курс «Глубокое обучение»

Сегментация изображений

Александр Дьяконов

10 октября 2022 года

Семантическая сегментация

Расстановка меток классов для пикселей

segmented

1: Person
2: Purse
3: Plants/Grass
4: Sidewalk
5: Building/Structures

3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	1	1	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	1	1	1	1	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	1	1	3	3	3	5	5	5	5	5	5	5
5	5	3	3	3	3	1	1	3	3	5	5	5	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	4	4	4	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	4	4	4	4	4	5	5	5
4	4	4	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	4	4	4	4	4	4	4

https://www.jeremyjordan.me/semantic-segmentation/

Семантическая сегментация

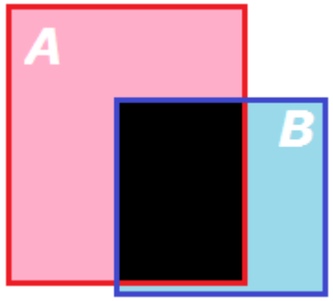
часто: Segmentation accuracy / Pixel accuracy

(для класса – обычная точность по пикселям)

если усредняем по классам – Mean Pixel Accuracy (MPA)

Меры качества – как для множеств

коэффициент Жаккара (Jaccard Index) / IoU – уже знакомый



$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|} \geq \alpha$$

Intersection Over Union

функции ошибки

Focal Loss, Dice loss, IoU, Boundary loss, Weighted cross-entropy, Lovász-Softmax loss

Семантическая сегментация: приложения

- беспилотное вождение
- медицинские изображения
- изображения со спутников
- для других задач (например ИИ)
  - понимание изображений
  - предложение похожего по фото (ex: элементы одежды)
- извлечение изображения отдельных объектов (киноиндустрия)

Датасеты

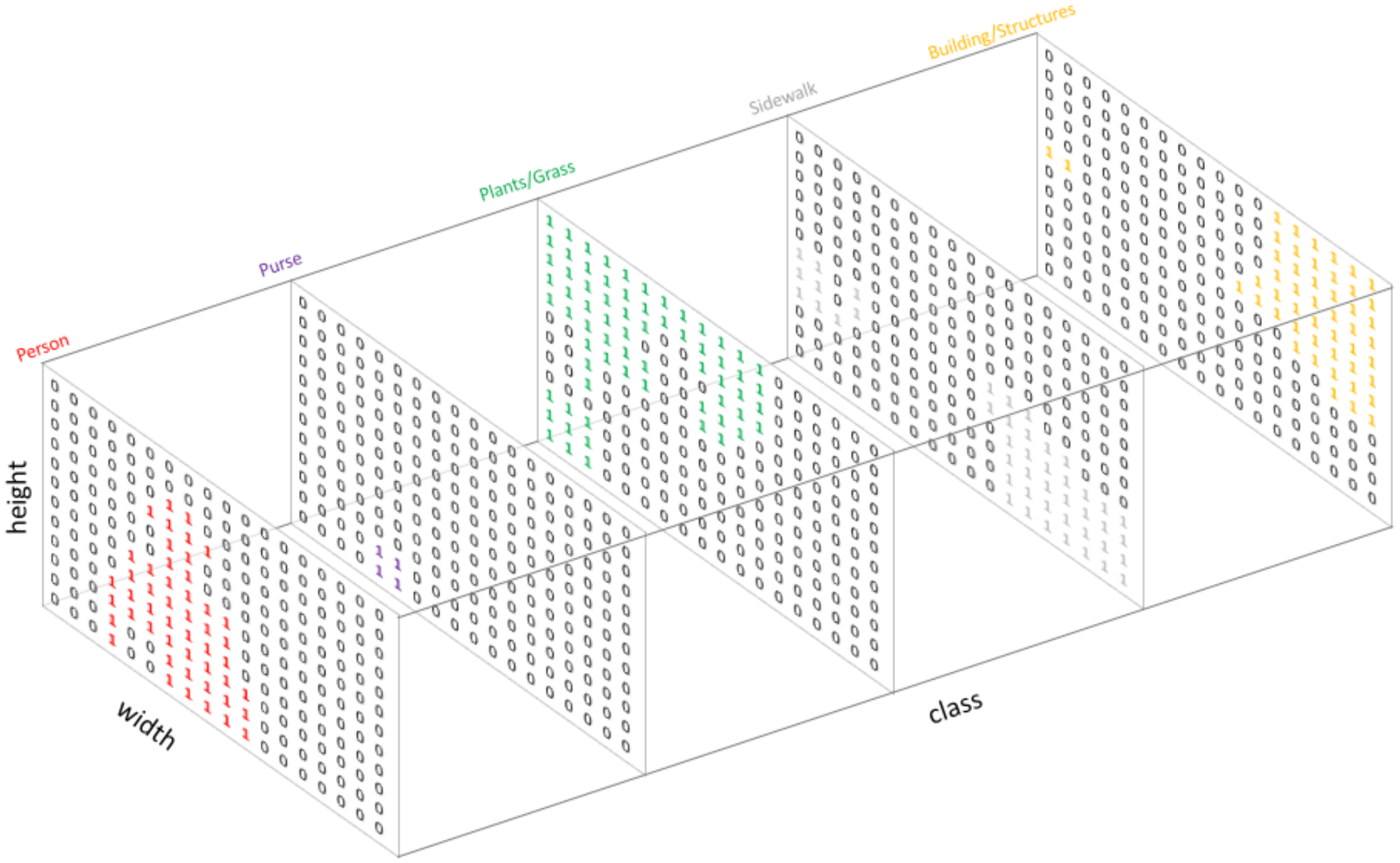
Natural Scenes	Berkeley Segmentation Dataset [140]
	PASCAL VOC [54]
	Stanford Background Dataset [72]
	Microsoft COCO [122]
	MIT Scene parsing data(ADE20K) [222, 223]
	Semantic Boundaries Dataset [75]
	Microsoft Research Cambridge Object Recognition Image Database (MSRC) [188]
Video Segmentation Dataset	Densely Annotated Video Segmentation(DAVIS) [168]
	Video Segmentation Benchmark(VSB100) [64]
	YouTube-Video object Segmentation [209]
Autonomous Driving	Cambridge-driving Labeled Video Database (CamVid) [23]
	Cityscapes: Semantic Urban Scene Understanding [41]
	Mapillary Vistas Dataset [155]
	SYNTHIA: Synthetic collection of Imagery and Annotations [178]
	KITTI Vision Benchmark Suite [67]
	Berkeley Deep Drive [212]
	India Driving Dataset(IDD) [202]
Scene Text Segmentation	KAIST Scene Text Database [112]
	COCO-Text [203]
	SVT: Street View Text Dataset [205]

Aerial Imaging	Inria Aerial Image Labeling Dataset [134]
	Aerial Image Segmentation Dataset [213]
	ISPRS Dataset collection [57]
	Google Open Street Map [8]
	DeepGlobe [49]
Medical Imaging	DRIVE:Digital Retinal Images for Vessel Extraction [191]
	Sunnybrook Cardiac Data [171]
	Multiple Sclerosis Database [129, 25]
	IMT: Intima Media Thickness Segmentation Dataset [148]
	SCR: Segmentation in Chest Radiographs [201]
	BRATS: Brain Tumor Segmentation [146]
	LITS: Liver Tumour Segmentation [74]
	BACH: Breast Cancer Histology [6]
	IDRiD: Indian Diabetic Retinopathy Image Dataset [169]
	ISLES: Ischemic Stroke Lesion Segmentation [135]
Saliency Detection	MSRA Salient Object Database [37]
	ECSSD: Extended Complex Scene Saliency Dataset [187]
	PASCAL-S DATASET [117]
	THUR15K: Group Saliency in Image [36]
	JuddDB: MIT saliency benchmark [18]
	DUT-OMRON Image Dataset [210]

https://arxiv.org/pdf/1907.06119v1.pdf

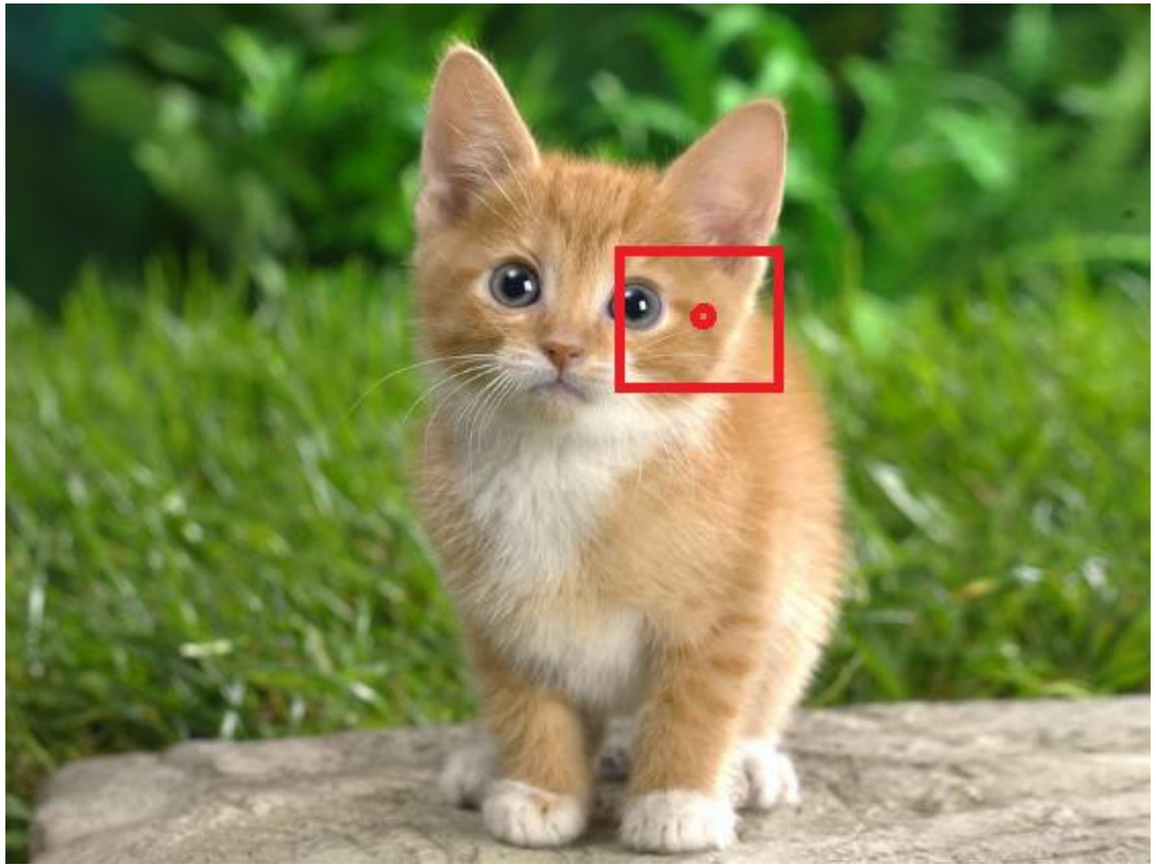
Семантическая сегментация

ONE-кодирование целевого вектора, ошибка = Cross-Entropy



Семантическая сегментация: базовая примитивная идея

Метка по классификации окна с центром в этом пикселе



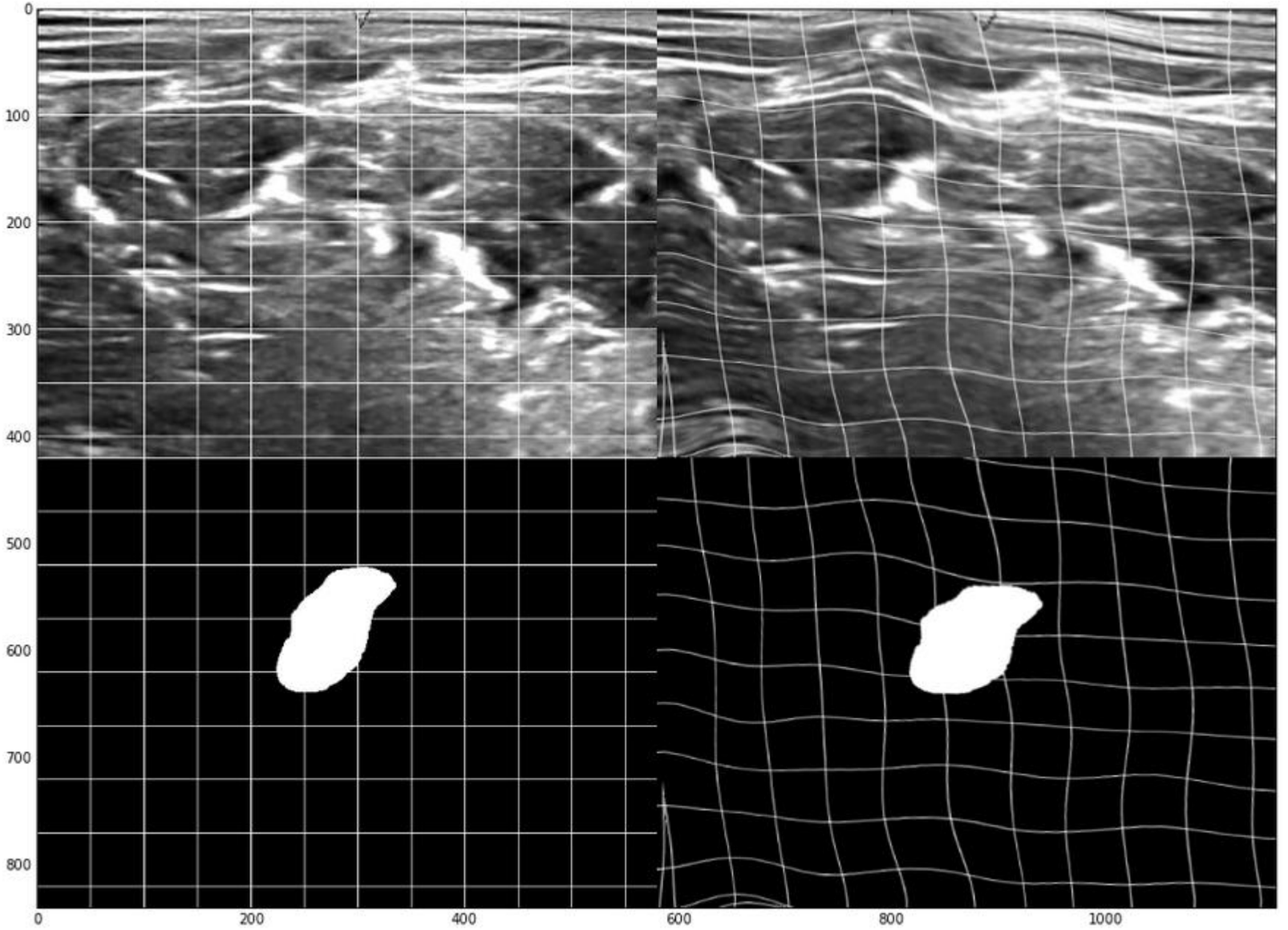
+ работоспособно

– очень долго…

– нет учёта глобального контекста

по маленькому окну не всегда ясен объект

Elastic Transform



https://www.kaggle.com/bguberfain/elastic-transform-for-data-augmentation

Классические методы сегментации: кластеризация по цвету

Можно делать кластеризацию в 5D-пространстве XYRGB
(+ объединение похожих регионов)

- k-means
- non-parametric Bayesian
- energy-based methods

(a) K=50 (b) K=100 (c) K=200 (d) K=300 (e) K=400

https://web.stanford.edu/class/cs231a/lectures/SegmentationLecture.pdf

Классические методы сегментации: условные случайные поля (CRF)

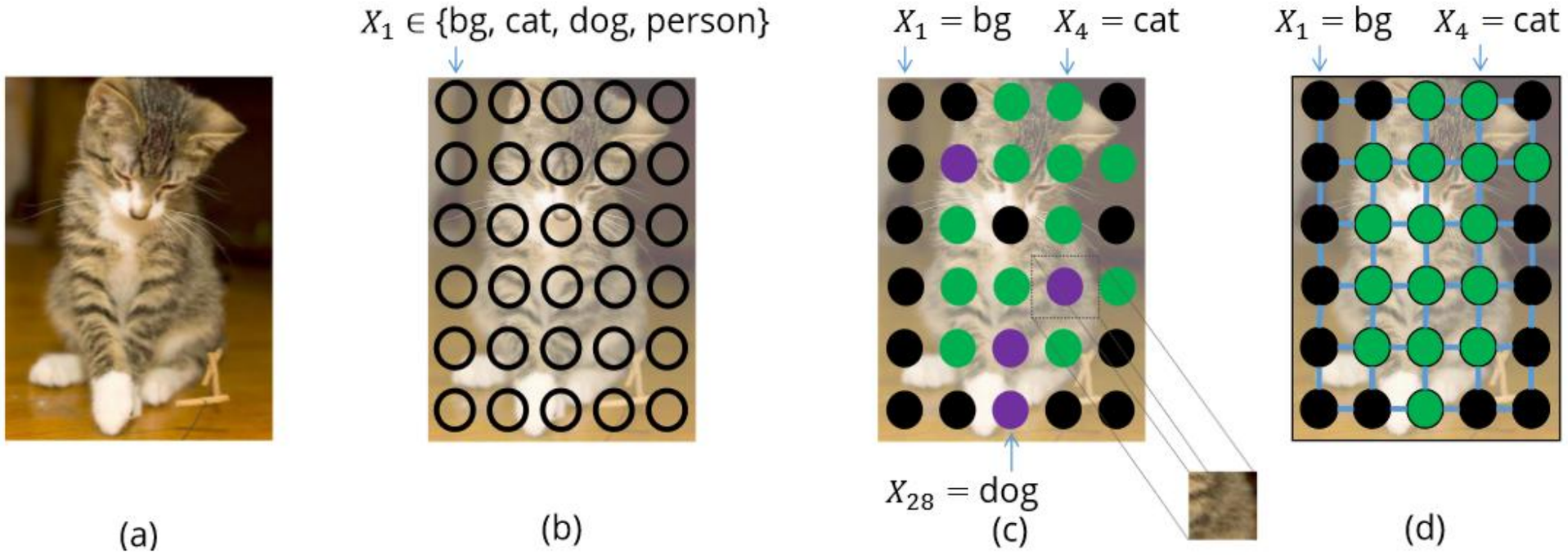
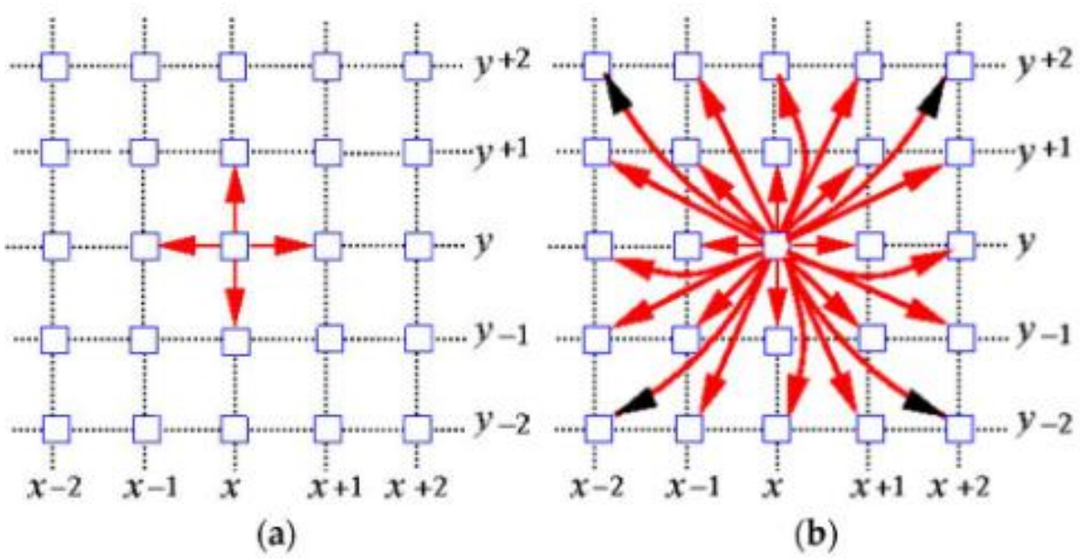


Fig. 2: **Overview of Semantic Segmentation.** Every pixel u is associated with a random variable X_u which takes on a label from a pre-defined label set (b). A naïve method of performing this would be to train a classifier to predict the semantic label of each pixel, using features derived from the image. However, as we can see from (c), this tends to result in very noisy segmentations. This is because in order to predict a pixel, the classifier would only have access to local pixel features, which are often not discriminative enough. By taking the relationship between different pixels into account (such as the fact that if a pixel has been labelled "cat", nearby ones are likely to take the same label since cats are continuous objects) with a CRF, we can improve our labelling (d).

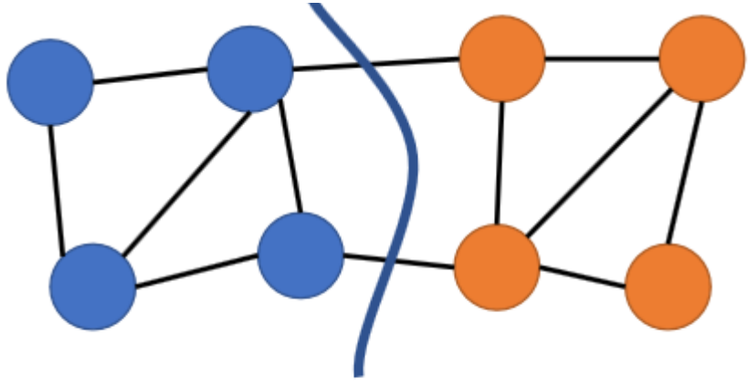
иллюстрация, что попиксельная классификация проваливается

объекты непрерывны ⇒ соседние пиксели должны иметь одинаковые метки

Классические методы сегментации: графовые методы



Задание графа и нахождение разреза

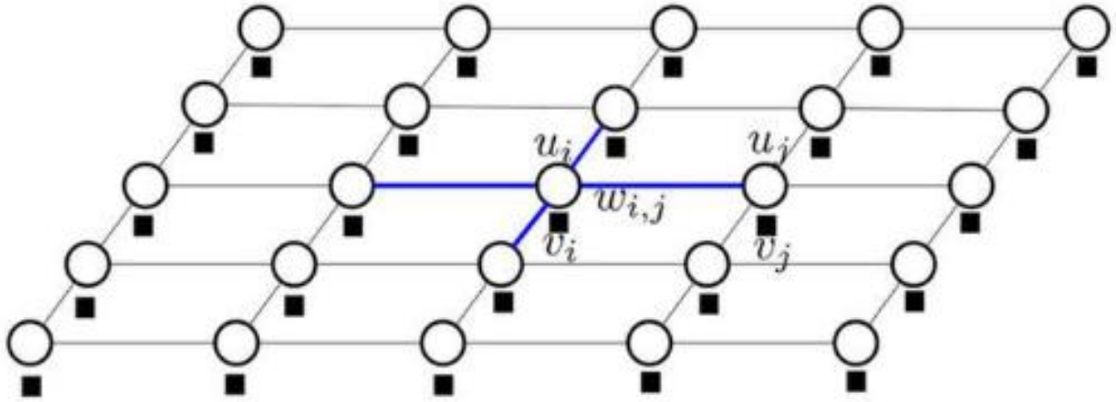


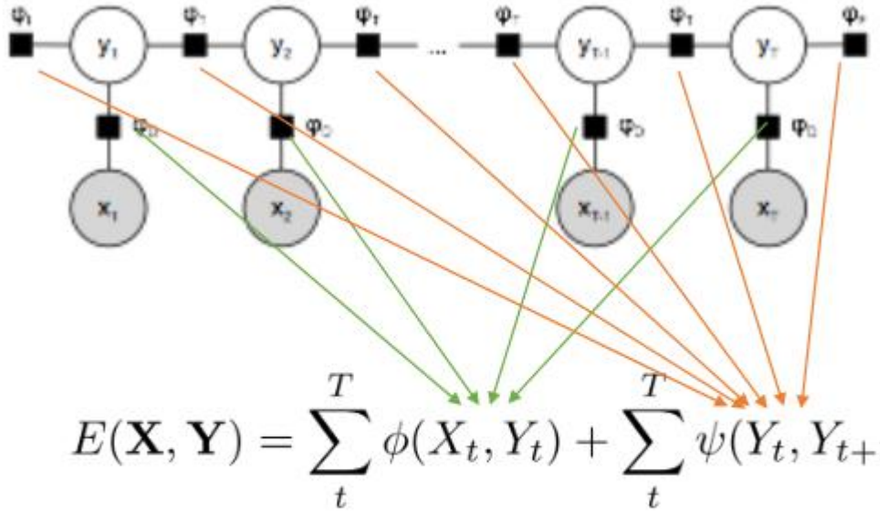
A B

$$\text{cut}(A,B) = \sum_{i\in A, j\in B} W_{ij}$$

чуть хитрее, на самом деле

Классические методы сегментации: 2D CRF





$$E(\mathbf{X},\mathbf{Y}) = \sum_t^T \phi(X_t, Y_t) + \sum_t^T \psi(Y_t, Y_{t+1})$$

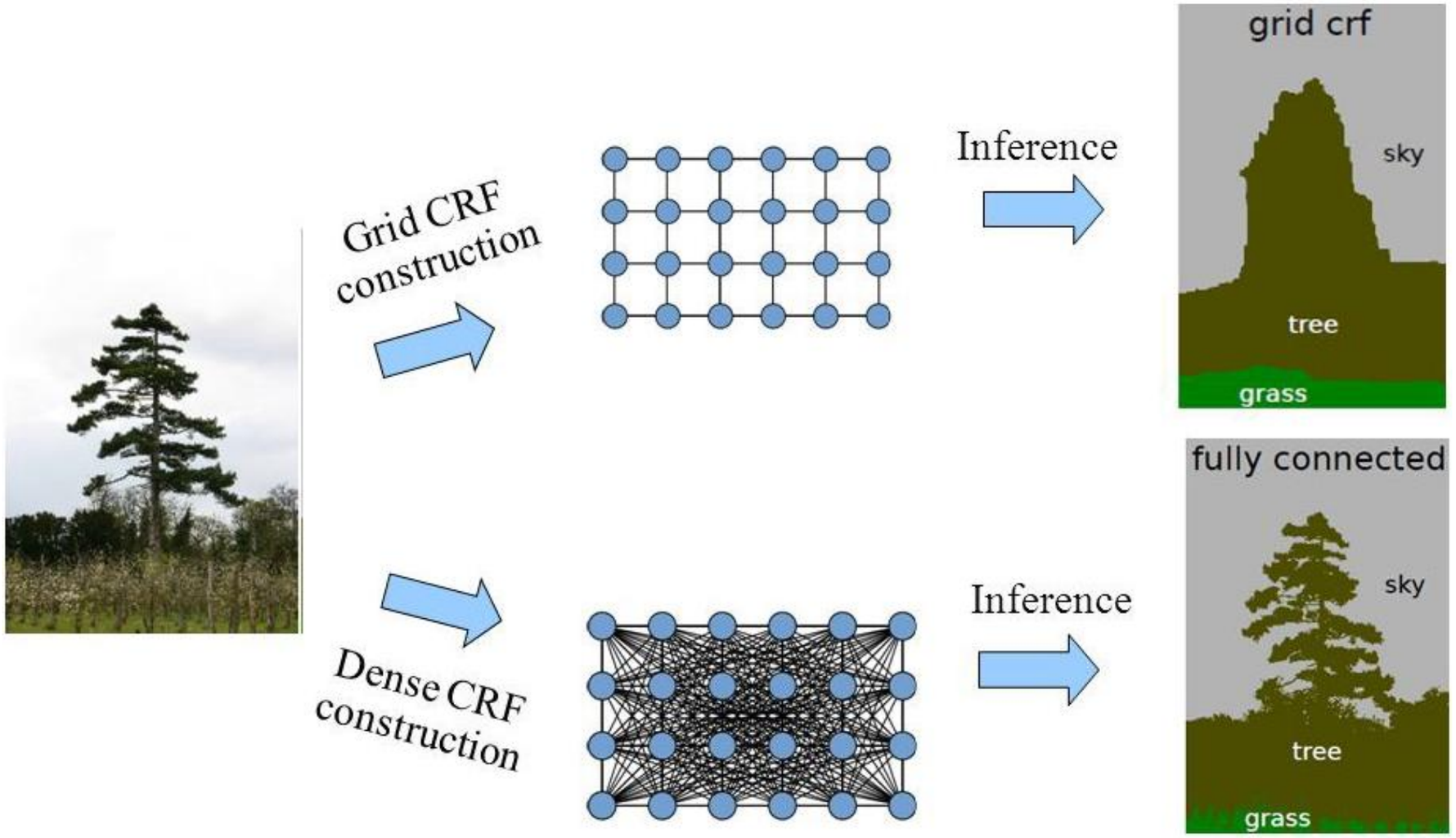
- **Belief Propagation**
- **MCMC**
  - **Metropolis Hastings**
  - **Alpha-expansion, alpha-beta swap**
- **Variational Inference**
  - **Аппроксимация энергии**

$$p(\mathbf{Y}|\mathbf{X}) = \frac{1}{Z}\exp(E(\mathbf{X},\mathbf{Y}))$$

Shuai Zheng «Conditional Random Fields as Recurrent Neural Networks» //
https://www.robots.ox.ac.uk/~szheng/papers/CRFasRNN.pdf

Anurag Arnab et al. «Conditional Random Fields Meet Deep Neural Networks for Semantic Segmentation» //
https://www.robots.ox.ac.uk/~tvg/publications/2017/CRFMeetCNN4SemanticSegmentation.pdf

Классические методы сегментации



в CRF можно подавать результат работы НС!

https://neurohive.io/ru/osnovy-data-science/semantic-segmention/

Гибридный метод: DL + CRF

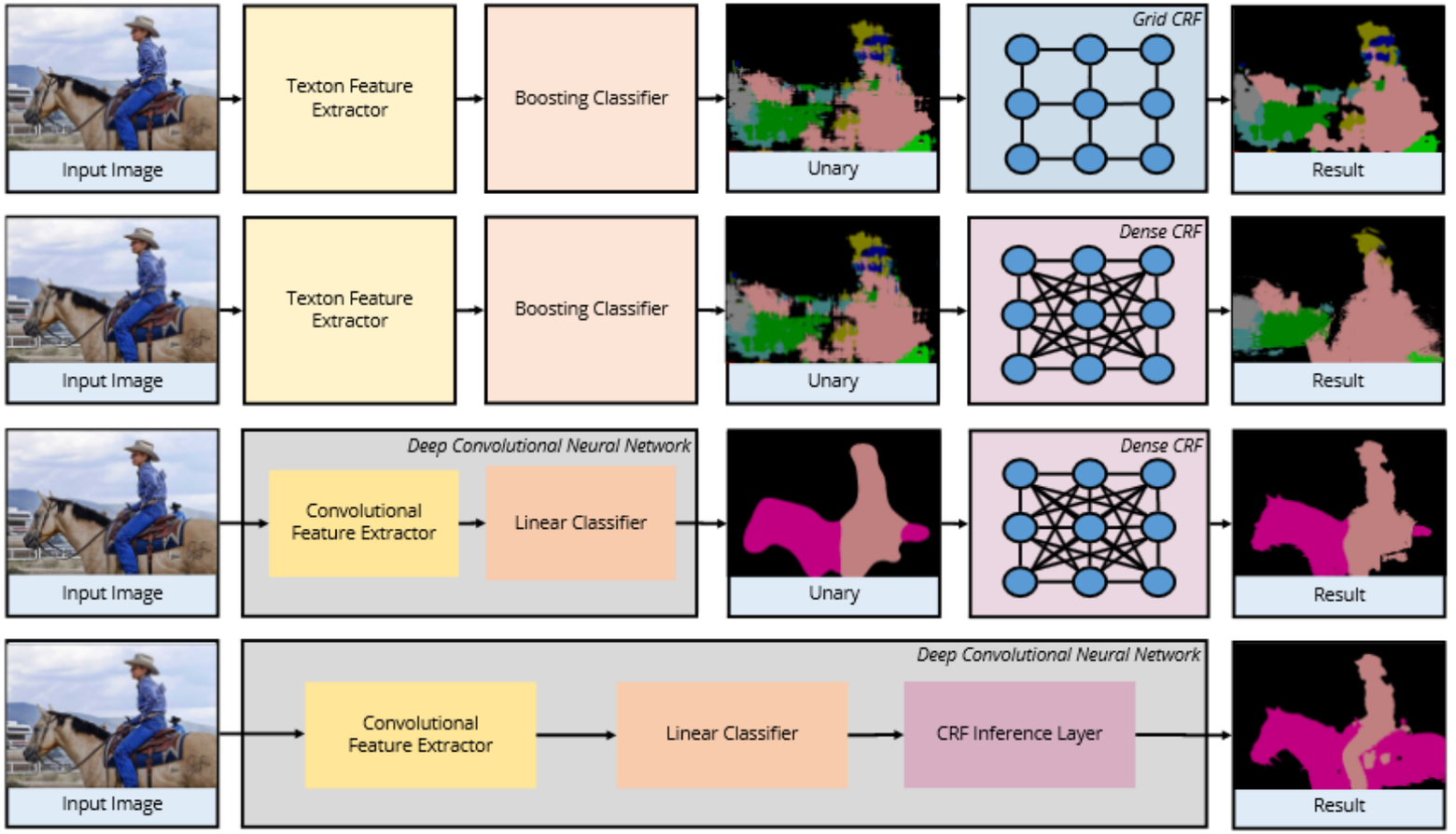
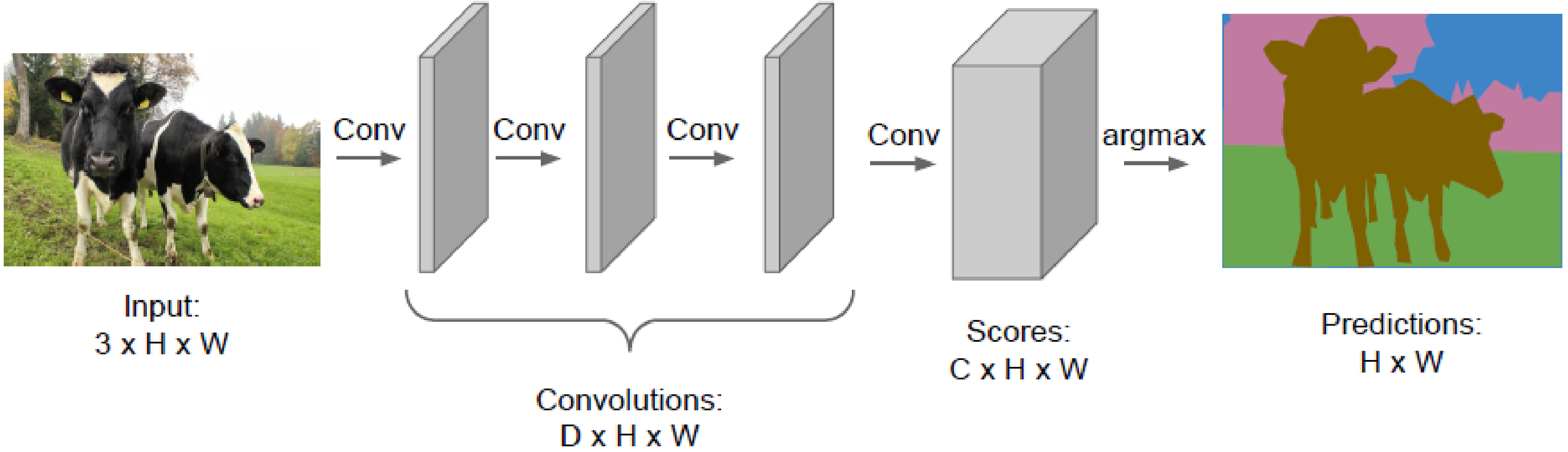


Fig. 3: **The evolution of Semantic Segmentation systems**. The first row shows the early "TextonBoost" work [3] that computed unary potentials using Texton [3] features, and a CRF with limited 8-connectivity. The DenseCRF work of Krahenbuhl and Koltun [4] used densely connected pairwise potentials and approximate mean-field inference. The more expressive model achieved significantly improved performance. Numerous works, such as [5] have replaced the early hand-crafted features with Deep Neural Networks which can learn features from large amounts of data, and used the outputs of a CNN as the unary potentials of a DenseCRF model. In fact, works such as [6] showed that unary potentials from CNNs (without any CRF) on their own achieved significantly better results than previous methods. Current state-of-the-art methods have all combined inference of a CRF within the deep neural network itself, allowing for end-to-end training [7], [8]. Result images for this figure were obtained using the publicly available code of [9], [4], [5], [7], [6].

