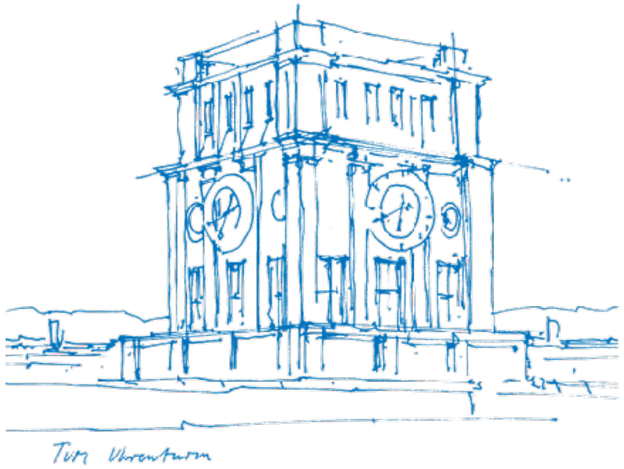
Computer Vision III:

Semantic segmentation

Nikita Araslanov
06.12.2022

Content credit:
Prof. Laura Leal-Taixé
https://dvl.in.tum.de



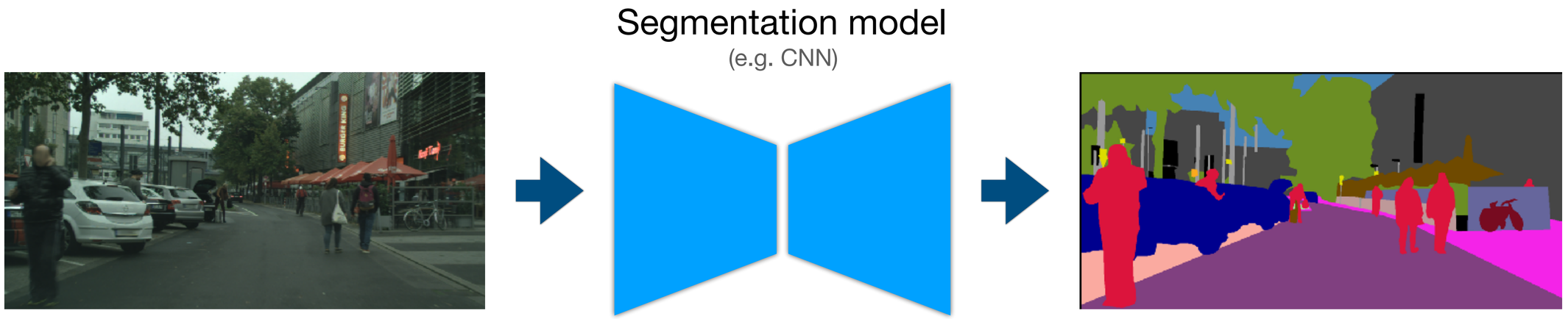
Course evaluation

- Course evaluation has started (05.12 – 23.12)
- You should have received invitations over email
- Results (without the comment fields) will be made available

- Please participate and give us feedback to improve the course!

Task definition

Label each pixel:



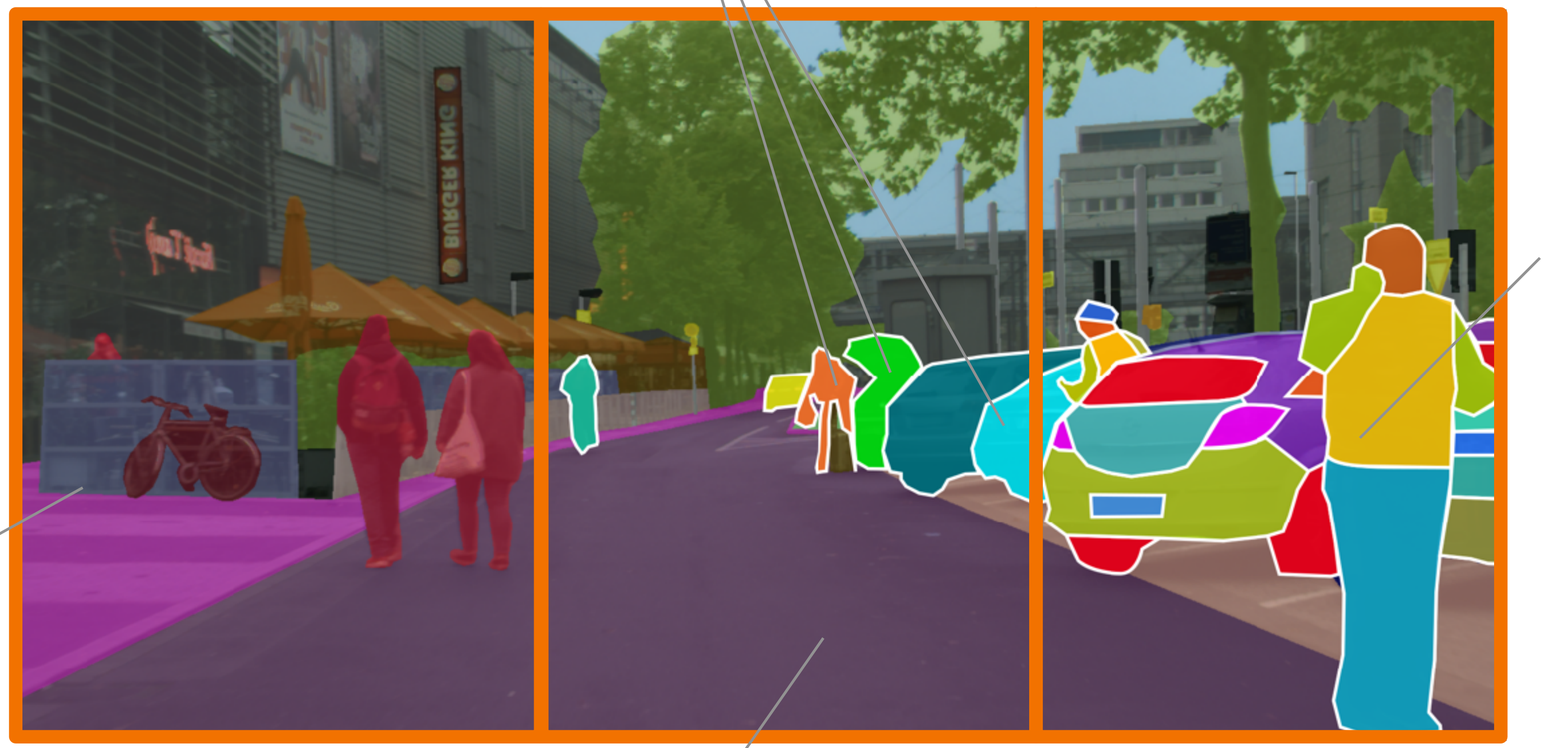
Flavours of image segmentation

- Semantic segmentation: label every pixel with a semantic category
- Instance segmentation: group object pixels as a separate category
  - Disregard background (“stuff”) classes, e.g. road, sky, building etc.
  - Can be class-agnostic or
  - with object classification: “semantic instance segmentation”
- Panoptic segmentation: semantic + instance segmentation
- Higher granularity, e.g. discriminating between object parts
  - “part segmentation”

Flavours of image segmentation

Example:

Instance segmentation
(only if we consider only “objects” / “things”)



Part segmentation
(typically one object)

Which part of instance

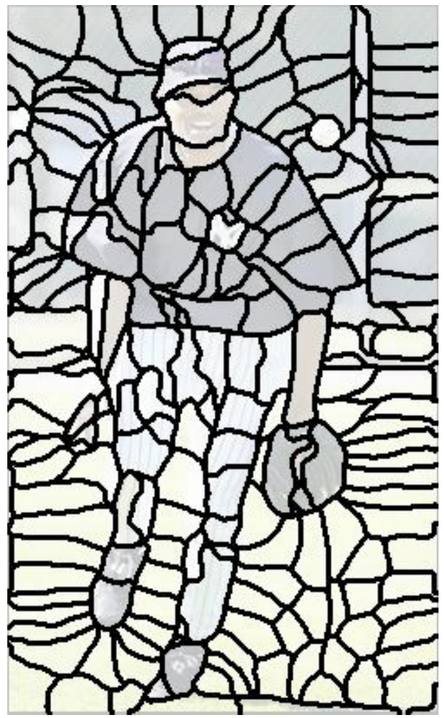
Semantic segmentation

2 person segmented some colour

Panoptic segmentation

Superpixels

- Local class-agnostic pixel grouping:



[Mori et al., 2004; Stefan Roth]

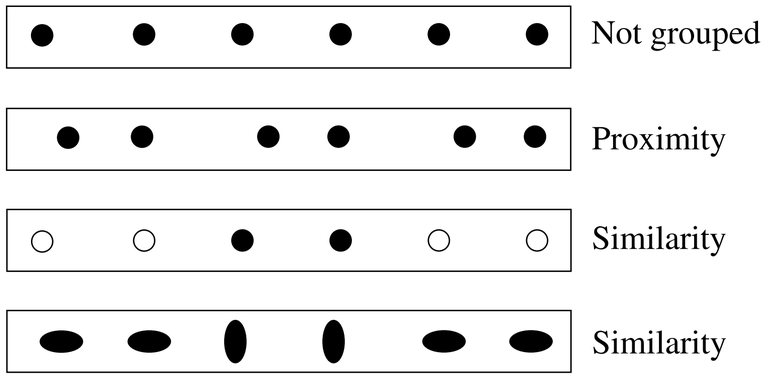
Why segment?

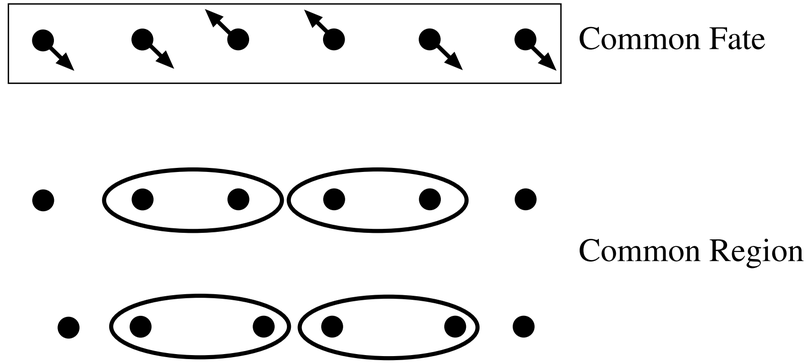
- Instead of having to work with all the pixels, a segmentation allows us to work with a much **more compact representation**.
- This is useful in practice, because this compact representation can make it easier to carry out certain tasks.
  - Scene understanding, object recognition, …
- Sometimes we are interested in the segmentation itself.
  - Especially in medical image analysis (e.g., segmenting out a tumor)

Adapted from [Stefan Roth]

What belongs together?

- To perform image segmentation, we need to decide which parts of the image belong together.
- What makes us humans believe parts of an image belong together?
- Gestalt Factors [Max Wertheimer, 1923]:





Adapted from [Stefan Roth]

Image segmentation with...

