Deep Learning for Vision & Language

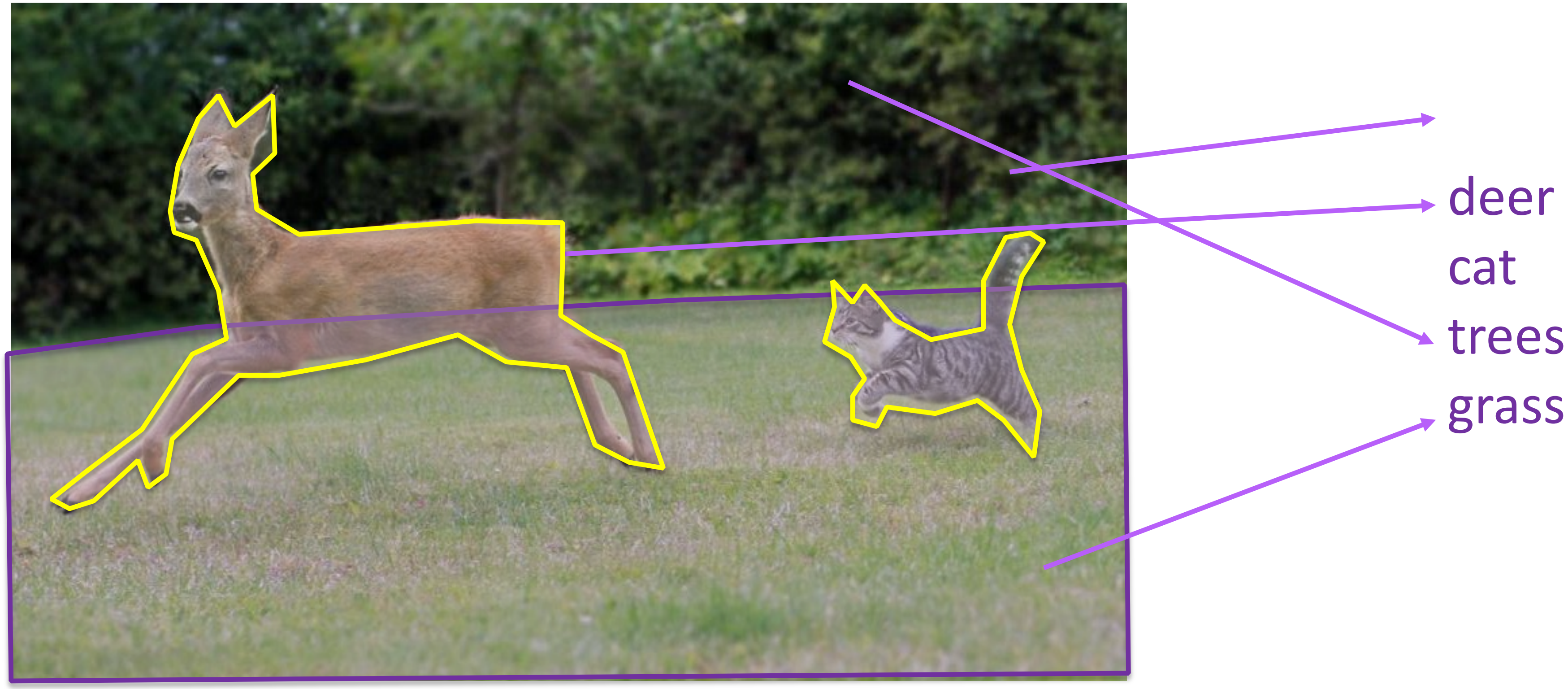
Convolutional Neural Networks for Segmentation

RICE UNIVERSITY

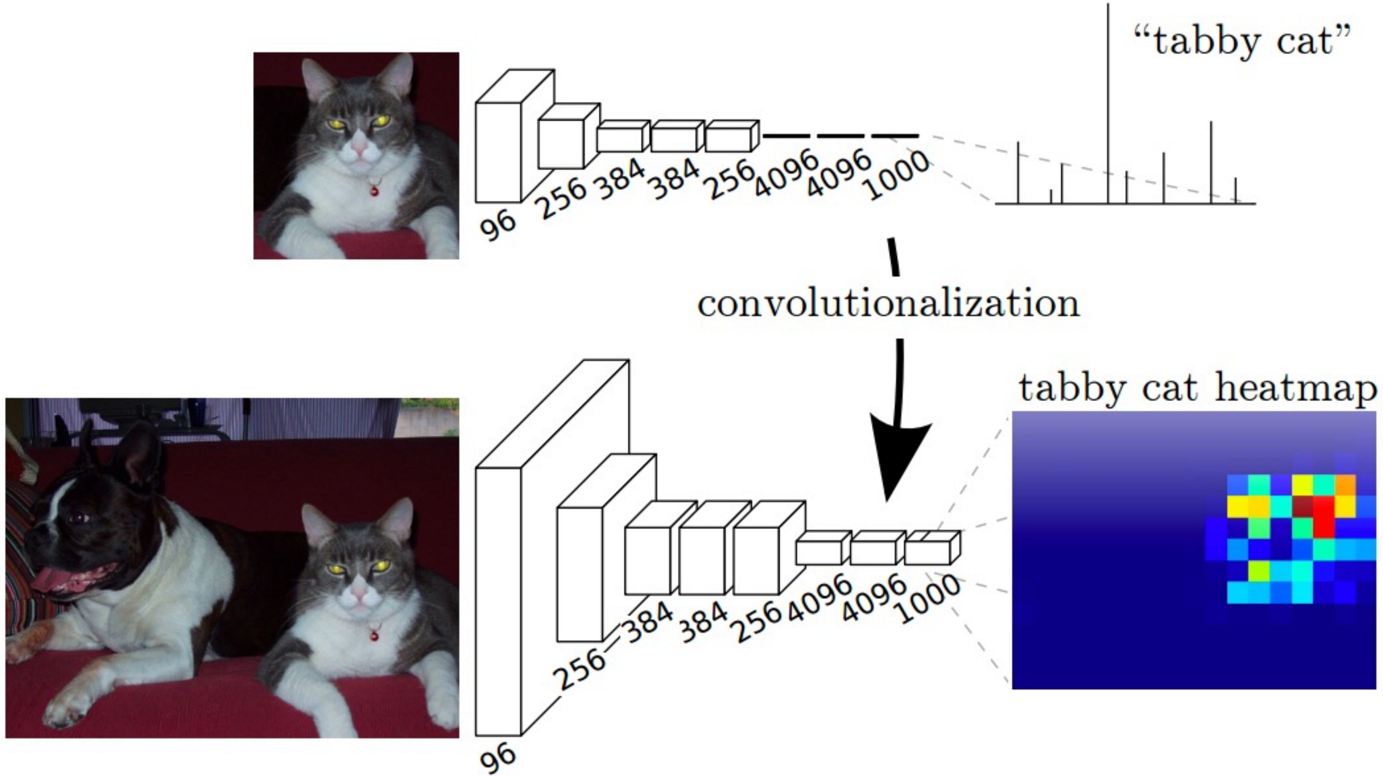
Final Project

- PDF Project report (4 pages)
  - Template: https://www.overleaf.com/read/fyjndmqhghjg
- Link to source code / github or google drive or dropbox links to code.
- 5 slides presenting your work -- ideally a video of you walking me through your project in case I have trouble running it or understanding your report
  [Motivation] [Problem Setup] [Model] [Experiments] [Results]

