Semantic Segmentation, Keypoints, Detection

Alexey Sidnev, 2020.11.20

alexey.sidnev@gmail.com

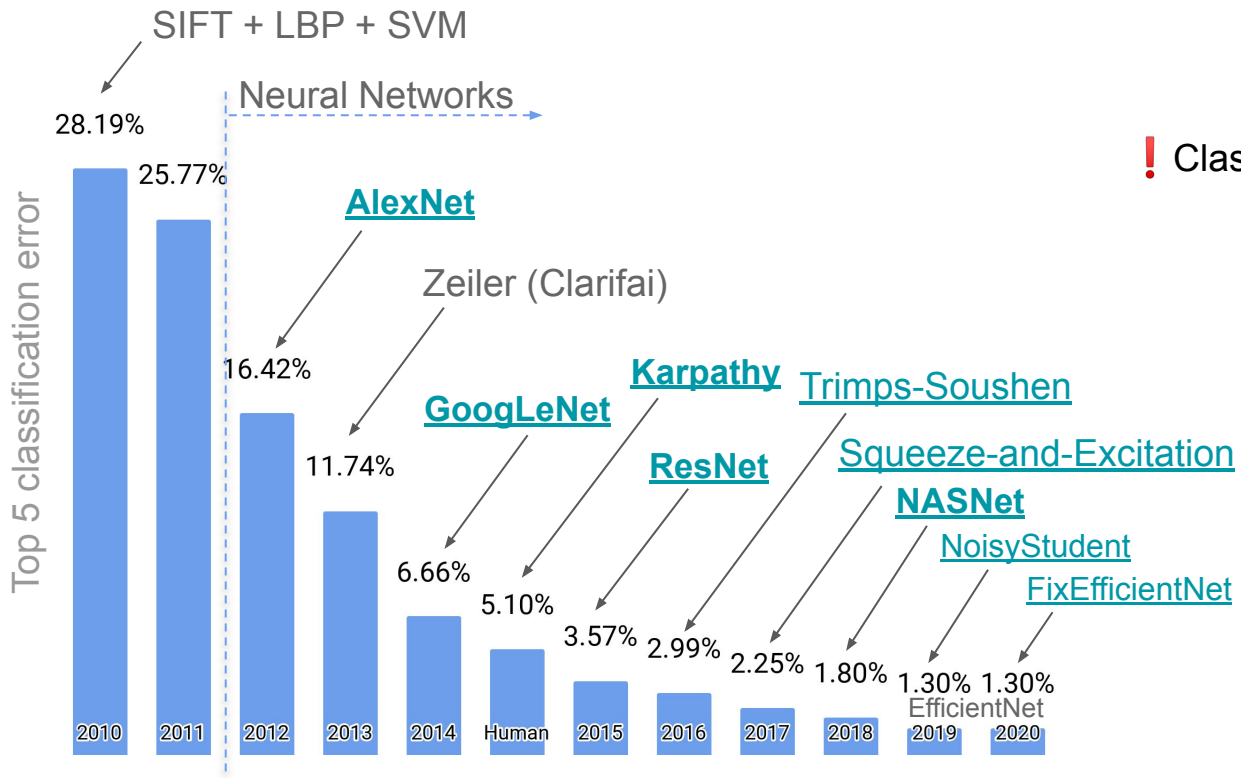
Agenda

1. Classification / Object detection / Instance segmentation / Semantic segmentation
2. Semantic segmentation datasets, evaluation metrics, and architectures
    1. FCN
    2. CRF / DeepLab v1 / DeepLab v2
    3. Parsenet
    4. U-Net
    5. SegNet
    6. ENet
    7. PSPNet
    8. ICNet
    9. DeepLab v3 / DeepLab v3+
    10. HRNet
3. Keypoint detection
4. Object detection

1. Computer Vision Task

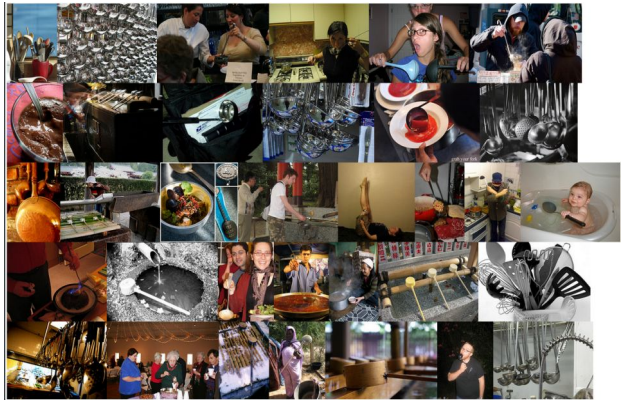
Classification, Object Detection
Semantic and Instance Segmentation

Image recognition (ImageNet ILSVRC)



❗ Classification ≠ Multi-label classification

Ladles



1000 categories, 1.2M train images, 100K test images

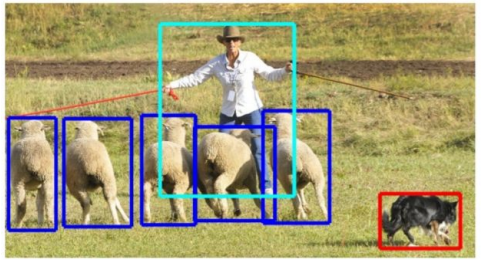
Image recognition: What do you see?

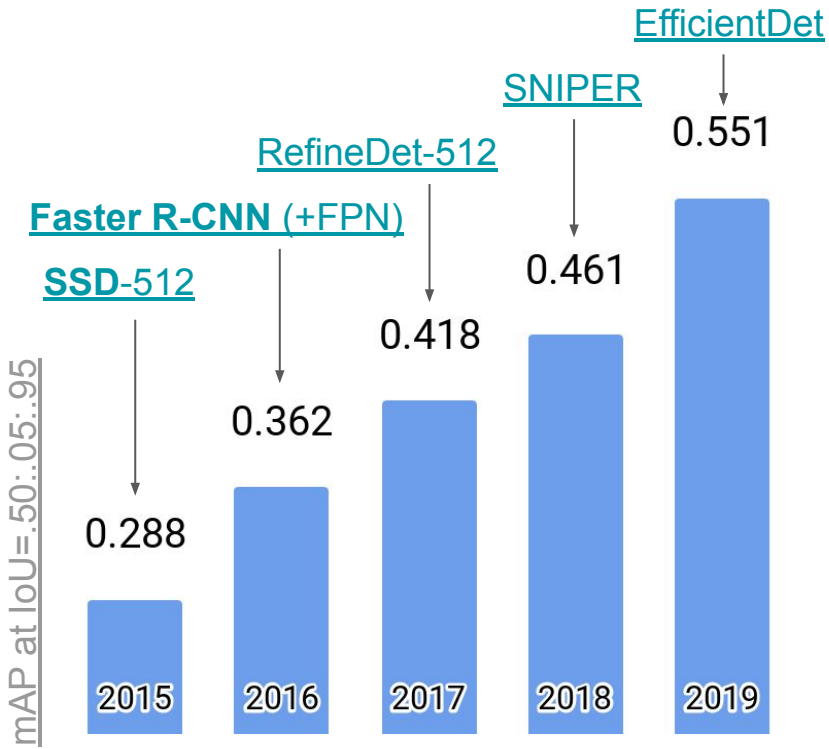
Image recognition: Annotation

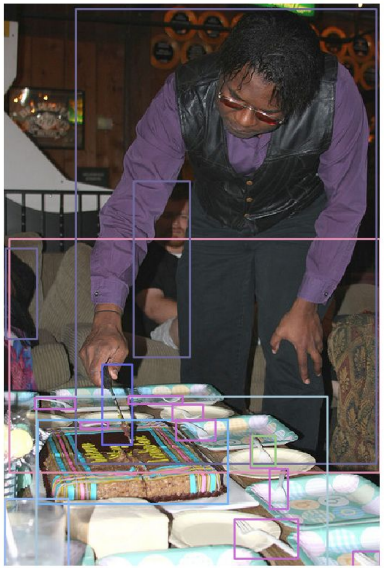
Spatula!

Pajama!!!

Object detection (MS COCO)

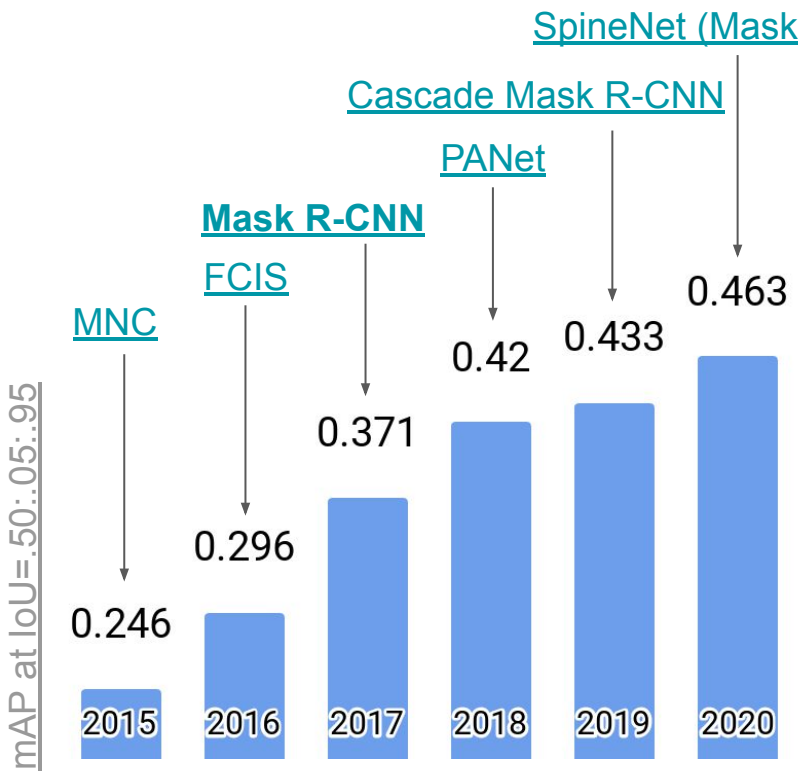






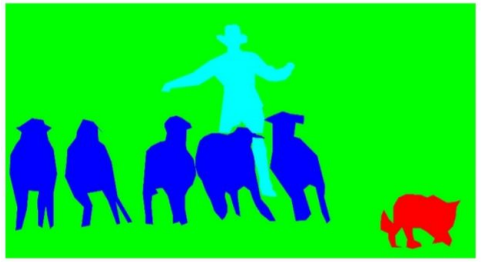
80 categories, 200K train images, 80K test images

Instance segmentation (MS COCO)



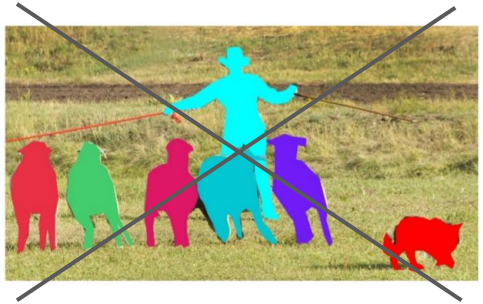
80 categories, 200K train images, 80K test images

Semantic segmentation (Cityscapes)



PSPNet: Pyramid Scene Parsing Network

! Semantic ≠ Instance



2. Semantic Segmentation

Datasets (semantic segmentation)

General:

- Pascal VOC 2012 - 11K images, 20 classes, 7K instances
- ADE20K / SceneParse150K - 22K images, 2 693 classes, 434K instances
- MS COCO - 200K images, 80 classes, instance segmentation
- DAVIS 2017 - video (review)

ADAS:

- Cityscapes - 25K images, 30 classes, 65K instances
- Mapillary Vistas - 20K images, 100 classes
- CamVid - 10 min video, 32 classes
- KITTI road/lane - 289 images
- CMP Facades - 606 images, 12 classes

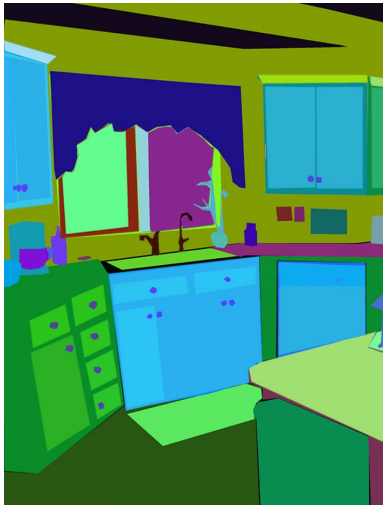
Aerial / Satellite:

- CITY-OSM - ISPRS Vaihingen and Potsdam
- DSTL Kaggle

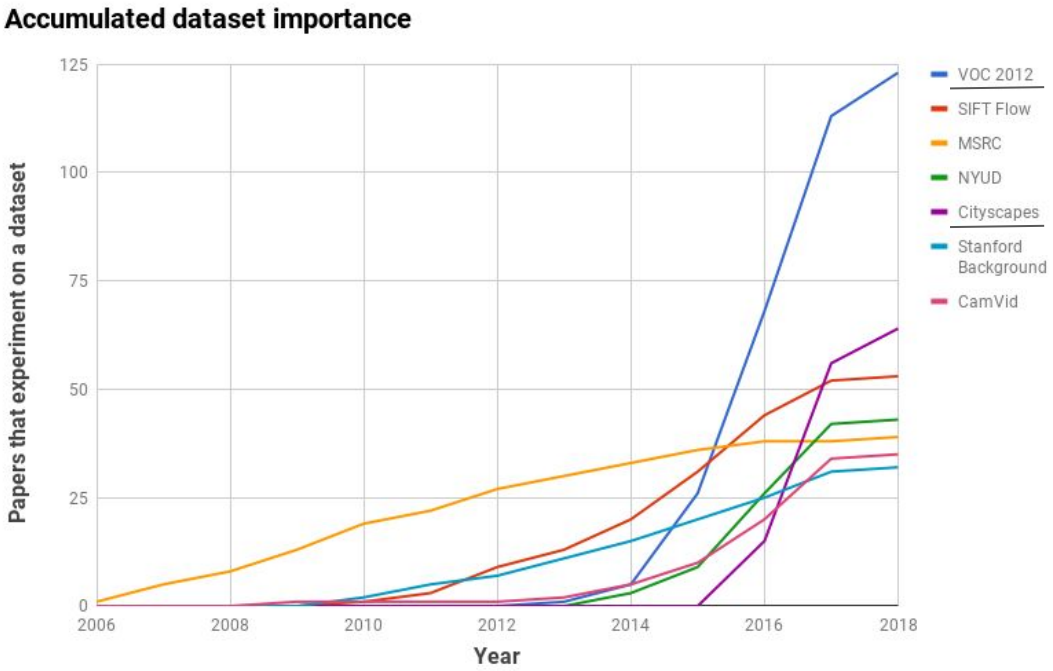
Human parsing:

- LIP (dataset) - 50K images, 19 classes
- MHP - 20K images

More datasets: http://riemenschneider.hayko.at/vision/dataset/



Datasets (papers) + VOC Leaderboard

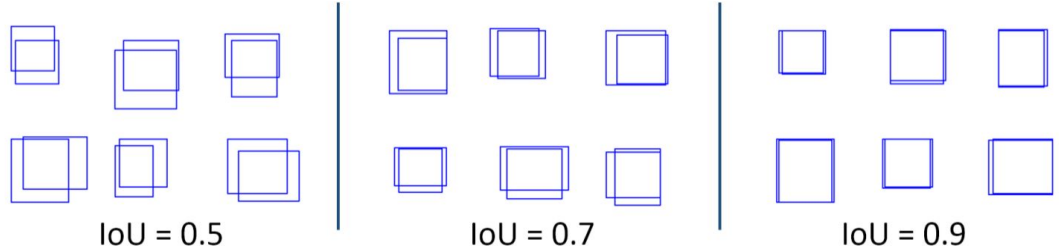


[A list of papers on Semantic Segmentation]

VOC 2012

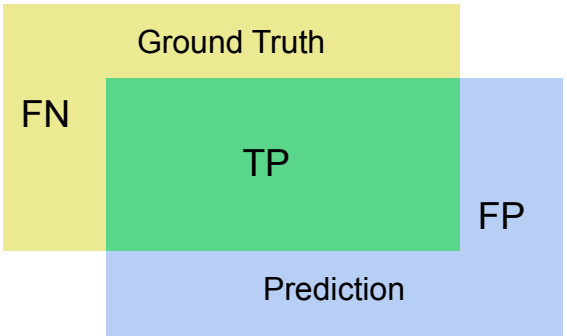
	mean
DeepLabv3+_JFT [?]	89.0
** DeepLabv3+_AASPP ** [?]	88.5
SRC-B-MachineLearningLab [?]	88.5
** ExFuse ** [?]	87.9
DeepLabv3+ [?]	87.8
DeepLabv3-JFT [?]	86.9
DIS [?]	86.8
CASIA_IVA_SDN [?]	86.6
IDW-CNN [?]	86.3
DFN [?]	86.2
EncNet [?]	85.9
HPN [?]	85.8
DeepLabv3 [?]	85.7
PSPNet [?]	85.4
** ResNet-38_COCO ** [?]	84.9
Multipath-RefineNet [?]	84.2
FDNet_16s [?]	84.0
Large_Kernel_Matters [?]	83.6
TKCNet [?]	83.2
ResNet-38_MS [?]	83.1
ResNet_DUC_HDC [?]	83.1
Deep Layer Cascade (LC) [?]	82.7
SegModel [?]	81.8
HikSeg_COCO [?]	81.4
DP_ResNet_CRF [?]	81.0
MSRSegNet-UW [?]	81.0
OBP-HJLCN [?]	80.4
CentraleSupelec Deep G-CRF [?]	80.2
** -- ** [?]	80.1
CMT-FCN-ResNet-CRF [?]	80.0
DeepLabv2-CRF [?]	79.7

Evaluation metrics

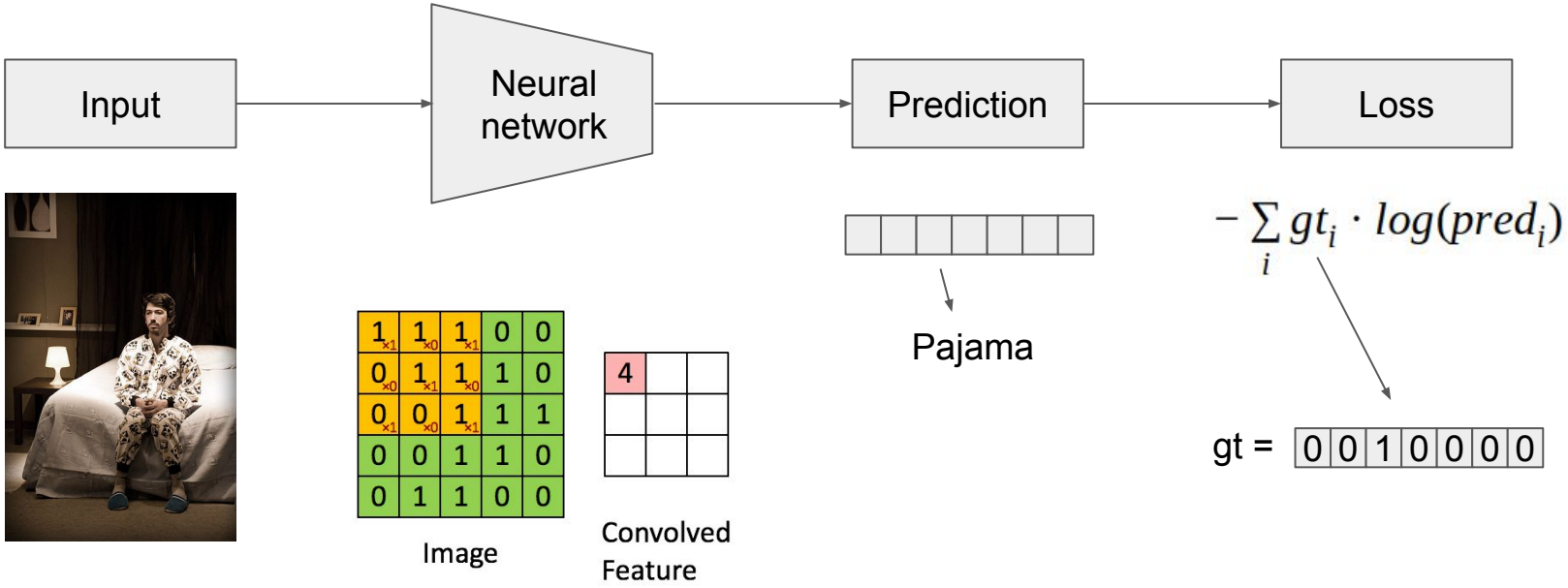


C. Lawrence Zitnick, P. Dollár. Edge Boxes : Locating Object Proposals from Edges. 2014

- Pixel accuracy (dominated by background class)
- Mean accuracy over classes
- Jaccard index = Intersection over Union (IoU) = (GT ∩ Pred) / (GT U Pred)
  - = TP / (TP + FN + FP)
  - Usually: mean over classes on the whole dataset
  - Can be weighted by inverse instance size (Cityscapes, important in traffic use cases)
- Dice index = F1 score = 2(GT ∩ Pred) / (GT + Pred)
  - = 2TP / (2TP + FN + FP)
  - = 2IoU / (1 + IoU)
- [Adjusted] Rand Index (RI) / Rand Error (RE)
  - $RI = (a + b)/ C_n^2,\ RE = 1 - RI$
  - a - the # of pairs that have the **same labels** in both prediction and GT
  - b - the # of pairs that have the **different labels** in both prediction and GT