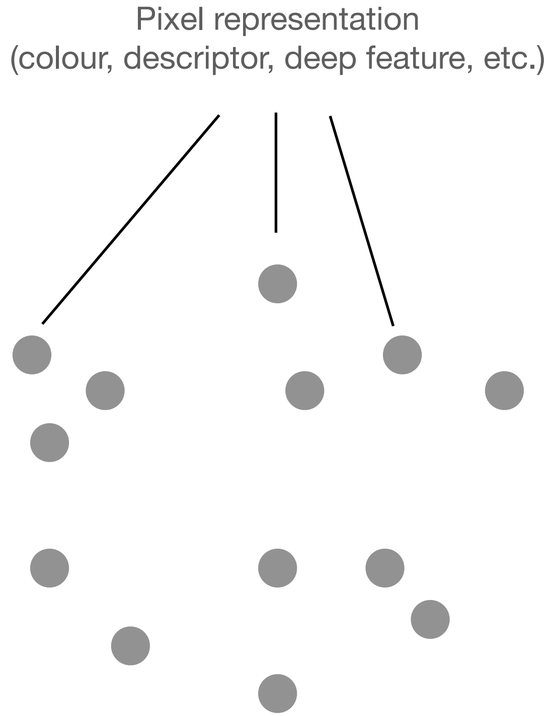
- Clustering, e.g.
  - K-means;
  - mean shift (Comaniciu and Meer, 2002);
  - spectral clustering, etc.
- Energy-based models
  - Conditional Random Fields (CRFs).
- Deep learning
  - Fully convolutional networks.

Image segmentation with...

- Clustering, e.g.
  - K-means;
  - mean shift (Comaniciu and Meer, 2002);
  - spectral clustering, etc.
- Energy-based models
  - Conditional Random Fields (CRFs).
- Deep learning
  - Fully convolutional networks.

