Architectures for classification and segmentation

Karel Zimmermann

http://cmp.felk.cvut.cz/~zimmerk/

Vision for Robotics and Autonomous Systems
https://cyber.felk.cvut.cz/vras/

Center for Machine Perception
https://cmp.felk.cvut.cz

Department for Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague

Outline

- Architectures of classification networks
- Architectures of segmentation networks
- Architectures of regression networks
- Architectures of detection networks
- Architectures of feature matching networks

IMAGENET

Classification results

http://image-net.org/challenges/LSVRC/2017/index

Label: Steel drum

IMAGENET

Classification results

http://image-net.org/challenges/LSVRC/2017/index

Label: Steel drum

Output:
Scale
T-shirt
Steel drum
Drumstick
Mud turtle

IMAGENET

Classification results

http://image-net.org/challenges/LSVRC/2017/index

Label: Steel drum

Output:
Scale
T-shirt
Steel drum
Drumstick
Mud turtle

✔

Output:
Scale
T-shirt
Giant panda
Drumstick
Mud turtle

✗

IMAGENET

Classification results

http://image-net.org/challenges/LSVRC/2017/index

Label: Steel drum

Output:
Scale
T-shirt
Steel drum
Drumstick
Mud turtle

✔

Output:
Scale
T-shirt
Giant panda
Drumstick
Mud turtle

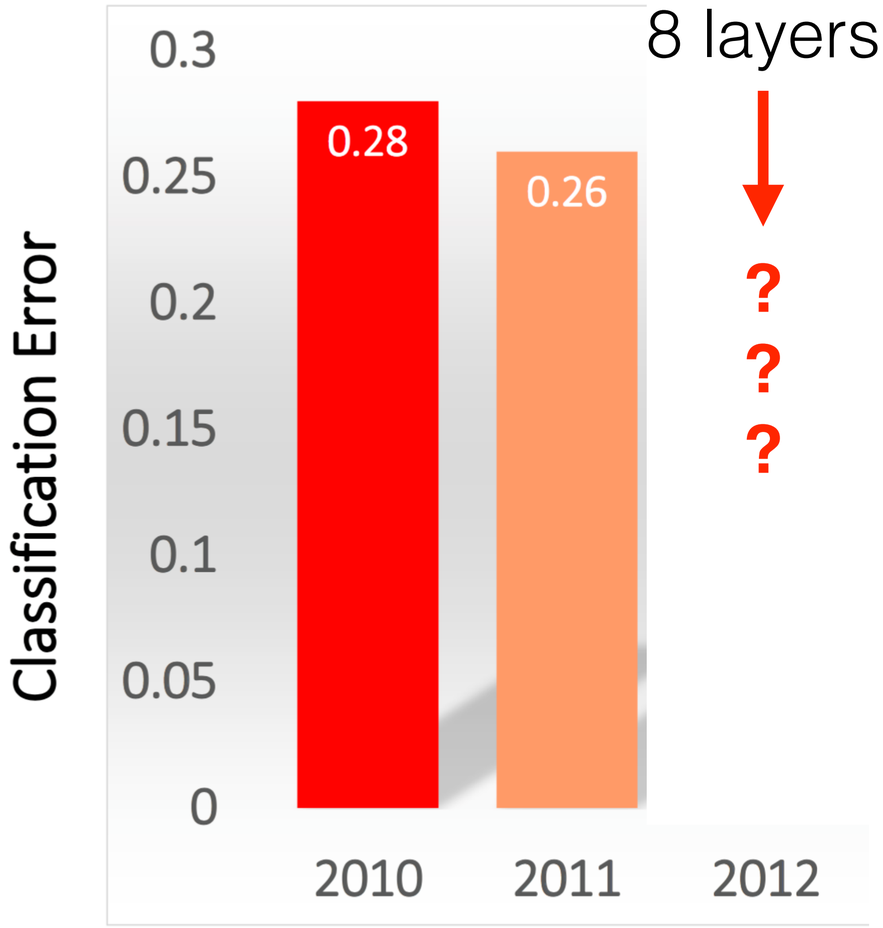
✗

$$\text{Error} = \frac{1}{100{,}000} \sum_{\substack{100{,}000 \\ \text{images}}} 1[\text{incorrect on image i}]$$

IMAGENET

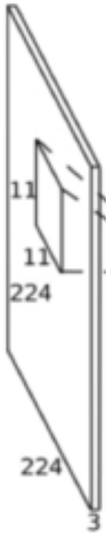
Classification results

AlexNet
8 layers



?
?
?

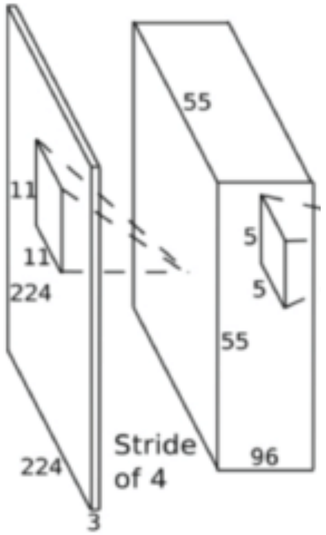
AlexNet on ImageNet 2012 (**over 27k citations !!!**)



- Param in layer1 (conv, 96 11x11 filters, stride=4, pad=0)?

Alex Krizhevsky et al, Imagenet classification with deep convolutional neural networks, NIPS, 2012
https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

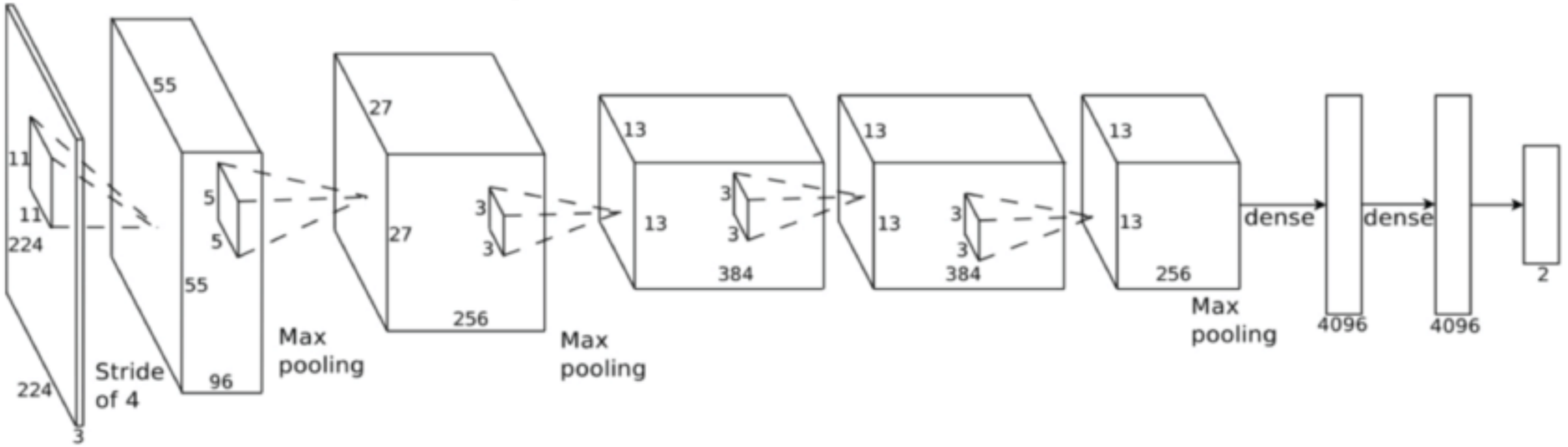
AlexNet on ImageNet 2012 (**over 27k citations !!!**)



- Param in layer1 (conv, 96 11x11 filters, stride=4, pad=0)?
- Param in layer2 (maxp,3x3 filters, stride=2, pad=0)?

Alex Krizhevsky et al, Imagenet classification with deep convolutional neural networks, NIPS, 2012
https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

AlexNet on ImageNet 2012 (**over 27k citations !!!**)



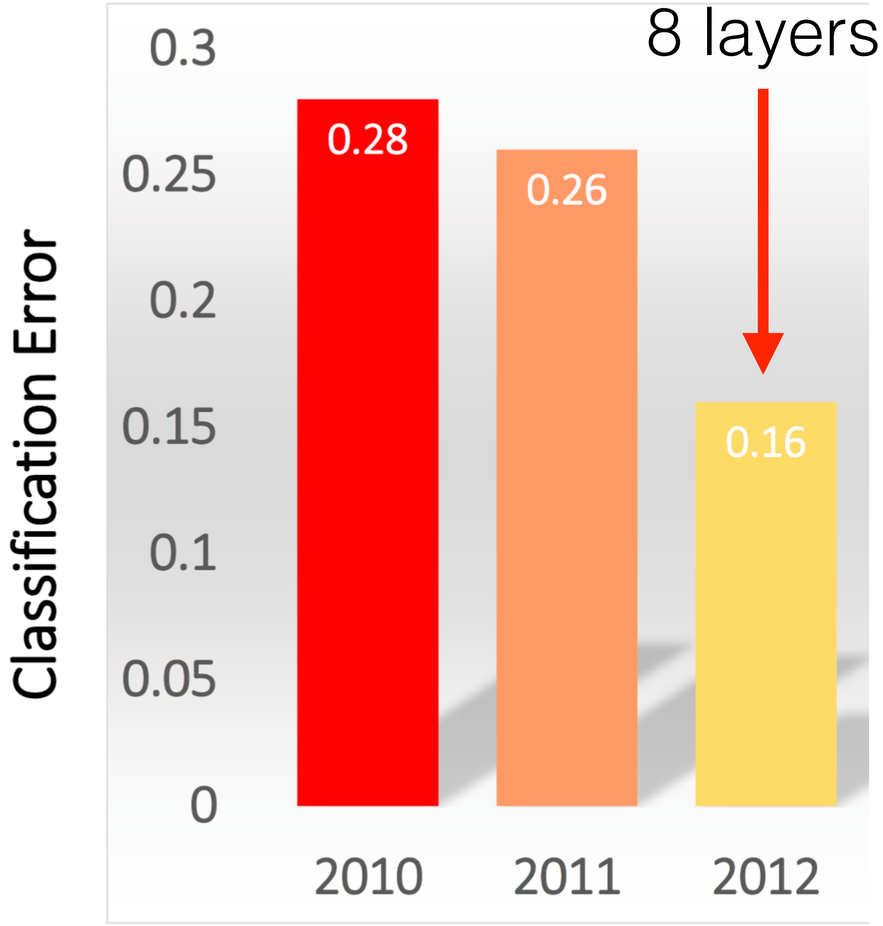
- Param in layer1 (conv, 96 11x11 filters, stride=4, pad=0)?
- Param in layer2 (maxp,3x3 filters, stride=2, pad=0)?
- Param in layer3 (conv, 256 5x5 filters, stride=1, pad=2?
- Parameters in total: 60M, Depth: 8 layers

Alex Krizhevsky et al, Imagenet classification with deep convolutional neural networks, NIPS, 2012
https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

IMAGENET

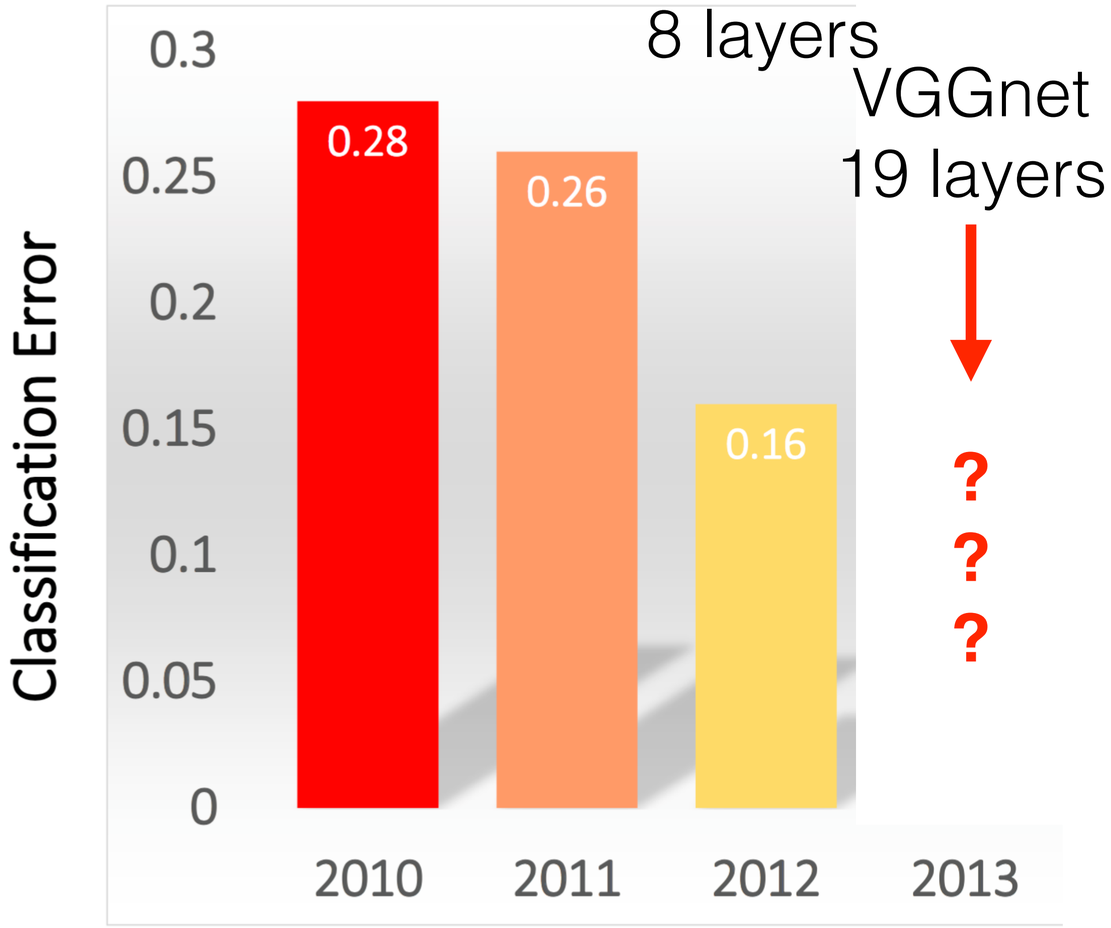
Classification results

AlexNet
8 layers

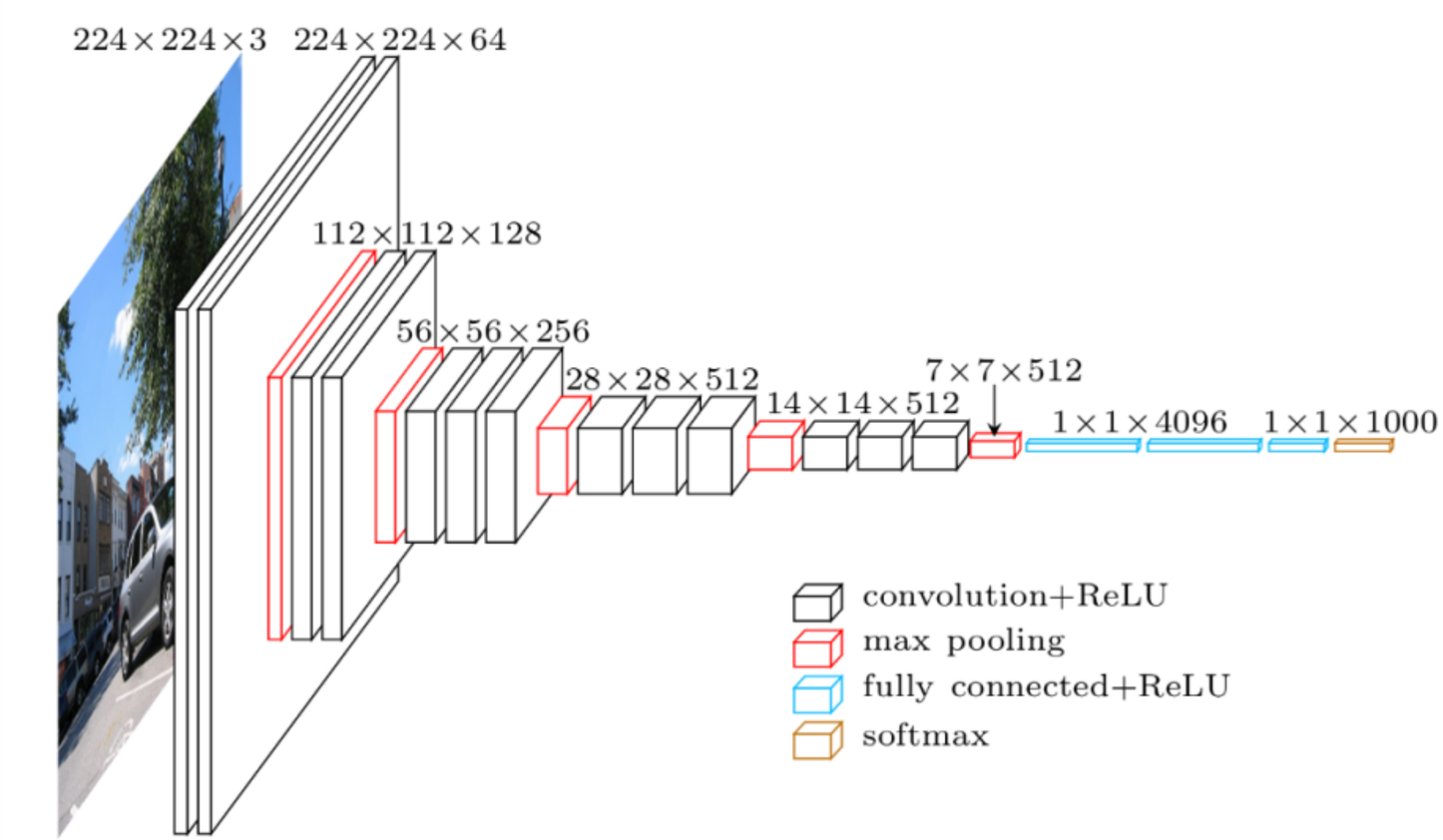


IMAGENET

Classification results



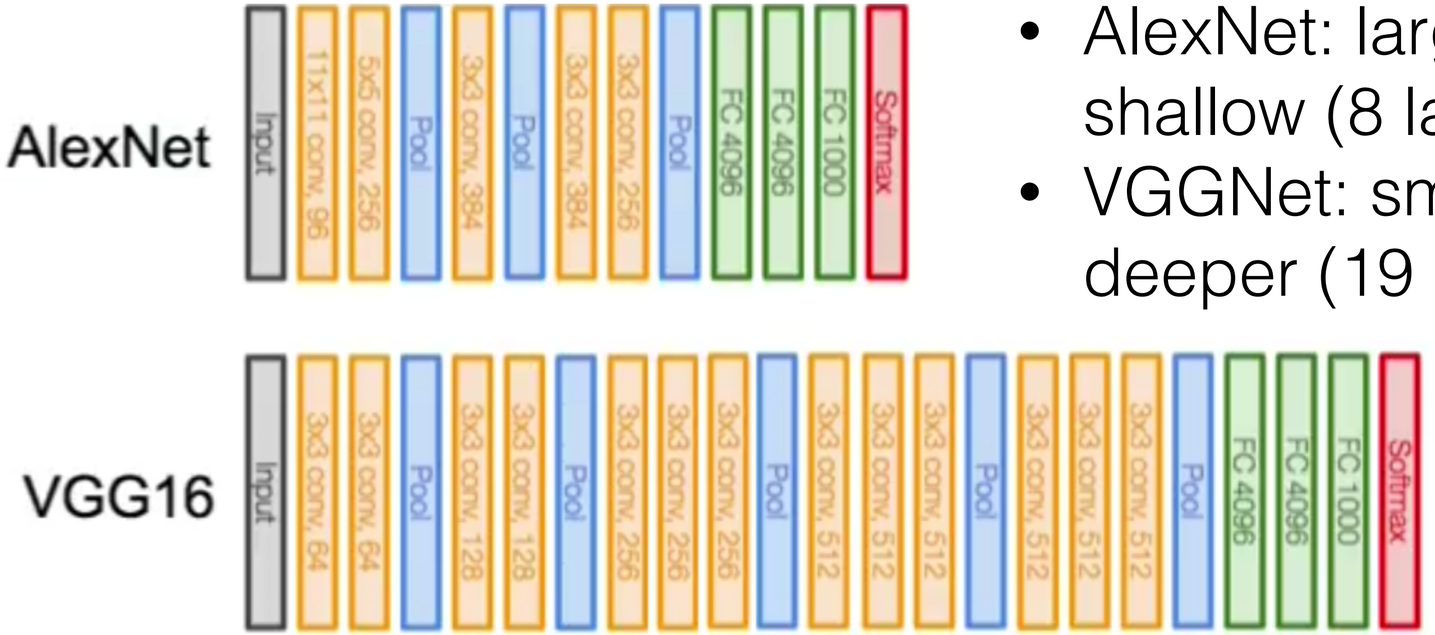
VGGNet



- Parameters in total: 138M, Depth: 19 layers

Simonyan and Zissermann,Very Deep Convolutional Networks for Large S
Recognition, 2014
https://arxiv.org/abs/1409.1556

VGGNet vs AlexNet

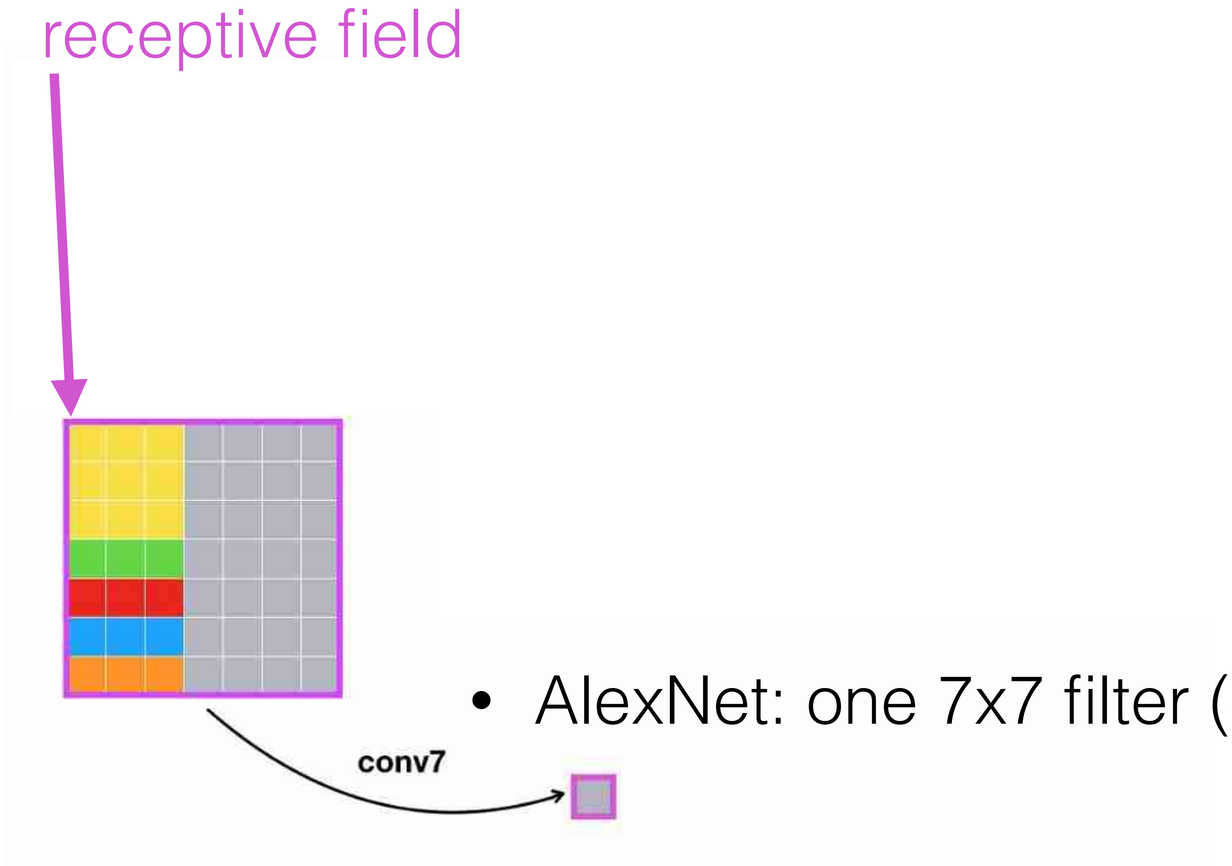


- AlexNet: large filters shallow (8 layers)
- VGGNet: small filters deeper (19 layers)

- Parameters in total: 138M, Depth: 19 layers

Simonyan and Zissermann,Very Deep Convolutional Networks for Large Scale Image Recognition, 2014
https://arxiv.org/abs/1409.1556

VGGNet vs AlexNet



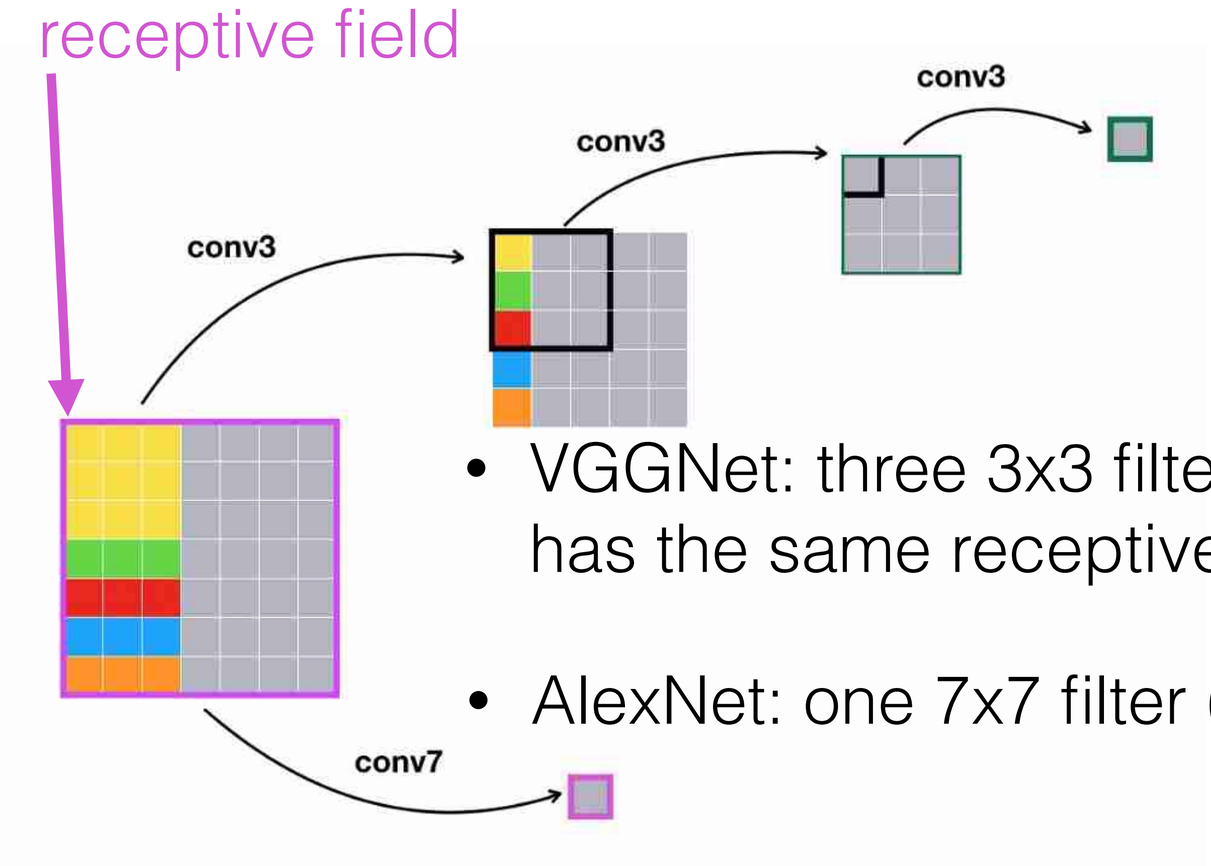
- AlexNet: one 7x7 filter (49+1 params)

Image from: https://mc.ai/cnn-architectures-vggnet/

Simonyan and Zissermann,Very Deep Convolutional Networks for Large Scale Image Recognition, 2014

https://arxiv.org/abs/1409.1556

VGGNet vs AlexNet



- VGGNet: three 3x3 filters (3x9+3 params) has the same receptive field
- AlexNet: one 7x7 filter (49+1 params)

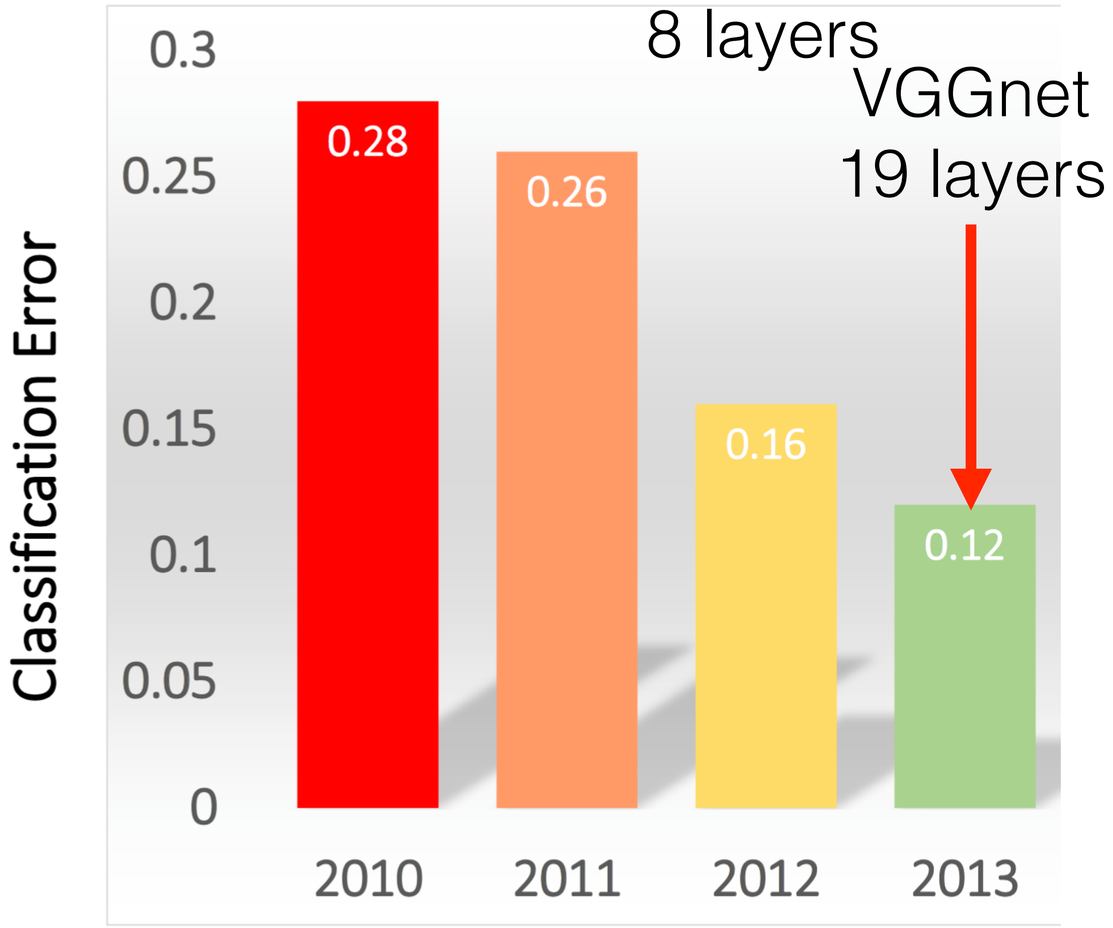
Image from: https://mc.ai/cnn-architectures-vggnet/