Semantic Segmentation / Image Parsing



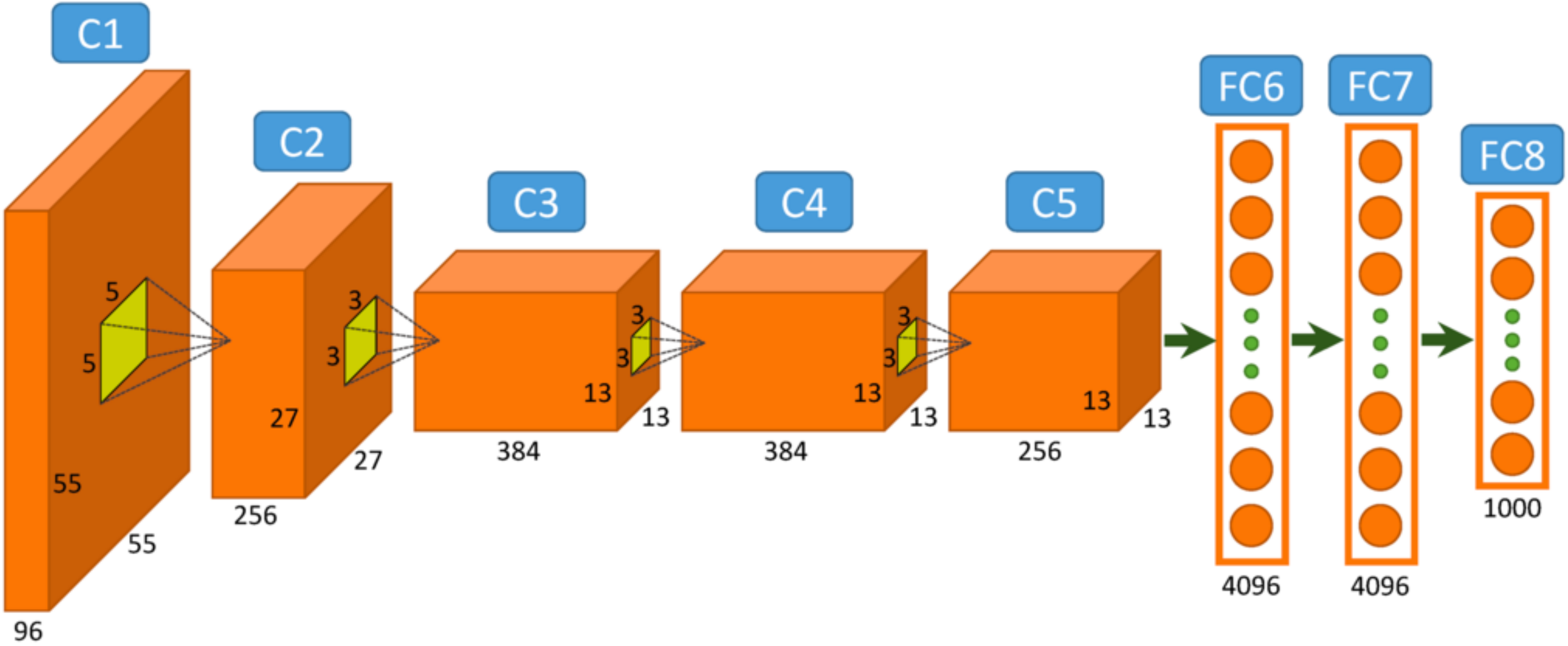
Idea 1: Convolutionalization



However resolution of the segmentation map is low.

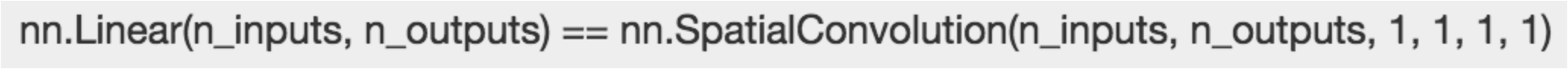
https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

Alexnet

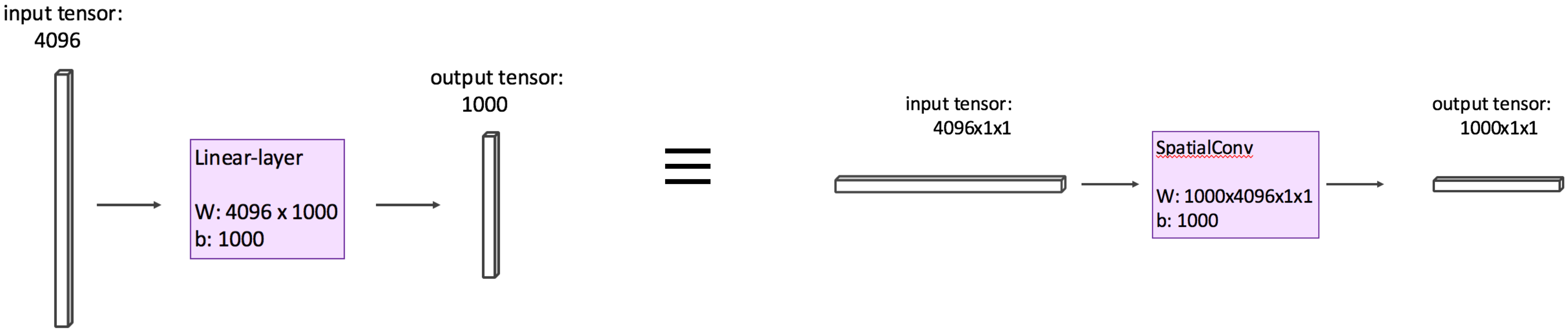


https://www.saagie.com/fr/blog/object-detection-part1

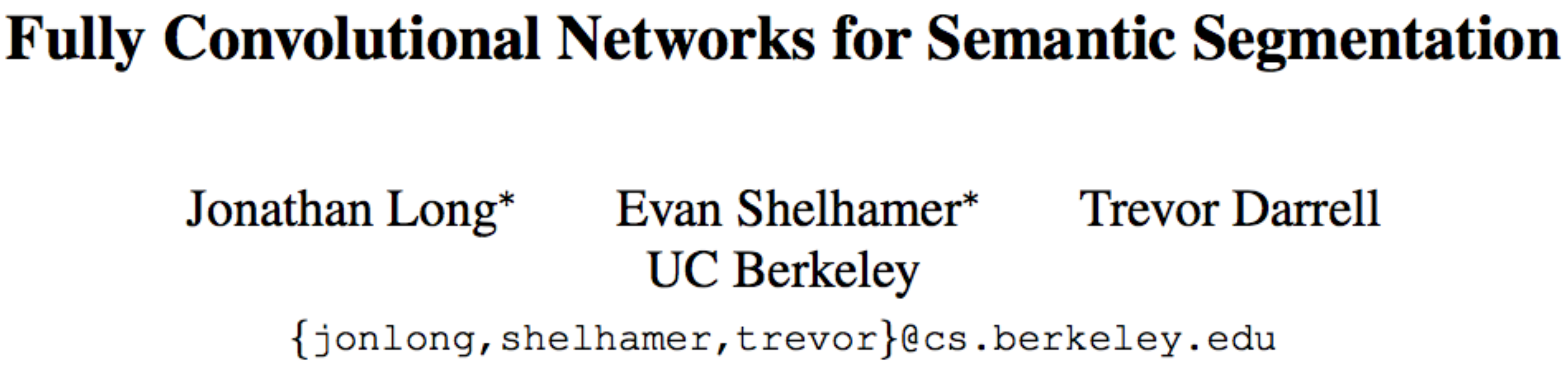
Idea 1: Convolutionalization



```
nn.Linear(n_inputs, n_outputs) == nn.SpatialConvolution(n_inputs, n_outputs, 1, 1, 1, 1)
```