Segmentation as clustering

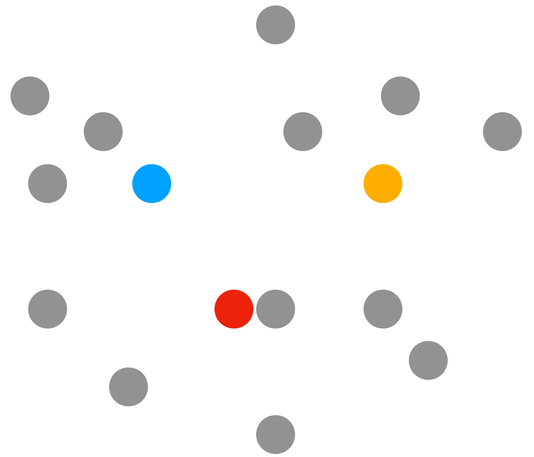
- K-means



Segmentation as clustering

- K-means

1. Initialise K cluster centres

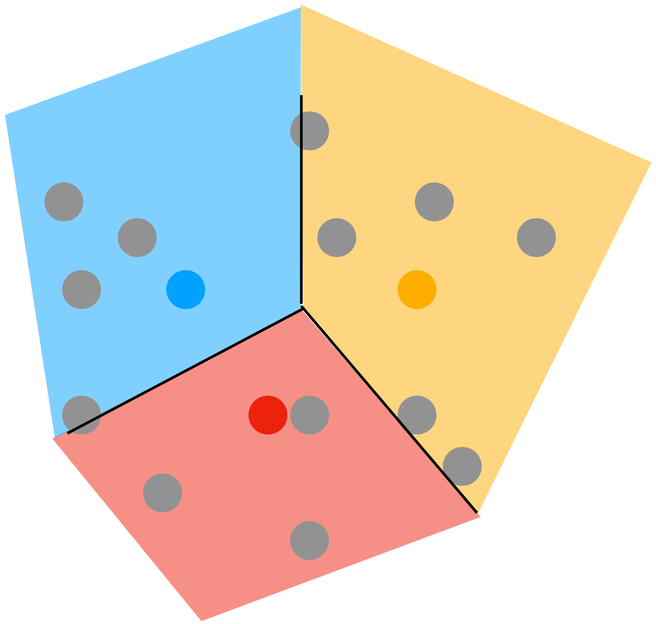


Segmentation as clustering

- K-means

1. Initialise (randomly) K cluster centres
2. Assign points to clusters

   using a distance metric

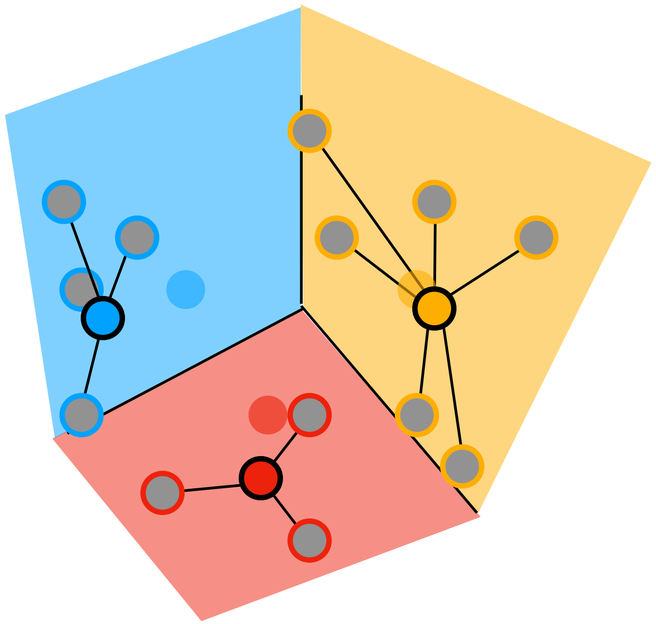


Segmentation as clustering

- K-means

1. Initialise (randomly) K cluster centres
2. Assign points to clusters

   using a distance metric
3. Update centres by averaging the points in the cluster

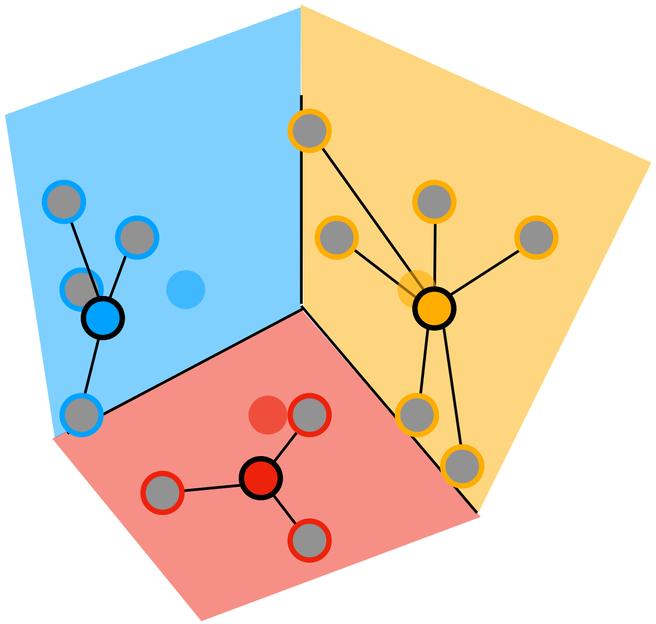


Segmentation as clustering

- K-means

1. Initialise (randomly) K cluster centres
2. Assign points to clusters

   using a distance metric
3. Update centres by averaging the points in the cluster
4. Repeat 2 and 3 until convergence

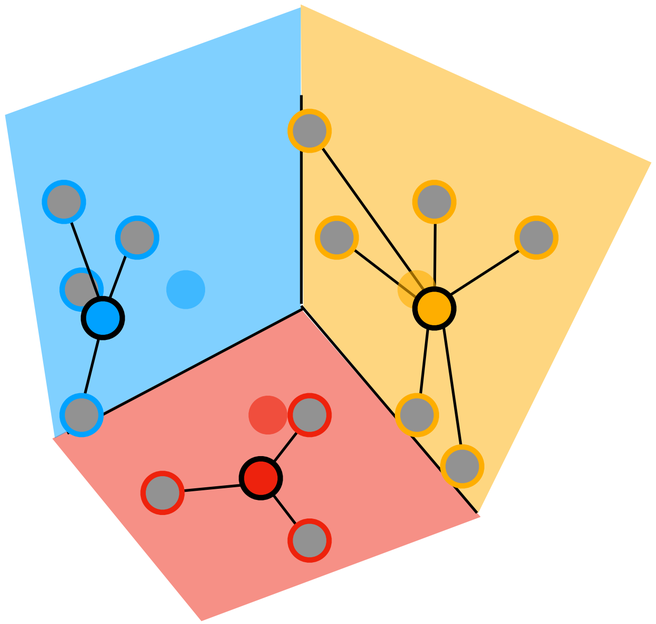


Segmentation as clustering

- K-means

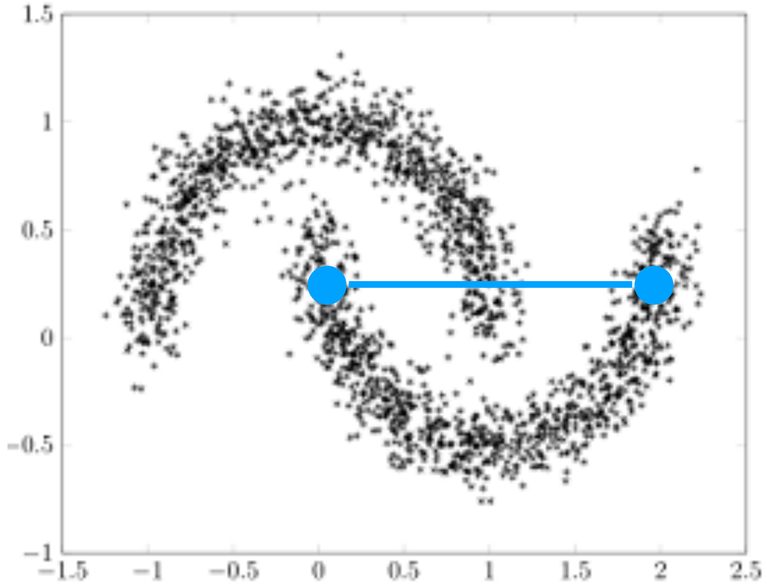
1. Initialise (randomly) K cluster centres
2. Assign points to clusters

   using a distance metric
3. Update centres by averaging the points in the cluster
4. Repeat 2 and 3 until convergence



Segmentation as clustering

- K-means
  - does it always work?



