Lecture: Deep Learning and Differential Programming

3.1 Computer Vision

https://liris.cnrs.fr/christian.wolf/teaching

Christian Wolf

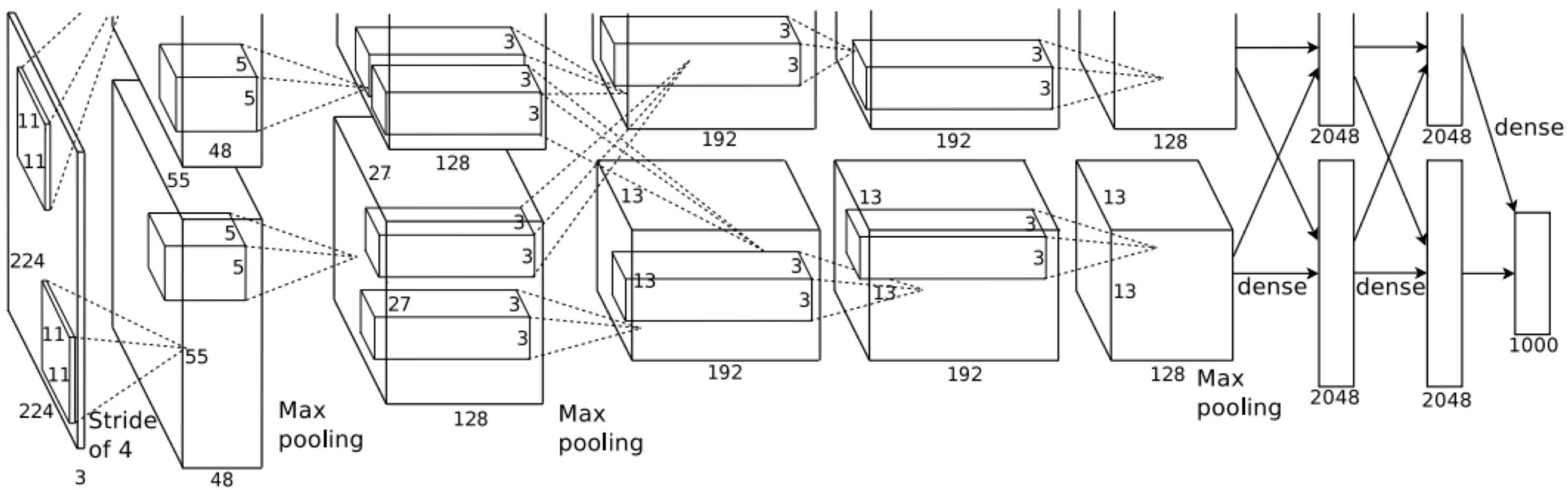
1. Image classification
2. Object detection and recognition
3. Semantic segmentation
4. Instance segmentation

AlexNet

The model which made deep learning hugely popular by winning the Imagenet competition in 2012.

8 trainable layers (5 convolutions, 3 fully connected).

Contribution: ReLU activation, dropout, multi-GPU training



[Krizhevsky, Sutskever, Hinton, 2012]

VGG 16

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

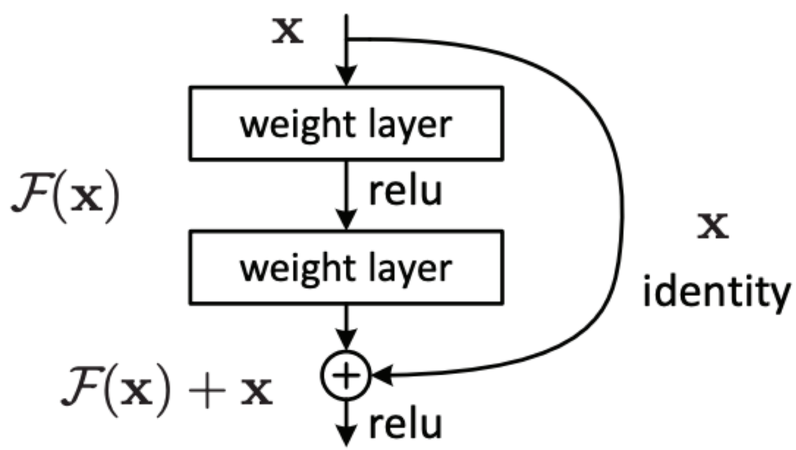
[Simonyan and Zissermann, ICLR 2015]

- 16 layers
- Only 3x3 convolutions

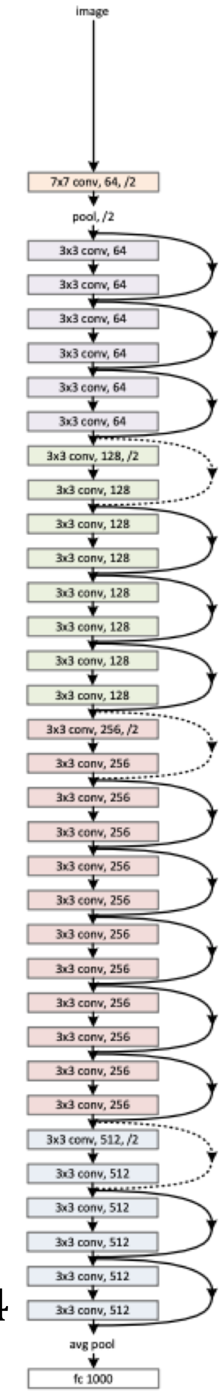
ResNet

Wins ImageNet 2015 competition.

Novelty: residual blocks – a block predicts the difference to its input.



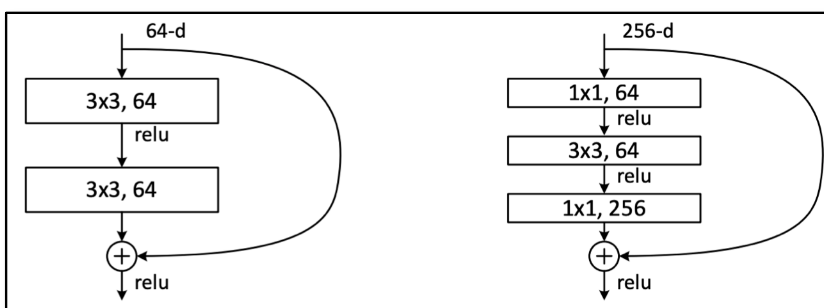
Resnet 34



[He, Zhang, Ren, Sun, CVPR 2016]

ResNet: variants

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9



Resnet 34 Resnet 50/101/152

[He, Zhang, Ren, Sun, CVPR 2016]

1 Image classification

2 Object detection and recognition

3 Semantic segmentation

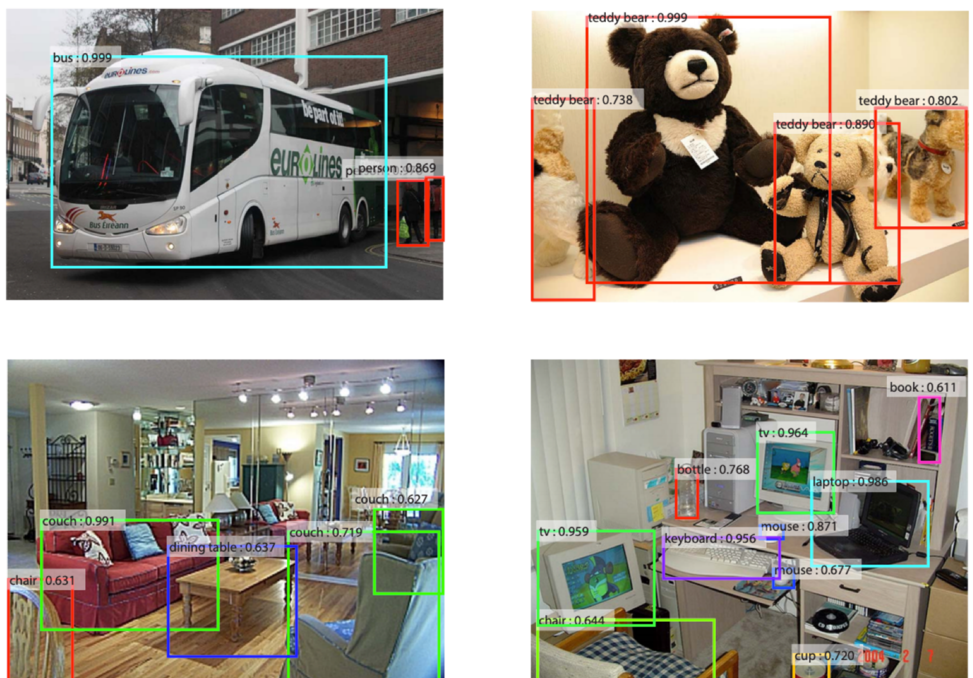
4 Instance segmentation

Detection, localization, recognition

A more complex problem. We need to:

- Detect whether an object exists
- Localize it (regress its bounding box coordinates)
- Recognize its class

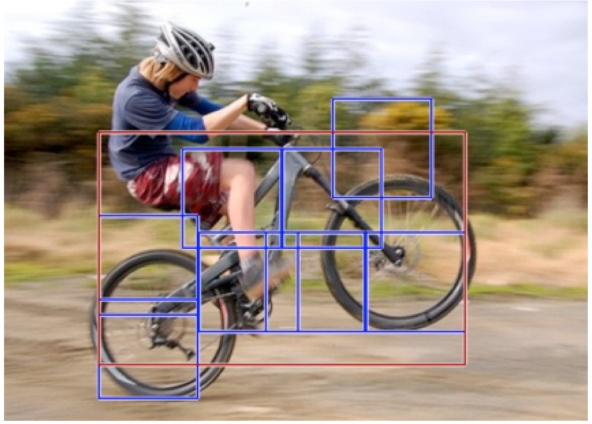
Multiple instances are possible



[Figure: Faster R-CNN, Ren, He, Girshick, Sun, NIPS 2015]