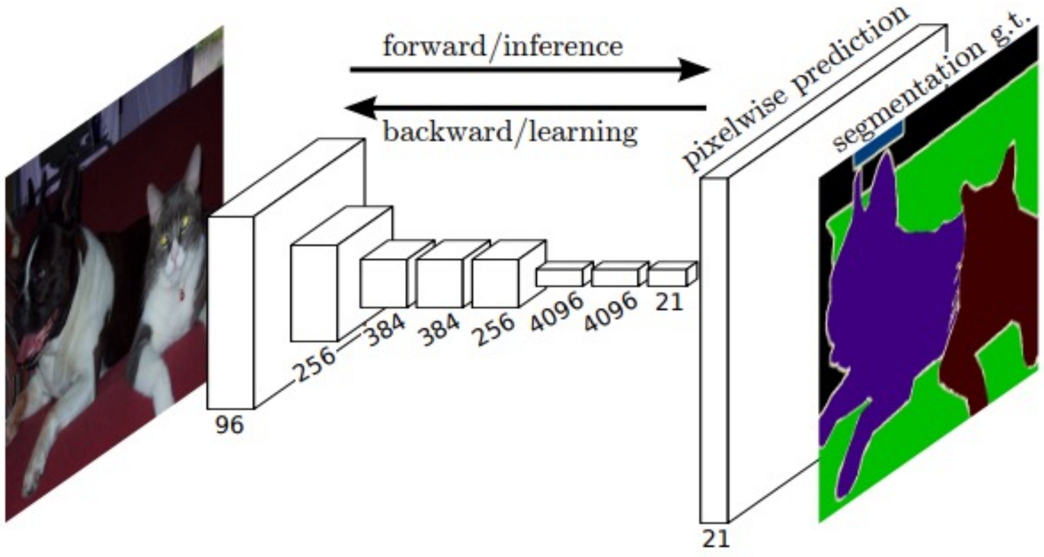


Fully Convolutional Networks (CVPR 2015)

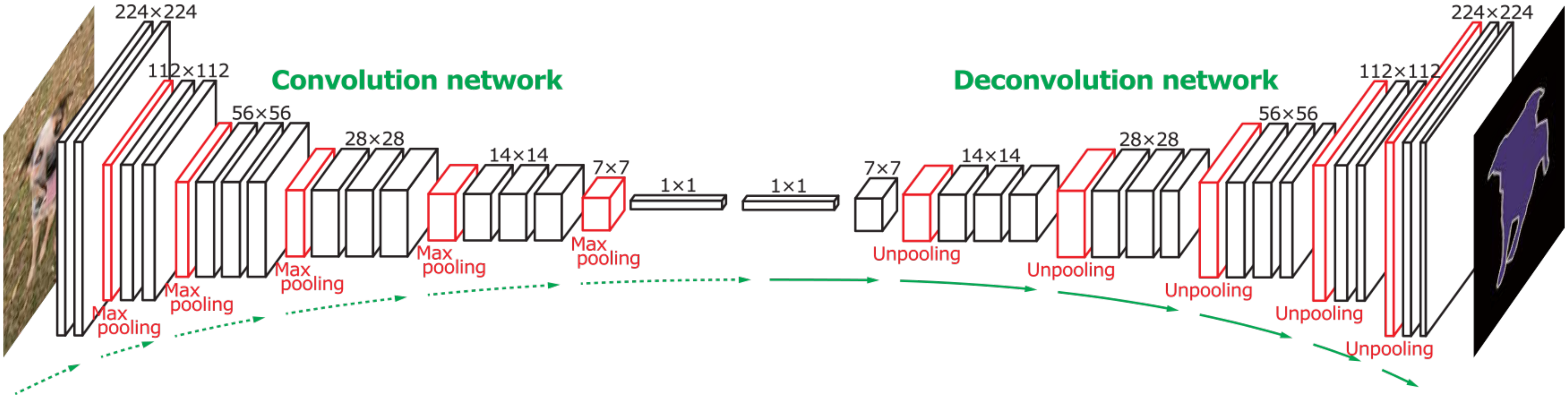


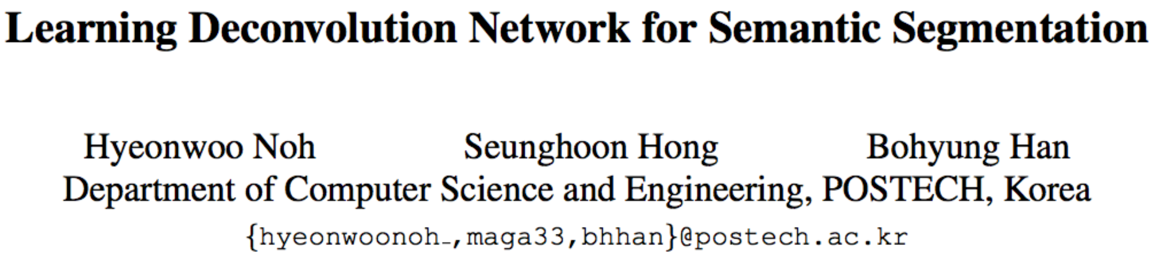
Fully Convolutional Networks for Semantic Segmentation

Jonathan Long* Evan Shelhamer* Trevor Darrell
UC Berkeley
{jonlong,shelhamer,trevor}@cs.berkeley.edu



Idea 2: Up-sampling Convolutions or ”Deconvolutions”



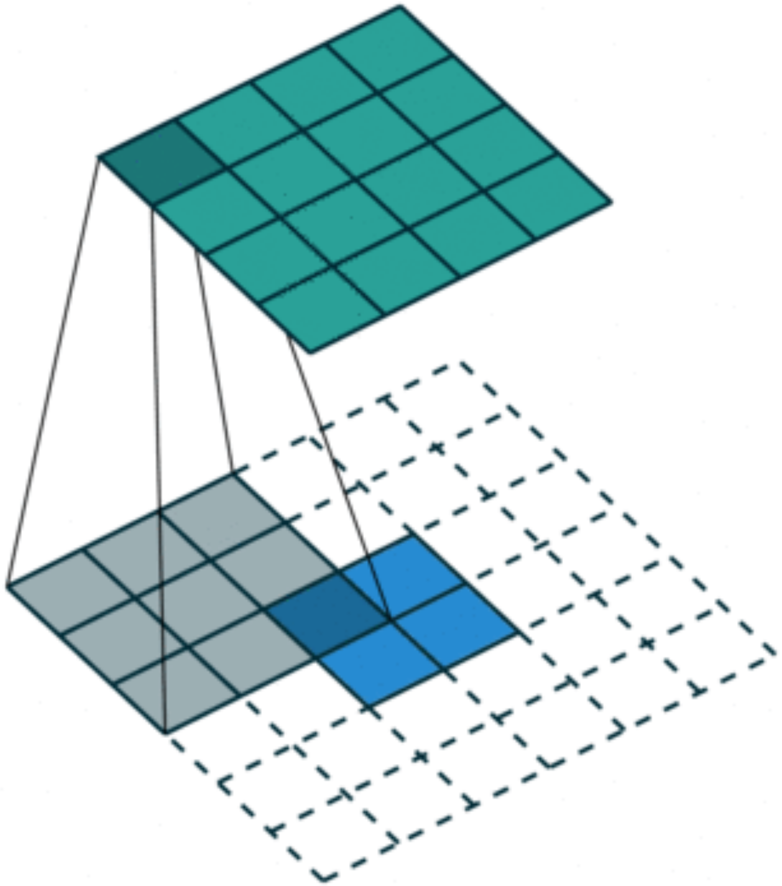


Learning Deconvolution Network for Semantic Segmentation

Hyeonwoo Noh   Seunghoon Hong   Bohyung Han
Department of Computer Science and Engineering, POSTECH, Korea
{hyeonwoonoh_,maga33,bhhan}@postech.ac.kr