Полностью свёрточная сеть – Fully Convolutional Network (FCN)

Идея: раз мы реализуем отображение
картинка (тензор) → картинка (тензор),
то почему бы не использовать
полностью свёрточную сеть – FCN



Полностью свёрточная сеть – Fully Convolutional Network (FCN)

Проблемы: высокое разрешение исходного изображения

«receptive field» очень маленькая

свёртки на большом разрешении – долго / дорого

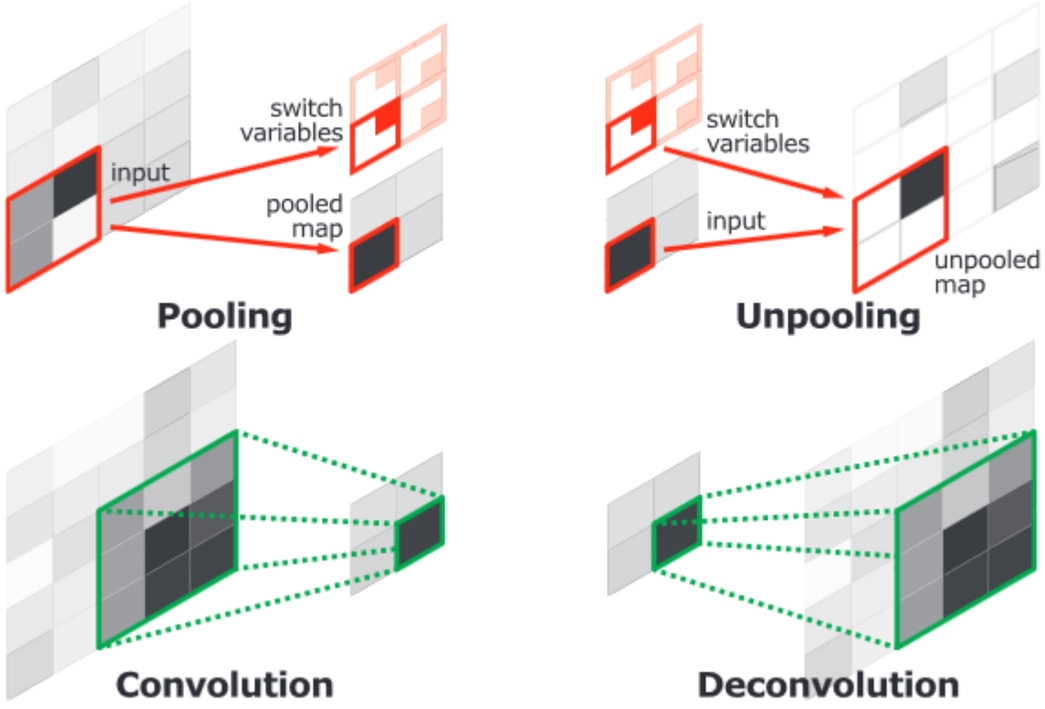


Figure 3. Illustration of deconvolution and unpooling operations.

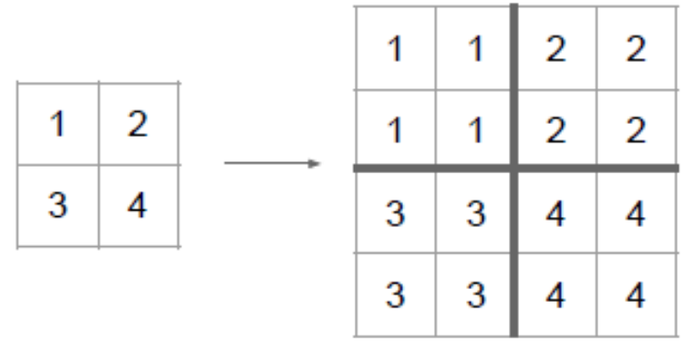
https://arxiv.org/pdf/1505.04366.pdf

- **надо его понизить (downsampling)**
  - **pooling**
  - **strided convolution**
- **потом надо восстановить (upsampling)**

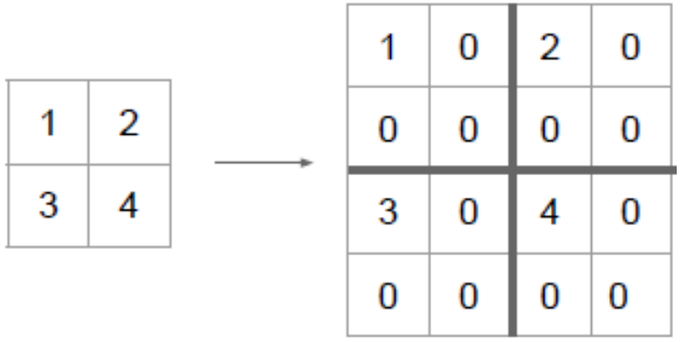
http://cs231n.stanford.edu

FCN: восстановление изображения (upsampling)

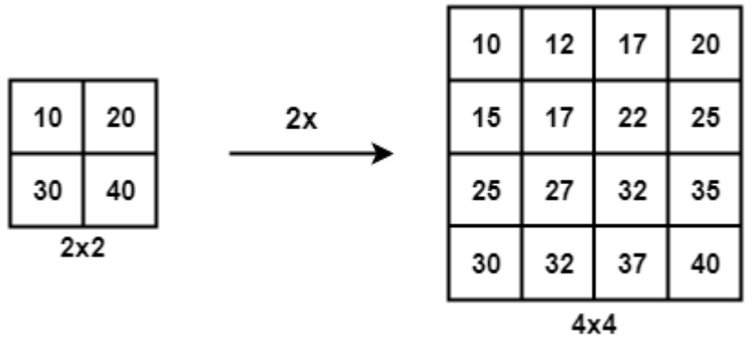
Nearest Neighbors



Bed Of Nails

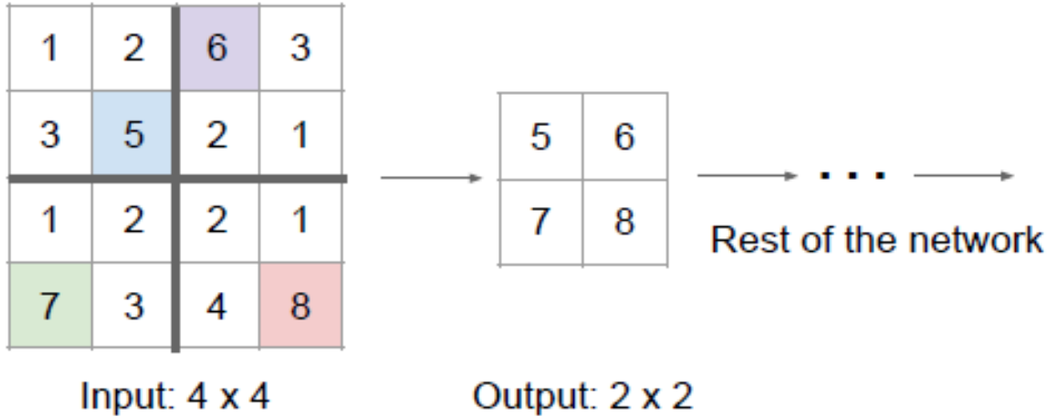


Bi-Linear Interpolation

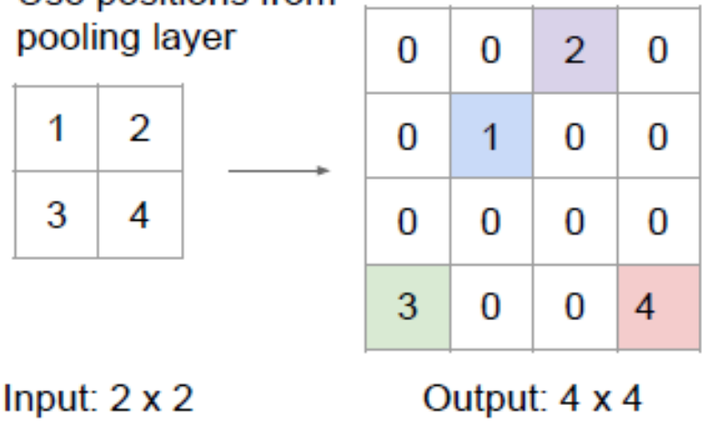


Max-Unpooling

Max Pooling
Remember which element was max!



Max Unpooling
Use positions from pooling layer

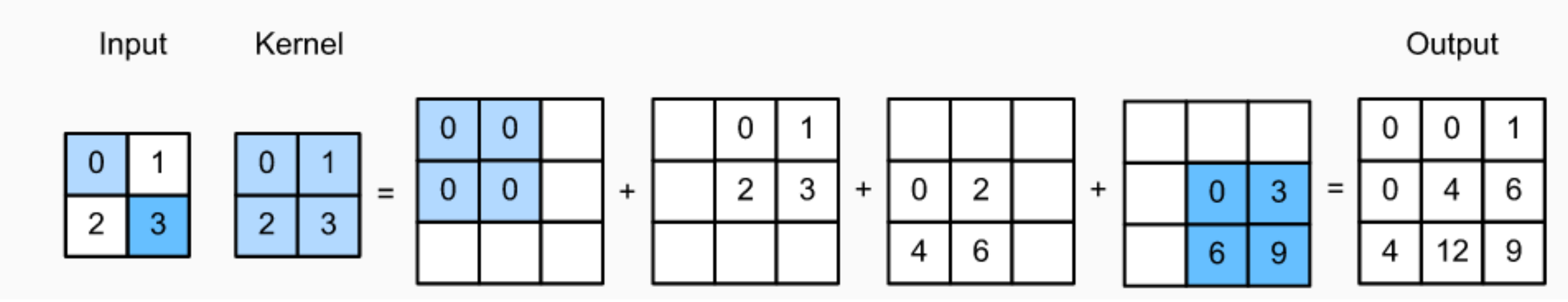


https://towardsdatascience.com/transposed-convolution-demystified-84ca81b4baba

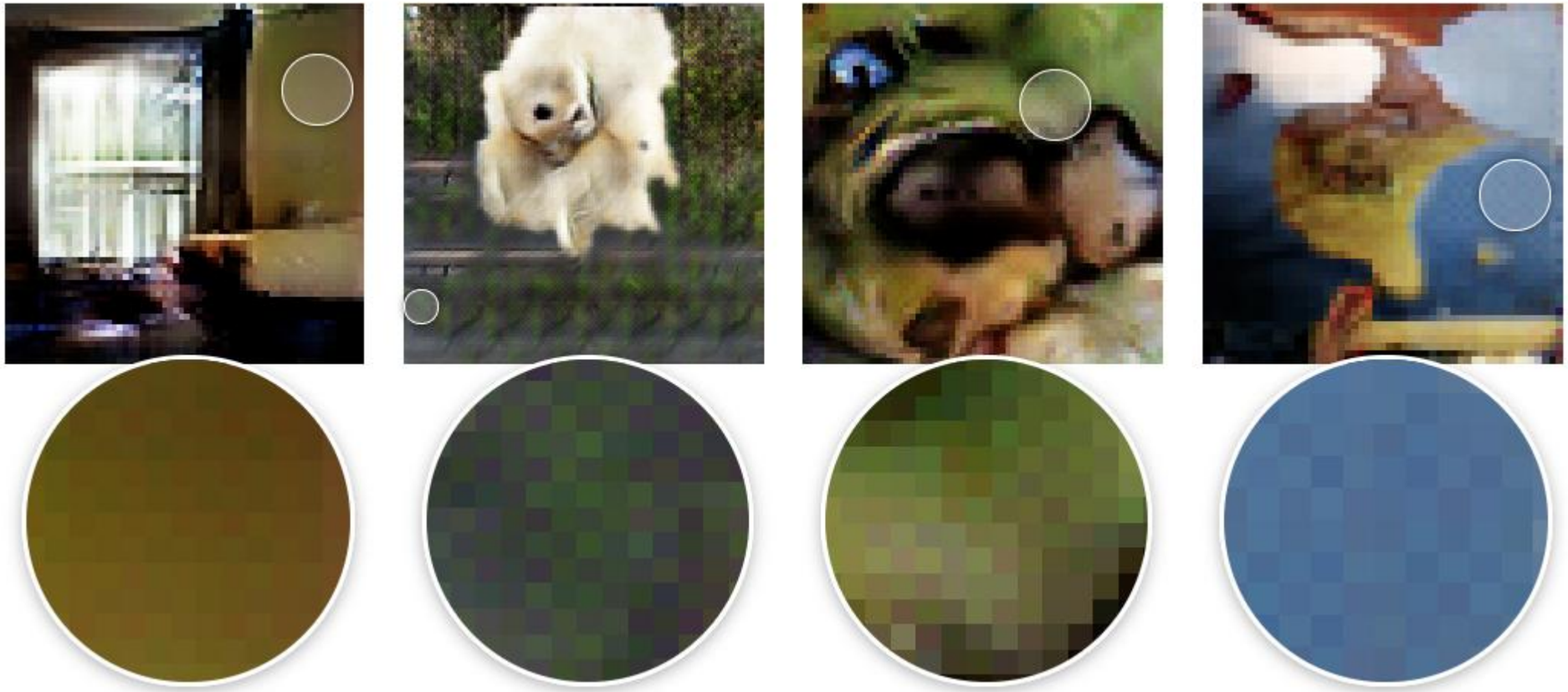
Transposed convolution

Следующий способ – обучаемое увеличенние

Уже была… «транспонированная свёртка»



Проблемы с «обратными» свёртками

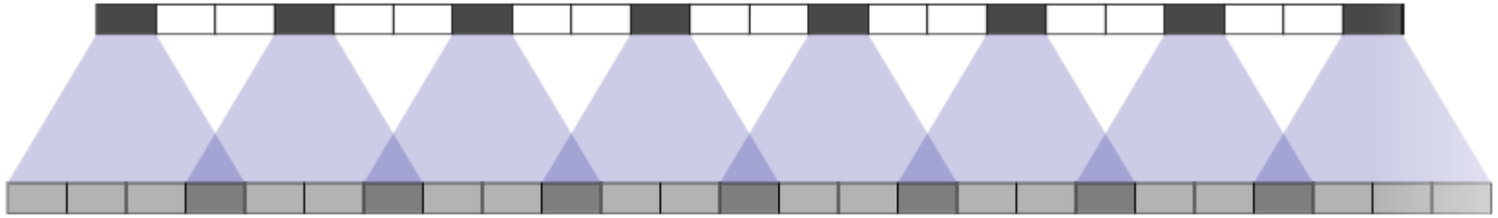


эффект «шахматной доски» – лучше если размер (kernel-size) делит шаг (stride)

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

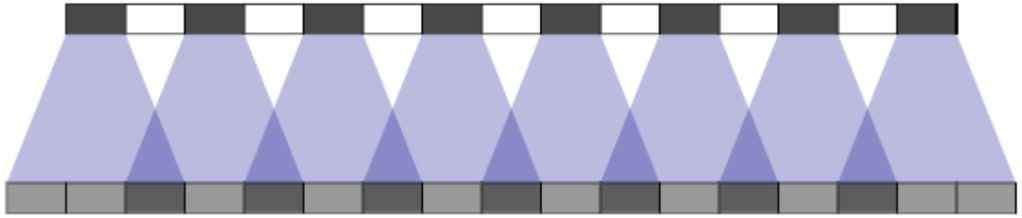
Причина шахматных сеток на изображениях

Как происходит обратная свёртка:



stride = 3

size = 4



stride = 2

size = 3

Один из способов борьбы: nearest neighbour resampling + convolution

Семантическая сегментация

В итоге что-то такое… «Deconvolution Network»

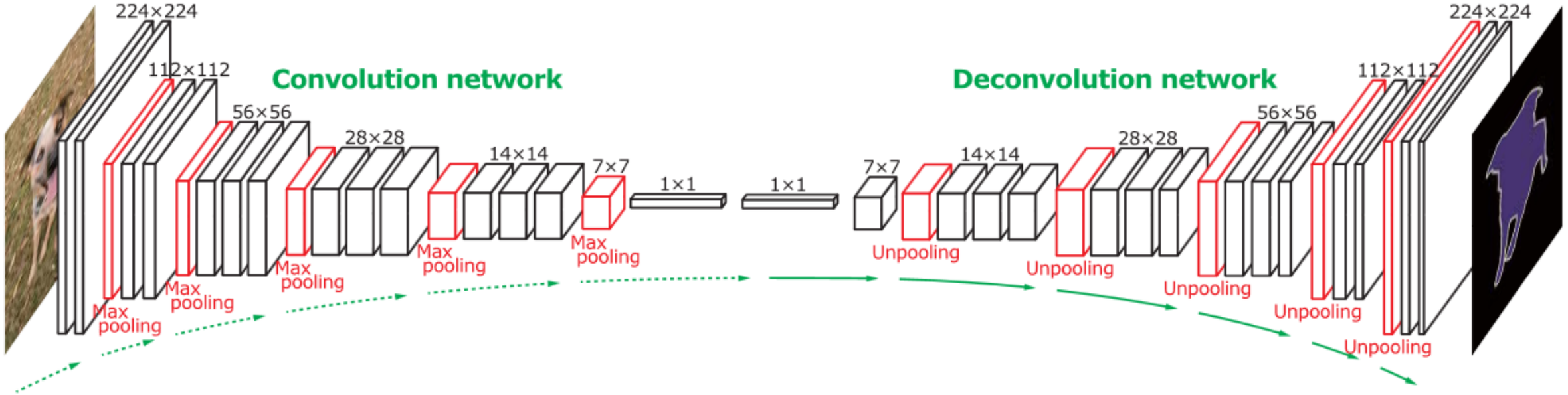


Figure 2. Overall architecture of the proposed network. On top of the convolution network based on VGG 16-layer net, we put a multi-layer deconvolution network to generate the accurate segmentation map of an input proposal. Given a feature representation obtained from the convolution network, dense pixel-wise class prediction map is constructed through multiple series of unpooling, deconvolution and rectification operations.

кодировщик из VGG16

уменьшаются размеры, но увеличивается число каналов!

https://arxiv.org/pdf/1505.04366.pdf

Fully Convolutional Networks (FCN): первая успешная попытка..

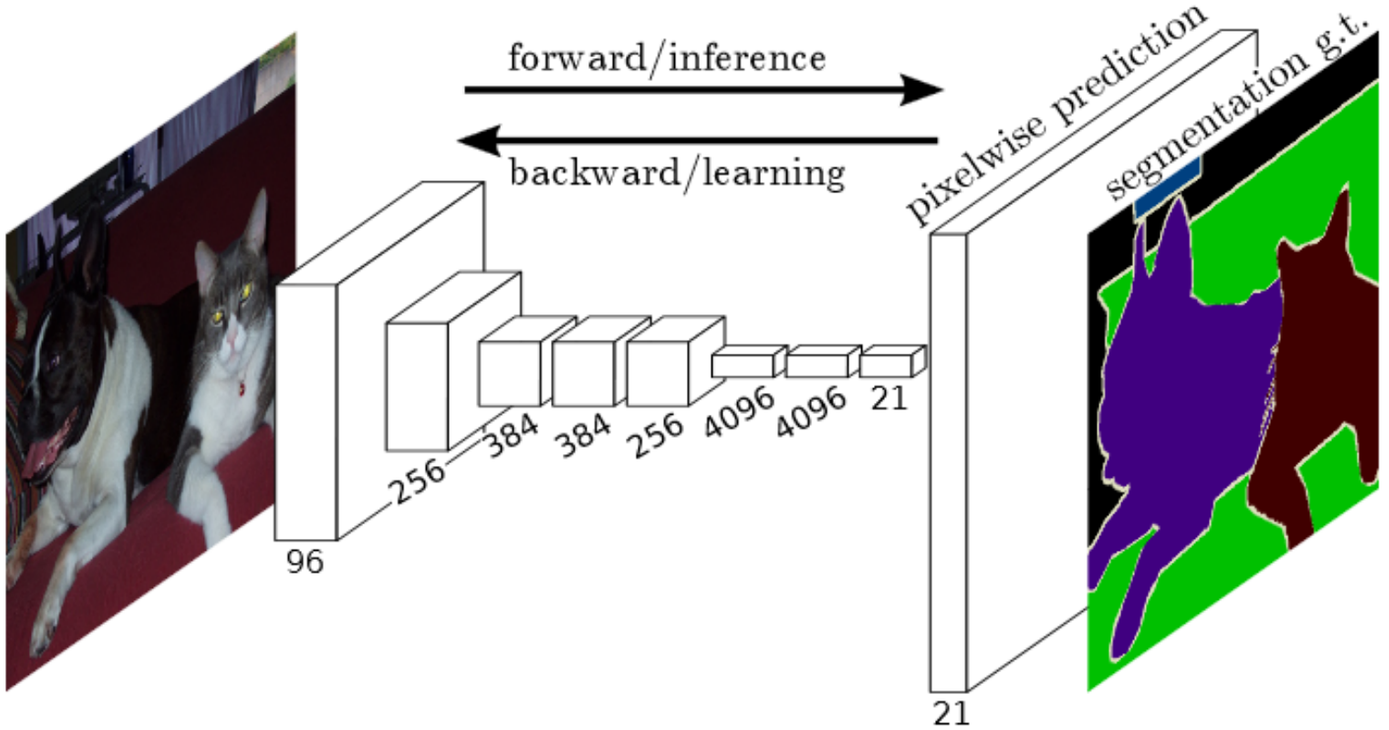


Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.

Long et al., «Fully Convolutional Networks for Semantic Segmentation», 2015 // https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

Fully Convolutional Networks (FCN)

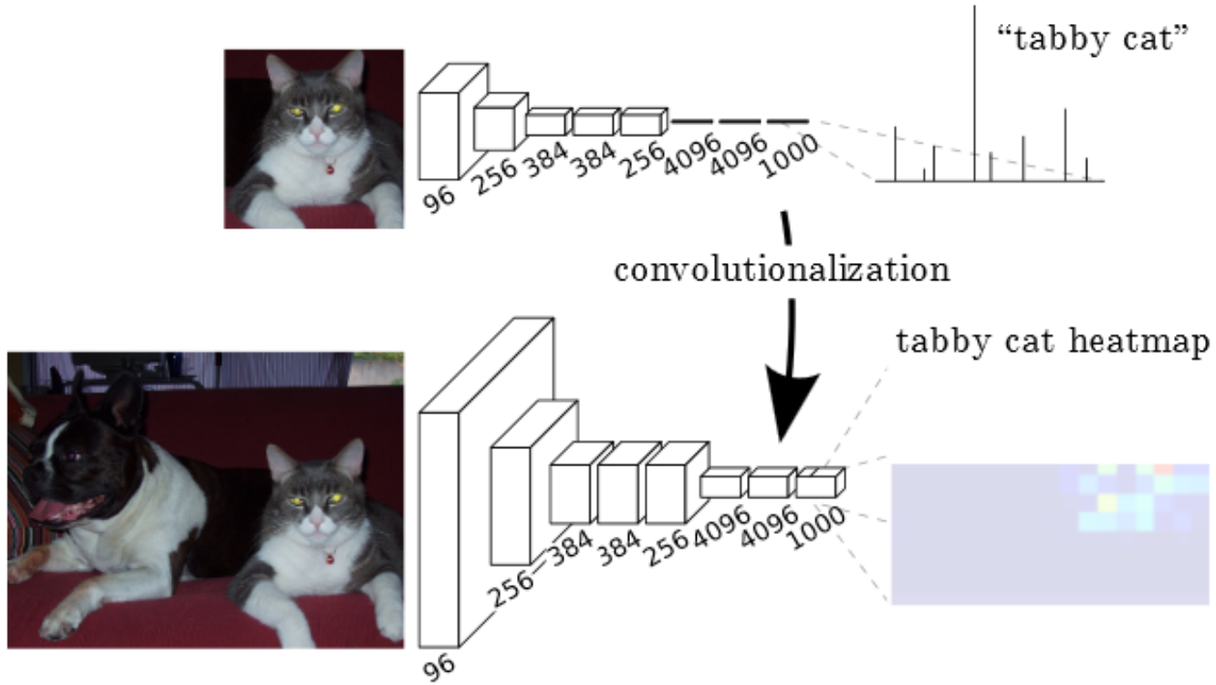


Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.

взять готовую сеть, считать, что полносвязность = свёртка на своём регионе

тогда пусть она этой свёрткой переводит в тензор, получаем подсветку классов

Fully Convolutional Networks (FCN)

+ Полуляризация end2end CNN подхода

+ Transfer Learning – использование предобученных сетей

+ Увеличение масштаба с помощью transposed layers

– при этом потеря информации

Loss: cross-entropy loss для каждого пикселя

Fully Convolutional Networks (FCN)

Понимаем «что» (на последних слоях),
но забываем «где» (было на первых)

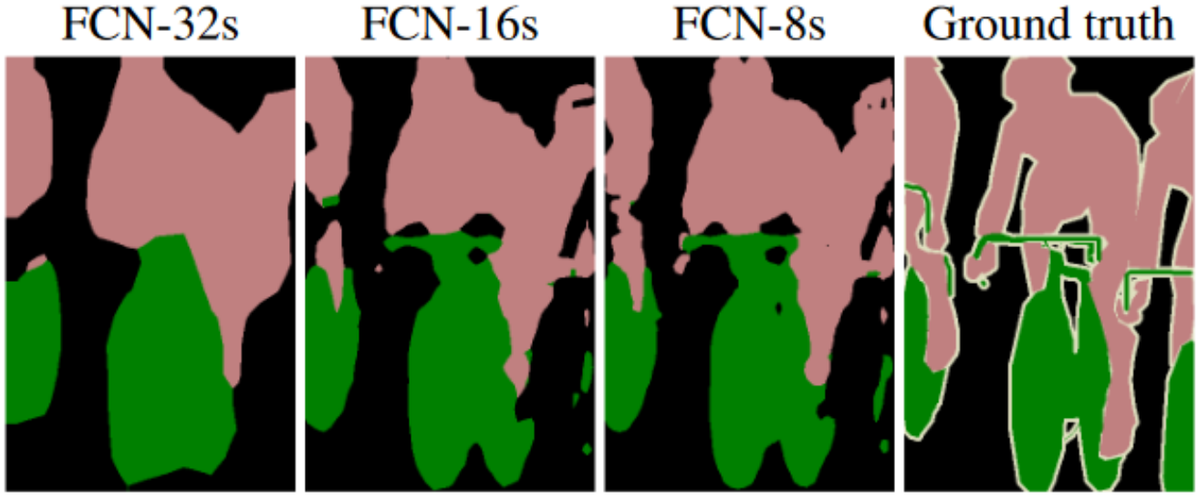


Figure 4. Refining fully convolutional nets by fusing information from layers with different strides improves segmentation detail. The first three images show the output from our 32, 16, and 8 pixel stride nets (see Figure 3).

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**

сейчас будем делать лучше и лучше (FCN-32s → FCN-8s)

Fully Convolutional Networks (FCN)

Решение – прокидывание связей (skip connections)

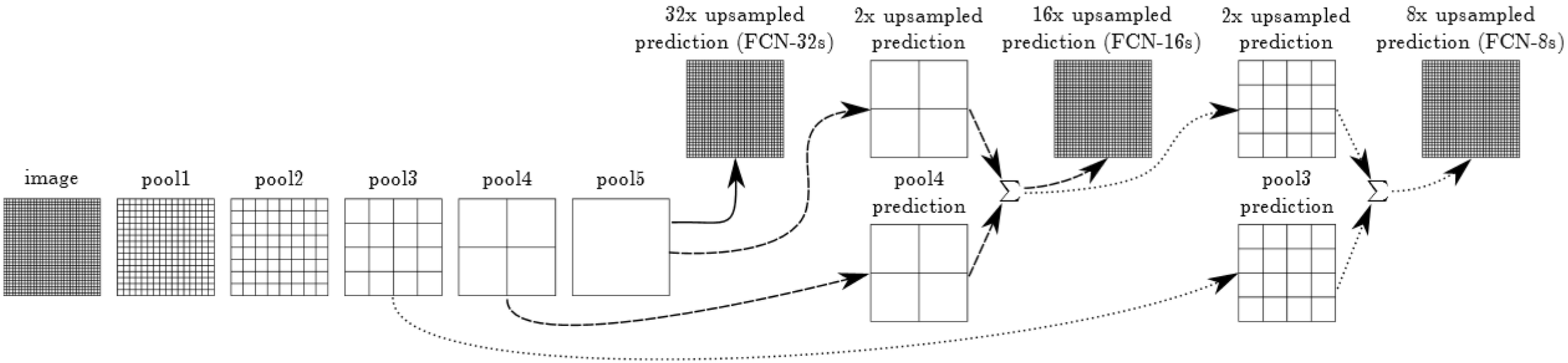


Figure 3. Our DAG nets learn to combine coarse, high layer information with fine, low layer information. Layers are shown as grids that reveal relative spatial coarseness. Only pooling and prediction layers are shown; intermediate convolution layers (including our converted fully connected layers) are omitted. Solid line (FCN-32s): Our single-stream net, described in Section 4.1, upsamples stride 32 predictions back to pixels in a single step. Dashed line (FCN-16s): Combining predictions from both the final layer and the `pool4` layer, at stride 16, lets our net predict finer details, while retaining high-level semantic information. Dotted line (FCN-8s): Additional predictions from `pool3`, at stride 8, provide further precision.