Simonyan and Zissermann,Very Deep Convolutional Networks for Large Scale Image Recognition, 2014

https://arxiv.org/abs/1409.1556

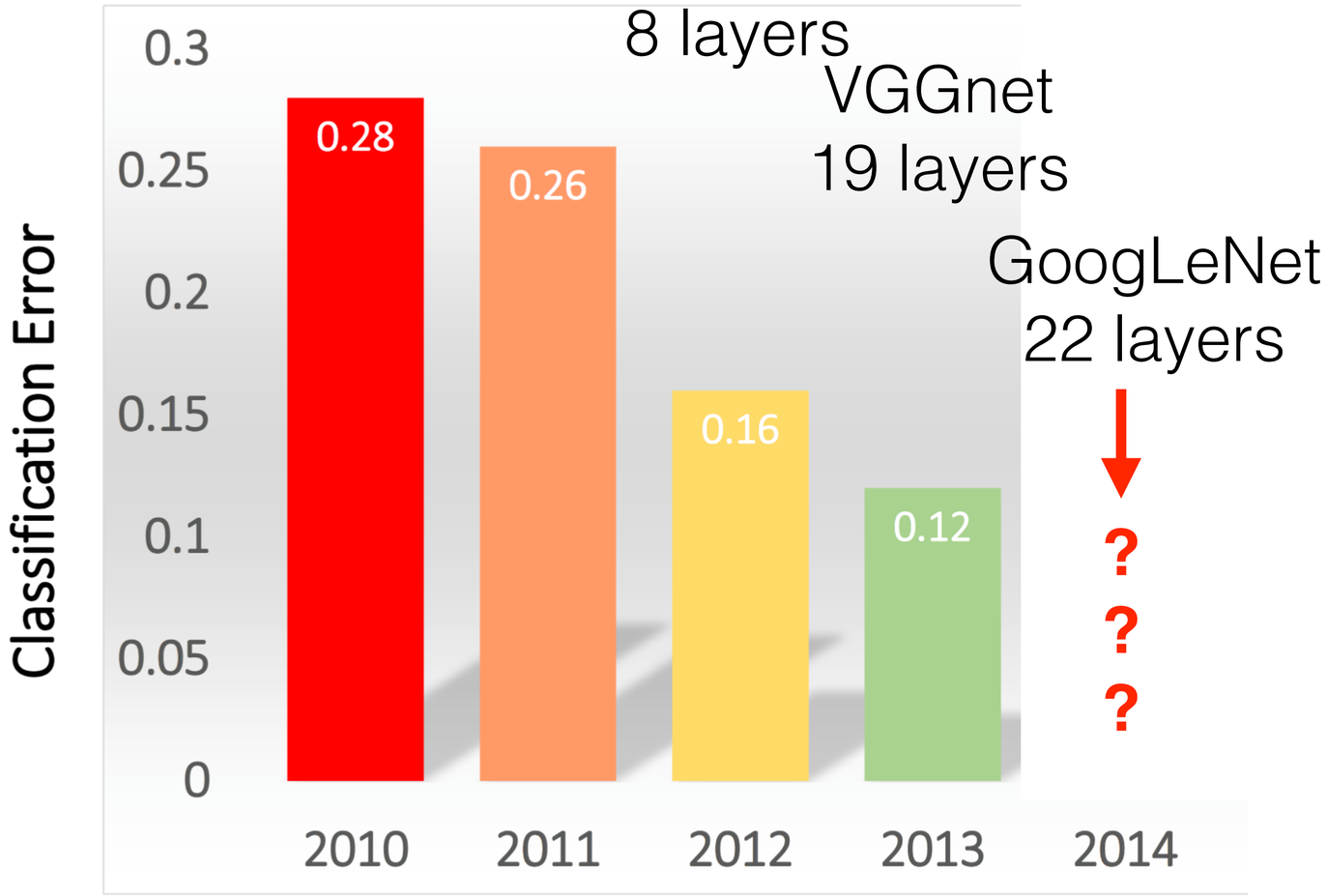
IMAGENET

Classification results

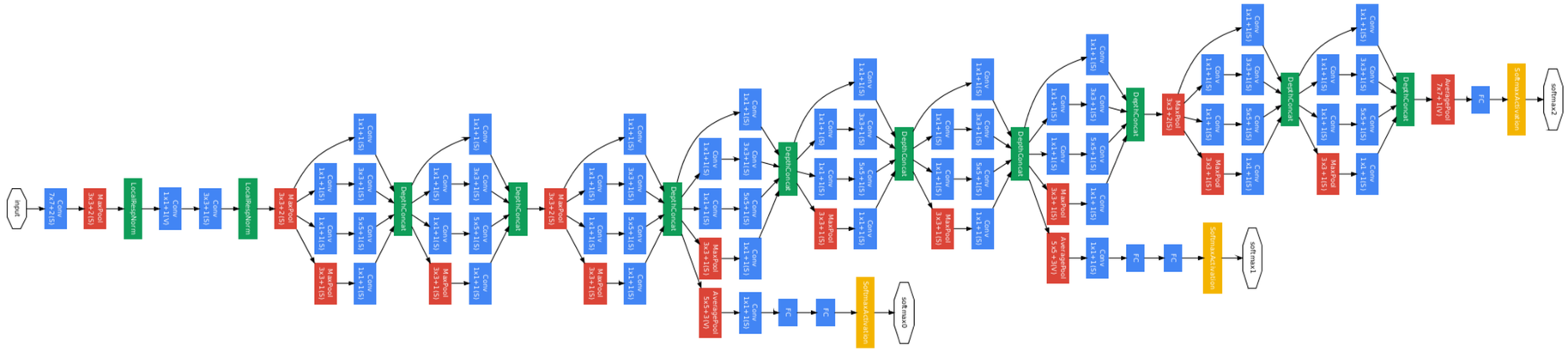


IMAGENET

Classification results

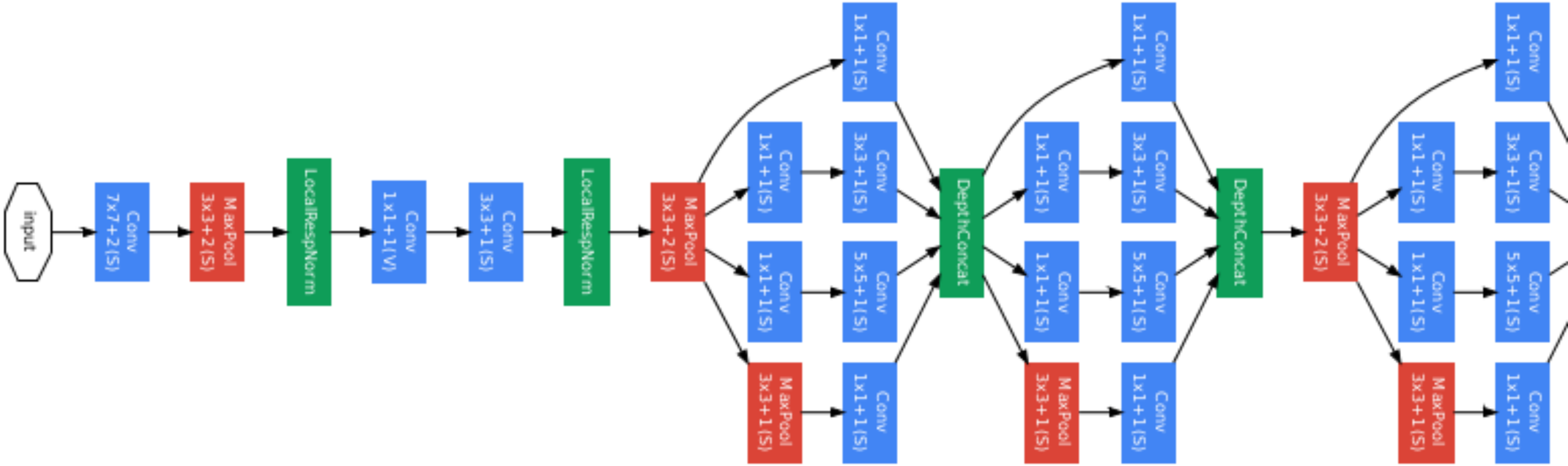


GoogLeNet



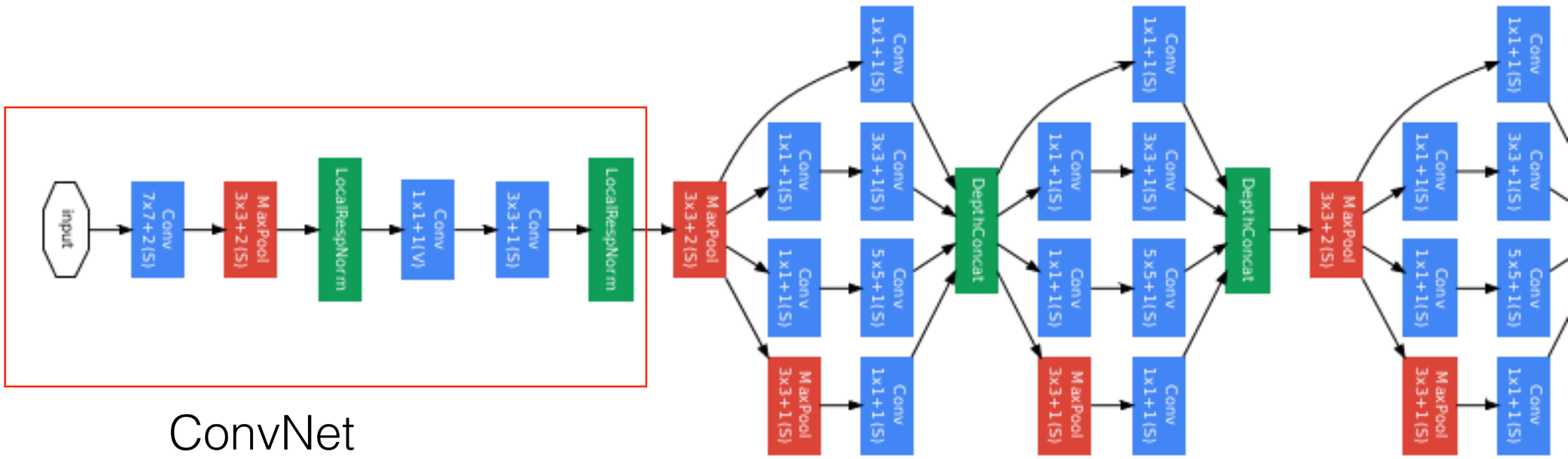
Szegedy et al. Going Deeper with Convolutions, CVPR, 2014
https://arxiv.org/abs/1409.4842

GoogLeNet



Szegedy et al. Going Deeper with Convolutions, CVPR, 2014
https://arxiv.org/abs/1409.4842

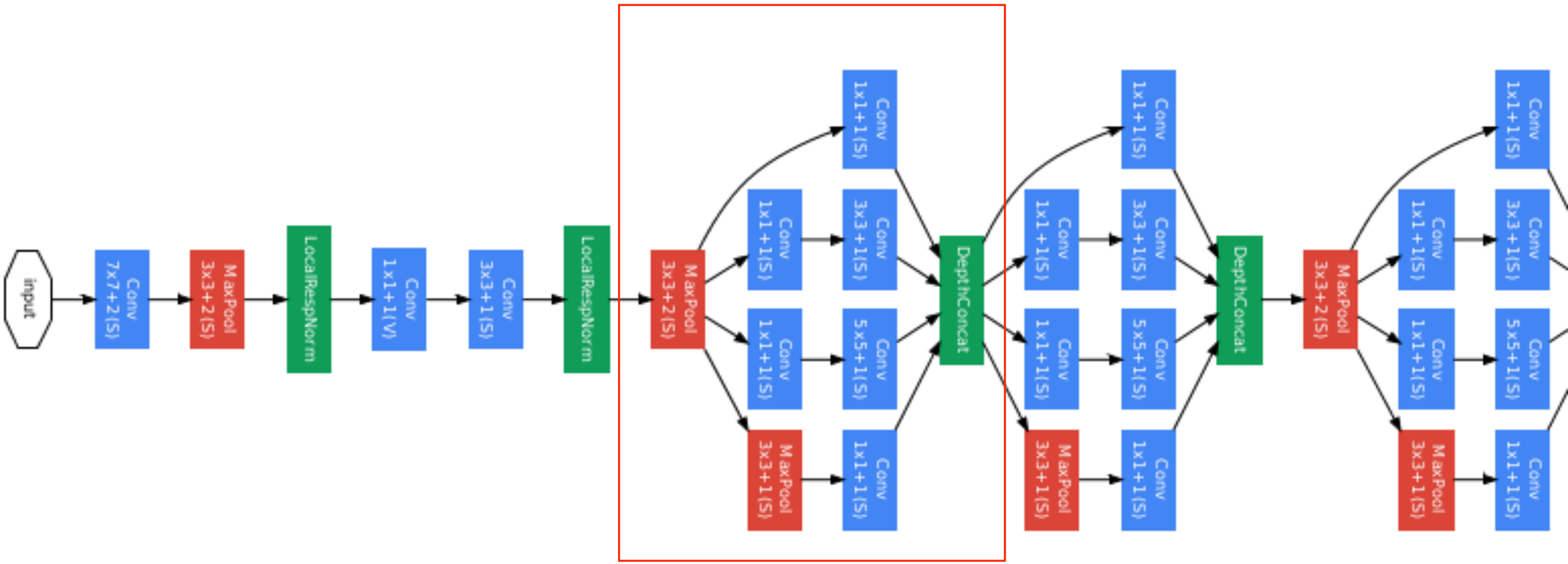
GoogLeNet



ConvNet

Szegedy et al. Going Deeper with Convolutions, CVPR, 2014
https://arxiv.org/abs/1409.4842

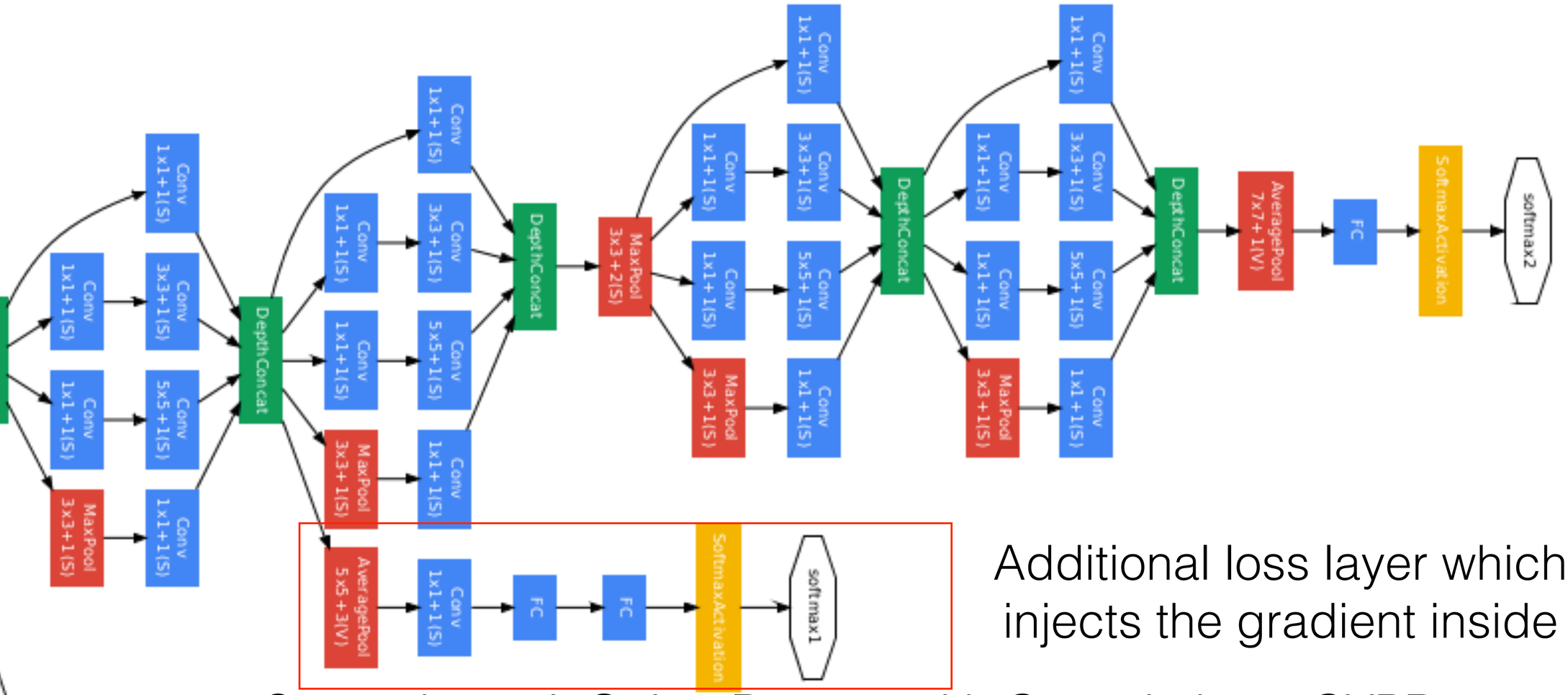
GoogLeNet



Inception module

Szegedy et al. Going Deeper with Convolutions, CVPR, 2014
https://arxiv.org/abs/1409.4842

GoogLeNet

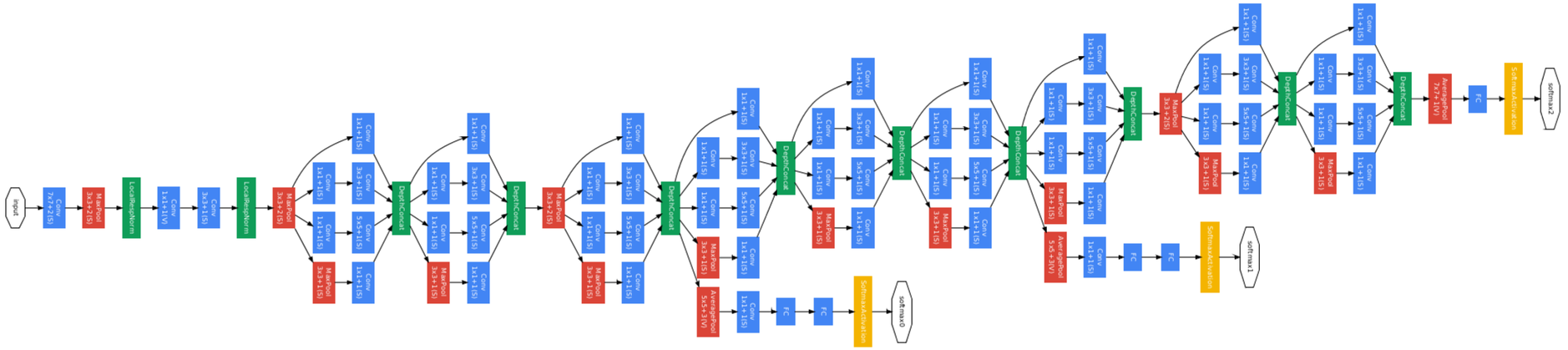


Additional loss layer which injects the gradient inside

Szegedy et al. Going Deeper with Convolutions, CVPR, 2014
https://arxiv.org/abs/1409.4842

image source: http://joelouismarino.github.io/images/blog_images/blog_googlenet_keras/googlenet_diagram.png

GoogLeNet



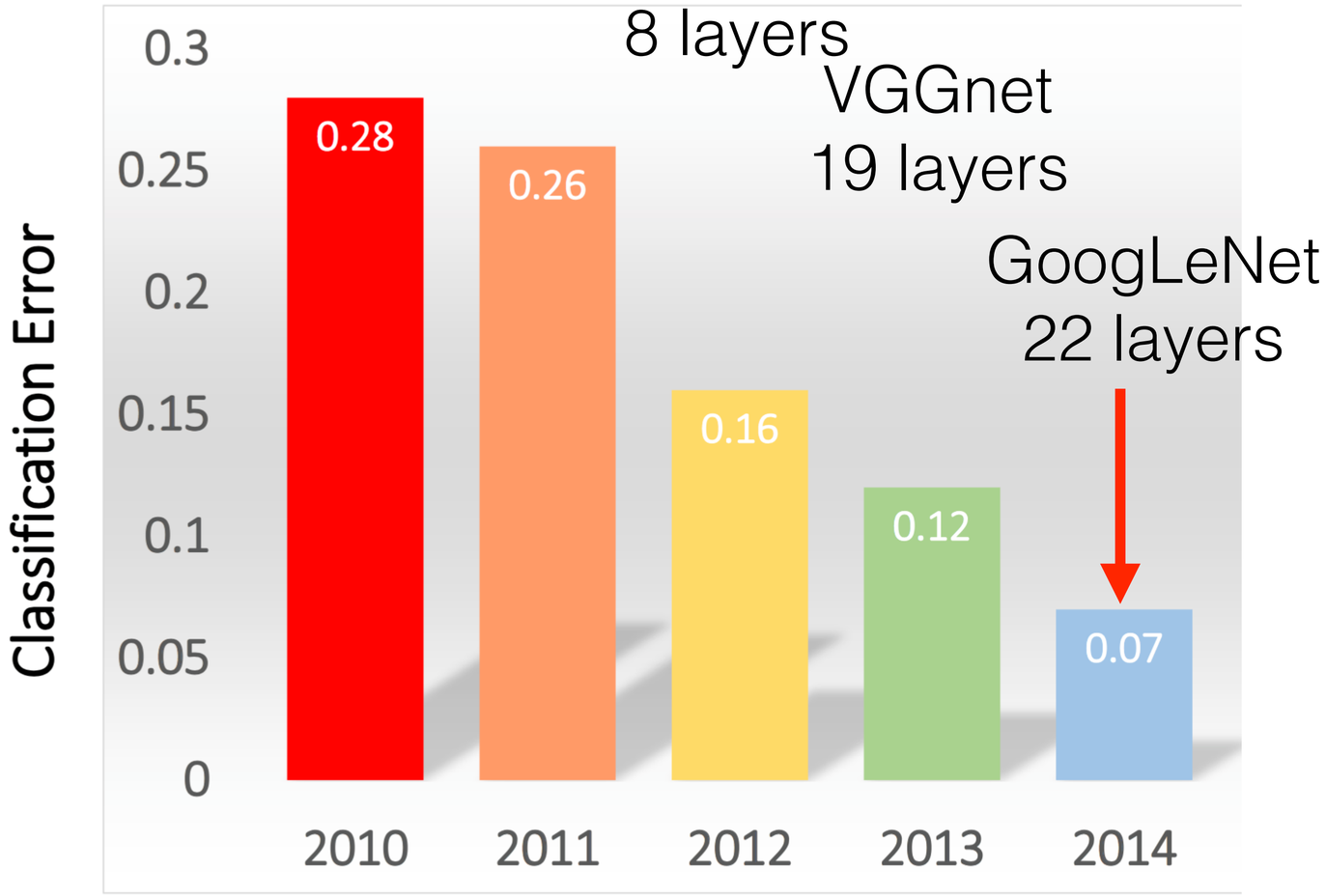
- 12x fewer parameters than AlexNet
- depth 22 layers
- training: few high-end GPU about a week

Szegedy et al. Going Deeper with Convolutions, CVPR, 2014
https://arxiv.org/abs/1409.4842

image source: http://joelouismarino.github.io/images/blog_images/blog_googlenet_keras/googlenet_diagram.png

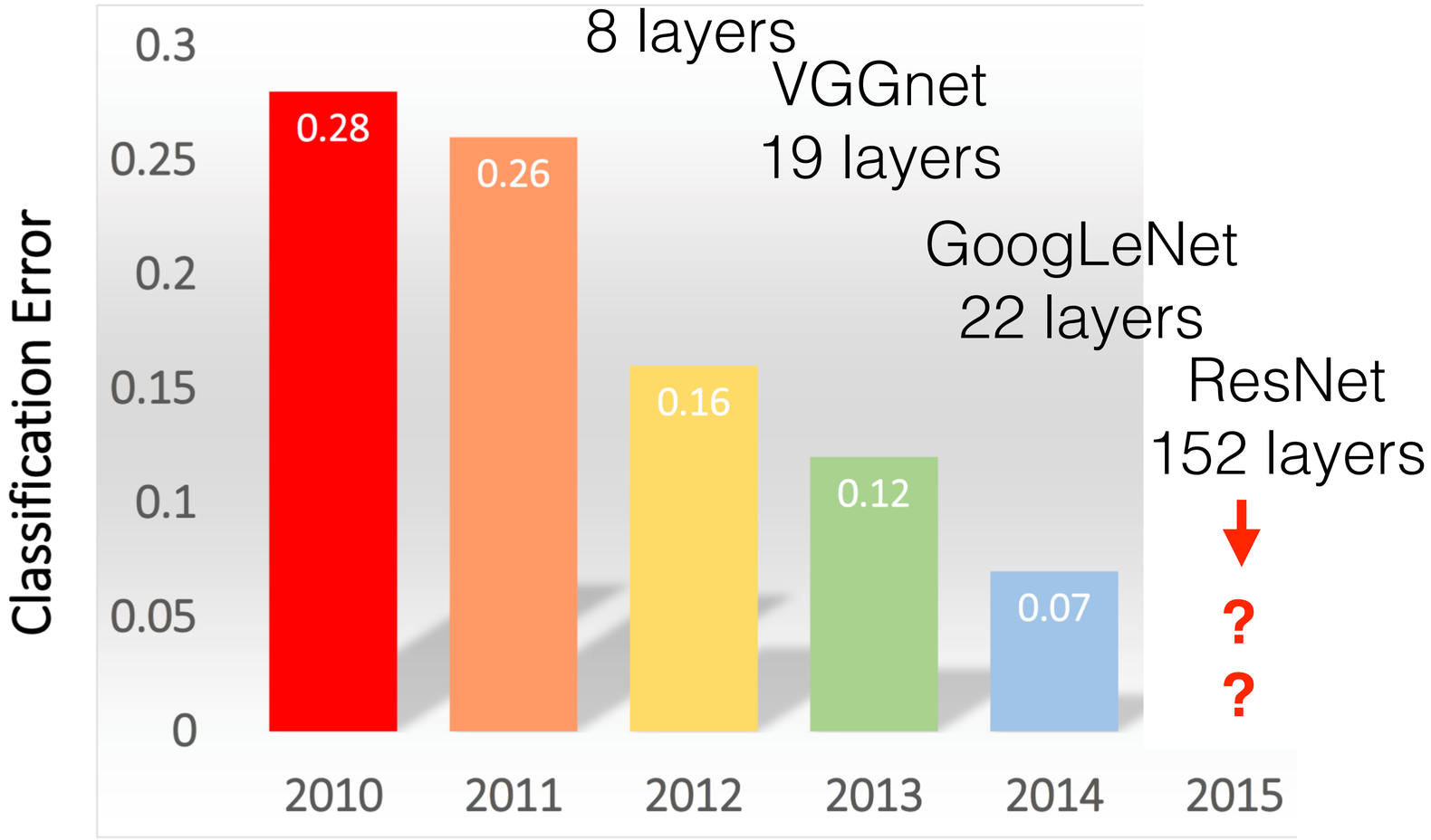
IMAGENET

Classification results



IMAGENET

Classification results



ResNet

The main idea is as follows:

Well said Leo, well said

- deeper ConvNet architectures yielded higher errors.
- error was higher even in training => no overfitting
- problem stems from the optimization (vanishing gradient)

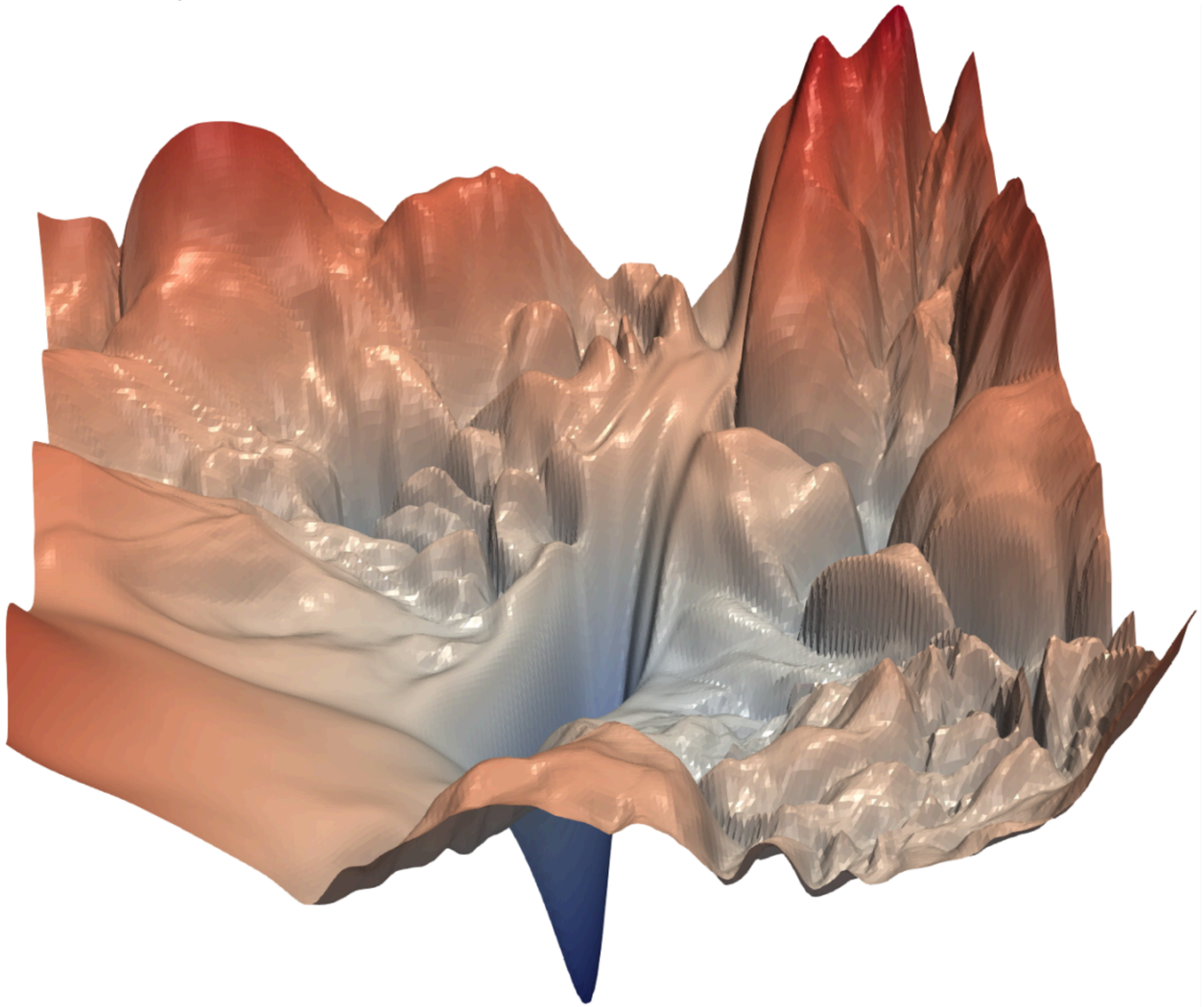
He et al. Going Deeper with Convolutions, CVPR, 2015
https://arxiv.org/abs/1512.03385