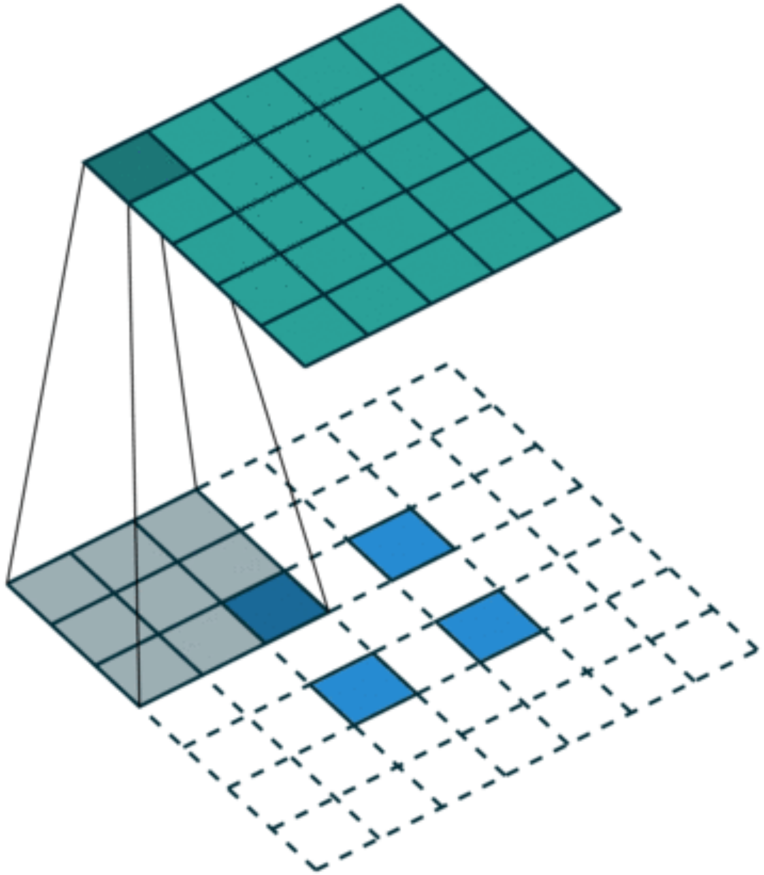
http://cvlab.postech.ac.kr/research/deconvnet/

Idea 2: Up-sampling Convolutions or ”Deconvolutions”



https://github.com/vdumoulin/conv_arithmetic

Idea 2: Up-sampling Convolutions or ”Deconvolutions”



https://github.com/vdumoulin/conv_arithmetic

Idea 2: Up-sampling Convolutions or ”Deconvolutions”

Deconvolutional Layers

Upconvolutional Layers

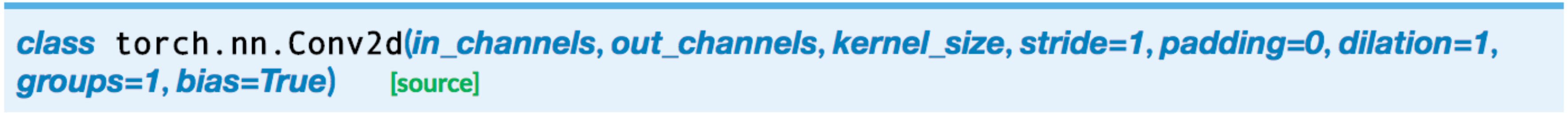
Backwards Strided
Convolutional Layers

Fractionally Strided
Convolutional Layers

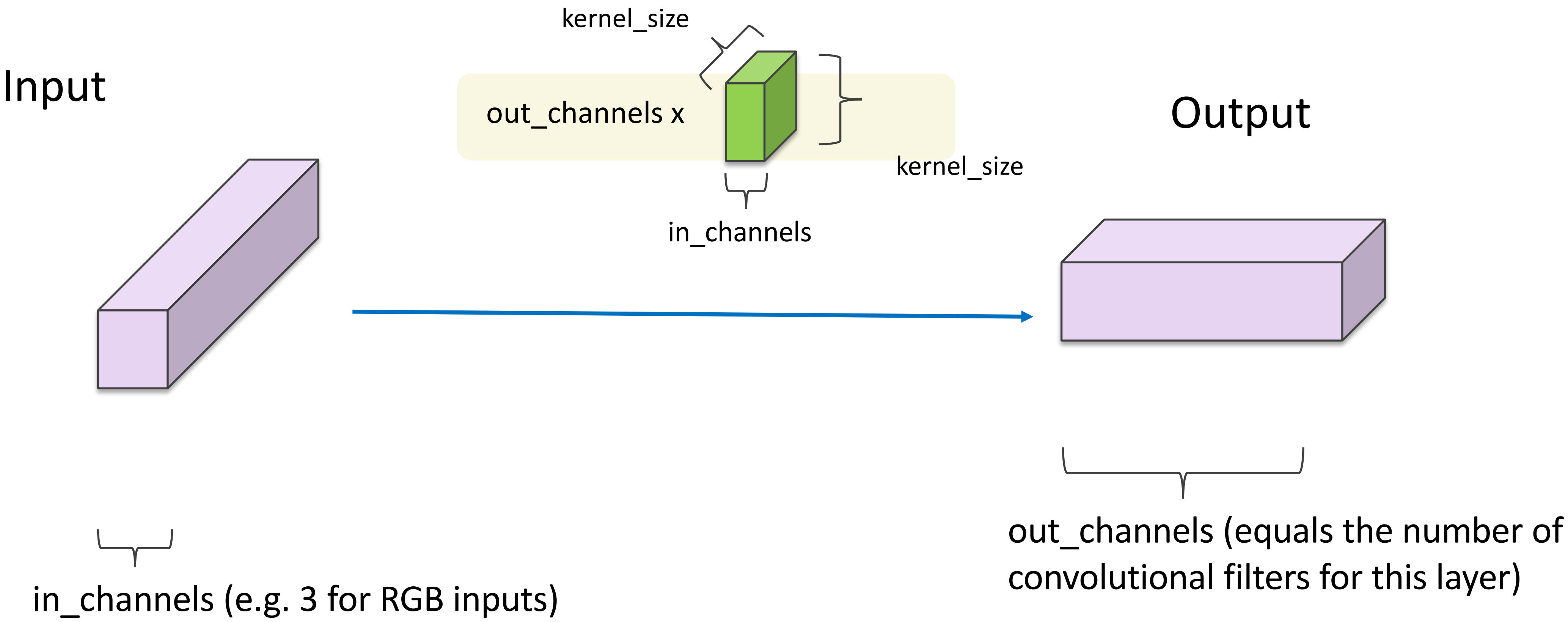
Transposed
Convolutional Layers

Spatial Full
Convolutional Layers

Convolutional Layer in pytorch



class torch.nn.Conv2d(*in_channels*, *out_channels*, *kernel_size*, *stride=1*, *padding=0*, *dilation=1*, *groups=1*, *bias=True*) [source]