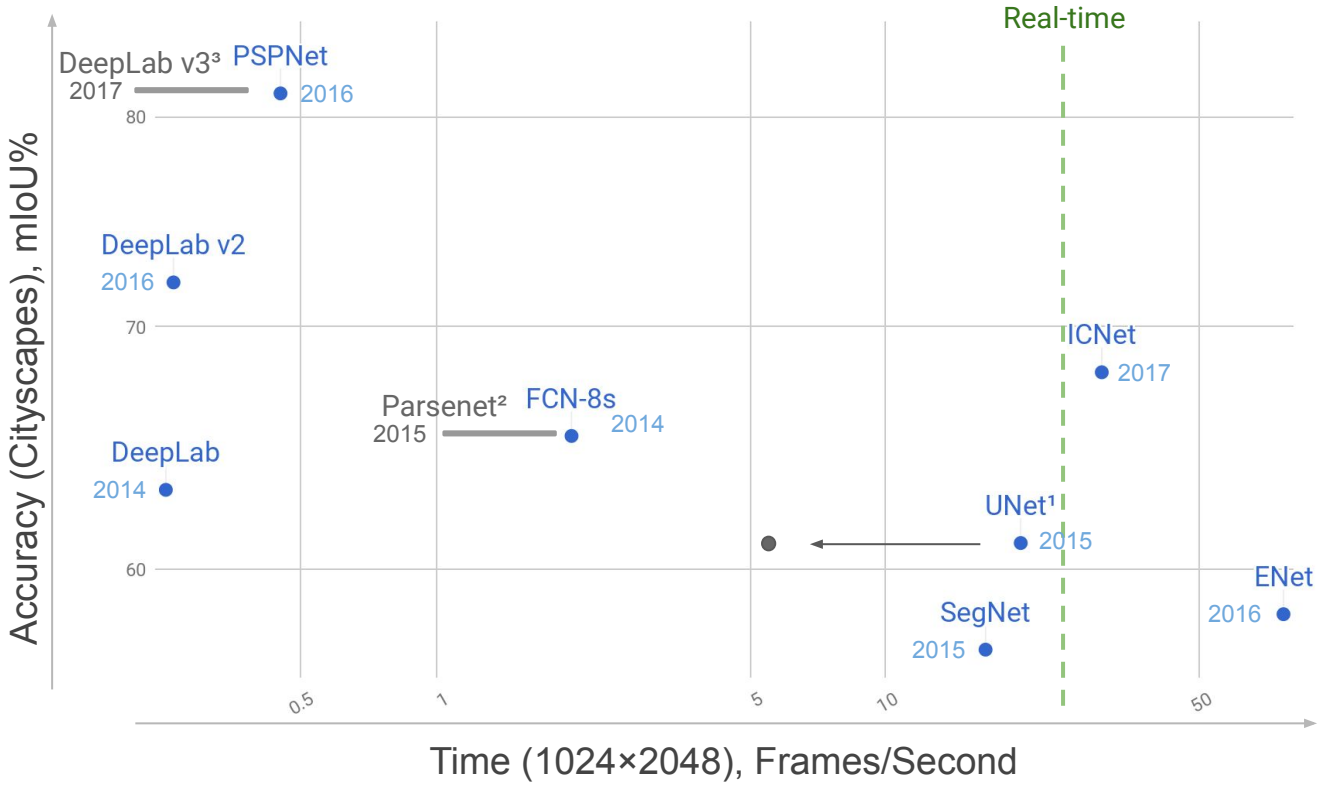


Neural networks (classification)



[3×3 convolution filter]

Semantic segmentation architectures (1)



Hardware: NVidia Titan X

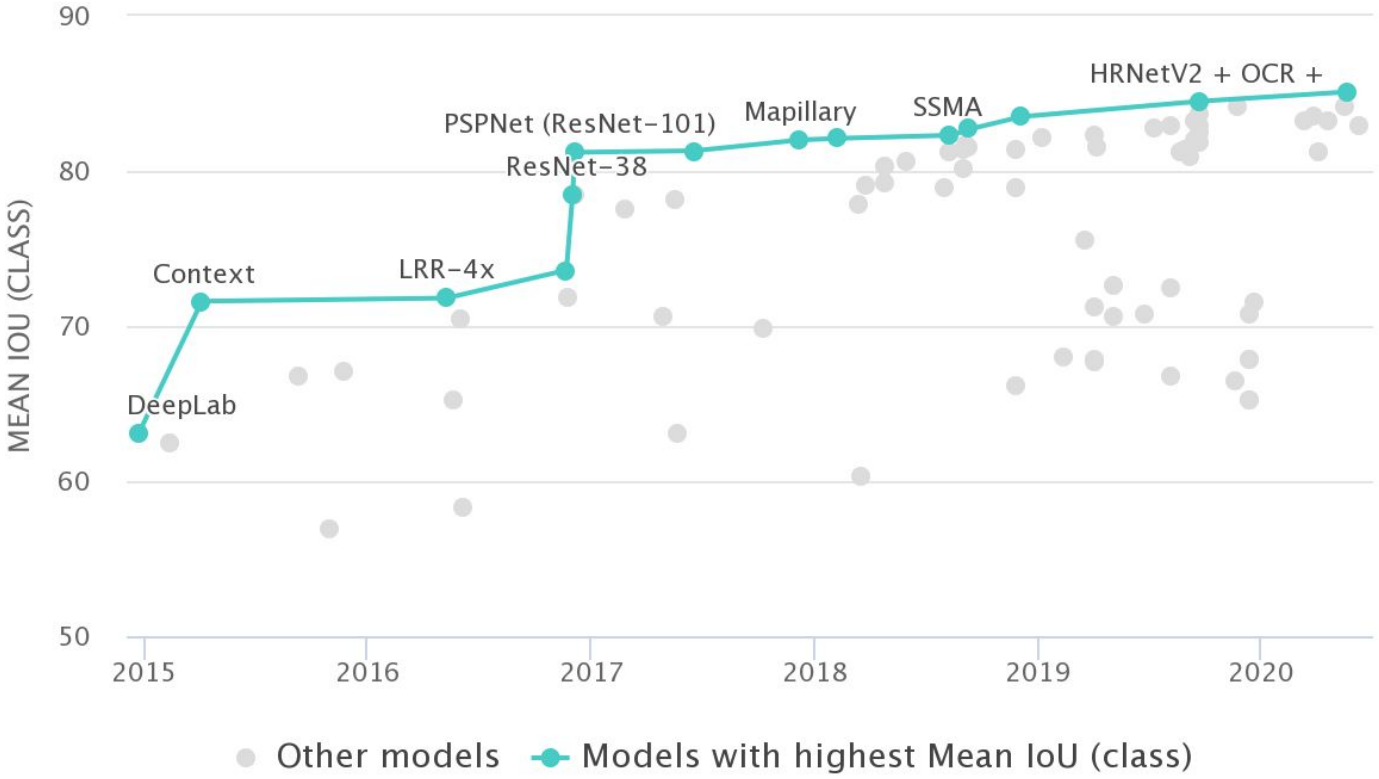
UNet¹ - 512×1024 frames (https://arxiv.org/pdf/1803.02758.pdf)

Parsenet² - no FPS results (http://ais.informatik.uni-freiburg.de/publications/papers/valada17icra.pdf)

DeepLab v3³ - no FPS results

[1704.08545] ICNet for Real-Time Semantic Segmentation on High-Resolution Images

Semantic segmentation architectures (2)



Papers with Code - Cityscapes test Benchmark (Semantic Segmentation)

FCN: Fully Convolutional Networks for Semantic Segmentation

- The fully connected layers can also be viewed as convolutions with kernels that cover their entire input regions
- The spatial output maps of these convolutionalized models make them a natural choice for dense problems like semantic segmentation.

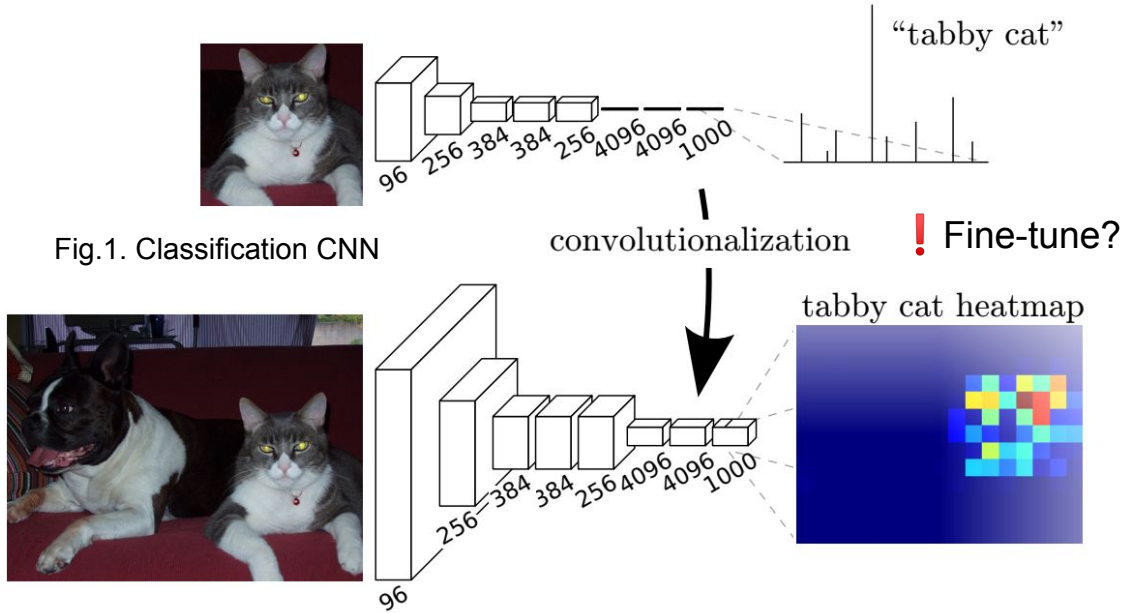


Fig.1. Classification CNN

Fig.2. FCN

[1411.4038] Fully Convolutional Networks for Semantic Segmentation

FCN: Architecture

Combines coarse, high layer information with fine, low layer information

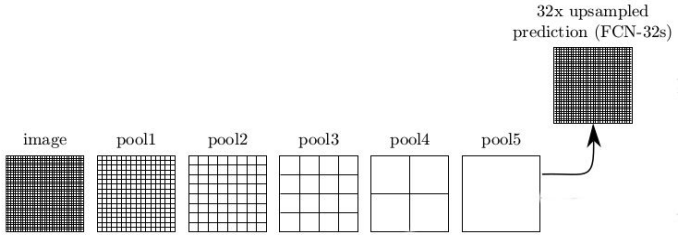


Fig.1. FCN-32s

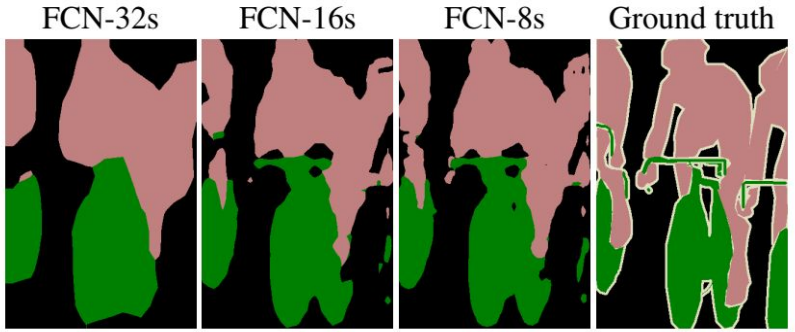


Fig.3. FCN results

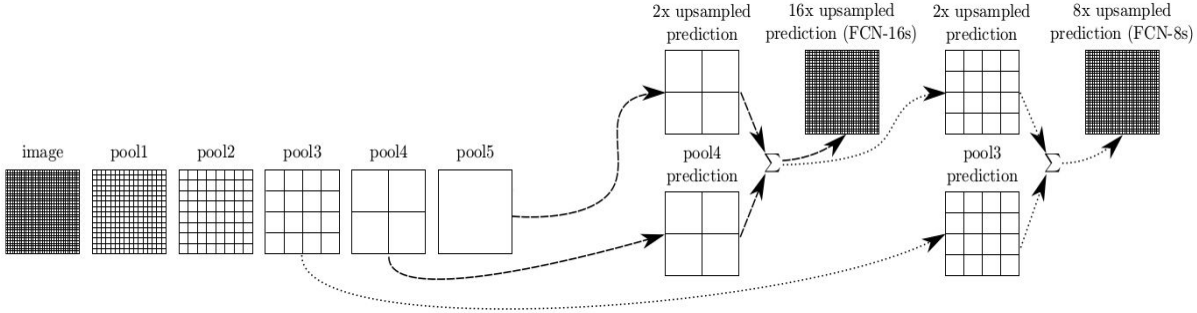
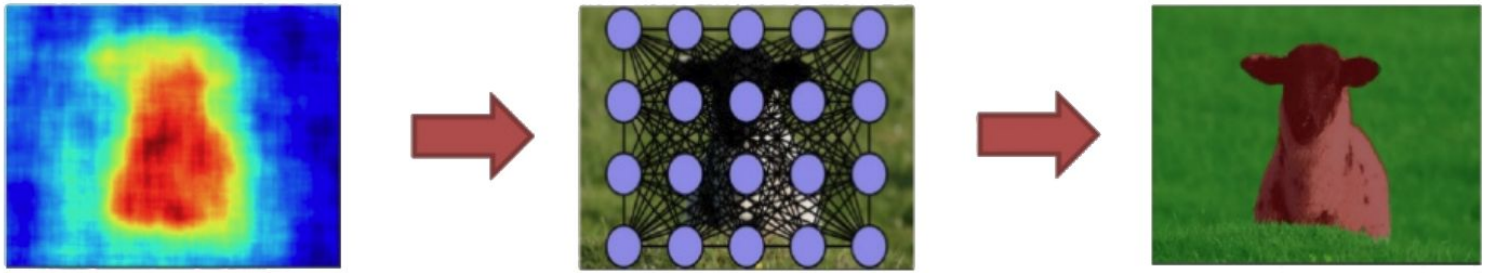


Fig.2. FCN-16s and FCN-8s

[1411.4038] Fully Convolutional Networks for Semantic Segmentation

CRF: Conditional Random Field (1)

- FCNs classify each pixel in segmentation map independently.
- Probabilistic graphical models, such as Conditional Random Fields (CRFs) have been used extensively in prior literature to predict structures and incorporate prior knowledge.



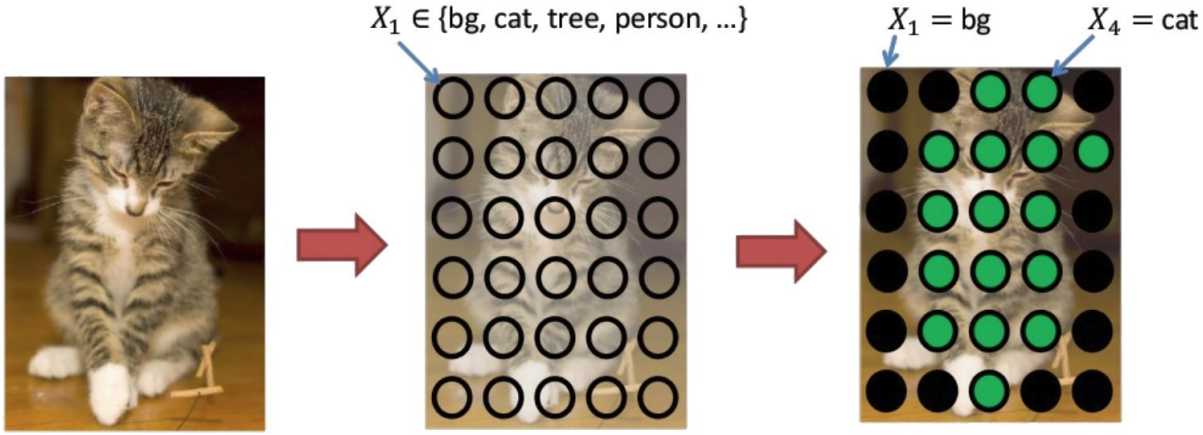
Coarse output from pixel-wise classifier

CRF modeling

Output after CRF inference

[1210.5644] Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials

CRF: Conditional Random Field (2)



- Define a discrete random variable, X_i, for each pixel i.
- Each X_i takes a value from the label set L.
- The random variables are connected to form a random field. The most probable assignment, conditioned on the image, is our semantic segmentation result.

[1210.5644] Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials

CRF: Conditional Random Field Details (1)

Let $\mathbf{I} = [I_1, ..., I_N]$ be an array of image pixels' color vectors.
Let $\mathbf{X} = [X_1, ..., X_N]$ be an array of pixels' labels.
Conditional random field (**I**,**X**) is characterized by a Gibbs distribution:

$$P(\mathbf{X}|\mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp\left(-\sum_{c \in C_G} \phi_c(\mathbf{X}_c|\mathbf{I})\right)$$

where *G* is a graph on **X**, C_G is a set of cliques in that graph. Each clique induces a potential ϕ_c.
Then MAP labeling is

$$x^* = \arg\max_{x \in L^N} P(x|\mathbf{I}) = \arg\min_{x \in L^N} \sum_{c \in C_G} \phi_c(\mathbf{X}_c|\mathbf{I}) \equiv \arg\min_{x \in L^N} E(x|\mathbf{I})$$

[1210.5644] Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials

CRF: Conditional Random Field Details (2)

In the fully connected pairwise CRF model G is the full graph on **X** andC_G is the set of all unary and pairwise cliques.

$$E(x) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j)$$

ψ_u is the output of a pixel classifier. When classifier is applied to different pixels independently, MAP labeling produced by this term alone is noisy and inconsistent.