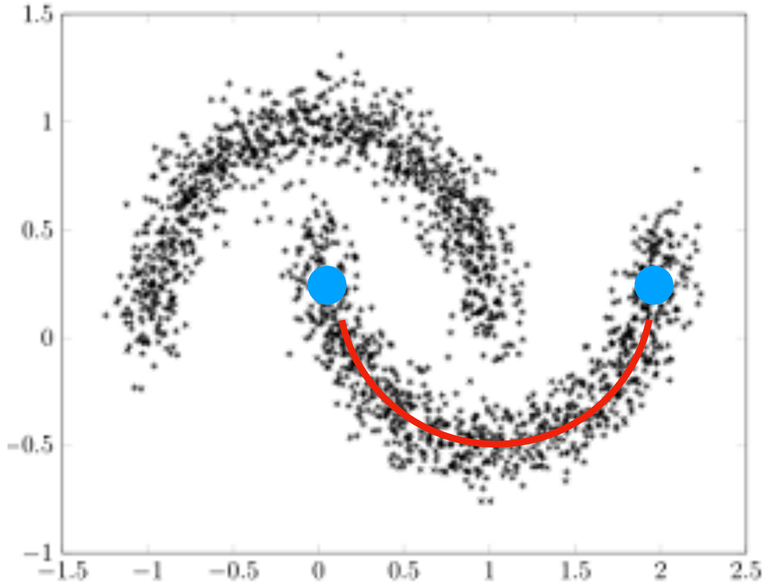
Shortest distance

[Mathworks]

- Minimising the total within-cluster distance may not be ideal...

Segmentation as clustering

- K-means
  - does it always work?



Shortest distance

[Mathworks]

- Minimising the total within-cluster distance may not be ideal...
  - ...unless we re-define the distance metric (recall metric learning)

Spectral clustering

- Represent all data points as fully connected (undirected) graph.
- Edge weights encode proximity between nodes.

↳ Yakinlik

$$\mathbf{A} := (a_{i,j}) \in \mathbb{R}^{n \times n}$$

- $a_{i,j} \geq 0$ measures the similarity of nodes i and j.
- we can use a distance metric: $a_{i,j} := \exp(-\gamma \cdot d(i,j))$
- $\mathbf{A}$ is symmetric.

Spectral clustering

- Define graph Laplacian:

$$L := D - A$$

- where D_{ii} is a diagonal matrix

$$D_{ii} = \sum_j A_{ij}$$

Spectral clustering

- We can show that:

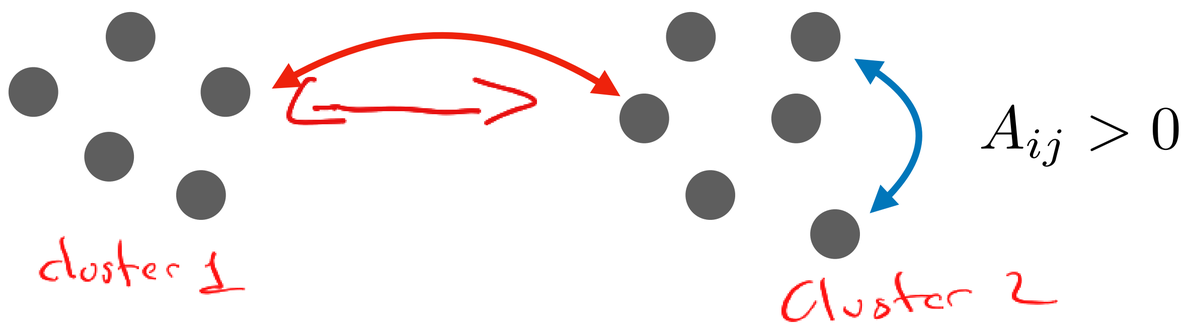
$$x^T L x = \frac{1}{2} \sum_{i,j} A_{ij} (x_i - x_j)^2$$

- L is symmetric and positive semi-definite.
- **Proposition**: The set of eigenvectors of L with eigenvalue 0 is spanned by indicator vectors $\mathbf{1}_0, \mathbf{1}_1, ,.., \mathbf{1}_K$ corresponding to K connected components of the graph.

Adapted from [Rudolph Triebel)

Spectral clustering

Example



- We can represent these n points as an $n \times K$ matrix ($K = 2$):

points in cluster 1

points in cluster 2

$$\begin{pmatrix} 1 & 0 \\ 1 & 0 \\ \vdots & \vdots \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

- The columns are eigenvectors of L corresponding to eigenvalue 0;
- Clustering these points in the K-dimensional space with K-means would now be more meaningful!

Spectral clustering

In nutshell:

- **Compute the first k eigenvectors $u_1, \ldots, u_k$ of L.**
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns.
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U.
- Cluster the points $(y_i)_{i=1,\ldots,n}$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, \ldots, C_k$.

Output: Clusters $A_1, \ldots, A_k$ with $A_i = \{j | y_j \in C_i\}$.

von Luxburg “A tutorial on spectral clustering” (2007)

Spectral clustering

Normalising graph Laplacian enhances cluster properties:

- Shi and Malik, 2000 (“normalised cuts”):