Before Deep Learning: Deformable Parts Models

- Model an object/human/activity as a collection of local parts
- Learn a filter for each part
- Learn an anchor position and deformation coefficeients for each part
- Test each image pixel whether it can be the center of the object (sliding window):
- For each possible center, optimize over (latent) local part positions

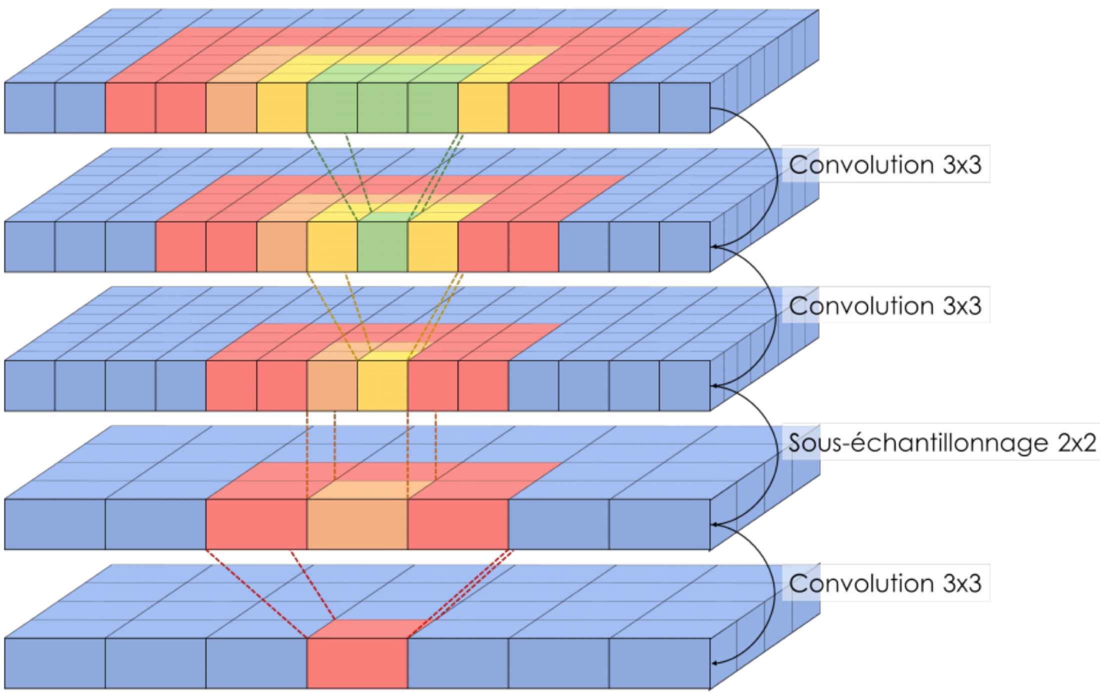


$$\underbrace{\sum_{i=0}^{n} F_i' \cdot \phi(H, p_i)}_{\text{Local appearance}} - \underbrace{\sum_{i=1}^{n} d_i \cdot \phi_d(dx_i, dy_i)}_{\text{Deformation}} + b,$$

[Felzenszwalb et al., PAMI 2010]

Spatial feature maps

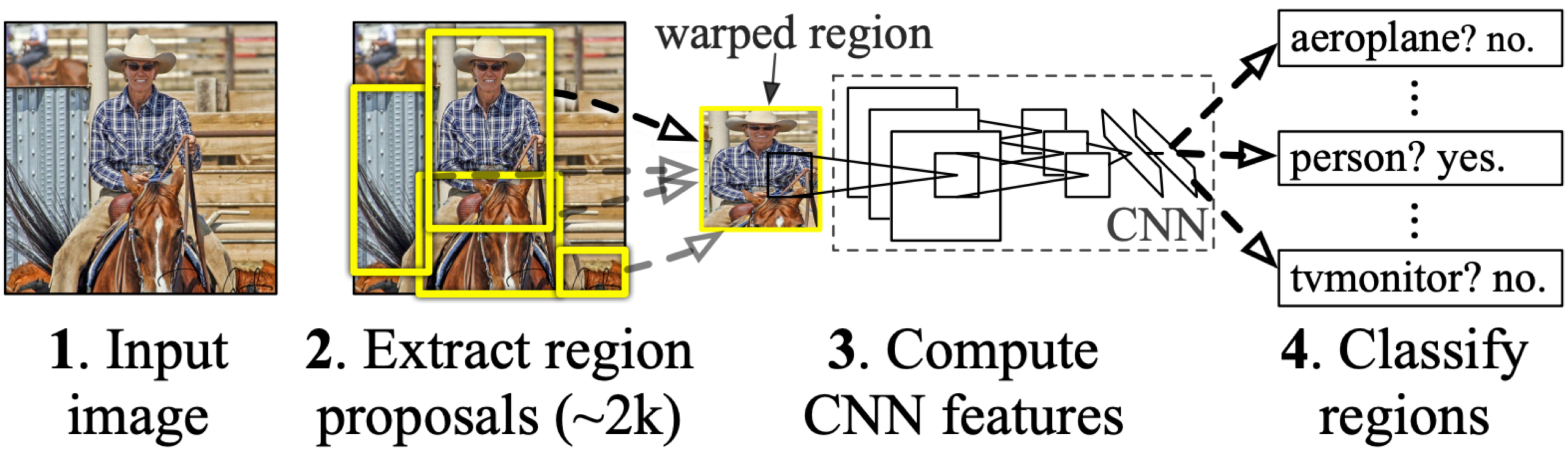
Most vision methods requiring localization exploit the fact that in convolutional neural networks (CNNs) intermediate layer activations have a spatial meaning: each cell corresponds to a rectangular area in the input image ("receptive field").



[Figure: Damien Fourure, PhD thesis, 2017]

R-CNN

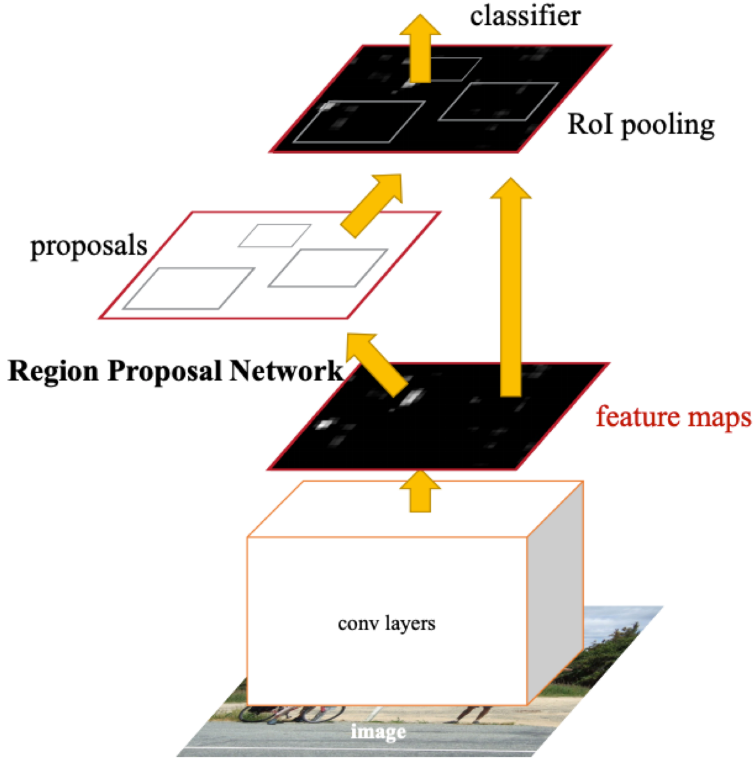
- Detect a large number of candidate regions (« region proposals ») with some heuristic method
- Feed each candidate region into a convolutional neural network for recognition



[Girshick, Donahue, Darrel, Malik, 2014]

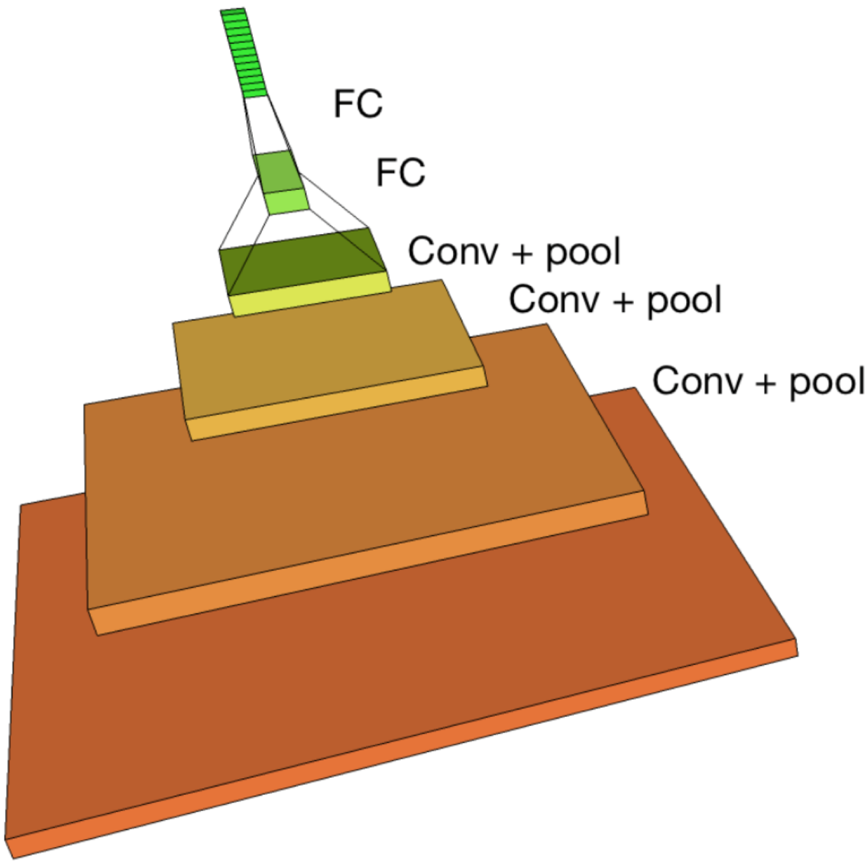
Faster R-CNN

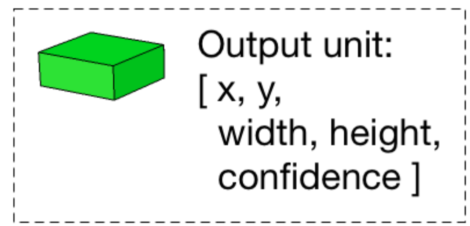
- The region proposals are now predicted by a neural network which is part of the full network
- For each proposal, features are collected from the bounding box of the proposal and fed to a classifier.



