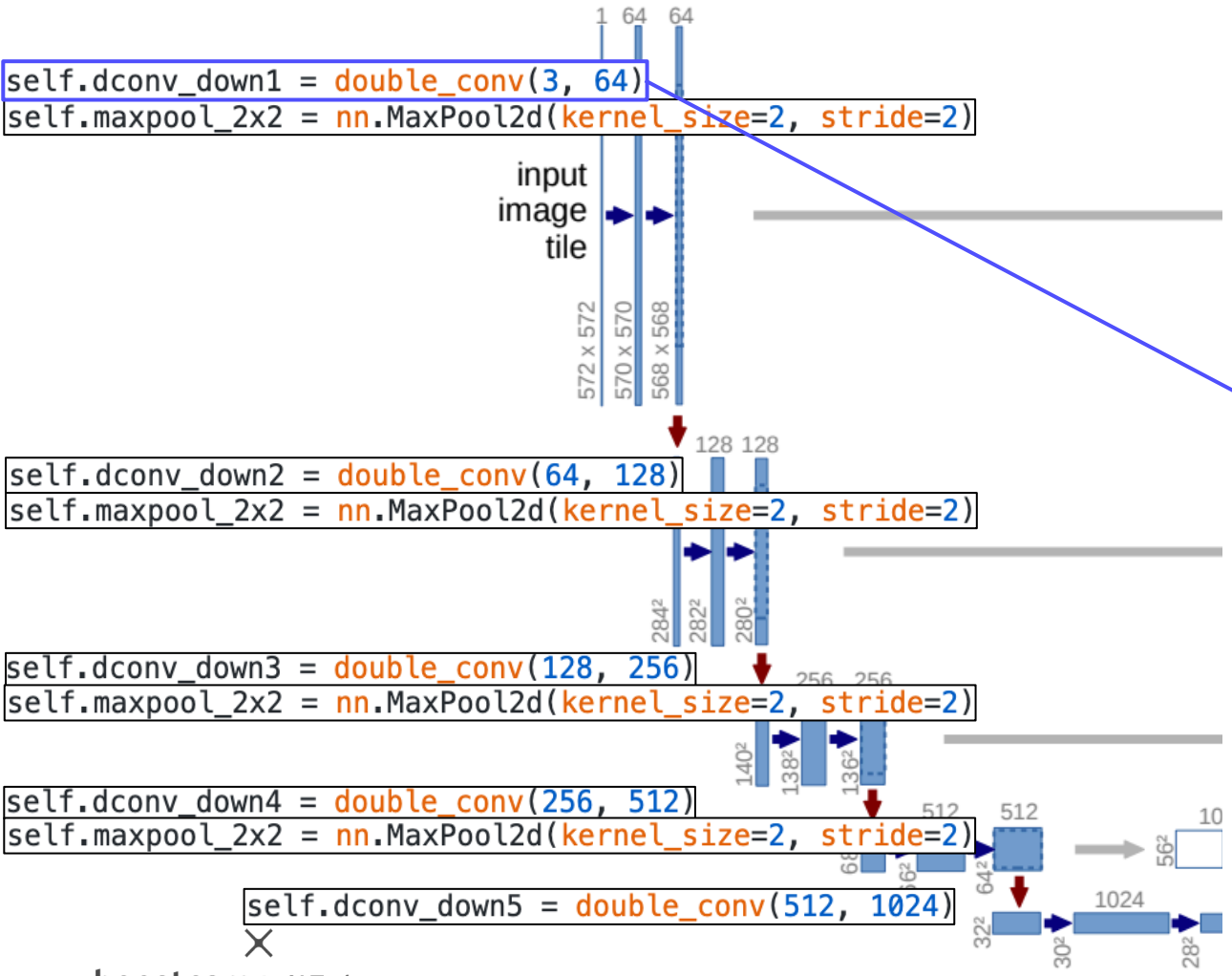


[https://github.com/milesial/Pytorch-UNet/]

2.3 U-Net

PyTorch code for U-Net

[Ronneberger et al., MICCAI 2015]