[1210.5644] Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials

CRF: Conditional Random Field Details (3)

$$\psi_p(x_i, x_j) = \mu(x_i, x_j) \sum_{m=1}^{K} w^{(m)} k^{(m)}(f_i, f_j) \equiv \mu(x_i, x_j) k(f_i, f_j)$$

where *k* is a Gaussian kernel, *f* is a feature vector describing corresponding pixel. Contrast-sensitive two-kernel potentials that depends on pixel intensities (*I*) and positions (*p*) is used:

$$k(f_i, f_j) = w^{(1)} \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2}\right) + w^{(2)} \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2}\right)$$

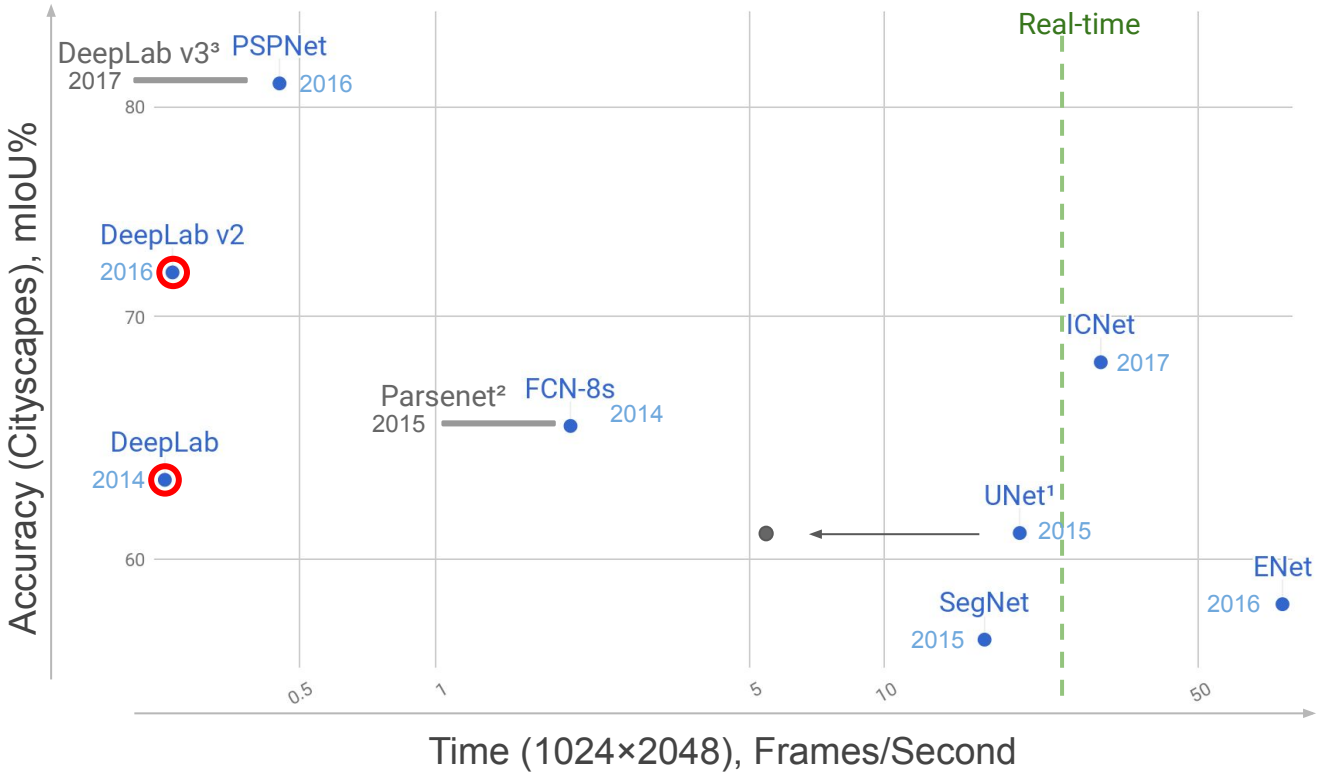
First term is appearance kernel (nearby pixels with same color are likely to be in the same class), second -- smoothness (penalizes small isolated regions).

$$\mu(x_i, x_j) = \mathbb{I}(x_i \neq x_j)$$

penalizes for nearby similar pixels that are assigned different labels.

[1210.5644] Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials

DeepLab v1 / DeepLab v2



DeepLab v1

Contributions:

- Brings together DL methods and probabilistic graphical models.
- First to apply dilated/atrous convolutions in deep learning.
- No decoder -- CRF as a refinement model.
- Sets SOTA on Pascal VOC (71.6 mIOU test).

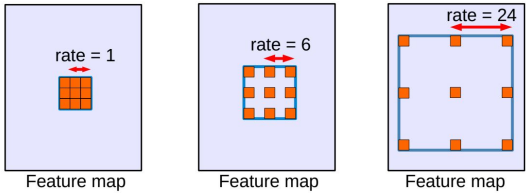


Fig.1. Dilated convolutions.

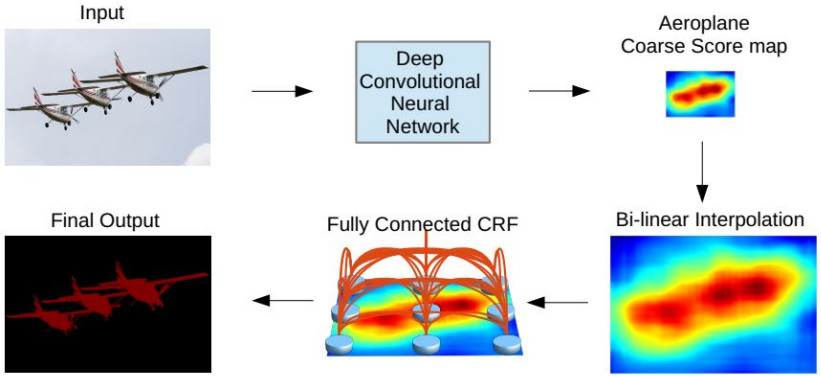


Fig.2. Pipeline.

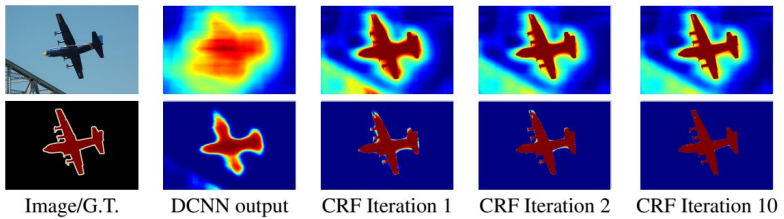


Fig.3. Results.

[1412.7062] Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs

DeepLab v1 (Details)

Method outline:

- Use VGG as a backbone.
- Replace two last poolings/strided convs with dilated convs.
- Reduce last conv's spatial and channel size to make model lighter + add dilations (LargeFOV).
- Use a hyper feature-like design: features from intermediate layers are passed directly to the output feature-map (MSc).
- Bilinearly upsample resulting segmentation and apply CRF for refinement.

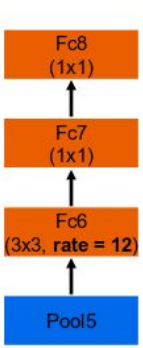
Method	mean
MSRA-CFM	61.8
FCN-8s	62.2
TTI-Zoomout-16	64.4
DeepLab-CRF	66.4
DeepLab-MSc-CRF	67.1
DeepLab-CRF-7x7	70.3
DeepLab-CRF-LargeFOV	70.3
DeepLab-MSc-CRF-LargeFOV	**71.6**

[1412.7062] Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs

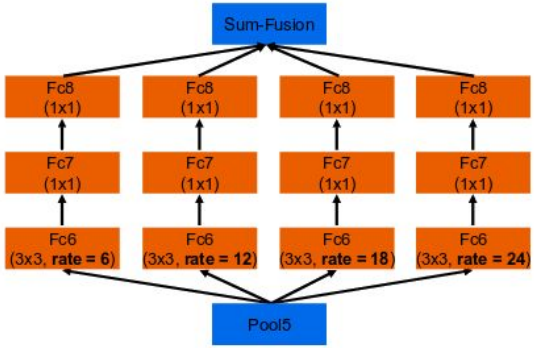
DeepLab v2

Contributions:

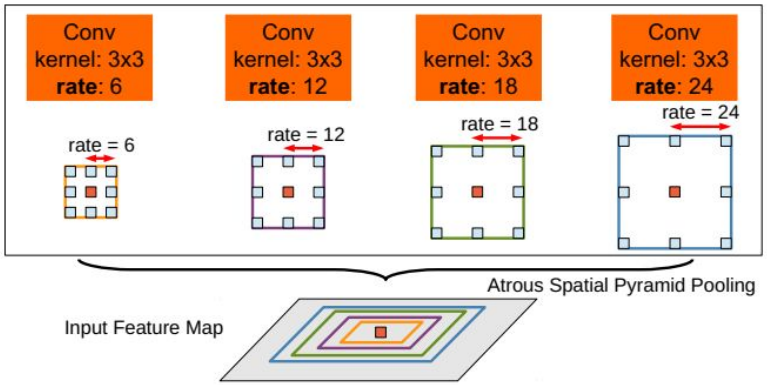
- Replace VGG with ResNet.
- Propose atrous spatial pyramid pooling (ASPP) to robustly do multiscale segmentation.
- Provide SOTA results on Pascal VOC (79.7 mIOU test with bells and whistles) and Citiscapes.



DeepLab v1



DeepLab v2: ASPP



ASPP kernels

[1606.00915] DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs

DeepLab v2 (Details)

Method outline:

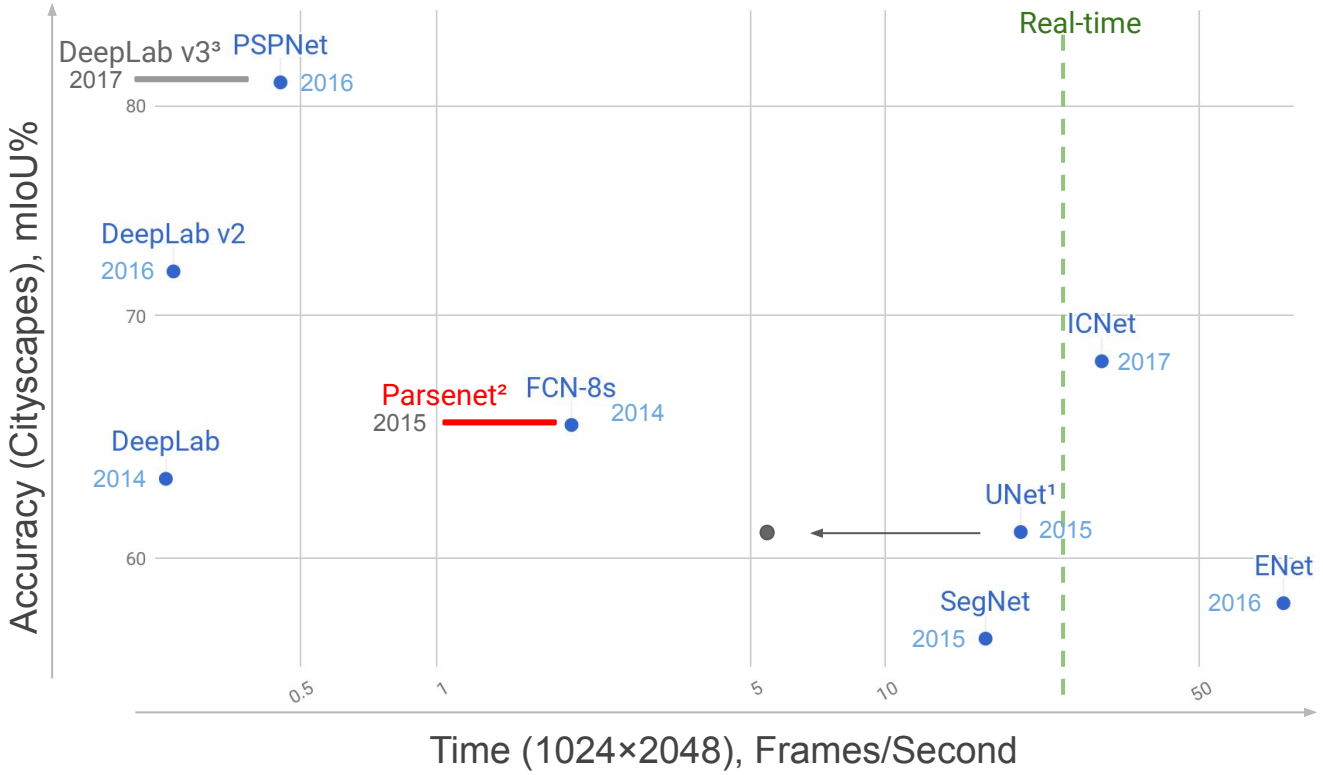
- Use ResNet-101 as backbone.
- Replace the last part of the net by ASPP block.
- Aggregate segmentation from three scales of original image.
- Pretrain on COCO.
- Random scaling augmentation.

MSC	COCO	Aug	LargeFOV	ASPP	CRF	mIOU
						68.72
✓						71.27
✓	✓					73.28
✓	✓	✓				74.87
✓	✓	✓	✓			75.54
✓	✓	✓		✓		76.35
✓	✓	✓		✓	✓	77.69

TABLE 4: Employing ResNet-101 for DeepLab on PASCAL VOC 2012 *val* set. **MSC**: Employing mutli-scale inputs with max fusion. **COCO**: Models pretrained on MS-COCO. **Aug**: Data augmentation by randomly rescaling inputs.

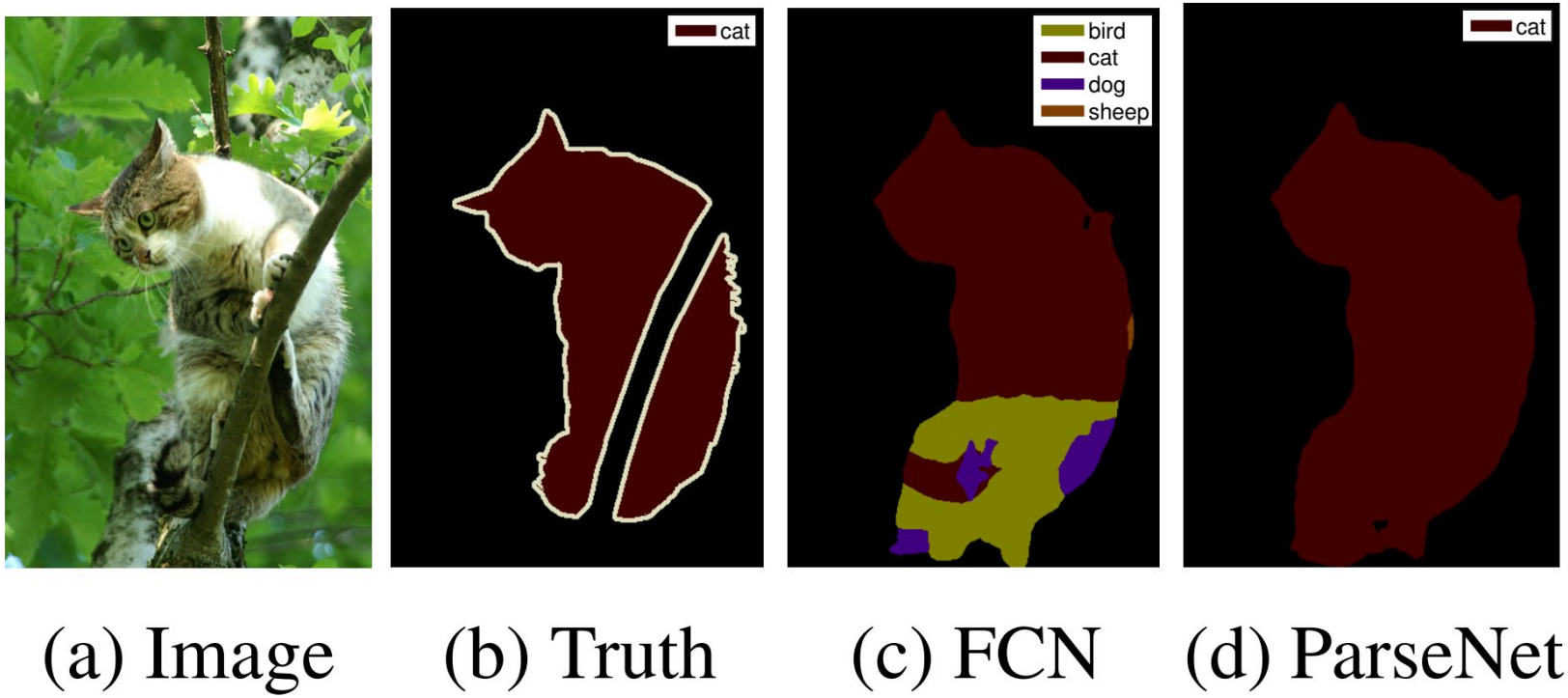
[1606.00915] DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs

Parsenet: Looking wider to see better



[1506.04579] ParseNet: Looking Wider to See Better

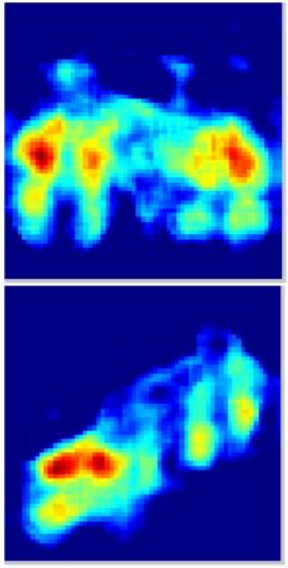
Parsenet: Global context



[1506.04579] ParseNet: Looking Wider to See Better

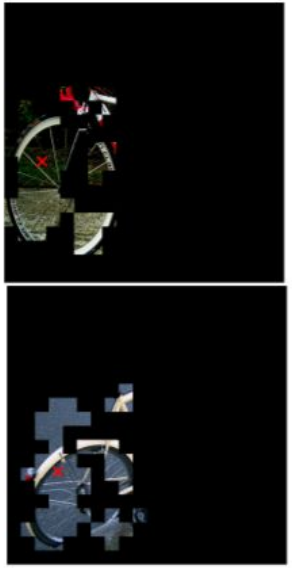
Parsenet: Theoretical vs Empirical receptive field

(a) Original image



(b) Activation map

(c) Theoretical Receptive Field



(d) Empirical Receptive Field

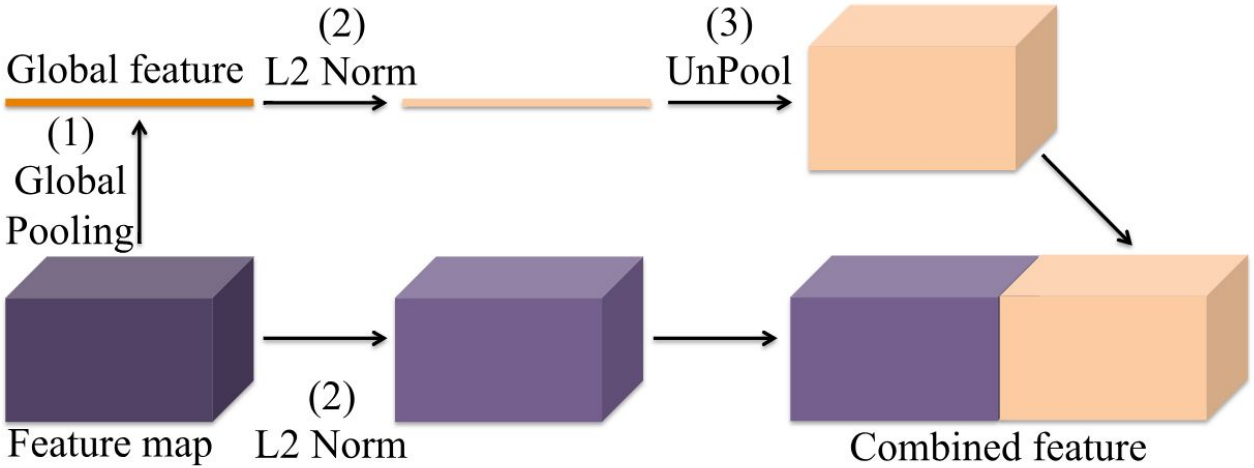
To evaluate “empirical” receptive field of a neuron, authors propose the following procedure:

- walk through image with a sliding window of small random noise,
- if the activation doesn’t change beyond a certain threshold - it means the window is outside the “empirical” RF.