[Ren, He, Girshick, Sun, NIPS 2015]

Real time detectors



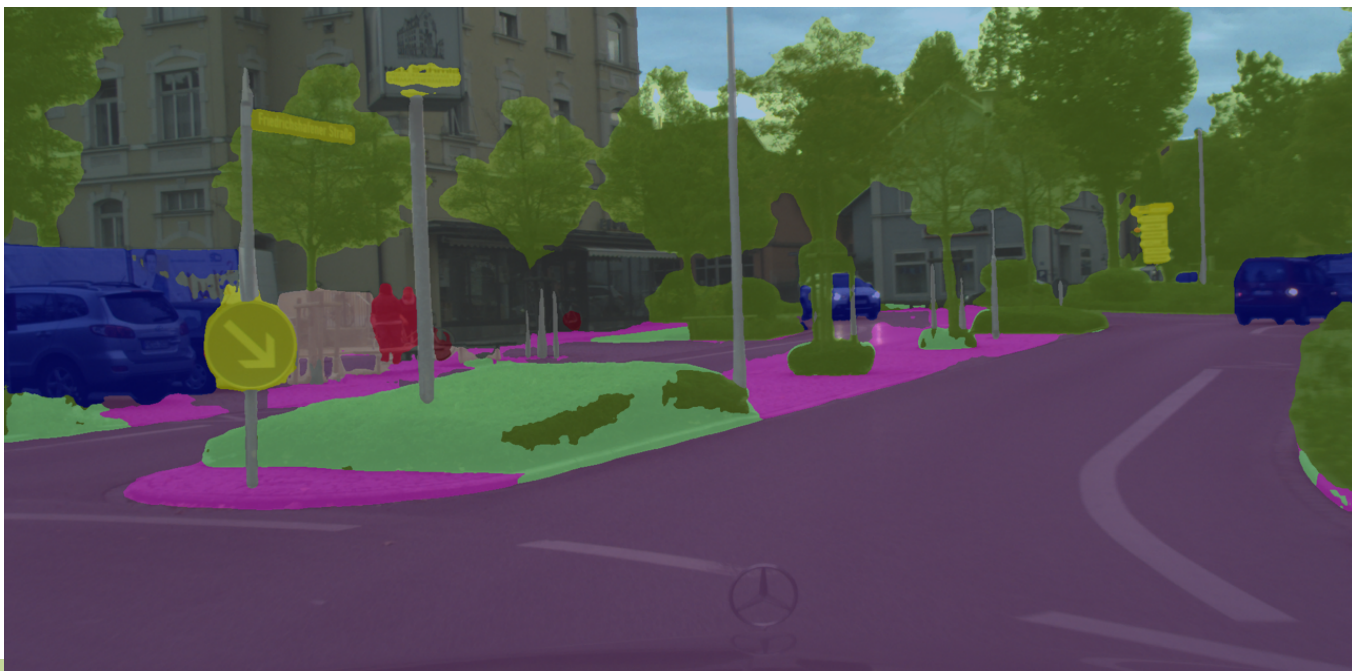


[Erhan et al.,2014] « Multibox »

[Redmon et al.,2016] « YOLO »

[Erhan et al.,2016] « SSD »

1. Image classification
2. Object detection and recognition
3. Semantic segmentation
4. Instance segmentation

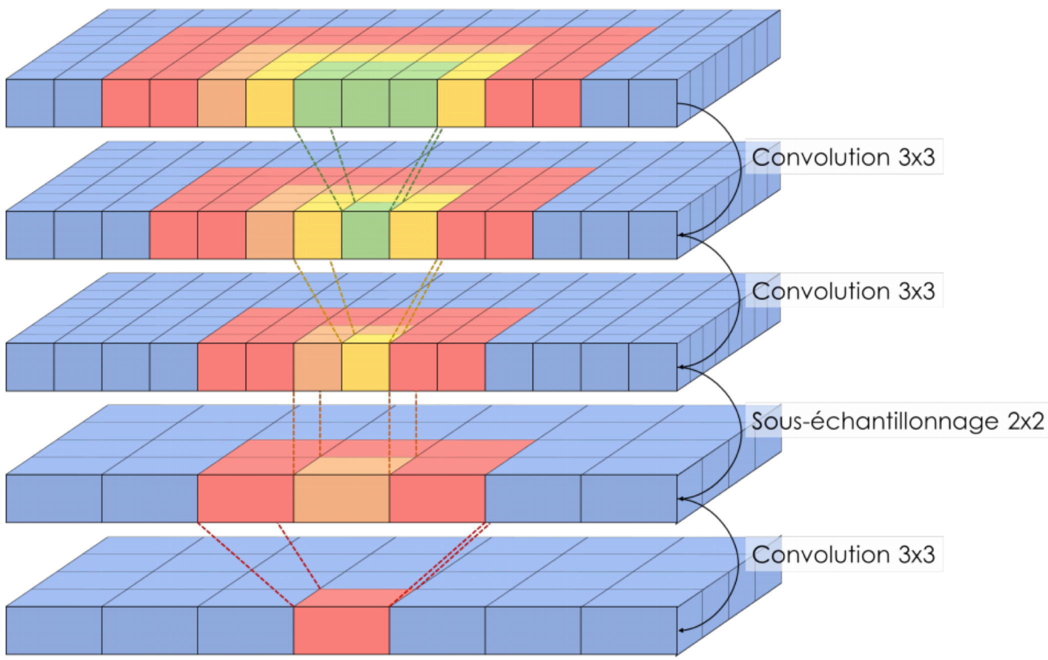


[Fourure et al., BMVC 2017]

Recall: receptive Fields

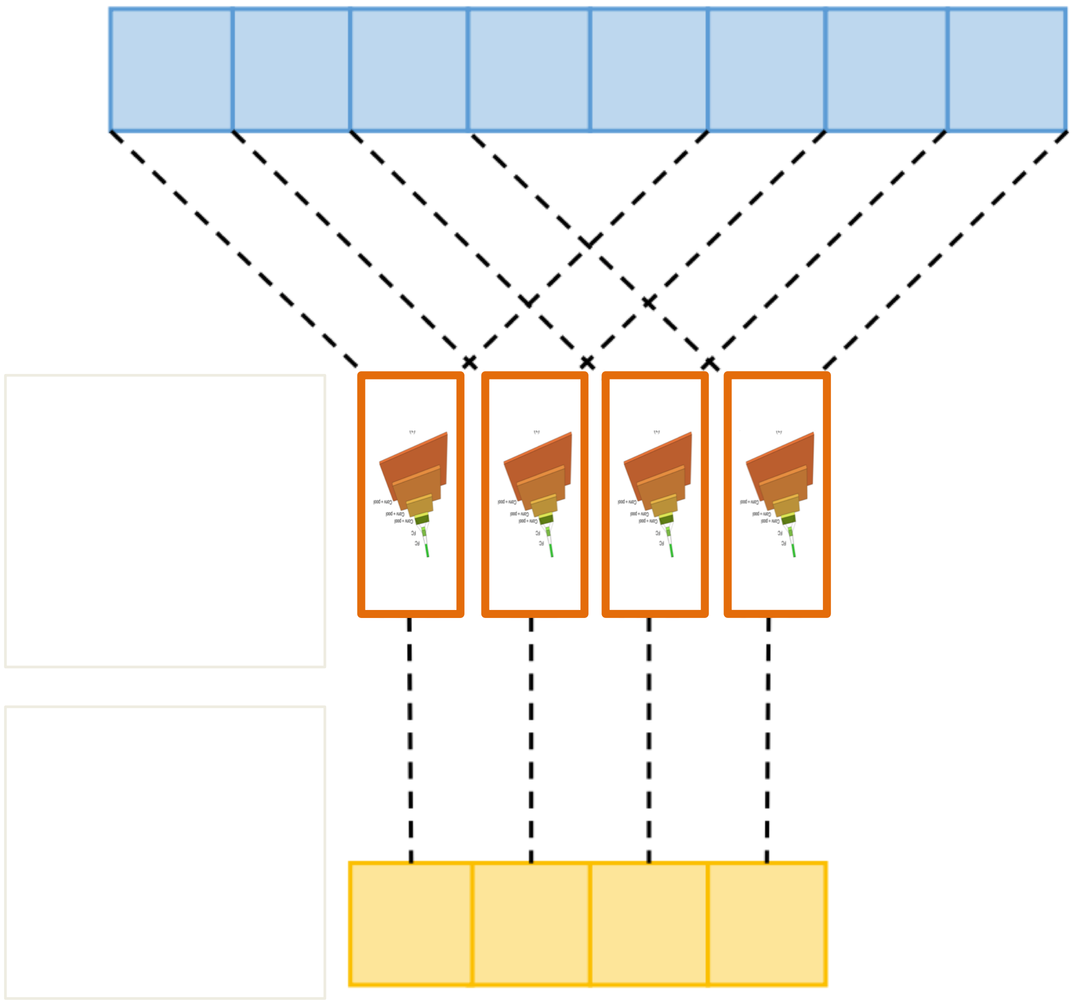
Pooling layers reduce the spatial resolution of intermediate activations and output layers.

How can we create dense predictions while keeping the output resolution equal to the input resolution?