– шахматные эффекты

– низкое разрешение на краях

Segnet

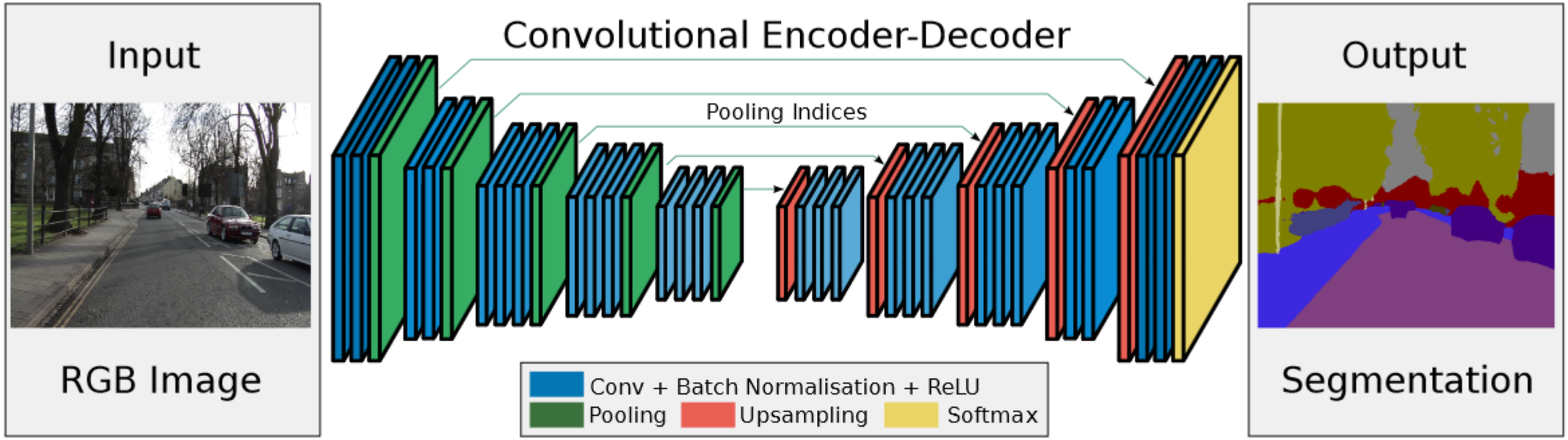


Fig. 2. An illustration of the SegNet architecture. There are no fully connected layers and hence it is only convolutional. A decoder upsamples its input using the transferred pool indices from its encoder to produce a sparse feature map(s). It then performs convolution with a trainable filter bank to densify the feature map. The final decoder output feature maps are fed to a soft-max classifier for pixel-wise classification.

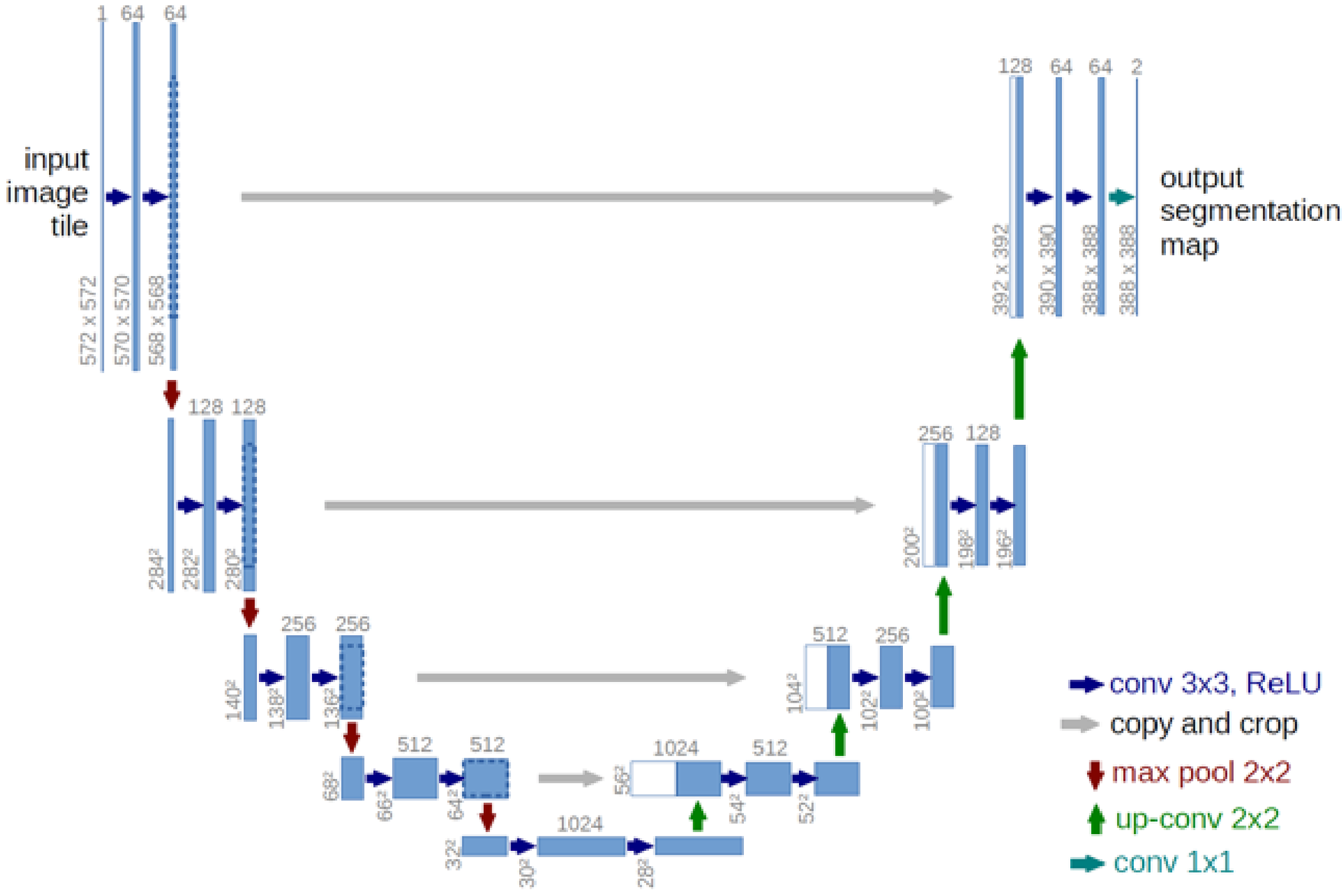
кодировщик ~ 13 свёрточных слоёв VGG16

относительно немного параметров

использование «обратного» пулинга для увеличения размеров (храним индексы)

https://arxiv.org/pdf/1511.00561.pdf

U-Net (самая популярная): long skip connections



«U-Net: Convolutional Networks for Biomedical Image Segmentation» [Ronneberger O. и др., 2015 https://arxiv.org/abs/1505.04597»

U-Net

Левая часть

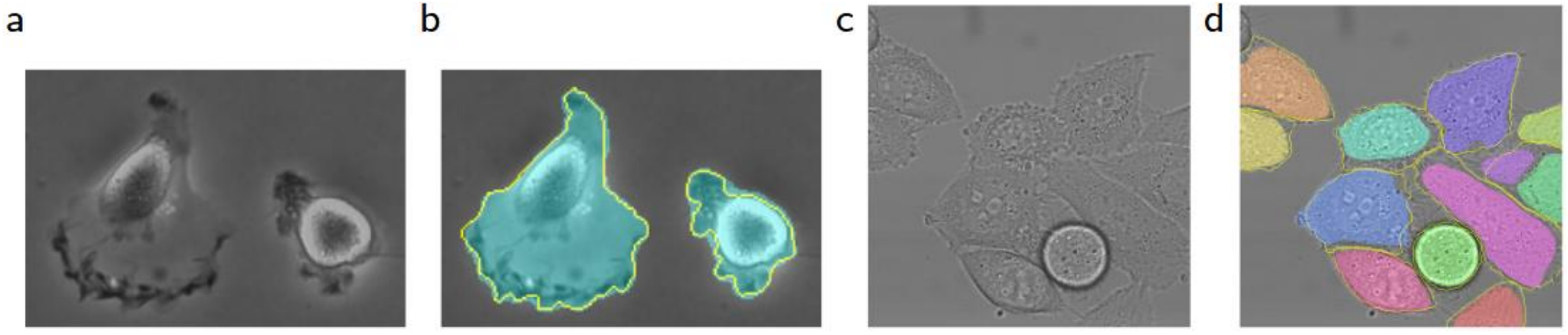
3×3-свёртки (unpadded), ReLU, 2×2-пулинг (шаг=2)

Правая часть

2×2 up-convolutions (половинит число каналов), конкатенация с обрезанными признаками из левой части (там больше H×W), 3×3-свёртки, ReLU

при конкатенации обрезка, чтобы совпали размеры

всего 23 свёрточных слоя, реализация – Caffe



U-Net

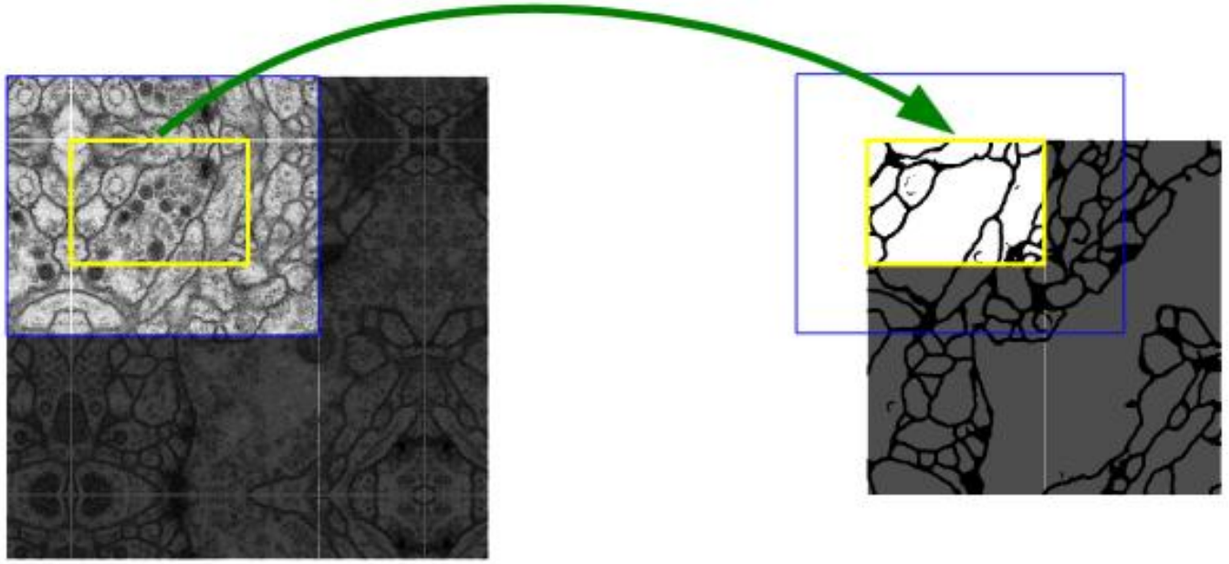


Fig. 2. Overlap-tile strategy for seamless segmentation of arbitrary large images (here segmentation of neuronal structures in EM stacks). Prediction of the segmentation in the yellow area, requires image data within the blue area as input. Missing input data is extrapolated by mirroring

Польза аугментации в некоторых задачах!

Это была фишка статьи – данных можно сделать много!

10 часов на NVidia Titan GPU (6 GB)

Минутка кода: U-Net

```
import torch.nn.functional as F
class UNet(nn.Module):
    def __init__(self, n_channels,
                       n_classes, bilinear=True):
        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
```

```
class DoubleConv(nn.Module):
    """(convolution => [BN] => ReLU) * 2"""
    def __init__(self, in_channels, out_channels, mid_channels=None):
        super().__init__()
        if not mid_channels:
            mid_channels = out_channels
        self.double_conv = nn.Sequential(
                                nn.Conv2d(in_channels, mid_channels,
                                          kernel_size=3, padding=1),
                                nn.BatchNorm2d(mid_channels),
                                nn.ReLU(inplace=True),
                                nn.Conv2d(mid_channels, out_channels,
                                          kernel_size=3, padding=1),
                                nn.BatchNorm2d(out_channels),
                                nn.ReLU(inplace=True)
                                )
    def forward(self, x):
        return self.double_conv(x)

class Down(nn.Module):
    """Downscaling with maxpool then double conv"""
    def __init__(self, in_channels, out_channels):
        super().__init__()
        self.maxpool_conv = nn.Sequential(
                                 nn.MaxPool2d(2),
                                 DoubleConv(in_channels,
                                            out_channels))
    def forward(self, x):
        return self.maxpool_conv(x)
```

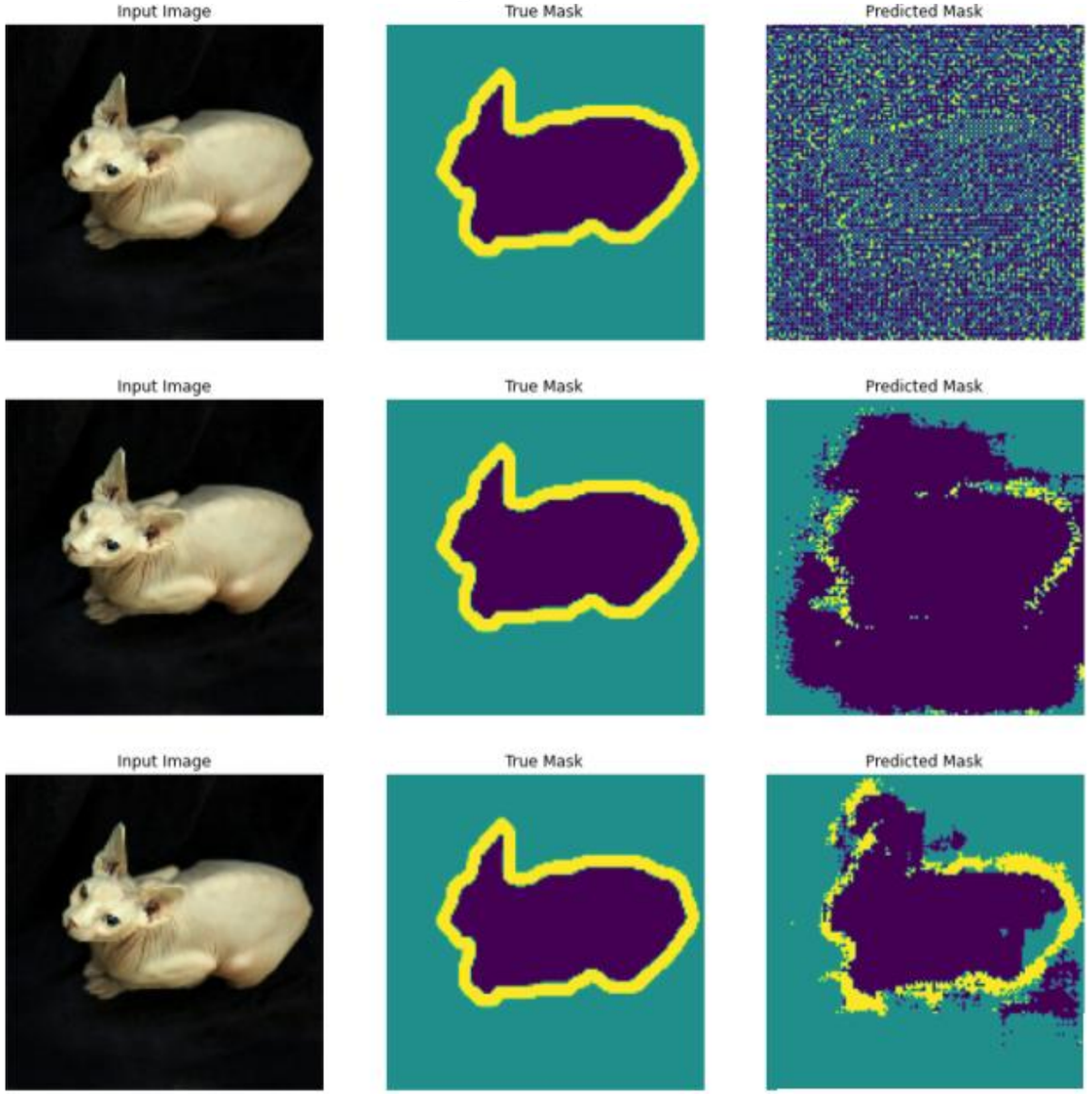
```
class Up(nn.Module):
    """Upscaling then double conv"""
    def __init__(self, in_channels, out_channels, bilinear=True):
        super().__init__()
        # if bilinear, use the normal convolutions to reduce the number of channels
        if bilinear:
            self.up = nn.Upsample(scale_factor=2, mode='bilinear', align_corners=True)
            self.conv = DoubleConv(in_channels, out_channels, in_channels // 2)
        else:
            self.up = nn.ConvTranspose2d(in_channels , in_channels // 2, kernel_size=2, stride=2)
            self.conv = DoubleConv(in_channels, out_channels)
    def forward(self, x1, x2):
        x1 = self.up(x1)
        # input is CHW
        diffY = x2.size()[2] - x1.size()[2]
        diffX = x2.size()[3] - x1.size()[3]

        x1 = F.pad(x1, [diffX // 2, diffX - diffX // 2,
                        diffY // 2, diffY - diffY // 2])
        x = torch.cat([x2, x1], dim=1)
        return self.conv(x)

class OutConv(nn.Module):
    def __init__(self, in_channels, out_channels):
        super(OutConv, self).__init__()
        self.conv = nn.Conv2d(in_channels, out_channels, kernel_size=1)
    def forward(self, x):
        return self.conv(x)
```

иногда BN не делают

Используют Upsample или Conv2DTranspose (обучаются веса)

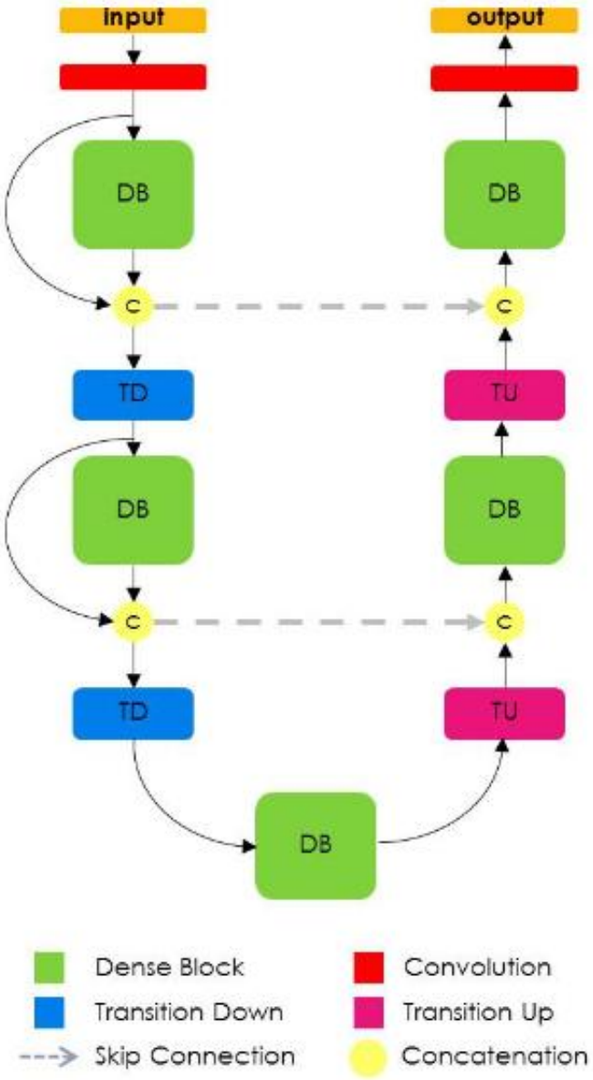
https://github.com/milesial/Pytorch-UNet/tree/master/unet

Как обучается U-net



[Practical Machine Learning for Computer Vision]

«Тирамису» = DenseNet + U-Net

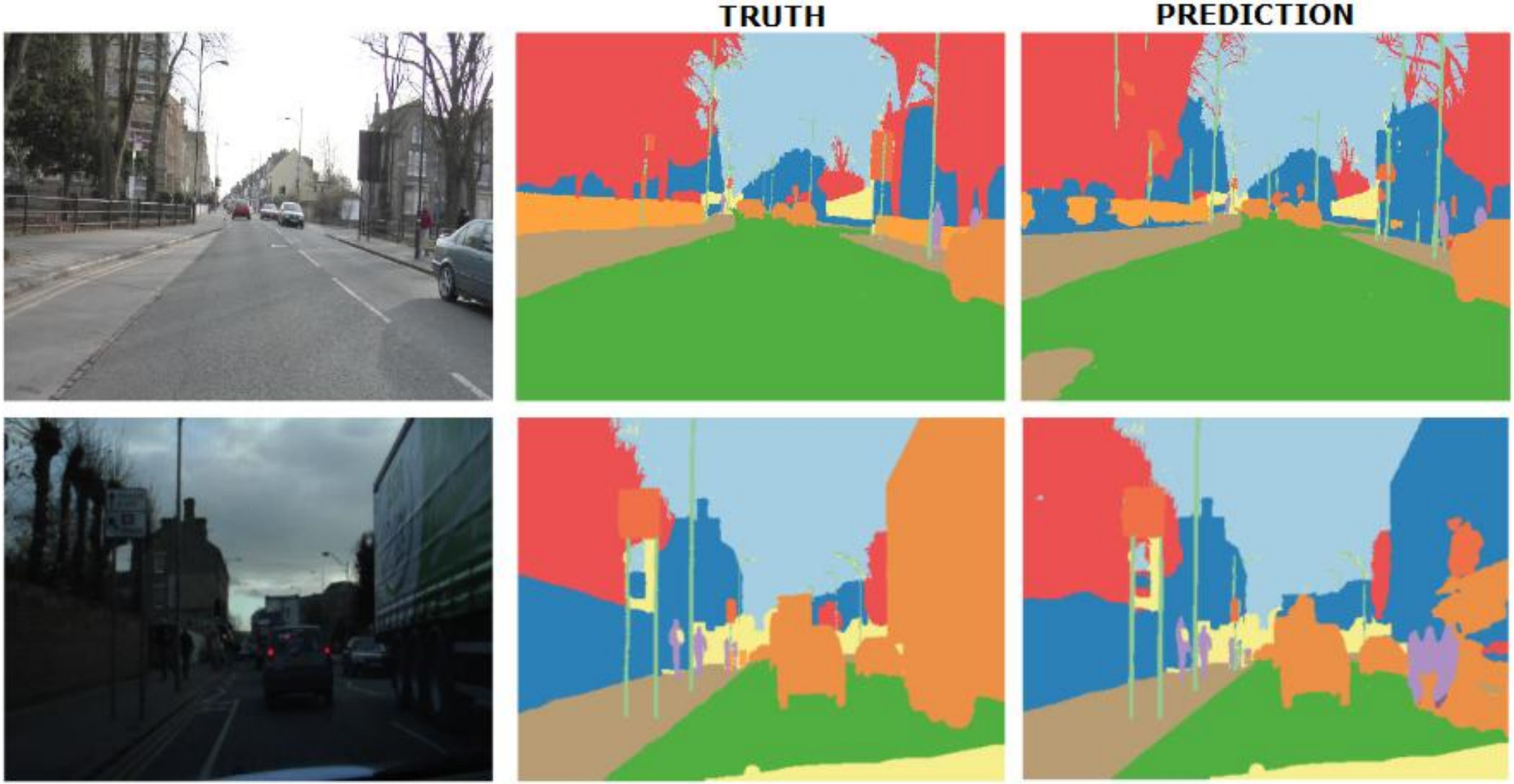


HeUniform + RMSprop + weight Dropout + BN

Input, $m = 3$
3×3 Convolution, $m = 48$
DB (4 layers) + TD, $m = 112$
DB (5 layers) + TD, $m = 192$
DB (7 layers) + TD, $m = 304$
DB (10 layers) + TD, $m = 464$
DB (12 layers) + TD, $m = 656$
DB (15 layers), $m = 896$
TU + DB (12 layers), $m = 1088$
TU + DB (10 layers), $m = 816$
TU + DB (7 layers), $m = 578$
TU + DB (5 layers), $m = 384$
TU + DB (4 layers), $m = 256$
1×1 Convolution, $m = c$
Softmax

«The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation»
[Jégou S. и др., 2017 https://arxiv.org/abs/1611.09326]

«Тирамису» = DenseNet + U-Net



– требует больших мощностей

V-Net

3D-сегментация медицинских снимков

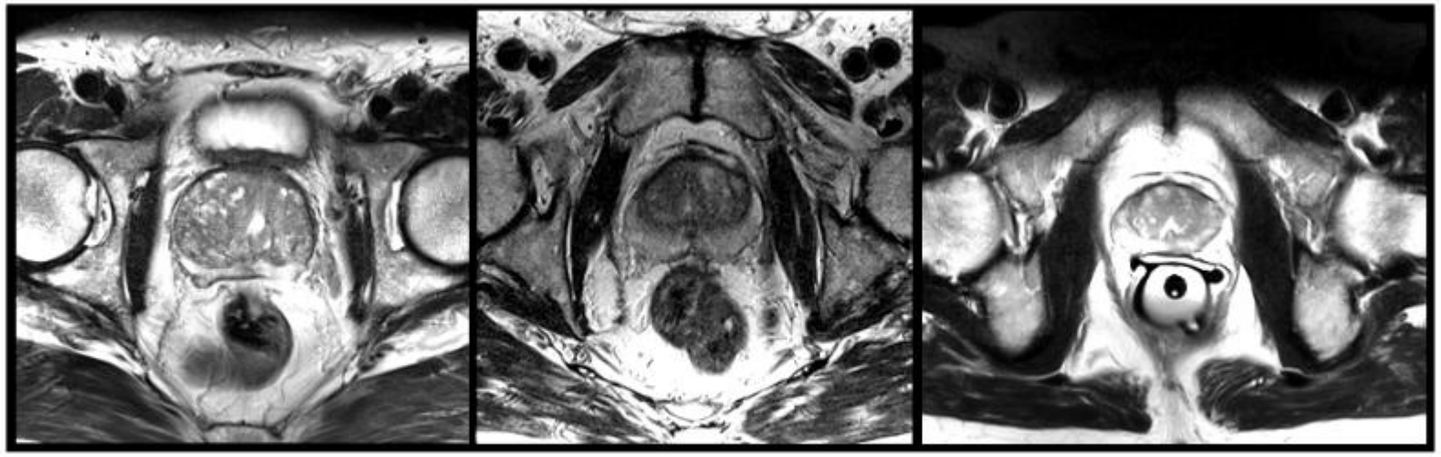


Fig. 1. Slices from MRI volumes depicting prostate. This data is part of the PROMISE2012 challenge dataset [7].

использование функции Дайса

$$D = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2}$$

$$\frac{\partial D}{\partial p_j} = 2\left[\frac{g_j\left(\sum_i^N p_i^2 + \sum_i^N g_i^2\right) - 2p_j\left(\sum_i^N p_i g_i\right)}{\left(\sum_i^N p_i^2 + \sum_i^N g_i^2\right)^2}\right]$$

Fausto Milletari et al. «V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation» // https://arxiv.org/abs/1606.04797

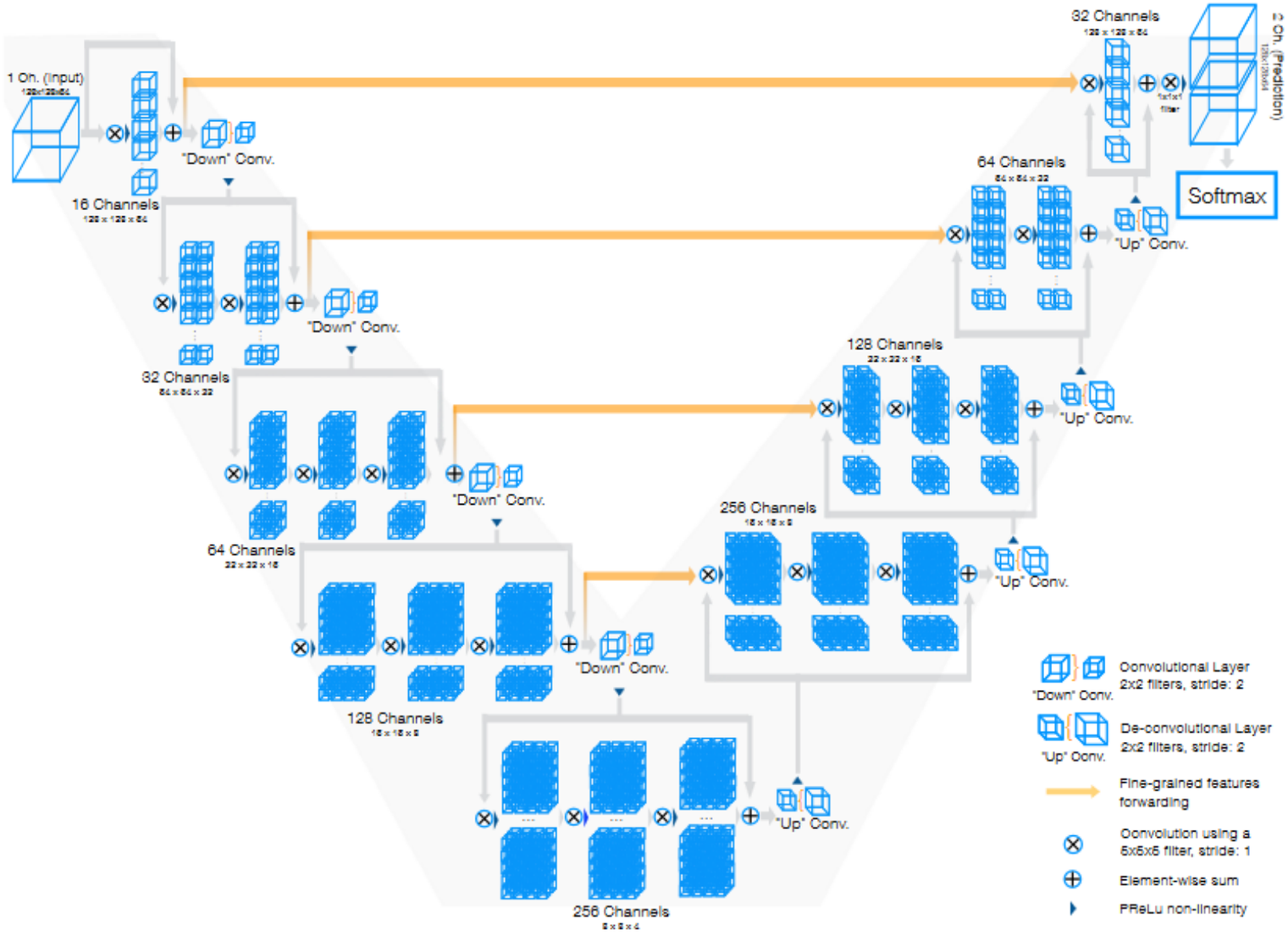


Fig. 2. Schematic representation of our network architecture. Our custom implementation of Caffe [5] processes 3D data by performing volumetric convolutions. Best viewed in electronic format.