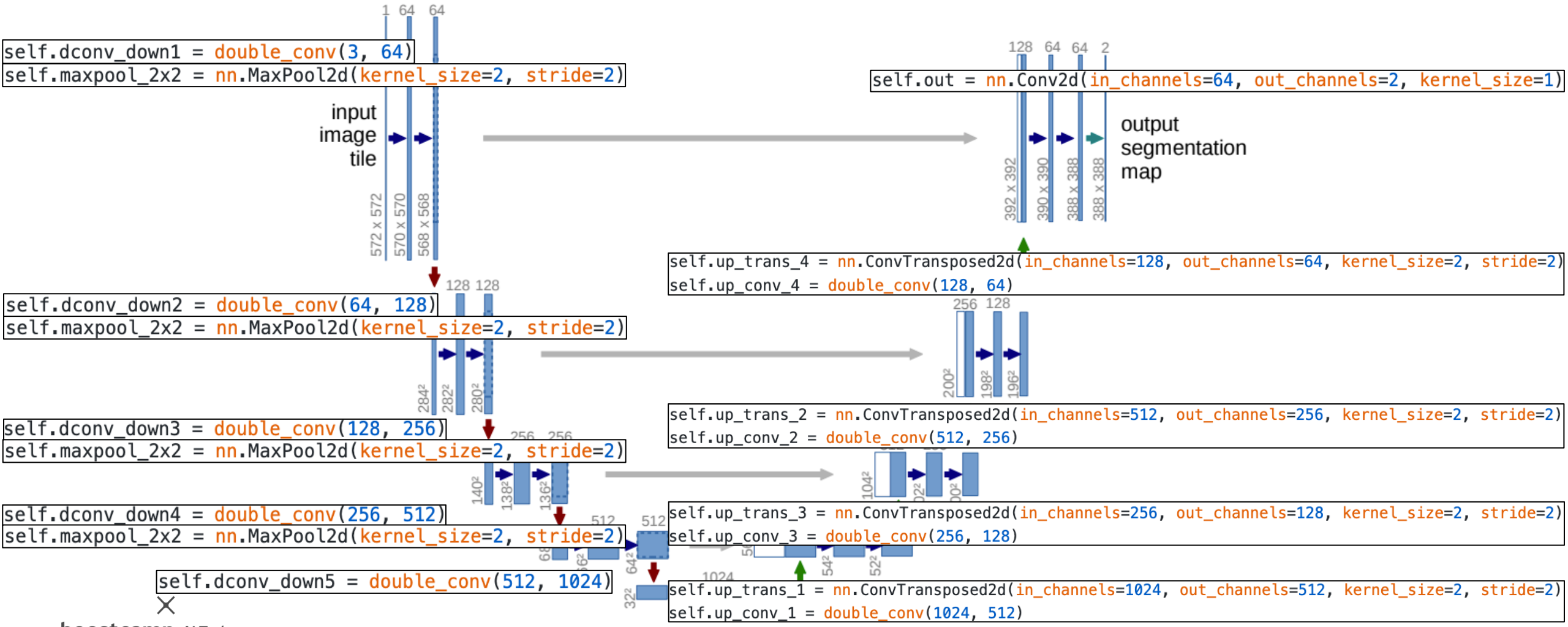



```python
def double_conv(in_channels, out_channels):
    return nn.Sequential(
        nn.Conv2d(in_channels, out_channels, 3),
        nn.ReLU(inplace=True),
        nn.Conv2d(out_channels, out_channels, 3),
        nn.ReLU(inplace=True)
    )
```

2.3 U-Net

PyTorch code for U-Net

[Ronneberger et al., MICCAI 2015]



2.4 DeepLab

- DeepLab v1 (2015)
  : Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. ICLR 2015.

- DeepLab v2 (2017)
  : DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. TPAMI 2017.

- DeepLab v3 (2017)
  : Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv 2017.