[Figure: Damien Fourure, PhD thesis, 2017]

Patchwise processing



A direct, simple and early method.

Not used anymore

Feed each pixel + neighborhood as input into a network.

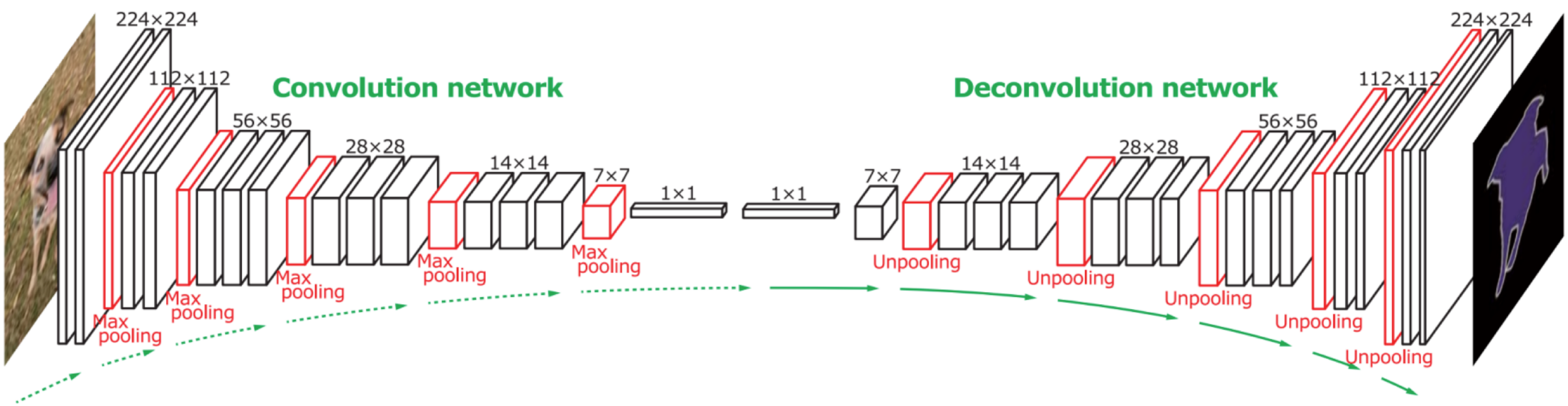
Slow!

[Figure: Damien Fourure, PhD thesis, 2017]

Conv-Deconv Networks

A classical ”encoder network” produces a vectorial representation through convolutions and pooling.
A second network (“decoder”) decodes this into an output image with the initial resolution.

All information must pass through the bottleneck layer!

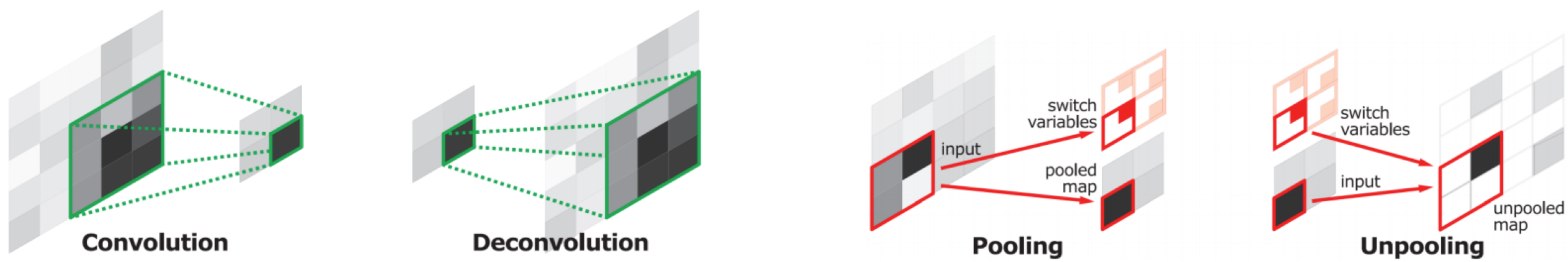


[Noh et al., 2015]

Conv-Deconv Networks

In the decoder:

- convolutions are replaced with "deconvolutions" or "transposed convolutions".
- Pooling is replaced with unpooling (switch variables keep the arg max location of the pooling layer)

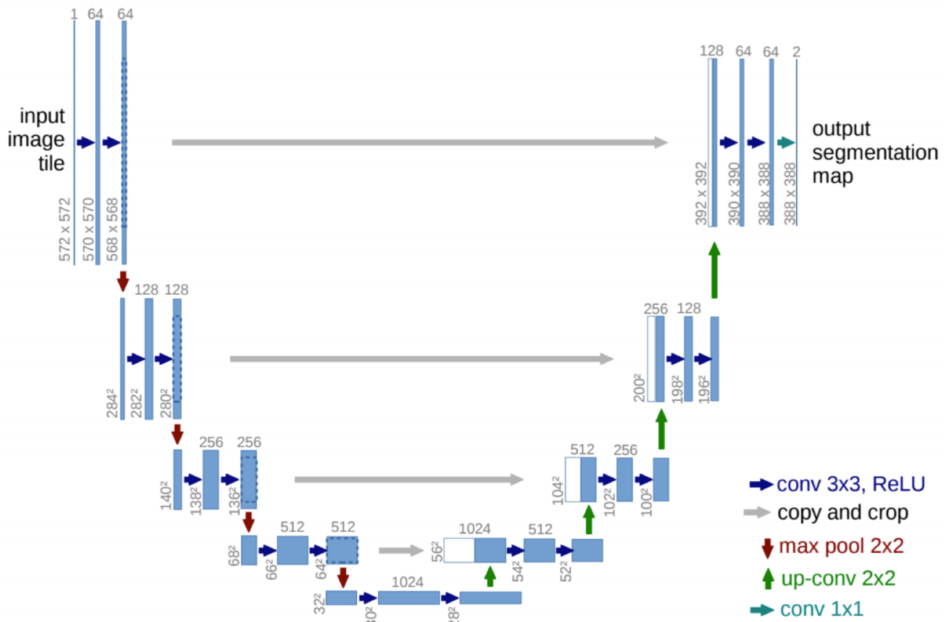


[Noh et al., 2015]

U-Nets

Conv-Deconv: all information passes through the bottleneck layer.

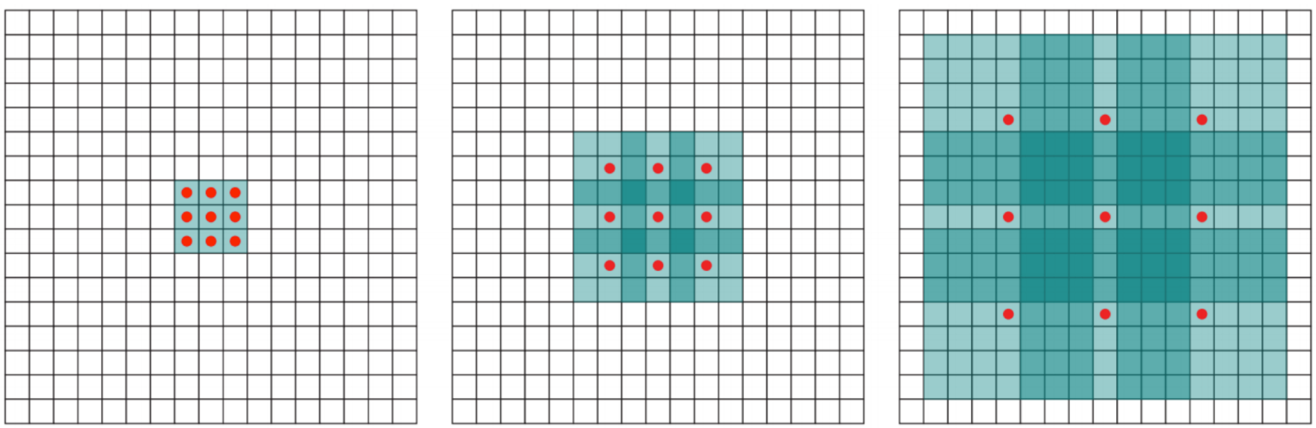
U-Nets add additional skip connections for low level information, close to pixels.



[Ronneberger et al., MICCAI 2015]