TernausNet

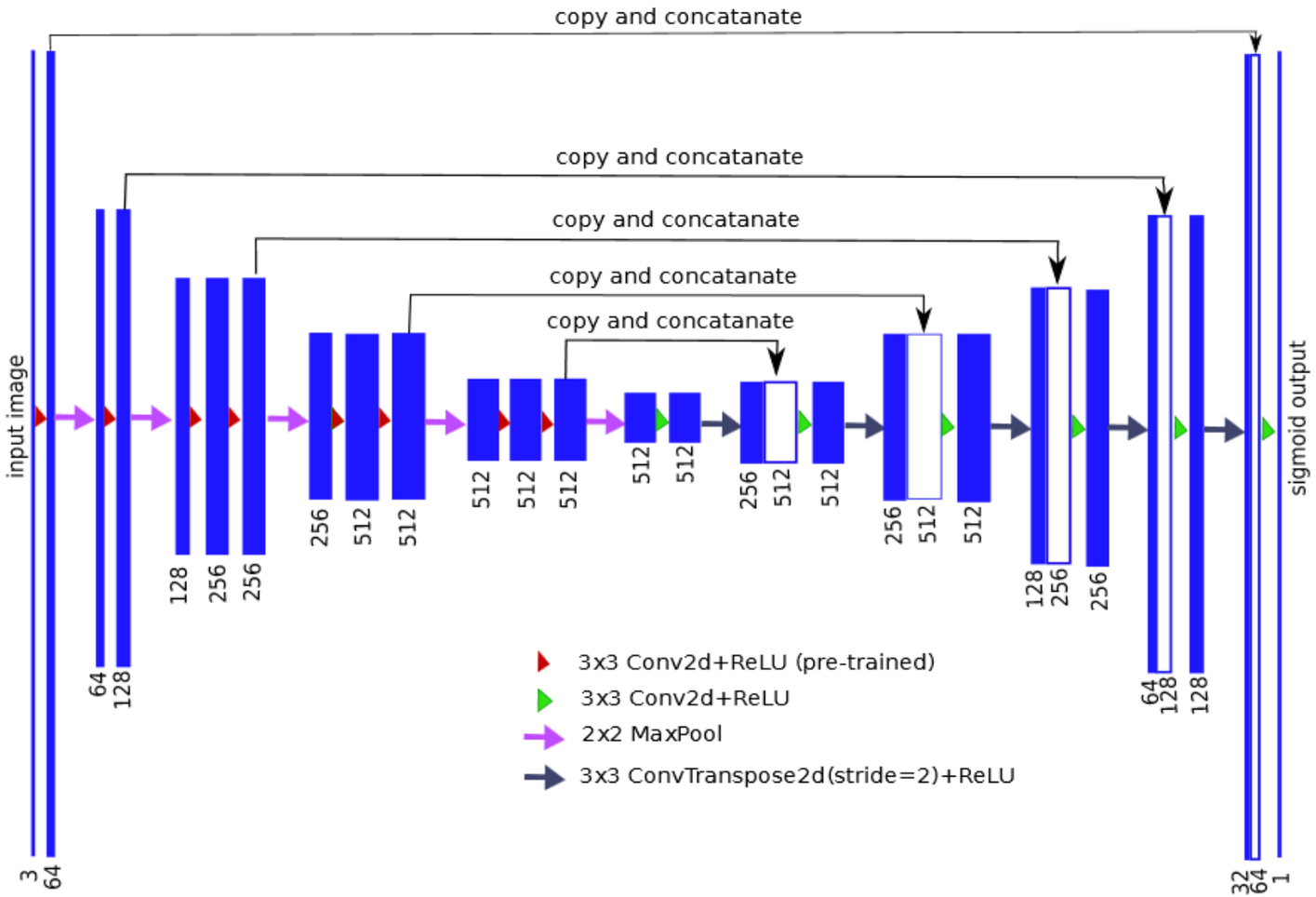


Fig. 1. Encoder-decoder neural network architecture also known as U-Net where VGG11 neural network without fully connected layers as its encoder. Each blue rectangular block represents a multi-channel features map passing through a series of transformations. The height of the rod shows a relative map size (in pixels), while their widths are proportional to the number of channels (the number is explicitly subscribed to the corresponding rod). The number of channels increases stage by stage on the left part while decrease stage by stage on the right decoding part. The arrows on top show transfer of information from each encoding layer and concatenating it to a corresponding decoding layer.

использование обученной VGG11 в качестве кодировщика https://arxiv.org/abs/1801.05746

Unet++

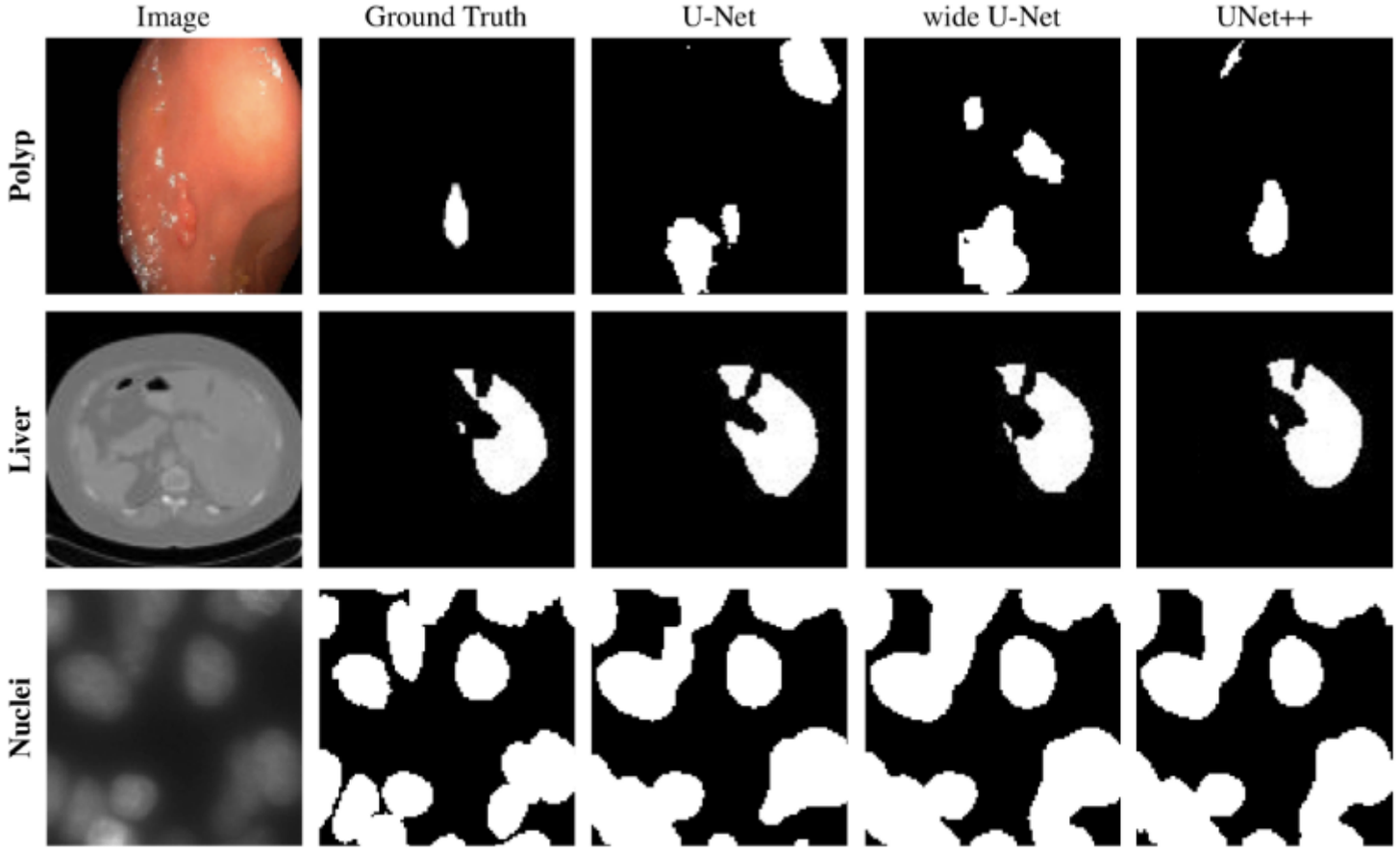


Fig. 2: Qualitative comparison between U-Net, wide U-Net, and UNet++, showing segmentation results for polyp, liver, and cell nuclei datasets (2D-only for a distinct visualization).

binary cross-entropy + dice

convolution layers on skip pathways (зелёным на сл. слайде)

dense skip connections on skip pathways (голубые)

having deep supervision (красные)

Zongwei Zhou et al. «UNet++: A Nested U-Net Architecture for Medical Image Segmentation» https://arxiv.org/abs/1807.10165

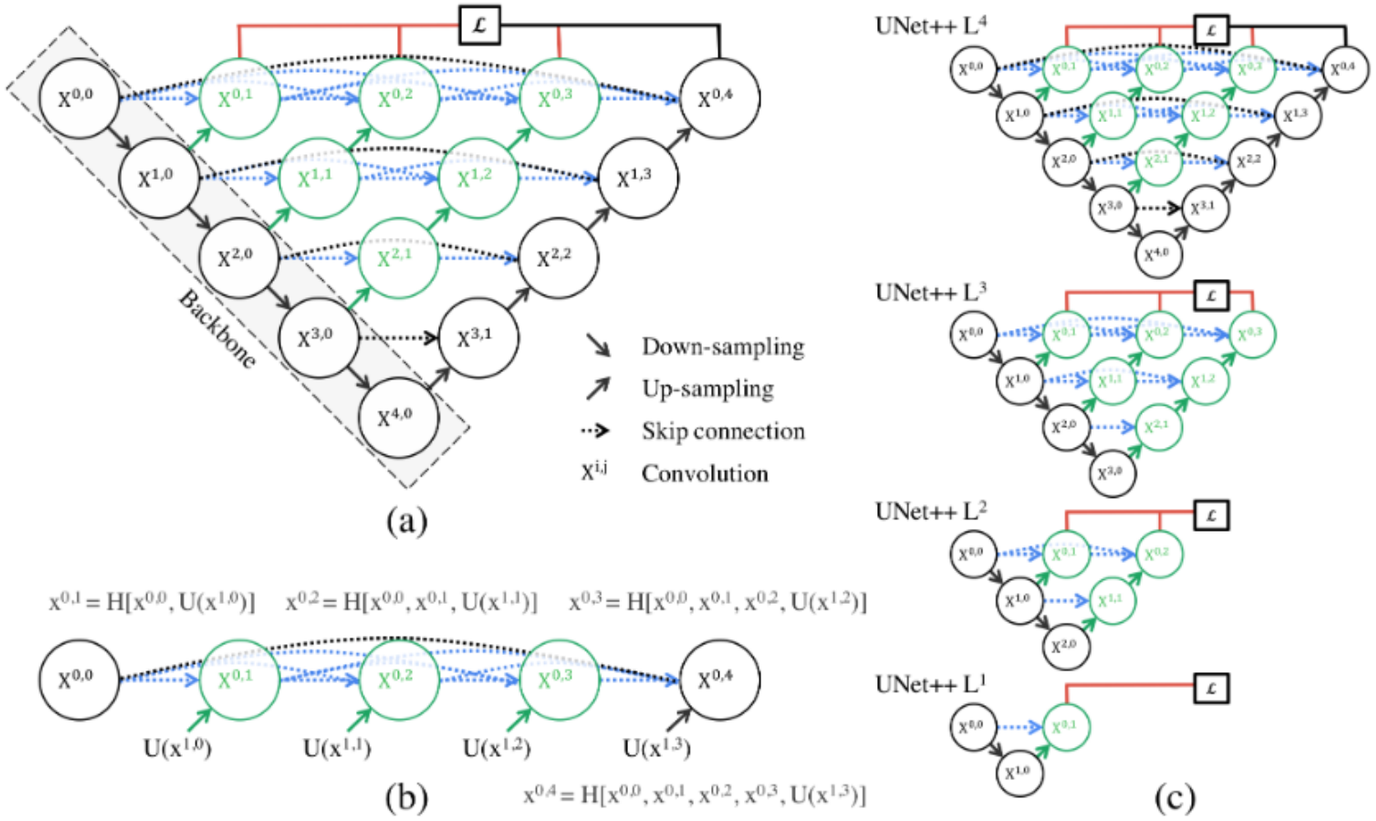


Fig. 1: (a) UNet++ consists of an encoder and decoder that are connected through a series of nested dense convolutional blocks. The main idea behind UNet++ is to bridge the semantic gap between the feature maps of the encoder and decoder prior to fusion. For example, the semantic gap between ($X^{0,0}$,$X^{1,3}$) is bridged using a dense convolution block with three convolution layers. In the graphical abstract, black indicates the original U-Net, green and blue show dense convolution blocks on the skip pathways, and red indicates deep supervision. Red, green, and blue components distinguish UNet++ from U-Net. (b) Detailed analysis of the first skip pathway of UNet++. (c) UNet++ can be pruned at inference time, if trained with deep supervision.

GridNet

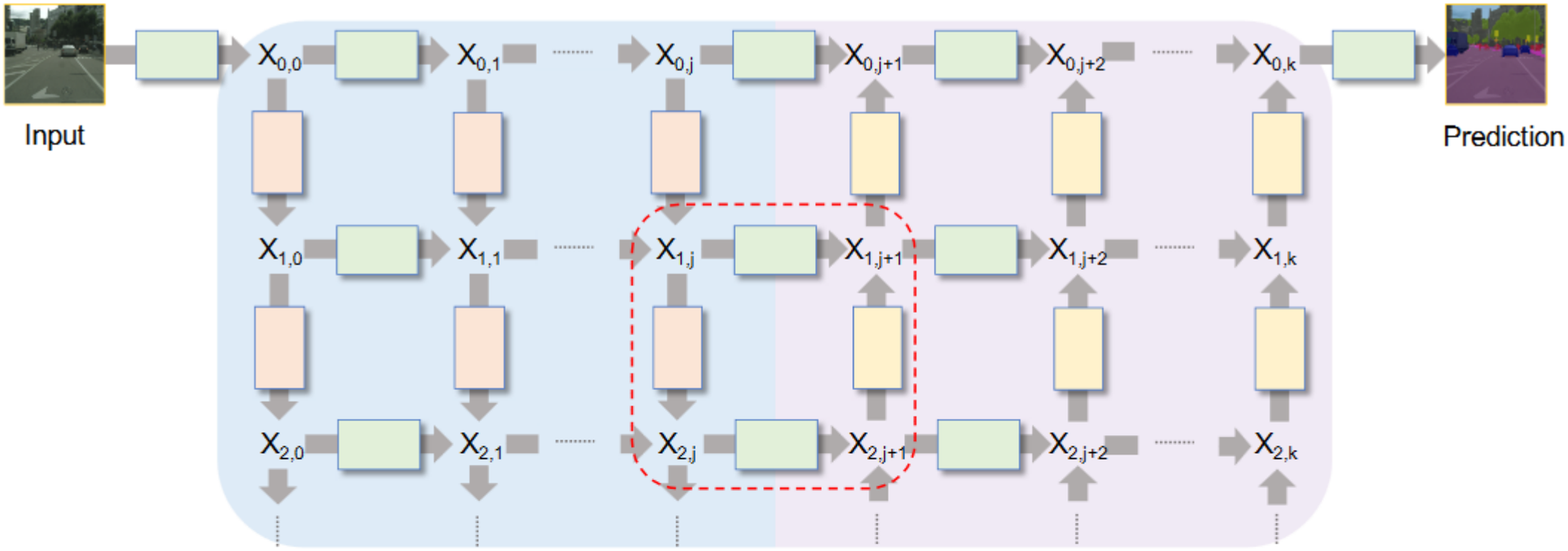


Figure 1: GridNet: each green unit is a residual bloc, which does not change the input map resolution nor the number of feature maps. Red blocks are convolutional units with resolution loss (subsampling) and twice the number of feature maps. Yellow units are deconvolutional blocks which increase the resolution (upsampling) and divide by two the number of feature maps. A zoom on the red square part with a detailed compositions of each blocks is shown in Figure 2

GridNet

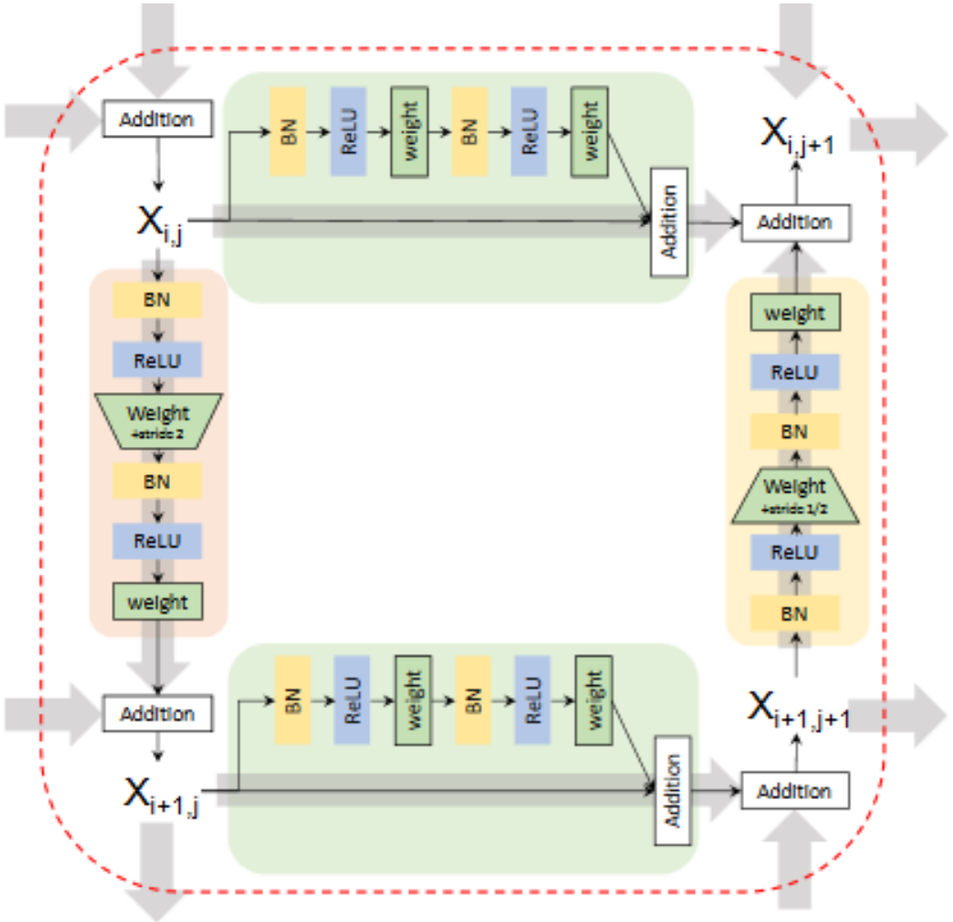


Figure 2: Detailed schema of a GridBlock. Green units are residual units keeping feature map dimensions constant between inputs and outputs. Red units are convolutional + subsampling and increase the feature dimensions. Yellow units are deconvolutional + upsampling and decrease the feature dimensions (back to the original one to allow the addition). Trapeziums illustrate the upsampling/subsampling operations obtained with strided convolutions. BN=Batch Normalization.

Damien Fourure «Residual Conv-Deconv Grid Network for Semantic Segmentation» // https://arxiv.org/abs/1707.07958

GridNet

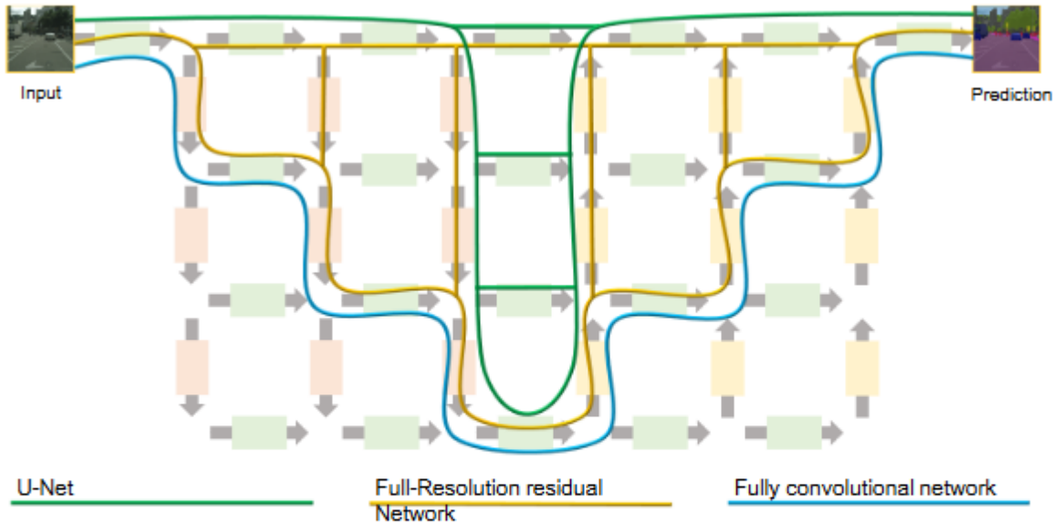


Figure 3: GridNets generalize several classical resolution preserving neural models such as conv-deconv networks [13] (blue connections), U-networks [18] (green connections) and Full Resolution Residual Networks (FRRN) [17] (yellow connections).

обобщает другие известные архитектуры

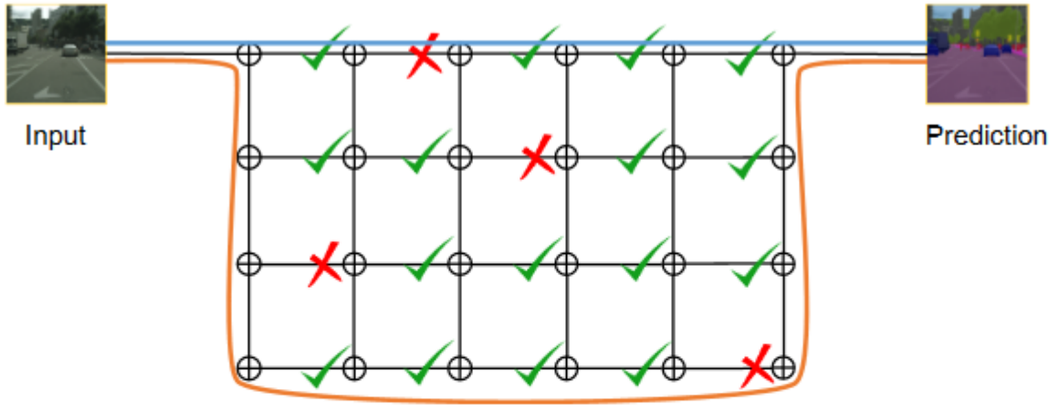


Figure 4: The blue path only using the high resolution stream is shorter than the orange path which also uses low resolution streams. To force the network to use all streams we randomly drop streams during training, indicated by red crosses.

можно так сделать DropOut

Сегментация – ключевые идеи

1) архитектура, которая получает на выходе маски

пример: полностью свёрточная

проблема высокого разрешения –
специальные архитектуры

2) учёт глобального контекста
– сбор информации с разных масштабов

Ключевые идеи: почему нужно учитывать разные масштабы

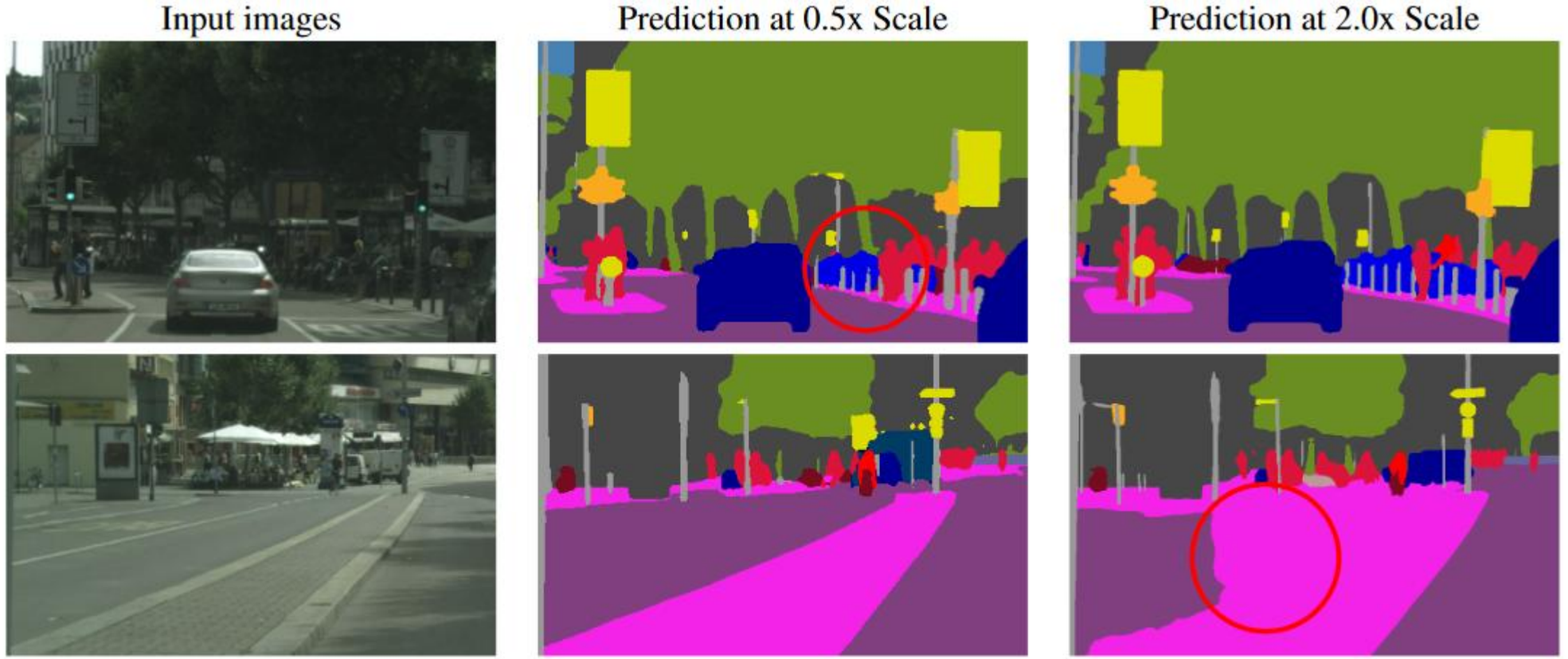


Figure 1: Illustration of common failures modes for semantic segmentation as they relate to inference scale. In the first row, the thin posts are inconsistently segmented in the scaled down (0.5x) image, but better predicted in the scaled-up (2.0x) image. In the second row, the large road / divider region is better segmented at lower resolution (0.5x).

https://arxiv.org/pdf/2005.10821v1.pdf

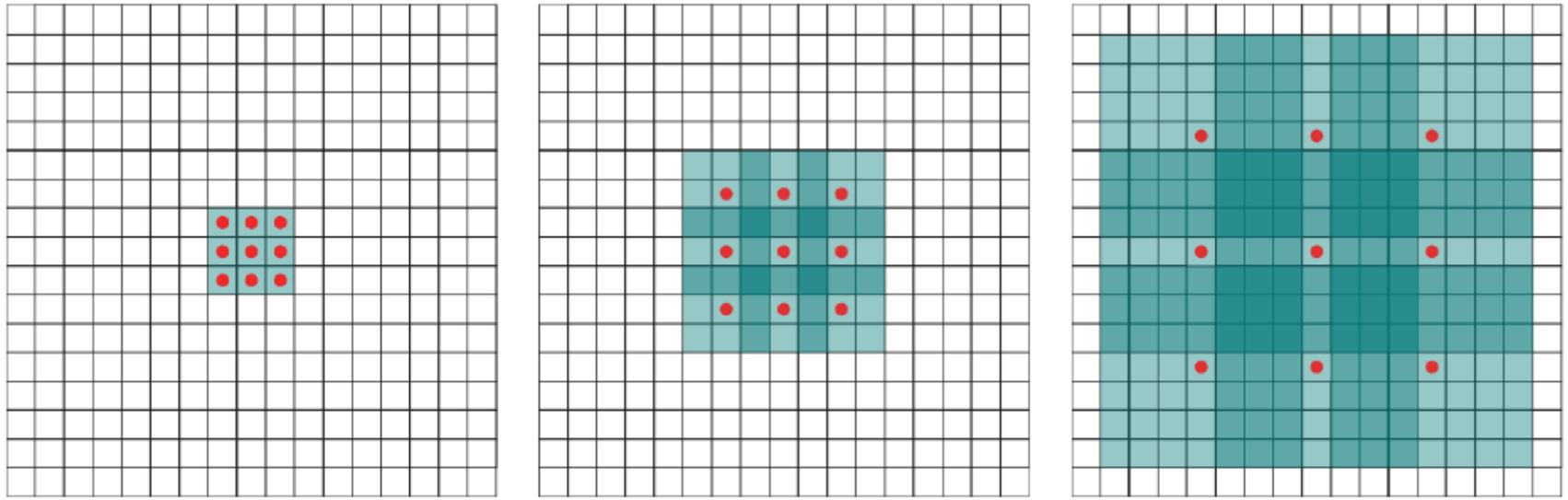
Расширенные свёртки (Dilated convolutions / Atrous Convolutions)

Discrete Convolution Operation:

$$(F * k)(\mathbf{p}) = \sum_{\mathbf{s}+\mathbf{t}=\mathbf{p}} F(\mathbf{s})\, k(\mathbf{t})$$

Dilated Convolution Operation:

$$(F *_l k)(\mathbf{p}) = \sum_{\mathbf{s}+l\mathbf{t}=\mathbf{p}} F(\mathbf{s})\, k(\mathbf{t})$$



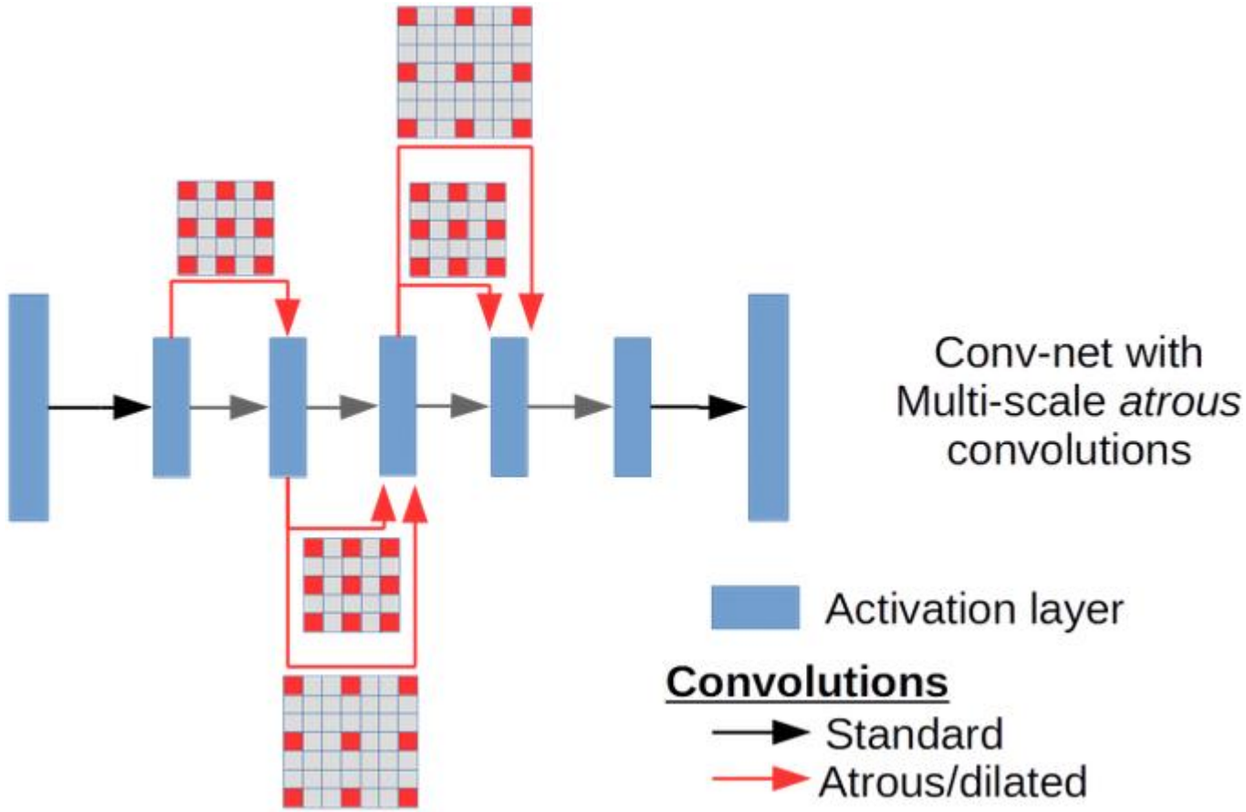
(a) (b) (c)

увеличение перцептивного поля (perception field)

https://arxiv.org/abs/1511.07122

Расширенные свёртки (Dilated convolutions / Atrous Convolutions)