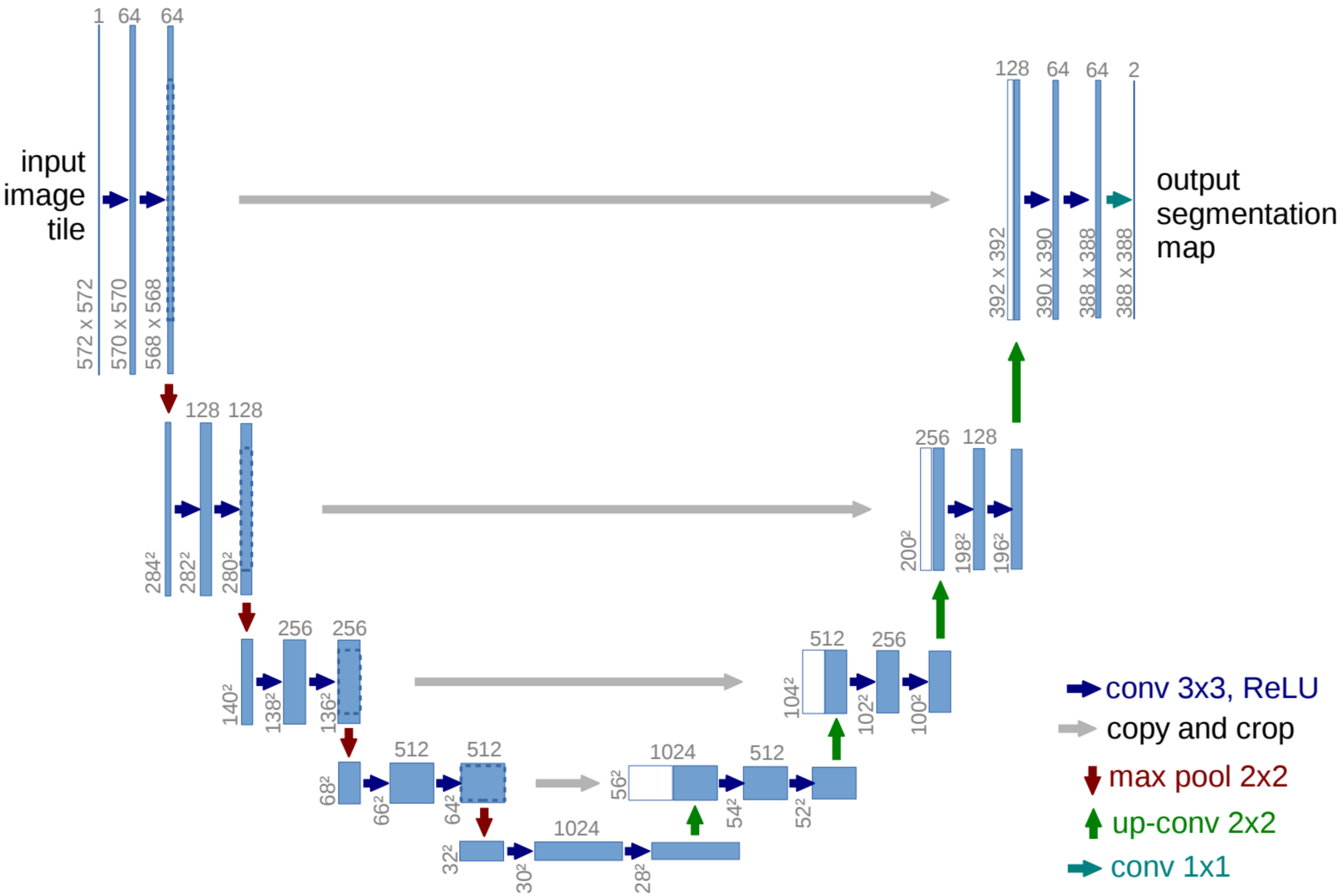
in_channels (e.g. 3 for RGB inputs)

out_channels (equals the number of convolutional filters for this layer)

U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

Computer Science Department and BIOSS Centre for Biological Signalling Studies,
University of Freiburg, Germany