Dilated Convolutions

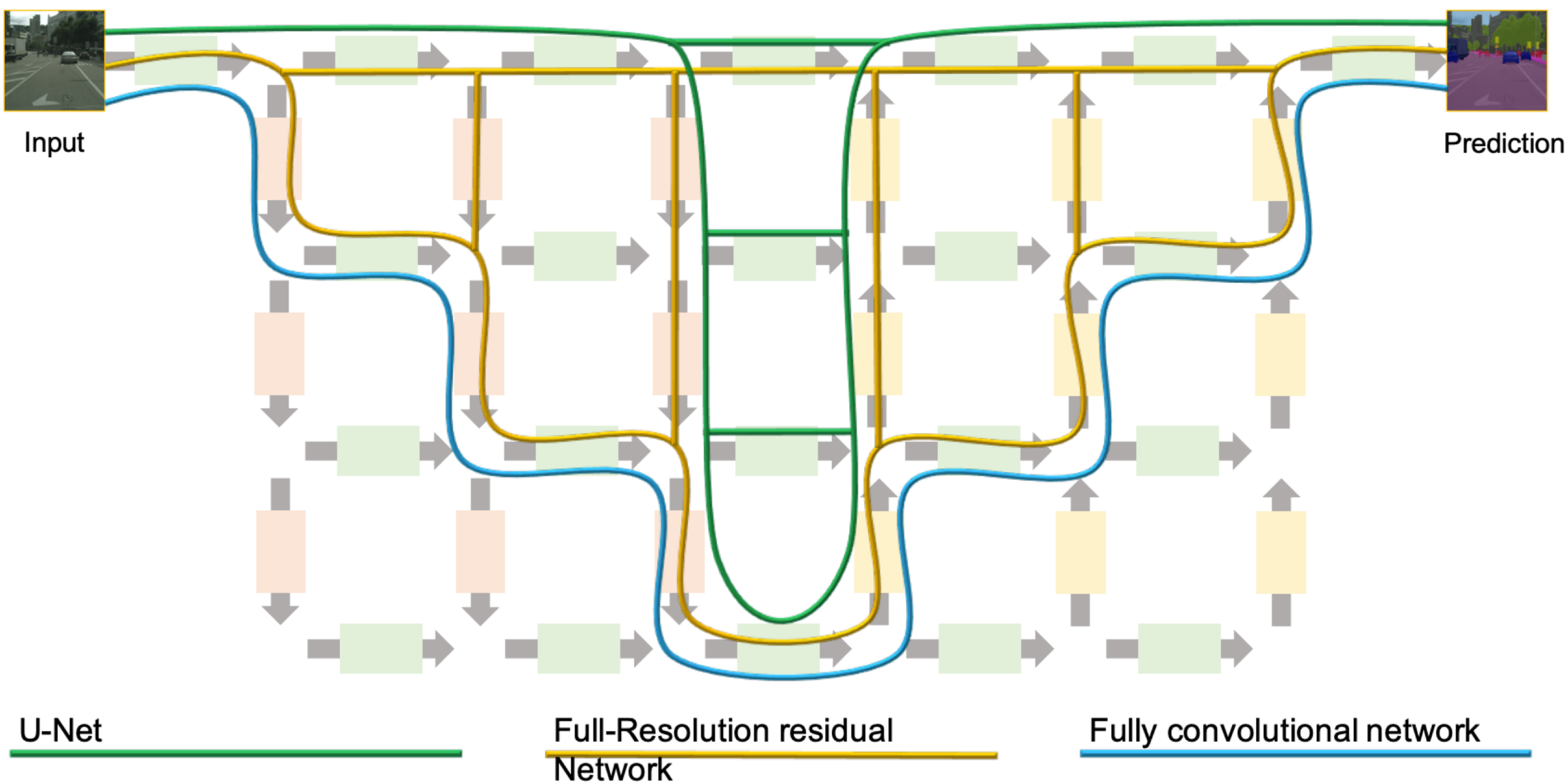
Alternative: do not use any pooling, keep spatial resolution throughout the network.
To increase the receptive field, change the size of the filters … w/o augmenting the number of parameters!



[Yu and Koltun, ICLR 2016]

Grid Networks

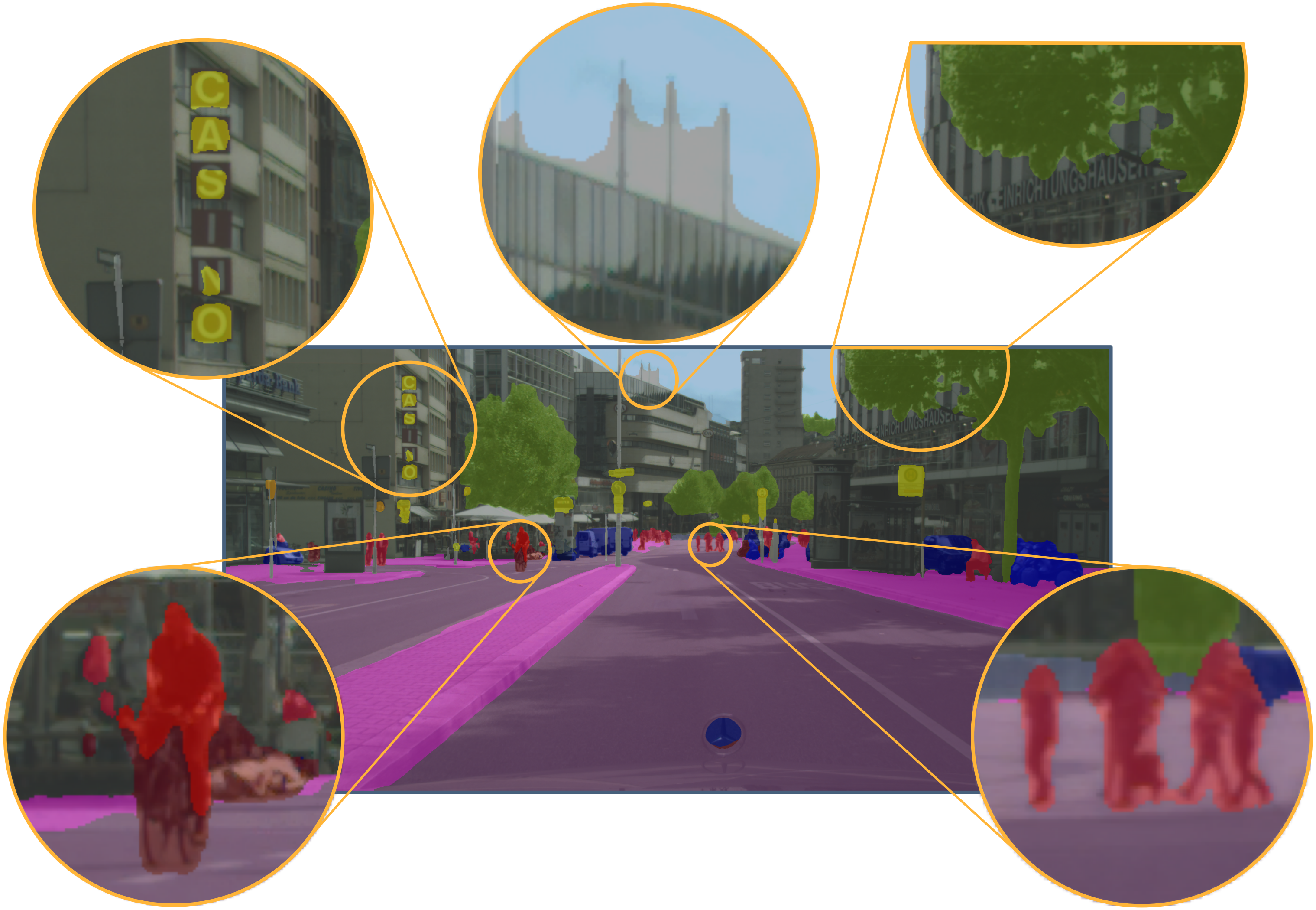
Grid Networks generalize a large number of networks.



[Fourure, Emonet, Fromont, Muselet, Tremeau, Wolf, BMVC 2017]

Grid Networks

[Fourure, Emonet, Fromont, Muselet, Tremeau, Wolf, BMVC 2017]



[Fourure, Emonet, Fromont, Muselet, Tremeau, Wolf, BMVC 2017]

1 Image classification

2 Object detection and recognition

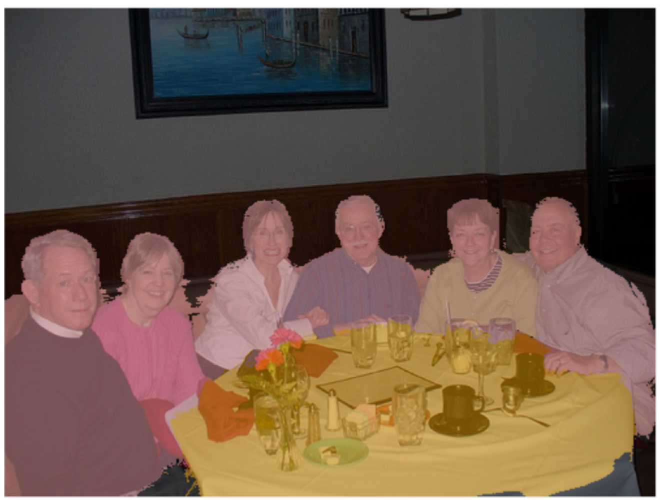
3 Semantic segmentation

4 Instance segmentation

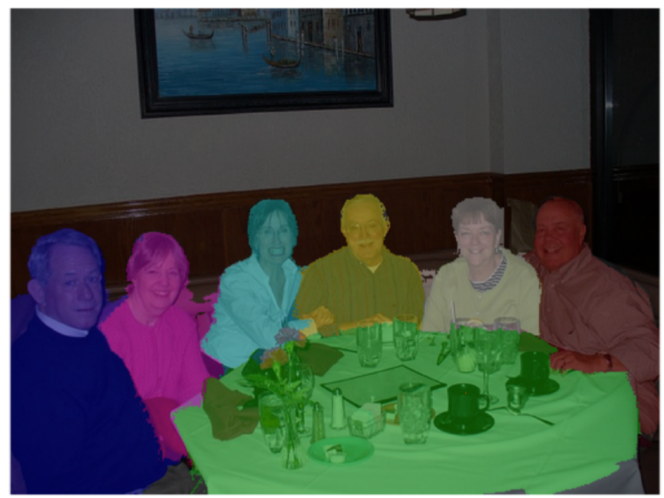
Instance segmentation

Bridges the gap between semantic segmentation and object detection:

- One region = 1 object instance
- Pixelwise boundaries instead of bounding boxes



Semantic Segmentation



Instance Segmentation

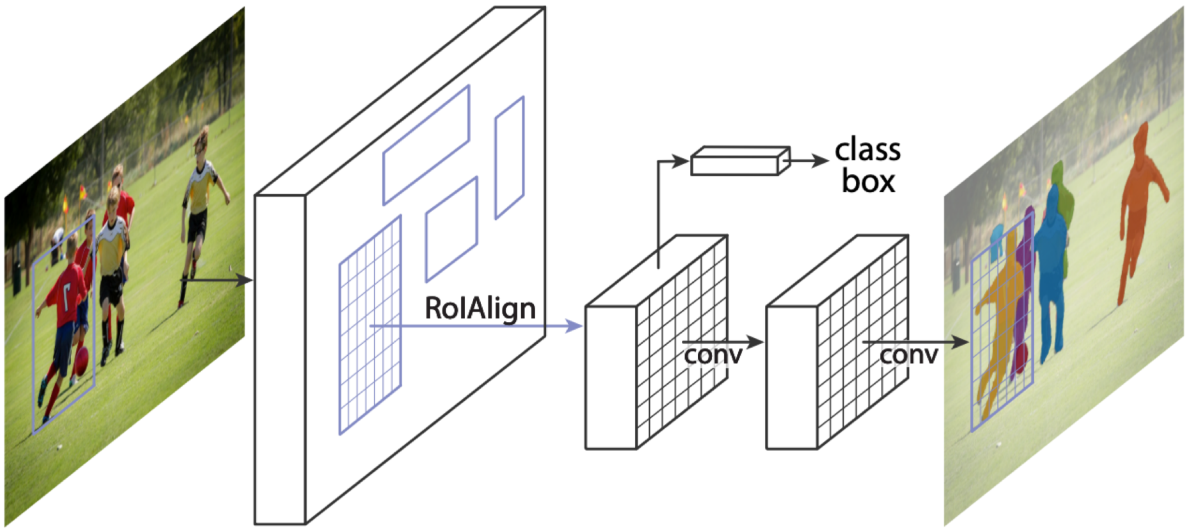
[Figure: Arnab et al., IEEE Signal Processing, 2018]

Mask R-CNN

Region proposals as in object detection

Conv-Deconv network like in semantic segmentation ... but for each region proposal.