$$L_{\mathrm{rw}} := D^{-1/2} L D^{-1/2} = I - D^{-1/2} A D^{-1/2}$$

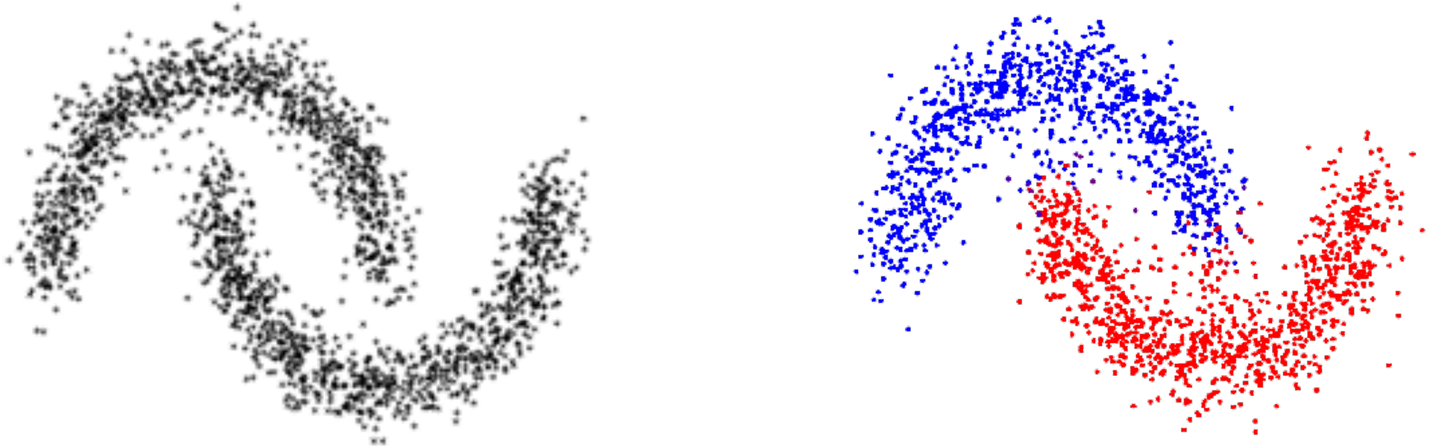
- Ng et al., 2002:

$$L_{\mathrm{sym}} := D^{-1} L = I - D^{-1} A$$

Good Example

von Luxburg “A tutorial on spectral clustering” (2007)

Spectral clustering



[Mathworks]

- Spectral clustering can handle complex distributions
- The complexity is $O(n^3)$ due to Eigendecomposition
- There are efficient variants (e.g., using sparse affinity matrices)

Adapted from [Rudolph Triebel)

Image segmentation with...

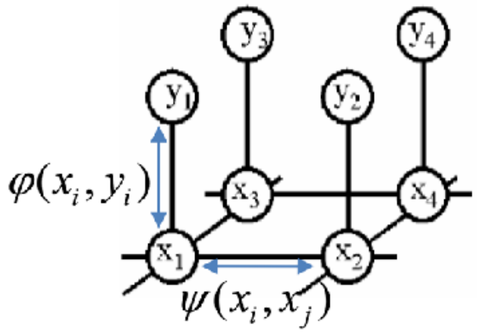
- Clustering, e.g.
  - K-means;
  - mean shift (Comaniciu and Meer, 2002);
  - spectral clustering, etc.
- Energy-based models
  - Conditional Random Fields (CRFs).
- Deep learning
  - Fully convolutional networks.

Energy formulation

- Energy function:

$$E(x, y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} \psi(x_i, x_j)$$

unary term pairwise term



- Unary potential:
  - encodes local information about a pixel/patch
  - how likely is it to belong to a certain class (e.g. foreground/background)?

↳ How likely given pixel is given class

- Pairwise potential:
  - encode neighbourhood information
  - how different is this pixel/patch to its neighbours (e.g. based on colour/texture/learned feature)?

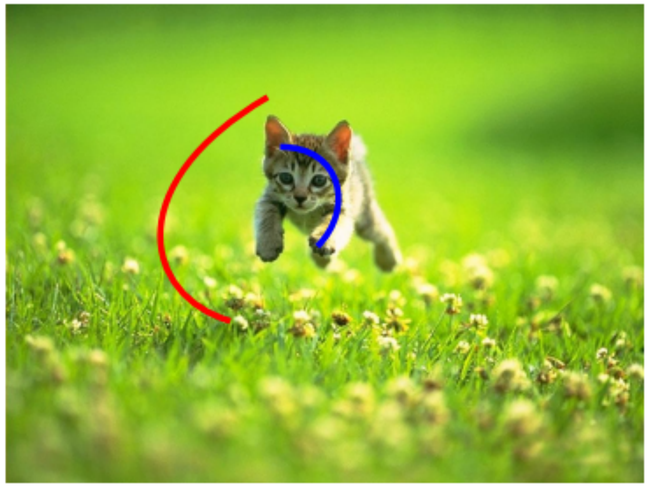
Slide credit: Bastian Leibe

Conditional Random Fields (CRF)

- Boykov and Jolly (2001)

$$E(x, y) = \sum_i \varphi(x_i, y_i) + \sum_{ij} \psi(x_i, x_j)$$

- Variables
  - x_i: Binary variable
    - ★ foreground/background
  - y_i: Annotation
    - ★ foreground/background/empty
- Unary term
  - $\varphi(x_i, y_i) = K[x_i \neq y_i]$
  - Pay a penalty for disregarding the annotation
- Pairwise term
  - $\psi(x_i, x_j) = [x_i \neq x_j] w_{ij}$
  - Encourage smooth annotations
  - w_{ij} affinity between pixels i and j



Slide credit: Philipp Krahenbuhl

Boykov and Jolly “Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images” (2001).