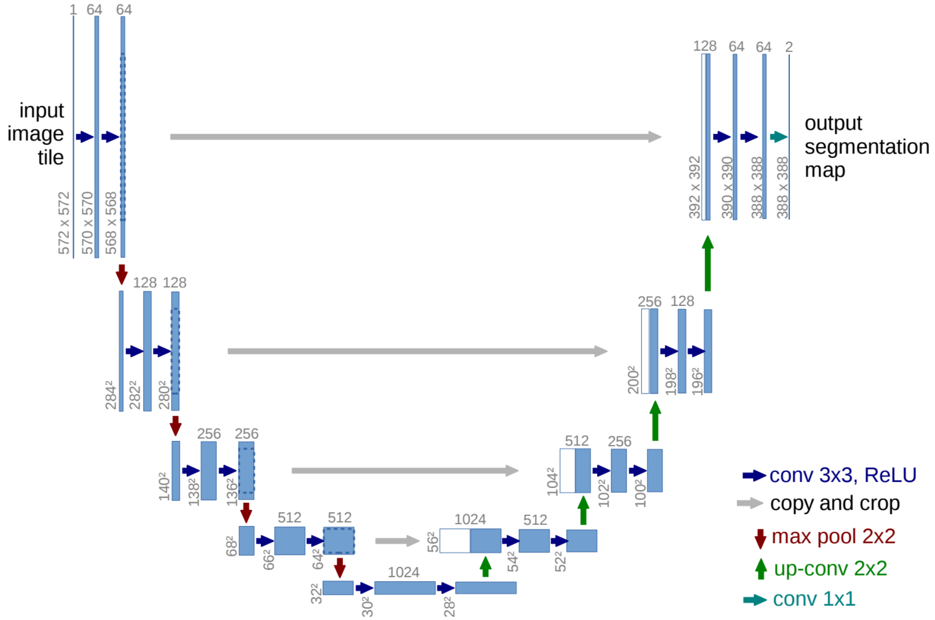


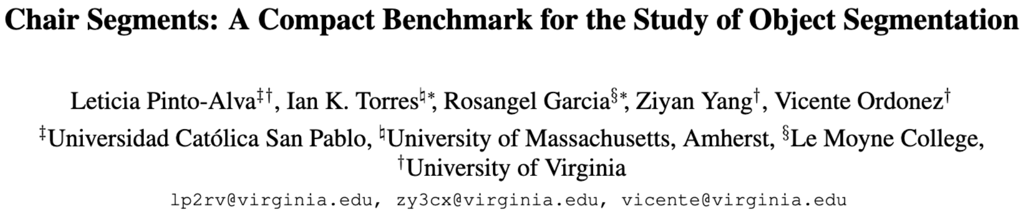
https://arxiv.org/abs/1505.04597

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

https://github.com/usuyama/pytorch-unet

Chair segmentation - Training

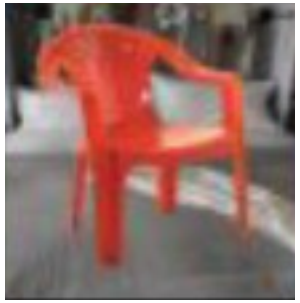


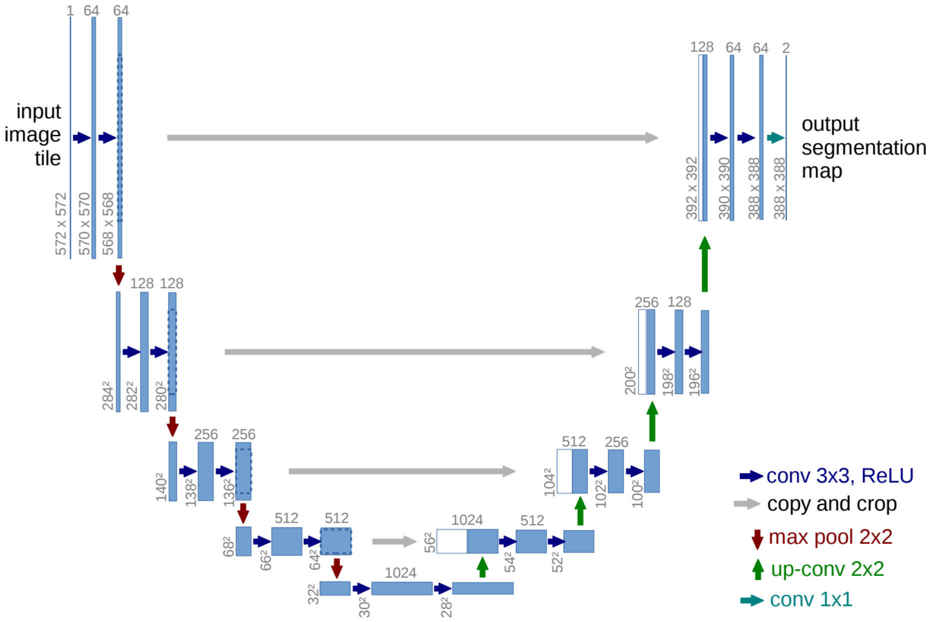


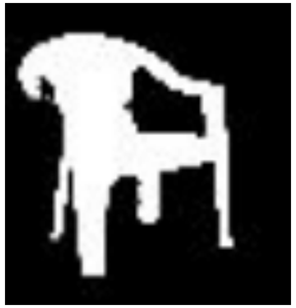
Chair Segments: A Compact Benchmark for the Study of Object Segmentation

Leticia Pinto-Alva[‡†], Ian K. Torres[♮*], Rosangel Garcia[§*], Ziyan Yang[†], Vicente Ordonez[†]
[‡]Universidad Católica San Pablo, [♮]University of Massachusetts, Amherst, [§]Le Moyne College,
[†]University of Virginia
lp2rv@virginia.edu, zy3cx@virginia.edu, vicente@virginia.edu