ResNet-101 Backbone.



[Hen, Gkioxkari, Dollar, Girshick, ICCV 2017]

Mask R-CNN

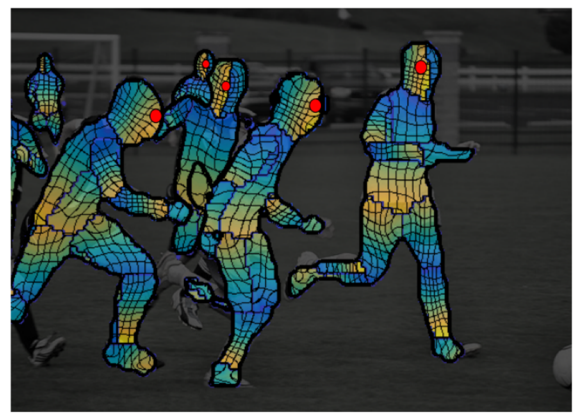
[Hen, Gkioxkari, Dollar, Girshick, ICCV 2017]

Dense Pose

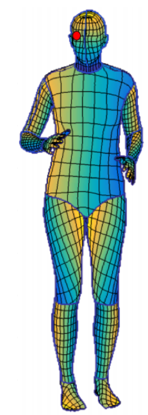
Densepose estimates human articulated pose in a dense manner:

One estimate per image pixel, corresponding to two parameters (u,v) on the human body.

input



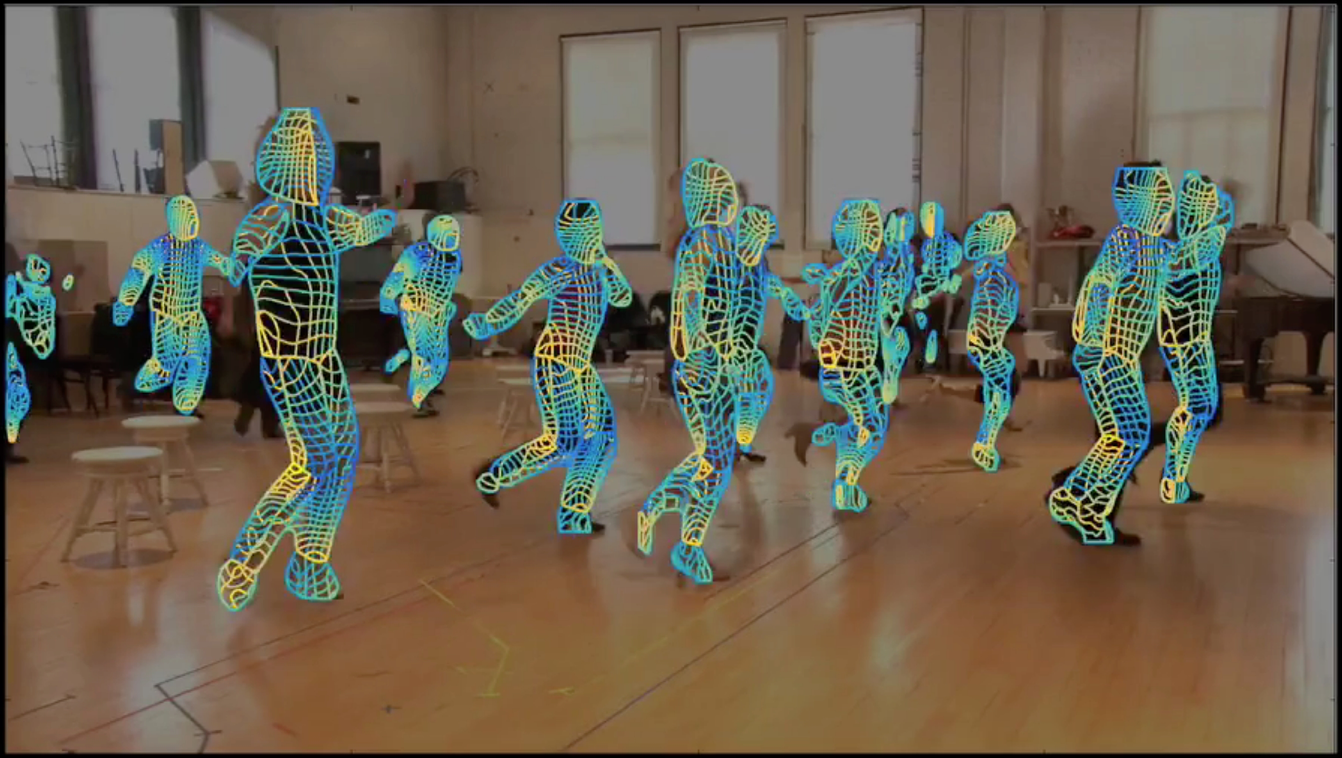
output



[Güler, Neverova, Kokkinos, CVPR 2018]

Slides: Natalia Neverova

PoseTrack dataset:
Visual results

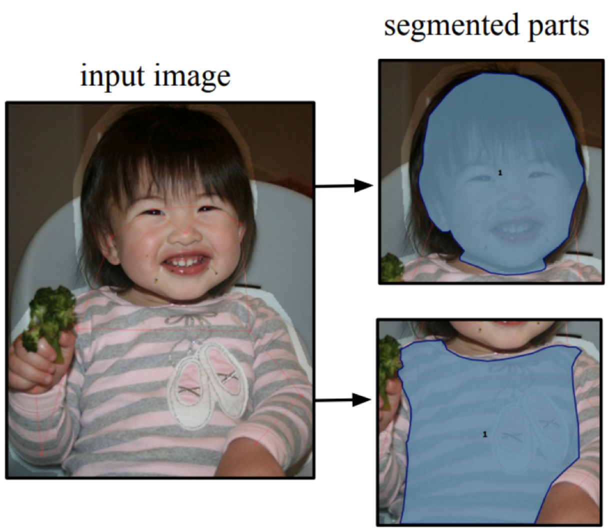


[Güler, Neverova, Kokkinos, CVPR 2018]

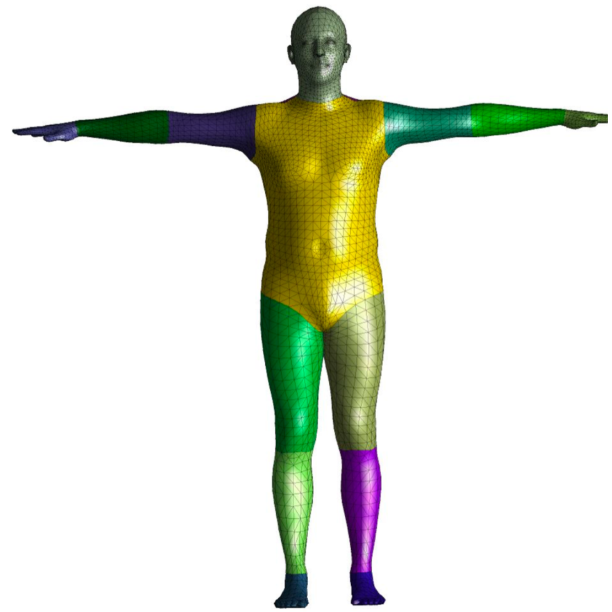
Slides: Natalia Neverova

The Densepose dataset

Annotation task 1: body part segmentation



TASK 1: Part Segmentation

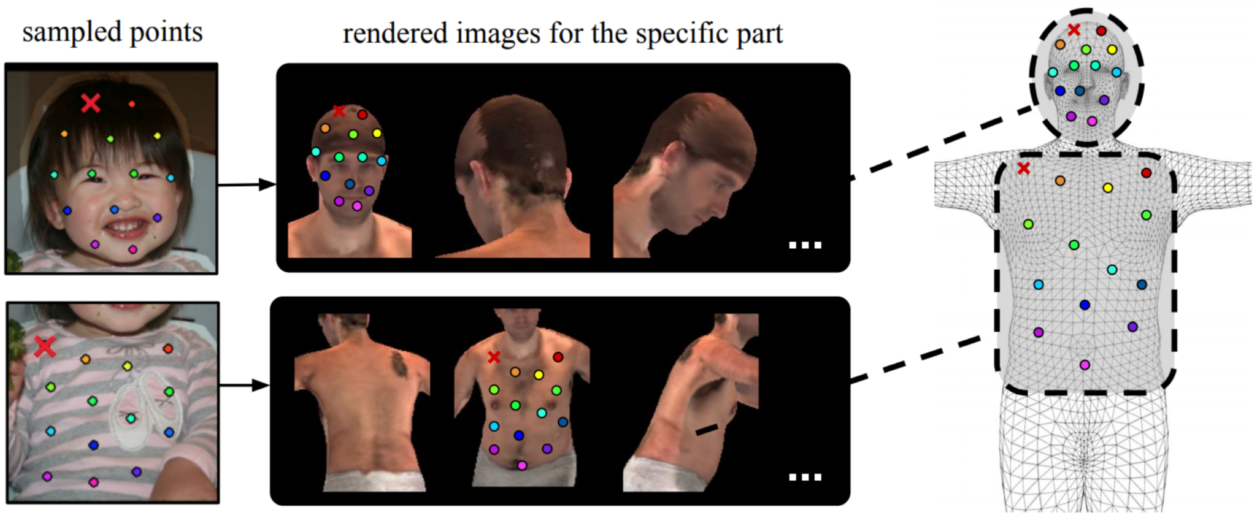


[Güler, Neverova, Kokkinos, CVPR 2018]