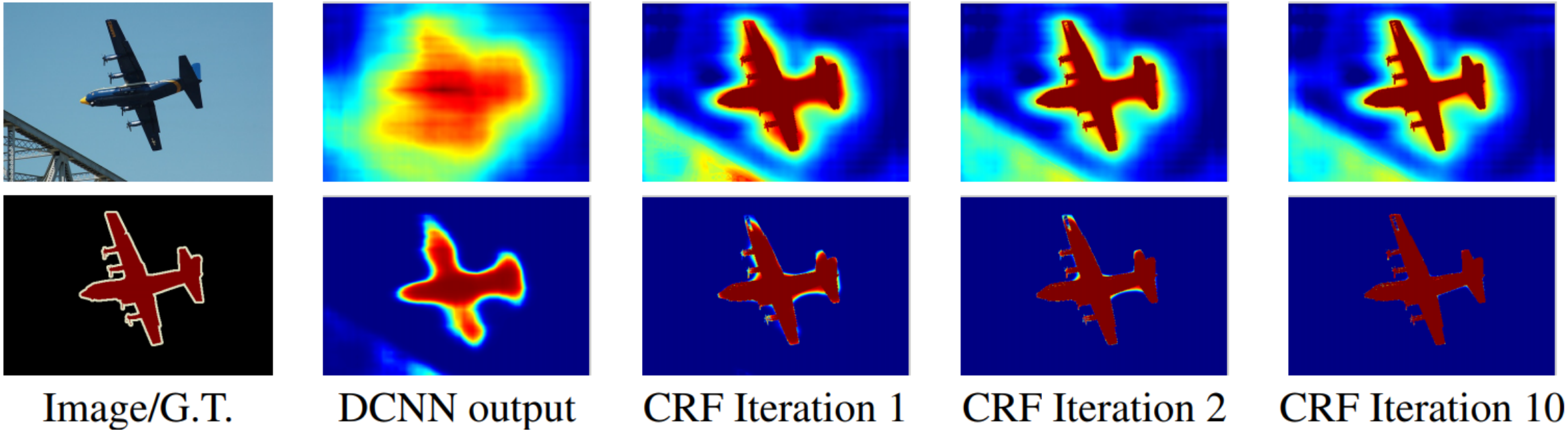
- DeepLab v3+ (2018)
  : Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. ECCV 2018.

2.4 DeepLab

Conditional Random Fields (CRFs)

[Chen et al., ICLR 2015]

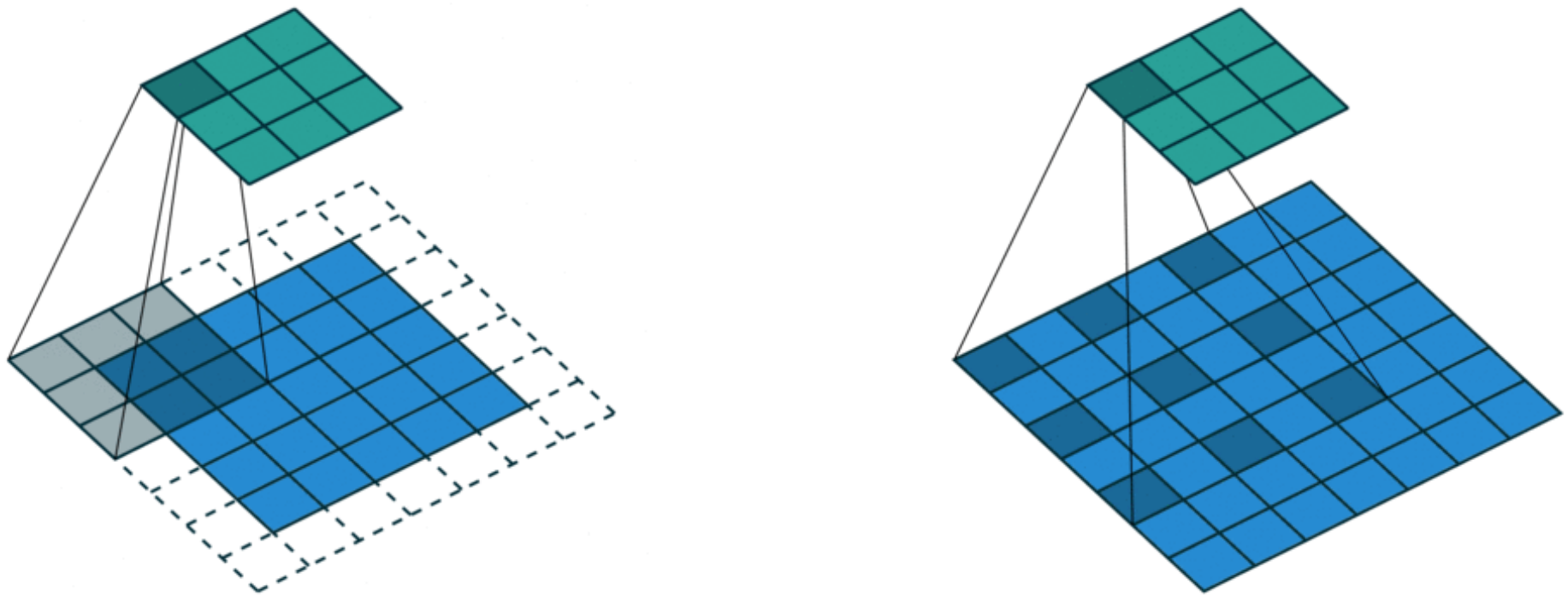
- CRF post-processes a segmentation map to be refined to follow image boundaries
- 1st row: score map (before softmax) / 2nd row : belief map (after softmax)