Chair segmentation - Prediction



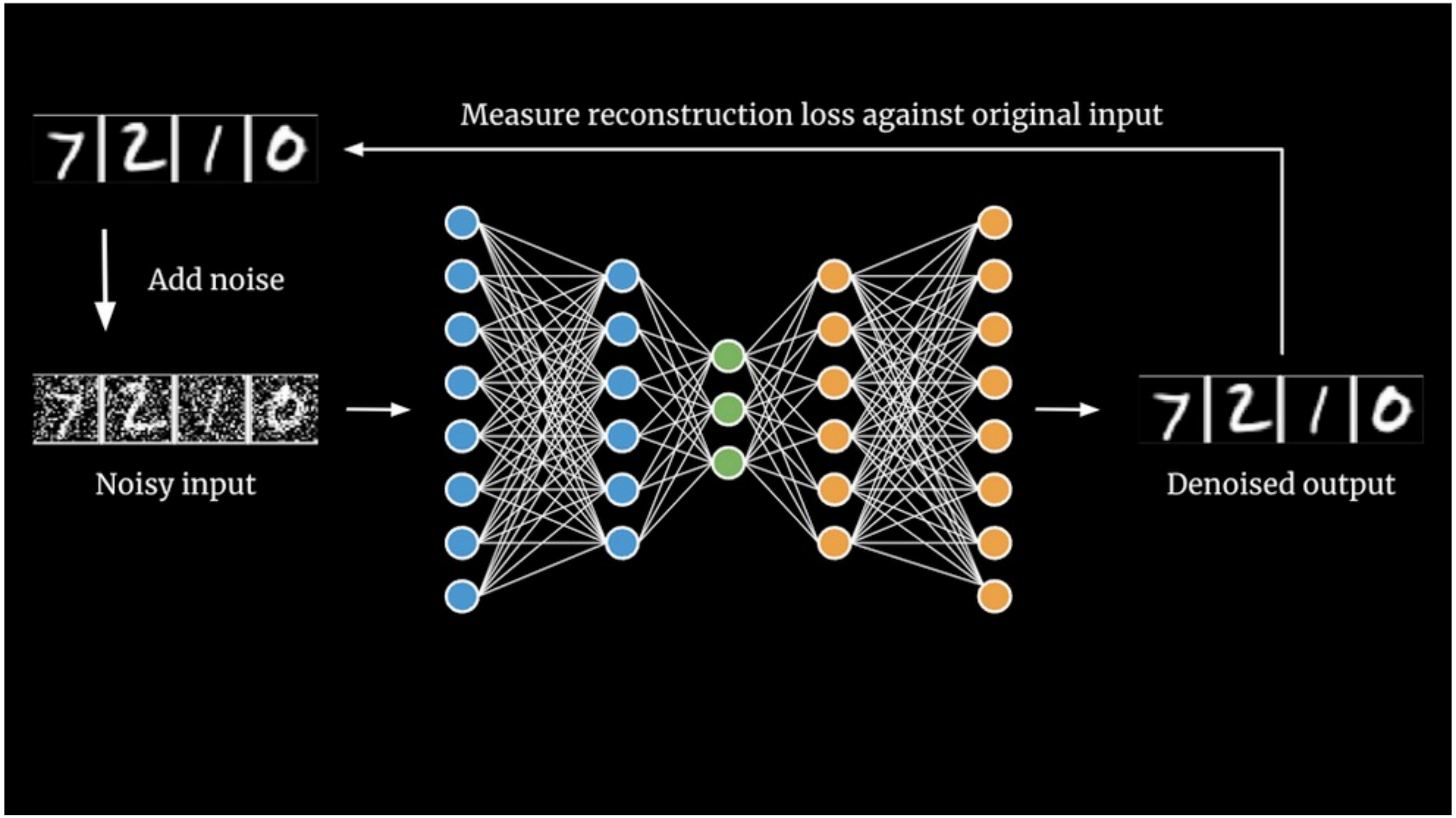




Chair Segments: A Compact Benchmark for the Study of Object Segmentation

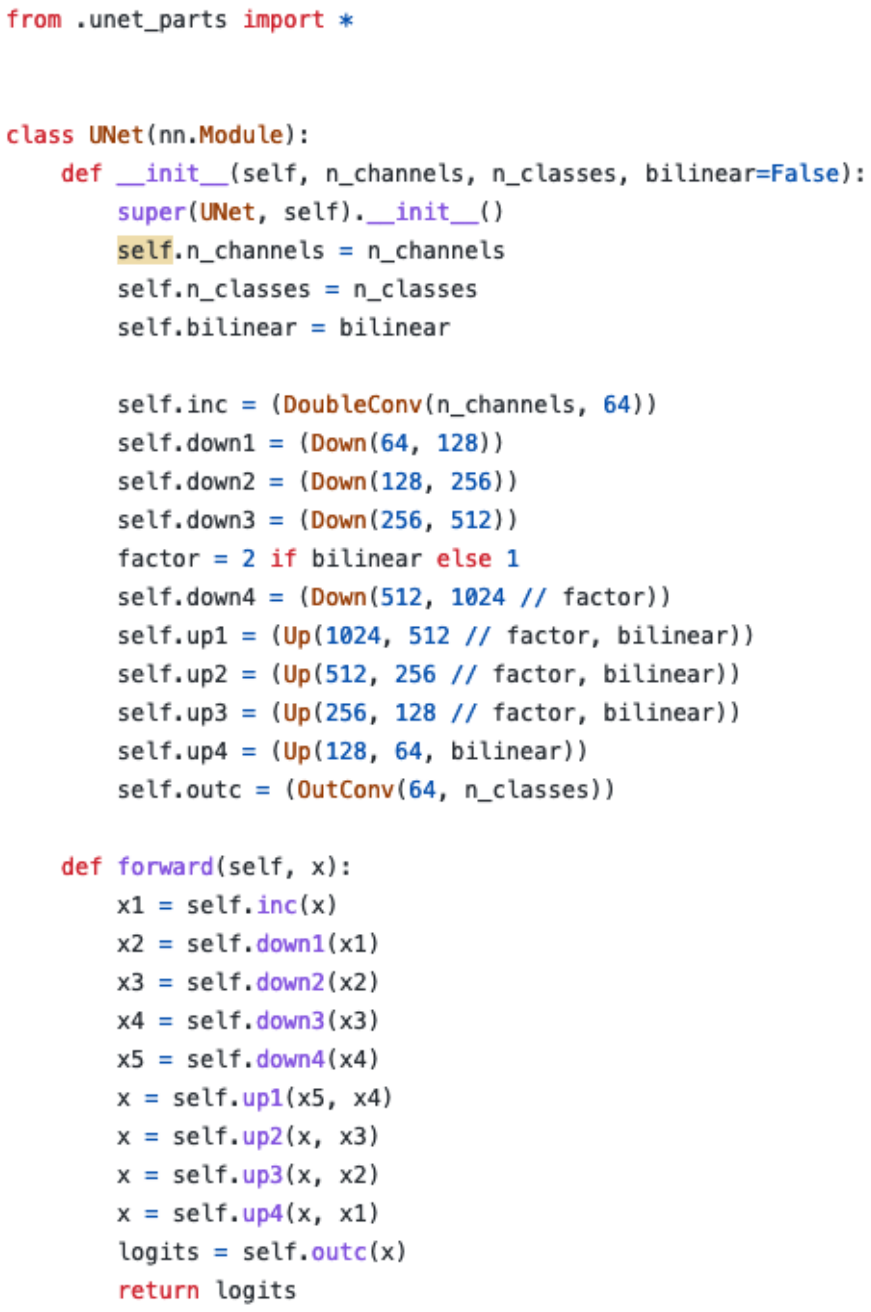
Leticia Pinto-Alva[‡†], Ian K. Torres[♮*], Rosangel Garcia[§*], Ziyan Yang[†], Vicente Ordonez[†]
[‡]Universidad Católica San Pablo, [♮]University of Massachusetts, Amherst, [§]Le Moyne College,
[†]University of Virginia
lp2rv@virginia.edu, zy3cx@virginia.edu, vicente@virginia.edu

AutoEncoders



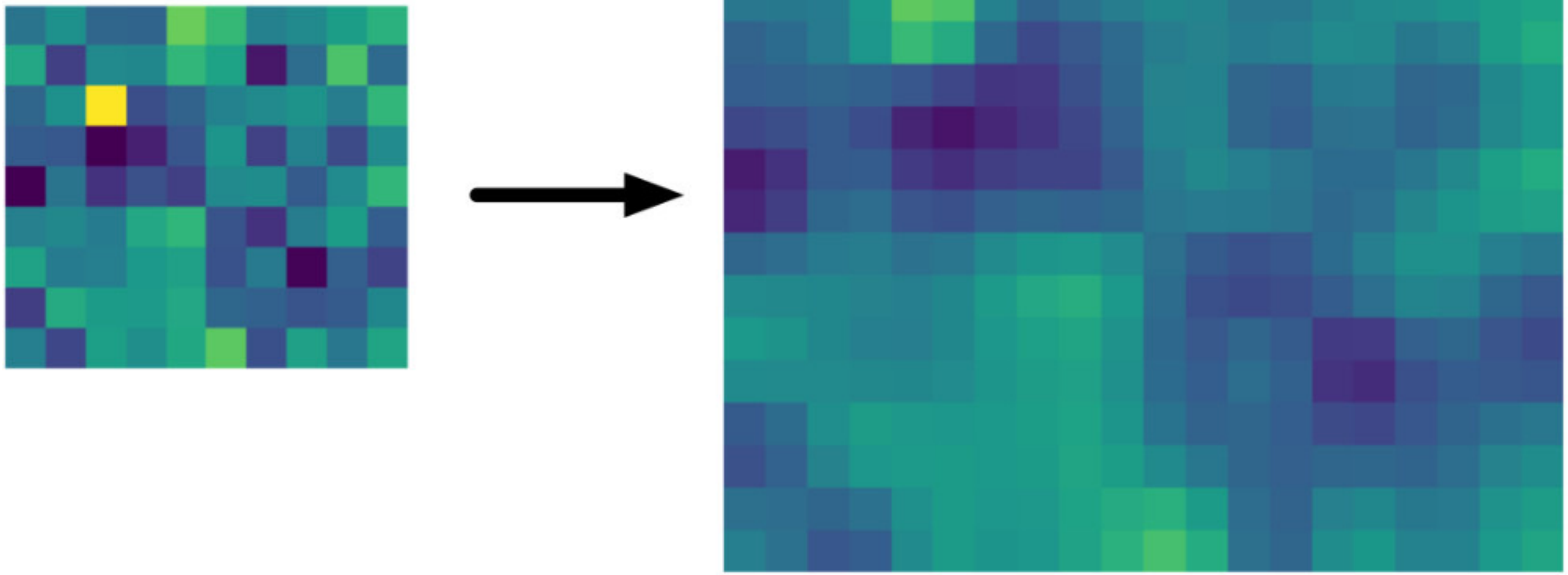
https://www.saberhq.com/blog/autoencoders