Experiment results show that the network tends to simply “learn patches” (but not context).

[1506.04579] ParseNet: Looking Wider to See Better

Parsenet: Architecture



1. **Global average pooling** from the last feature map (pooling from other layers is possible, if necessary).

2. **L2 Normalization** normalize each individual feature first, and also learn to scale each differently, it makes the training more stable and improves performance.

3. **UnPooling** - replication of the feature vector until it has the corresponding size.

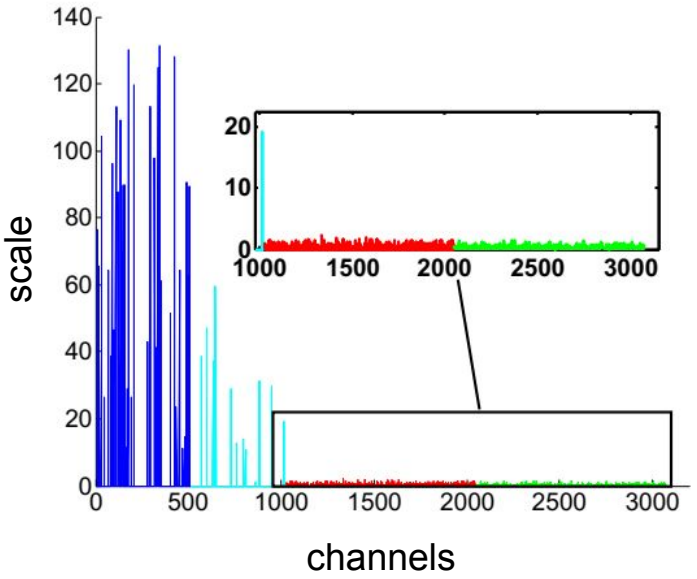
[1506.04579] ParseNet: Looking Wider to See Better

Parsenet: Combining Local and Global features

Results on PASCAL-Context

model \ accuracy	mean IoU	
	w/o Norm	w/ Norm
FCN-32s	36.6	36.2
FCN-32s + global context	38.2	37.6
FCN-16s + global context	39.5	39.9
FCN-8s + global context	36.5	40.2
FCN-4s + global context	0.009	**40.4**

Features from 4 different layers have different scales: conv4, conv5, fc7, pool6



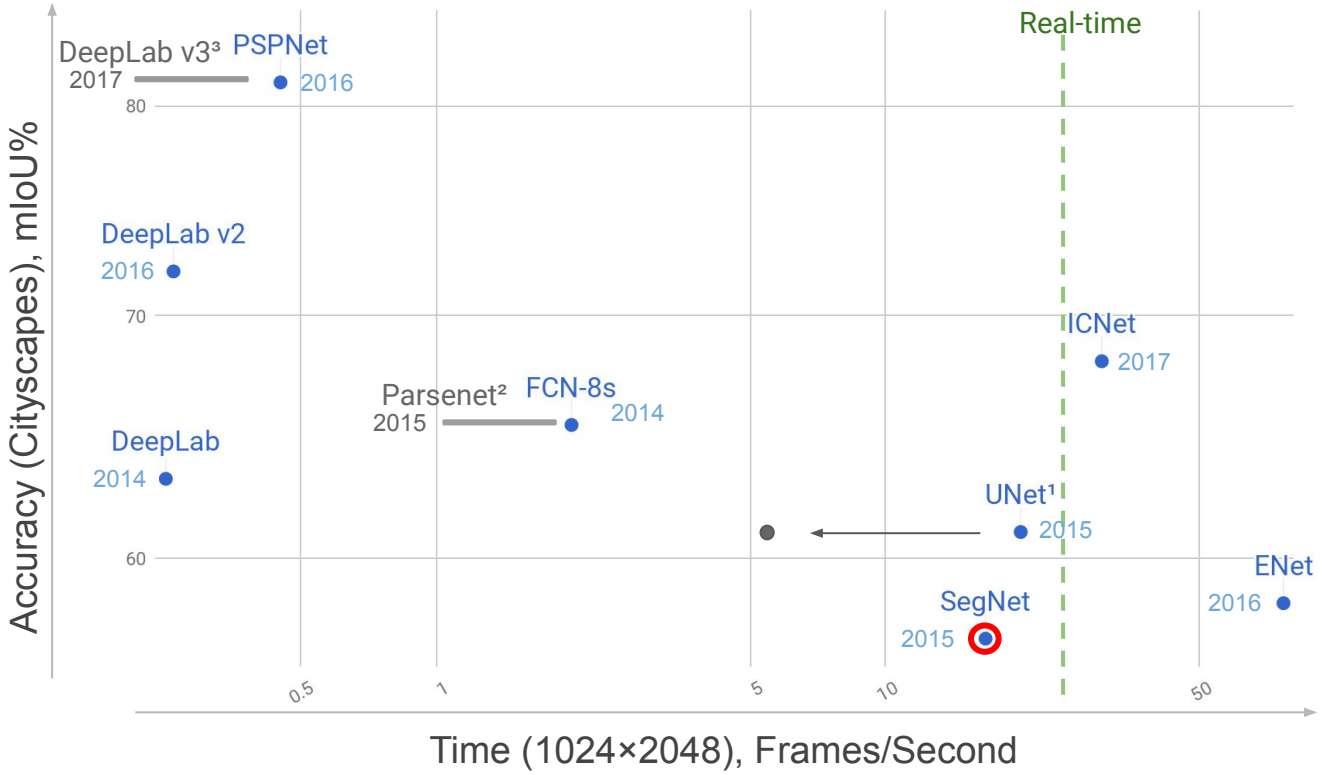
[1506.04579] ParseNet: Looking Wider to See Better

How to increase spatial dimension?

How to increase spatial dimension?

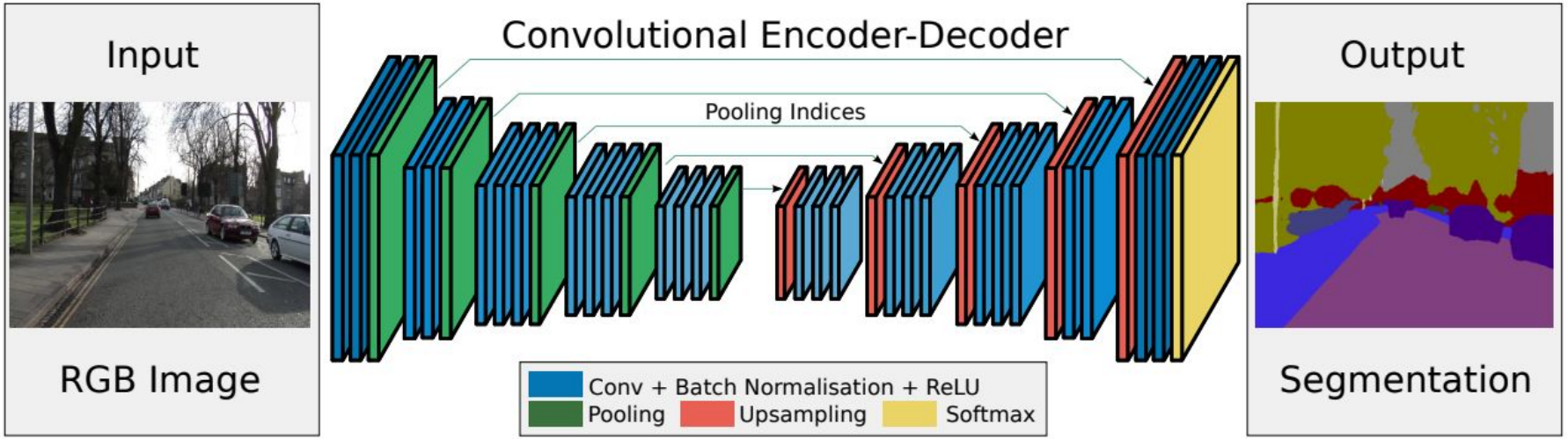
1. Copy.
2. Bi-linear interpolation.
3. Fill with zeros.
4. Max unpooling (**SegNet**).
5. Transposed convolution.

SegNet: A Deep Convolutional Encoder-Decoder



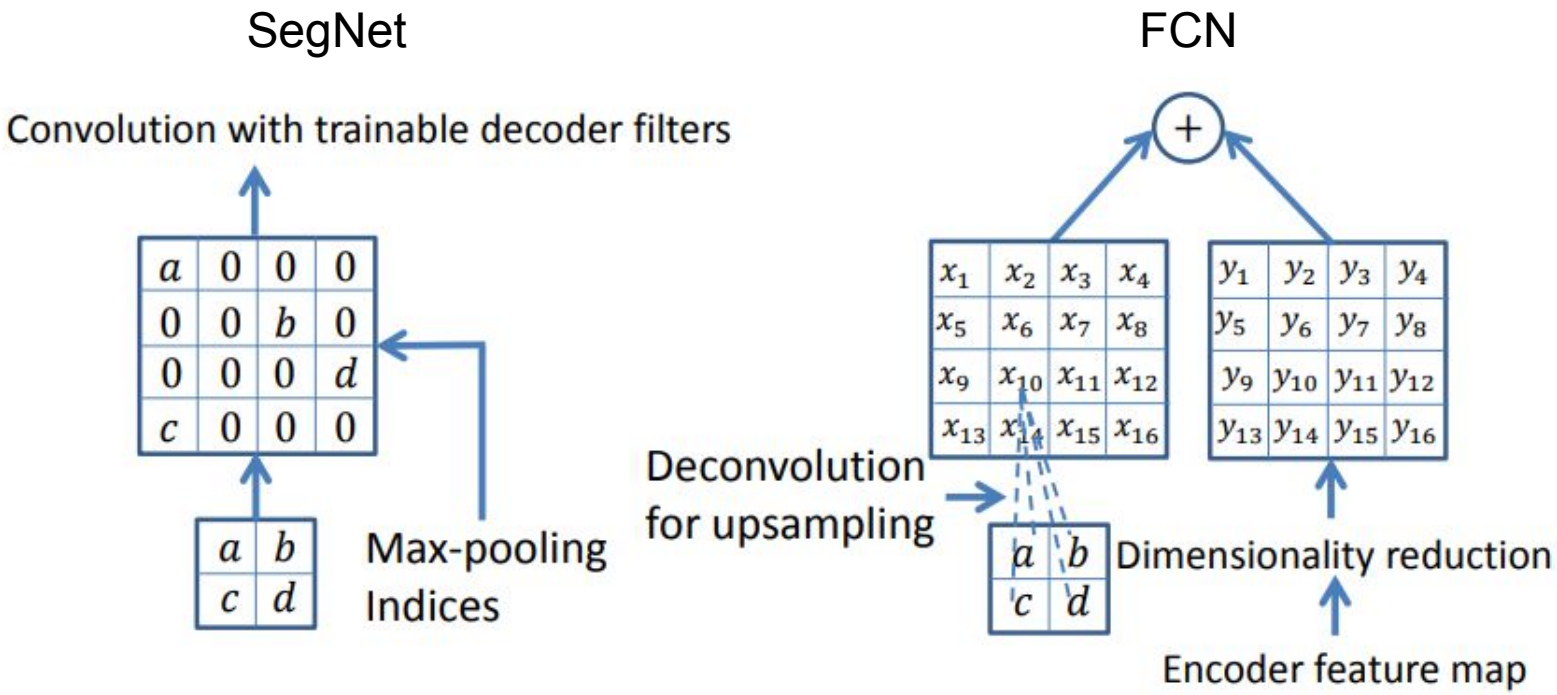
[1511.00561] SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation



[1511.00561] SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

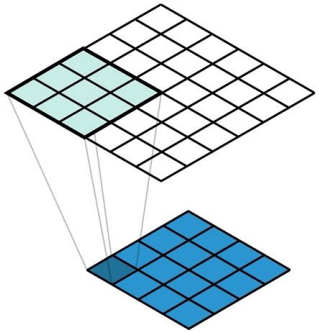
SegNet: Fast decoder



[1511.00561] SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

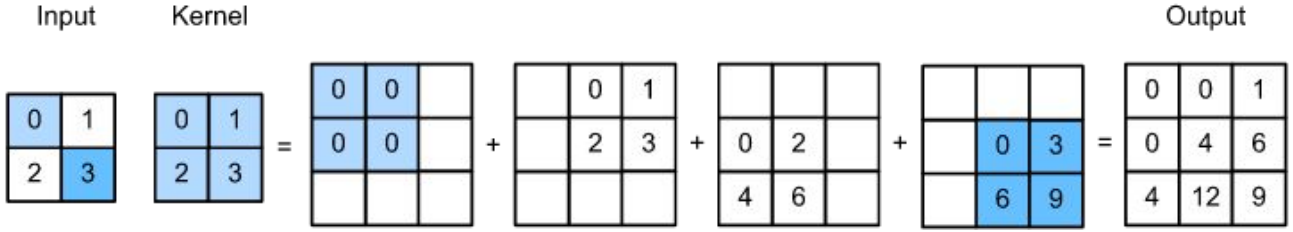
Transposed convolution (1)

Up-sampling with transposed convolution



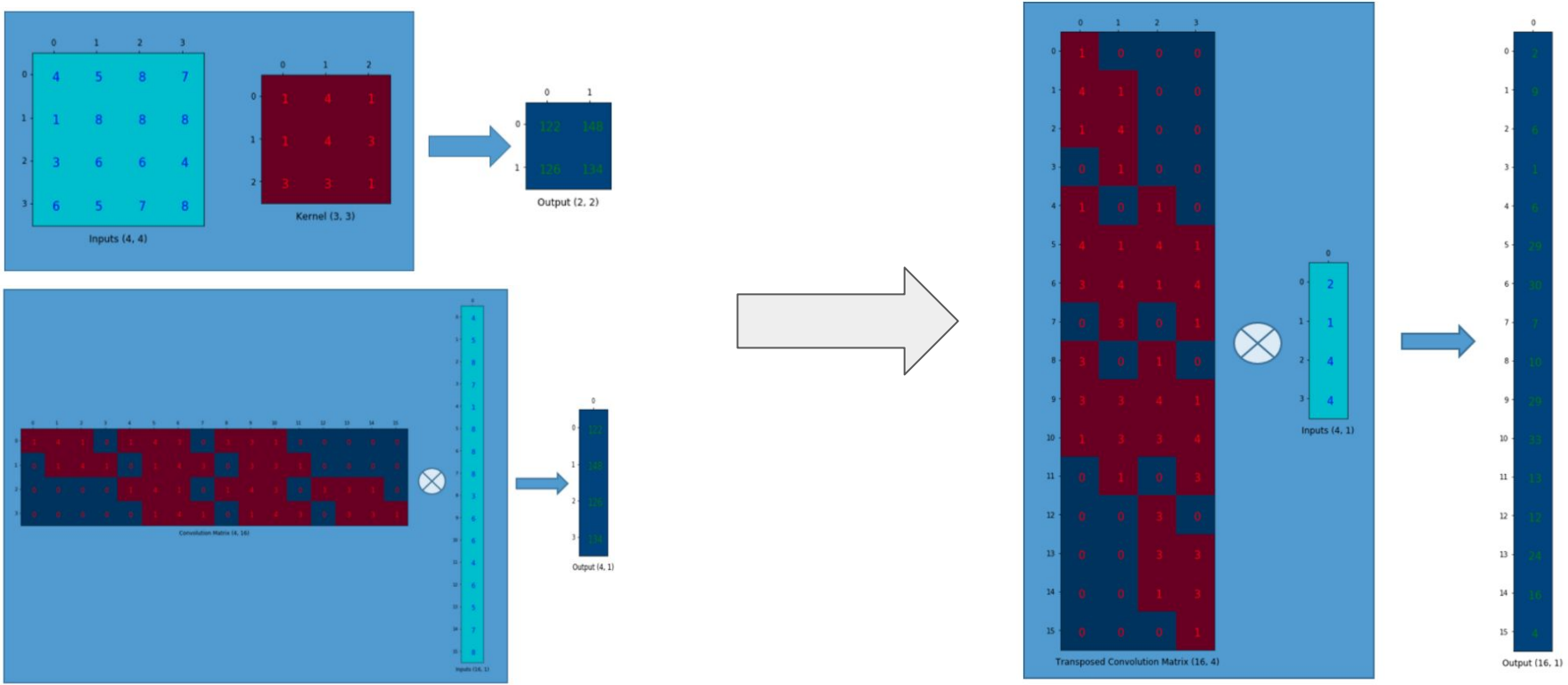
kernel - 3x3, padding - 0, stride - 1

https://medium.com/apache-mxnet/transposed-convolutions-explained-with-ms-excel-52d13030c7e8



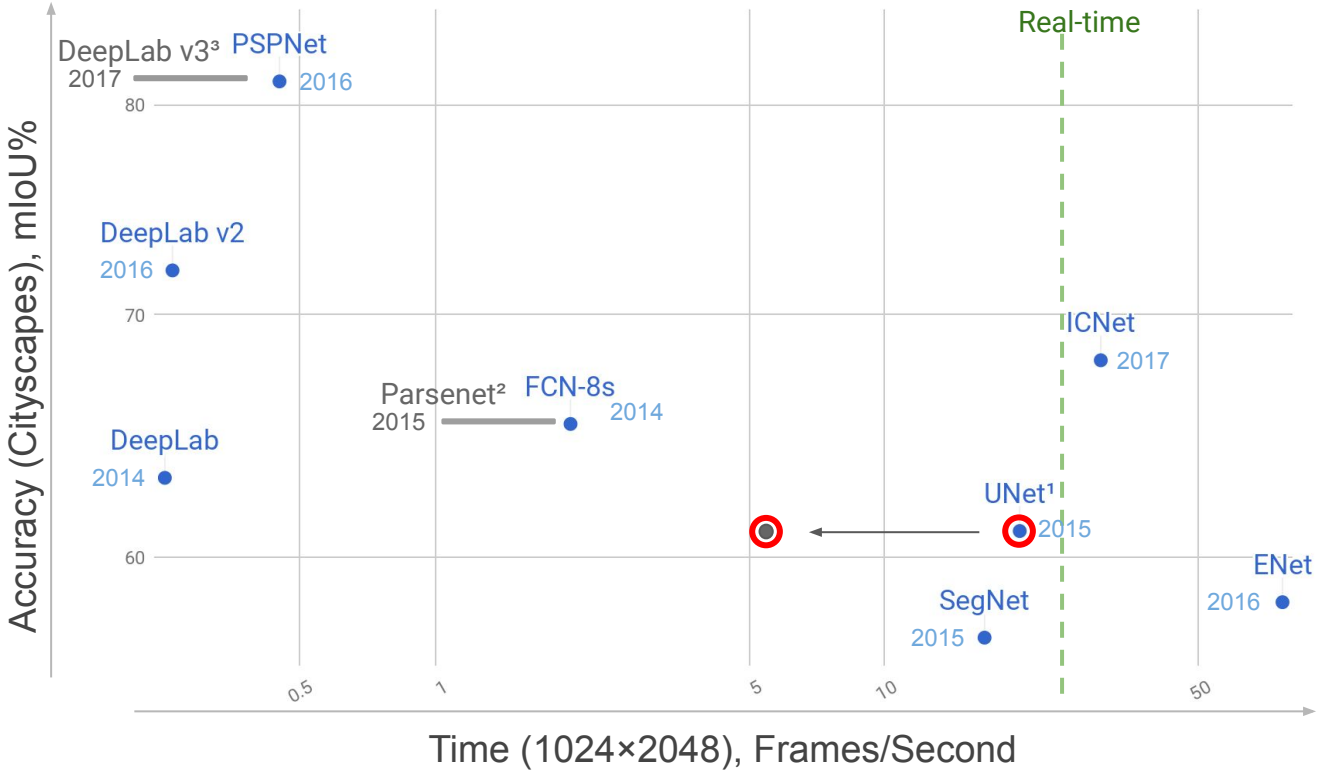
http://d2l.ai/chapter_computer-vision/transposed-conv.html