2.4 DeepLab

Dilated convolution

- Atrous convolution
- Inflate the kernel by inserting spaces between the kernel element (Dilation factor)
- Enable exponential expansion of the receptive field



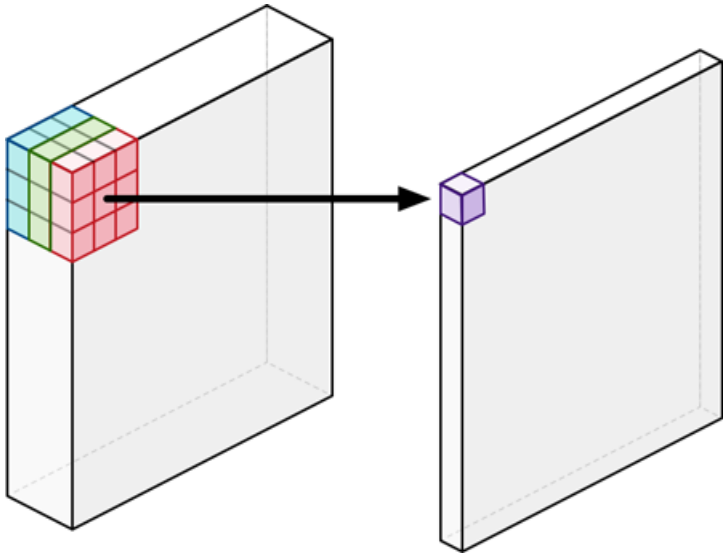
[https://github.com/vdumoulin/conv_arithmetic]

2.4 DeepLab

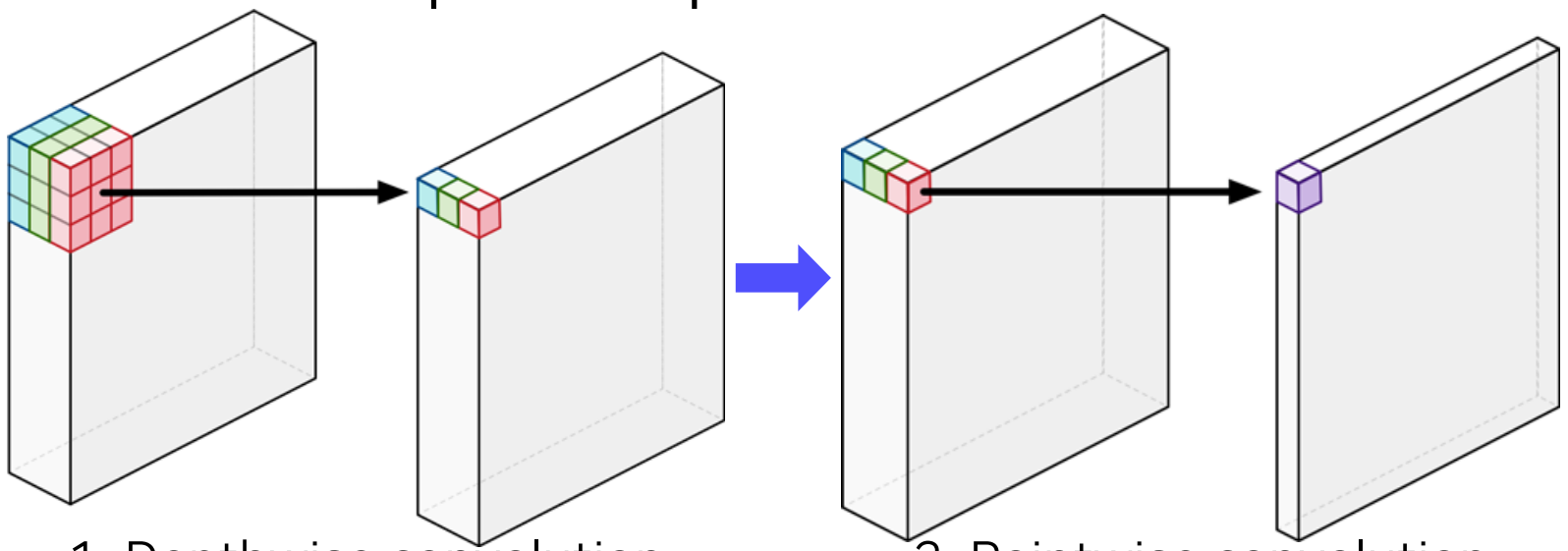
Depthwise separable convolution (proposed by Howard et al.)