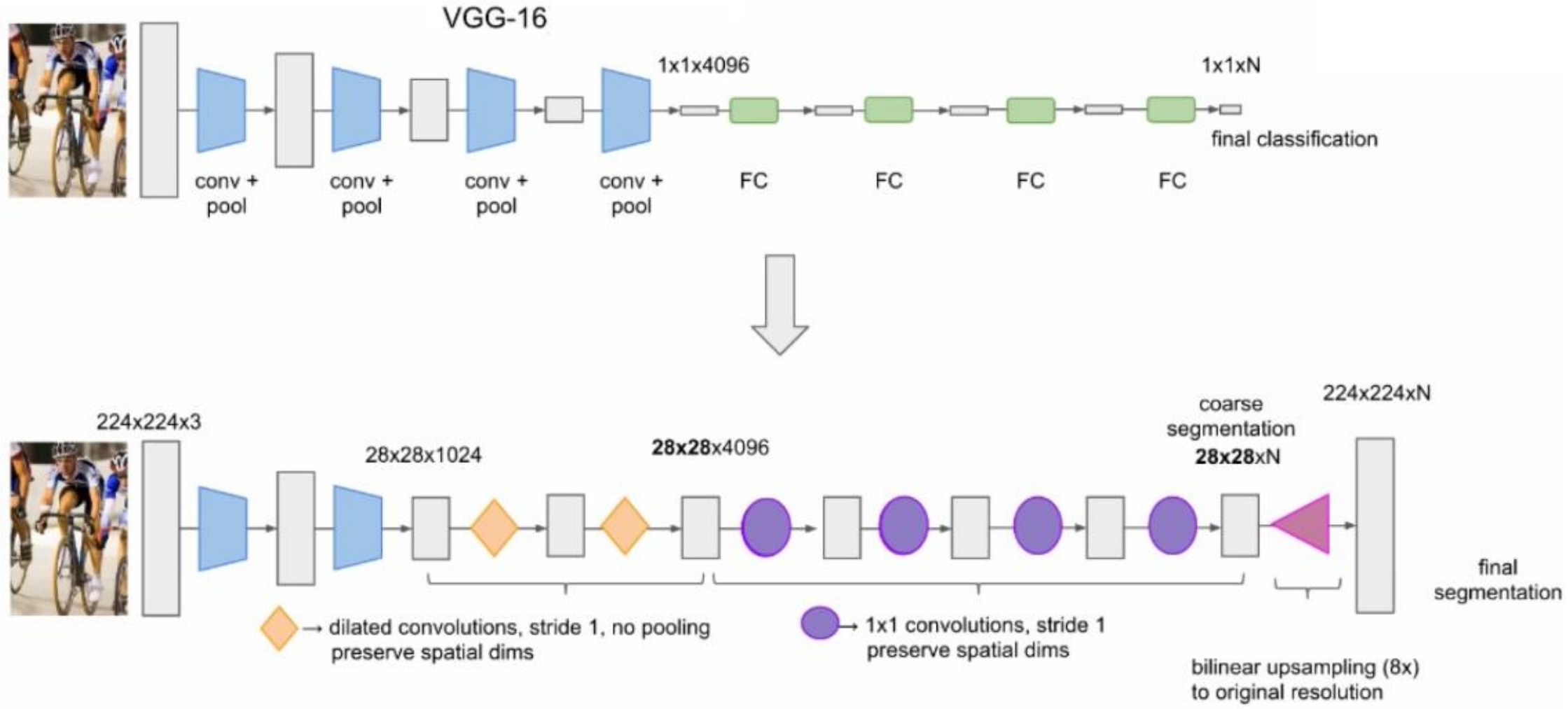
можно комбинировать признаки с разных масштабов с помощью расширенных свёрток



также в задачах, например увеличения разрешения

https://dzone.com/articles/atrous-convolutions-amp-u-net-architectures-for-de

DeepLabv1/2: использование расширенных свёрток



нет пулинга – последние свёртки и пулинги заменили на расширенные свёртки (stride=1)

постпроцессинг – CRF

DeepLabv1

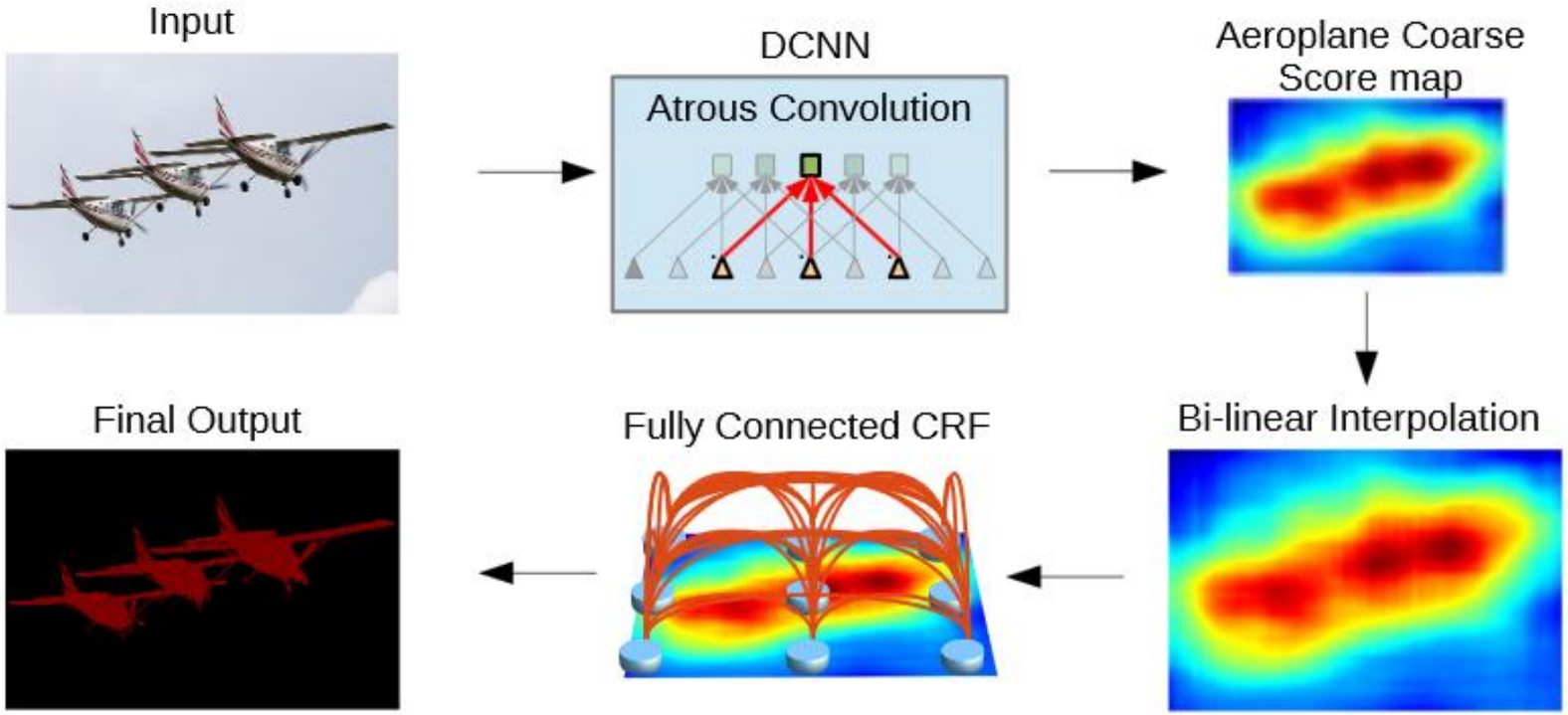


Fig. 1: Model Illustration. A Deep Convolutional Neural Network such as VGG-16 or ResNet-101 is employed in a fully convolutional fashion, using atrous convolution to reduce the degree of signal downsampling (from 32x down 8x). A bilinear interpolation stage enlarges the feature maps to the original image resolution. A fully connected CRF is then applied to refine the segmentation result and better capture the object boundaries.

Liang-Chieh Chen «DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs» // https://arxiv.org/pdf/1606.00915.pdf

DeepLabv1

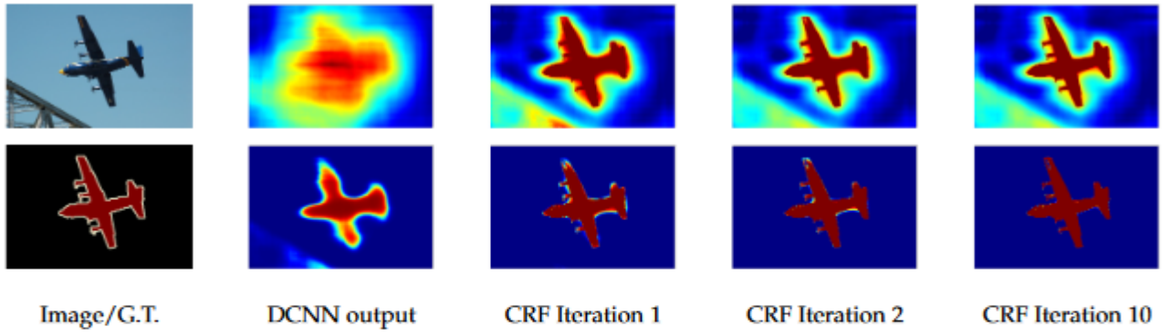


Fig. 5: Score map (input before softmax function) and belief map (output of softmax function) for Aeroplane. We show the score (1st row) and belief (2nd row) maps after each mean field iteration. The output of last DCNN layer is used as input to the mean field inference.

$$E(\boldsymbol{x}) = \sum_i \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j)$$

$$\theta_i(x_i) = -\log P(x_i)$$

по выходу нейронки

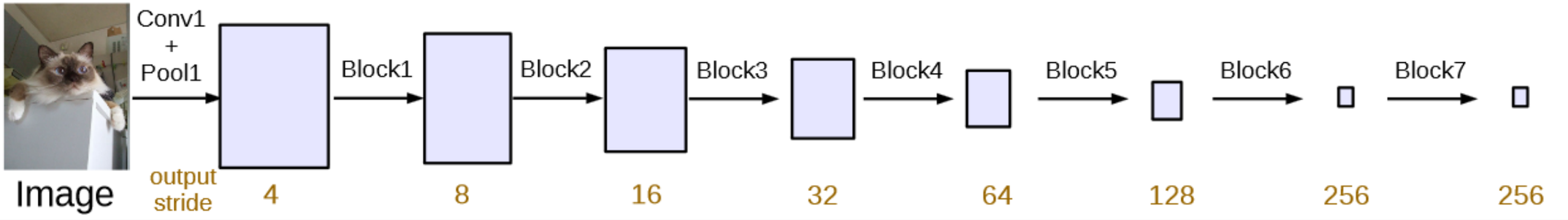
$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j) \left[w_1 \exp\left(-\frac{||p_i - p_j||^2}{2\sigma_\alpha^2} - \frac{||I_i - I_j||^2}{2\sigma_\beta^2} \right) + w_2 \exp\left(-\frac{||p_i - p_j||^2}{2\sigma_\gamma^2} \right) \right]$$

$$\mu(x_i, x_j) = 1 \text{ if } x_i \neq x_j$$

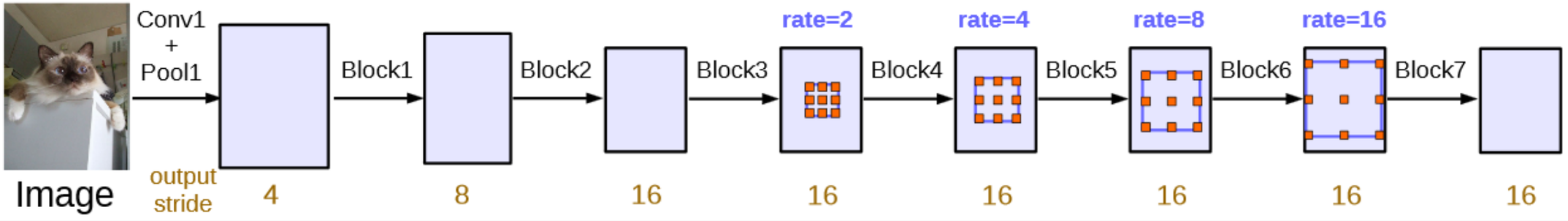
p – позиция

I – RGB

DeepLabv3: использование расширенных свёрток



(a) Going deeper without atrous convolution.



(b) Going deeper with atrous convolution. Atrous convolution with $rate > 1$ is applied after block3 when $output\_stride = 16$.

Figure 3. Cascaded modules without and with atrous convolution.

можно комбинировать признаки с разных масштабов: пространственные размеры остаются, а информация собирается более глобально

output stride = input image spatial resolution / final output resolution

Liang-Chieh Chen, George Papandreou, Florian Schroff, Hartwig Adam «Rethinking Atrous Convolution for Semantic Image Segmentation» // https://arxiv.org/pdf/1706.05587.pdf

DeepLabv3: использование расширенных свёрток

«Параллельный режим»

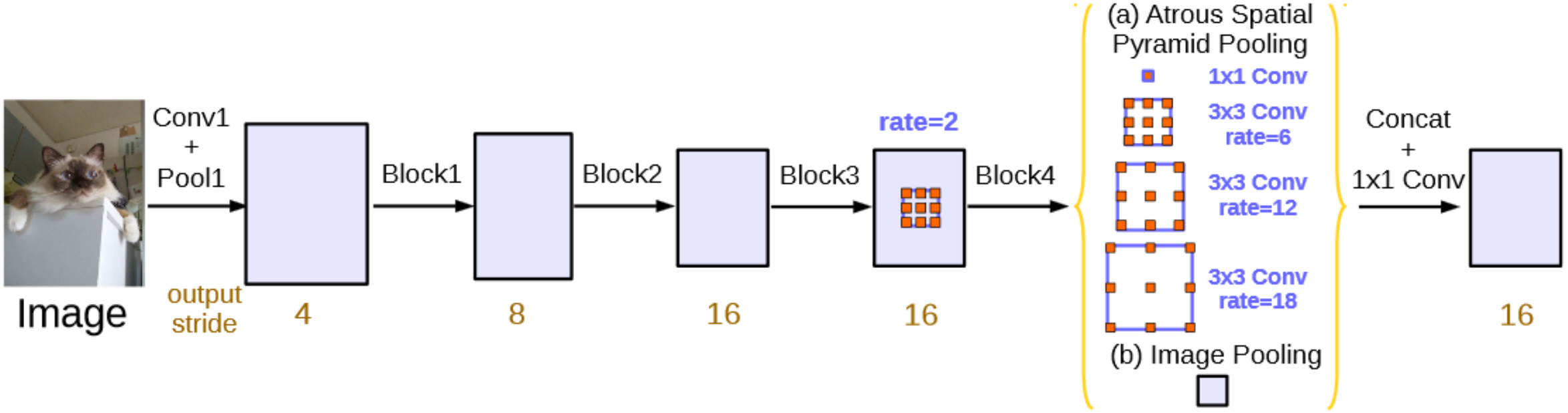
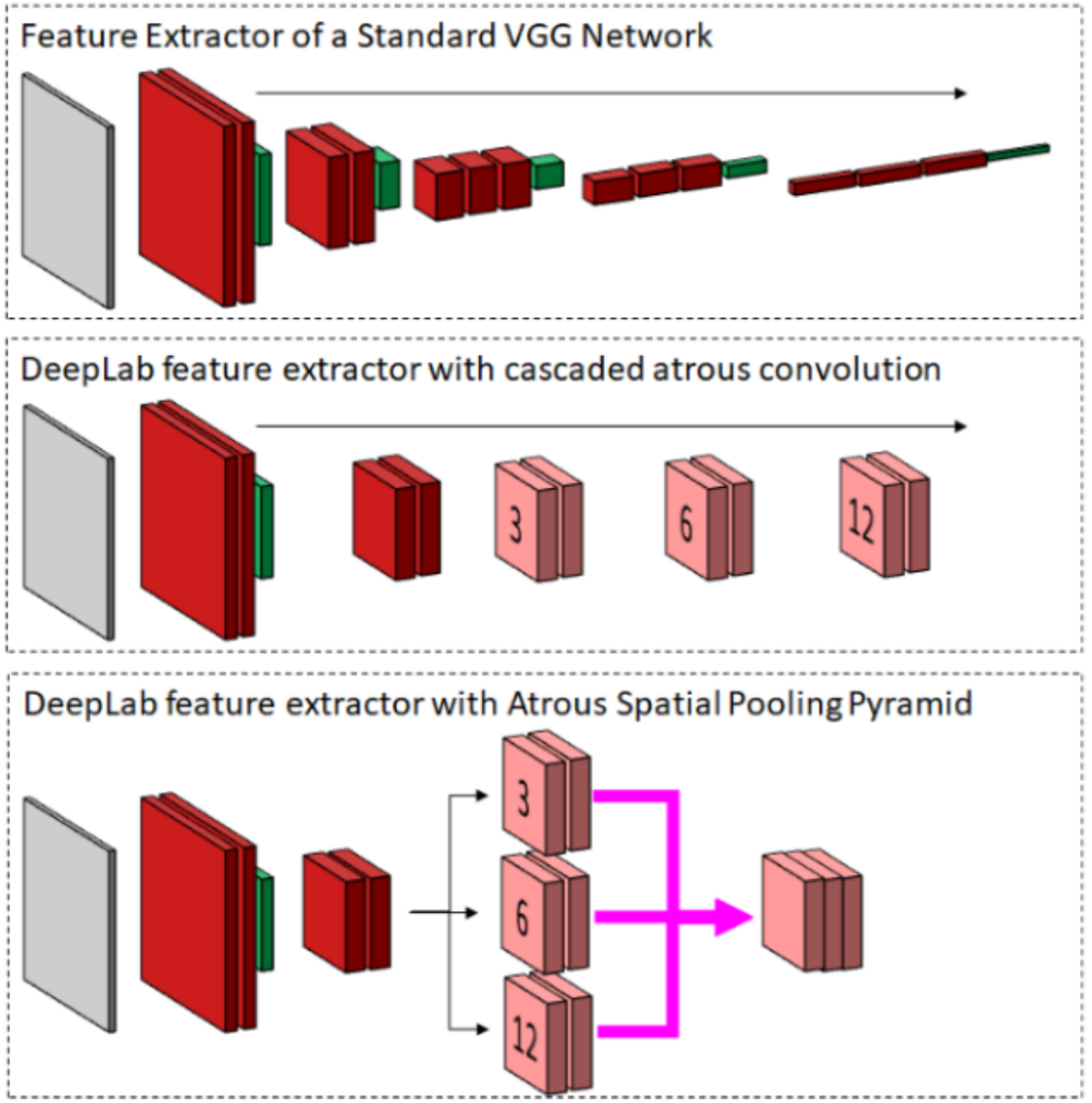


Figure 5. Parallel modules with atrous convolution (ASPP), augmented with image-level features.

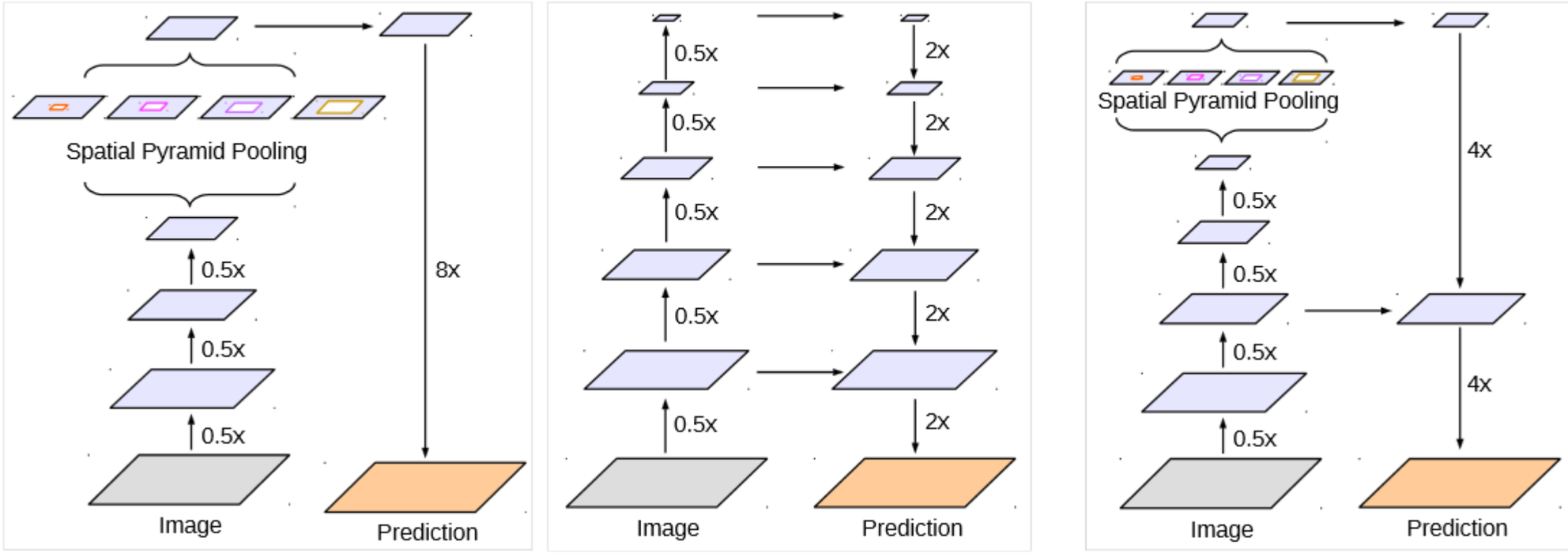
ASPP = Atrous Spatial Pyramid Pooling

https://arxiv.org/pdf/1706.05587.pdf

DeepLabv 1–3


Feature Extractor of a Standard VGG Network
DeepLab feature extractor with cascaded atrous convolution
3
6
12
DeepLab feature extractor with Atrous Spatial Pooling Pyramid
3
6
12

DeepLabv3+



(a) Spatial Pyramid Pooling (b) Encoder-Decoder (c) Encoder-Decoder with Atrous Conv

Fig. 1. We improve DeepLabv3, which employs the spatial pyramid pooling module (a), with the encoder-decoder structure (b). The proposed model, DeepLabv3+, contains rich semantic information from the encoder module, while the detailed object boundaries are recovered by the simple yet effective decoder module. The encoder module allows us to extract features at an arbitrary resolution by applying atrous convolution.

DeepLabv3+

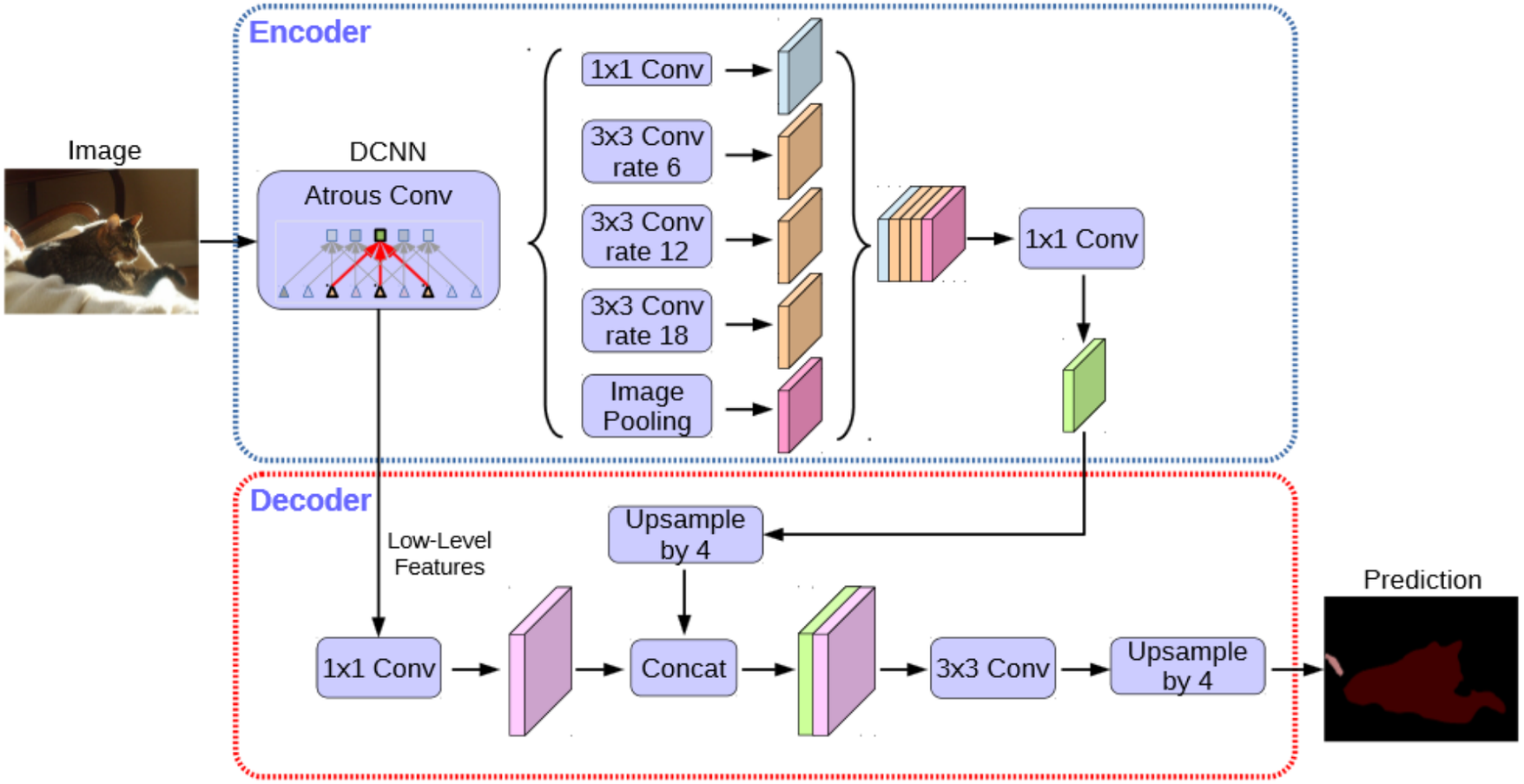
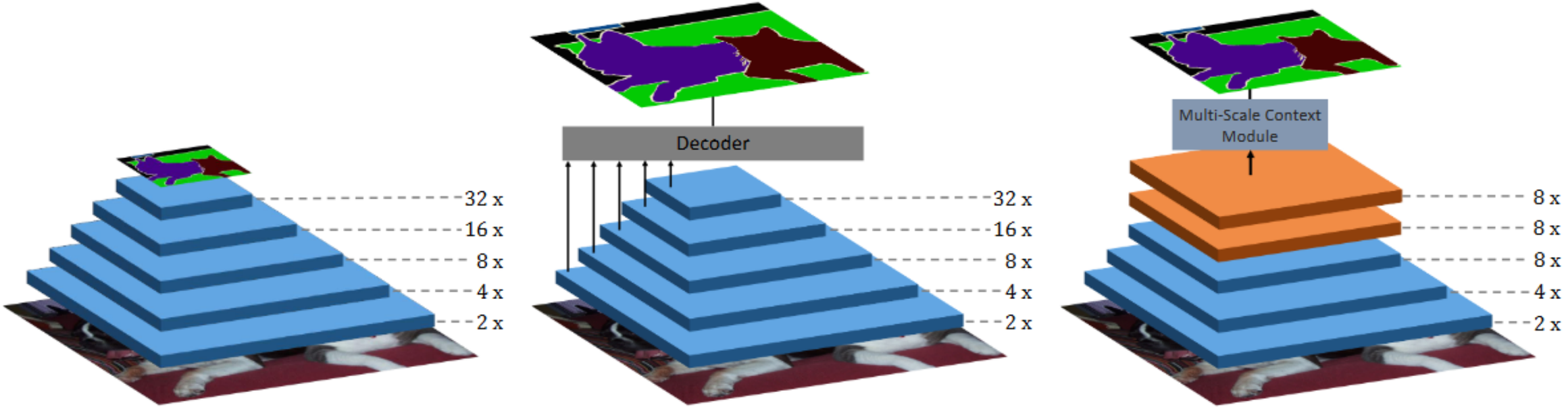


Fig. 2. Our proposed DeepLabv3+ extends DeepLabv3 by employing a encoder-decoder structure. The encoder module encodes multi-scale contextual information by applying atrous convolution at multiple scales, while the simple yet effective decoder module refines the segmentation results along object boundaries.

https://arxiv.org/pdf/1802.02611.pdf

FastFCN

Эволюция подходов:



(a) FCN [22] (b) EncoderDecoder (c) DilatedFCN

Figure 1: **Different types of networks for semantic segmentation.** (a) is the original FCN, (b) follows the encoder-decoder style, and (c) employs dilated convolutions to obtain high-resolution final feature maps. Best viewed in color.

Huikai Wu «FastFCN: Rethinking Dilated Convolution in the Backbone for Semantic Segmentation» // https://arxiv.org/abs/1903.11816

FastFCN

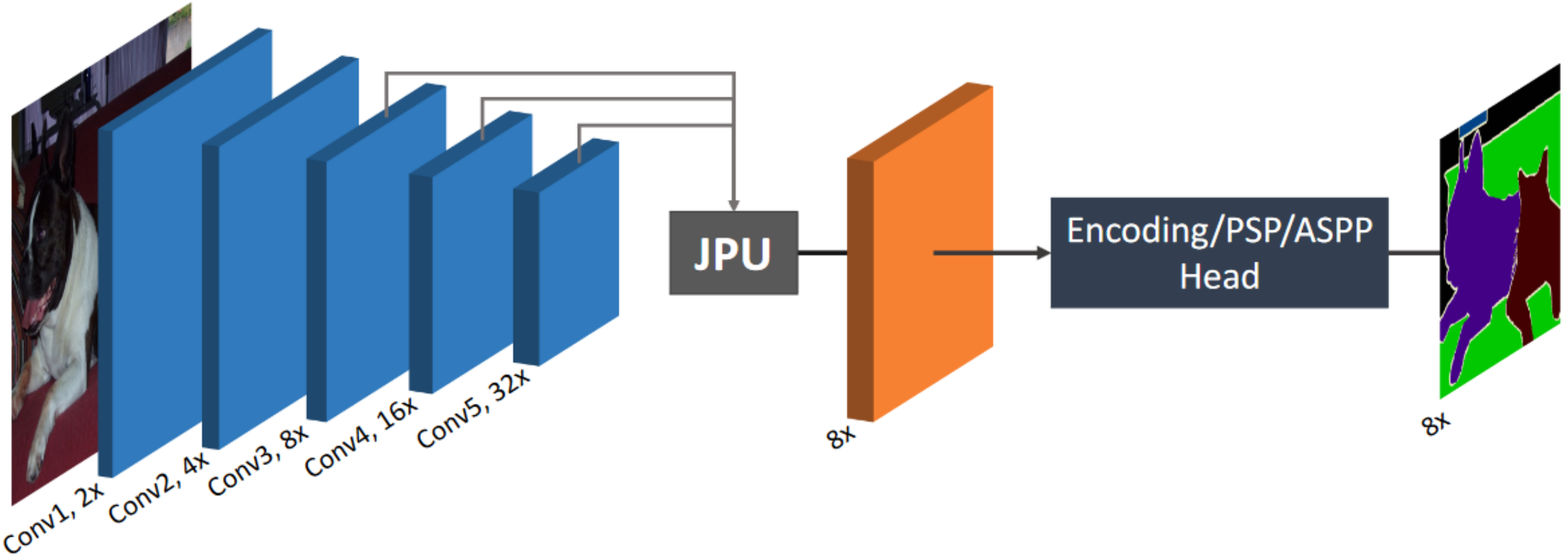


Figure 2: **Framework Overview of Our Method.** Our method employs the same backbone as the original FCN. After the backbone, a novel upsampling module named Joint Pyramid Upsampling (JPU) is proposed, which takes the last three feature maps as the inputs and generates a high-resolution feature map. A multi-scale/global context module is then employed to produce the final label map. Best viewed in color.