Visualizing Loss Landscape of Neural Nets

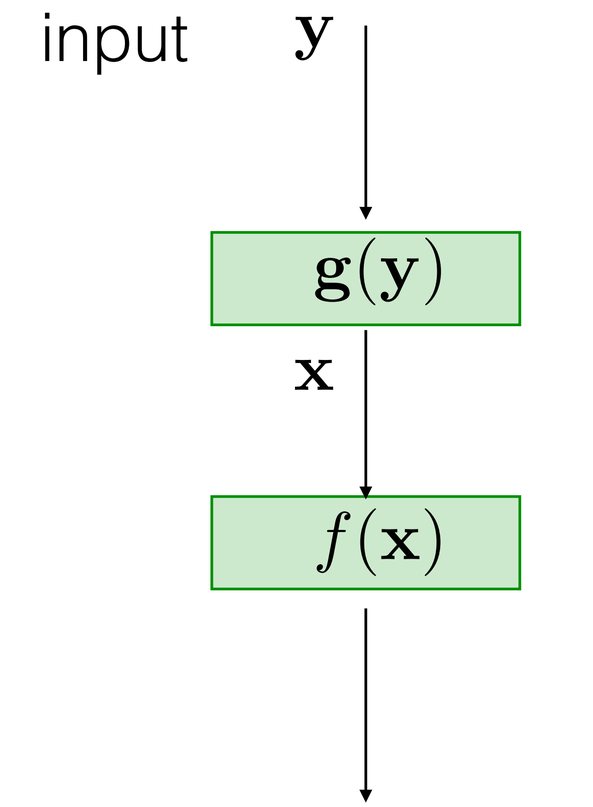
[Li et al, NIPS, 2018] https://arxiv.org/pdf/1712.09913.pdf

$$f(\alpha, \beta) = \mathcal{L}(\mathbf{w}^* + \alpha\mathbf{u} + \beta\mathbf{v})$$

for randomly chosen (and normalized) directions $\mathbf{u}, \mathbf{v}$

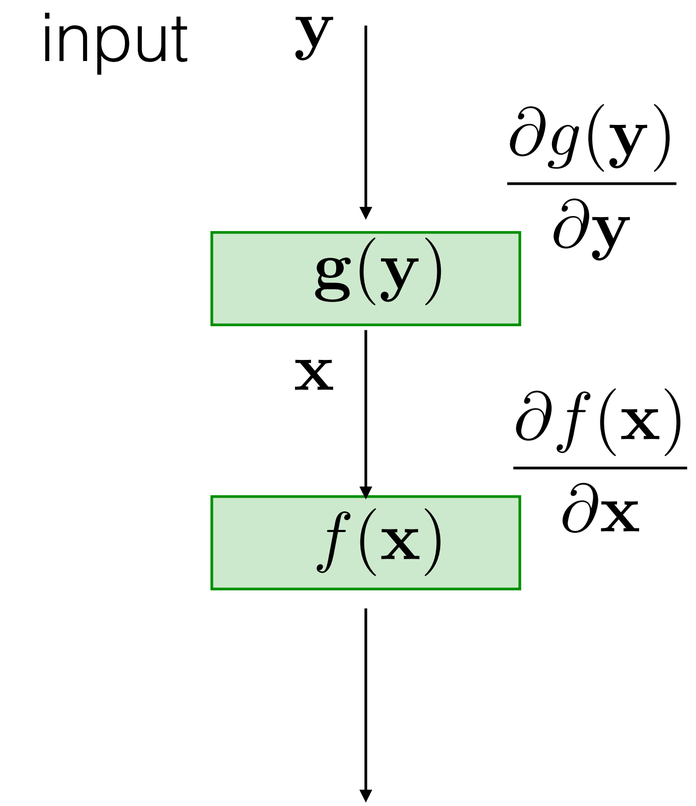


forward pass



output $\mathbf{z} = f(g(\mathbf{y}))$

forward pass:



output $\mathbf{z} = f(g(\mathbf{y}))$

backward pass:

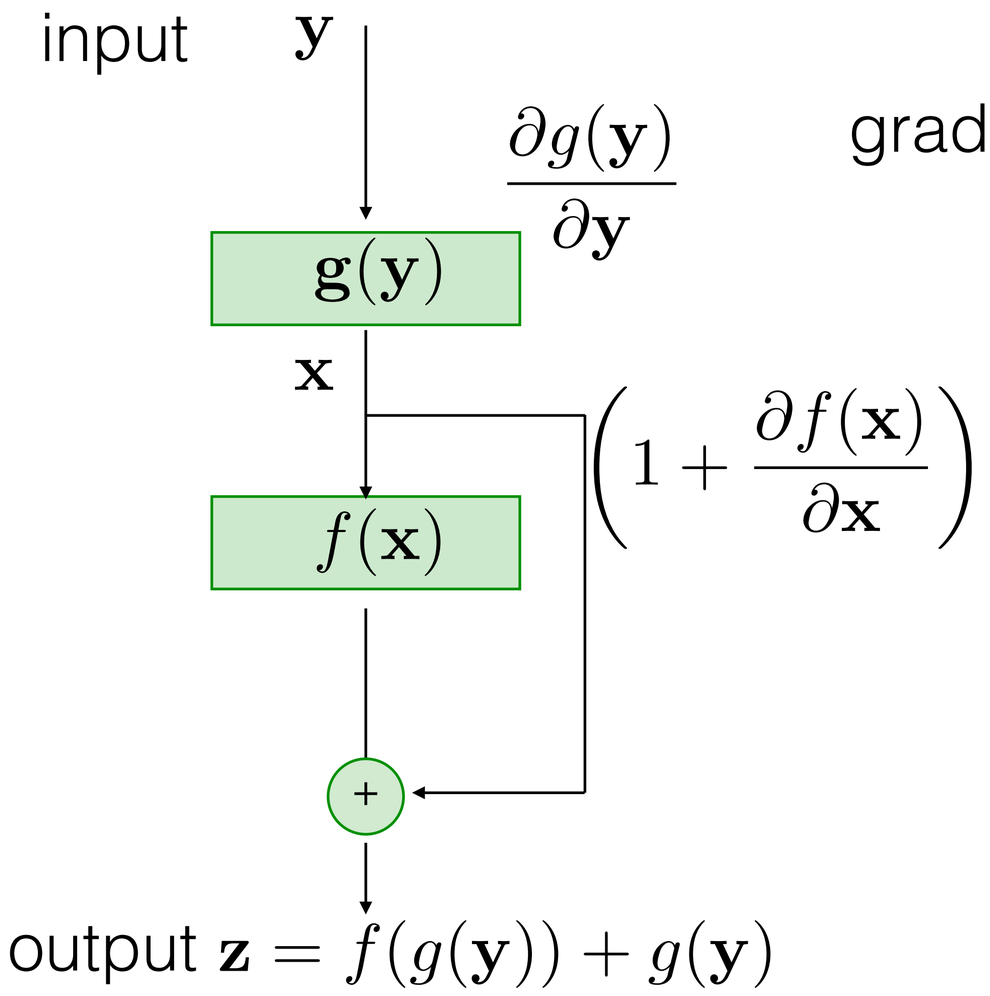
gradient $\frac{\partial \mathbf{z}}{\partial \mathbf{y}} = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \frac{\partial g(\mathbf{y})}{\partial \mathbf{y}}$

if $\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \approx \mathbf{0}$

then gradient is always zero

$$\frac{\partial \mathbf{z}}{\partial \mathbf{y}} \approx \mathbf{0}$$

forward pass:



backward pass:

gradient $$\frac{\partial \mathbf{z}}{\partial \mathbf{y}} = \left(1 + \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right) \frac{\partial g(\mathbf{y})}{\partial \mathbf{y}}$$

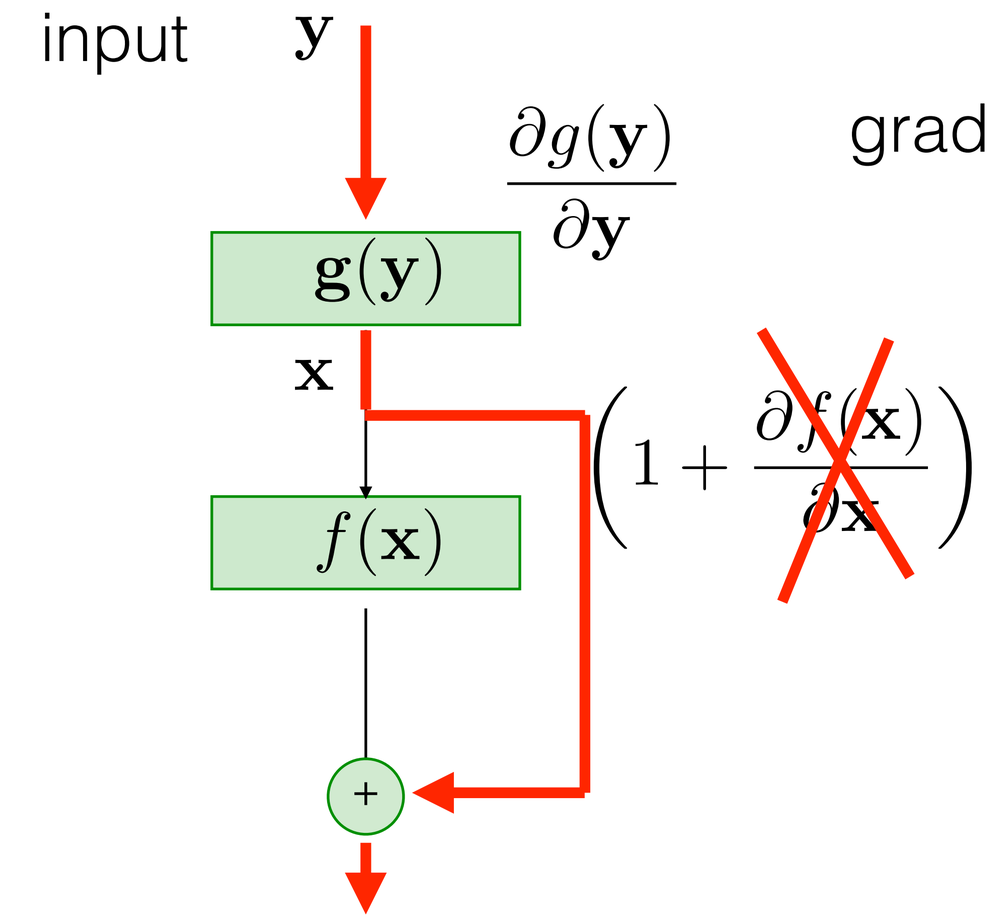
if $$\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \approx \mathbf{0}$$

then gradient is

$$\frac{\partial \mathbf{z}}{\partial \mathbf{y}} \approx \frac{\partial \mathbf{g}(\mathbf{y})}{\partial \mathbf{y}}$$

forward pass:

backward pass:



$$\frac{\partial g(\mathbf{y})}{\partial \mathbf{y}}$$

$$\left(1 + \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right)$$

gradient $$\frac{\partial \mathbf{z}}{\partial \mathbf{y}} = \left(1 + \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right) \frac{\partial g(\mathbf{y})}{\partial \mathbf{y}}$$

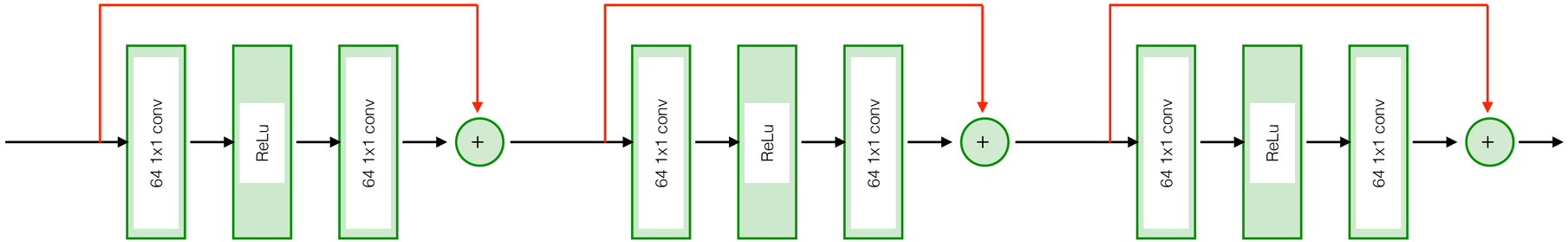
if $$\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \approx \mathbf{0}$$

then gradient is

$$\frac{\partial \mathbf{z}}{\partial \mathbf{y}} \approx \frac{\partial \mathbf{g}(\mathbf{y})}{\partial \mathbf{y}}$$

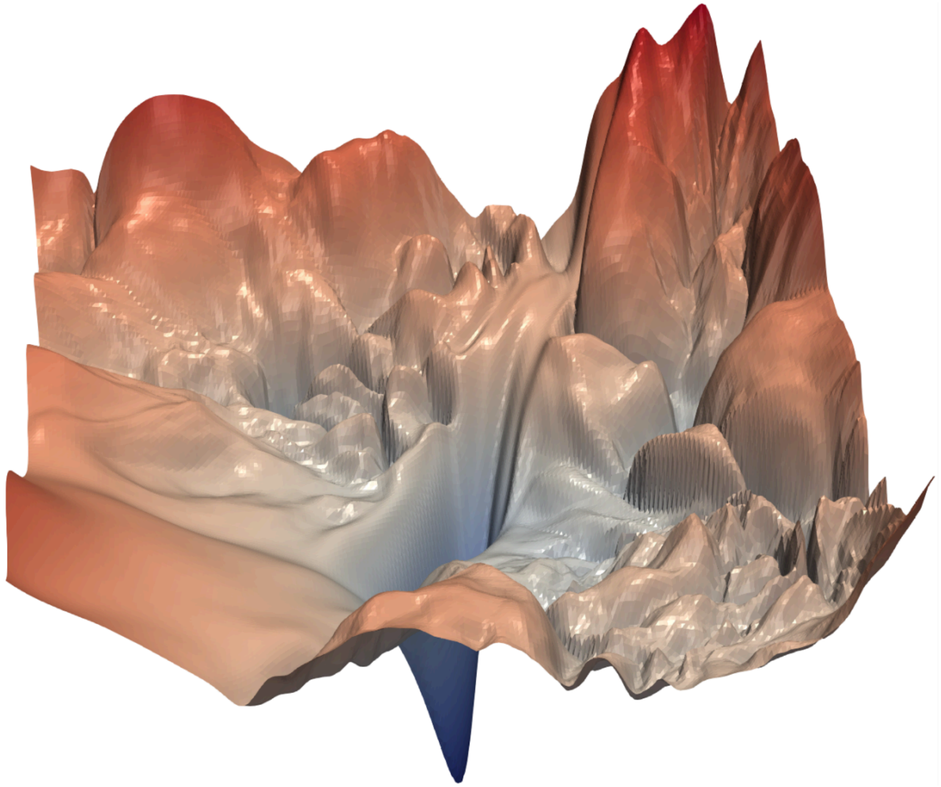
Input values can still influence the output via skip connection !

ResNet - gradient flow

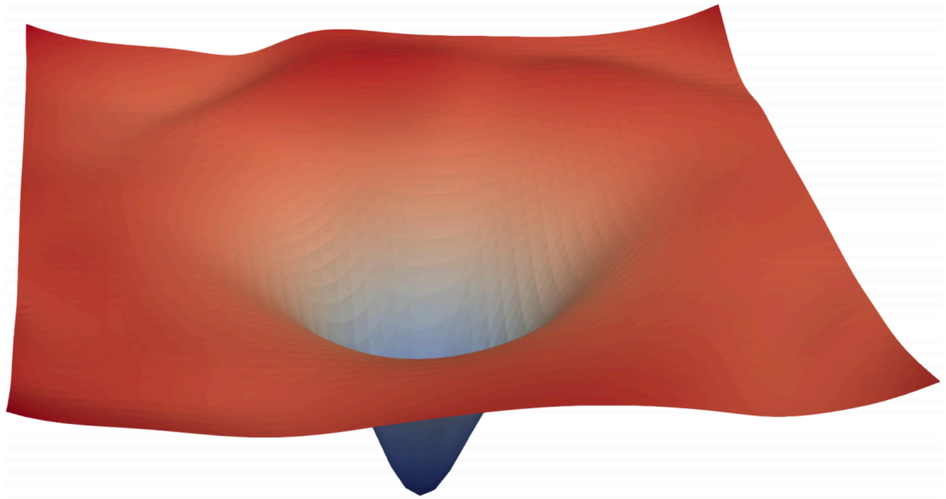


- Skip connections partially avoids diminishing gradient
- The weights from the beginning of the net has strong influence on the output! (via red skip-connections)

Visualizing Loss Landscape of Neural Nets
[Li et al, NIPS, 2018] https://arxiv.org/pdf/1712.09913.pdf

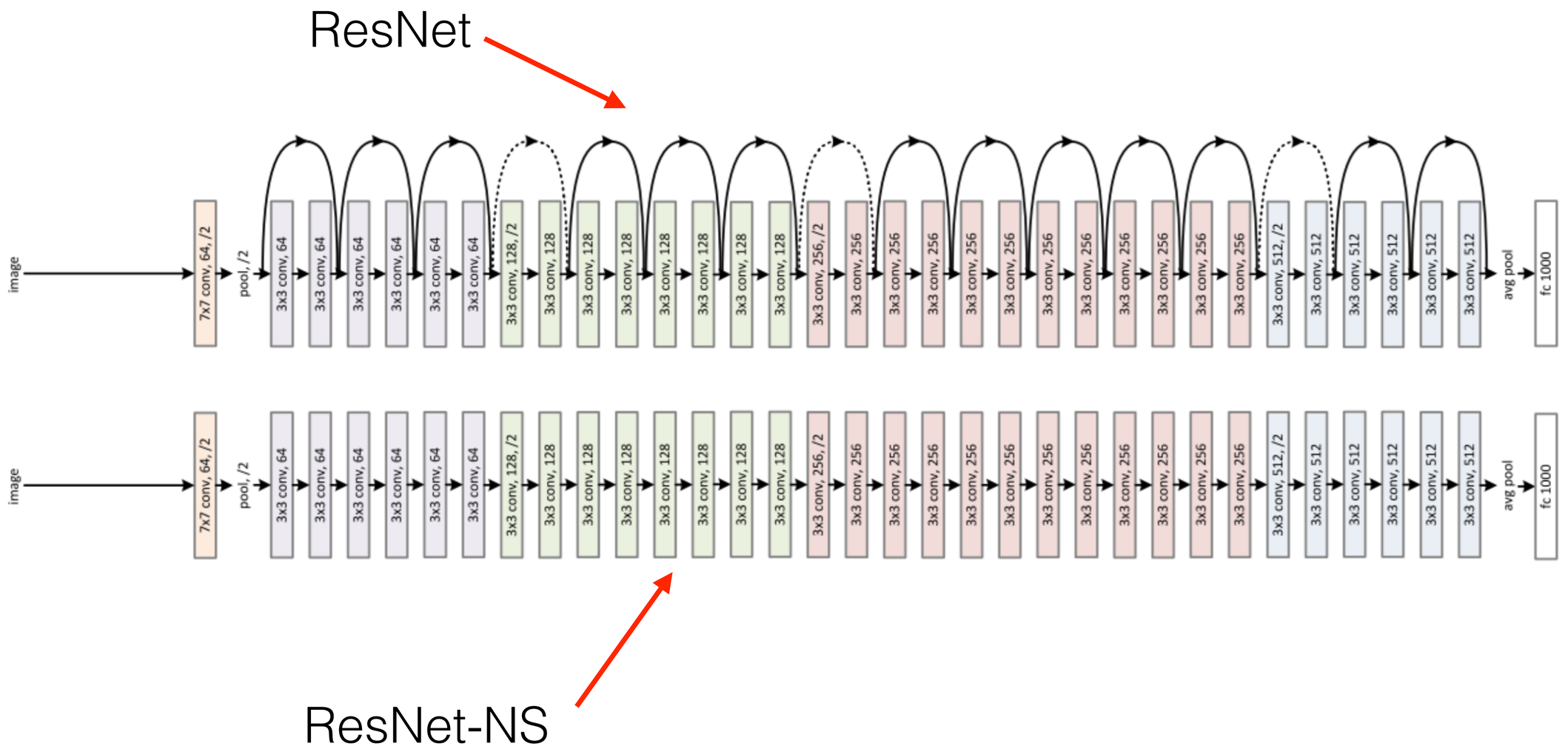


(a) without skip connections



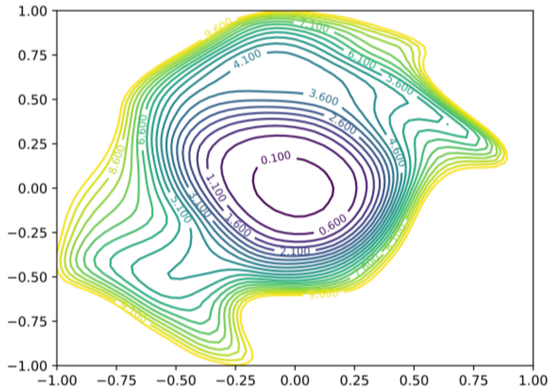
(b) with skip connections

ResNet: deep ConvNet with skip connections

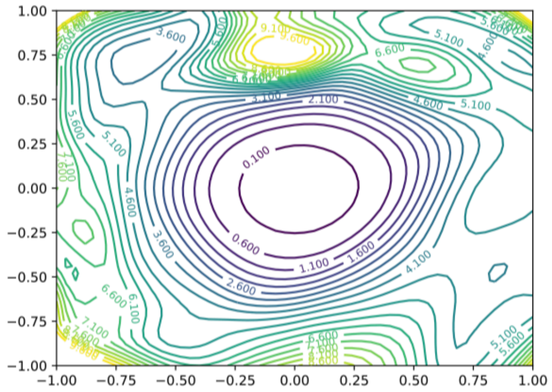


ResNet: deep ConvNet with skip connections
[Li et al, NIPS, 2018] https://arxiv.org/pdf/1712.09913.pdf

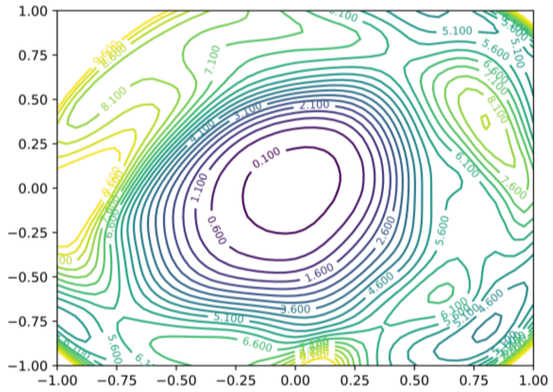
ResNet



(a) ResNet-20, 7.37%

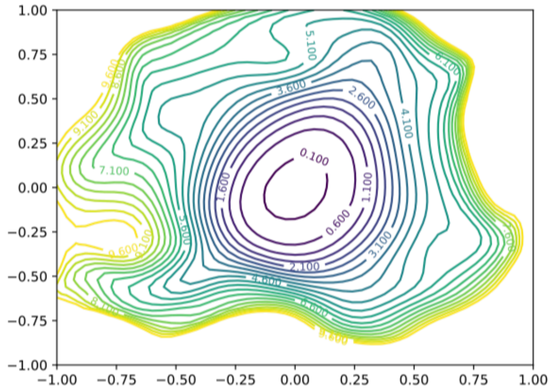


(b) ResNet-56, 5.89%

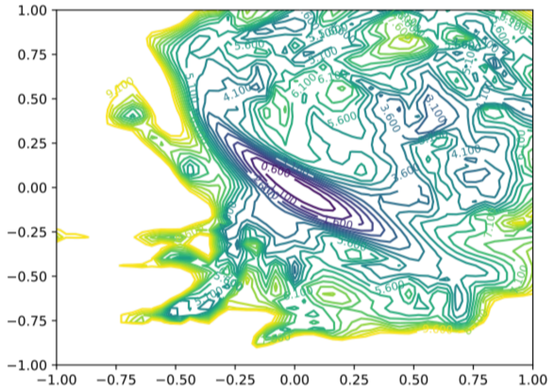


(c) ResNet-110, 5.79%

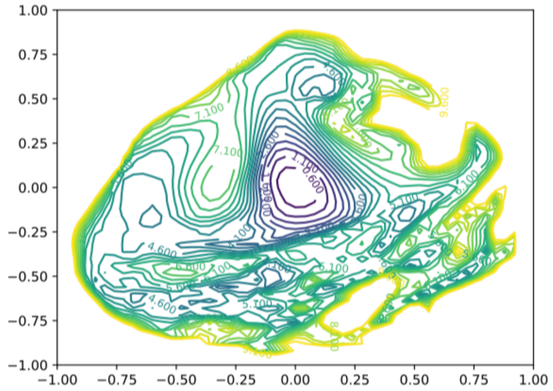
ResNet-NS



(d) ResNet-20-NS, 8.18%

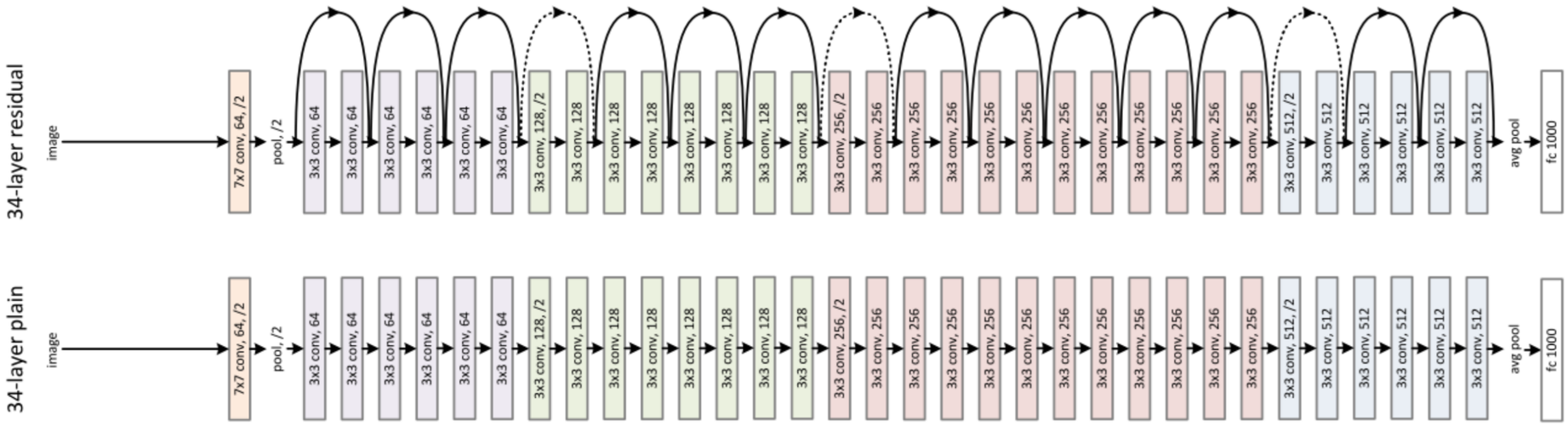


(e) ResNet-56-NS, 13.31%



(f) ResNet-110-NS, 16.44%

ResNet: deep ConvNet with skip connections



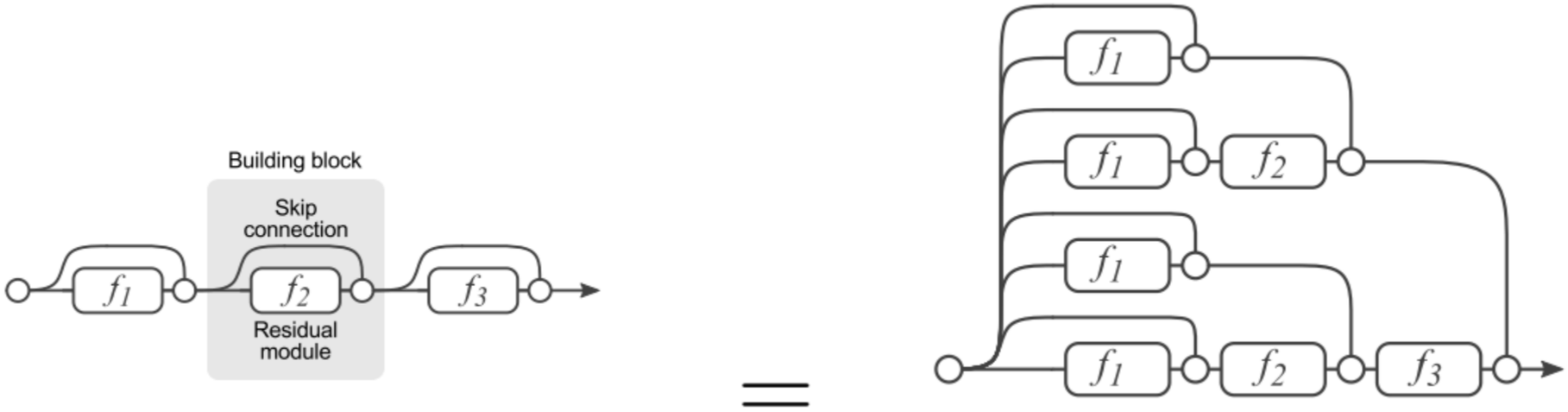
- Competition time about 152 layers ResNet,
- Recently they are able to train 1k layers ResNet
- Initialization with zero weights is meaningful
- Better gradient flow (many independent paths)
- Robustness wrt noise and layer removal

https://www.kaggle.com/keras/resnet50/home

He et al. Going Deeper with Convolutions, CVPR, 2015

https://arxiv.org/abs/1512.03385

Unraveled view of ResNet

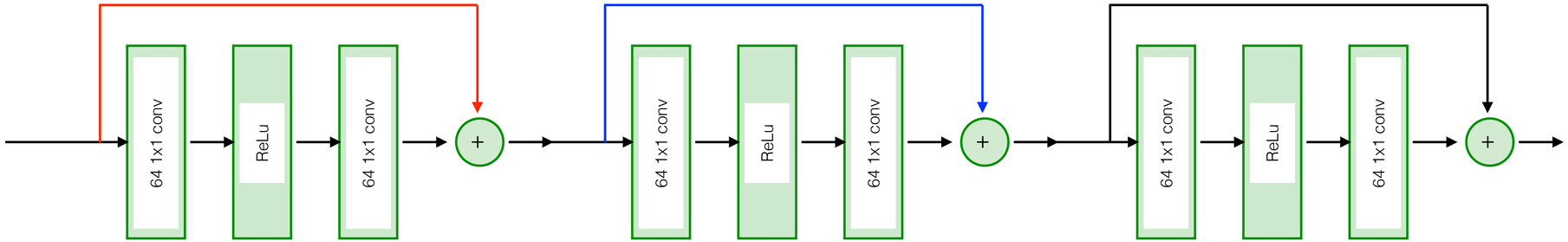


(a) Conventional 3-block residual network

(b) Unraveled view of (a)