- No. parameters
  - Standard conv.: $D_K^2 M N D_F^2$
  - Depthwise separable conv.: $D_K^2 M D_F^2 + M N D_F^2$

Standard convolution



Depthwise separable convolution



1. Depthwise convolution

2. Pointwise convolution

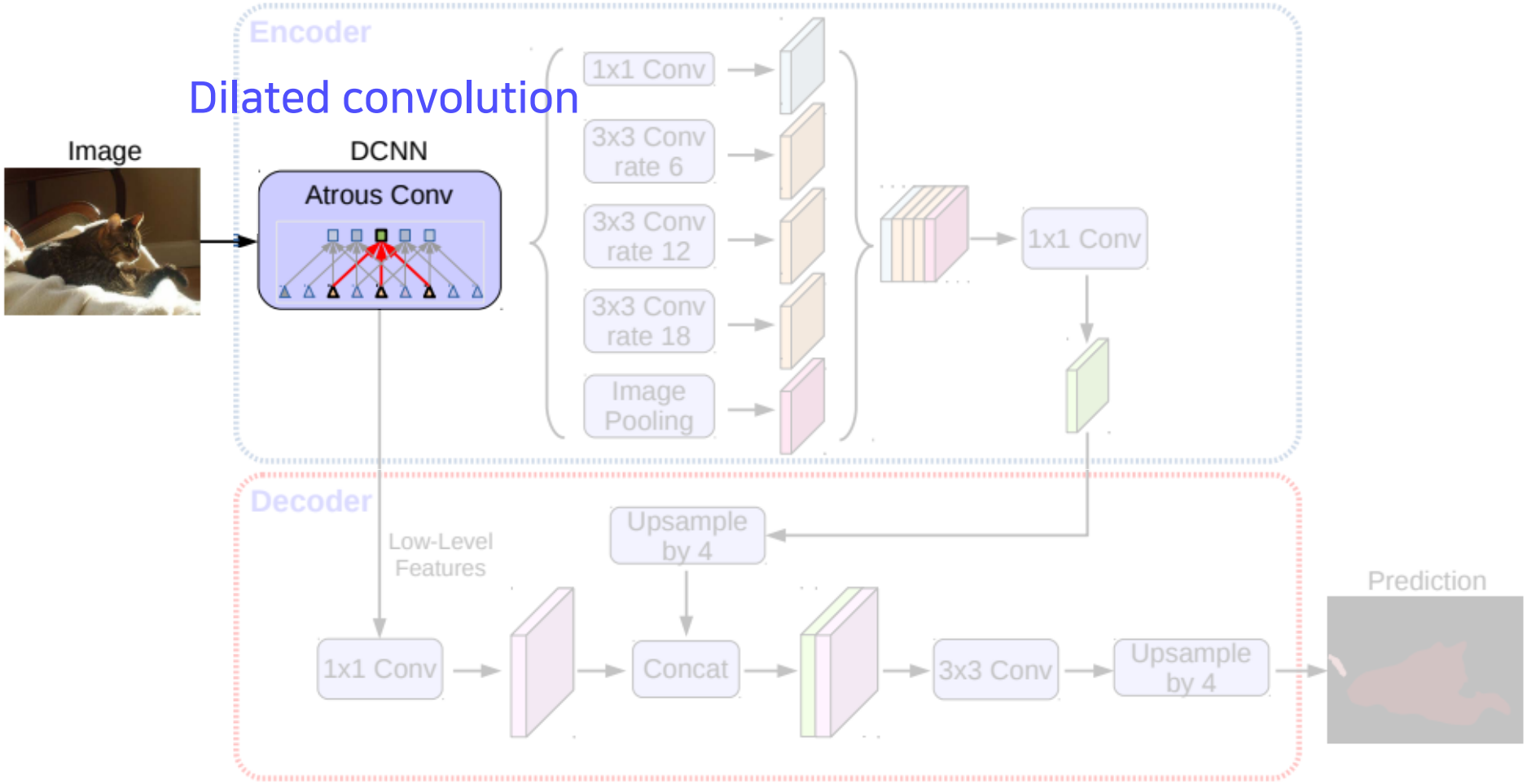
[http://machinethink.net/blog/googles-mobile-net-architecture-on-iphone/]]