Transposed convolution (2)



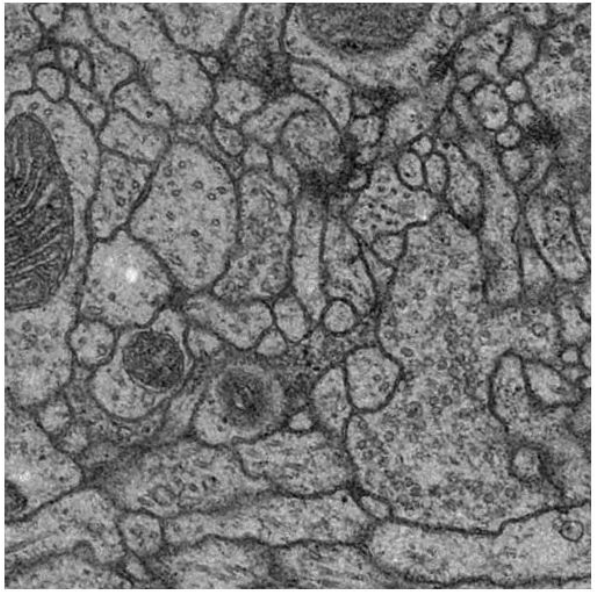
https://naokishibuya.medium.com/up-sampling-with-transposed-convolution-9ae4f2df52d0

U-Net: Convolutional Networks for Biomedical Image Segmentation

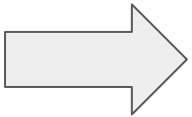


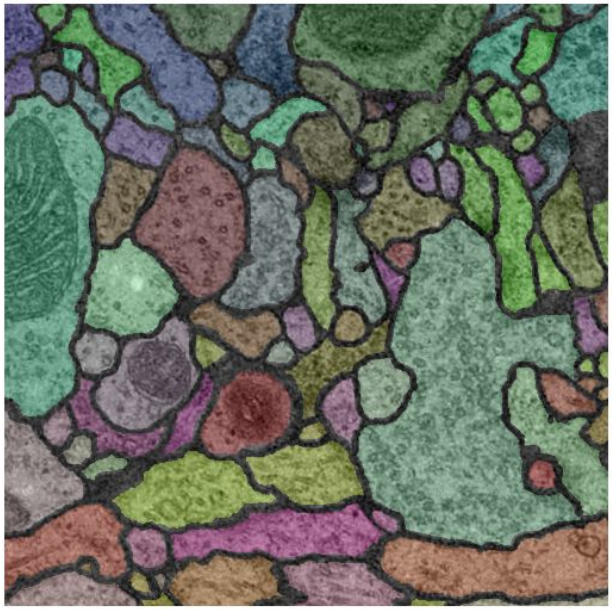
[1505.04597] U-Net: Convolutional Networks for Biomedical Image Segmentation

U-Net: Biomedical Image Segmentation



Drosophila first instar larva
ventral nerve cord



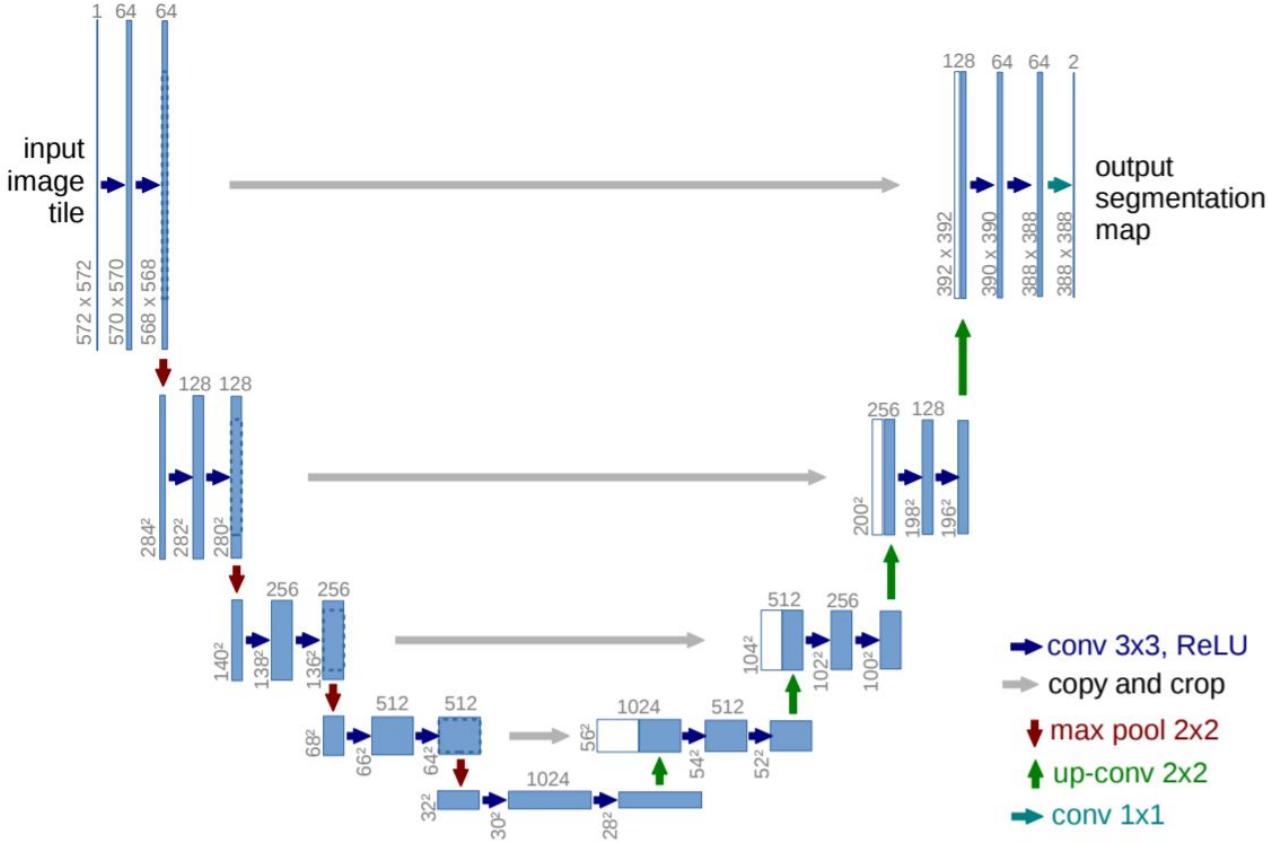


Ground truth segmentation
ISBI Challenge: Segmentation of neuronal structures in EM stacks

[1505.04597] U-Net: Convolutional Networks for Biomedical Image Segmentation

U-Net: Architecture

- **U**-Net
- Fully convolutional network
- Encoder topology with skip connections



[1505.04597] U-Net: Convolutional Networks for Biomedical Image Segmentation

U-Net: Training

Loss: per-pixel softmax + cross-entropy with weighting
(compensate class frequency and emphasize edges)

$$E = \sum_{\mathbf{x} \in \Omega} w(\mathbf{x}) \log(p_{\ell(\mathbf{x})}(\mathbf{x}))$$

$$w(\mathbf{x}) = w_c(\mathbf{x}) + w_0 \cdot \exp\left(-\frac{(d_1(\mathbf{x}) + d_2(\mathbf{x}))^2}{2\sigma^2}\right)$$

$l: \Omega \rightarrow \{1, \ldots, K\}$ – is the true label of each pixel

w_c – precompute weight map

d_1, d_2 – distance to the border of the two nearest cells

$w_0 = 10$

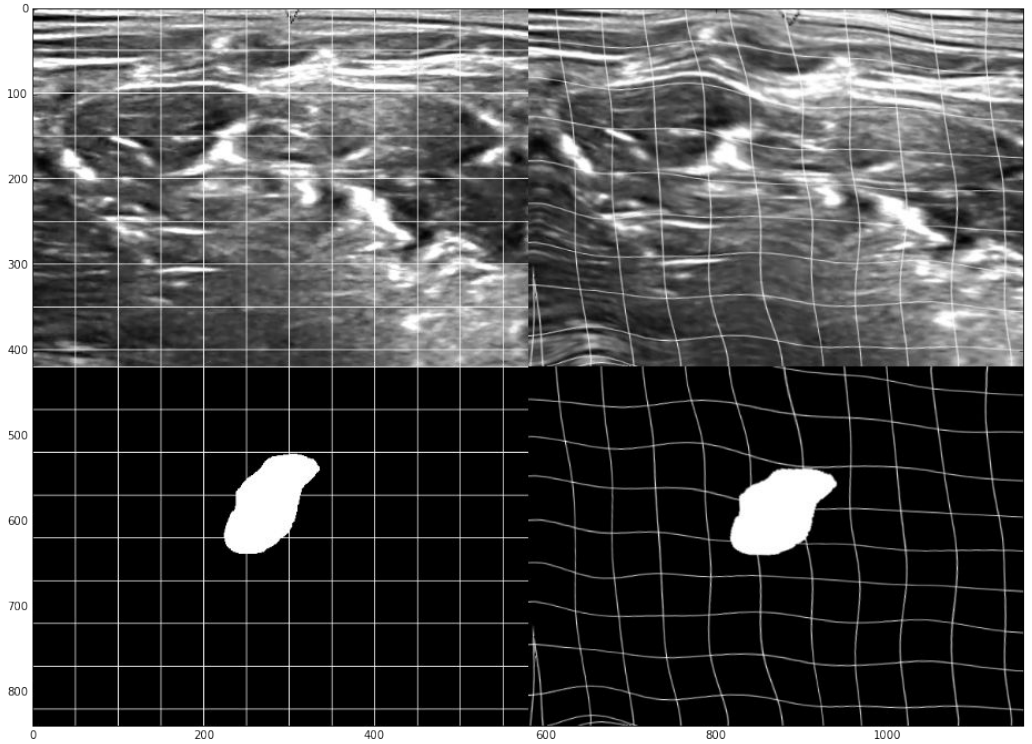
$\sigma \approx 5$ pixels

[1505.04597] U-Net: Convolutional Networks for Biomedical Image Segmentation

U-Net: Augmentations

- **General:** smooth deformations.
- **Additional**: shifting, rotating, gray value variations.

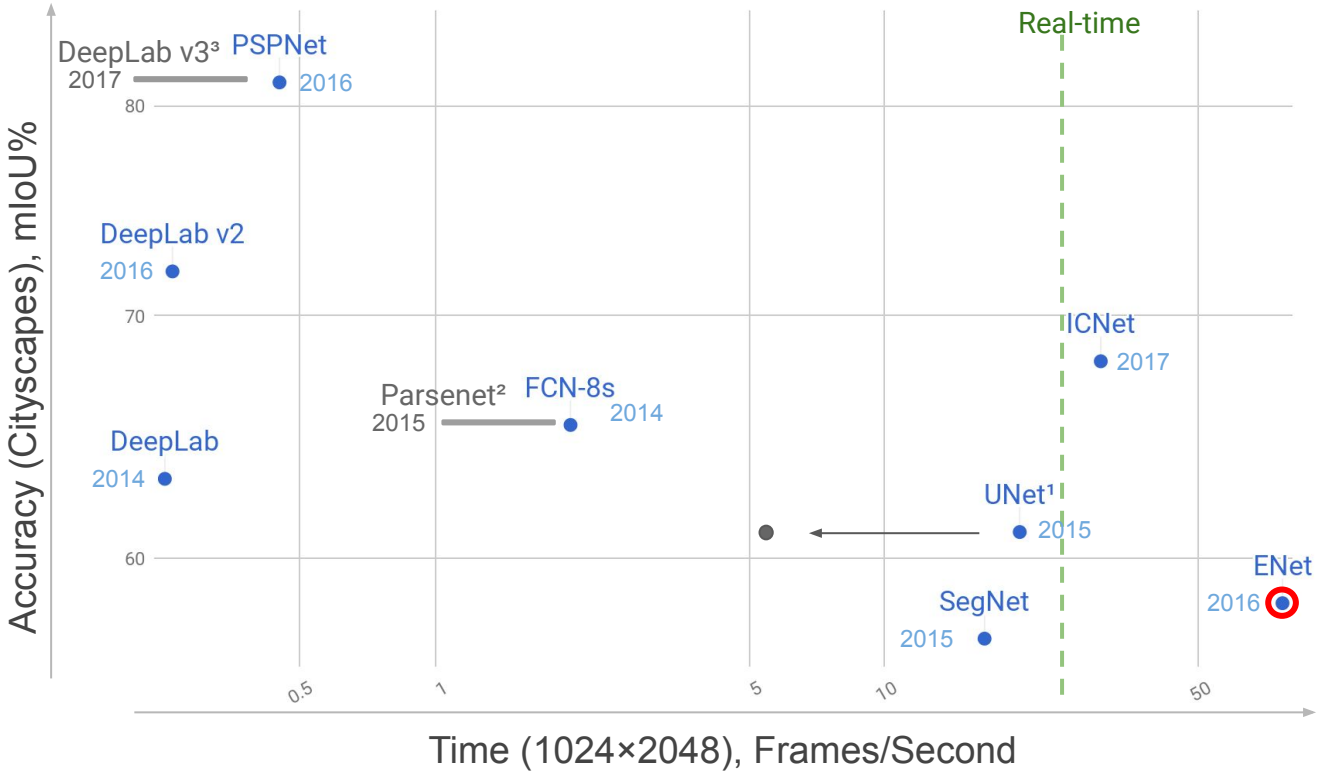
Example of smooth deformation



https://www.slideshare.net/Eduardyantov/ultrasound-segmentation-kaggle-review

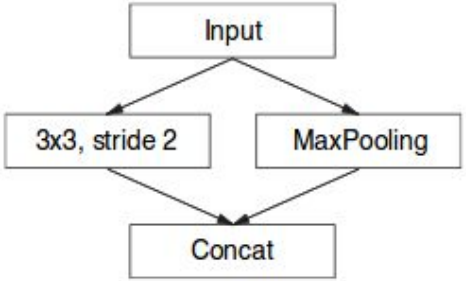
[1505.04597] U-Net: Convolutional Networks for Biomedical Image Segmentation

ENet: Real-Time Semantic Segmentation

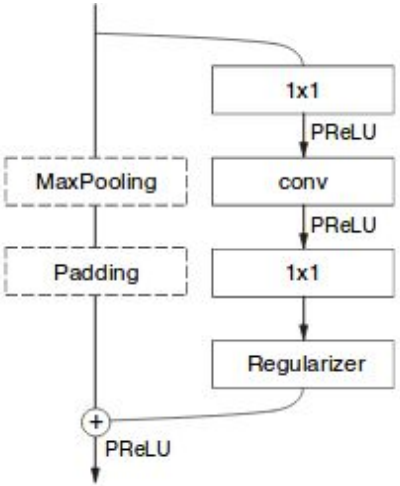


[1511.00561] SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation



Initial block



Bottleneck

	Name	Type	Output size
Encoder	initial		$16 \times 256 \times 256$
	bottleneck1.0	downsampling	$64 \times 128 \times 128$
	4× bottleneck1.x		$64 \times 128 \times 128$
	bottleneck2.0	downsampling	$128 \times 64 \times 64$
	bottleneck2.1		$128 \times 64 \times 64$
	bottleneck2.2	dilated 2	$128 \times 64 \times 64$
	bottleneck2.3	asymmetric 5	$128 \times 64 \times 64$
	bottleneck2.4	dilated 4	$128 \times 64 \times 64$
	bottleneck2.5		$128 \times 64 \times 64$
	bottleneck2.6	dilated 8	$128 \times 64 \times 64$
	bottleneck2.7	asymmetric 5	$128 \times 64 \times 64$
	bottleneck2.8	dilated 16	$128 \times 64 \times 64$
	Repeat section 2, without bottleneck2.0		
Decoder	bottleneck4.0	upsampling	$64 \times 128 \times 128$
	bottleneck4.1		$64 \times 128 \times 128$
	bottleneck4.2		$64 \times 128 \times 128$
	bottleneck5.0	upsampling	$16 \times 256 \times 256$
	bottleneck5.1		$16 \times 256 \times 256$
	fullconv		$C \times 512 \times 512$

[1606.02147] ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation

ENet: Performance Analysis

Performance comparison

Model	NVIDIA TX1						NVIDIA Titan X					
	480x320		640x360		1280x720		640x360		1280x720		1920x1080	
	ms	fps	ms	fps	ms	fps	ms	fps	ms	fps	ms	fps
SegNet	757	1.3	1251	0.8	-	-	69	14.6	289	3.5	637	1.6
ENet	47	21.1	69	14.6	262	3.8	7	135.4	21	46.8	46	21.6

FLOPs are estimated for an input of 3x640x360

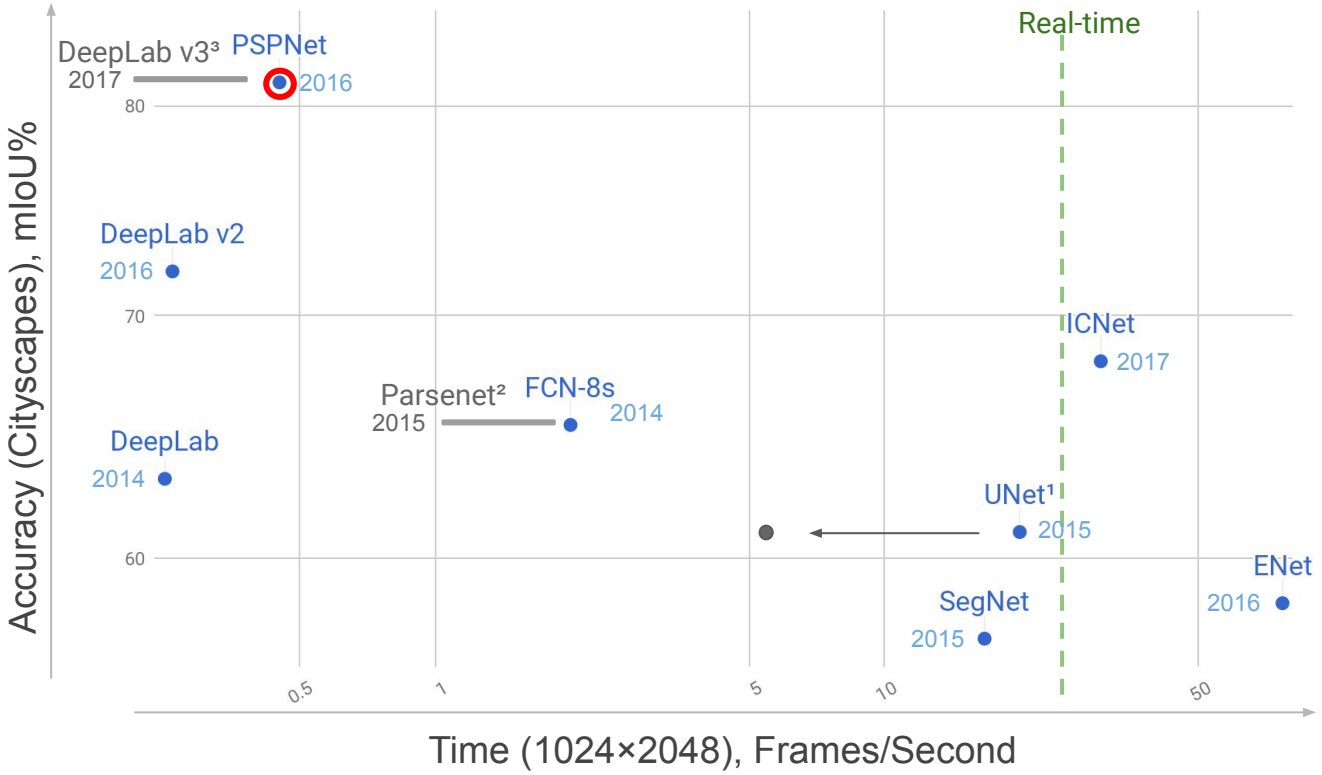
Model	GFLOPs	Parameters	Model size (fp16)
SegNet	286.03	29.46M	56.2 Mb
ENet	3.83	0.37M	0.7 Mb