Slides: Natalia Neverova

The Densepose dataset

Annotation task 2: marking sparse correspondences

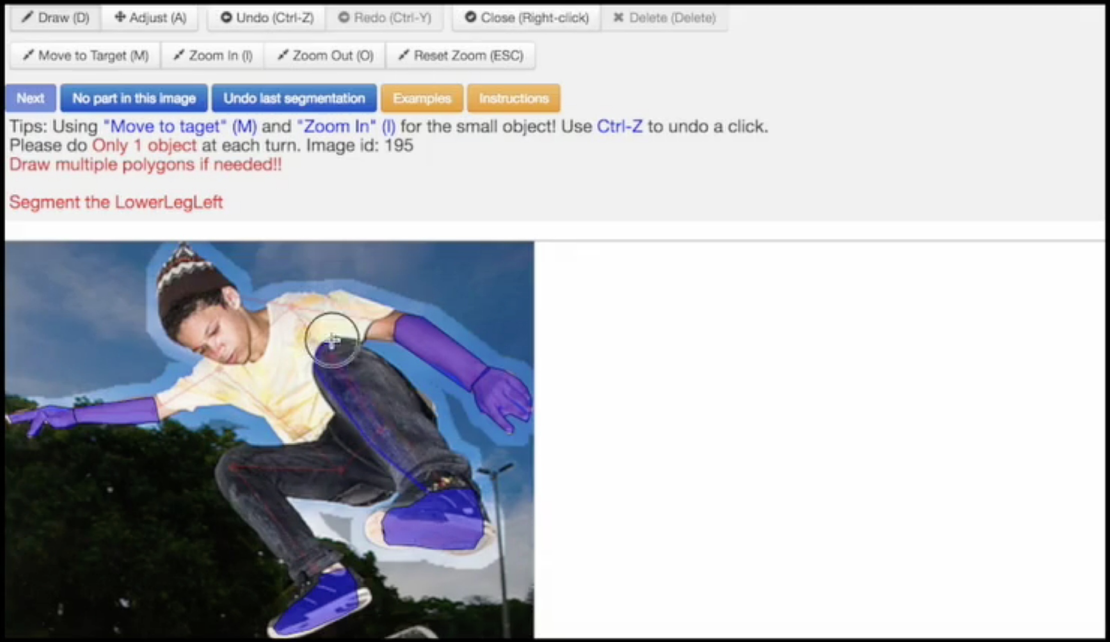


TASK 2: Marking Correspondences

[Güler, Neverova, Kokkinos, CVPR 2018]

Slides: Natalia Neverova

COCO DensePose:
Collecting Data



Task - 1

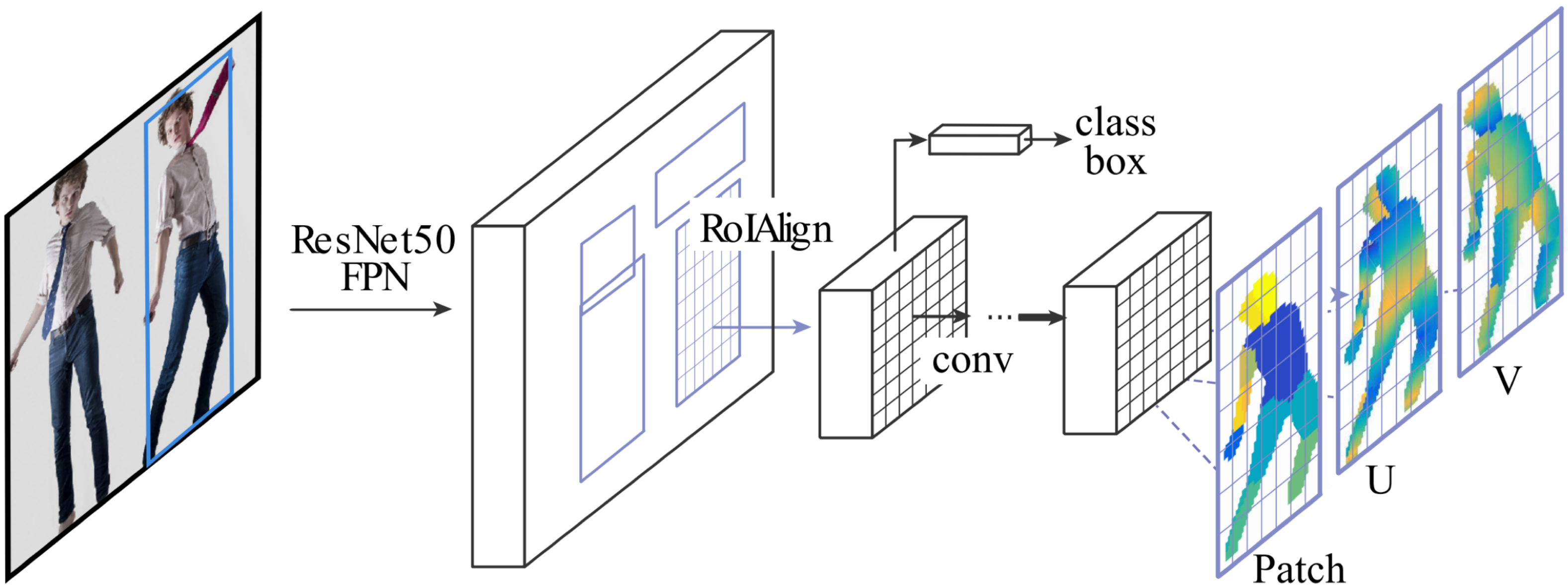
Part Segmentation

[Güler, Neverova, Kokkinos, CVPR 2018]

Slides: Natalia Neverova

The neural architecture

Densepose builds on the Marks R-CNN architecture.



[Güler, Neverova, Kokkinos, CVPR 2018]

Model Zoos

Different libraries propose « model zoos » containing well-known network architectures, sometimes with pre-trained parameters learned from standard datasets.

For PyTorch: torchvison.models:

- AlexNet
- VGG
- ResNet
- SqueezeNet
- DenseNet
- Inception v3
- GoogLeNet
- ShuffleNet v2
- MobileNet v2
- ResNeXt
- Wide ResNet
- MNASNet

The torchvision model zoo

Construct models with random weights:

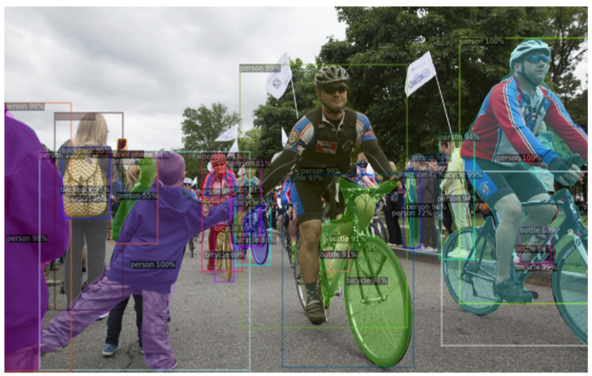
```
import torchvision.models as models
resnet18 = models.resnet18()
alexnet = models.alexnet()
vgg16 = models.vgg16()
```

Construct models with pre-trained weights (on ImageNet):

```
import torchvision.models as models
resnet18 = models.resnet18(pretrained=True)
alexnet = models.alexnet(pretrained=True)
vgg16 = models.vgg16(pretrained=True)
```