* D_K / D_F : Kernel / feature map size
  M / N : input / output channels

2.4 DeepLab

Deeplab v3+

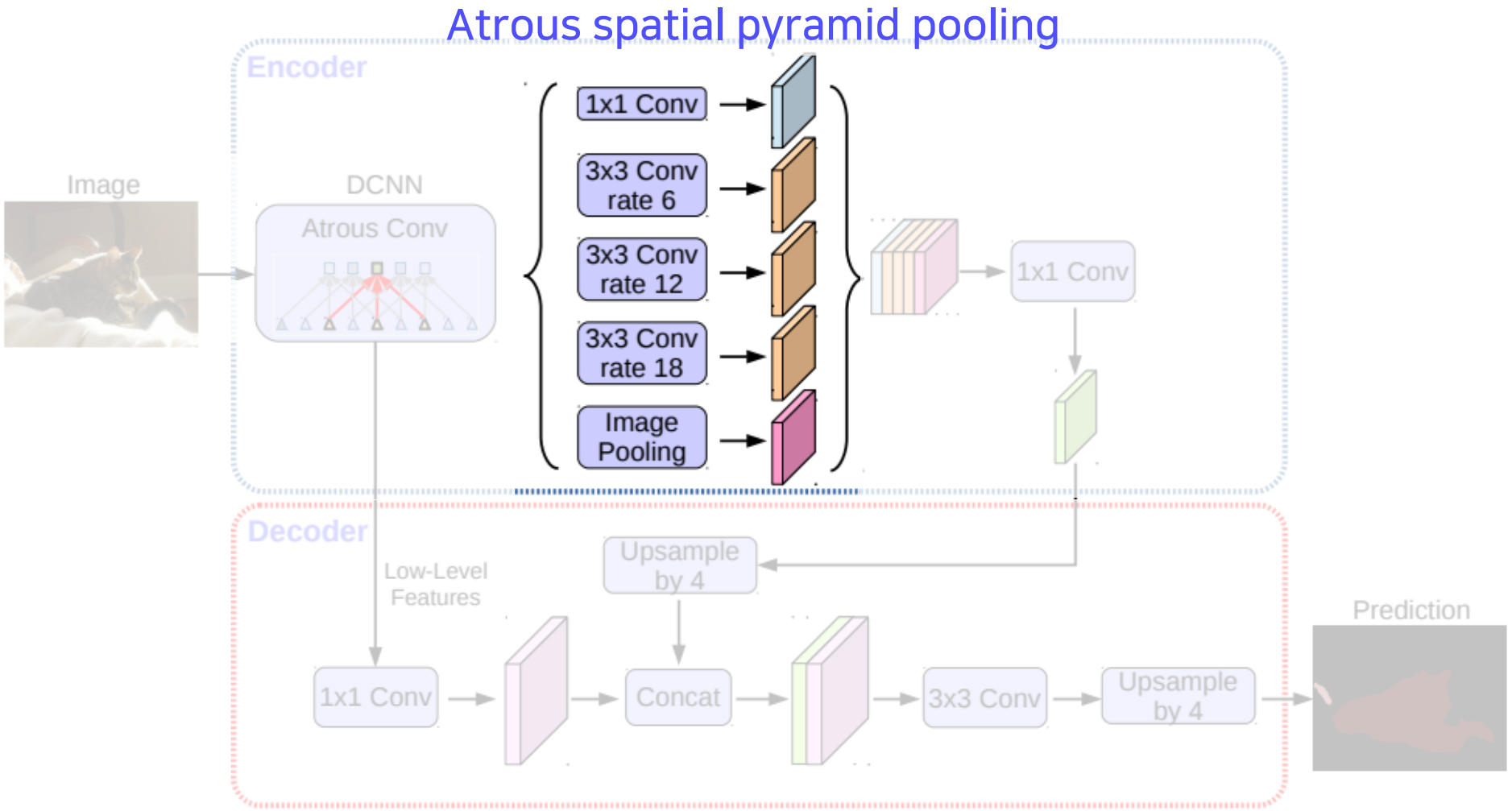
[Chen et al., ECCV 2018]