Conditional Random Fields (CRF)

- Grid structured random fields
  - efficient solution using Maxflow/Mincut
  - Optimal solution for binary labelling
  - Boykov & Komogorov “An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision” (2004)
- Fully connected models:
  - Efficient solution using mean-field
  - Krähenbühl & Koltun “Efficient Inference in Fully-Connected CRFs with Gaussian edge potentials” (2011)

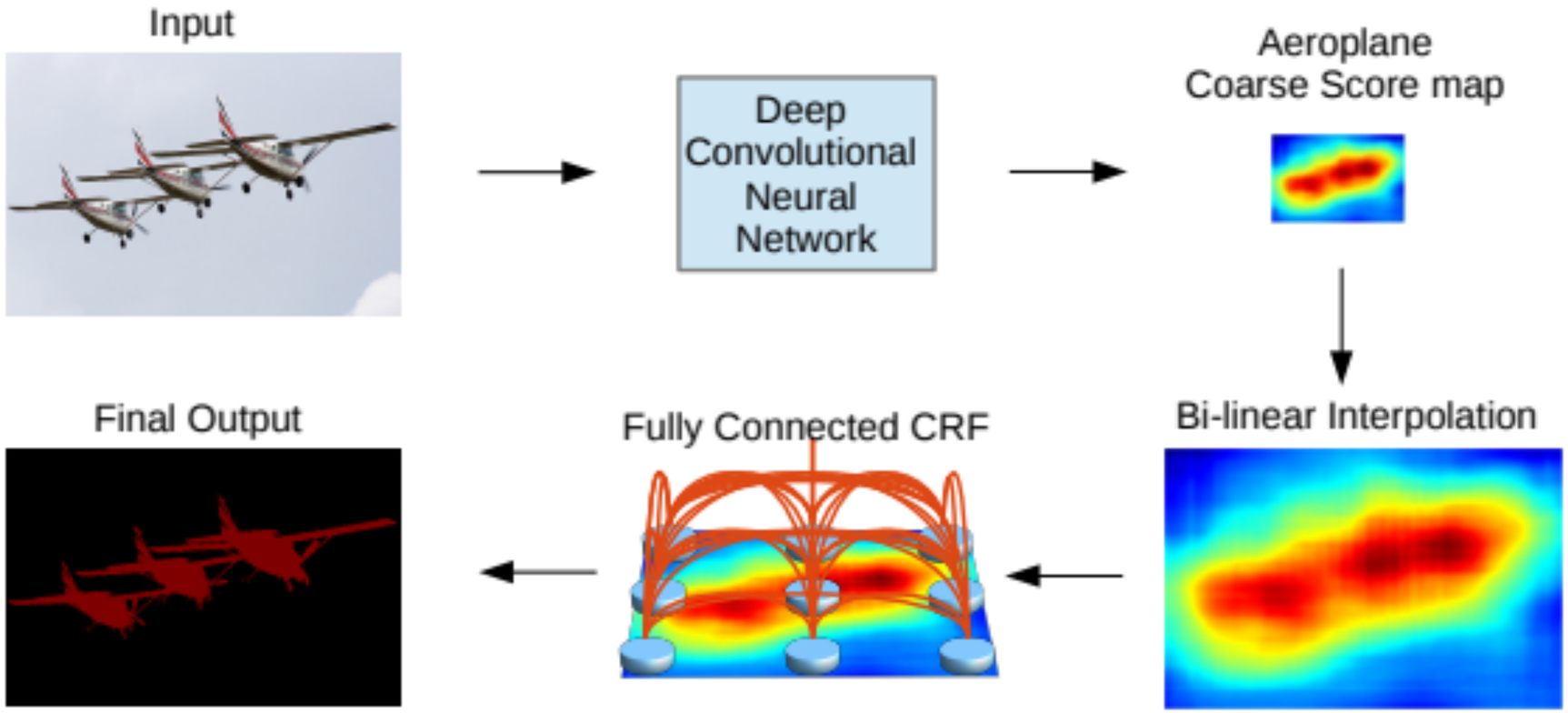
Slide credit: Philipp Krahenbuhl

Boykov and Jolly “Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images” (2001).

Remark

- “Classical” methods are still highly relevant today
- Deep networks learn useful feature representation (or metrics)
  - standard clustering (e.g. K-means) can be applied
- Producing deep features for every image pixel is challenging (later today)
  - (fully connected) CRFs is a useful off-the-shelf post-processing tool;
  - still in heavy today.

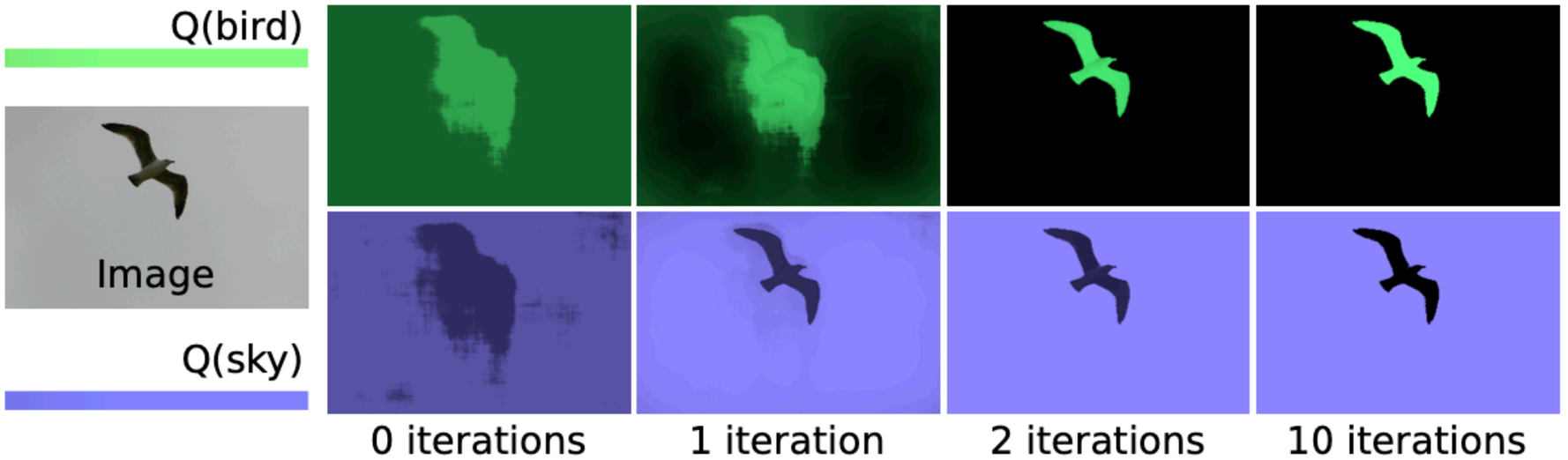
DeepLab



Chen et al., “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs” (2016).