Cityscapes test set result

Model	Class IoU	Class iIoU	Cat. IoU	Cat. iIoU
SegNet	56.1	34.2	79.8	66.4
ENet	58.3	34.4	80.4	64.0

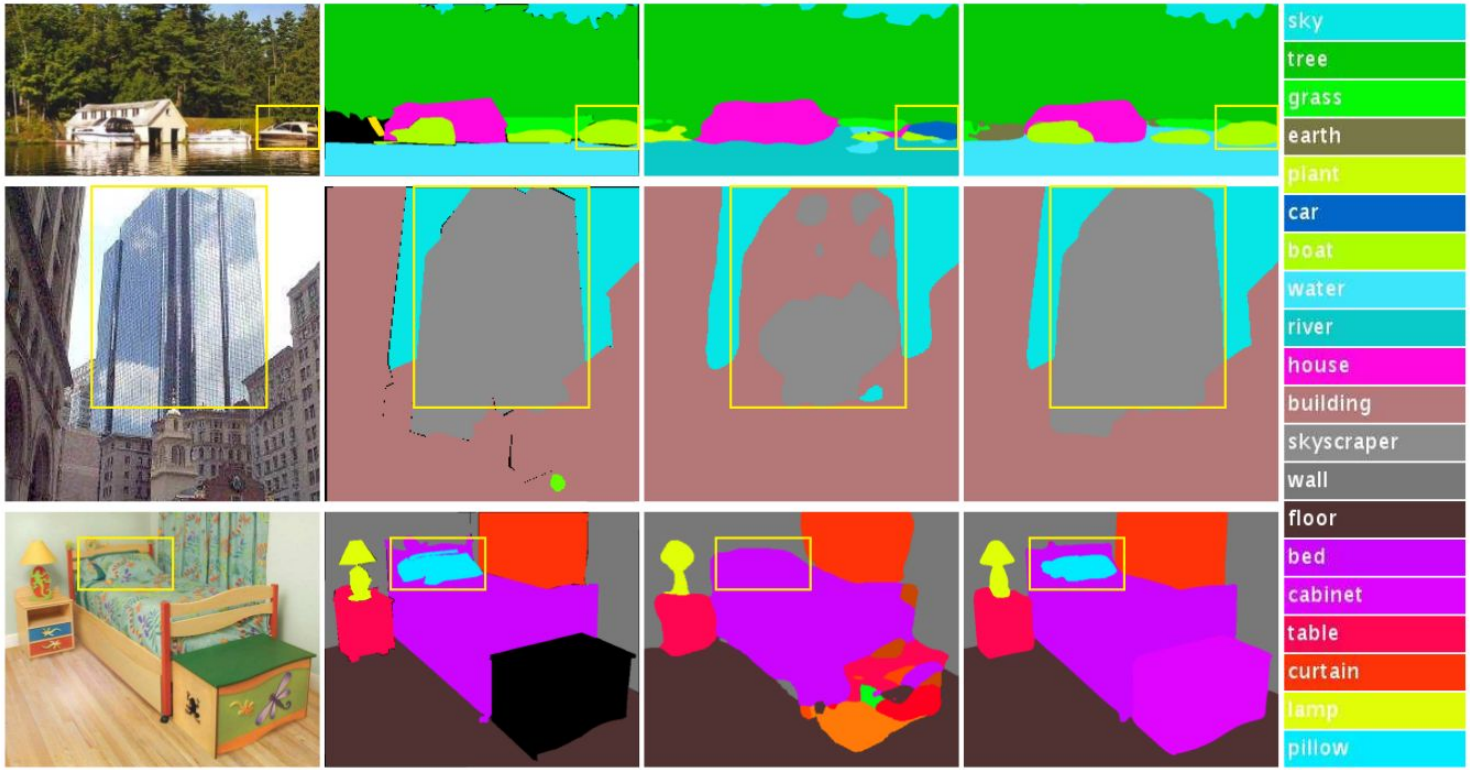
[1606.02147] ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation

PSPNet: Pyramid Scene Parsing Network



[1612.01105] Pyramid Scene Parsing Network

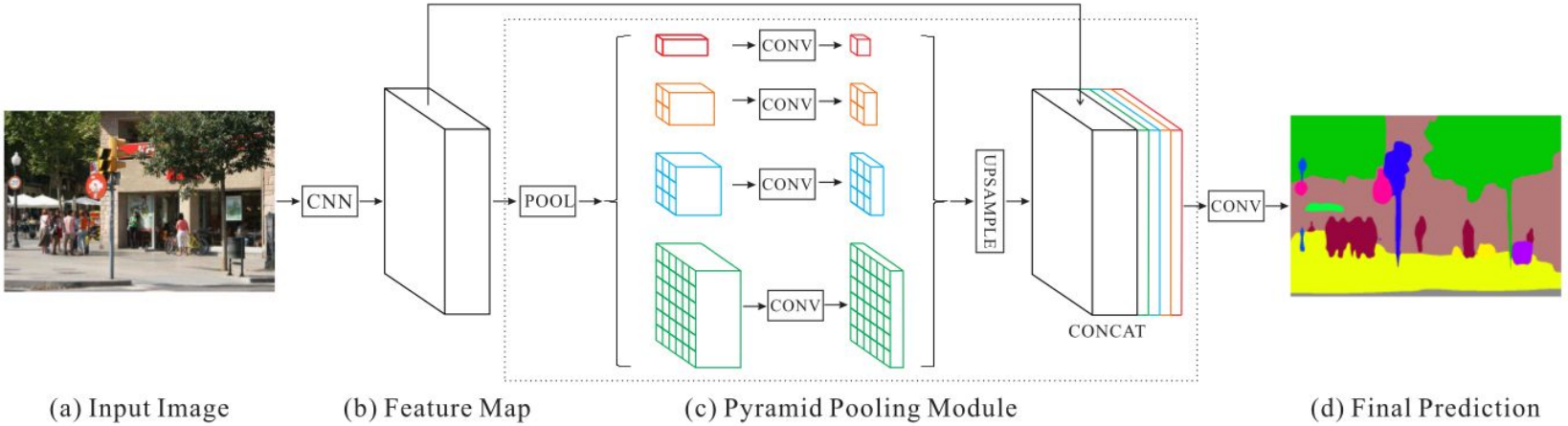
PSPNet: FCN issues



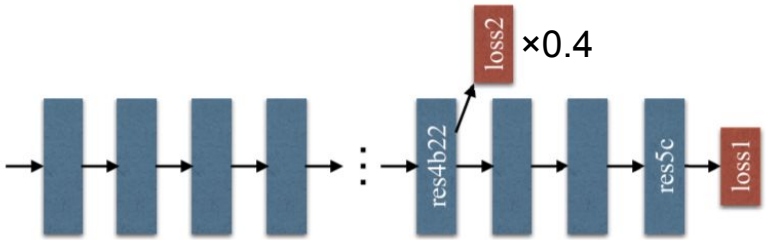
(a) Image (b) Ground Truth (c) FCN (d) PSPNet (e) ColorMap

[1612.01105] Pyramid Scene Parsing Network

PSPNet: Architecture

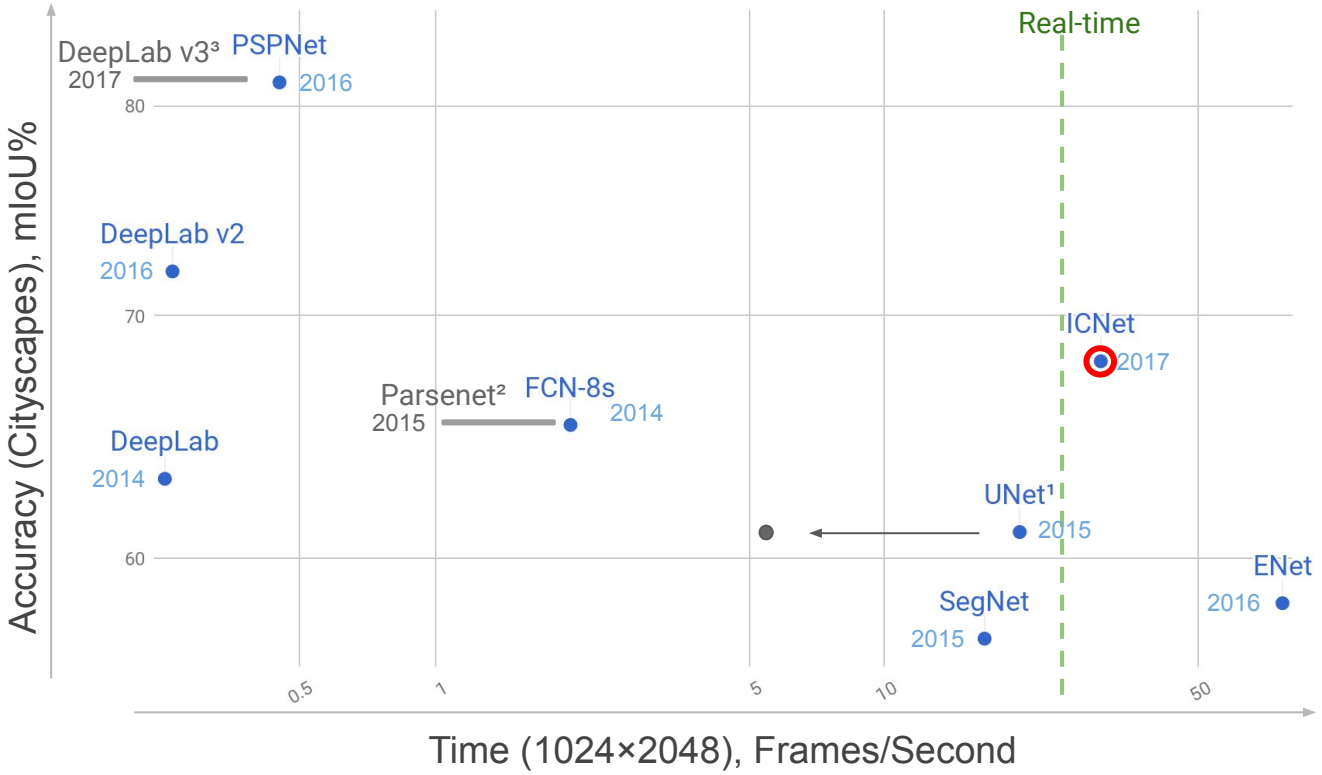


1. Pooling - AVE
2. Dimension reduction after pooling
3. Auxiliary loss
4. Multi-scale testing



[1612.01105] Pyramid Scene Parsing Network

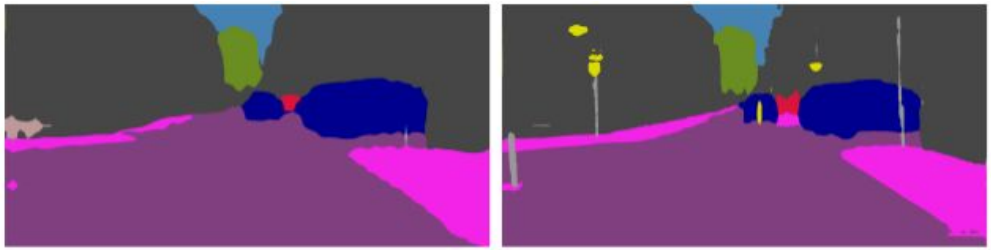
ICNet for Real-Time Semantic Segmentation



[1704.08545] ICNet for Real-Time Semantic Segmentation on High-Resolution Images

ICNet: Intuitive Speedup

1. Downsampling input



(a) scale 0.25 (42ms/60.7%) (b) scale 0.5 (123ms/68.4%)

(c) scale 1 (446ms/71.7%) (d) ground truth

2. Downsampling features

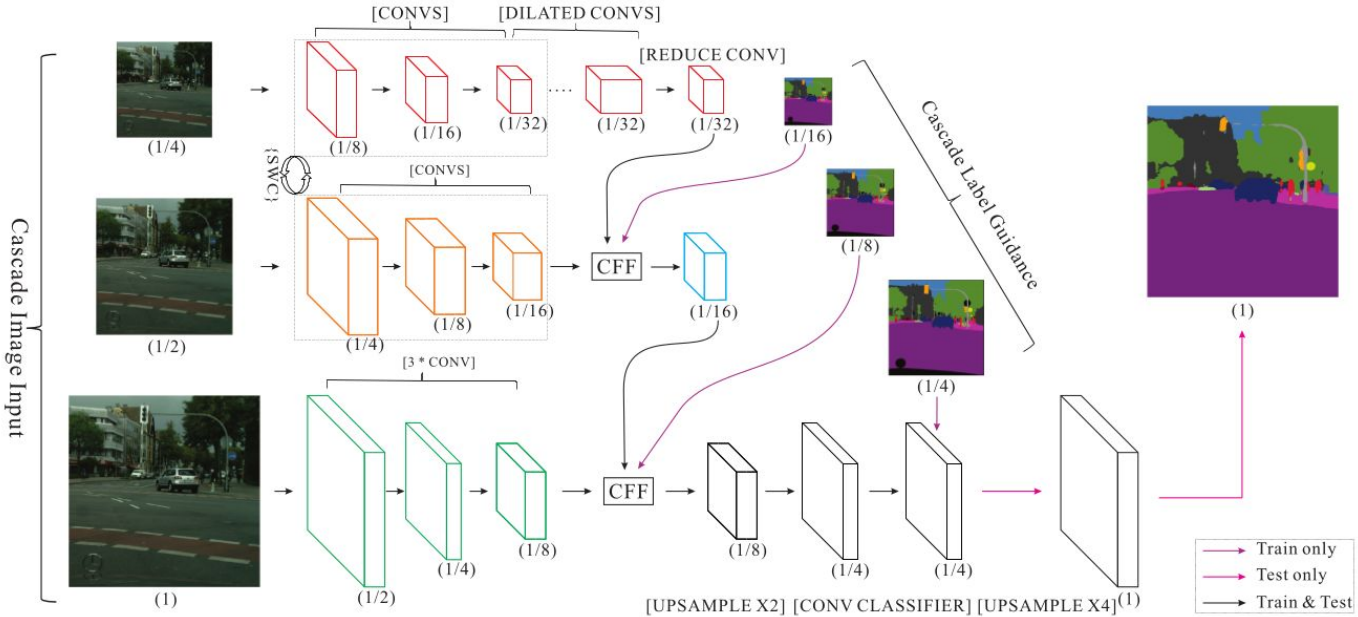
Downsample Size	8	16	32
mIoU (%)	71.7	70.2	67.1
Time (ms)	446	177	131

3. Model compression

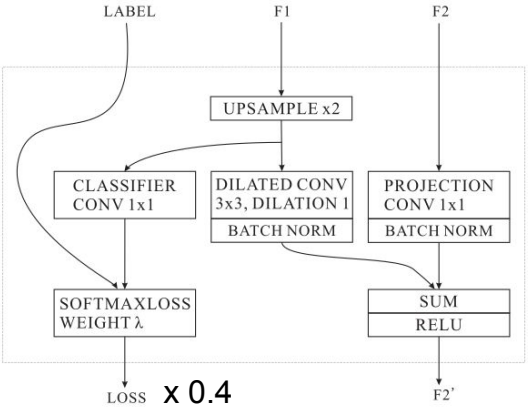
Kernel Keeping Rates	1	0.5	0.25
mIoU (%)	71.7	67.9	59.4
Time (ms)	446	170	72

[1704.08545] ICNet for Real-Time Semantic Segmentation on High-Resolution Images

ICNet: Architecture

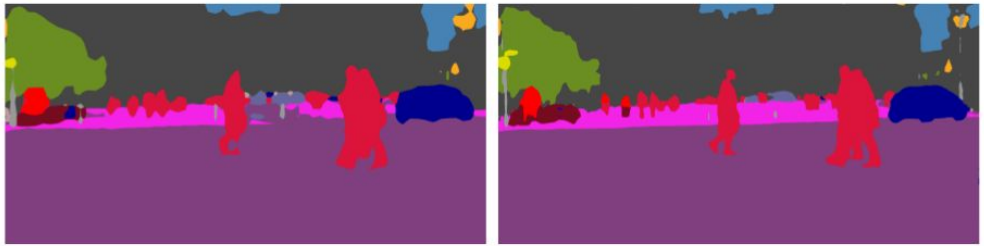


Cascade Feature Fusion (CFF) module



[1704.08545] ICNet for Real-Time Semantic Segmentation on High-Resolution Images

ICNet: Results



(a) first branch (b) second branch

(c) third branch (d) ground truth

5x+ speedup of inference, reduces memory consumption by 5+ times.

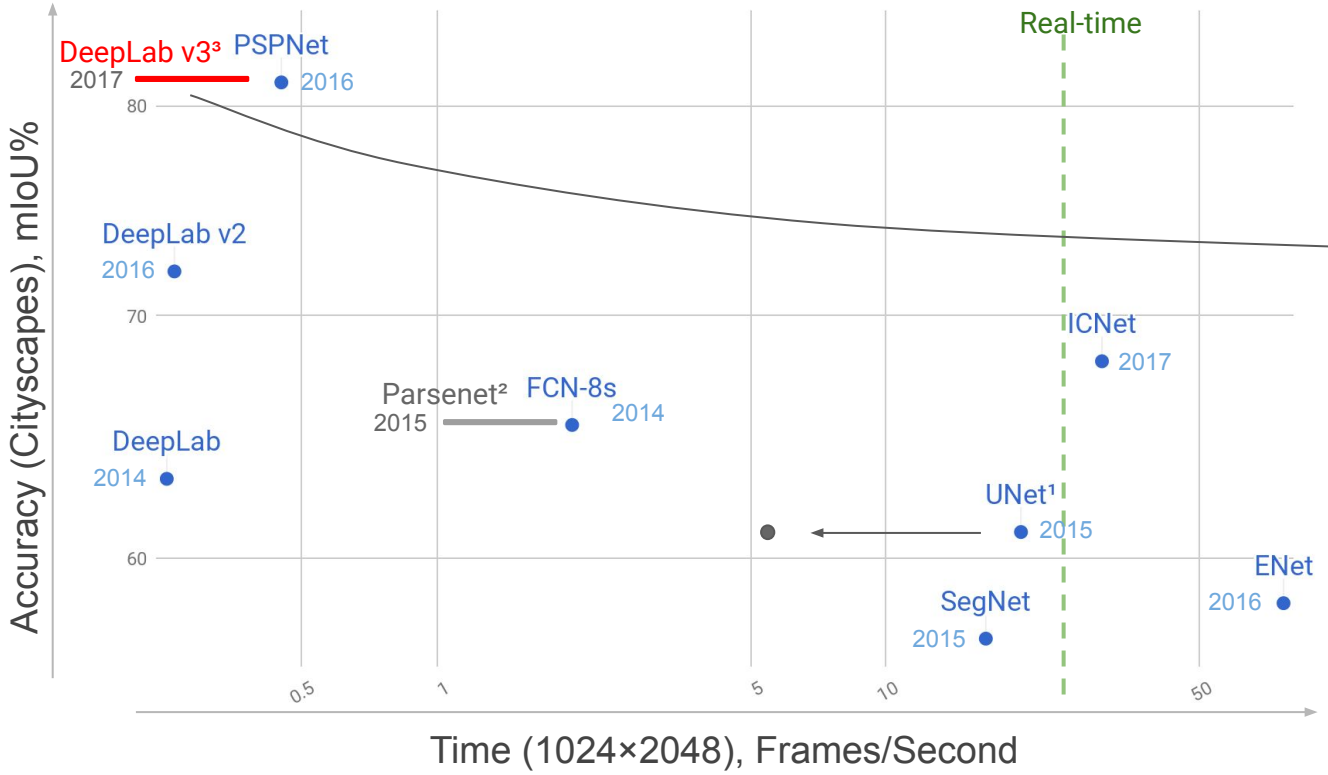
30.3 FPS at resolution 1024×2048.