2.4 DeepLab

Deeplab v3+

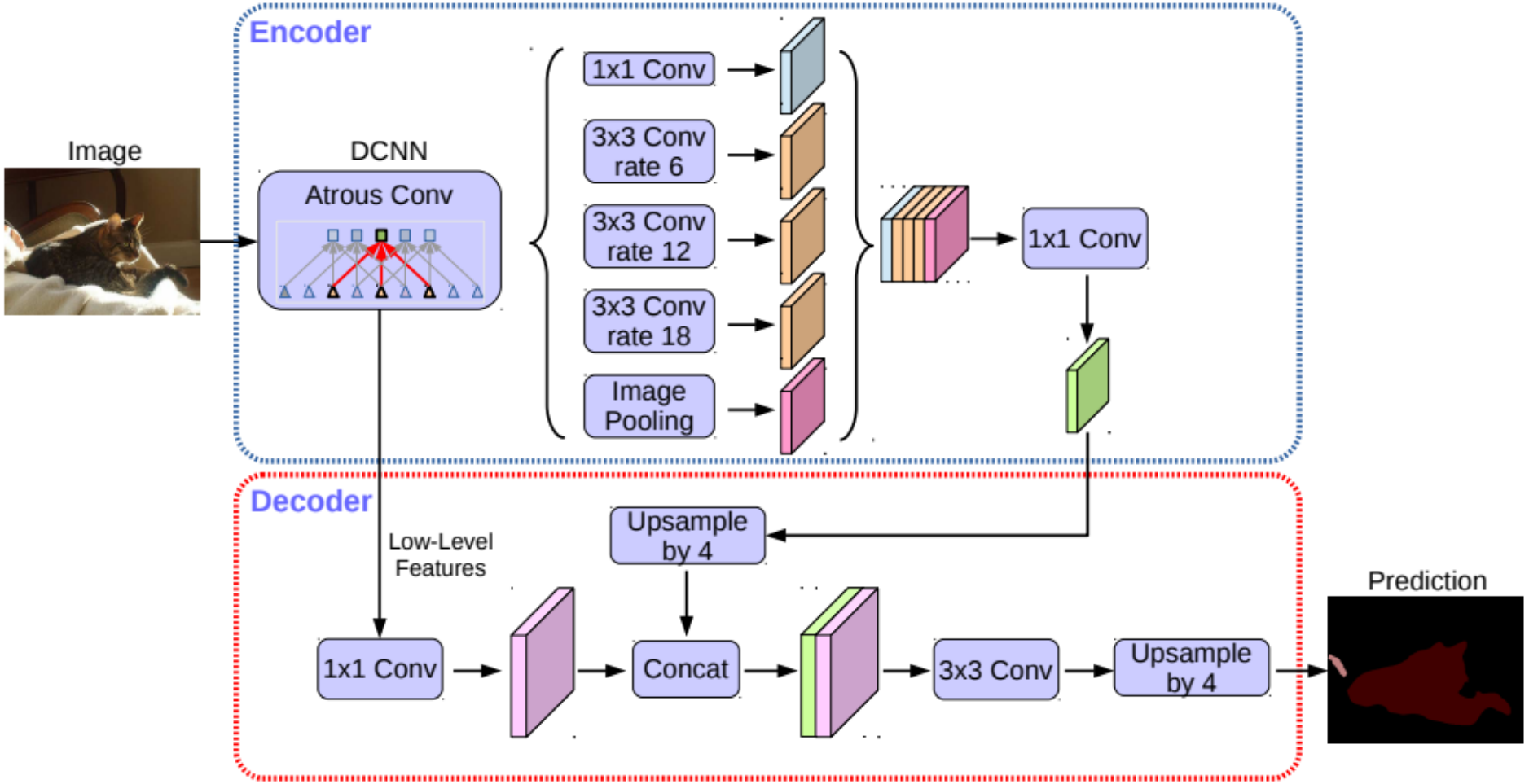
[Chen et al., ECCV 2018]



2.4 DeepLab

Deeplab v3+

[Chen et al., ECCV 2018]



Reference

1. Semantic segmentation

- Chen et al., Rethinking Atrous Convolution for Semantic Image Segmentation, arXiv 2017
- Novikov et al., Fully Convolutional Architectures for Multi-Class Segmentation in Chest Radiographs, T-MI 2016
- Aksoy et al., Semantic Soft Segmentation, SIGGRAPH 2018

2. Semantic segmentation architectures

- Long et al., Fully Convolutional Networks for Semantic Segmentation, CVPR 2015
- Hariharan et al., Hypercolumns for Object Segmentation and Fine-Grained localization, CVPR 2015
- Ronneberger et al., U-Net: Convolutional Networks for Biomedical Image Segmentation, MICCAI 2015
- Chen et al., Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs, ICLR 2015
- Howard et al., MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, arXiv 2017
- Chen et al., Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation, ECCV 2018

End of Document
Thank You.