- There exists many “almost independent” paths
- Unravelling of ResNet architecture allows to understand robustness wrt noise and layer removal

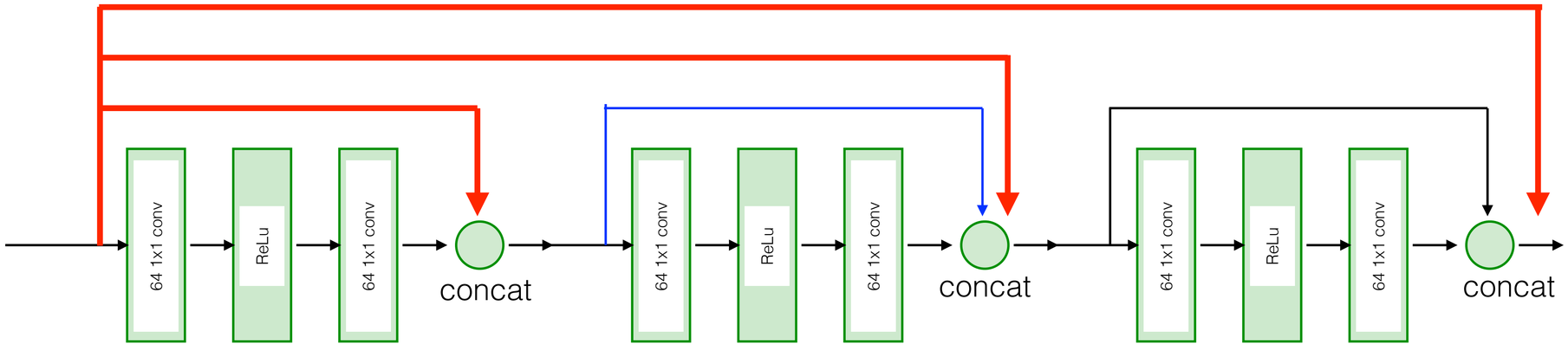
ResNet =>DenseNet



Start with multilayer ResNet architecture

Huang, Densely Connected Convolutional Networks, CVPR 2017. https://arxiv.org/abs/1608.06993

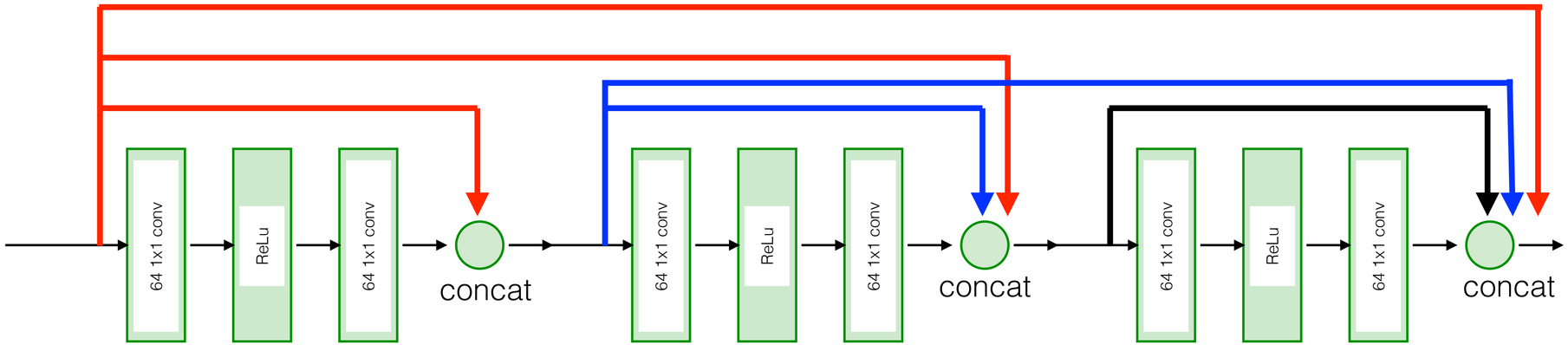
DenseNet



- Directly propagate each feature map to all following layers

Huang, Densely Connected Convolutional Networks, CVPR 2017. https://arxiv.org/abs/1608.06993

DenseNet

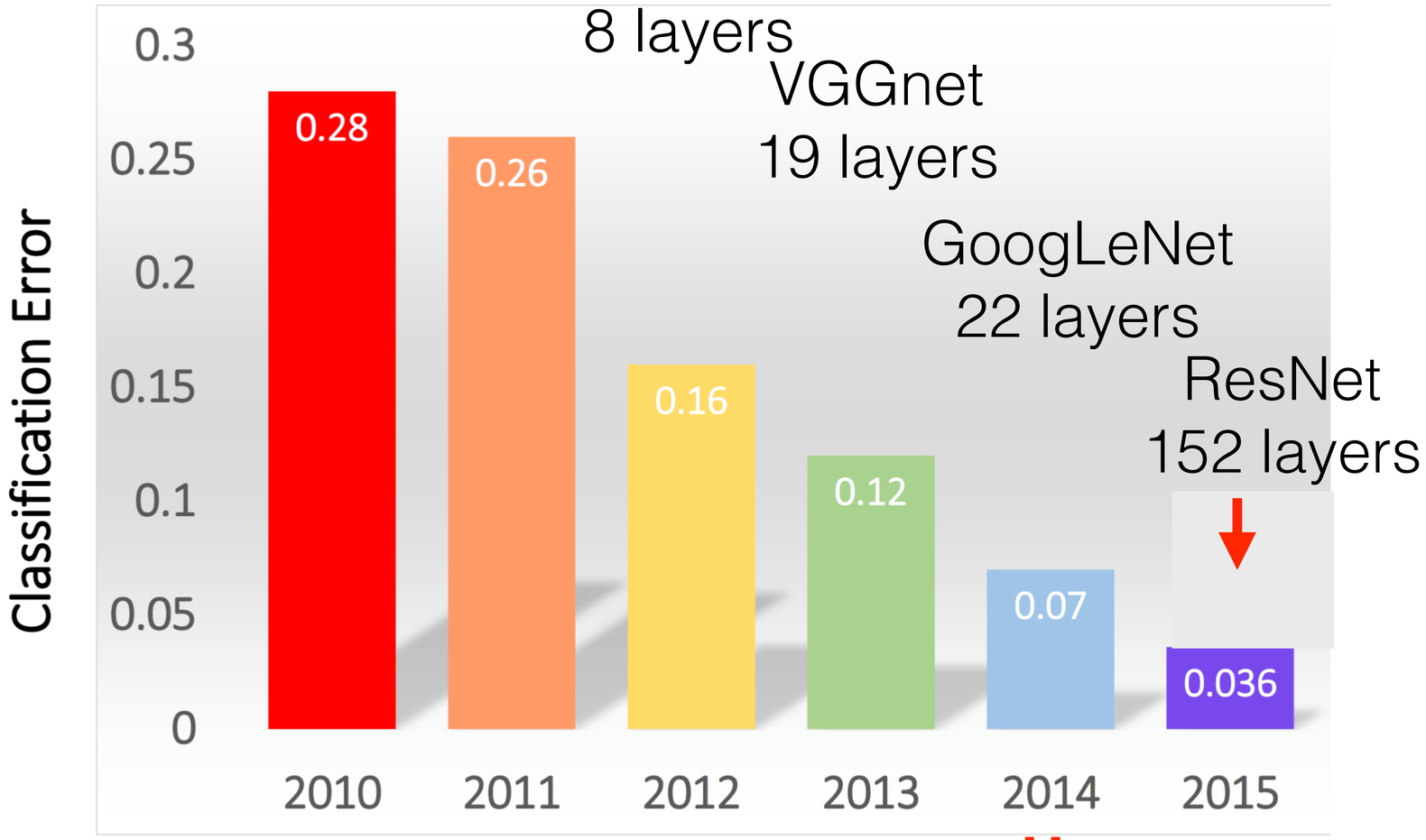


- Directly propagate each feature map to all following layers
- Improves gradient flow in backward pass

Huang, Densely Connected Convolutional Networks, CVPR 2017. https://arxiv.org/abs/1608.06993

IMAGENET

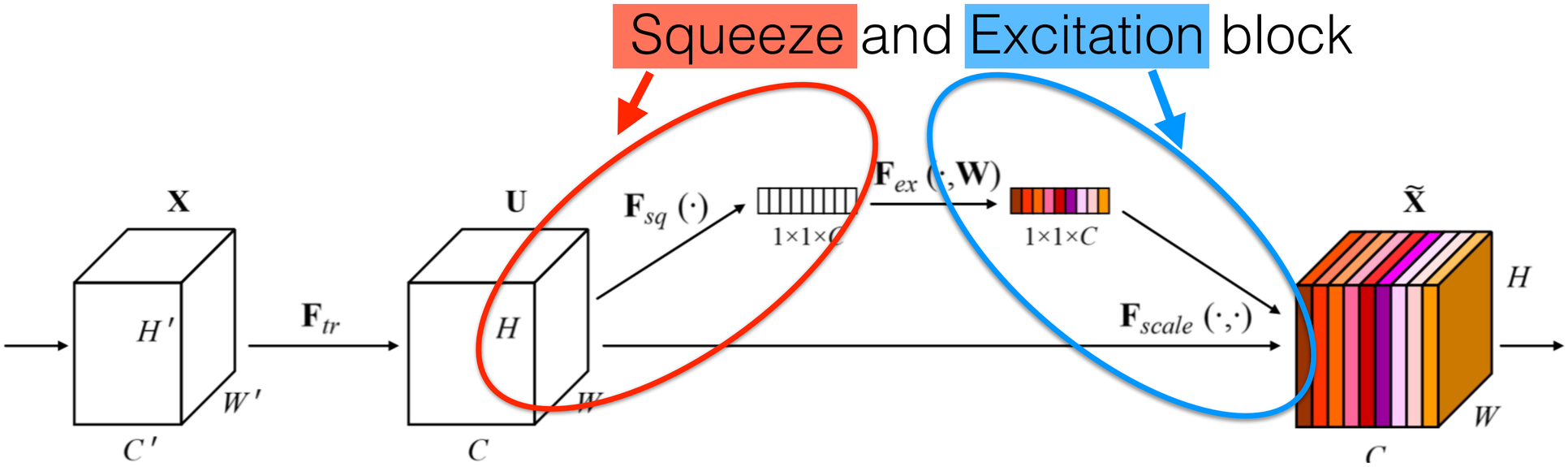
Classification results



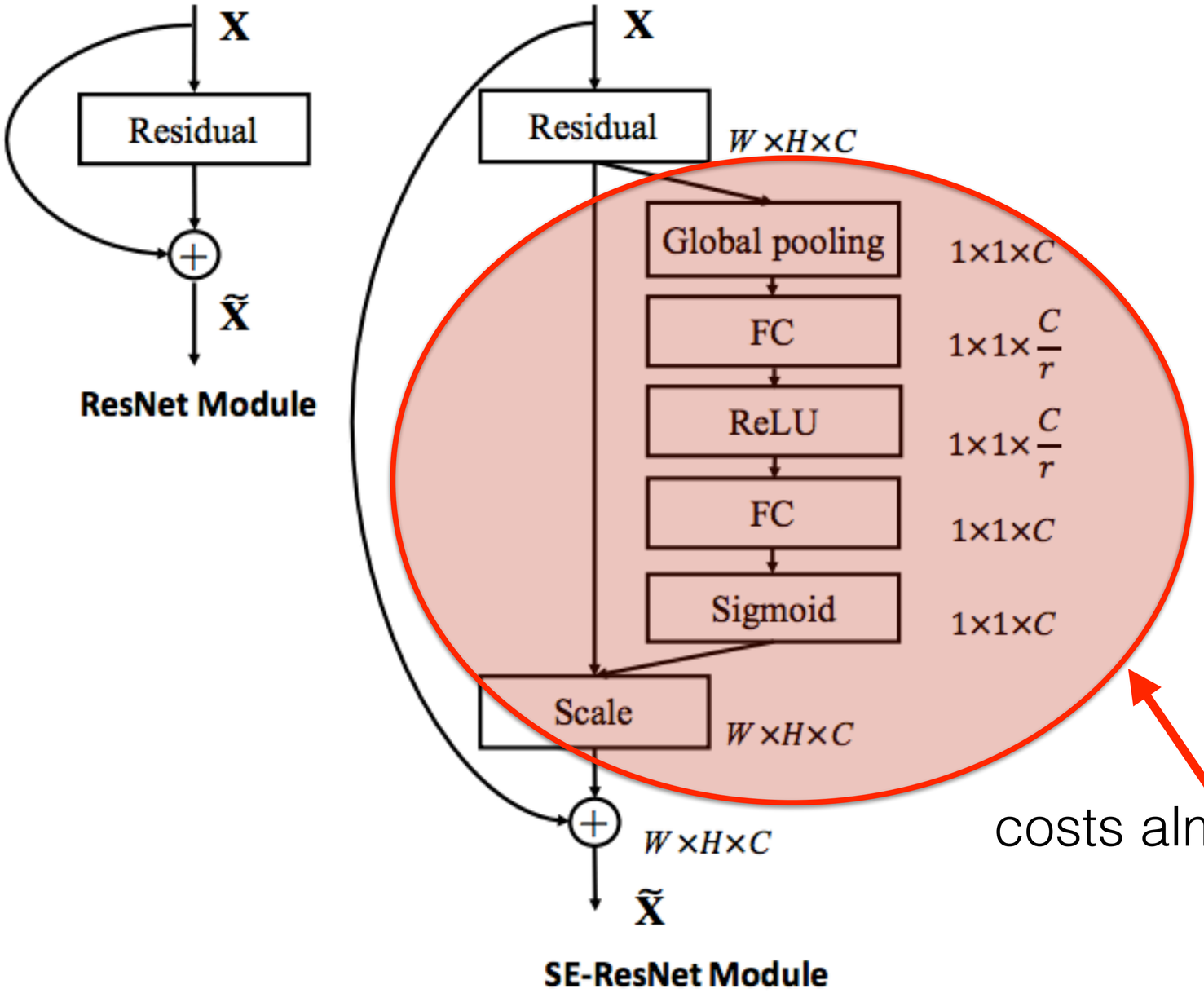
Human error around 5%

Squeeze and Excitation Networks [Hu et al, CVPR oral, 2017]
https://arxiv.org/pdf/1709.01507.pdf

- Winner of ILSVRC 2017
- Enhancement of ResNet, InceptionNet and DenseNet architectures by SE blocks consistently decrease error on ImageNet, COCO, …

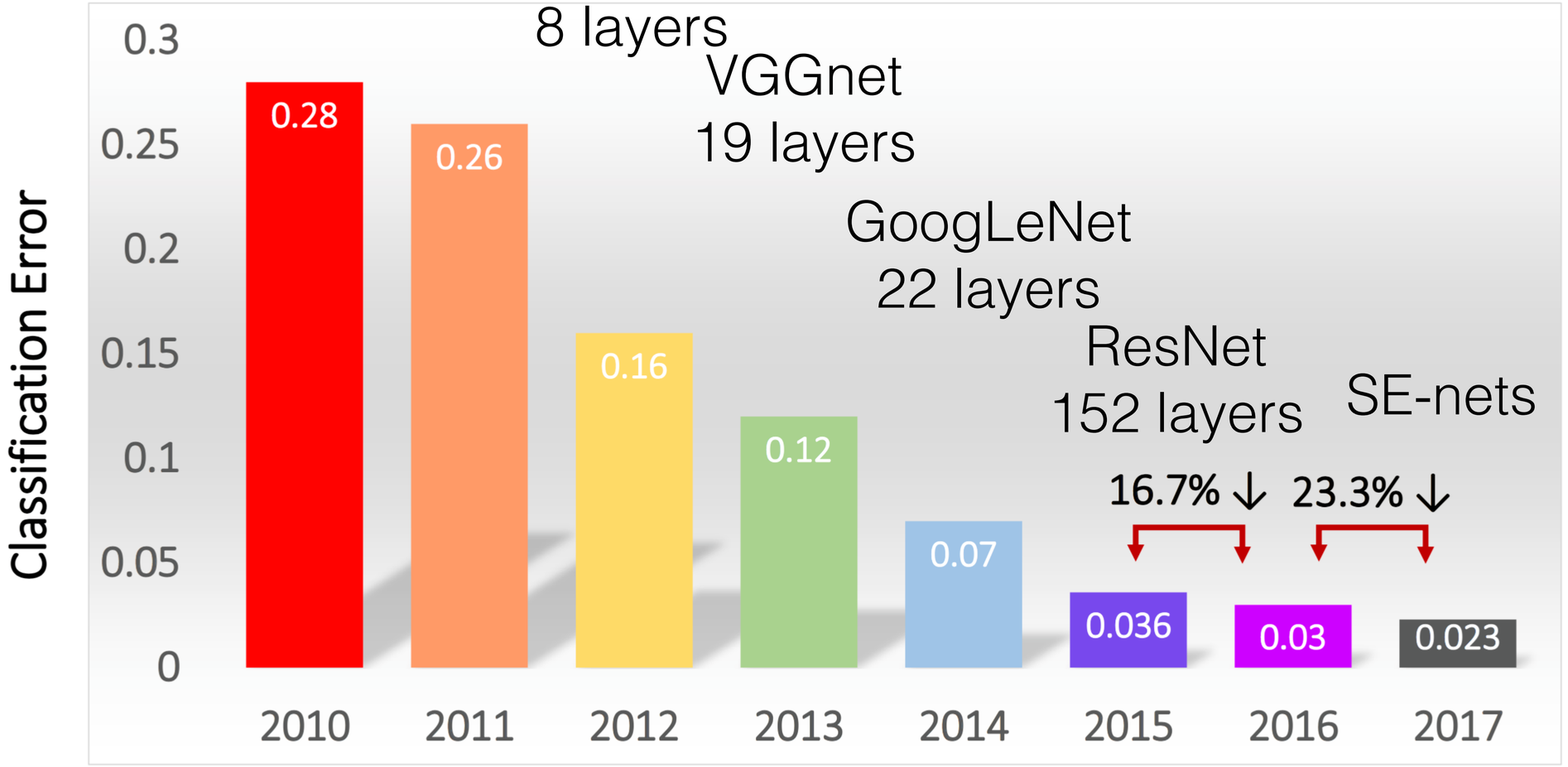


Squeeze and Excitation Networks [Hu et al, CVPR oral, 2017]
https://arxiv.org/pdf/1709.01507.pdf



IMAGENET

Classification results



Summary classification architectures

- It seems that the deeper the better
- ResNet-50/ResNet-101 is easy, well-studied architecture
  => consider as a starting point
- You should be careful about combining DropOut with BN
  https://arxiv.org/abs/1801.05134

Outline

- Architectures of classification networks
- Architectures of segmentation networks
- Architectures of regression networks
- Architectures of detection networks
- Architectures of regression networks
- Architectures of feature matching networks

Semantic segmentation

road

sideway

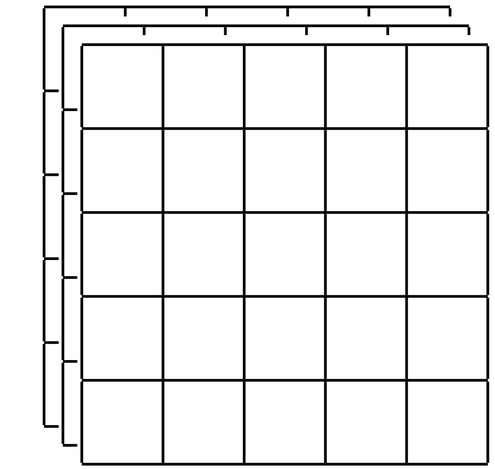
pedestrian

traffic sign

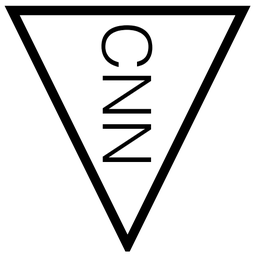
trees

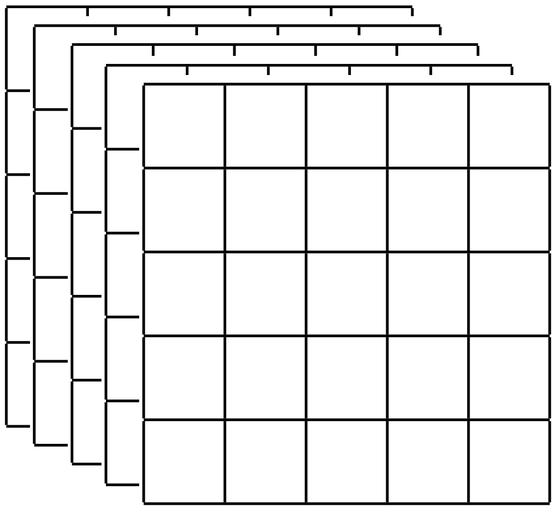
sky

Semantic segmentation



RGB image
(HxWx3)

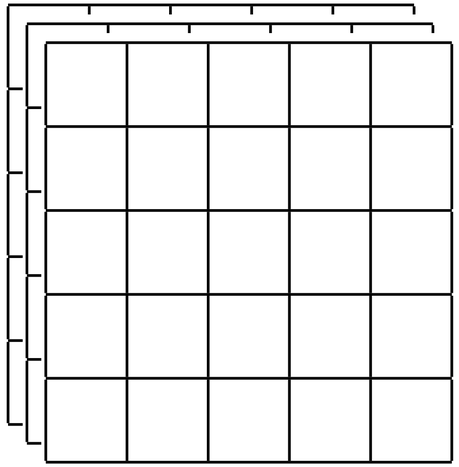




labels
(HxWxN)

- road
- sideway
- pedestrian
- traffic sign
- trees
- sky

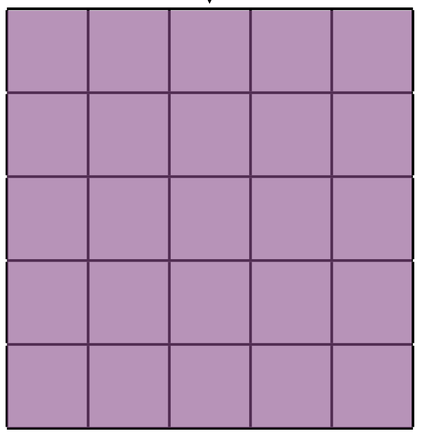
Semantic segmentation



CNN

RGB image
(HxWx3)

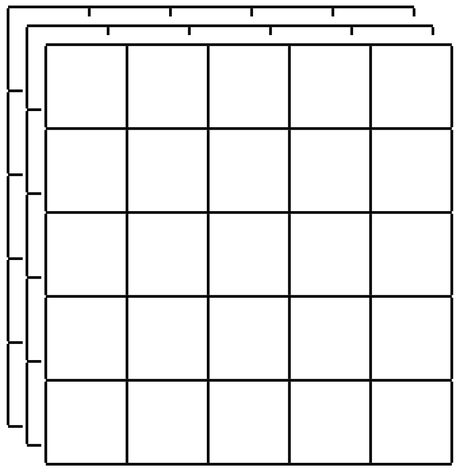
road
sideway
pedestrian
traffic sign
trees
sky



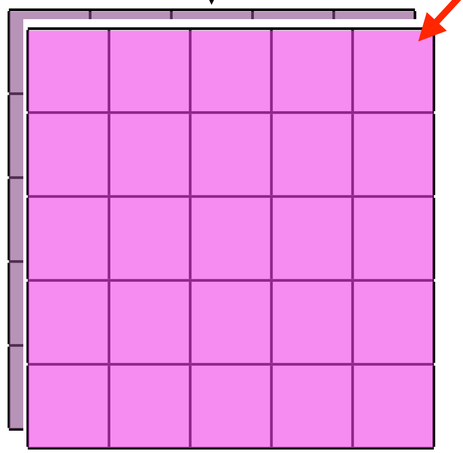
pixel-wise probability
of being **road**

channel 1

Semantic segmentation



CNN



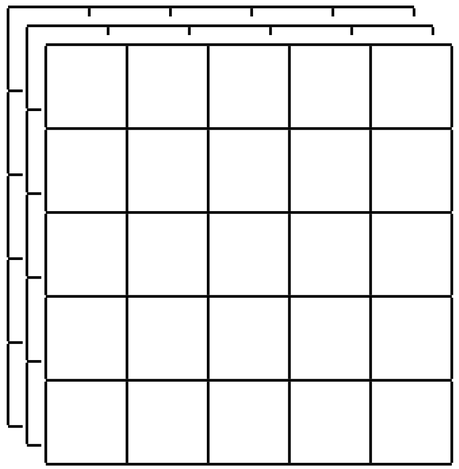
RGB image
(HxWx3)

road
sideway
pedestrian
traffic sign
trees
sky

pixel-wise probability
of being **sideway**

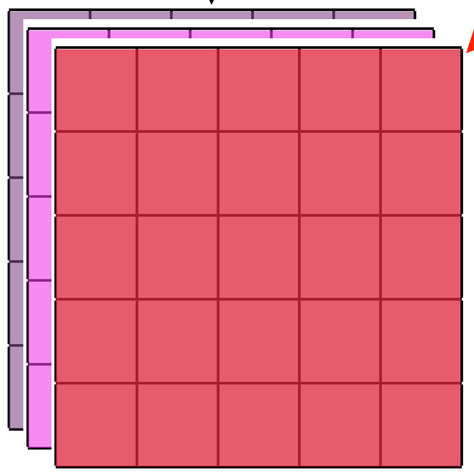
channel 2

Semantic segmentation



RGB image
(HxWx3)

CNN

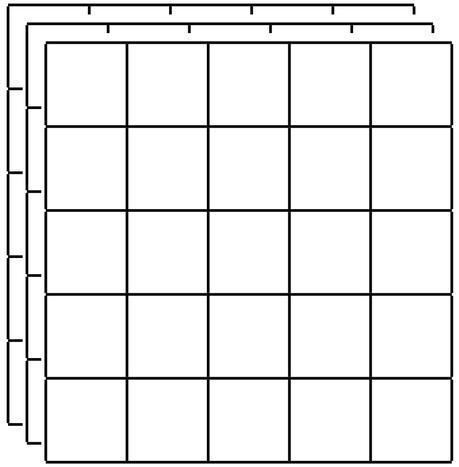


pixel-wise probability
of being **pedestrian**

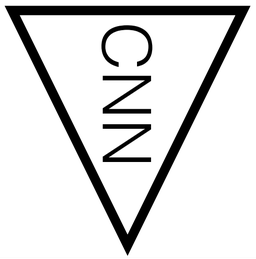
channel 3

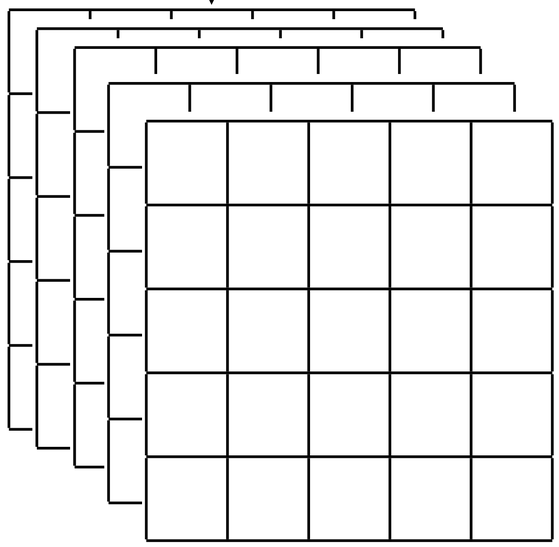
- road
- sideway
- pedestrian
- traffic sign
- trees
- sky

Semantic segmentation



RGB image
(HxWx3)

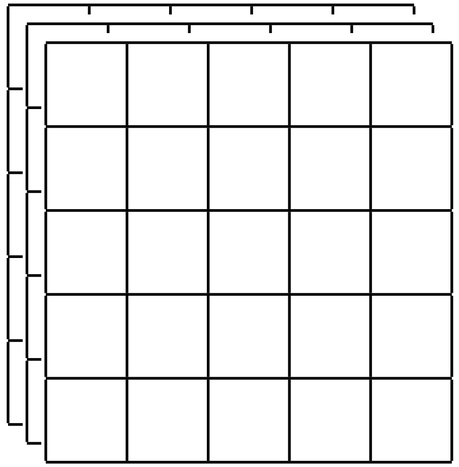




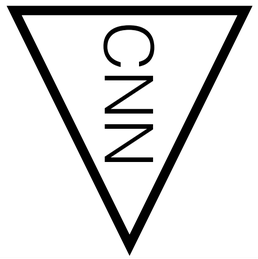
as many output channels
as semantic labels