FastFCN: Joint Pyramid Upsampling

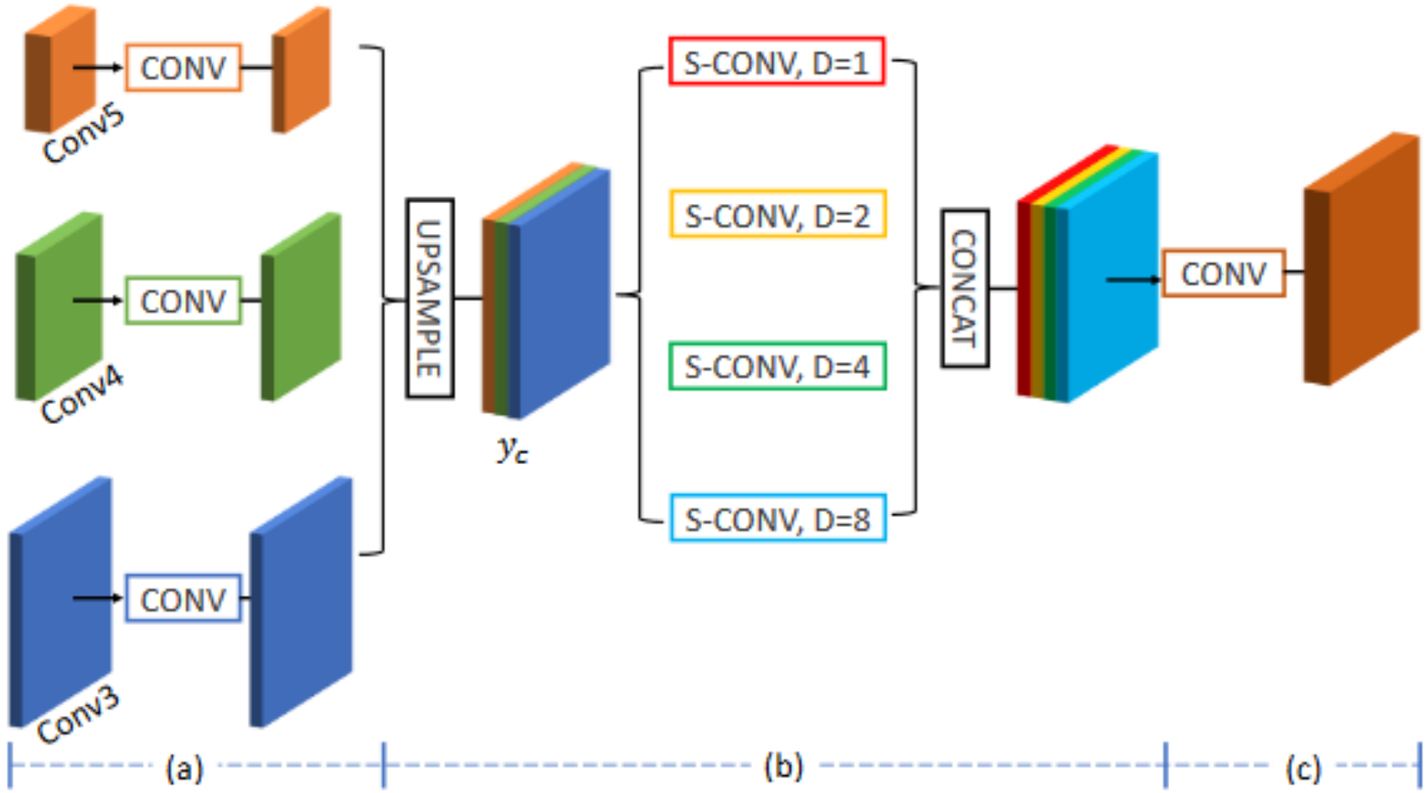
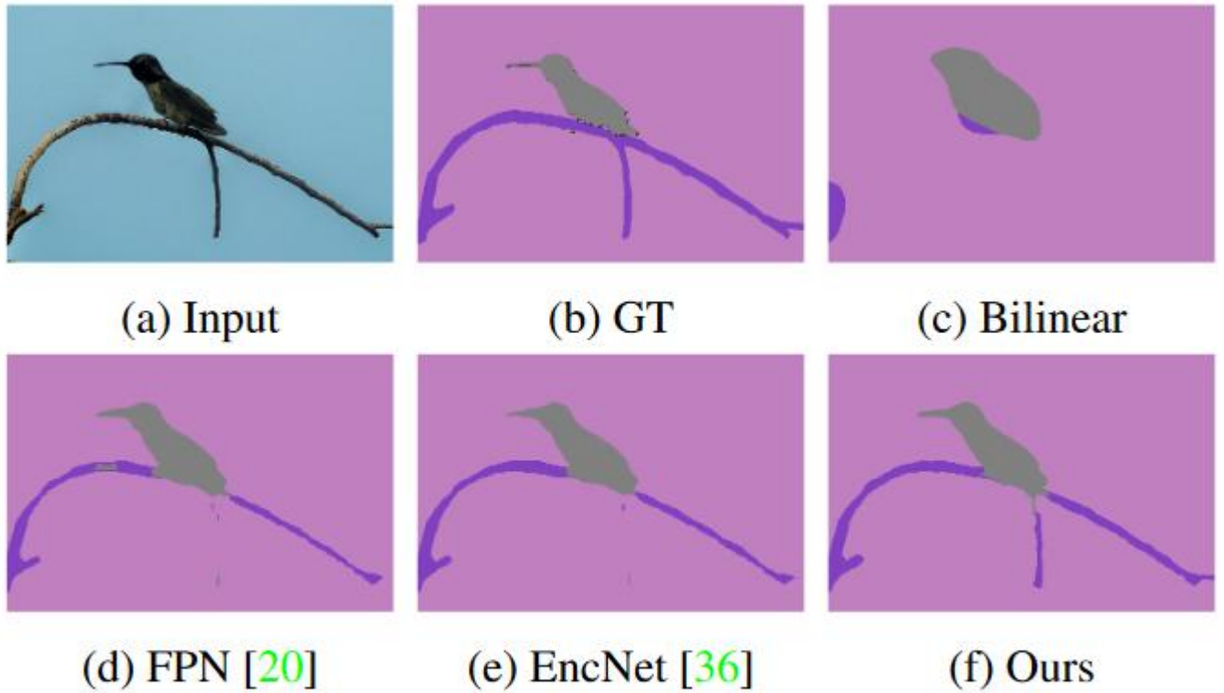


Figure 4: **The Proposed Joint Pyramid Upsampling (JPU).** Best viewed in color.

Сначала свёртки

Потом 4 separable convolutions с разными dilation rates (1, 2, 4, 8)

FastFCN: разные стратегии масштабирования (Upsampling)

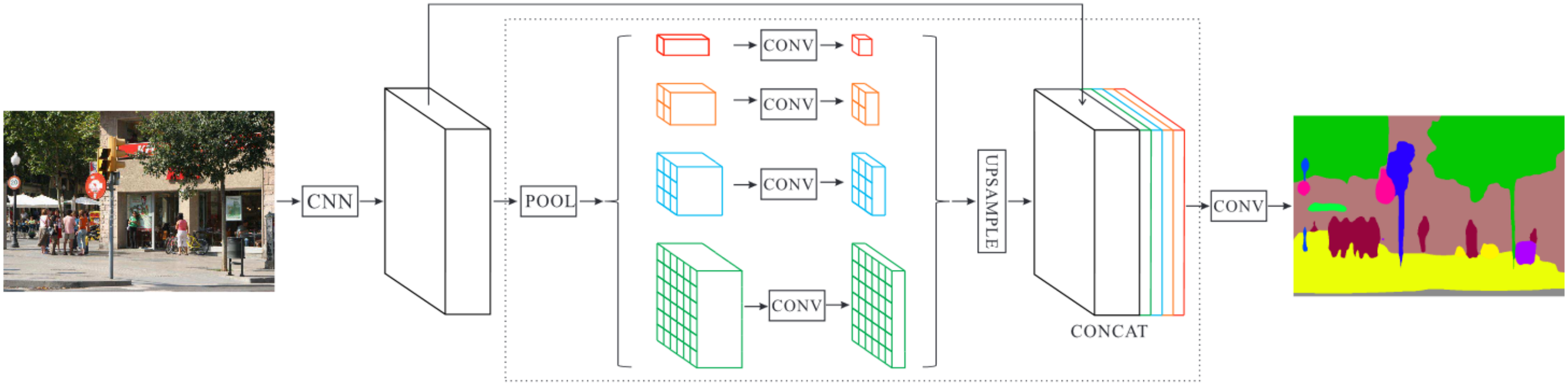


(a) Input (b) GT (c) Bilinear

(d) FPN [20] (e) EncNet [36] (f) Ours

Figure 6: Visual comparison of different upsampling modules with Encoding Head and ResNet-50 as the backbone.

PSP-Net = Pyramid Scene Parsing Network



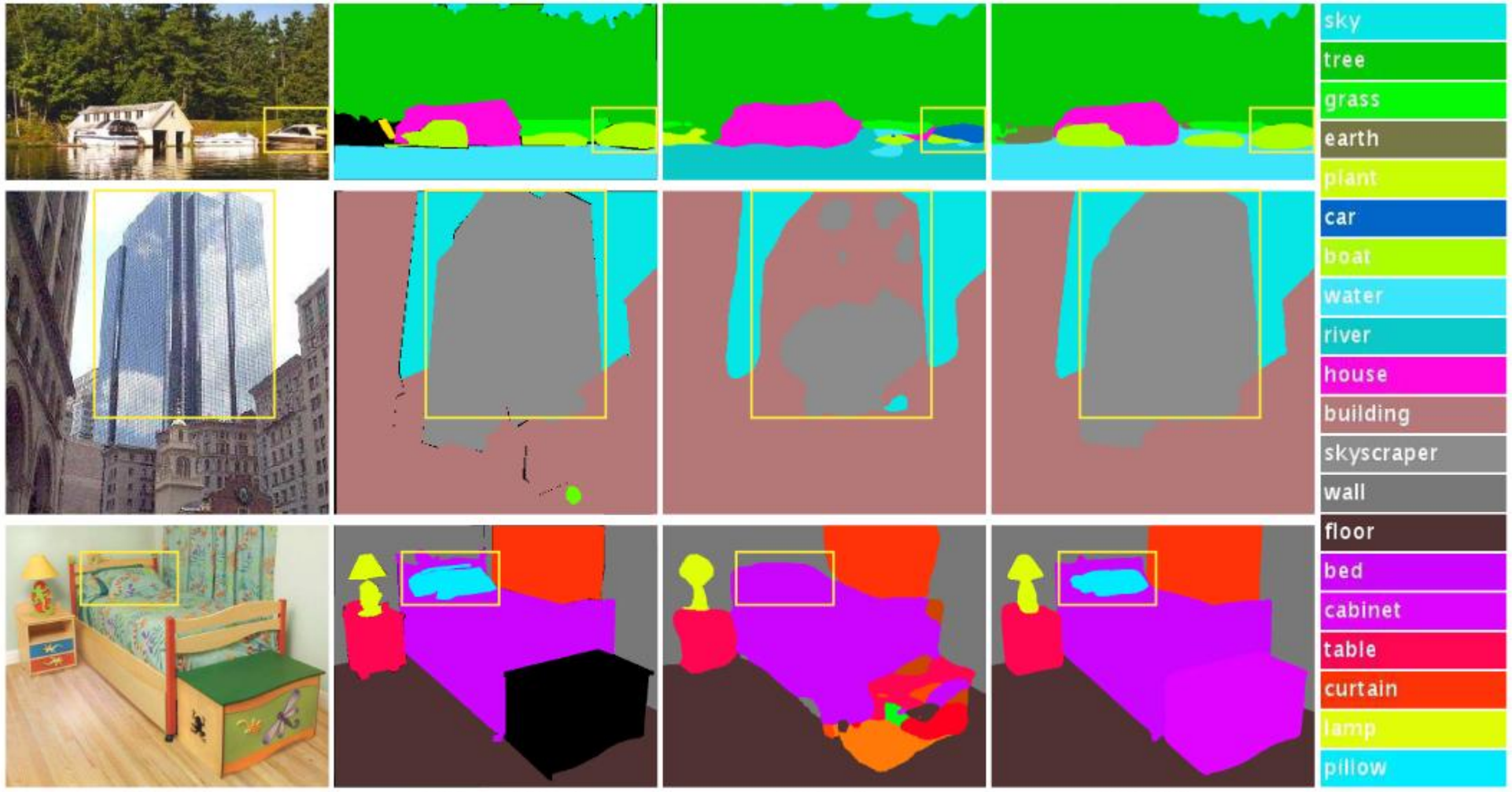
(a) Input Image (b) Feature Map (c) Pyramid Pooling Module (d) Final Prediction

Figure 3. Overview of our proposed PSPNet. Given an input image (a), we first use CNN to get the feature map of the last convolutional layer (b), then a pyramid parsing module is applied to harvest different sub-region representations, followed by upsampling and concatenation layers to form the final feature representation, which carries both local and global context information in (c). Finally, the representation is fed into a convolution layer to get the final per-pixel prediction (d).

идея сбора информации с разных масштабов

Hengshuang Zhao «Pyramid Scene Parsing Network» // https://arxiv.org/abs/1612.01105

PSP-Net

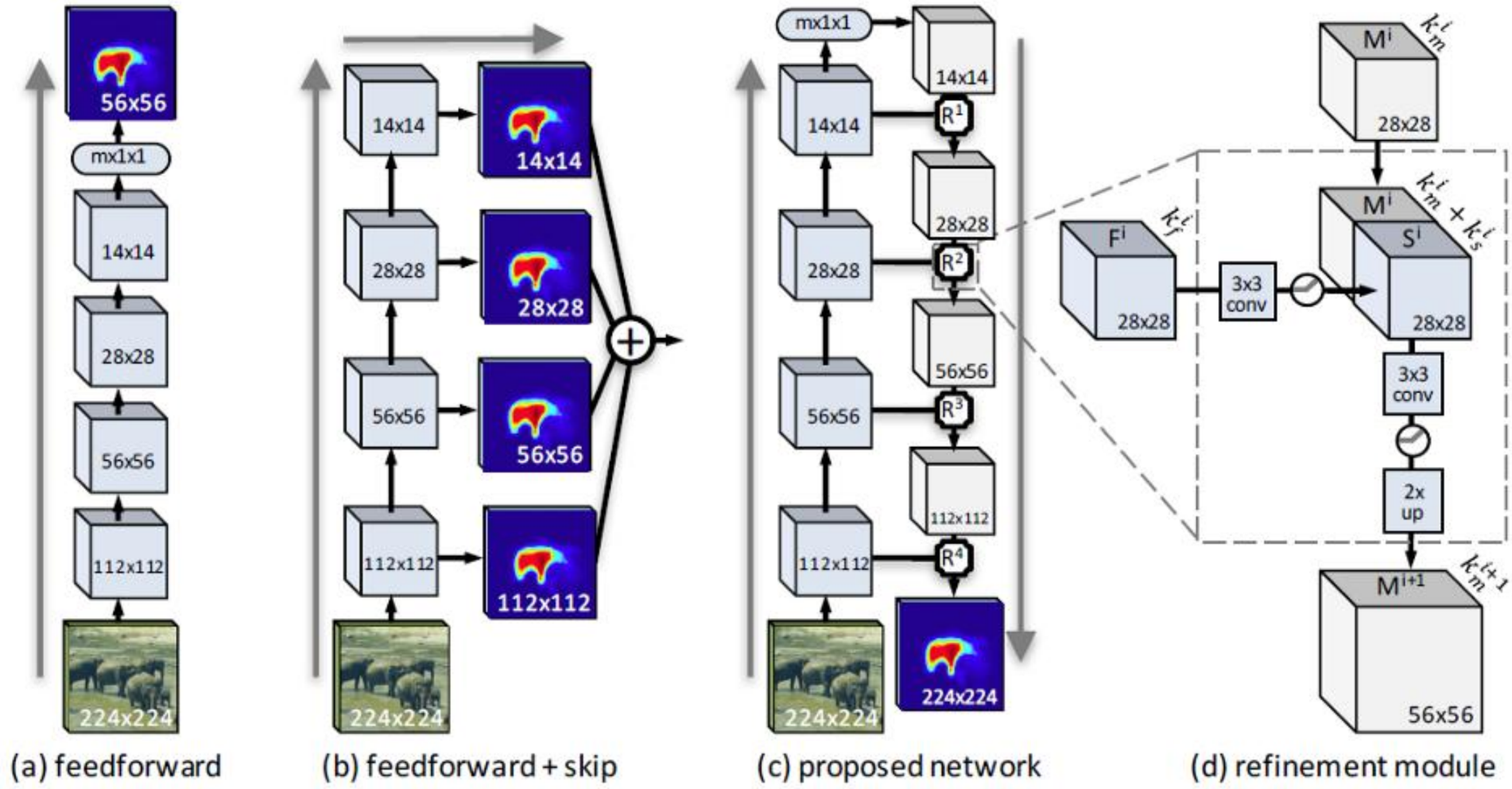


(a) Image (b) Ground Truth (c) FCN (d) PSPNet (e) ColorMap

Figure 2. Scene parsing issues we observe on ADE20K [43] dataset. The first row shows the issue of mismatched relationship – cars are seldom over water than boats. The second row shows confusion categories where class "building" is easily confused as "skyscraper". The third row illustrates inconspicuous classes. In this example, the pillow is very similar to the bed sheet in terms of color and texture. These inconspicuous objects are easily misclassified by FCN.

SharpMask: Top to Down Refinement

encoder-decoder



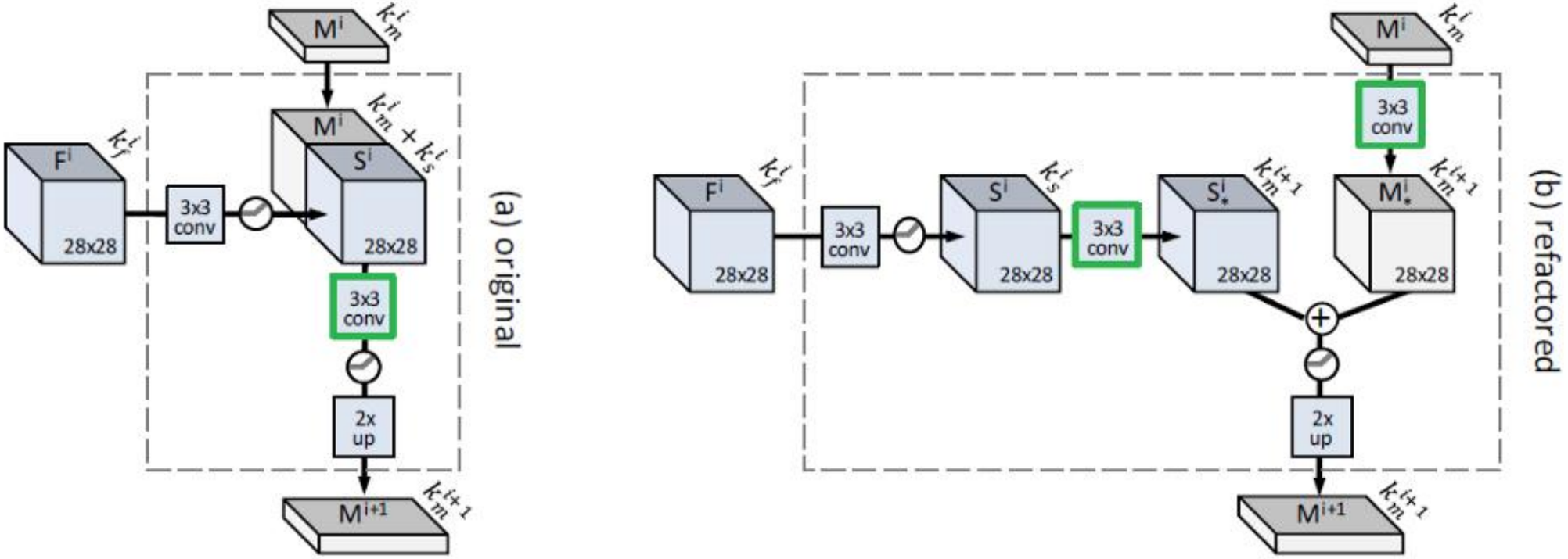
Learning to refine object segments, ECCV 2016

https://towardsdatascience.com/review-sharpmask-instance-segmentation-6509f7401a61

SharpMask: Top to Down Refinement

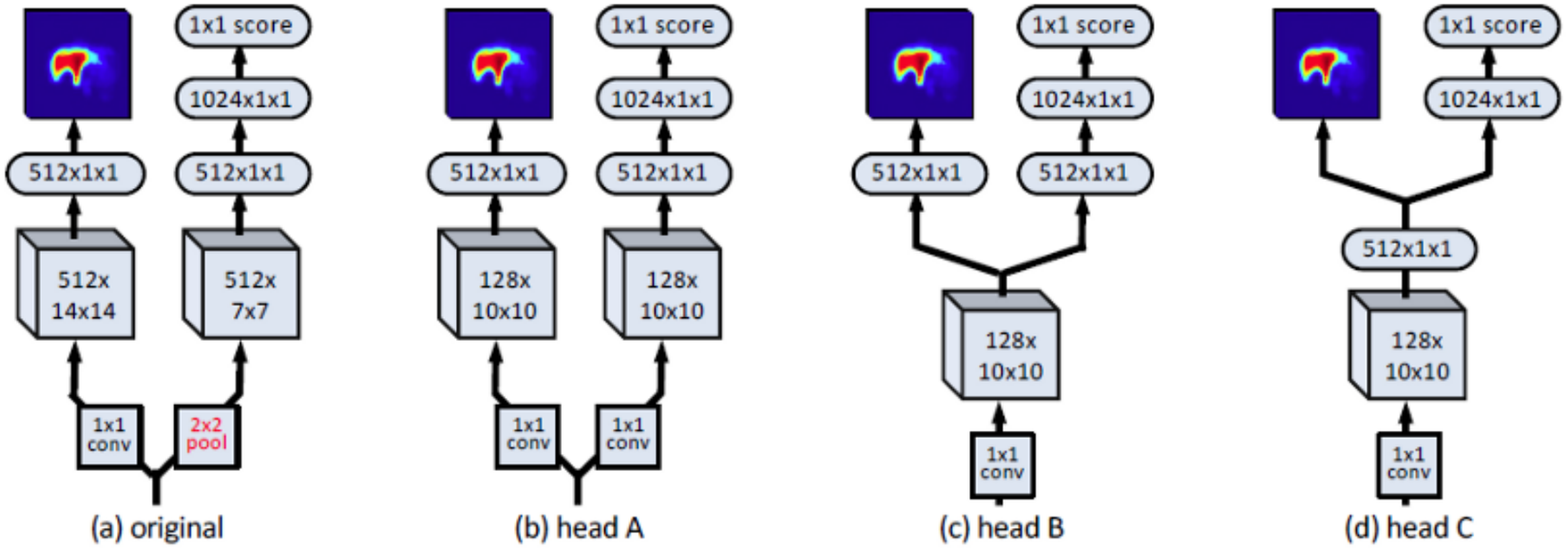
На проходе вверх: **ImageNet-Pretrained 50-layer ResNet**

На проходе вниз (top-down): **3×3 свёртки + 2× upsampled (билинейная интерполяция)**
Конкатенация, перед которой также 3×3 свёртки (+ сокращение числа каналов)



справа – более эффективная реализация

SharpMask: Head Architecture



Пробовали по-разному

Обучение

есть тонкости

сначала – проход вверх, чтобы получать грубую маску

потом он замораживается…

Работа – можно уточнять (refined) только «многообещающие» области

SharpMask: примеры работы

«High-Resolution Representations»

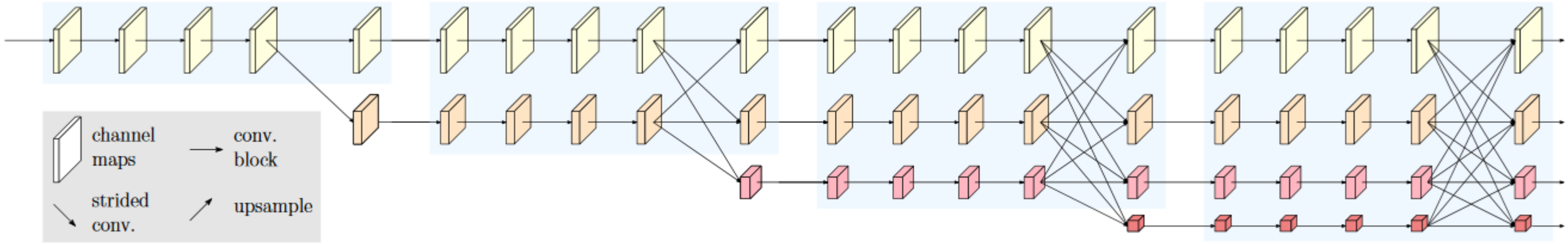


Figure 1. A simple example of a high-resolution network. There are four stages. The 1st stage consists of high-resolution convolutions. The 2nd (3rd, 4th) stage repeats two-resolution (three-resolution, four-resolution) blocks. The detail is given in Section 3.

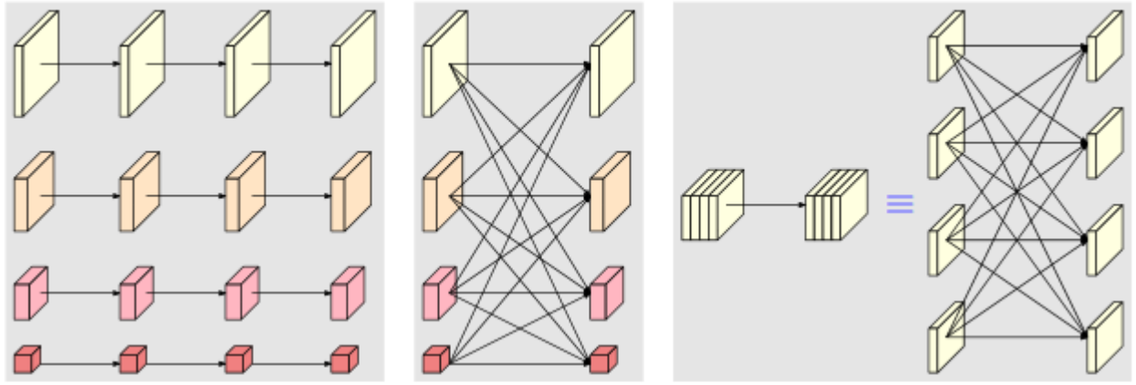


Figure 2. Multi-resolution block: (a) multi-resolution group convolution and (b) multi-resolution convolution. (c) A normal convolution (left) is equivalent to fully-connected multi-branch convolutions (right).

примеры архитектур для задач типа сегментации, детекции, оценки позы и т.п.

использование информации на разных разрешениях вместе

«High-Resolution Representations»

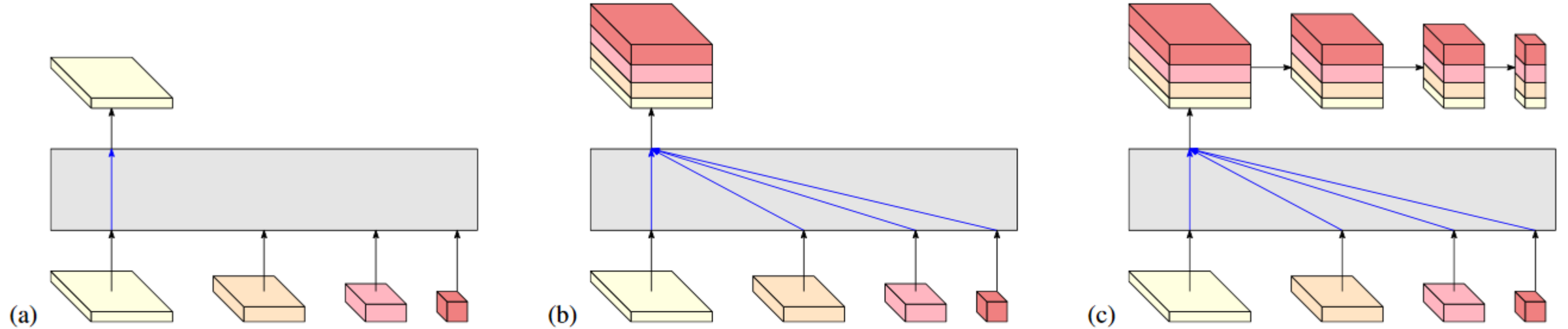


Figure 3. (a) The high-resolution representation proposed in [91] (HRNetV1); (b) Concatenating the (upsampled) representations that are from all the resolutions for semantic segmentation and facial landmark detection (HRNetV2); (c) A feature pyramid formed over (b) for object detection (HRNetV2p). The four-resolution representations at the bottom in each sub-figure are outputted from the network in Figure 1, and the gray box indicates how the output representation is obtained from the input four-resolution representations.

Ke Sun et al. «High-Resolution Representations for Labeling Pixels and Regions» // https://arxiv.org/abs/1904.04514

Сети с вниманием: DANet = Dual Attention Network

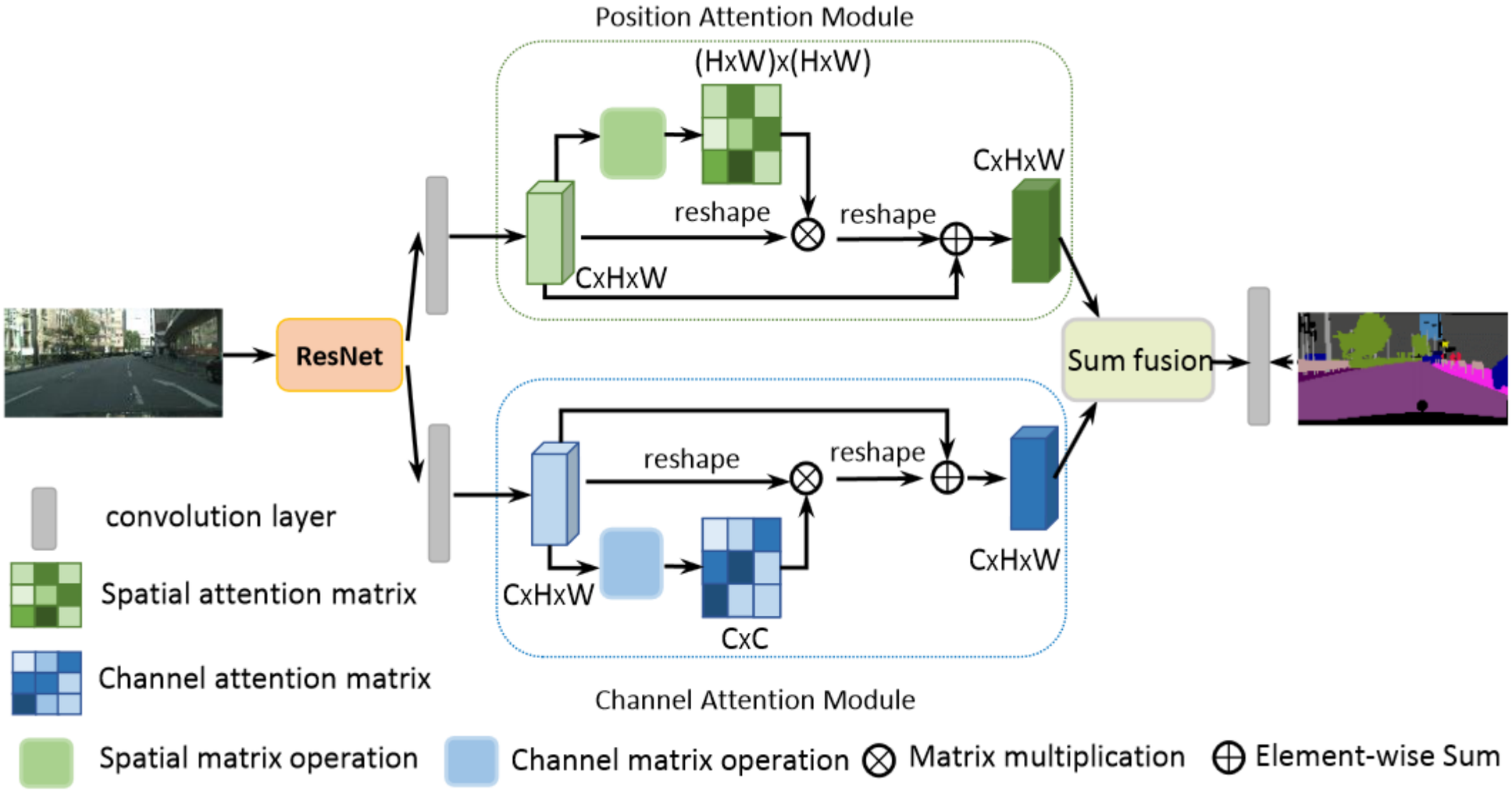
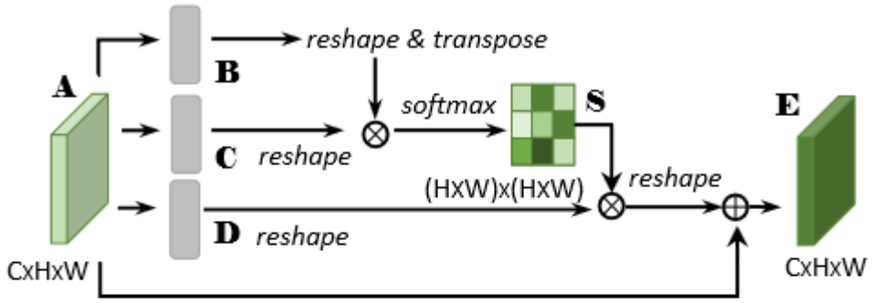


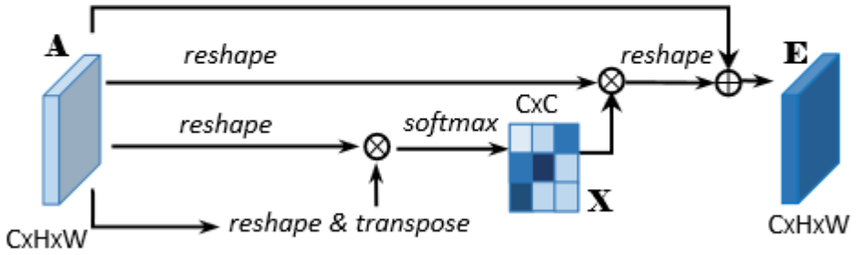
Figure 2: An overview of the Dual Attention Network. (Best viewed in color)

Jun Fu «Dual Attention Network for Scene Segmentation» https://arxiv.org/abs/1809.02983

Сети с вниманием: DANet = Dual Attention Network



A. Position attention module



B. Channel attention module

Figure 3: The details of Position Attention Module and Channel Attention Module are illustrated in (A) and (B). (Best viewed in color)

Два разных механизма внимания: по пикселям и по каналам

оба механизма важны

про внимание – дальше

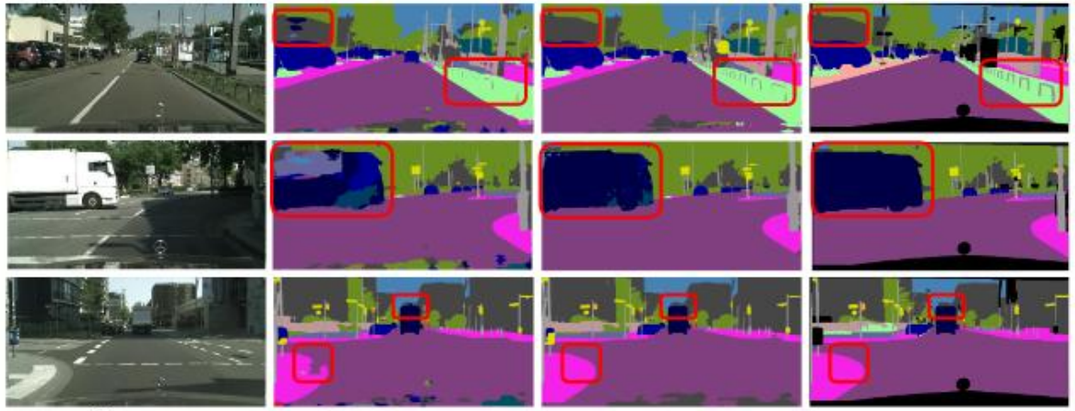


Figure 4: Visualization results of position attention module on Cityscapes val set.