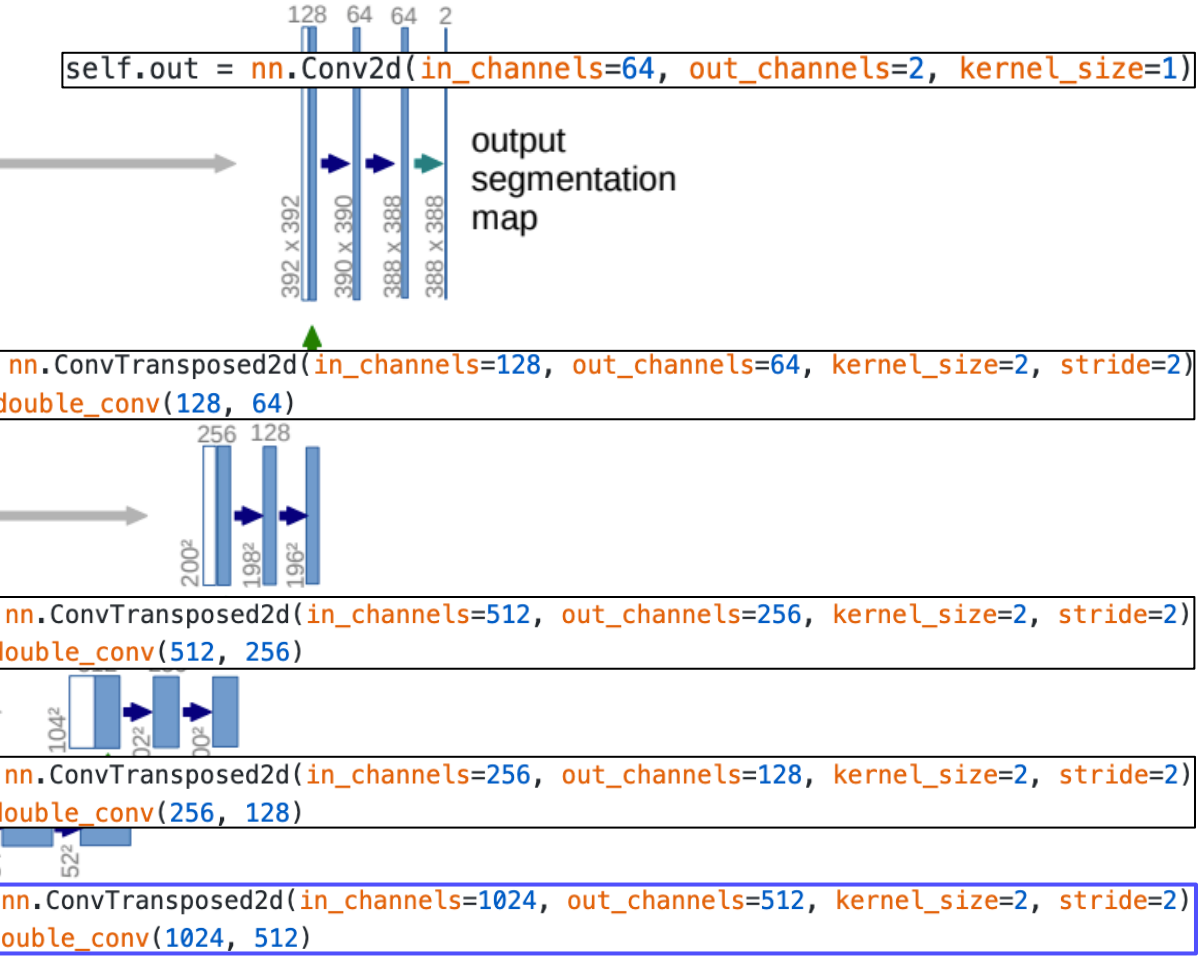
Appendix (2.3 U-Net)

PyTorch code for U-Net

[Ronneberger et al., MICCAI 2015]

```
x = self.up_trans_1(x_out) #x_out = output of contracting path
y = crop_img(x_c,x) #x_c = feature map from contracting path
x = self.up_conv_1(torch.cat([x,y],1))
```

crop_img(x, y)
: crop x_c as same size as x



[https://github.com/milesial/Pytorch-UNet/]