- road
- sideway
- pedestrian
- traffic sign
- trees
- sky

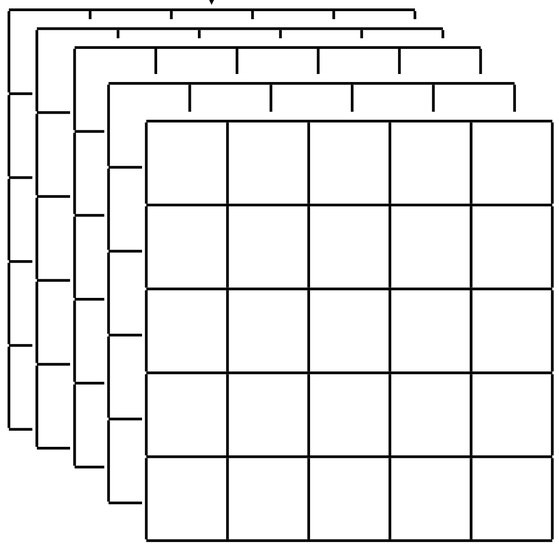
Semantic segmentation

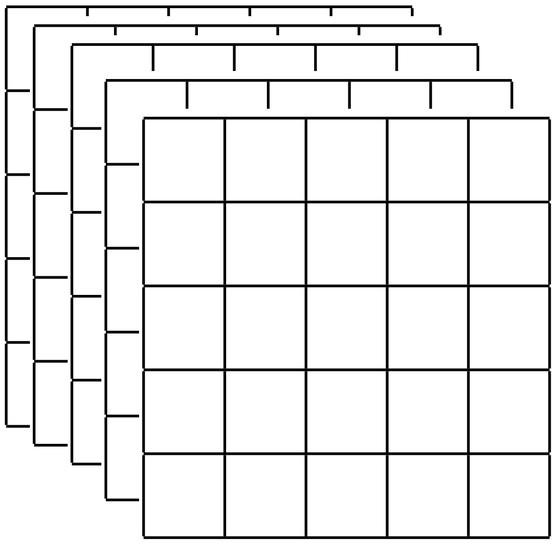


RGB image
(HxWx3)

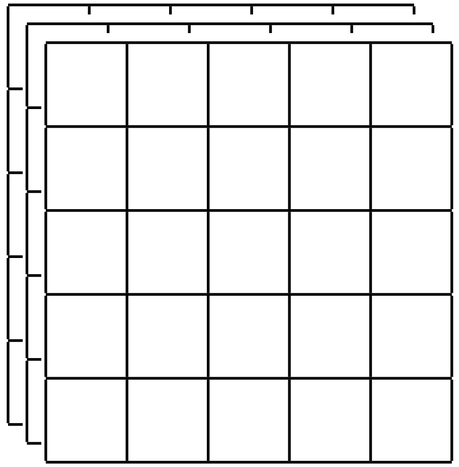


ground truth (1-hot encoding)

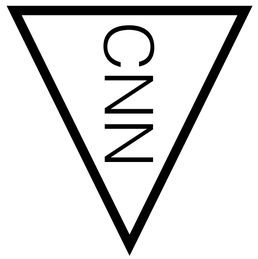




Semantic segmentation

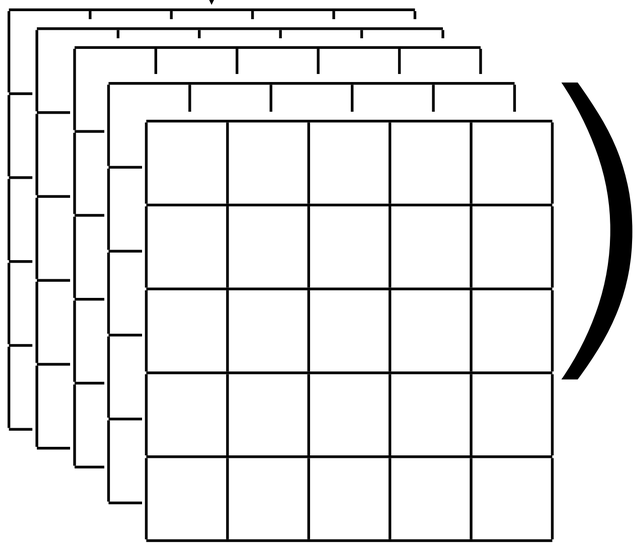


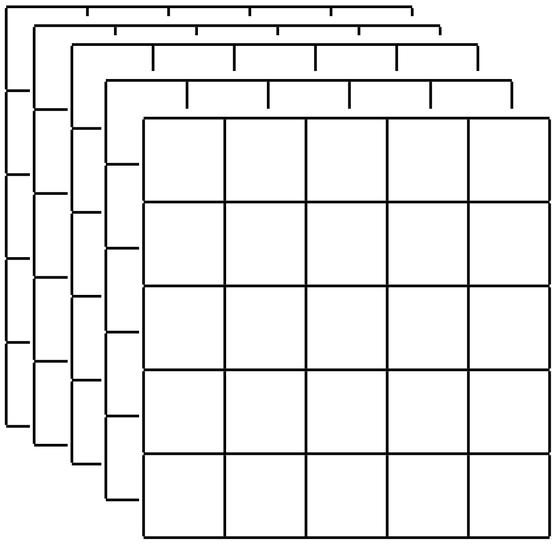
RGB image
(HxWx3)



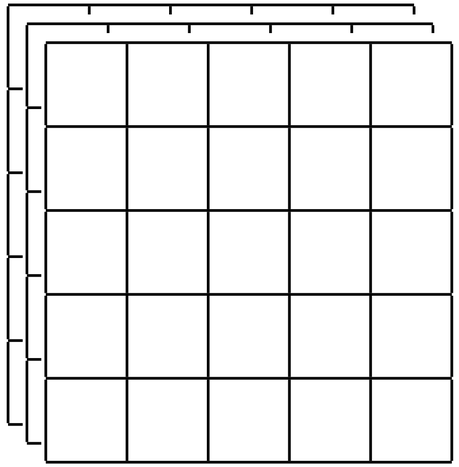
ground truth (1-hot encoding)

$$-\log\Big(\quad\Big)\odot$$





Semantic segmentation



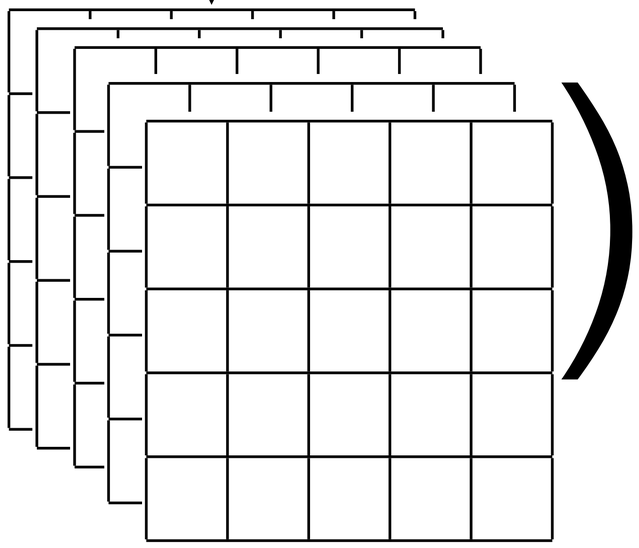
RGB image
(HxWx3)

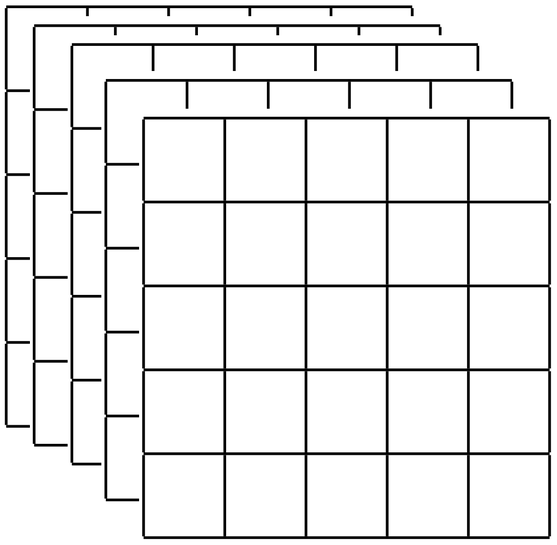
CNN

cross-entropy loss

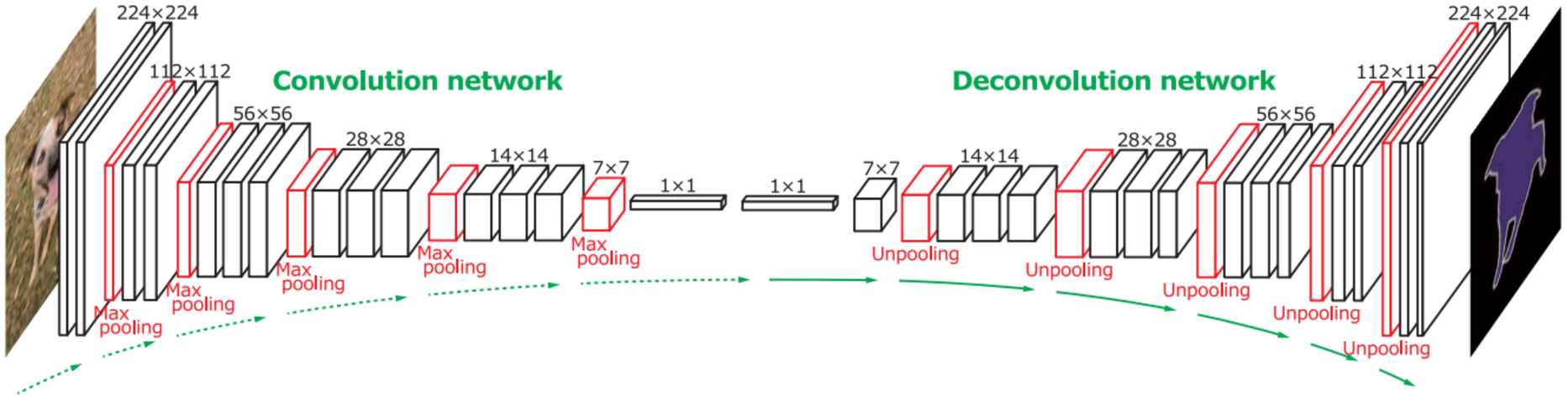
ground truth (1-hot encoding)

$$\sum_{\text{pixels}} -\log\Big(\quad\Big) \odot$$





Semantic segmentation

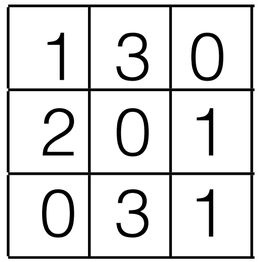
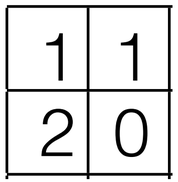


- Loss: cross entropy/focal loss summed over all pixels
- Convolution layers:
  - decrease spatial resolution
  - increase number of channels
- Deconvolution layers: exactly opposite

[Noh et al ICCV 2015] https://arxiv.org/pdf/1505.04366.pdf

Transposed convolution

image (3x3) kernel (2x2)

trconv ( , ) =

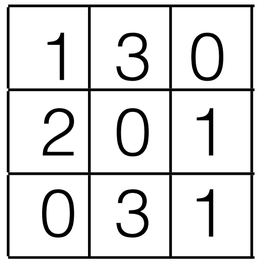
1	3	0
2	0	1
0	3	1

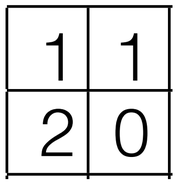
1	1
2	0

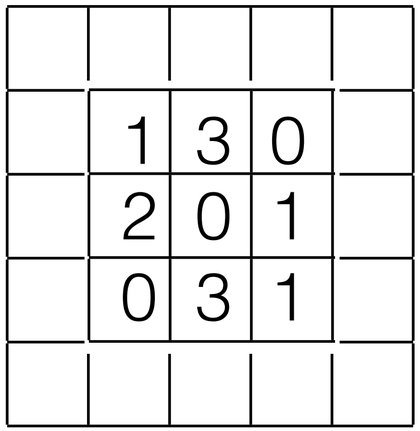
stride=1,
padding=0

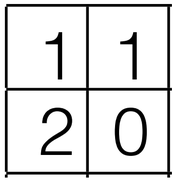
Transposed convolution

image (NxN) | kernel (KxK) | image (N+K-1)x(N+K-1) | kernel (KxK)

trconv (

, 

) = conv (

, 

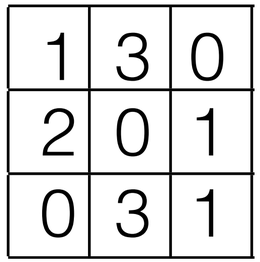
) =

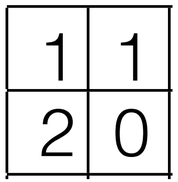
stride=1, padding=0

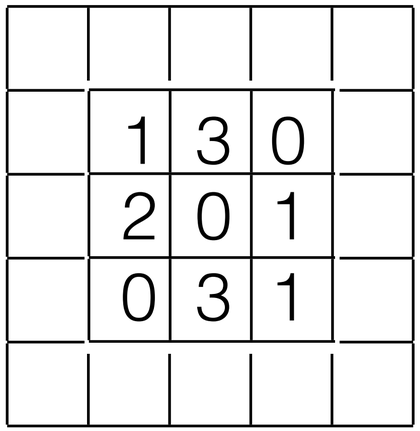
stride=1, padding=K-1

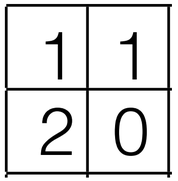
Transposed convolution

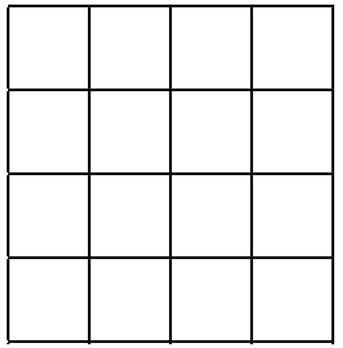
image (NxN) | kernel (KxK) | image (N+K-1)x(N+K-1) | kernel (KxK) | output (4x4)

trconv (

, 

) = conv (

, 

) = 

stride=1, padding=0

stride=1, padding=K-1

Transposed convolution

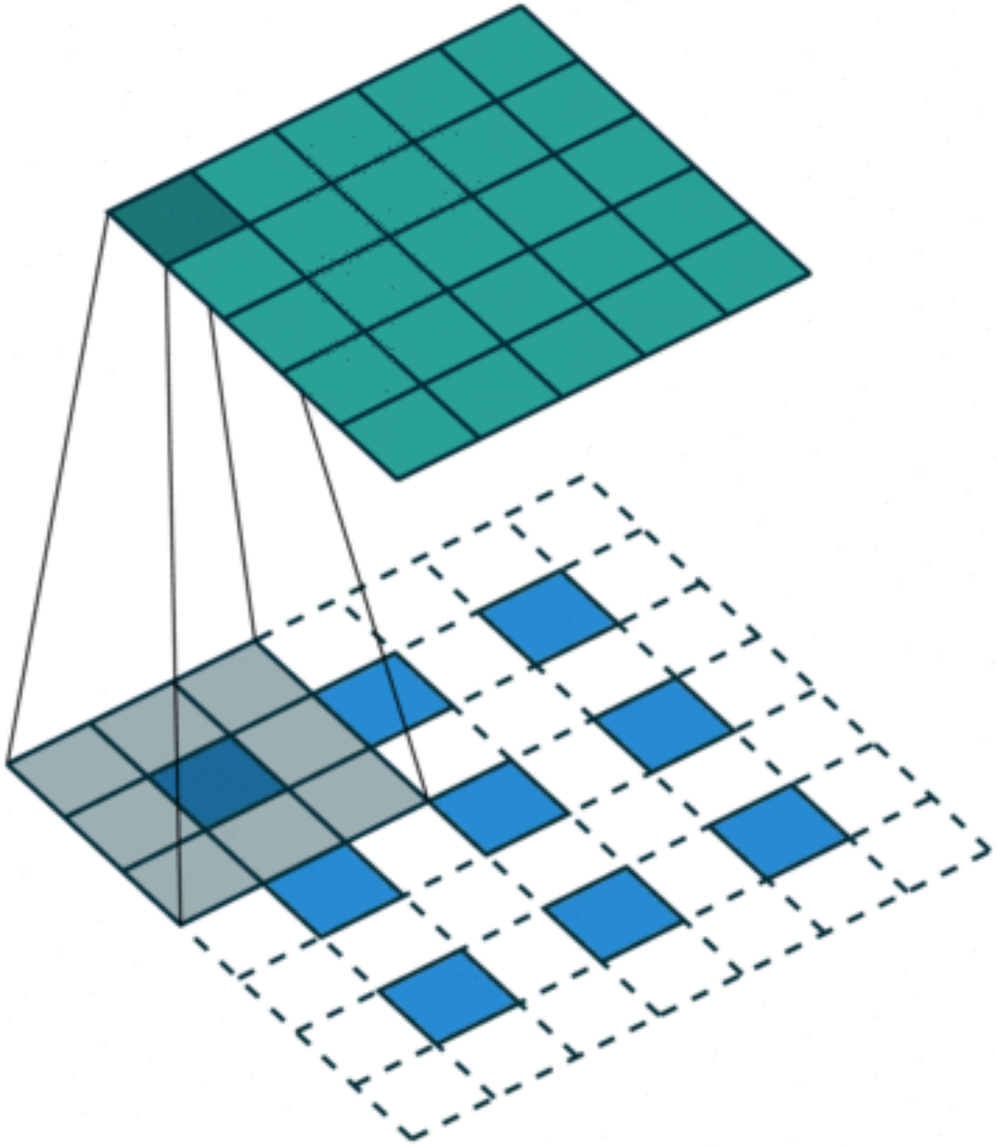
image (NxN) | kernel (KxK) | image ((N-1)*K+K-1)x((N-1)*K+K-1) | kernel (KxK) | output (6x6)

trconv ($\begin{bmatrix} 1 & 3 & 0 \\ 2 & 0 & 1 \\ 0 & 3 & 1 \end{bmatrix}$, $\begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix}$) = conv ($\begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & 3 & \cdot & 0 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 2 & \cdot & 0 & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 0 & \cdot & 3 & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$, $\begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix}$) = $\begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$

stride=2, padding=0

stride=1, padding=K-1

Transposed convolution



https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md

unpooling

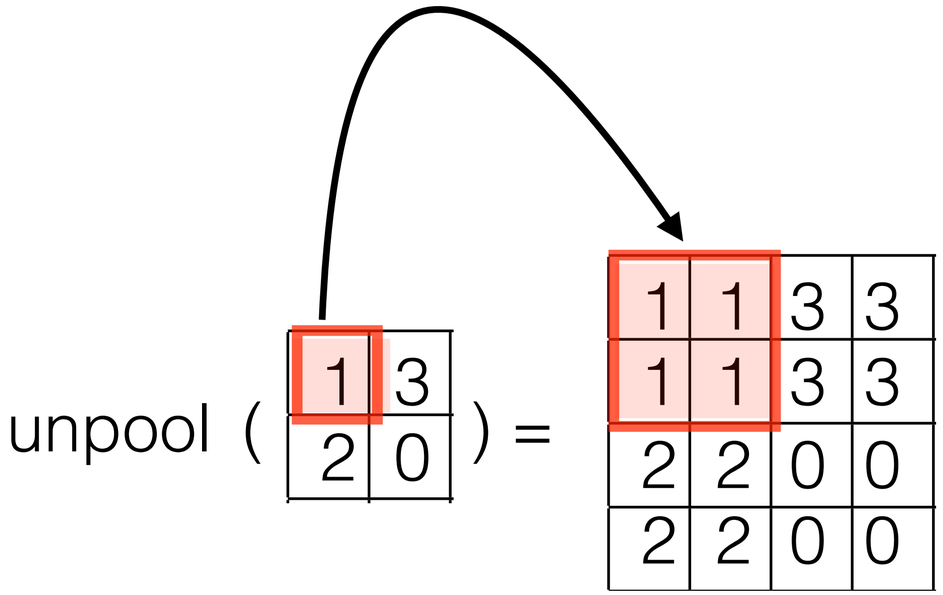


image (2x2)

output (**4x4**)

copy everywhere unpooling

max-unpooling

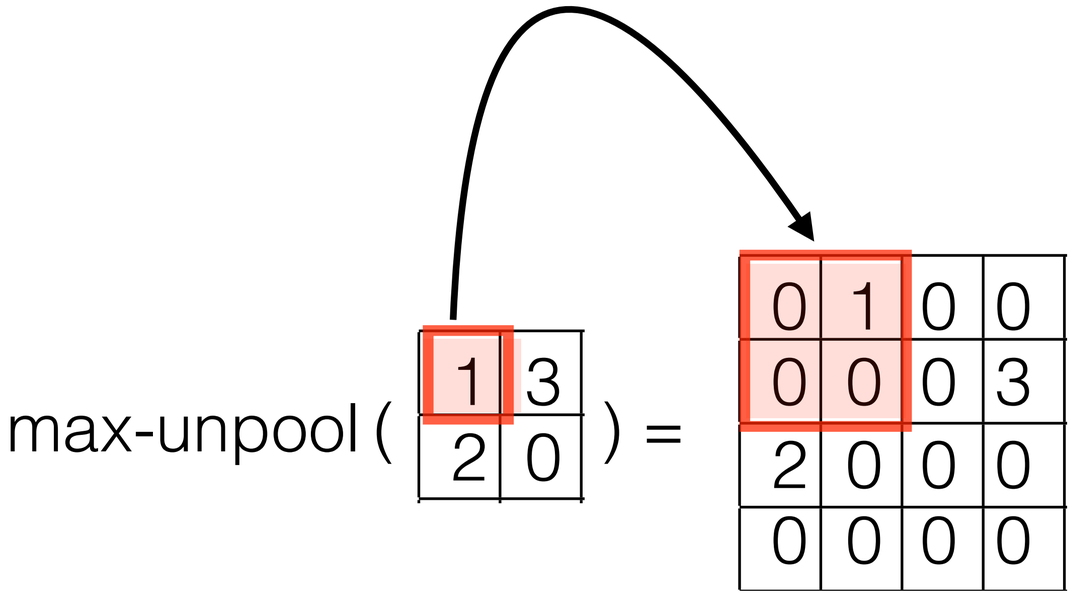


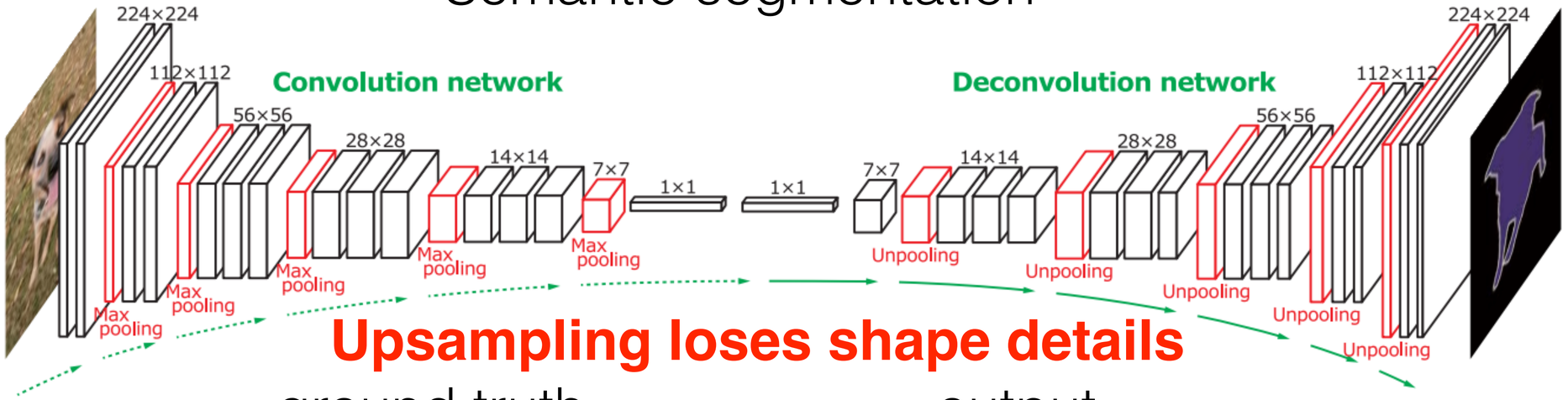
image (2x2)

output (**4x4**)

bed-of-nails unpooling

remember position of the maximum from max-pooling layer

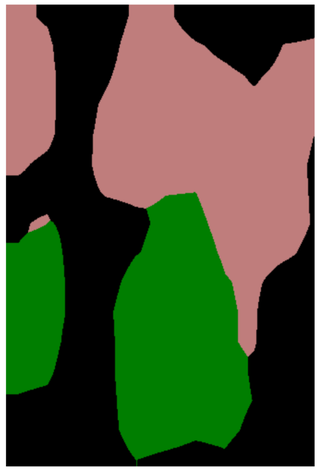
Semantic segmentation



Upsampling loses shape details

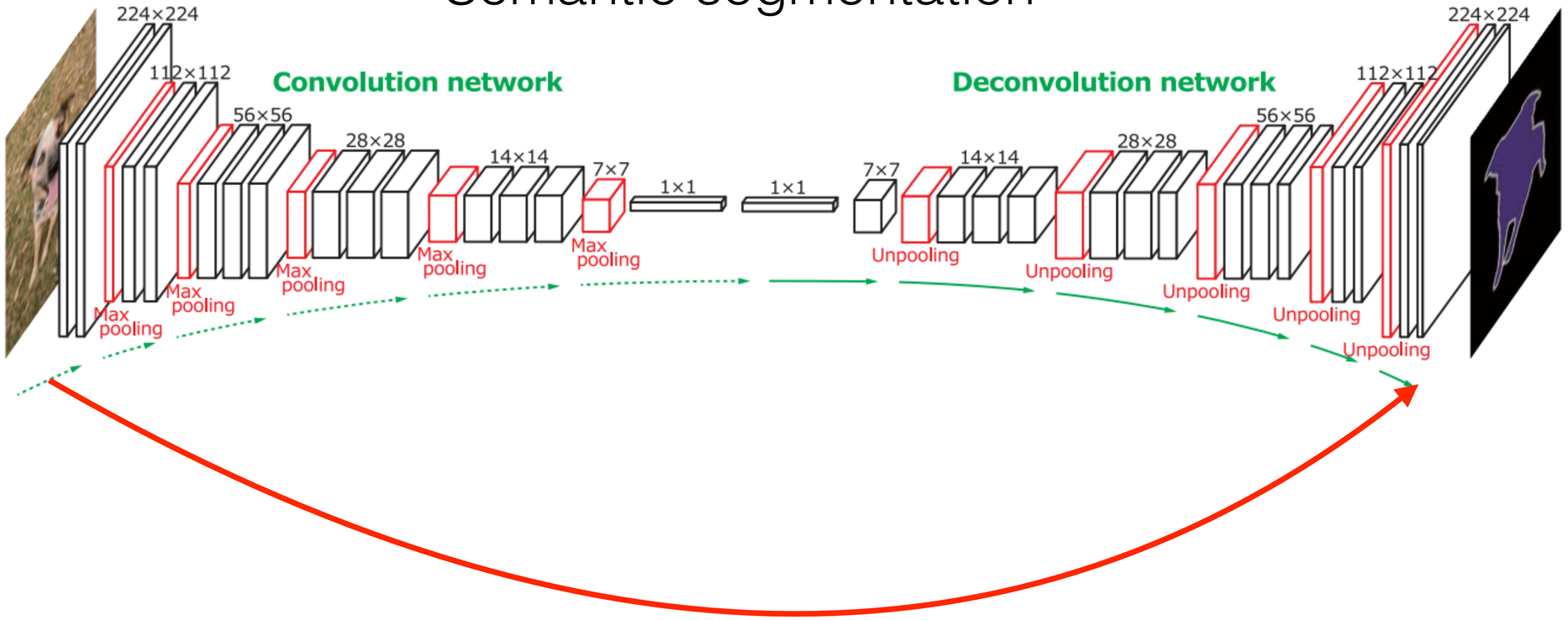
ground truth

output



[Long et al CVPR 2015] https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

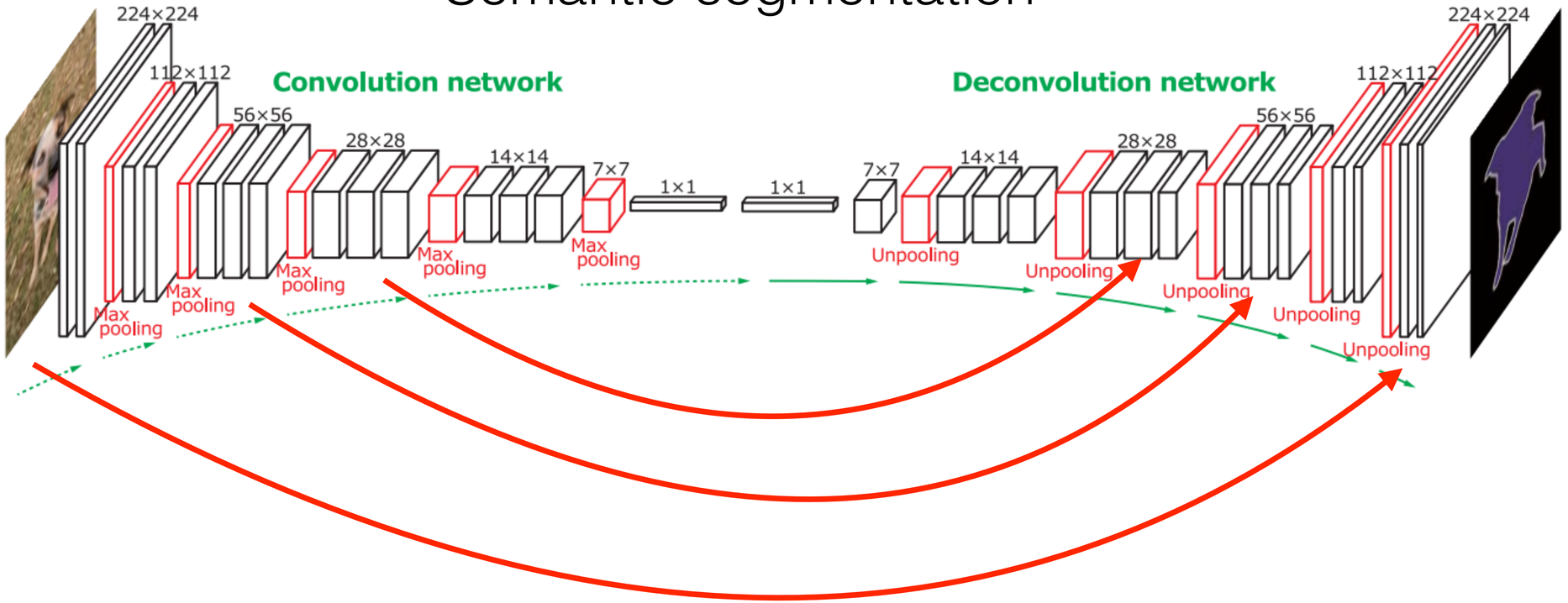
Semantic segmentation



concatenate deconvolution feature map with the original feature map

[Long et al CVPR 2015] https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