UNet in Pytorch



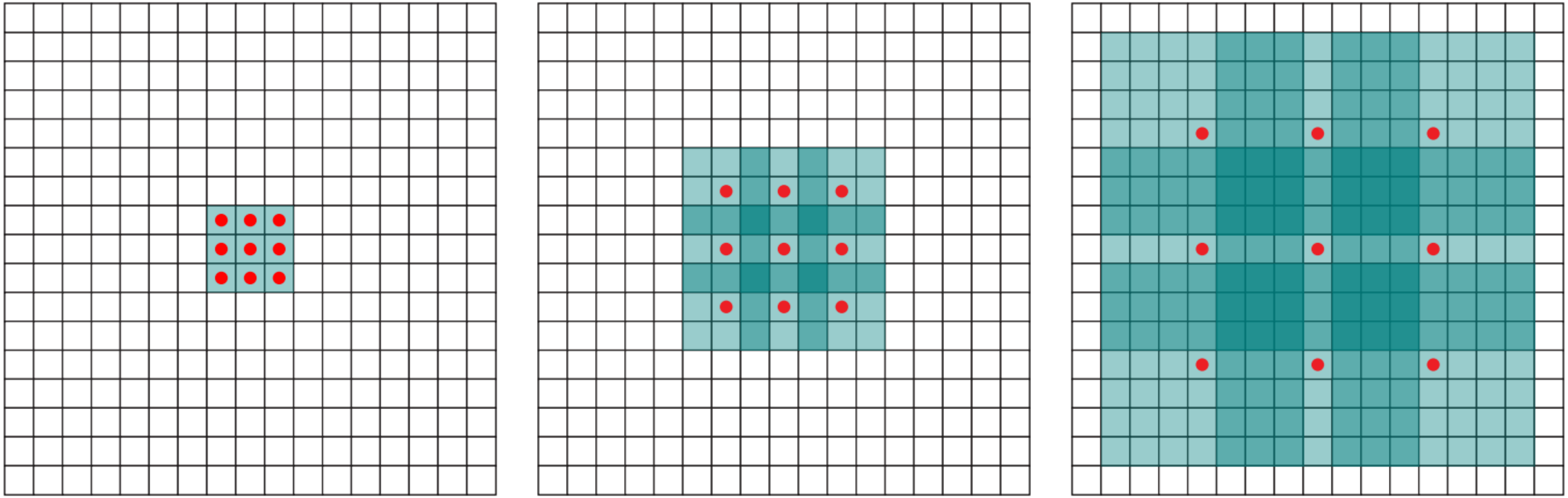
```
from .unet_parts import *


class UNet(nn.Module):
    def __init__(self, n_channels, n_classes, bilinear=False):
        super(UNet, self).__init__()
        self.n_channels = n_channels
        self.n_classes = n_classes
        self.bilinear = bilinear

        self.inc = (DoubleConv(n_channels, 64))
        self.down1 = (Down(64, 128))
        self.down2 = (Down(128, 256))
        self.down3 = (Down(256, 512))
        factor = 2 if bilinear else 1
        self.down4 = (Down(512, 1024 // factor))
        self.up1 = (Up(1024, 512 // factor, bilinear))
        self.up2 = (Up(512, 256 // factor, bilinear))
        self.up3 = (Up(256, 128 // factor, bilinear))
        self.up4 = (Up(128, 64, bilinear))
        self.outc = (OutConv(64, n_classes))

    def forward(self, x):
        x1 = self.inc(x)
        x2 = self.down1(x1)
        x3 = self.down2(x2)
        x4 = self.down3(x3)
        x5 = self.down4(x4)
        x = self.up1(x5, x4)
        x = self.up2(x, x3)
        x = self.up3(x, x2)
        x = self.up4(x, x1)
        logits = self.outc(x)
        return logits
```

Bilinear Upsampling Layer



https://machinethink.net/blog/coreml-upsampling/

Idea 3: Dilated Convolutions



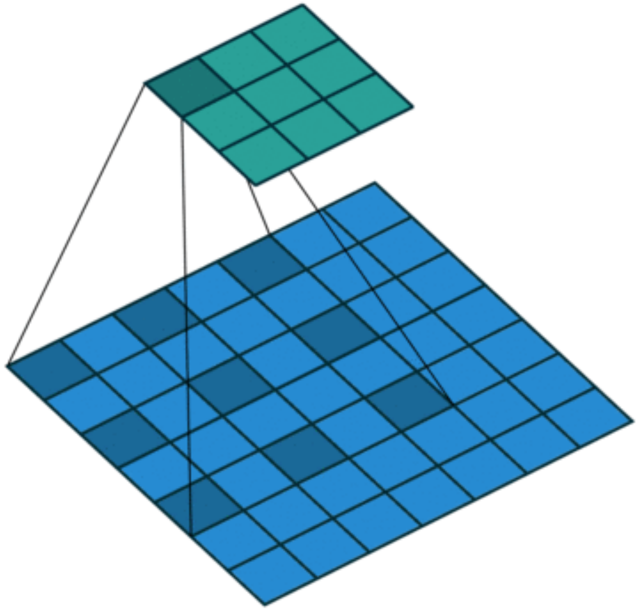
Multi-Scale Context Aggregation by Dilated Convolutions

Fisher Yu
Princeton University

Vladlen Koltun
Intel Labs

ICLR 2016

Idea 3: Dilated Convolutions



MULTI-SCALE CONTEXT AGGREGATION BY DILATED CONVOLUTIONS

Fisher Yu
Princeton University

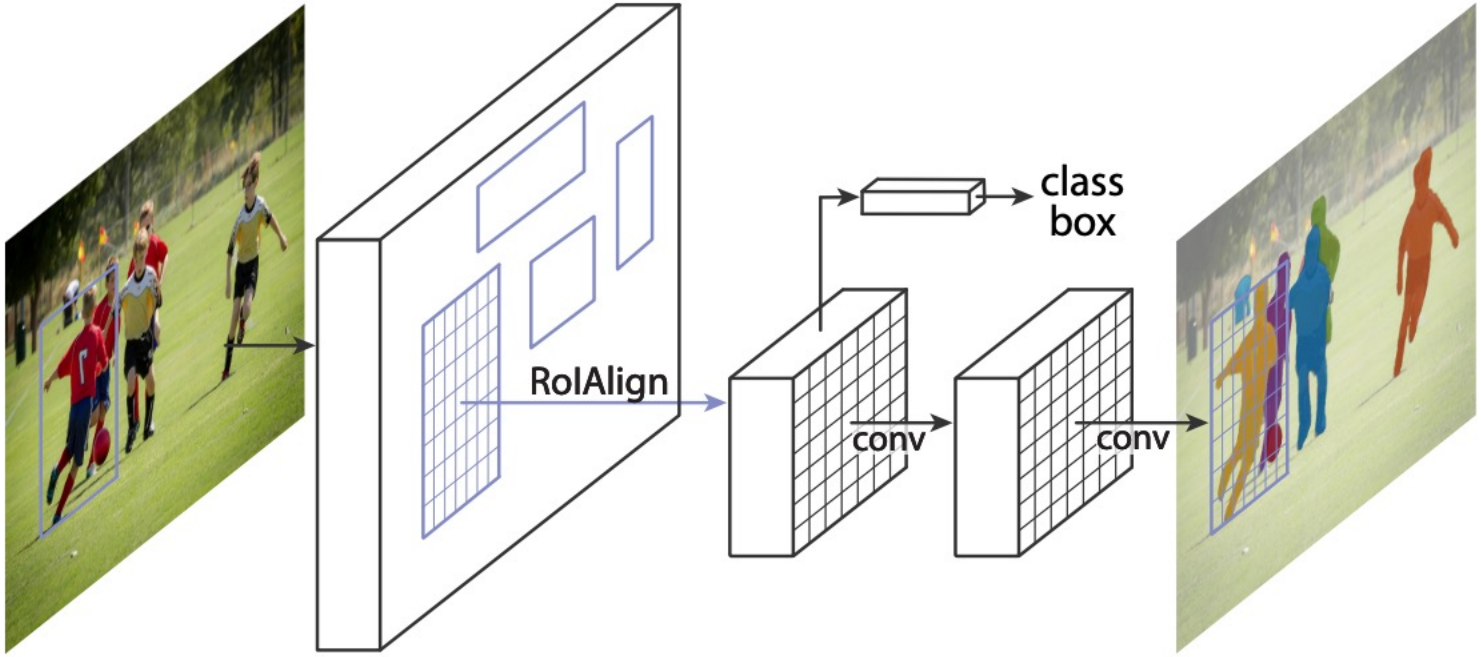
Vladlen Koltun
Intel Labs

ICLR 2016

Mask R-CNN

Kaiming He Georgia Gkioxari Piotr Dollár Ross Girshick

Facebook AI Research (FAIR)



https://github.com/facebookresearch/detectron2

https://arxiv.org/abs/1703.06870

Questions