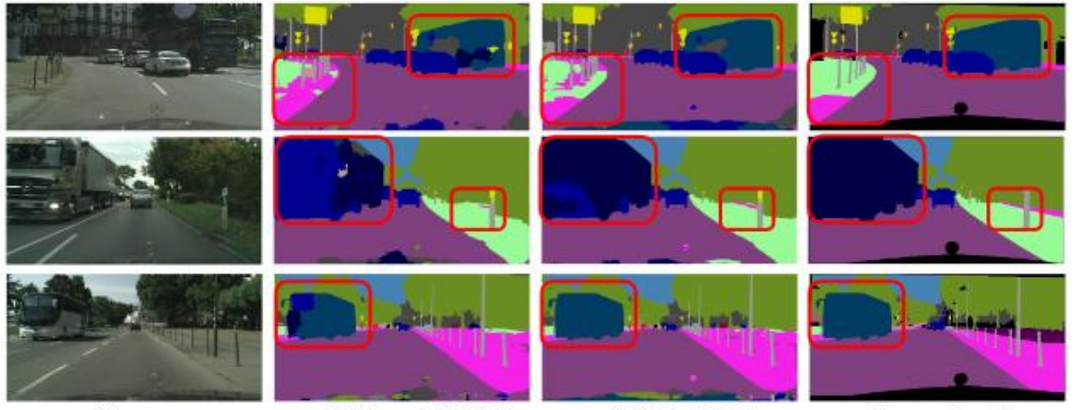


Figure 5: Visualization results of channel attention module on Cityscapes val set.

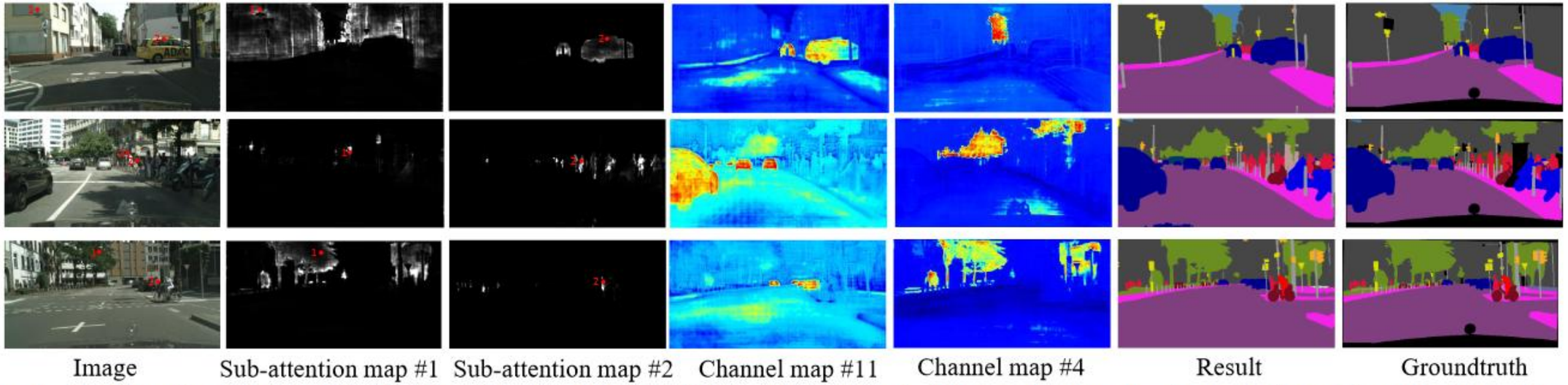


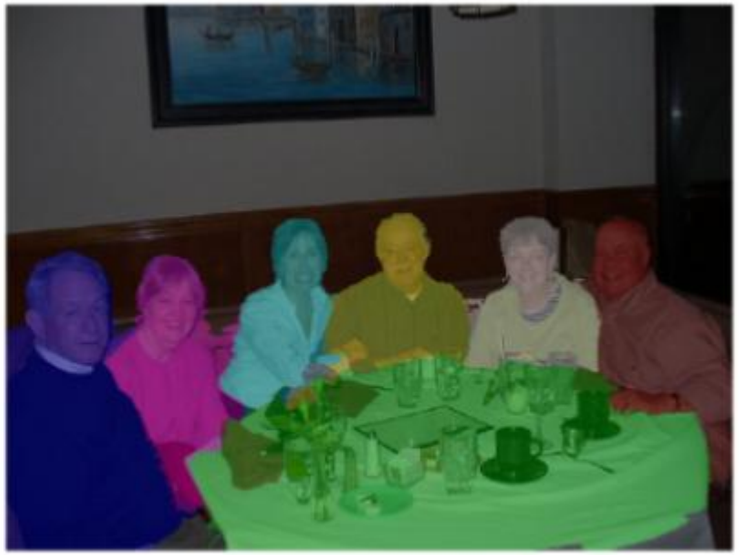
Figure 6: Visualization results of attention modules on Cityscapes val set. For each row, we show an input image, two sub-attention maps $(H \times W)$ corresponding to the ponits marked in the input image. Meanwhile, we give two channel maps from the outputs of channel attention module, where the maps are from 4^{th} and 11^{th} channels, respectively. Finally, corresponding result and groundtruth are provided.

Аналогичная задача: удаление фона

https://medium.com/@yunanwu2020/go-selfies-how-to-do-photo-background-removal-using-deep-learning-segmentation-codes-6131d59a2ca9

Сегментация объектов (Instance segmentation)

Semantic Segmentation



Instance Segmentation

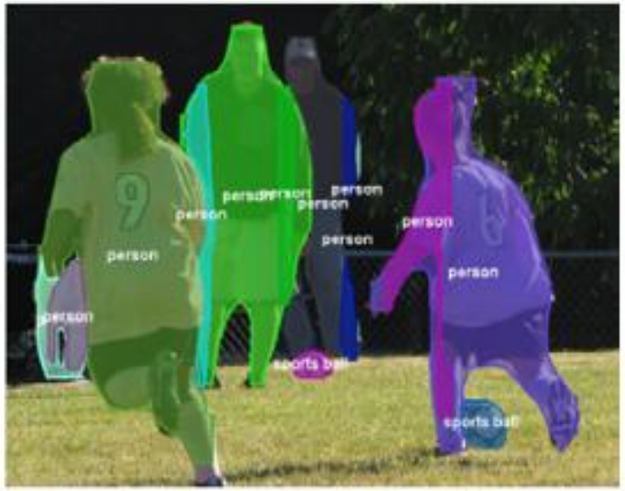
в семантической сегментации каждому пикселю – класс
в сегментации объектов надо различать группы пикселей формально одного класса, но принадлежащие разным объектам

https://neurohive.io/ru/osnovy-data-science/semantic-segmention/

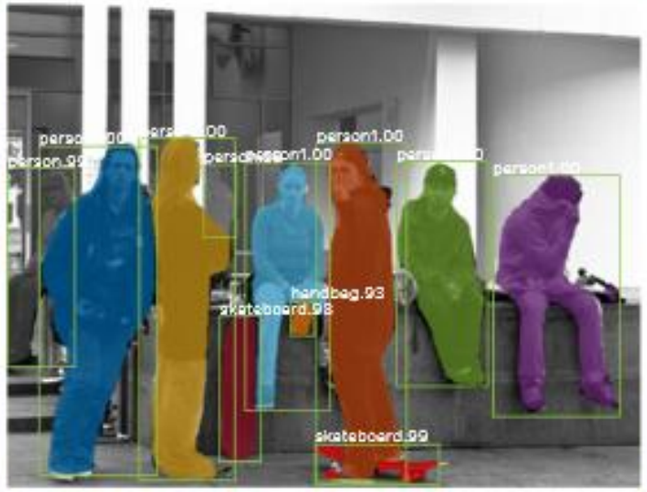
Сегментация объектов (Instance segmentation)

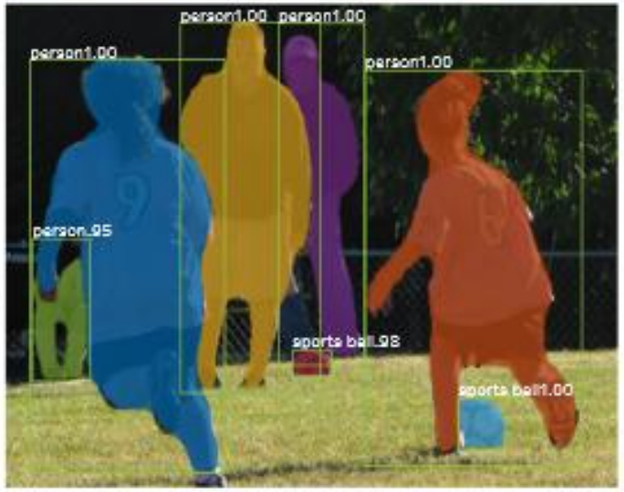
не просто найти рамку, а выделить объект!

FCIS



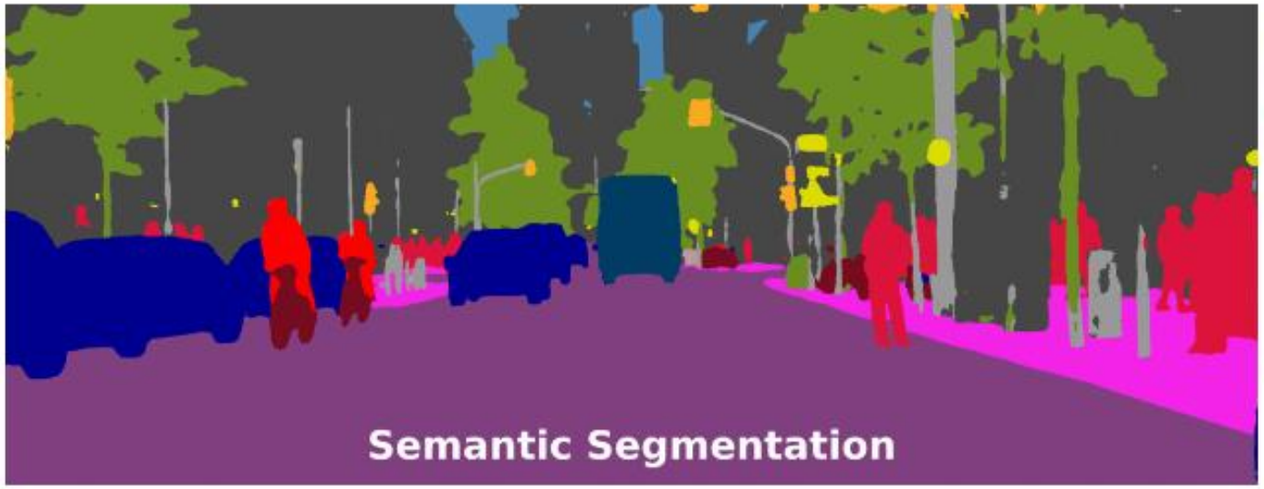
Mask R-CNN

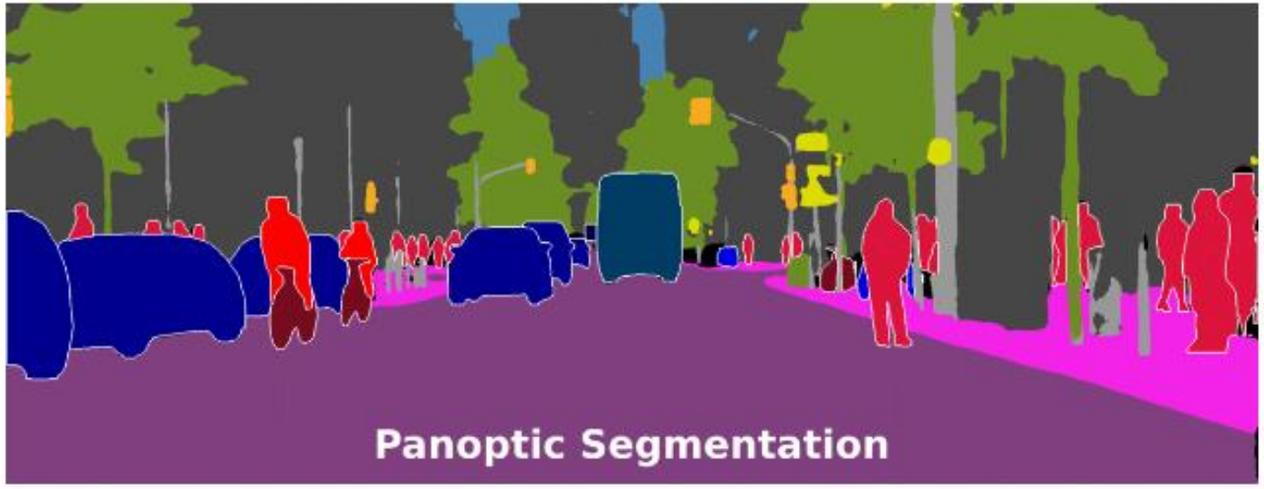




Mask R-CNN https://arxiv.org/pdf/1703.06870.pdf

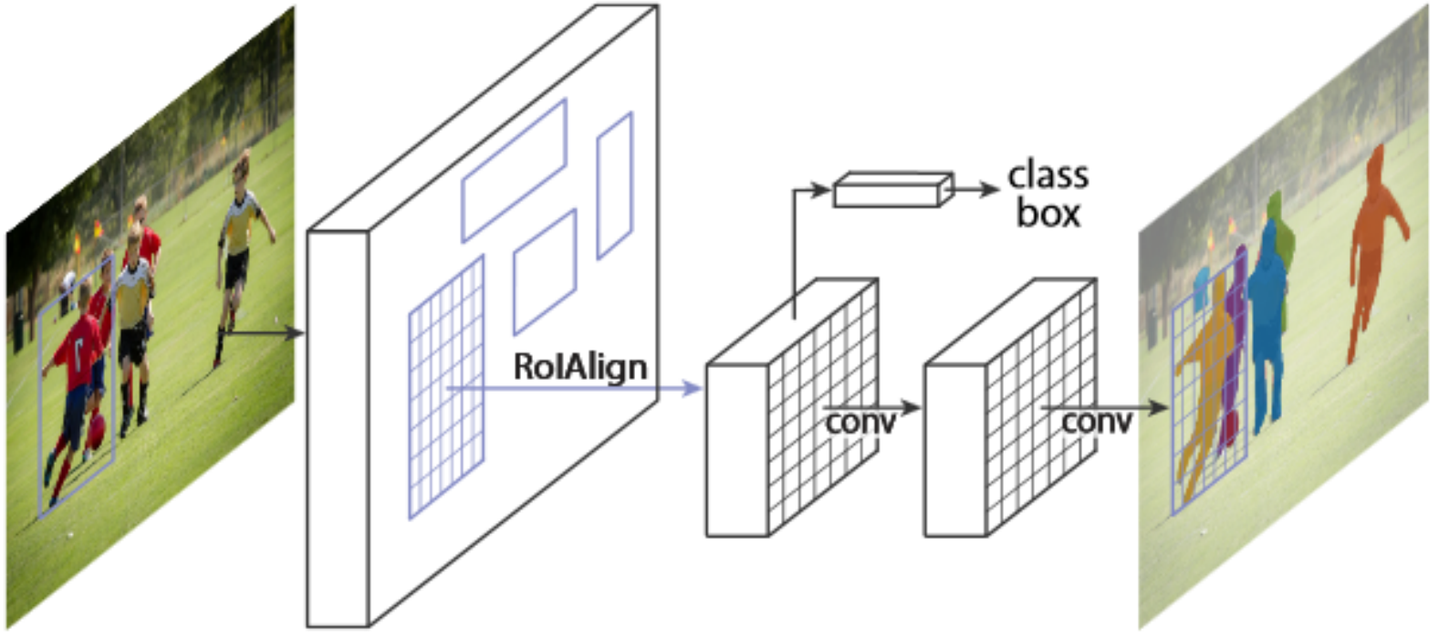
Виды сегментации





Panoptic Segmentation – выделение объектов на переднем и дальнем планах

Mask R-CNN



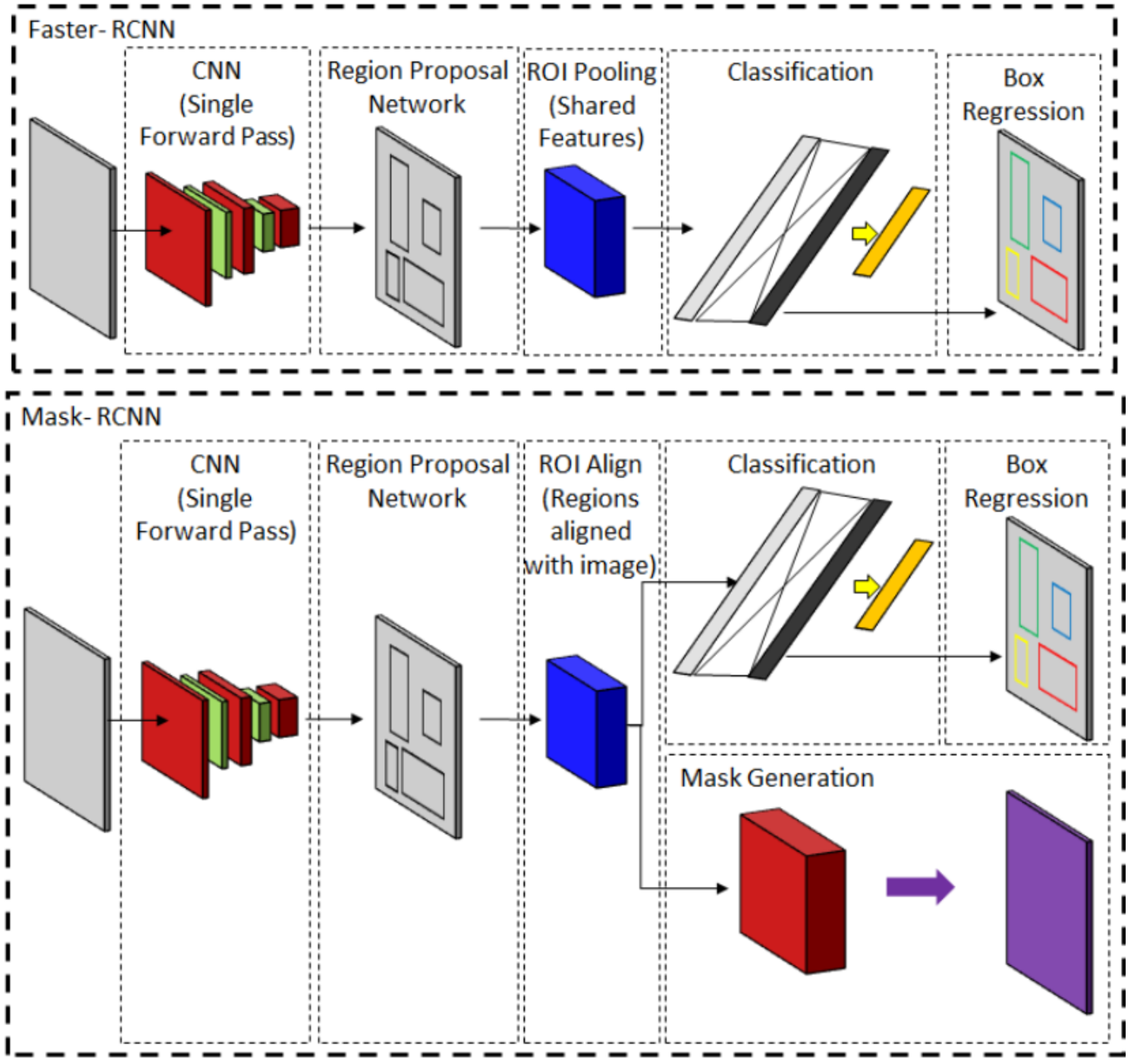
Faster R-CNN + определение сегментационной маски (маленькая FCN)

+ борьба за более точное определение границ

Хороши также для определения позы

специальные маски для детектирования опорных точек (локоть, плечо и т.п.)

– один пиксель =1, остальные =0


Faster- RCNN
CNN (Single Forward Pass)
Region Proposal Network
ROI Pooling (Shared Features)
Classification
Box Regression
Mask- RCNN
CNN (Single Forward Pass)
Region Proposal Network
ROI Align (Regions aligned with image)
Classification
Box Regression
Mask Generation

Mask R-CNN

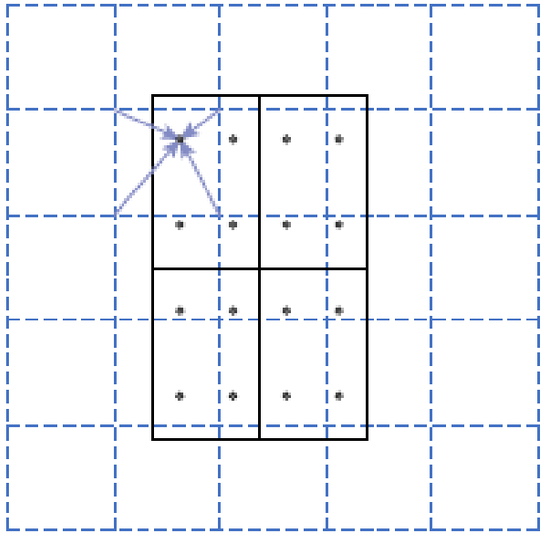
маска вида 1/0 – принадлежит или нет объекту (class-agnostic)

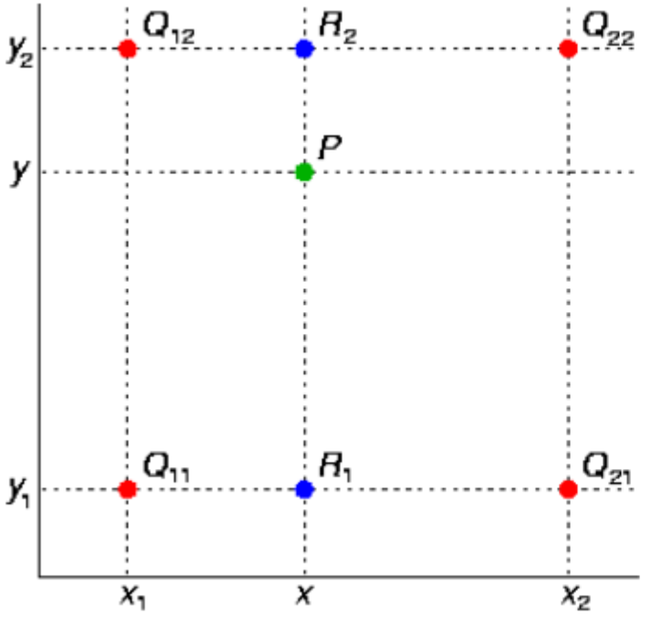
для опорных точек – ОНЕ таких точек

ошибка = сумма трёх (класс, регрессия региона, маска)

RoIPool → RoIAlign

т.к. на карте признаков уменьшенного (в крупном разрешении) не получается точно разместить регион: билинейная интерполяция





$$R_1 = (x, y_1)$$

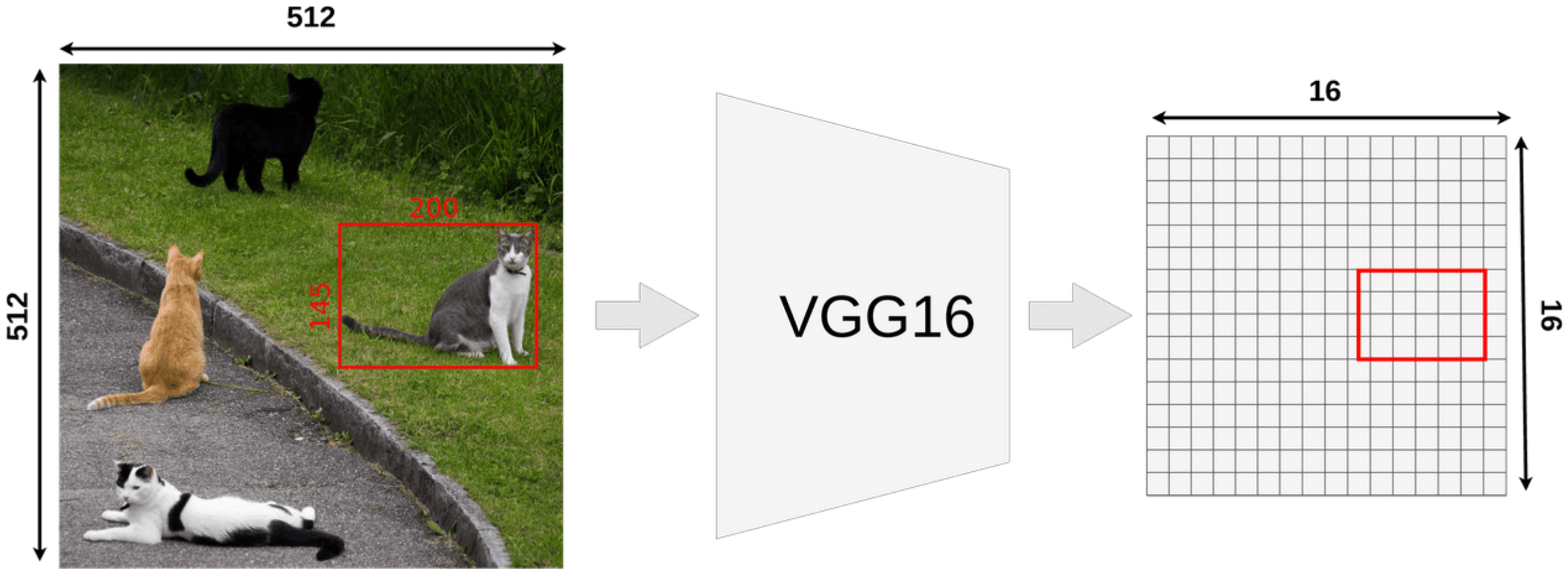
$$R_2 = (x, y_2)$$

$$f(R_1) \approx \frac{(x_2 - x)}{(x_2 - x_1)} f(Q_{11}) + \frac{(x - x_1)}{(x_2 - x_1)} f(Q_{21})$$

$$f(R_2) \approx \frac{(x_2 - x)}{(x_2 - x_1)} f(Q_{12}) + \frac{(x - x_1)}{(x_2 - x_1)} f(Q_{22})$$

$$f(P) \approx \frac{(y_2 - y)}{(y_2 - y_1)} f(R_1) + \frac{(y - y_1)}{(y_2 - y_1)} f(R_2)$$

Mask R-CNN: RoIPool → RoIAlign



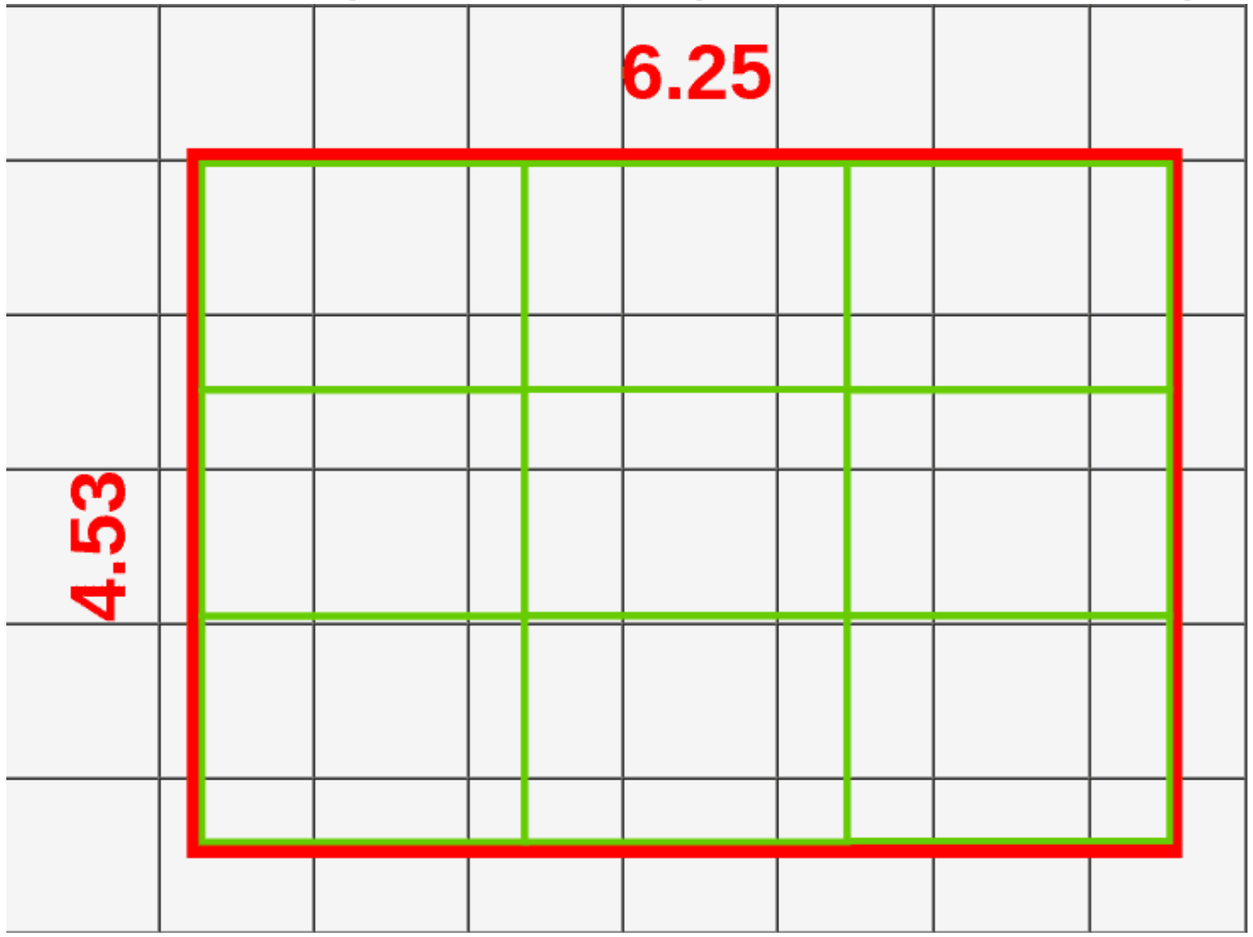
На уровне более крупных пикселей – мы должны резать «по ним»

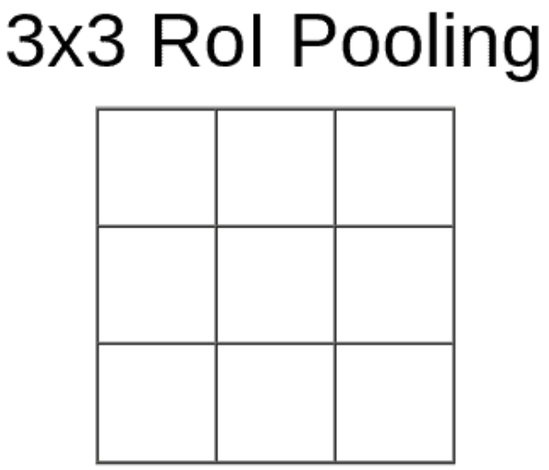
Раньше было так, и это не проблема на уровне рамок – сейчас попиксельные ответы

https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

Mask R-CNN: RoIPool → RoIAlign

Пусть после пулинга хотим получить 3×3-тензор





первая ячейка захватывает 6 пикселей

делаем интерполяцию 4х точек и выбираем max (см. след. слайд)

$$P \approx \frac{7.5-7.01}{7.5-6.5}\left(\frac{11.5-10.64}{11.5-10.5}0.2+\frac{10.64-10.5}{11.5-10.5}0.3\right)+\frac{7.01-6.5}{7.5-6.5}\left(\frac{11.5-10.64}{11.5-10.5}0.7+\frac{10.64-10.5}{11.5-10.5}0.2\right)$$

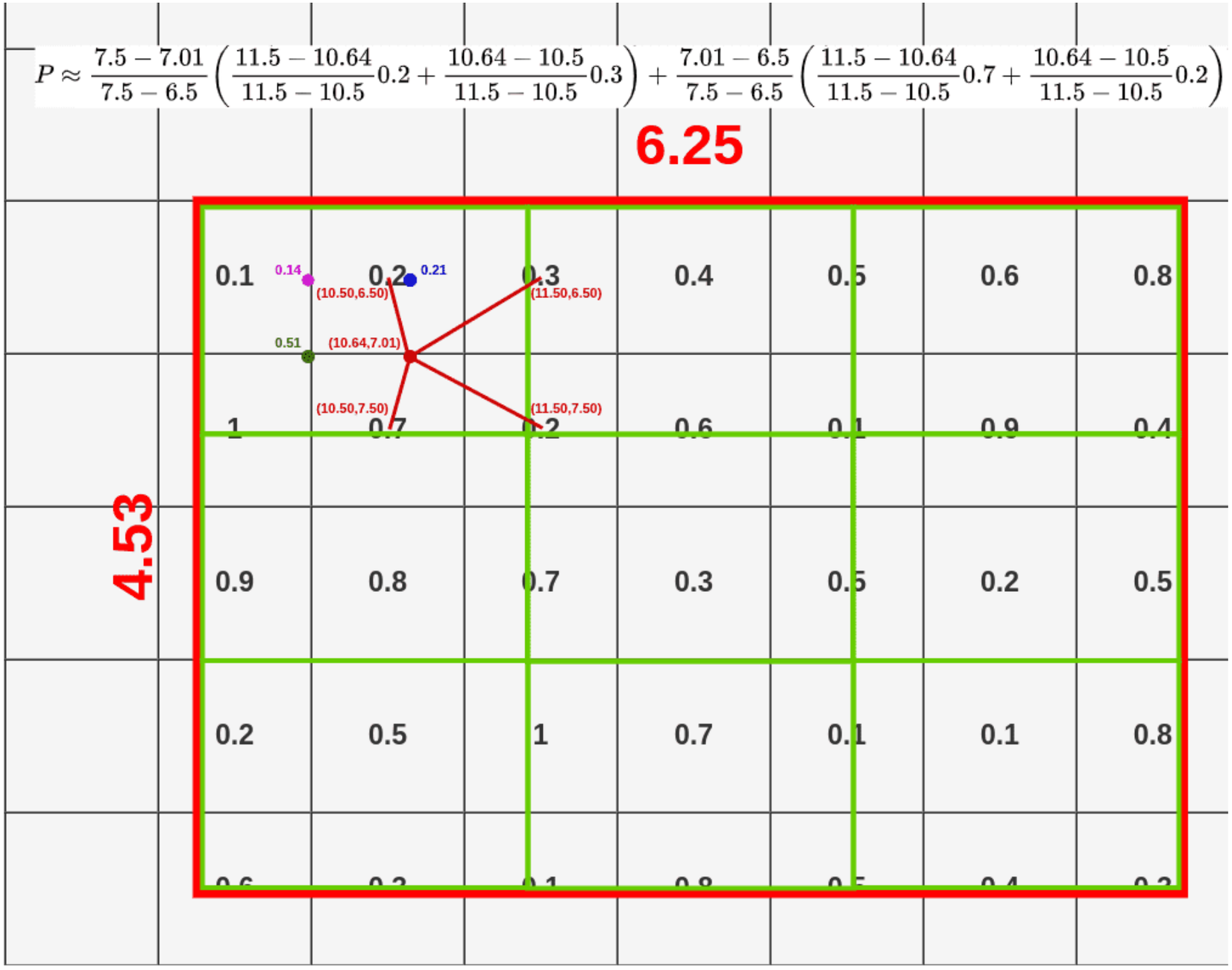


Figure 5. More results of **Mask R-CNN** on COCO test images, using ResNet-101-FPN and running at 5 fps, with 35.7 mask AP (Table 1).