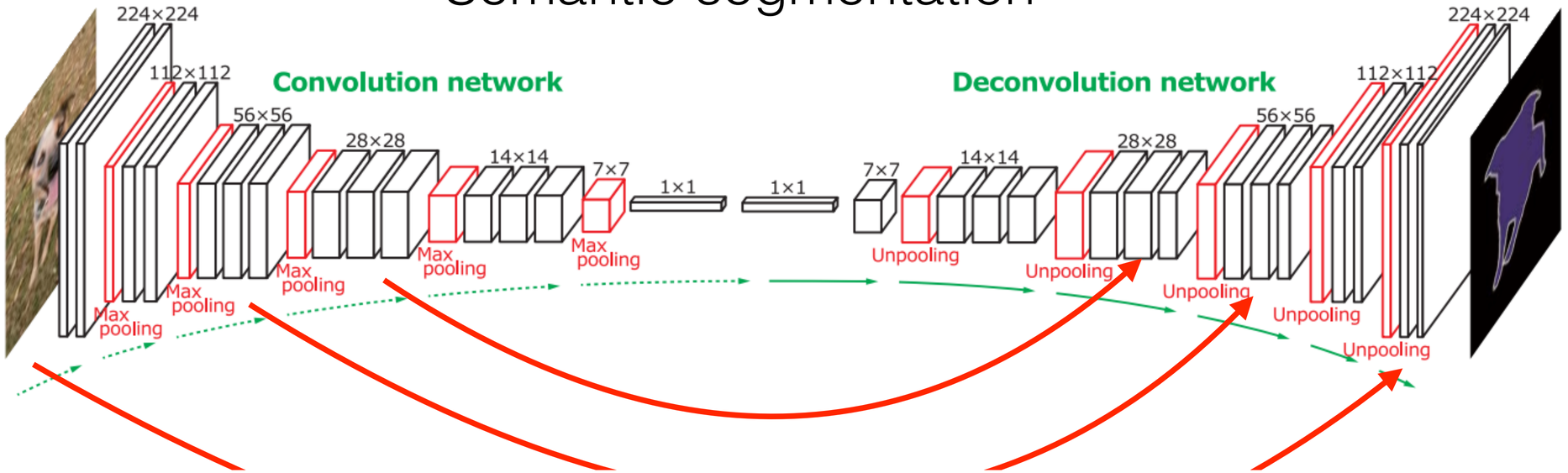
Semantic segmentation



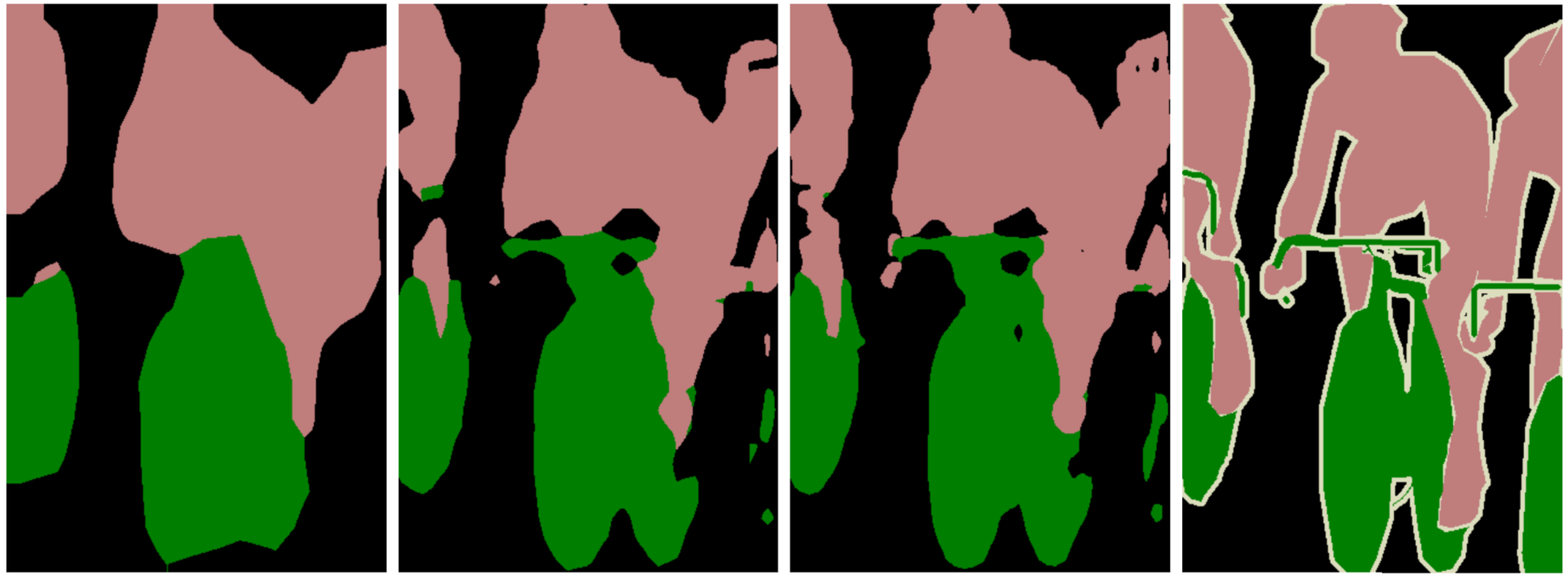
concatenate deconvolution feature map with the original feature map

[Long et al CVPR 2015] https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

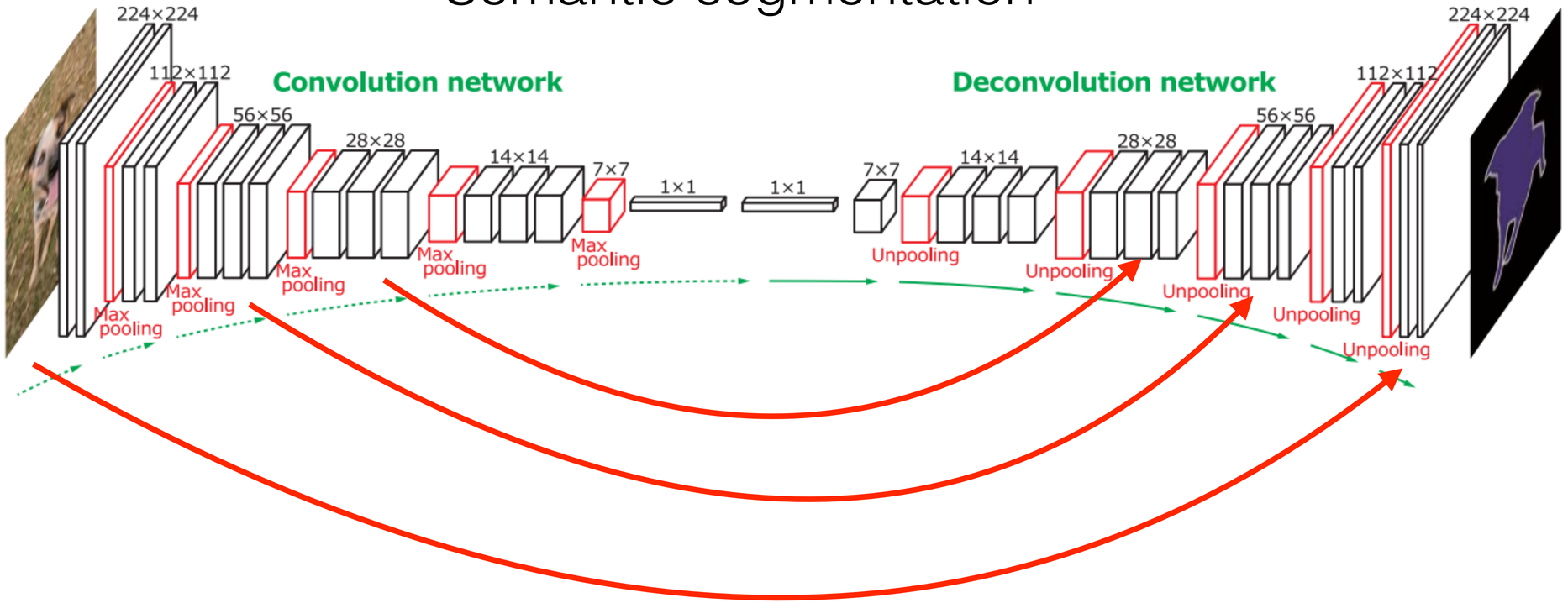
Semantic segmentation



FCN-32s FCN-16s FCN-8s Ground truth

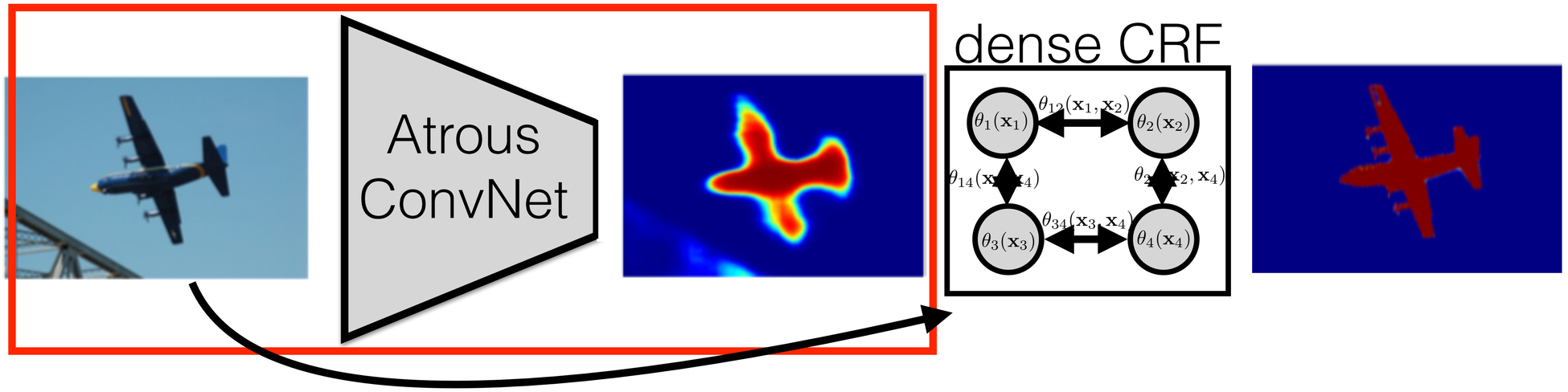


Semantic segmentation



- Autonomous driving applications require segmentation of objects on very different scales.
- Instead of segmenting on hires images, downsampling, detecting on midres images downsampling… upsampling
- People introduced atrous convolution

DeepLab v3



- Replace “maxpooling+conv” by “atrous convolution”
- Replace deconvolutions by bi-linear interp+CRF

[Chen et al. TPAMI 2018] https://arxiv.org/pdf/1606.00915.pdf

Convolution layer

Dilatation rate = 1

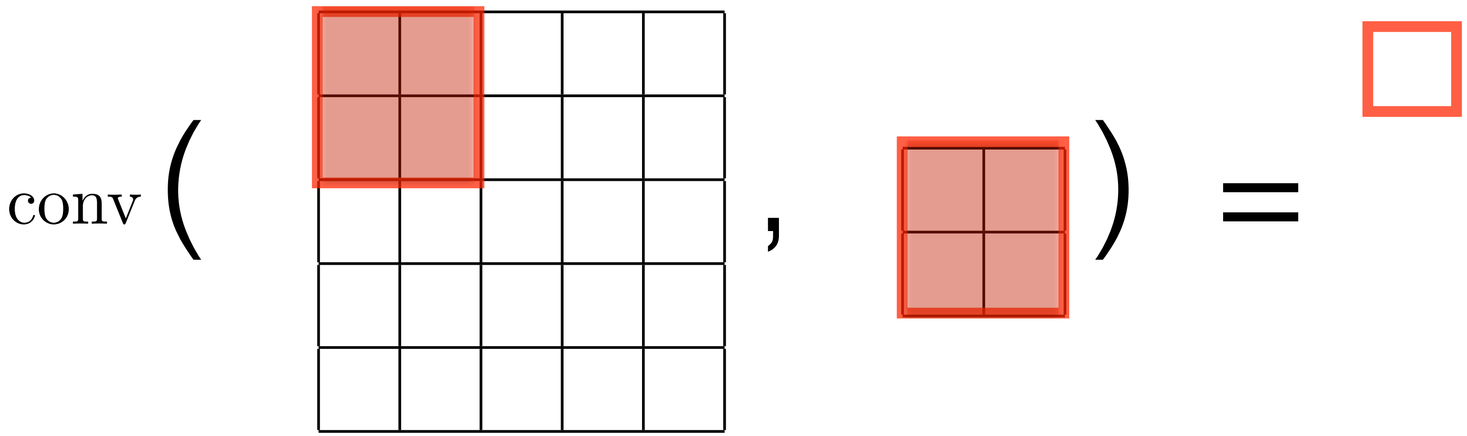


image
(5x5)

kernel
(2x2)

output
(**? x ?**)

Atrous convolution layer

Dilatation rate = 2

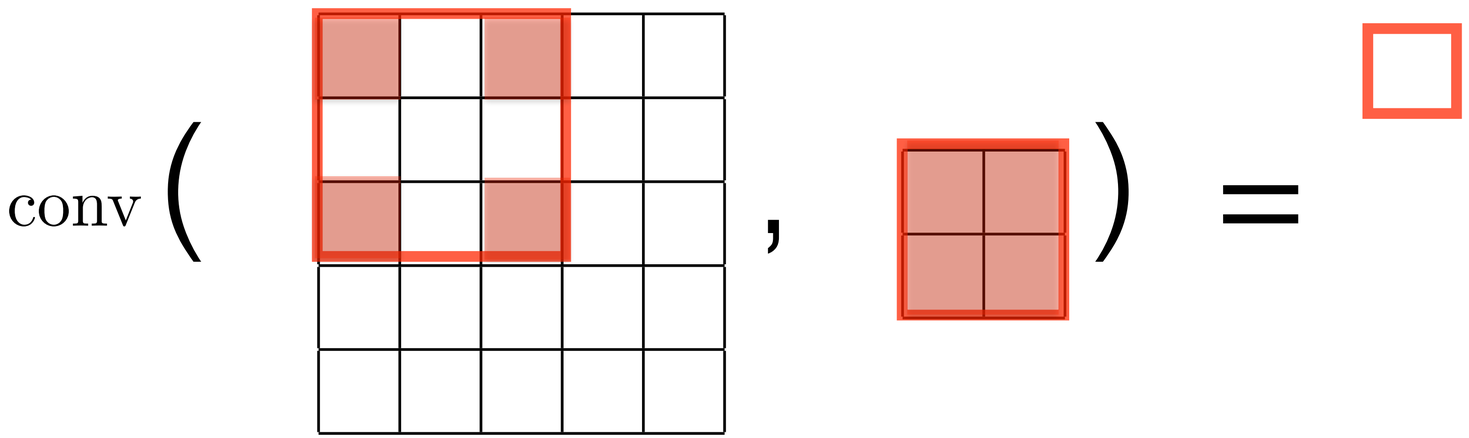
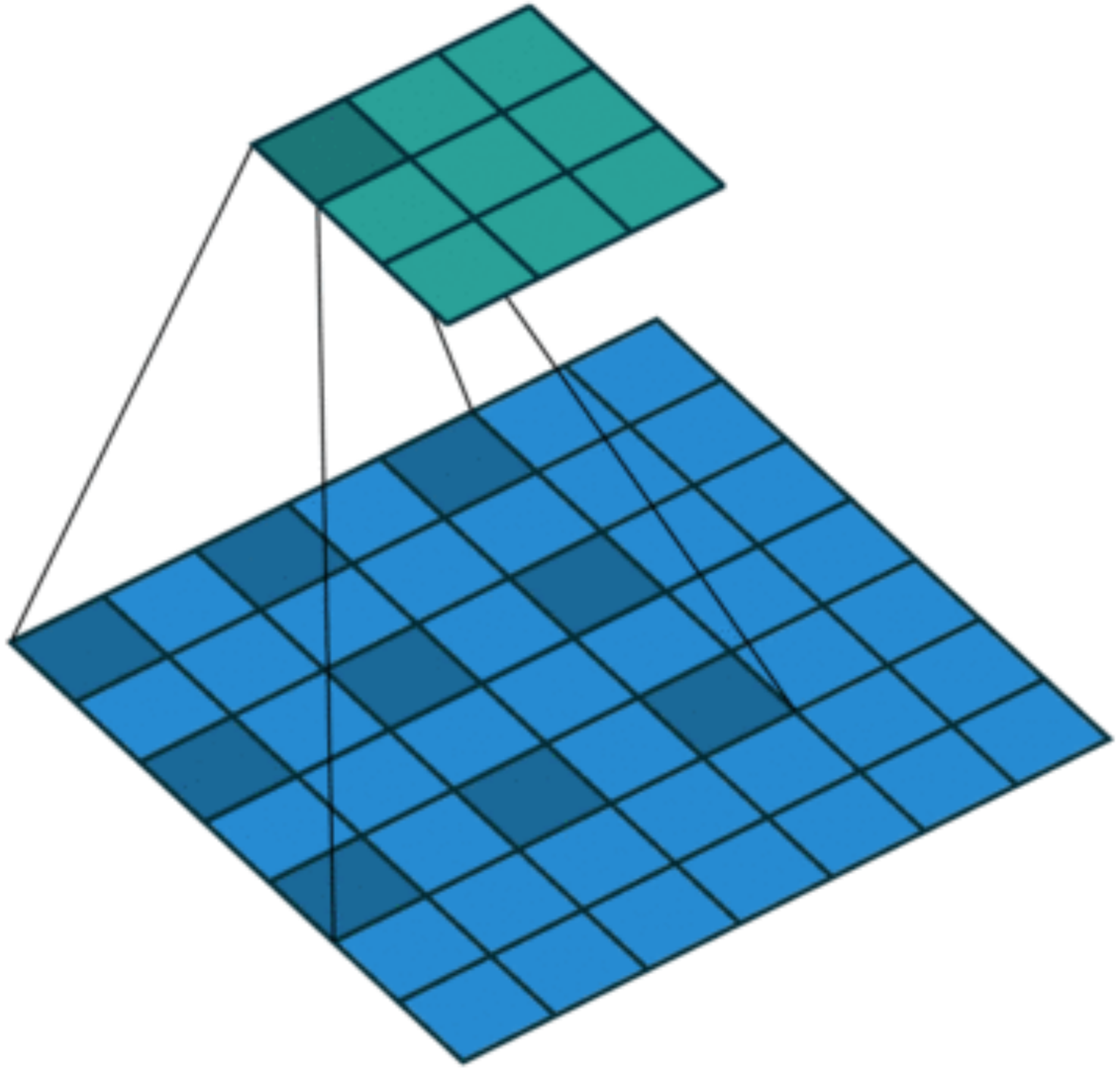


image
(5x5)

kernel
(2x2)

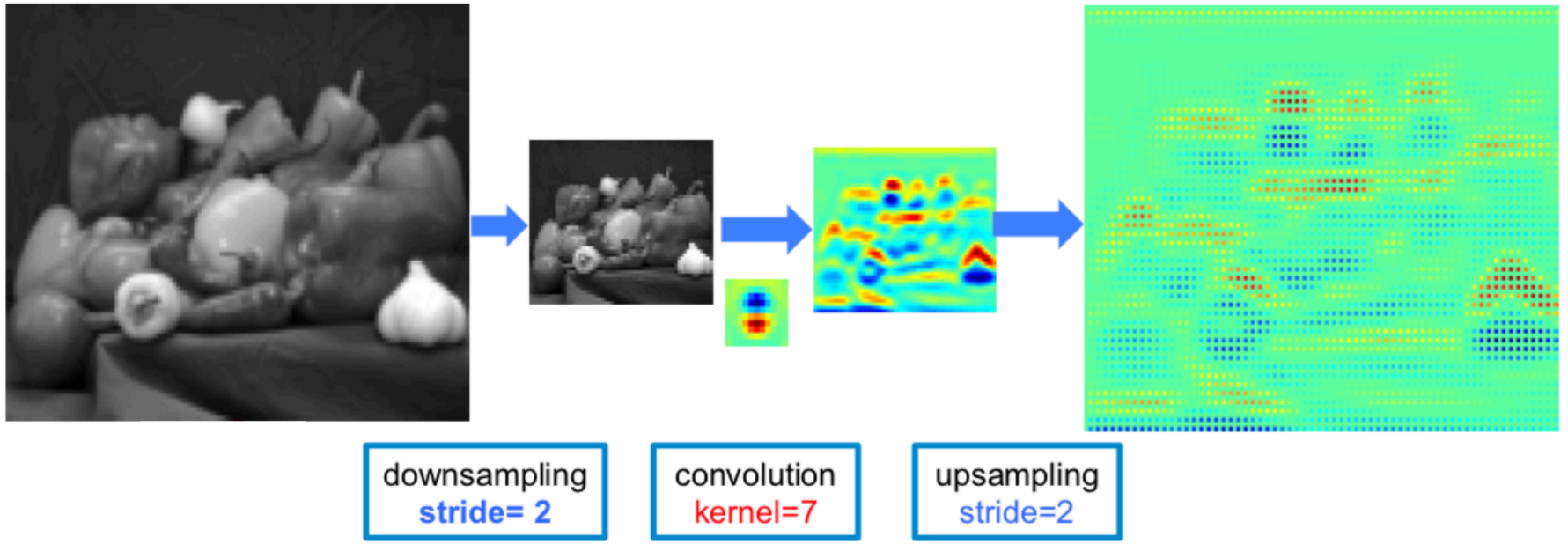
output
(**? x ?**)

Atrous convolution layer



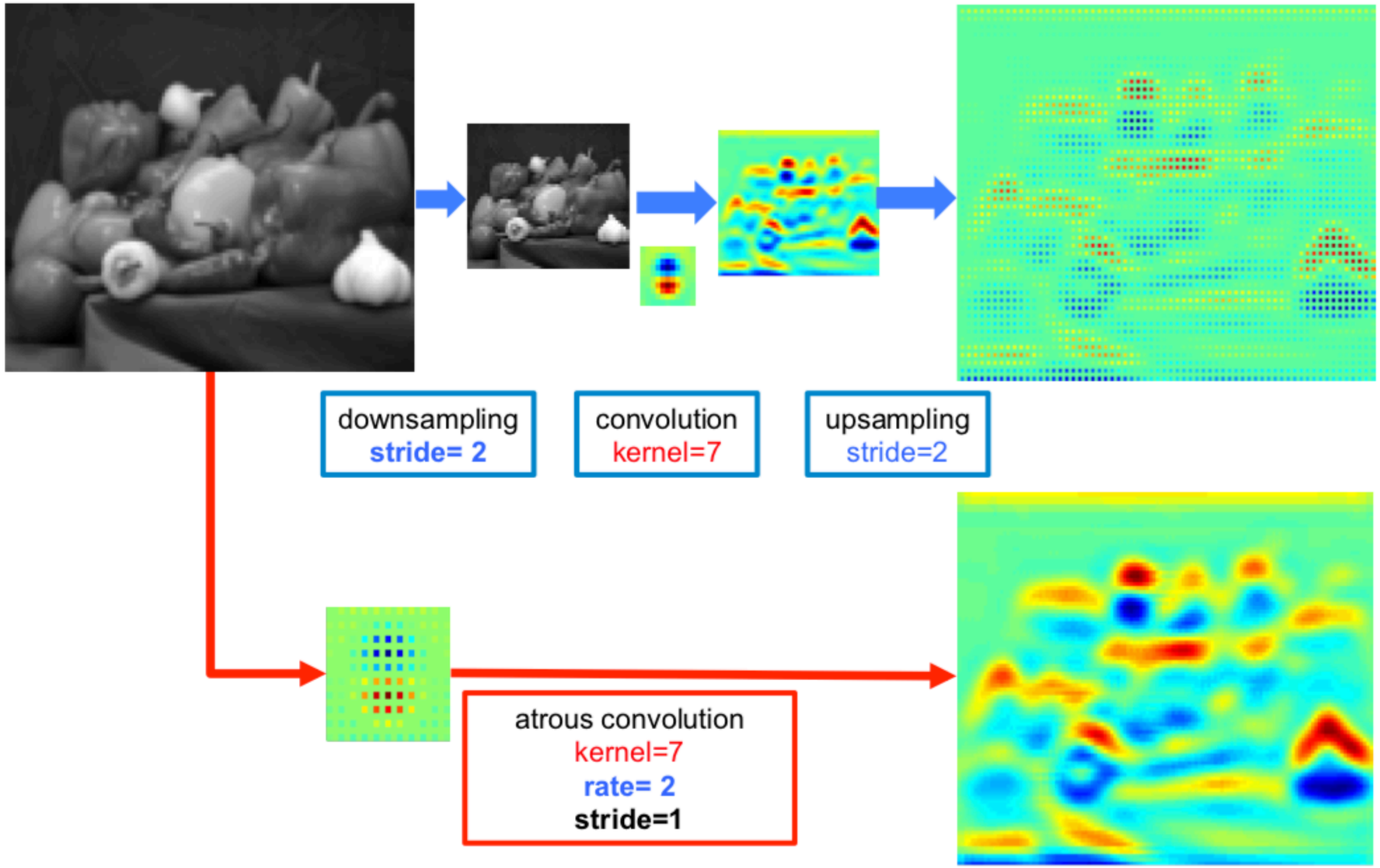
https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md

Atrous vs standard convolution for segmentation



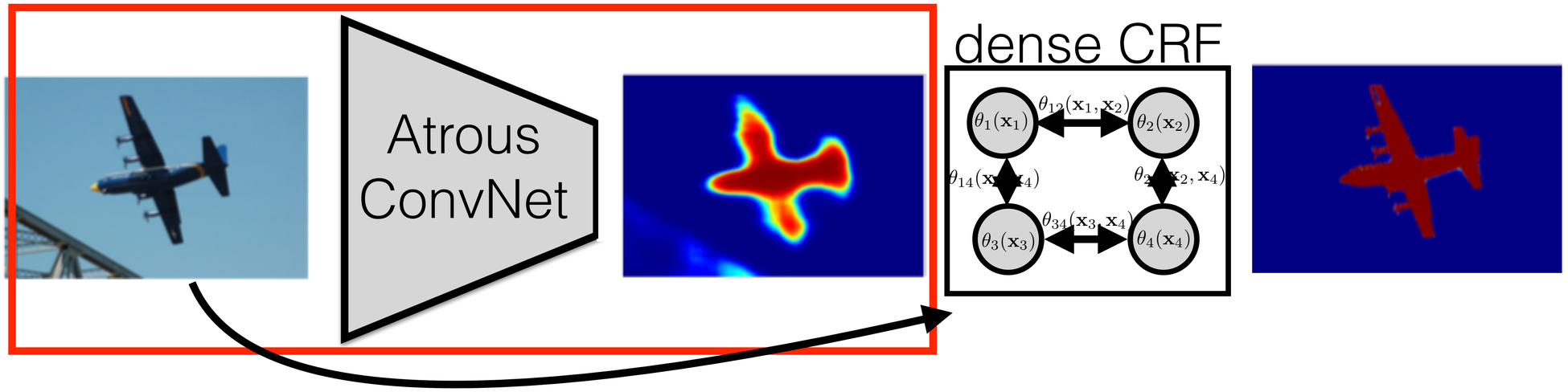
[Chen et al. TPAMI 2018] https://arxiv.org/pdf/1606.00915.pdf

Atrous vs standard convolution for segmentation



[Chen et al. TPAMI 2018] https://arxiv.org/pdf/1606.00915.pdf

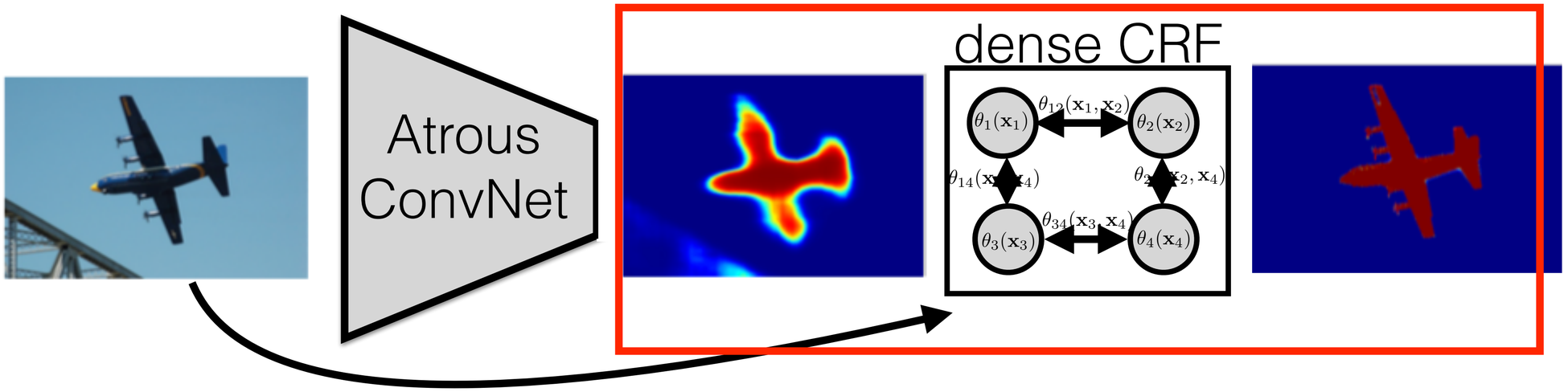
DeepLab v3



- Replace conv+deconv by Atrous Convolution

[Chen et al. TPAMI 2018] https://arxiv.org/pdf/1606.00915.pdf

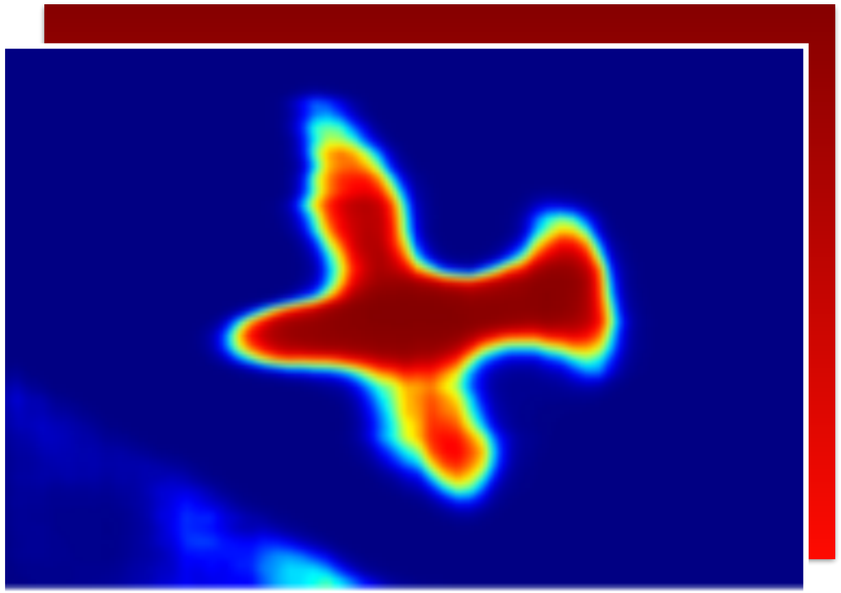
DeepLab v3



- Replace conv+deconv by Atrous Convolution

[Chen et al. TPAMI 2018] https://arxiv.org/pdf/1606.00915.pdf

DeepLab v3 - Conditional Random Fields (CRF)