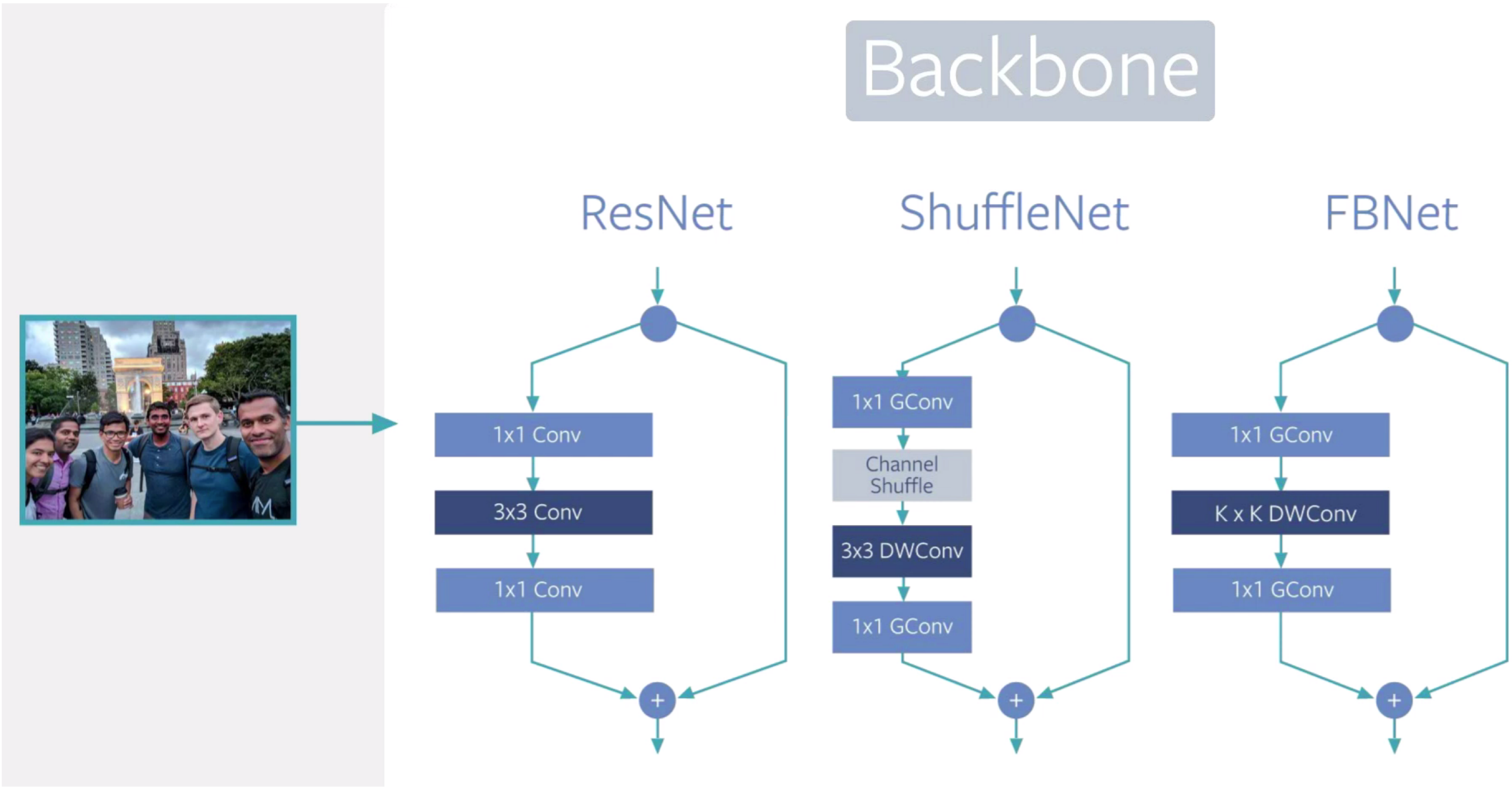
Facebook Detectron2

A single method capable of creating different predictions of different granularity using the same choice of “backbone” network (=network calculating features).



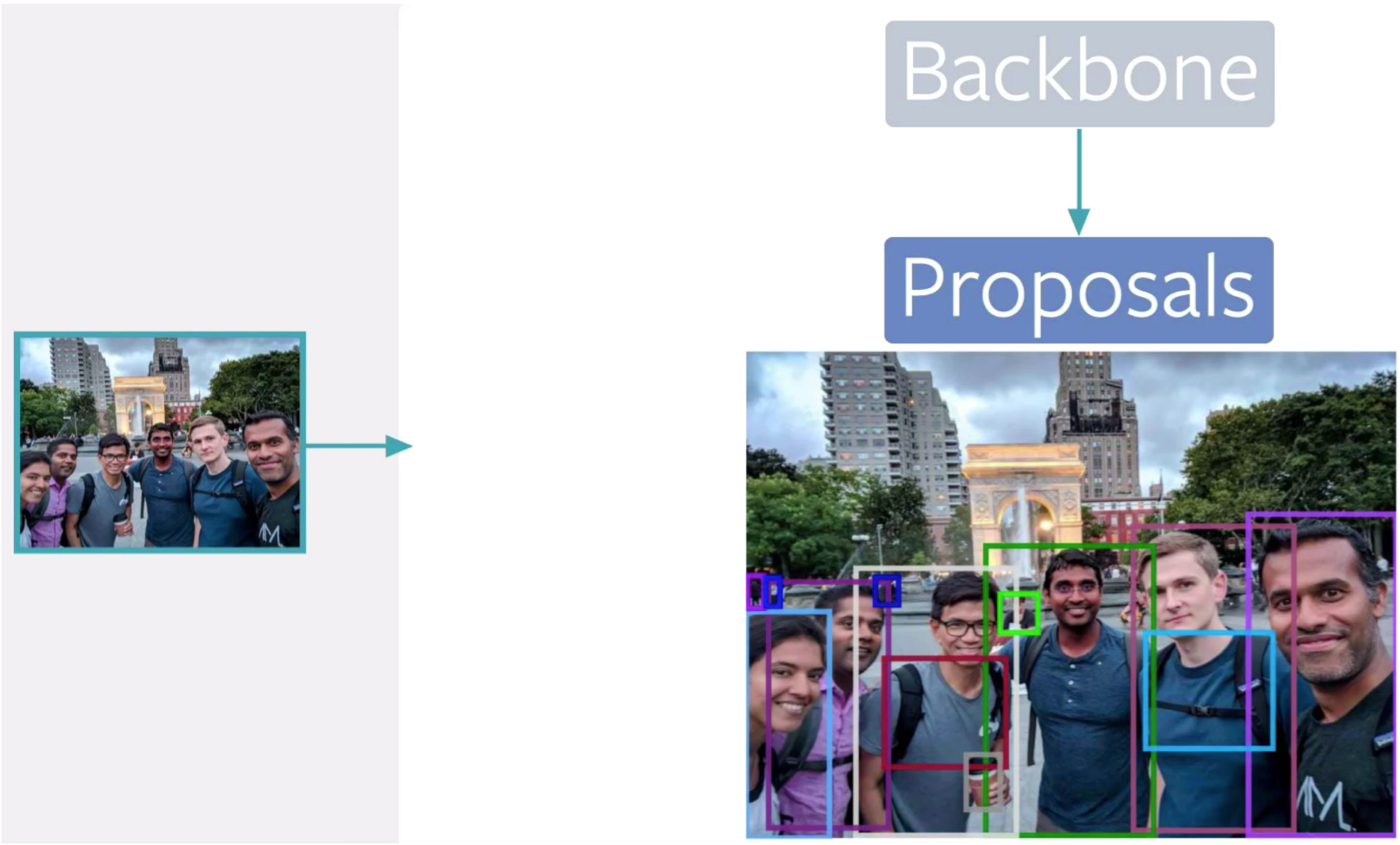


Detectron backbones



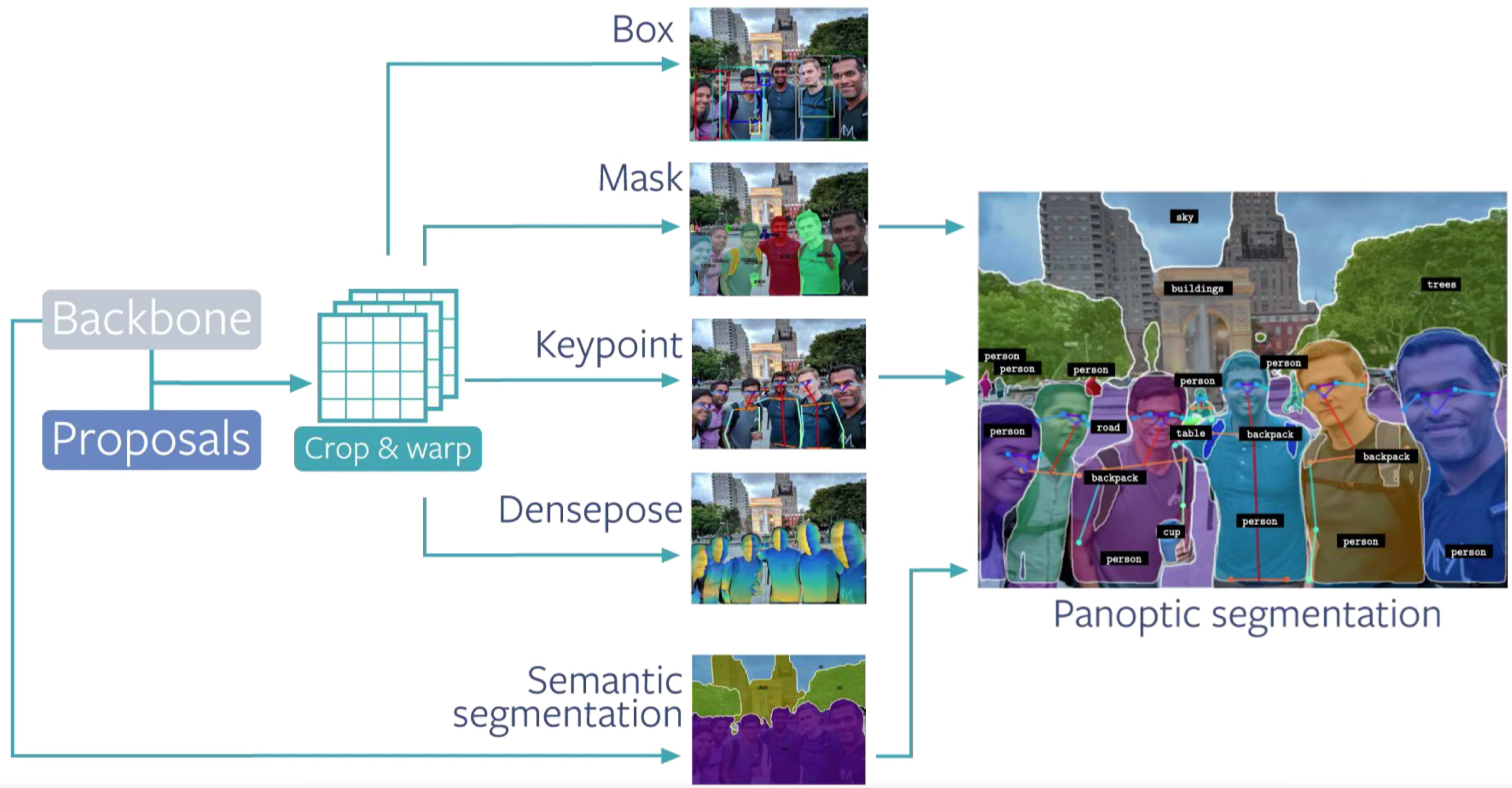
[Figure: Facebook]

Detectron: proposals



[Figure: Facebook]

Detectron: heads



[Figure: Facebook]