PANet: Path Aggregation Network

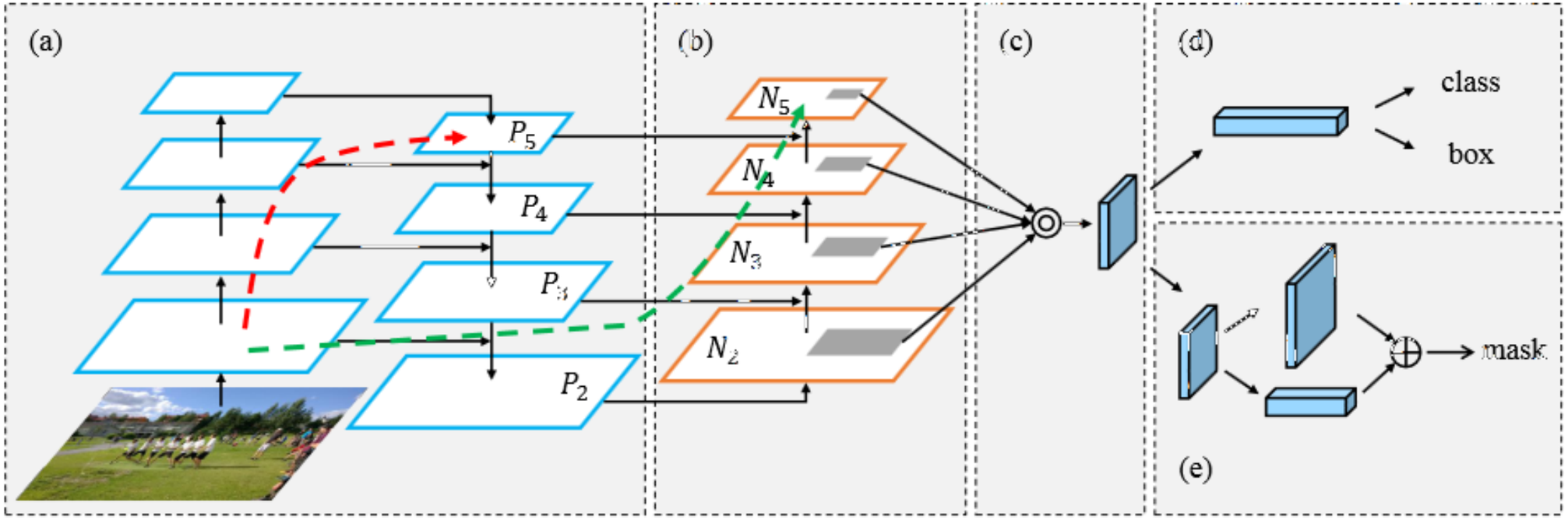


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path augmentation. (c) Adaptive feature pooling. (d) Box branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature maps in (a) and (b) for brevity.

Mask R-CNN + FPN

Shu Liu et al. «Path Aggregation Network for Instance Segmentation» // https://arxiv.org/pdf/1803.01534.pdf

PolarMask: 2019

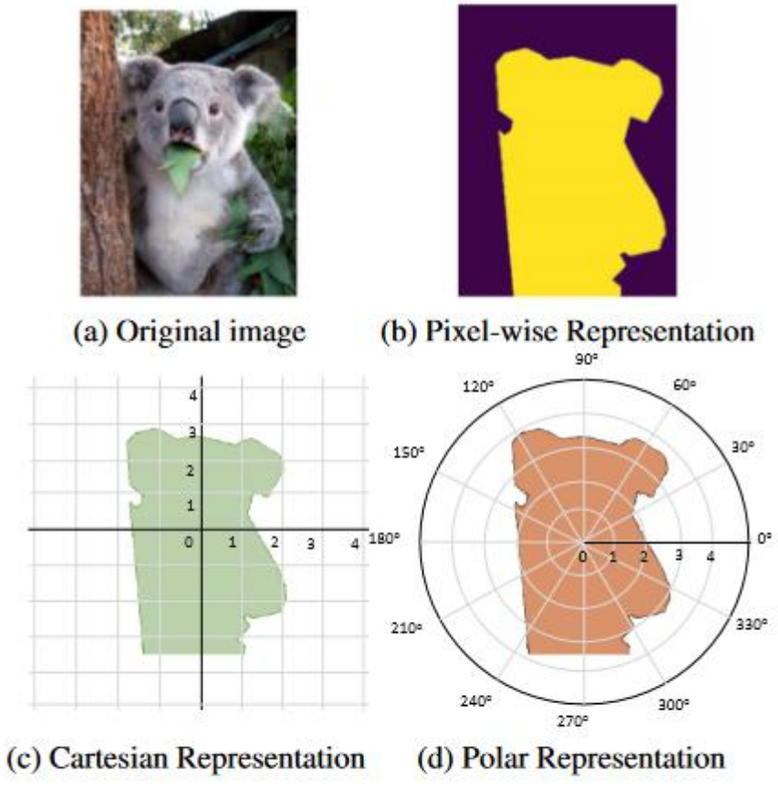


Figure 1 – Instance segmentation with different mask representations. (a) is the original image. (b) is the pixel-wise mask representation. (c) and (d) represent a mask by its contour, in the Cartesian and Polar coordinates, respectively.

Проведём несколько (k) лучей из конкретной точки (они равномерно по углам разбивают 360°)

Аппроксимируем маску многоугольником с вершинами, в которых лучи пересекают границу

Тогда каждая аппроксимация описывается k числами

Идея: их восстанавливать для каждой точки

– не всегда хорошая аппроксимация
+ маски могут пересекаться

Enze Xie, et al. «PolarMask: Single Shot Instance Segmentation with Polar Representation»
https://arxiv.org/pdf/1909.13226.pdf

PolarMask: 2019

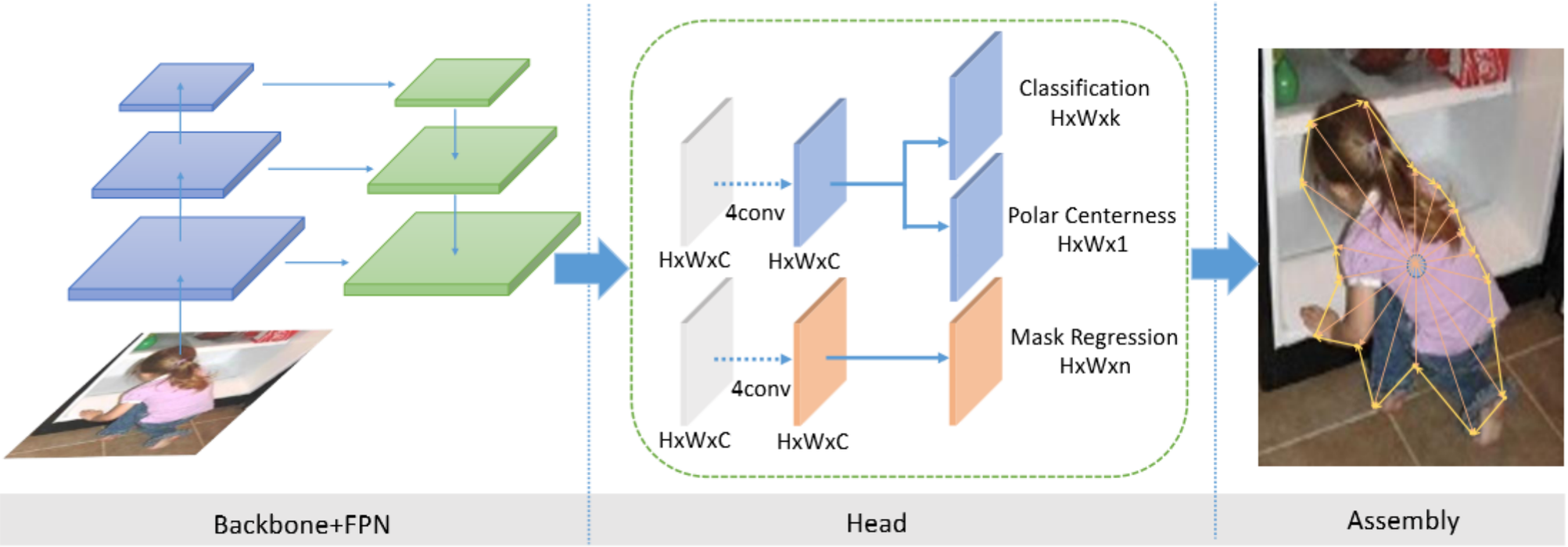


Figure 2 – The overall pipeline of PolarMask. The left part contains the backbone and feature pyramid to extract features of different levels. The middle part is the two heads for classification and polar mask regression. H, W, C are the height, width, channels of feature maps, respectively, and k is the number of categories (e.g., $k = 80$ on the COCO dataset), n is the number of rays (e.g., $n = 36$)

TensorMask: 2019

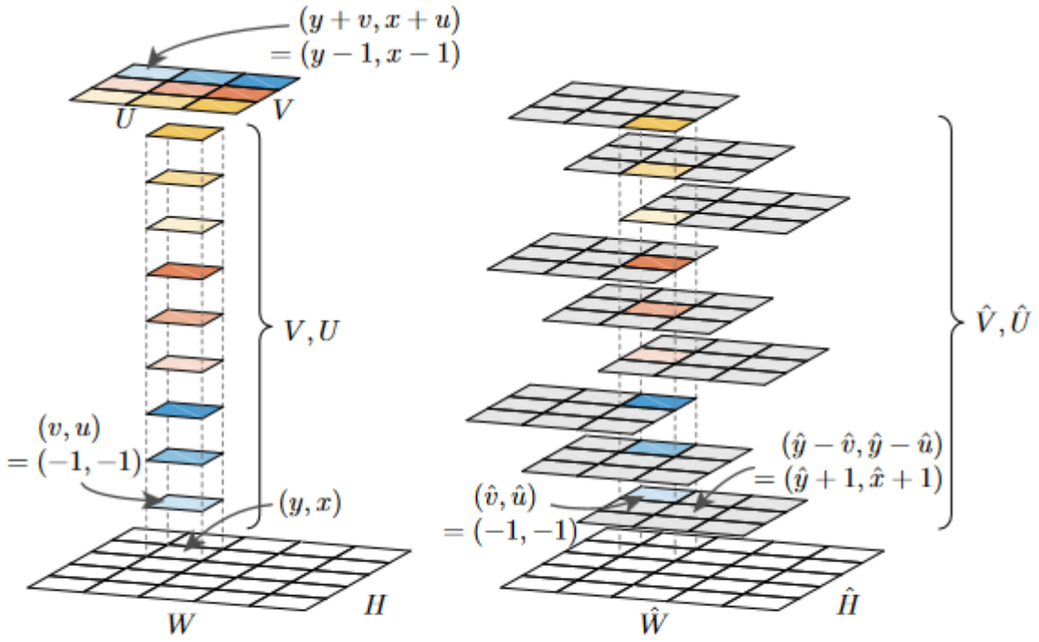


Figure 3. Left: **Natural representation**. The (V, U) sub-tensor at a pixel represents a window centered at this pixel. Right: **Aligned representation**. The $(\hat{V}, \hat{U})$ sub-tensor at a pixel represents the values at this pixel in each of the windows overlapping it.

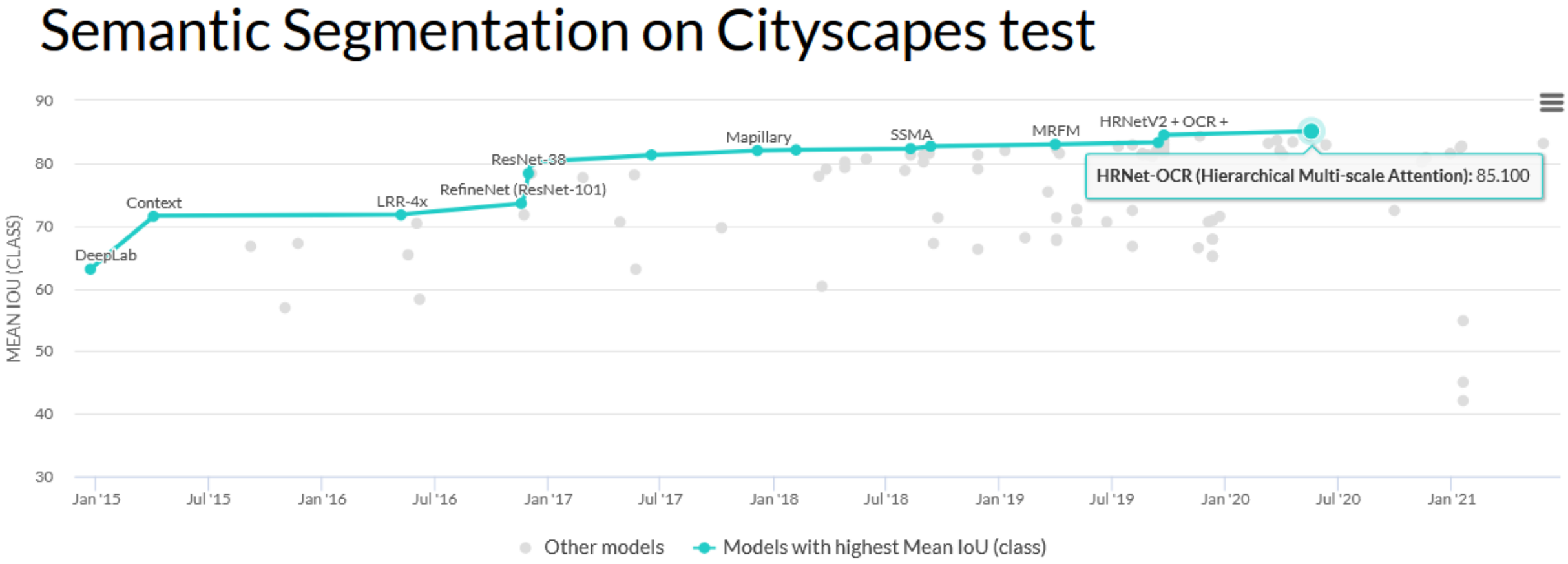
Для каждой точки (из 2D) предсказываем 2х-мерную матрицу – маску её объекта (H×W×H×W)

– тяжело учится

+ хороша, когда объекты плотно расположены

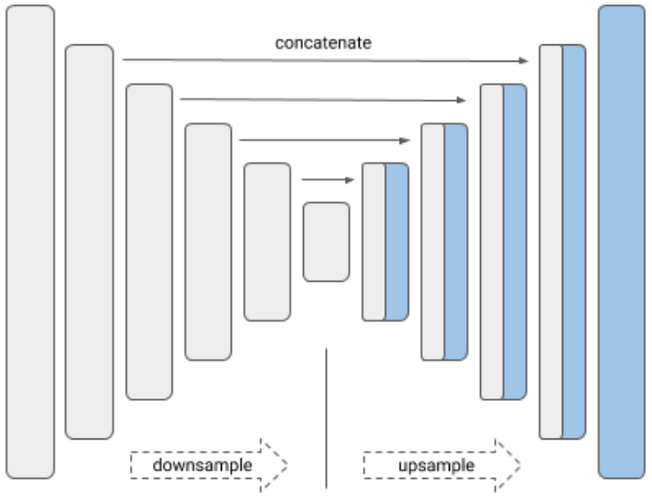
Xinlei Chen et al. «TensorMask: A Foundation for Dense Object Segmentation»
https://arxiv.org/pdf/1903.12174.pdf

SOTA 2021

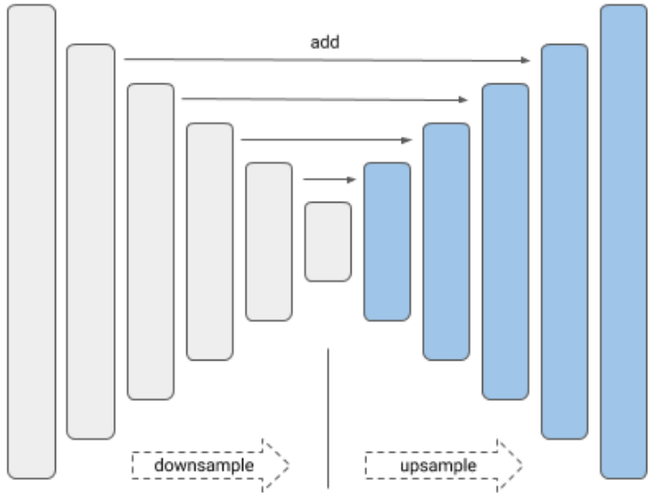


Tao at al. «Hierarchical Multi-Scale Attention for Semantic Segmentation» //
https://arxiv.org/pdf/2005.10821v1.pdf
https://paperswithcode.com/sota/semantic-segmentation-on-cityscapes

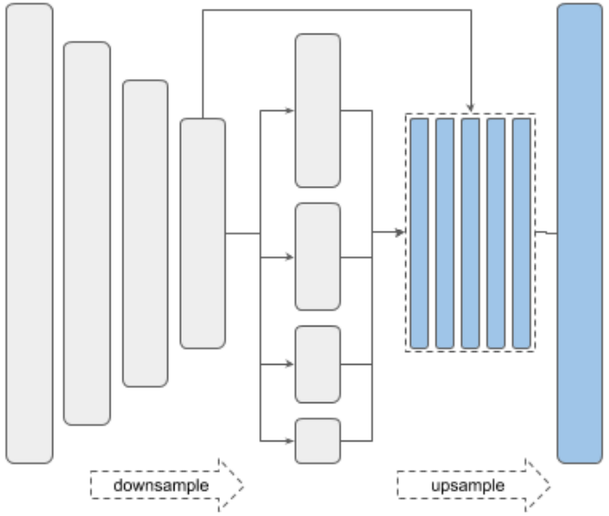
Итог: Модели для сегментации



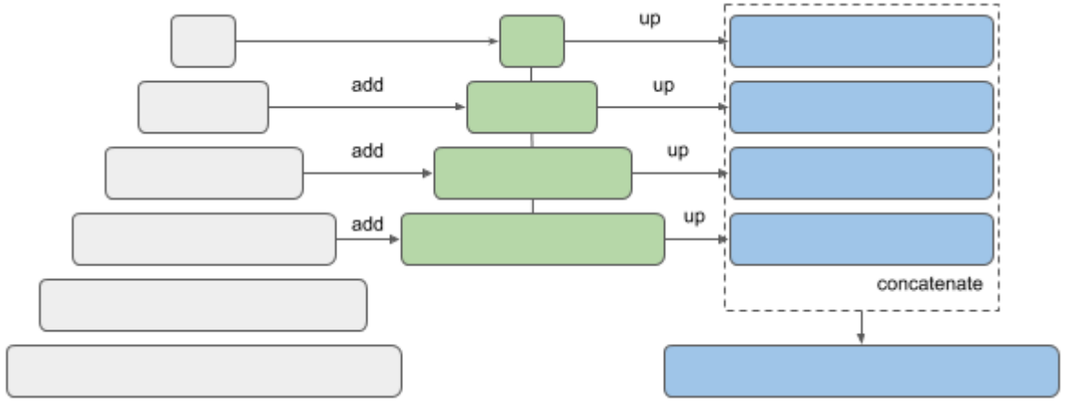
Unet



Linknet



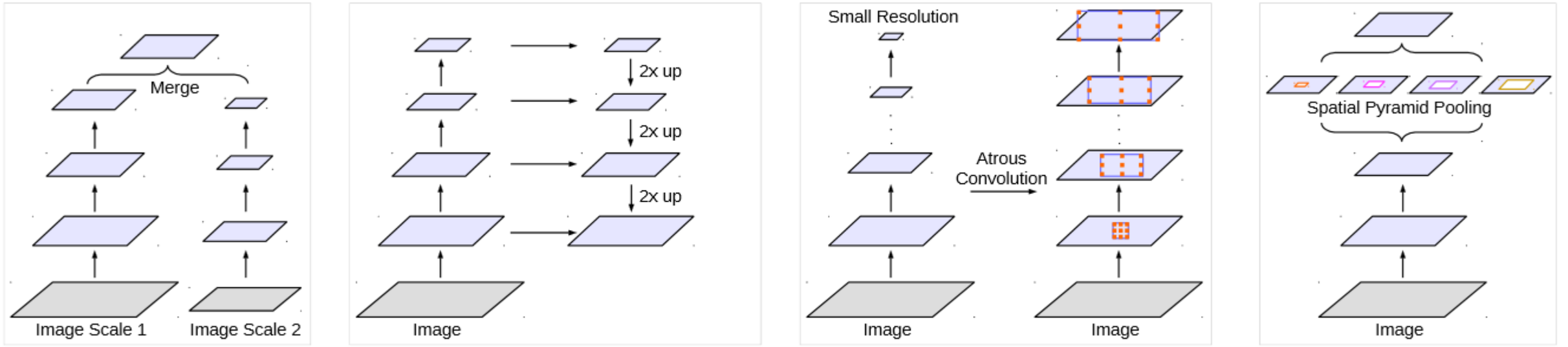
PSPNet



FPN

https://github.com/qubvel/segmentation_models

Итог: Подходы к использованию информации с разных масштабов



(a) Image Pyramid (b) Encoder-Decoder (c) Deeper w. Atrous Convolution (d) Spatial Pyramid Pooling

Figure 2. Alternative architectures to capture multi-scale context.

https://arxiv.org/pdf/1706.05587.pdf

Итог: история методов

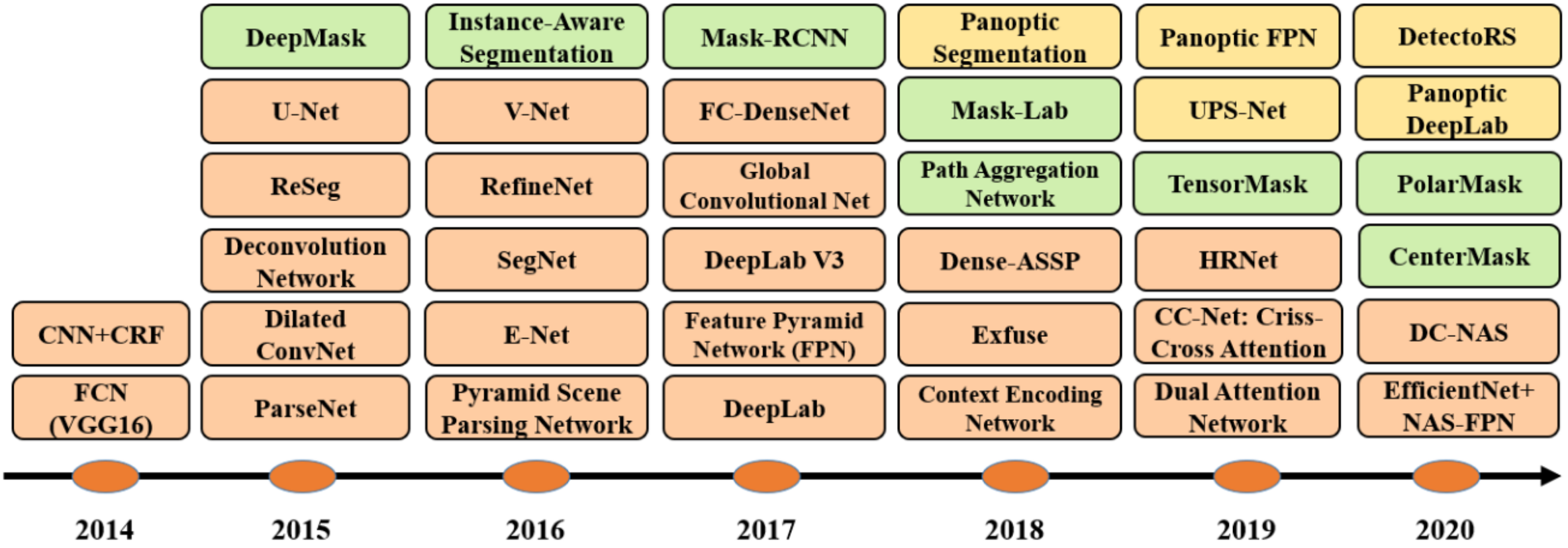


Fig. 32. The timeline of DL-based segmentation algorithms for 2D images, from 2014 to 2020. Orange, green, andn yellow blocks refer to semantic, instance, and panoptic segmentation algorithms respectively.

Shervin Minaee et al «Image Segmentation Using Deep Learning: A Survey» //
https://arxiv.org/pdf/2001.05566.pdf

Итог: история методов

Accuracies of segmentation models on the PASCAL VOC test set. (* Refers to the model pre-trained on another dataset (such as MS-COCO, ImageNet, or JFT-300M).)

Method	Backbone	mIoU
FCN [31]	VGG-16	62.2
CRF-RNN [39]	-	72.0
CRF-RNN* [39]	-	74.7
BoxSup* [117]	-	75.1
Piecewise* [40]	-	78.0
DPN* [41]	-	77.5
DeepLab-CRF [78]	ResNet-101	79.7
GCN* [118]	ResNet-152	82.2
RefineNet [115]	ResNet-152	84.2
Wide ResNet [119]	WideResNet-38	84.9
PSPNet [56]	ResNet-101	85.4
DeeplabV3 [12]	ResNet-101	85.7
PSANet [98]	ResNet-101	85.7
EncNet [114]	ResNet-101	85.9
DFN* [99]	ResNet-101	86.2
Exfuse [120]	ResNet-101	86.2
SDN* [45]	DenseNet-161	86.6
DIS [123]	ResNet-101	86.8
DM-Net* [58]	ResNet-101	87.06
APC-Net* [60]	ResNet-101	87.1
EMANet [95]	ResNet-101	87.7
DeeplabV3+ [83]	Xception-71	87.8
Exfuse [120]	ResNeXt-131	87.9
MSCI [61]	ResNet-152	88.0
EMANet [95]	ResNet-152	88.2
DeeplabV3+* [83]	Xception-71	89.0
EfficientNet+NAS-FPN [135]	-	90.5

TABLE 2
Accuracies of segmentation models on the Cityescapes dataset.

Method	Backbone	mIoU
FCN-8s [31]	-	65.3
DPN [41]	-	66.8
Dilation10 [79]	-	67.1
DeeplabV2 [78]	ResNet-101	70.4
RefineNet [115]	ResNet-101	73.6
FoveaNet [124]	ResNet-101	74.1
Ladder DenseNet [125]	Ladder DenseNet-169	73.7
GCN [118]	ResNet-101	76.9
DUC-HDC [80]	ResNet-101	77.6
Wide ResNet [119]	WideResNet-38	78.4
PSPNet [56]	ResNet-101	85.4
BiSeNet [126]	ResNet-101	78.9
DFN [99]	ResNet-101	79.3
PSANet [98]	ResNet-101	80.1
DenseASPP [81]	DenseNet-161	80.6
SPGNet [127]	2xResNet-50	81.1
DANet [93]	ResNet-101	81.5
CCNet [96]	ResNet-101	81.4
DeeplabV3 [12]	ResNet-101	81.3
AC-Net [129]	ResNet-101	82.3
OCR [44]	ResNet-101	82.4
GS-CNN [128]	WideResNet	82.8
HRNetV2+OCR (w/ASPP) [44]	HRNetV2-W48	83.7
Hierarchical MSA [137]	HRNet-OCR	85.1

TABLE 3
Accuracies of segmentation models on the MS COCO stuff dataset.

Method	Backbone	mIoU
RefineNet [115]	ResNet-101	33.6
CCN [59]	Ladder DenseNet-101	35.7
DANet [93]	ResNet-50	37.9
DSSPN [130]	ResNet-101	37.3
EMA-Net [95]	ResNet-50	37.5
SGR [131]	ResNet-101	39.1
OCR [44]	ResNet-101	39.5
DANet [93]	ResNet-101	39.7
EMA-Net [95]	ResNet-50	39.9
AC-Net [129]	ResNet-101	40.1
OCR [44]	HRNetV2-W48	40.5

TABLE 5
Instance Segmentation Models Performance on COCO test-dev 2017

Method	Backbone	FPS	AP
YOLACT-550 [76]	R-101-FPN	33.5	29.8
YOLACT-700 [76]	R-101-FPN	23.8	31.2
RetinaMask [170]	R-101-FPN	10.2	34.7
TensorMask [69]	R-101-FPN	2.6	37.1
SharpMask [171]	R-101-FPN	8.0	37.4
Mask-RCNN [64]	R-101-FPN	10.6	37.9
CenterMask [74]	R-101-FPN	13.2	38.3

Итог

сегментация: опять свёрточные решения

сегментация и детекция – есть общие принципы

Идеи:

полносвёрточная парадигма

кодировщик-декодировщик

+ прокидывание связей

собирать информацию разного разрешения

- **EncoderDecode + skip**
- **ED → Top to Down Refinement**
- **DilatedFCN**
- **dilated convolutions → Joint Pyramid Upsampling(JPU)**

Ссылки

Convolutional Neural Networks for Visual Recognition

http://cs231n.stanford.edu/

Трюки с сегментацией на Kaggle

https://neptune.ai/blog/image-segmentation-tips-and-tricks-from-kaggle-competitions

Обзор

Shervin Minaee et al «Image Segmentation Using Deep Learning: A Survey» //
https://arxiv.org/pdf/2001.05566.pdf

Обзор (взяты некоторые картинки)

«Understanding Deep Learning Techniques for
Image Segmentation» // https://arxiv.org/pdf/1907.06119v1.pdf