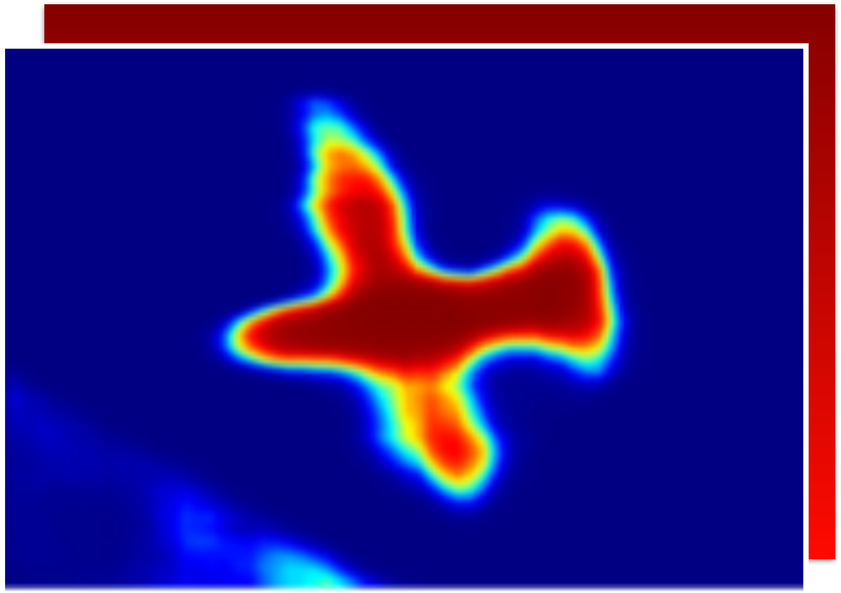


$p(\mathbf{x}_i)$

label of pixel i
$\mathbf{x}_i \in \{0, 1\}$

[Chen et al. TPAMI 2018] https://arxiv.org/pdf/1606.00915.pdf

DeepLab v3 - Conditional Random Fields (CRF)

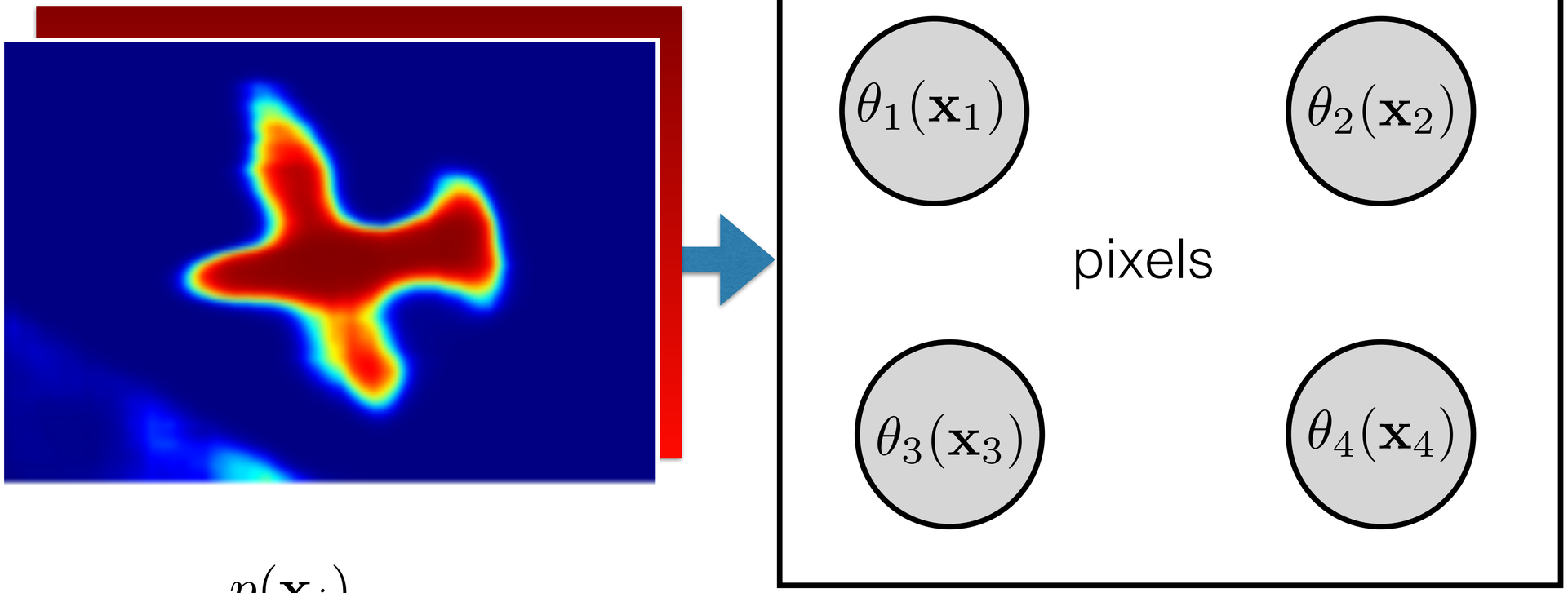


$p(\mathbf{x}_i)$

output of DCNN in pixel i (probability that pixel i has label $\mathbf{x}_i$)

[Chen et al. TPAMI 2018] https://arxiv.org/pdf/1606.00915.pdf

DeepLab v3 - Conditional Random Fields (CRF)

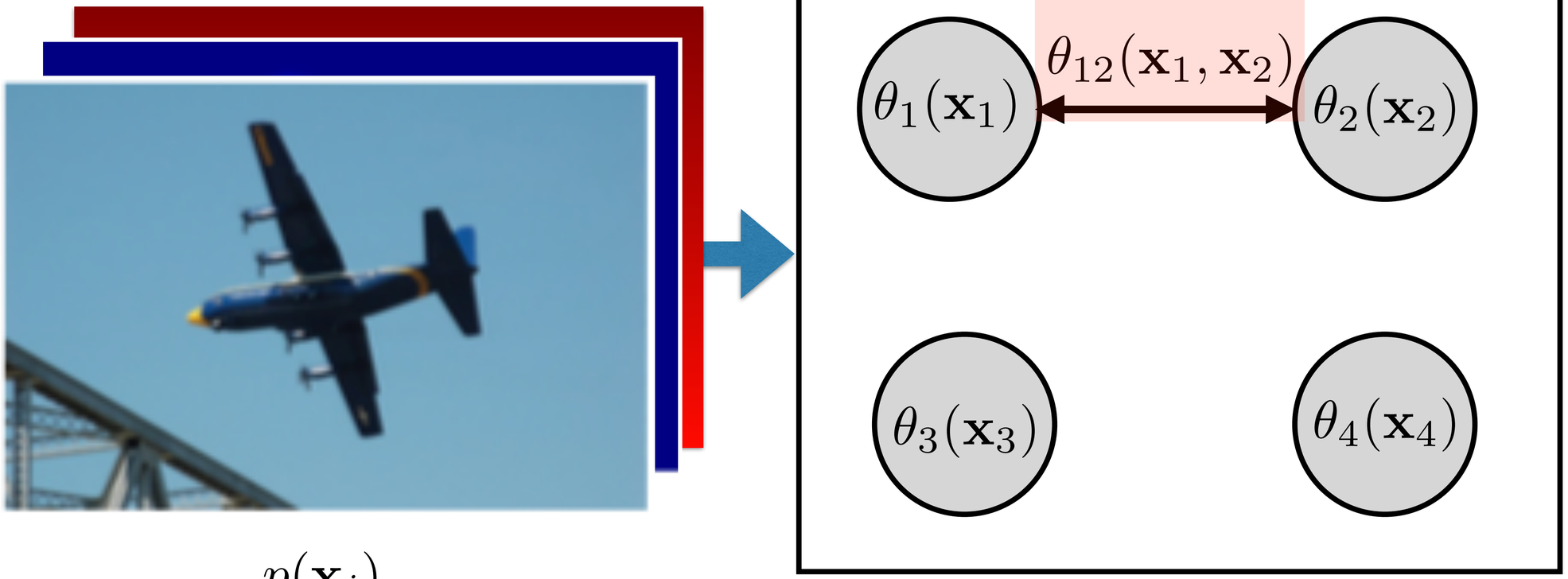


$p(\mathbf{x}_i)$

$\theta_i(\mathbf{x}_i) = -\log(p(\mathbf{x}_i))$ ..... "penalty for not following the estimated probability $p(\mathbf{x}_i)$"

$E(\mathbf{x}) = \sum_i \theta_i(\mathbf{x}_i)$ => $\arg \min_{\mathbf{x} \in \{0,1\}^N} E(\mathbf{x})$ => greedy solution

DeepLab v3 - Conditional Random Fields (CRF)

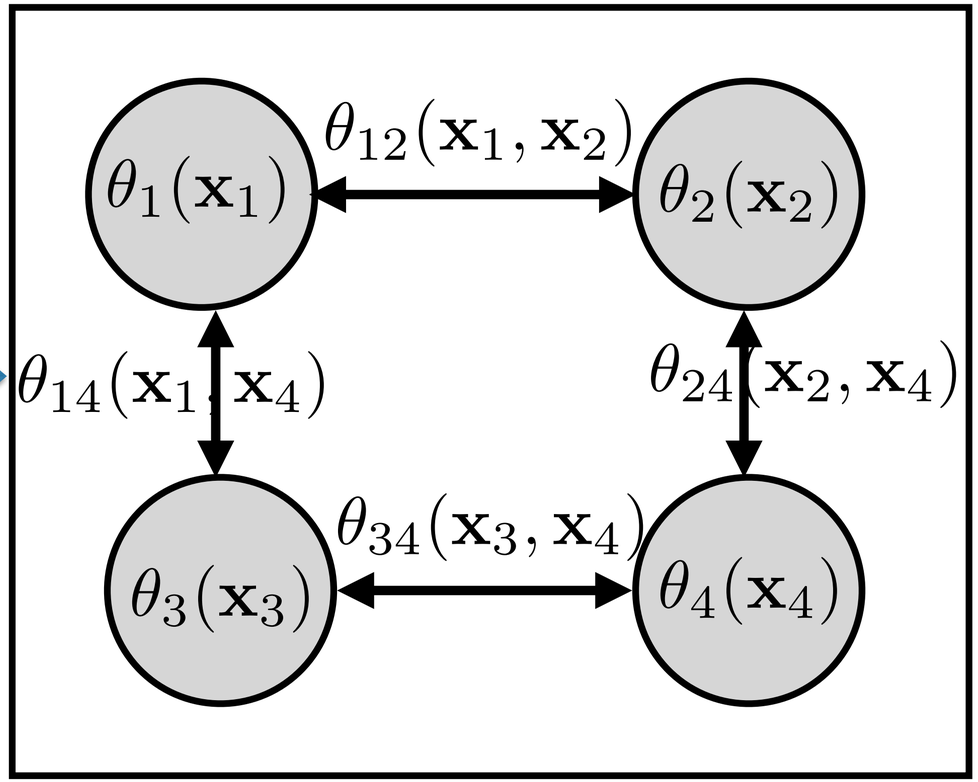


$p(\mathbf{x}_i)$

$\theta_i(\mathbf{x}_i) = -\log(p(\mathbf{x}_i))$ ..... "penalty for not following the estimated probability $p(\mathbf{x}_i)$"

$\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ ... "penalty for dissimilar labels on similar pixels"

DeepLab v3 - Conditional Random Fields (CRF)

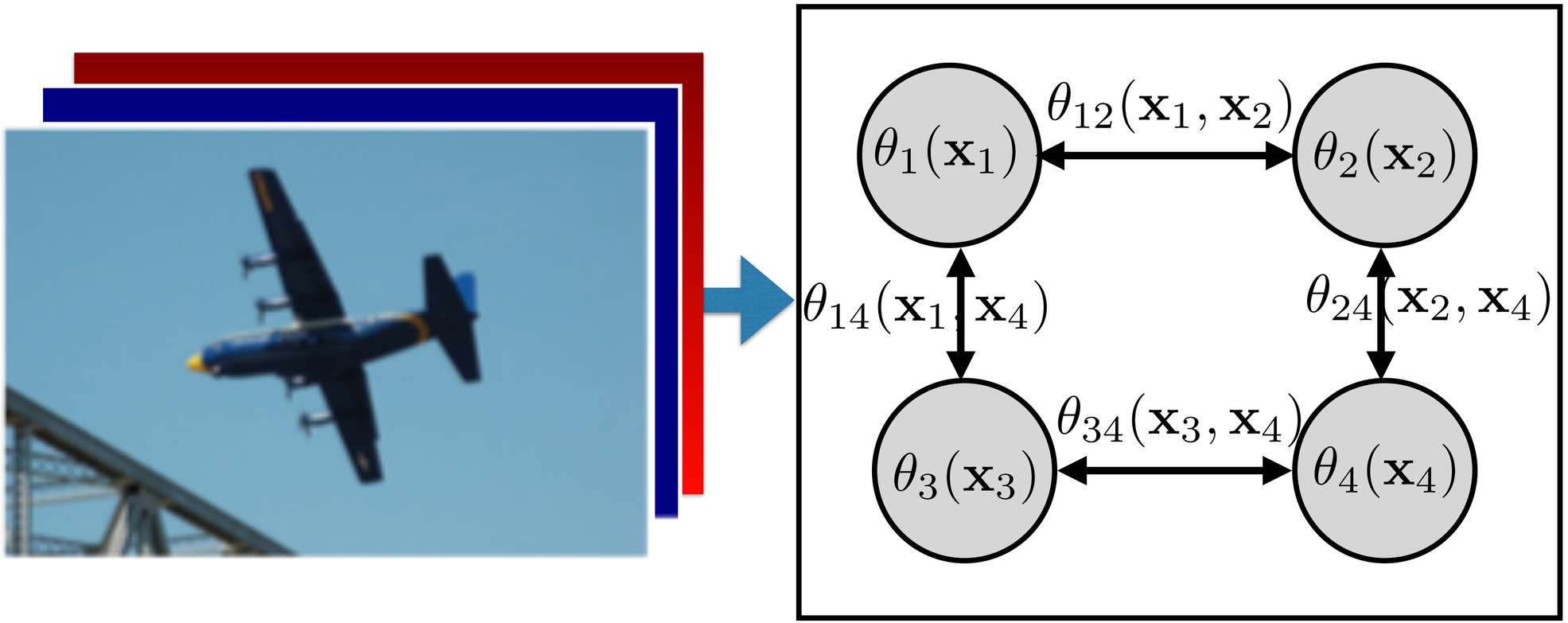


$p(\mathbf{x}_i)$

$\theta_i(\mathbf{x}_i) = -\log(p(\mathbf{x}_i))$ ..... "penalty for not following the estimated probability $p(\mathbf{x}_i)$"

$\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ ... "penalty for dissimilar labels on similar pixels"

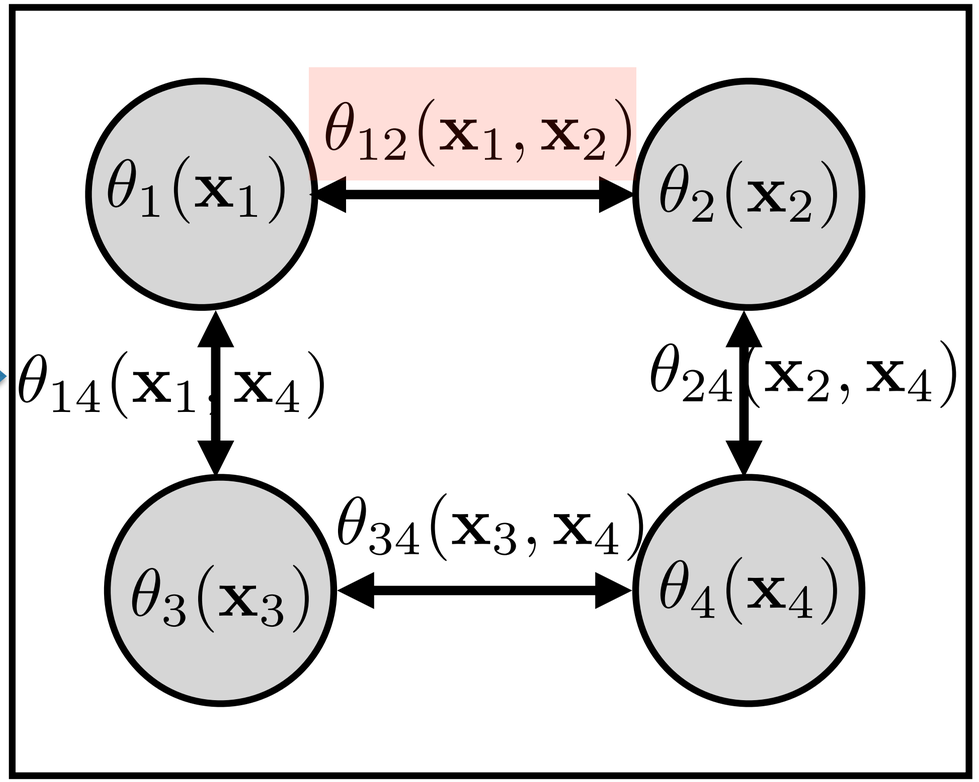
DeepLab v3 - Conditional Random Fields (CRF)



$\theta_i(\mathbf{x}_i) = -\log(p(\mathbf{x}_i))$ ..... "penalty for not following the estimated probability $p(\mathbf{x}_i)$"

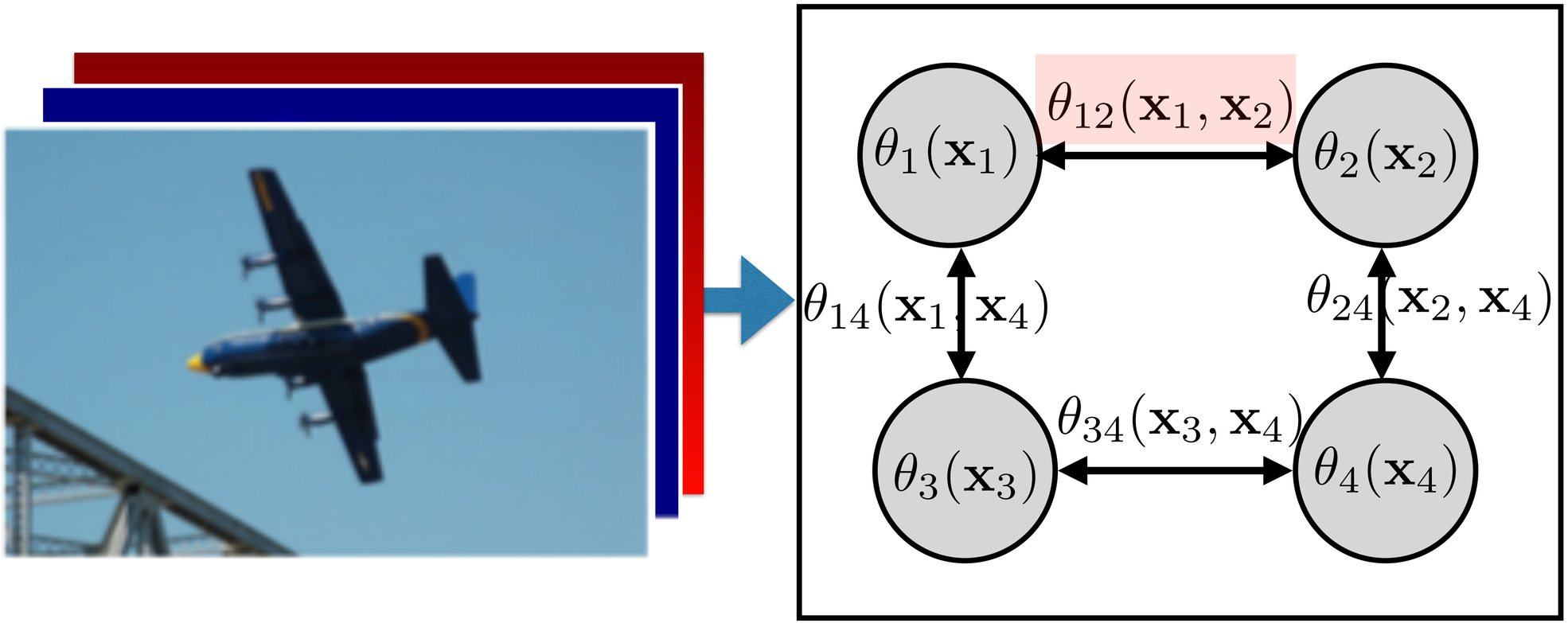
$\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ ... "penalty for dissimilar labels on similar pixels"

DeepLab v3 - Conditional Random Fields (CRF)



$$\theta_{ij}(x_i, x_j) =$$

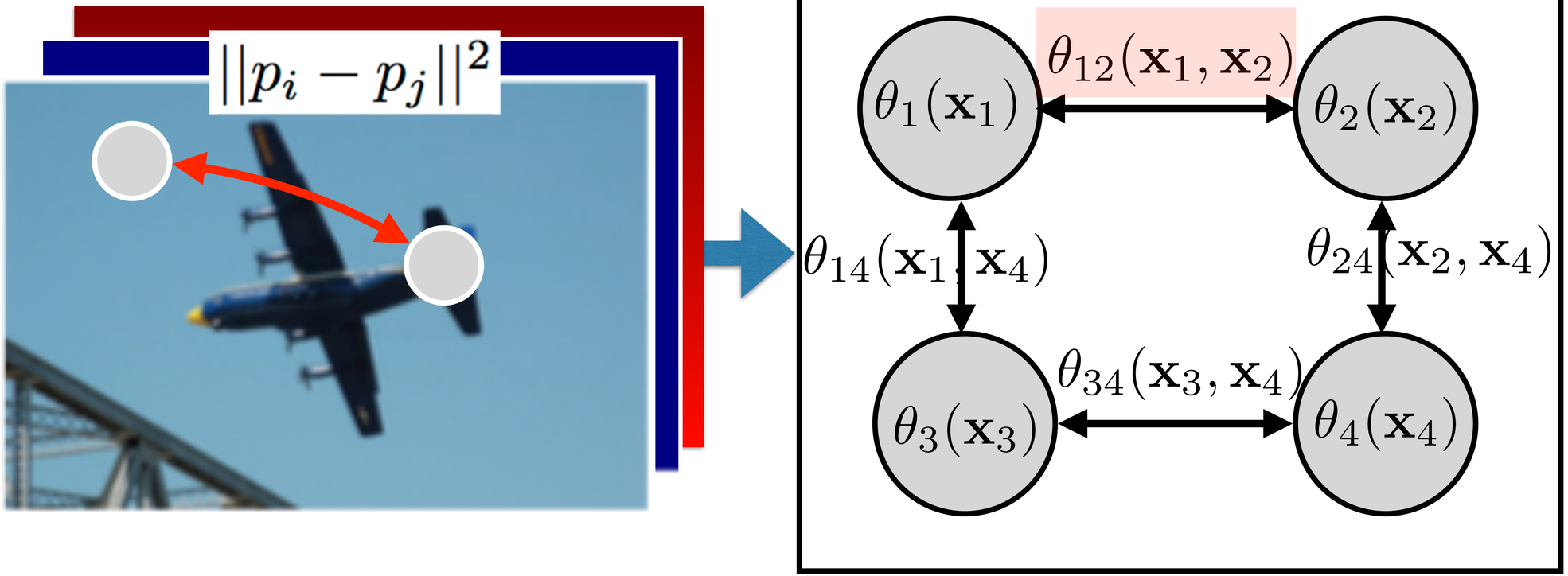
DeepLab v3 - Conditional Random Fields (CRF)



$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j) \begin{cases} 1 & x_i \neq x_j \\ 0 & x_i = x_j \end{cases}$$

same labels are not penalized

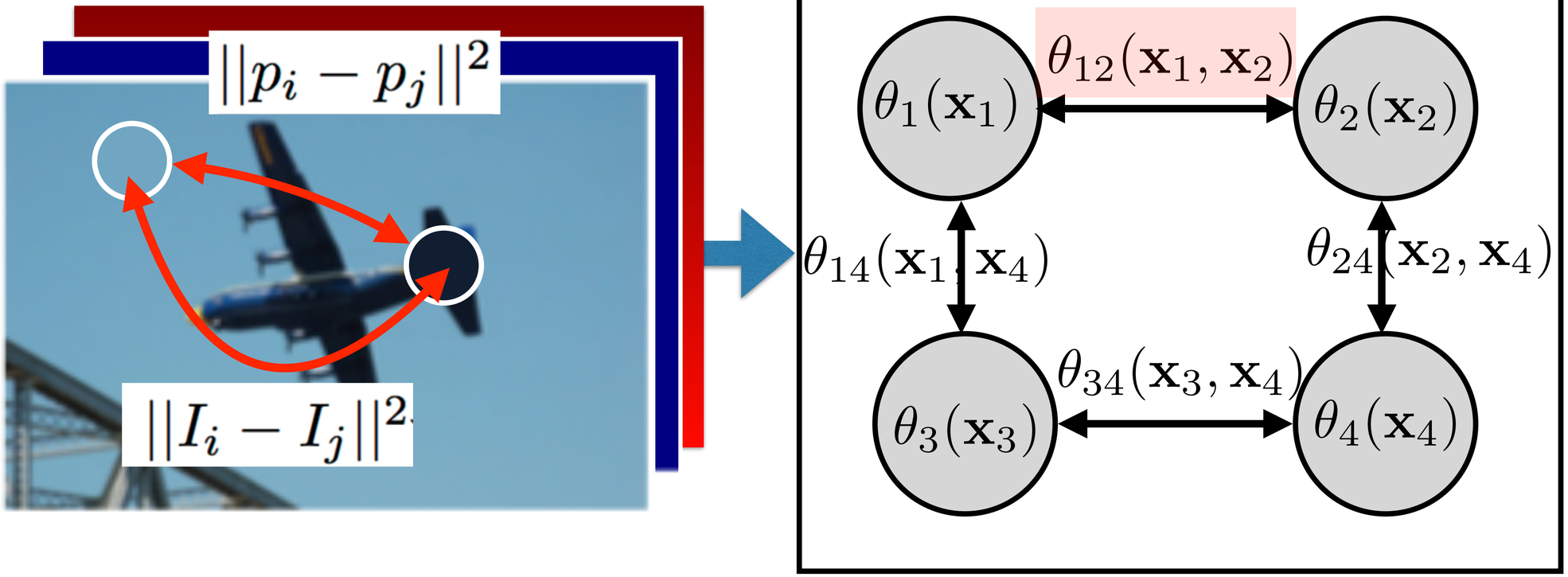
DeepLab v3 - Conditional Random Fields (CRF)



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

high penalty for different labels, when pixels are
(i) spatially close and (ii) has similar color

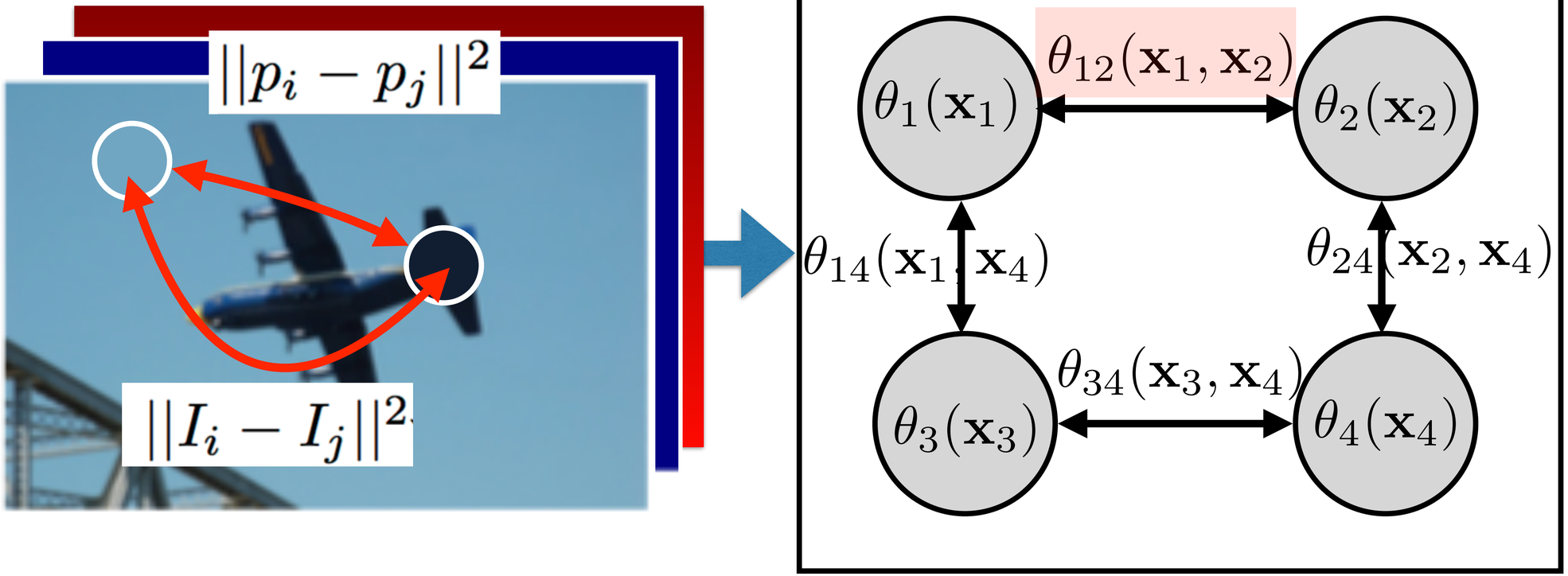
DeepLab v3 - Conditional Random Fields (CRF)



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

high penalty for different labels, when pixels are
(i) spatially close and (ii) has similar color

DeepLab v3 - Conditional Random Fields (CRF)