Items	Baseline	sub4	sub24	sub124
mIoU (%)	67.9	59.6	66.5	**67.7**
Time (ms)	170	18	25	33
Frame (fps)	5.9	55.6	40	**30.3**
Speedup	1×	9.4×	6.8×	**5.2**×
Memory (GB)	9.2	0.6	1.1	1.6
Memory Save	1×	15.3×	8.4×	**5.8**×

PSPNet50 with 0.5 compression

[1704.08545] ICNet for Real-Time Semantic Segmentation on High-Resolution Images

DeepLab v3 & DeepLab v3+



No CRF postprocessing this time!

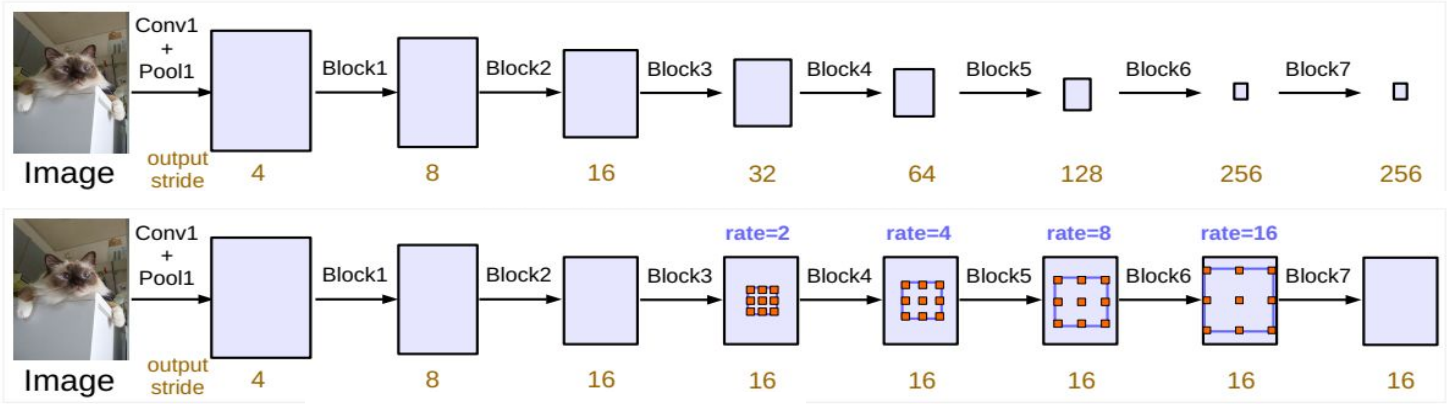
2018 DeepLab v3+ - 82.1

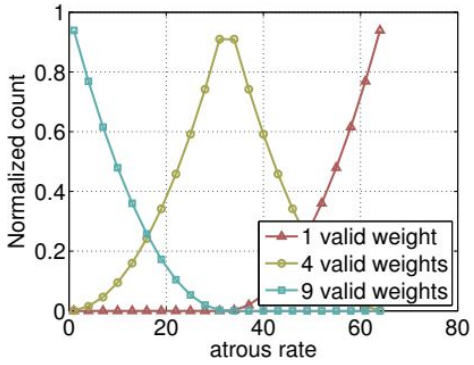
2017 DeepLab v3 - 81.3

2016 PSPNet - 81.2

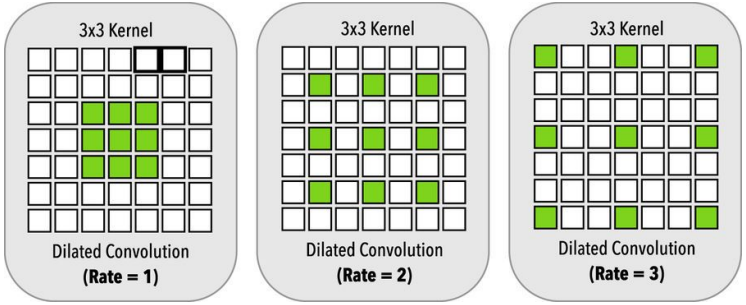
[1706.05587] Rethinking Atrous Convolution for Semantic Image Segmentation

DeepLab v3: Rethinking Atrous Convolution





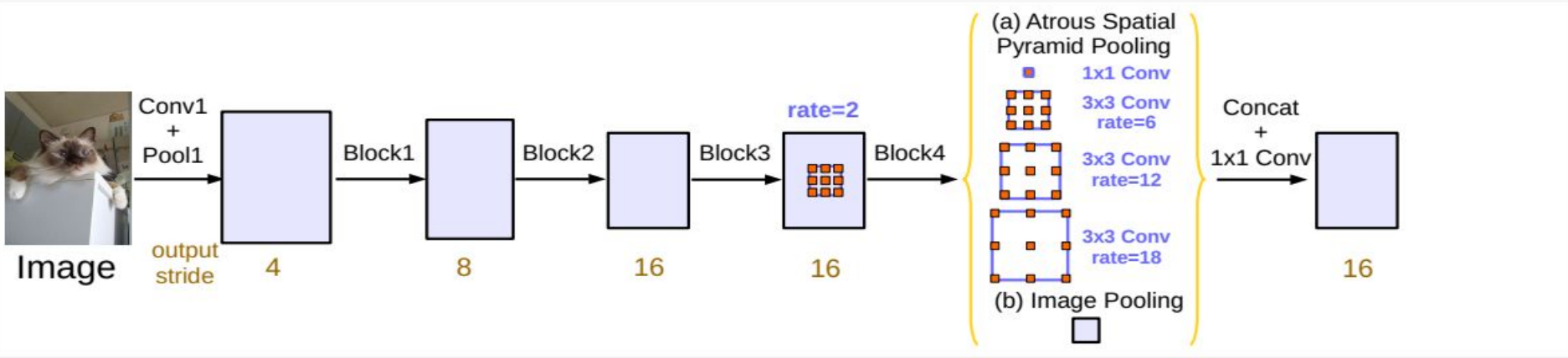
65x65 feature map



https://www.nature.com/articles/s41598-018-24304-3

[1706.05587] Rethinking Atrous Convolution for Semantic Image Segmentation

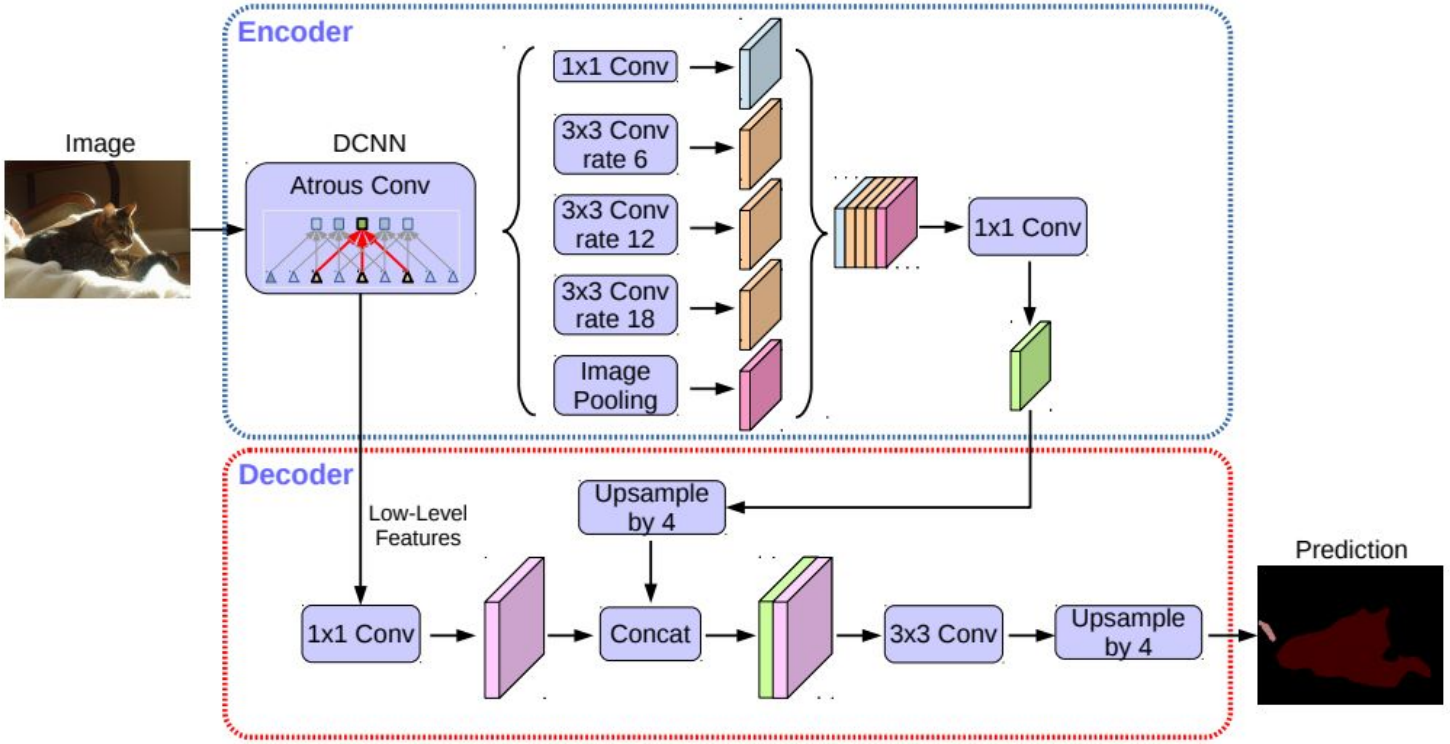
DeepLab v3: Architecture (ResNet-based)



- Multi-grid method: Block4 has three 3x3 convolutions with rates (2, 4, 8).
- Augment ASPP with global context and batch normalization.
- Upsample logits but not GT.
- Use lower output resolution but larger batches on early stages, that freeze BN and reduce batch size.

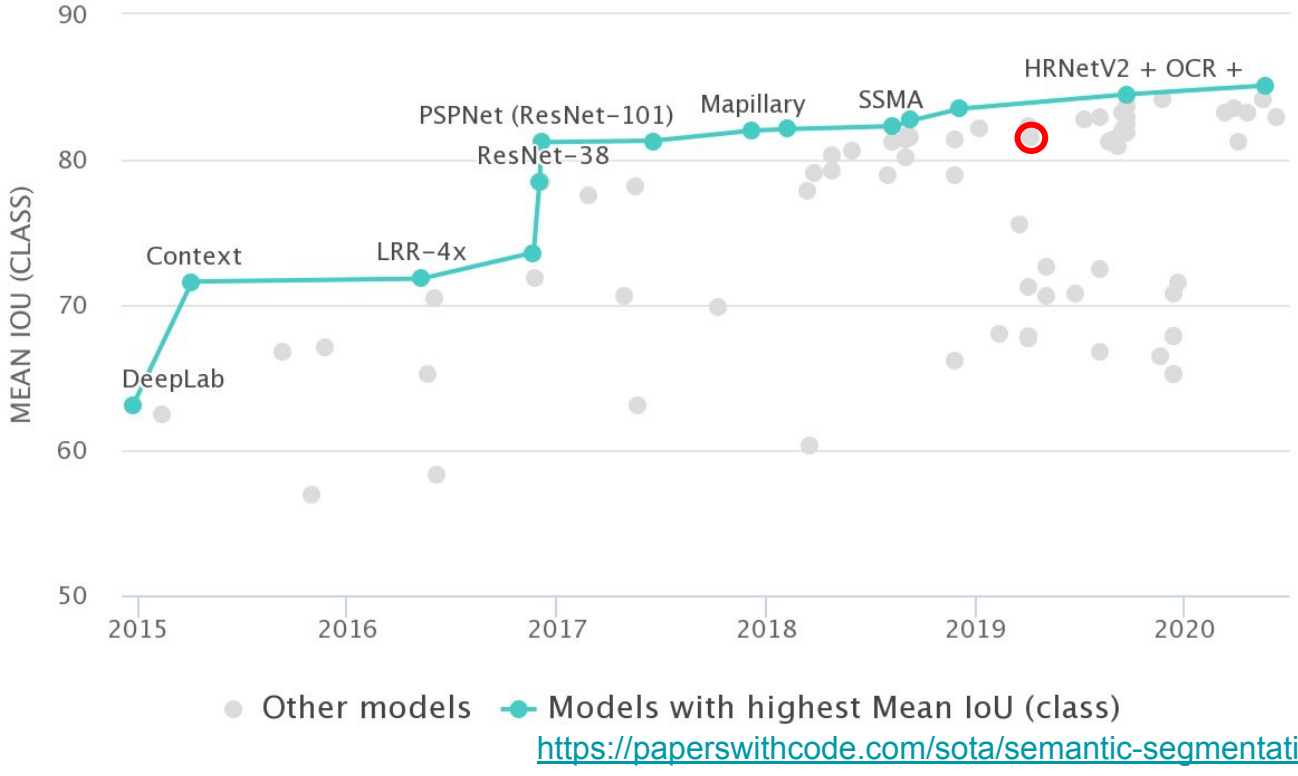
[1706.05587] Rethinking Atrous Convolution for Semantic Image Segmentation

DeepLab v3+: Architecture (Xception-based)



[1802.02611] Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation

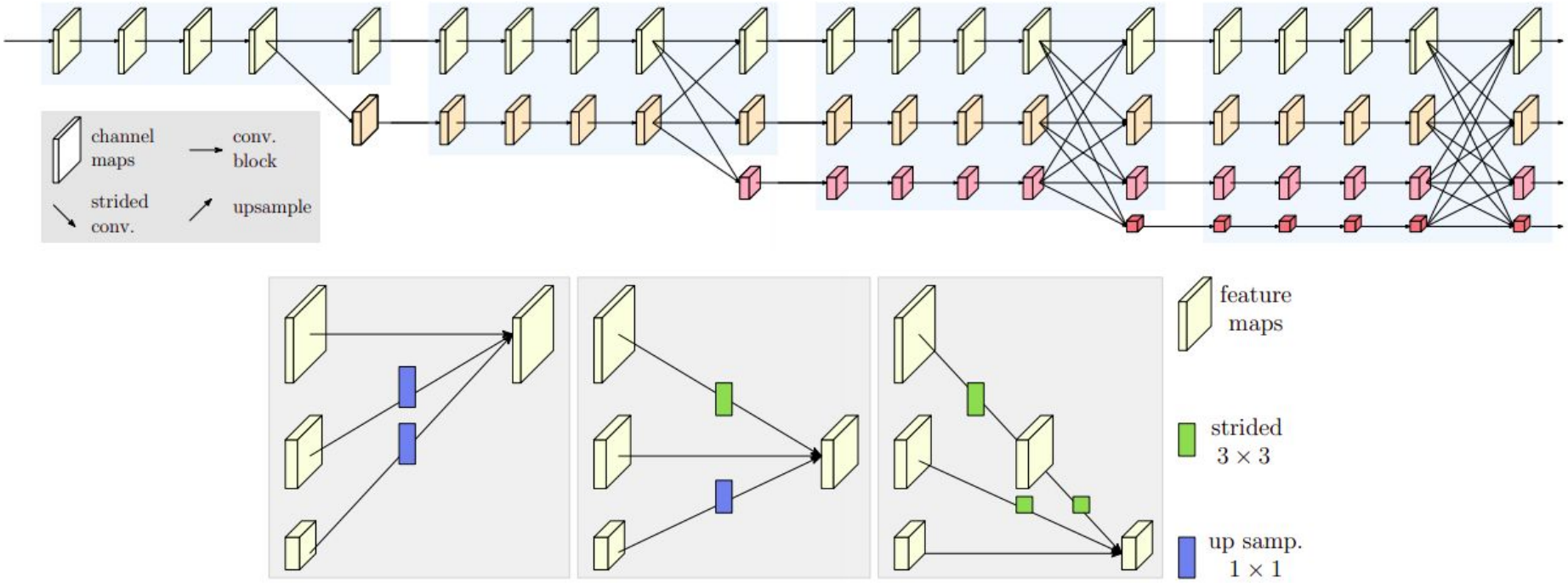
HRNet: High-Resolution Representations for Labeling Pixels and Regions



https://paperswithcode.com/sota/semantic-segmentation-on-cityscapes

[1904.04514] High-Resolution Representations for Labeling Pixels and Regions

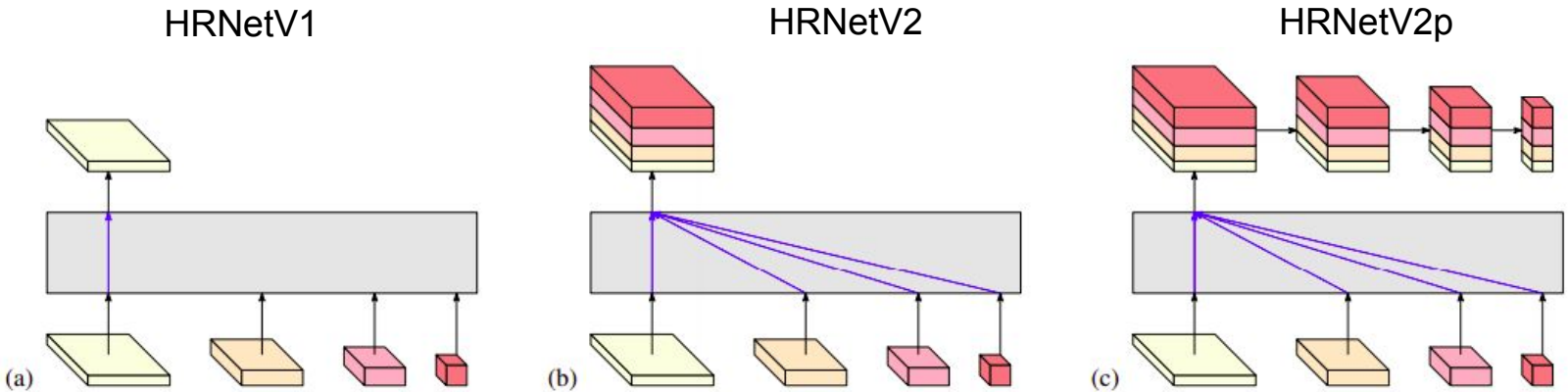
HRNet: High-Resolution Representations for Labeling Pixels and Regions



[1902.09212] Deep High-Resolution Representation Learning for Human Pose Estimation