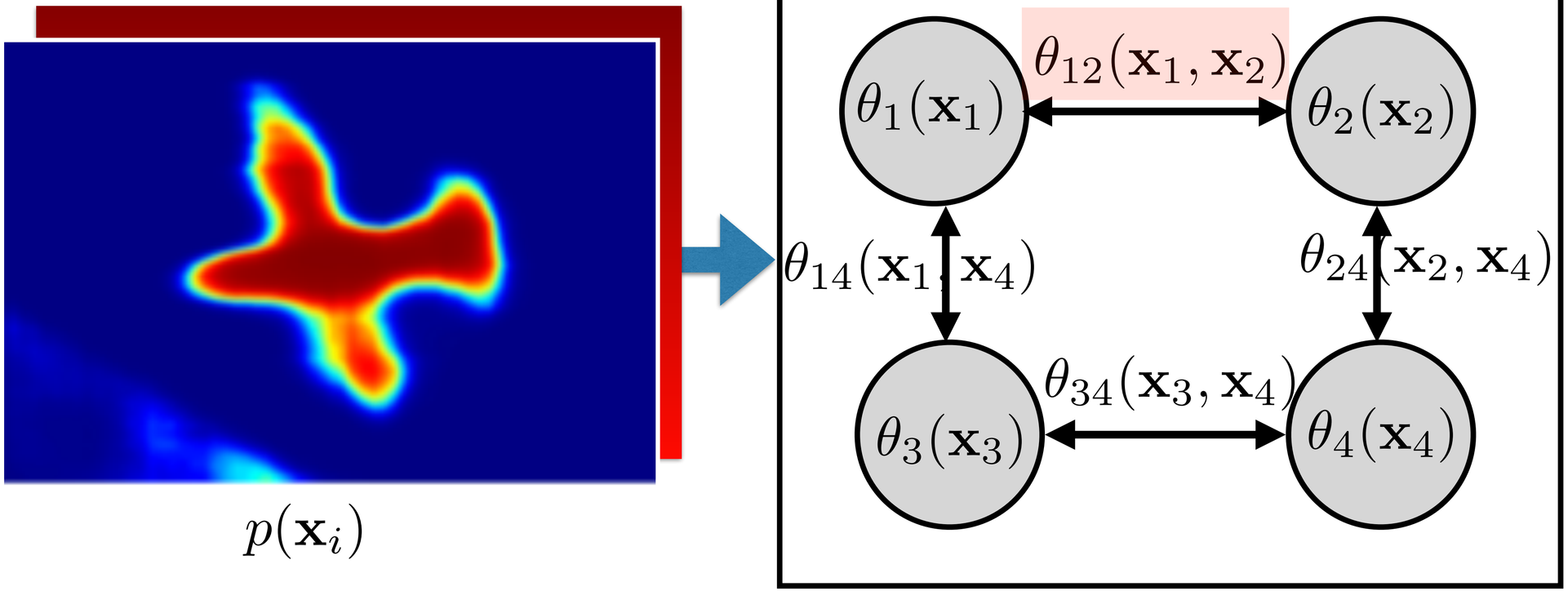


$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j) \left[w_1 \exp\left(-\frac{||p_i - p_j||^2}{2\sigma_\alpha^2} - \frac{||I_i - I_j||^2}{2\sigma_\beta^2} \right) \right.$$

another penalty for close pixels with dissimilar labels

$$\left. + w_2 \exp\left(-\frac{||p_i - p_j||^2}{2\sigma_\gamma^2} \right) \right]$$

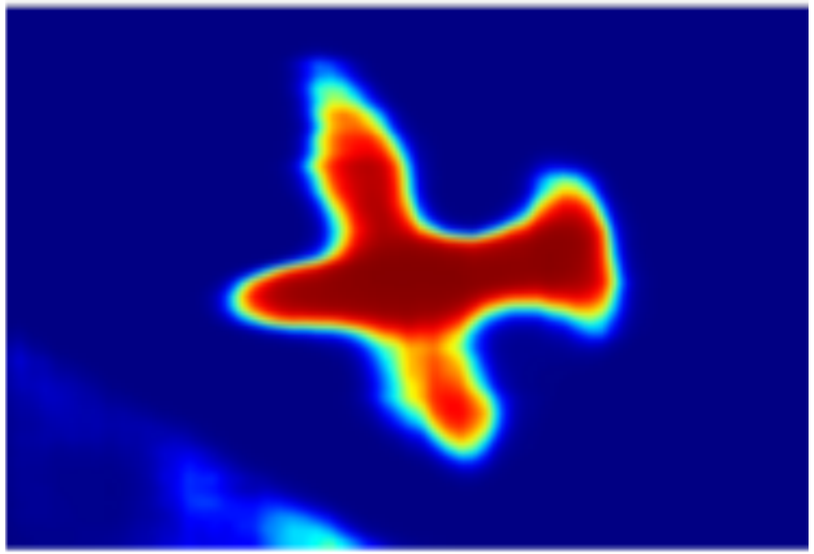
DeepLab v3 - Conditional Random Fields (CRF)



$$E(\mathbf{x}) = \sum_i \theta_i(\mathbf{x}_i) + \sum_{ij} \theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$

$$\arg\min_{\mathbf{x}} E(\mathbf{x})$$

DeepLab v3 - Conditional Random Fields (CRF)



$p(\mathbf{x}_i)$

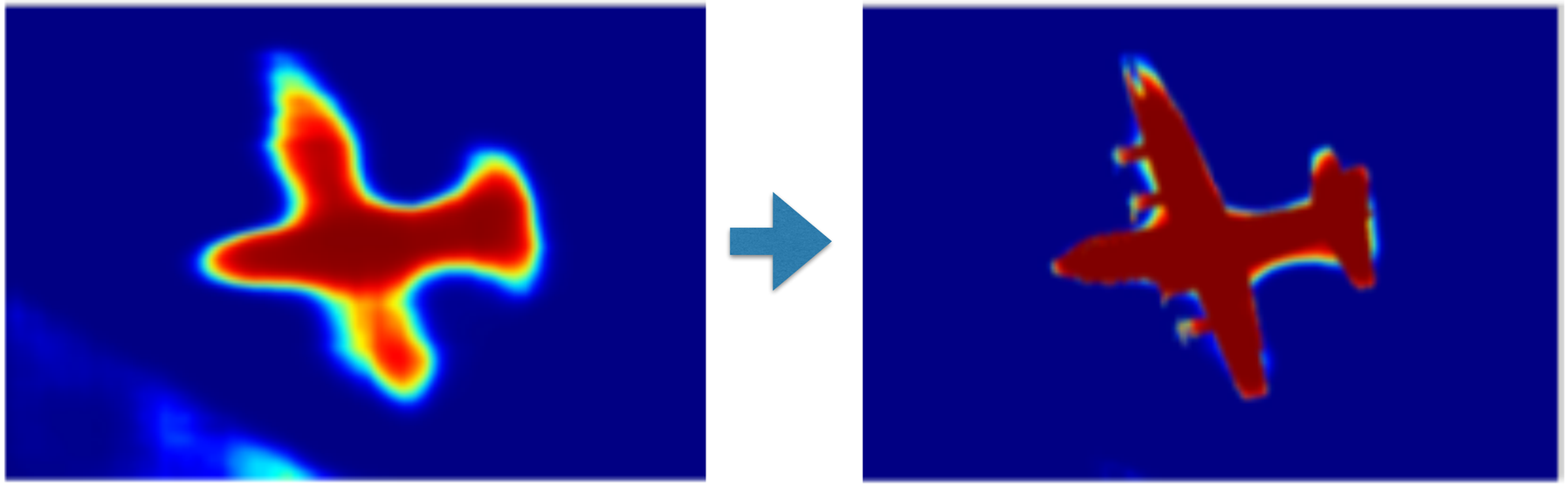
$$\arg\min_{\mathbf{x}} E(\mathbf{x})$$

$$E(\mathbf{x}) = \sum_i \theta_i(\mathbf{x}_i) + \sum_{ij} \theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$

- Direct optimization complicated (NP complete)
- Mean field approximation [Koltun NIPS 2011] https://arxiv.org/pdf/1210.5644.pdf

DeepLab v3 - Conditional Random Fields (CRF)

1. iteration



$p(\mathbf{x}_i)$

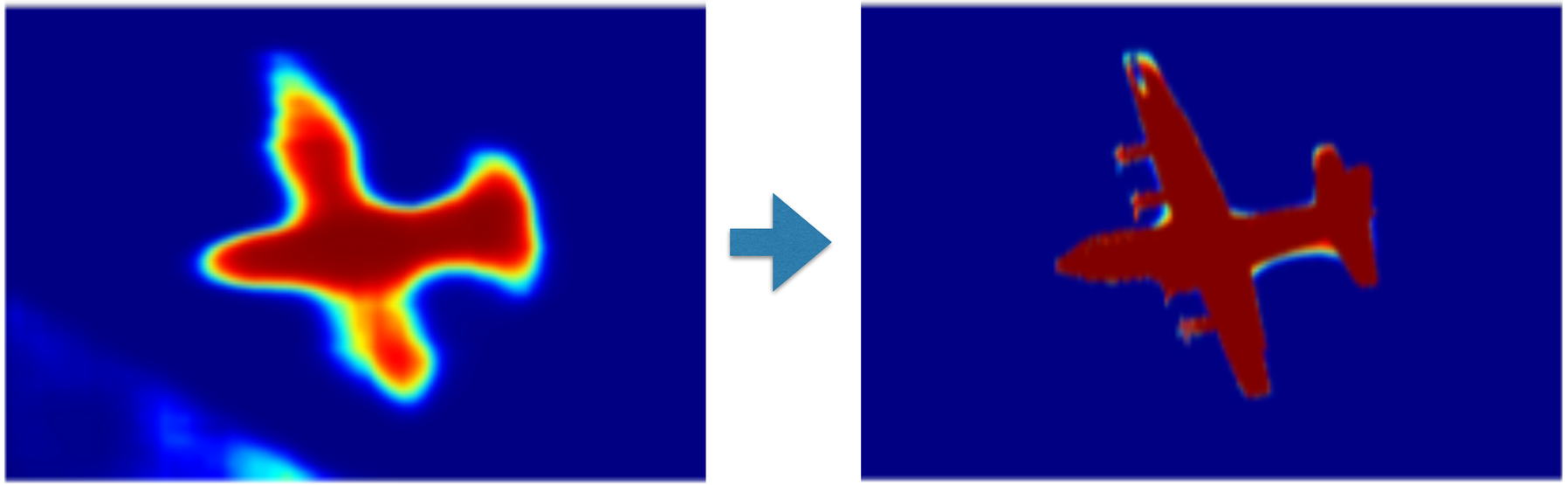
$\arg\min_{\mathbf{x}} E(\mathbf{x})$

$$E(\mathbf{x}) = \sum_i \theta_i(\mathbf{x}_i) + \sum_{ij} \theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$

- Direct optimization complicated (NP complete)
- Mean field approximation [Koltun NIPS 2011] https://arxiv.org/pdf/1210.5644.pdf

DeepLab v3 - Conditional Random Fields (CRF)

2. iteration



$p(\mathbf{x}_i)$

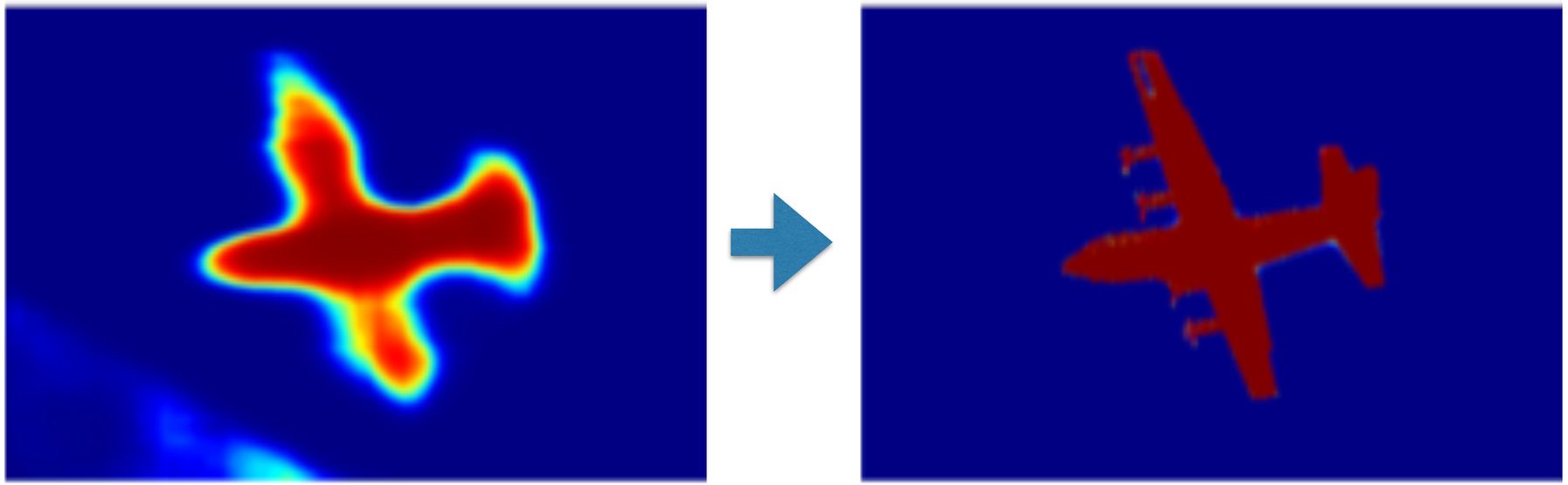
$\arg\min_{\mathbf{x}} E(\mathbf{x})$

$$E(\mathbf{x}) = \sum_i \theta_i(\mathbf{x}_i) + \sum_{ij} \theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$

- Direct optimization complicated (NP complete)
- Mean field approximation [Koltun NIPS 2011] https://arxiv.org/pdf/1210.5644.pdf

DeepLab v3 - Conditional Random Fields (CRF)

10. iteration



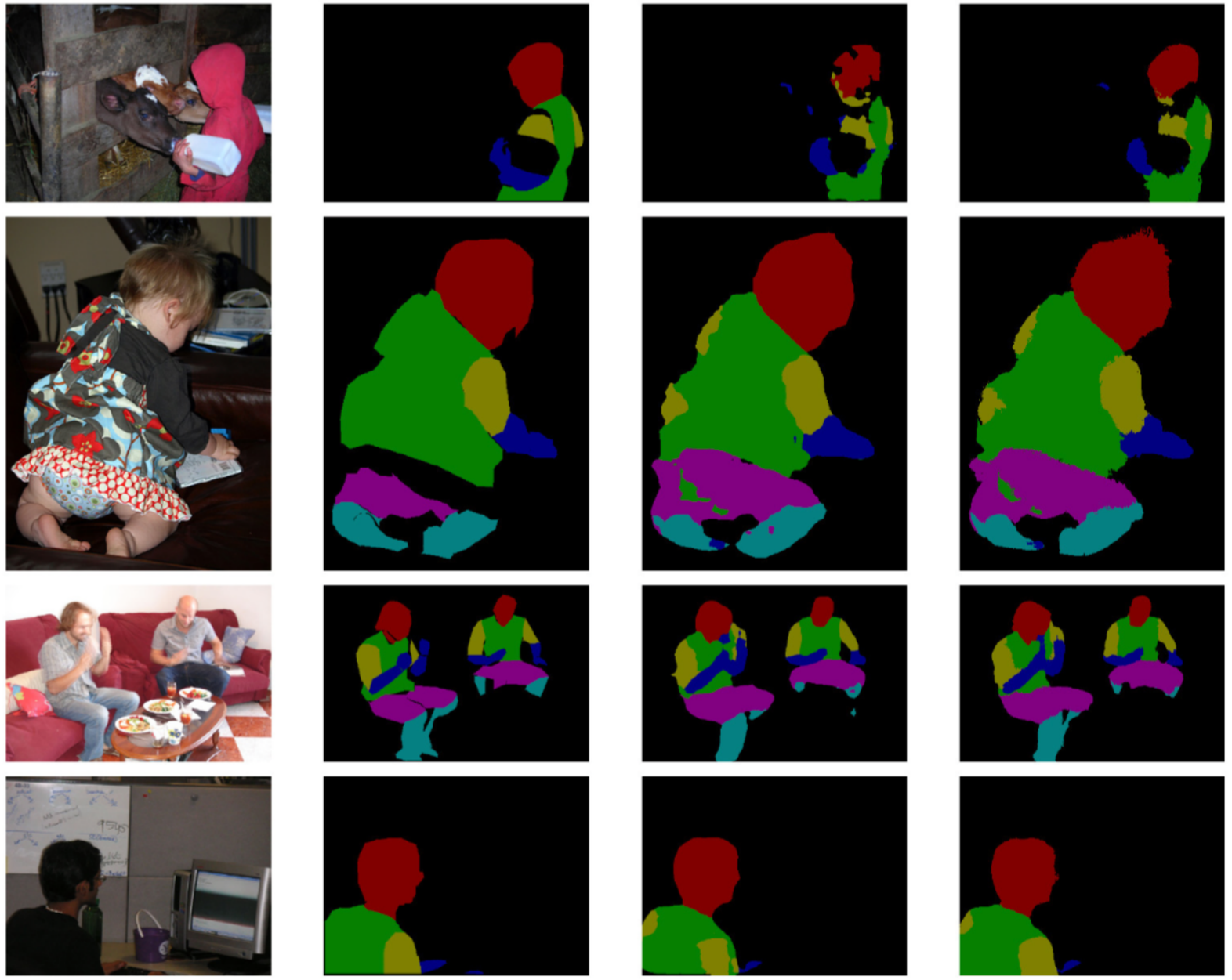
$p(\mathbf{x}_i)$

$\arg\min_{\mathbf{x}} E(\mathbf{x})$

$$E(\mathbf{x}) = \sum_i \theta_i(\mathbf{x}_i) + \sum_{ij} \theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$

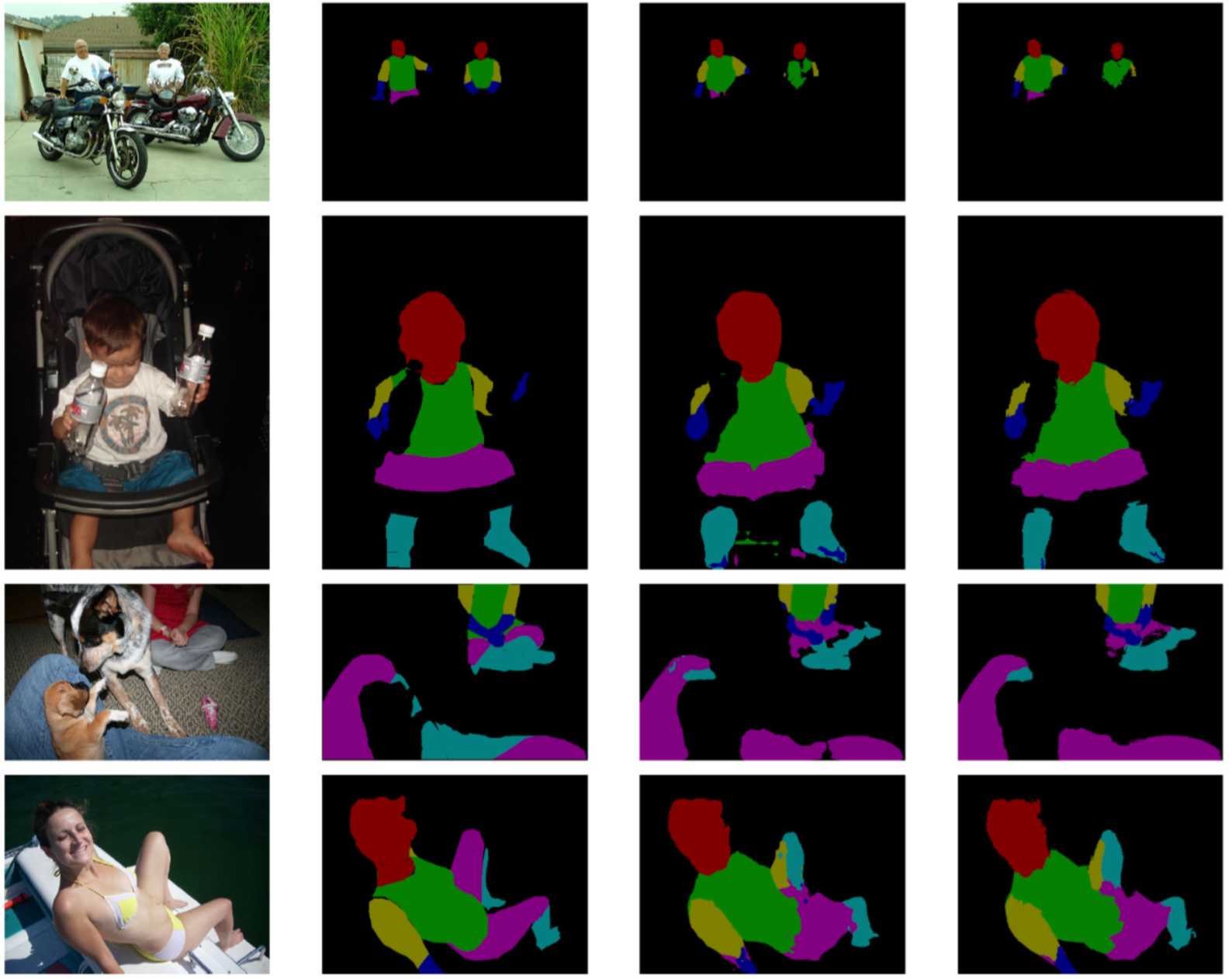
- Direct optimization complicated (NP complete)
- Mean field approximation [Koltun NIPS 2011] https://arxiv.org/pdf/1210.5644.pdf

DeepLab v3 - results



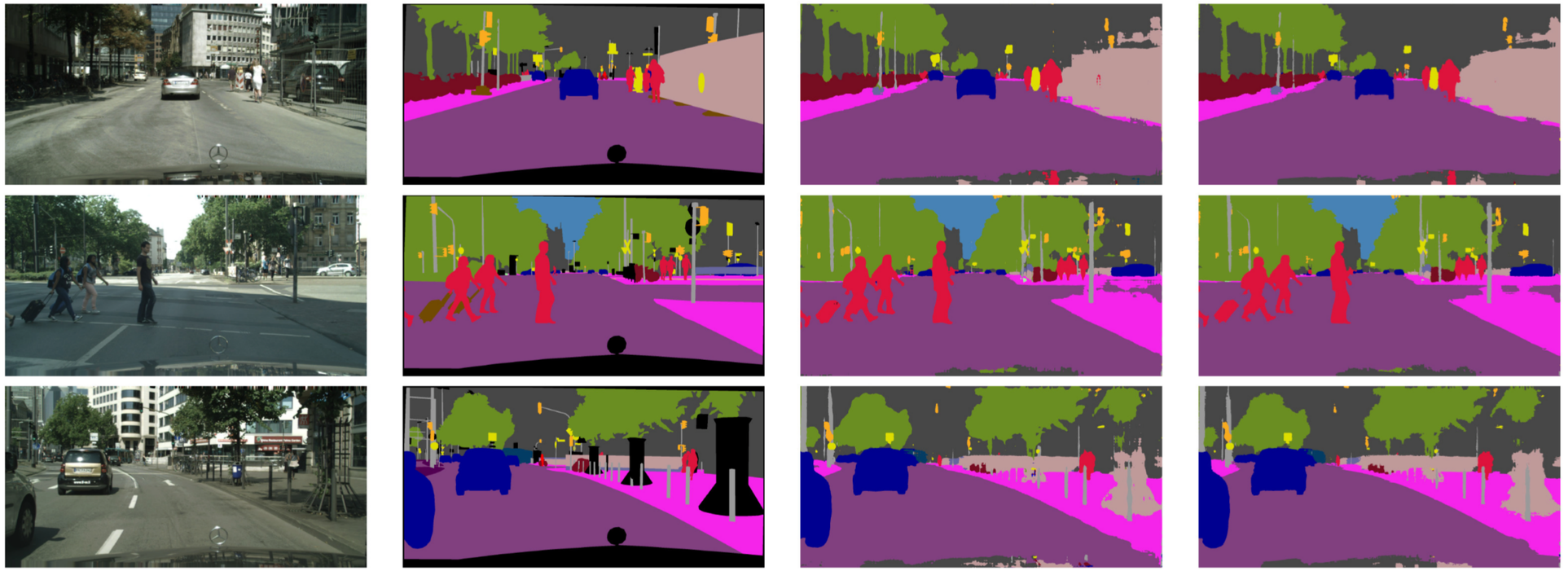
(a) Image (b) G.T. (c) Before CRF (d) After CRF

DeepLab v3 - results



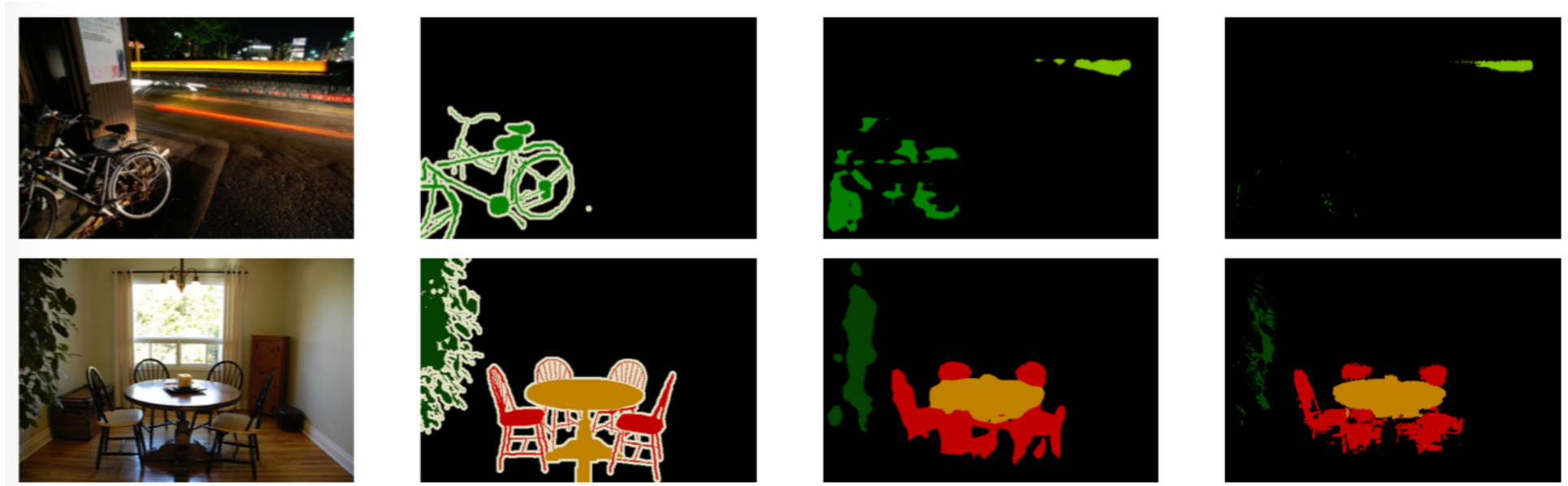
(a) Image (b) G.T. (c) Before CRF (d) After CRF

DeepLab v3 - results



(a) Image (b) G.T. (c) Before CRF (d) After CRF

DeepLab v3 - results



(a) Image (b) G.T. (c) Before CRF (d) After CRF

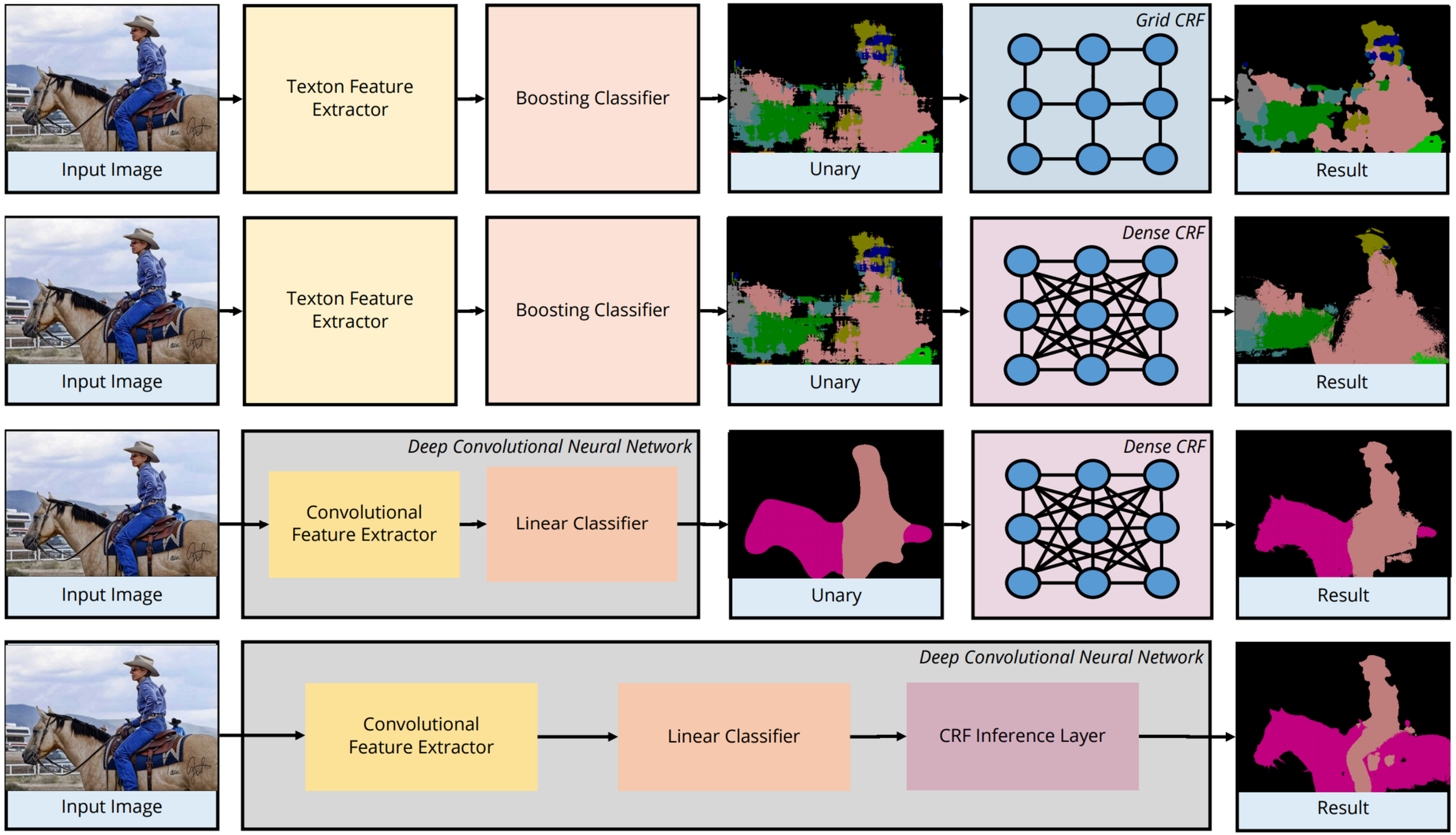
CRF failure cases

DeepLab v3 - summary

- significantly outperforms state-of-the-art on several datasets
- CRF improves mIOU about 2%
- ASPP improves mIOU about 3%
- codes available: https://github.com/tensorflow/models/tree/master/research/deeplab
- state-of-the-art benchmarks: http://www.robustvision.net/leaderboard.php?benchmark=semantic

Segmentation summary, [Torr et al. SPM, 2018]

http://www.robots.ox.ac.uk/~tvg/publications/2017/CRFMeetCNN4SemanticSegmentation.pdf



PyTorch implementation of differentiable ConvCRF layer
[Teichmann & Cipolla, BMVC, 2019]
https://arxiv.org/pdf/1805.04777.pdf

```
pred = gausscrf.forward(unary=unary_var, img=img_var)
```