[1904.04514] High-Resolution Representations for Labeling Pixels and Regions

HRNet: High-Resolution Representations for Labeling Pixels and Regions

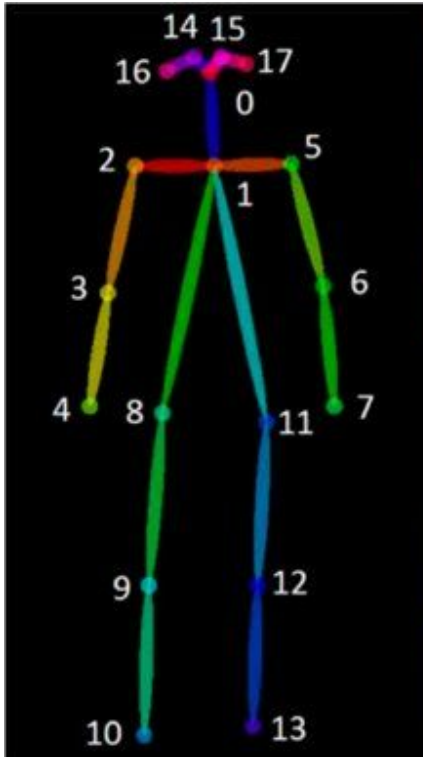


	backbone	#param.	GFLOPs	mIoU
UNet++ [133]	ResNet-101	59.5M	748.5	75.5
DeepLabv3 [14]	Dilated-ResNet-101	58.0M	1778.7	78.5
DeepLabv3+ [16]	Dilated-Xception-71	43.5M	1444.6	79.6
PSPNet [126]	Dilated-ResNet-101	65.9M	2017.6	79.7
Our approach	HRNetV2-W40	45.2M	493.2	80.2
Our approach	HRNetV2-W48	65.9M	747.3	**81.1**

[1904.04514] High-Resolution Representations for Labeling Pixels and Regions

3. Keypoint Detection

Human Keypoints



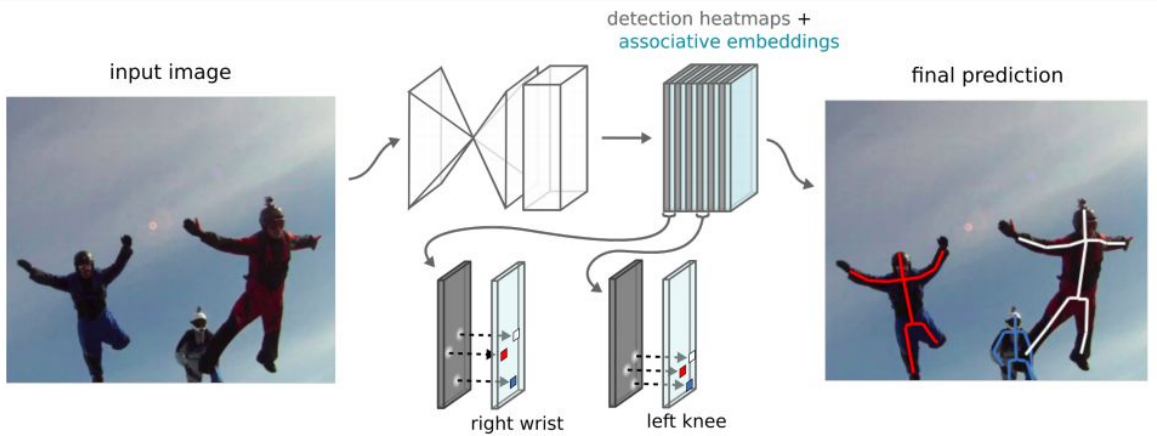
The ‘DEEP’ Landing Error Scoring System

Multiple people in the image?

Associative Embedding (1)

x_{nk} is the ground truth pixel location of the k-th body joint of the n-th person.

$$\bar{h}_n = \frac{1}{K} \sum_k h_k(x_{nk})$$

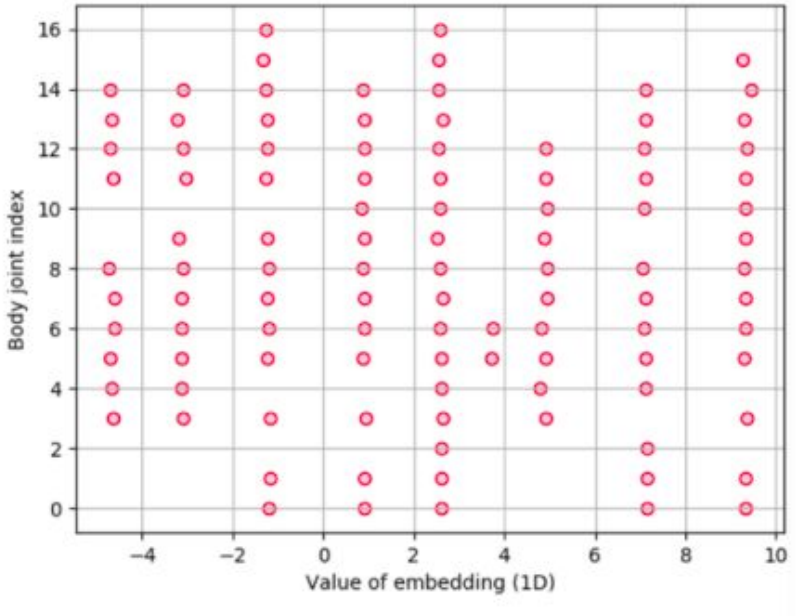


The grouping loss:

$$L_g(h, T) = \frac{1}{N} \sum_n \sum_k \left(\bar{h}_n - h_k(x_{nk},)\right)^2 + \frac{1}{N^2} \sum_n \sum_{n'} \exp\{-\frac{1}{2\sigma^2} \left(\bar{h}_n - \bar{h}_{n'}\right)^2\}$$

[1611.05424] Associative Embedding: End-to-End Learning for Joint Detection and Grouping

Associative Embedding (2)



[1611.05424] Associative Embedding: End-to-End Learning for Joint Detection and Grouping

4. Object Detection

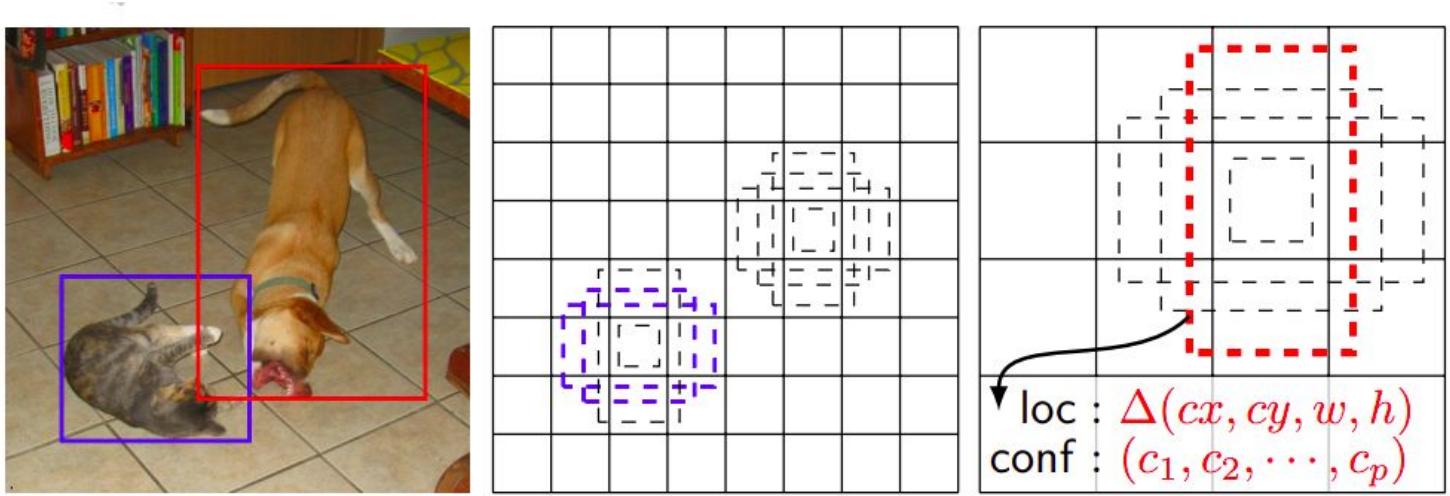
Object Detection Approaches

1. * R-CNN (review by Ross Girshick)

ICCV 2019 Tutorial. Object Detection and Instance Segmentation (slides) - Ross Girshick

2. Anchor-based single shot detectors (SSD, YOLO)
3. Keypoint-based (CornerNet, CenterNet)

Single Shot Detector (SSD)



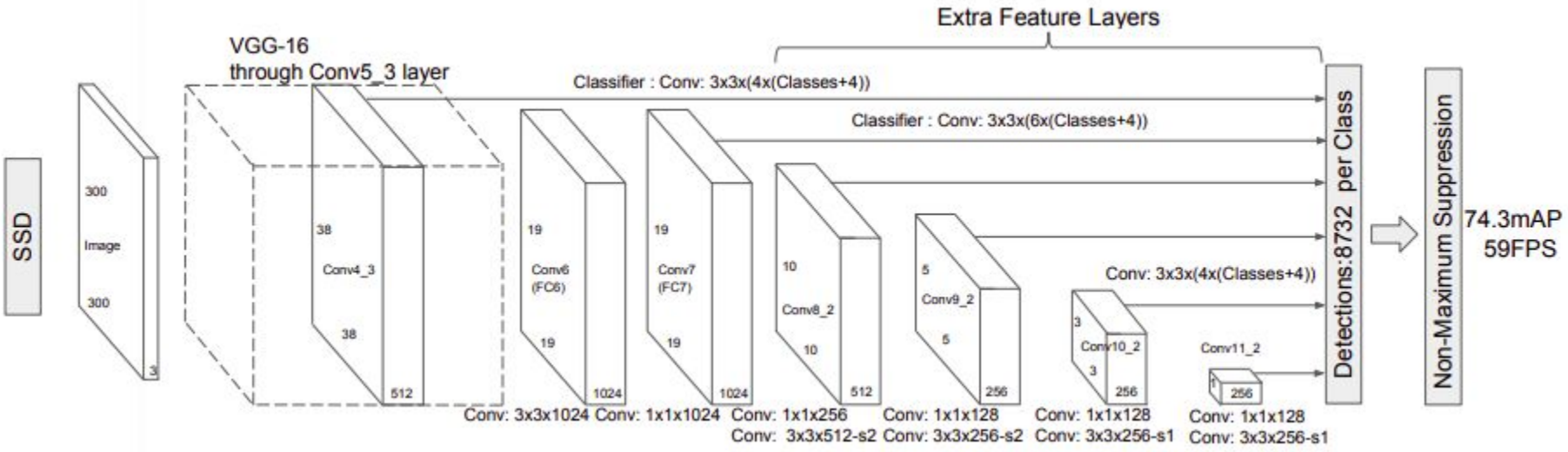
(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

Anchors (priors):

1. Aspect ratio = {1, 2, 3, ½, ⅓}
2. Scale: $s_k = s_{\min} + \frac{s_{\max} - s_{\min}}{m - 1}(k - 1), \quad k \in [1, m]$

[1512.02325] SSD: Single Shot MultiBox Detector

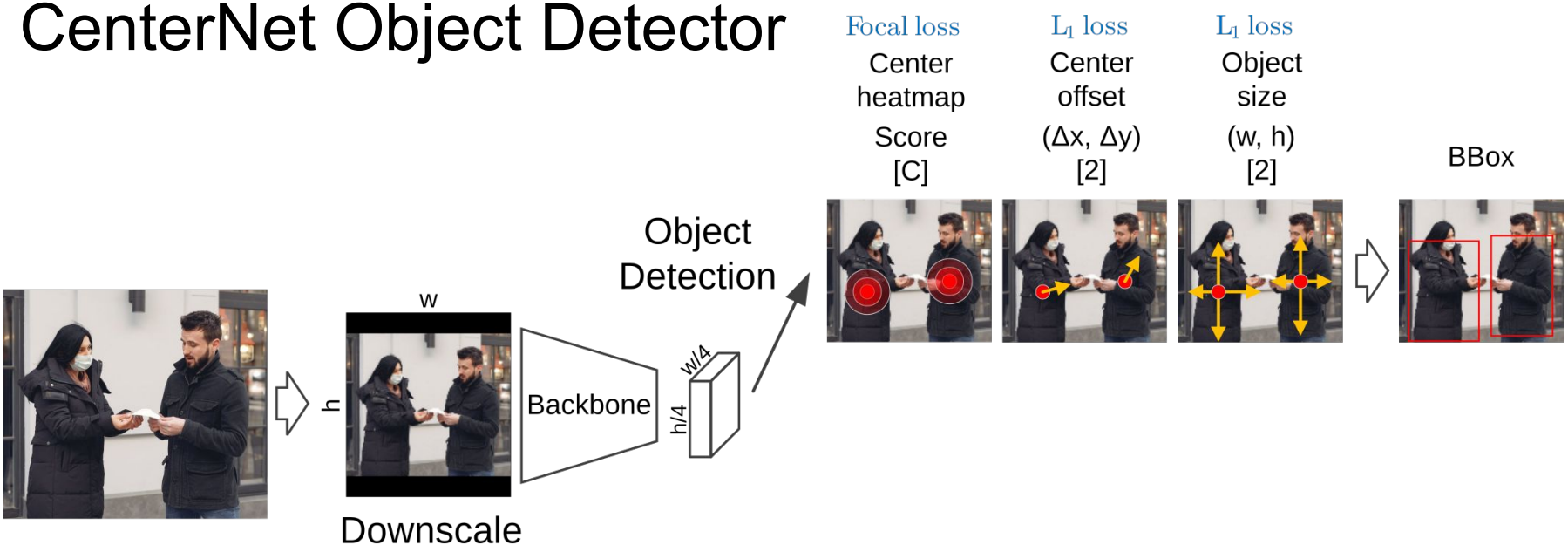
Single Shot Detector (SSD)



1. OHEM (the ratio between the negatives and positives is at most 3:1).
2. W, H - log.

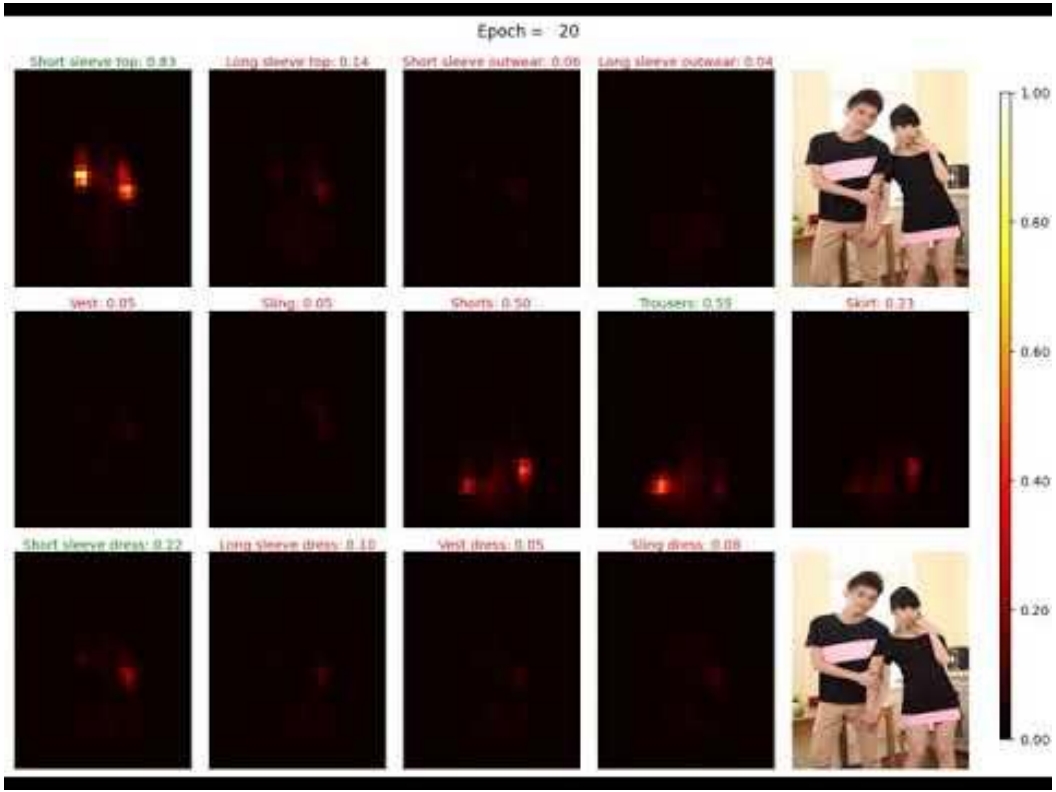
[1512.02325] SSD: Single Shot MultiBox Detector

CenterNet Object Detector



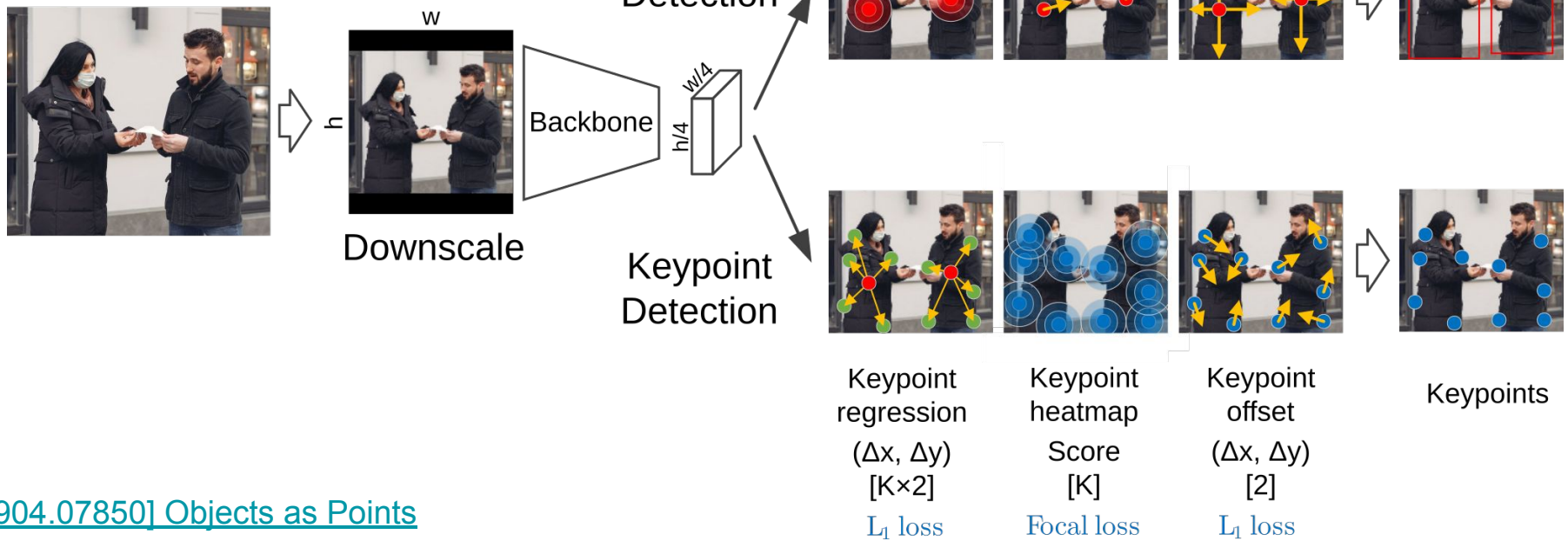
[1904.07850] Objects as Points

CenterNet training for clothing detection (object centers)



[2006.00710] DeepMark++: Real-time Clothing Detection at the Edge

CenterNet + Keypoints



[1904.07850] Objects as Points

DeepMark++: Real-time Clothing Detection at the Edge

[2006.00710] DeepMark++: Real-time Clothing Detection at the Edge