CRF Iterations

Using fully connected CRFs:



(b) Distributions $Q(X_i = \text{“bird”})$ (top) and $Q(X_i = \text{“sky”})$ (bottom)

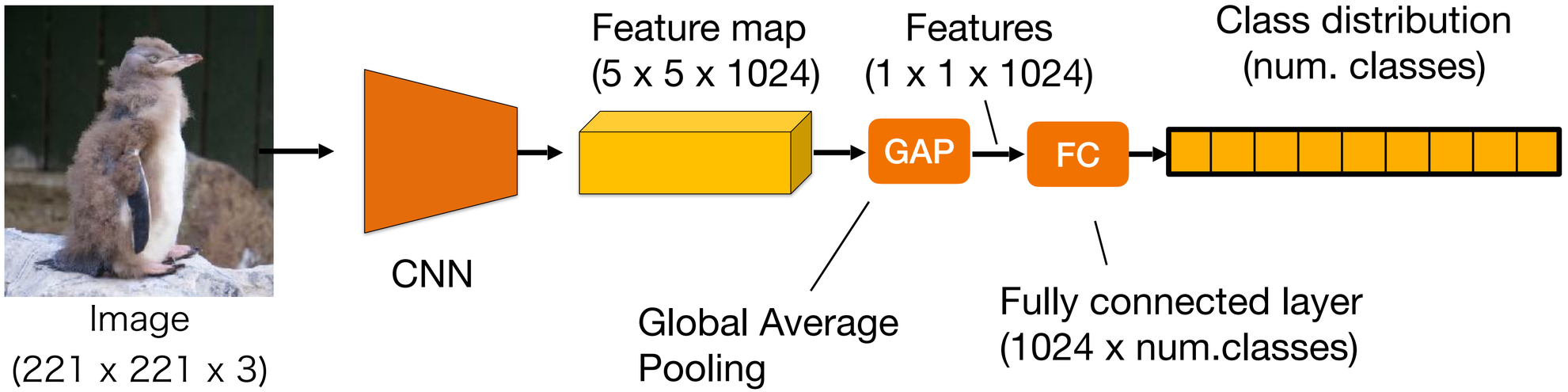
Krähenbühl & Koltun, 2011

Reduce segmentation to...

- Clustering, e.g.
  - K-means;
  - mean shift (Comaniciu and Meer, 2002);
  - spectral clustering, etc.
- Energy-based models
  - Conditional Random Fields (CRFs).
- Deep learning
  - Fully convolutional networks.

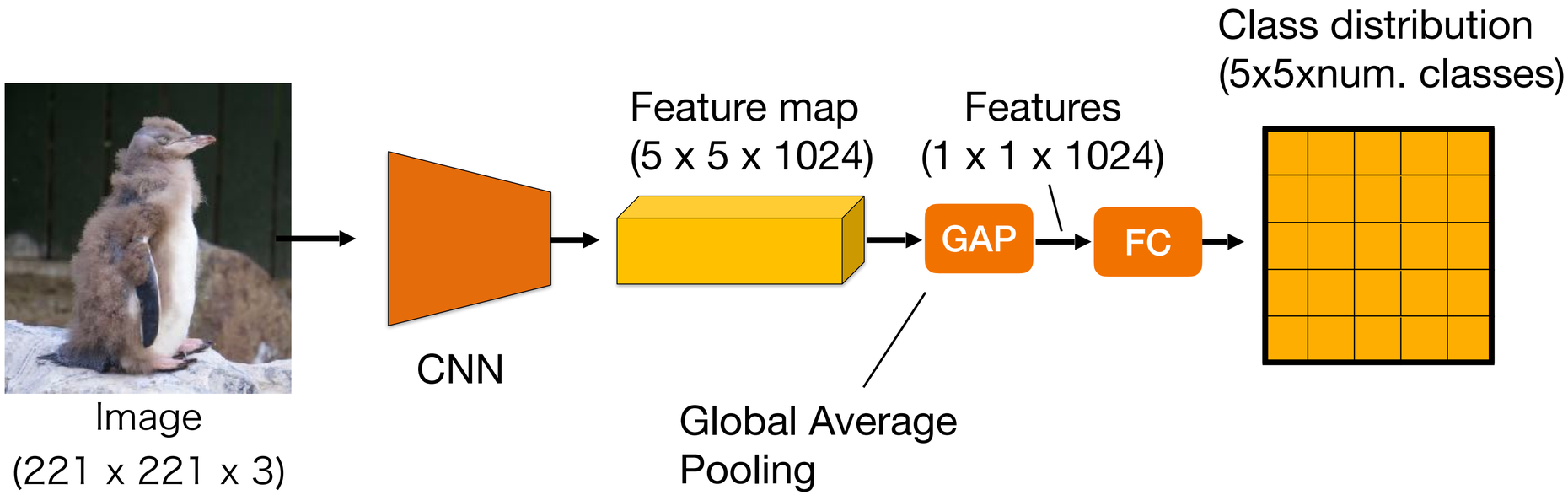
Deep networks for classification

- Recall deep networks for classification



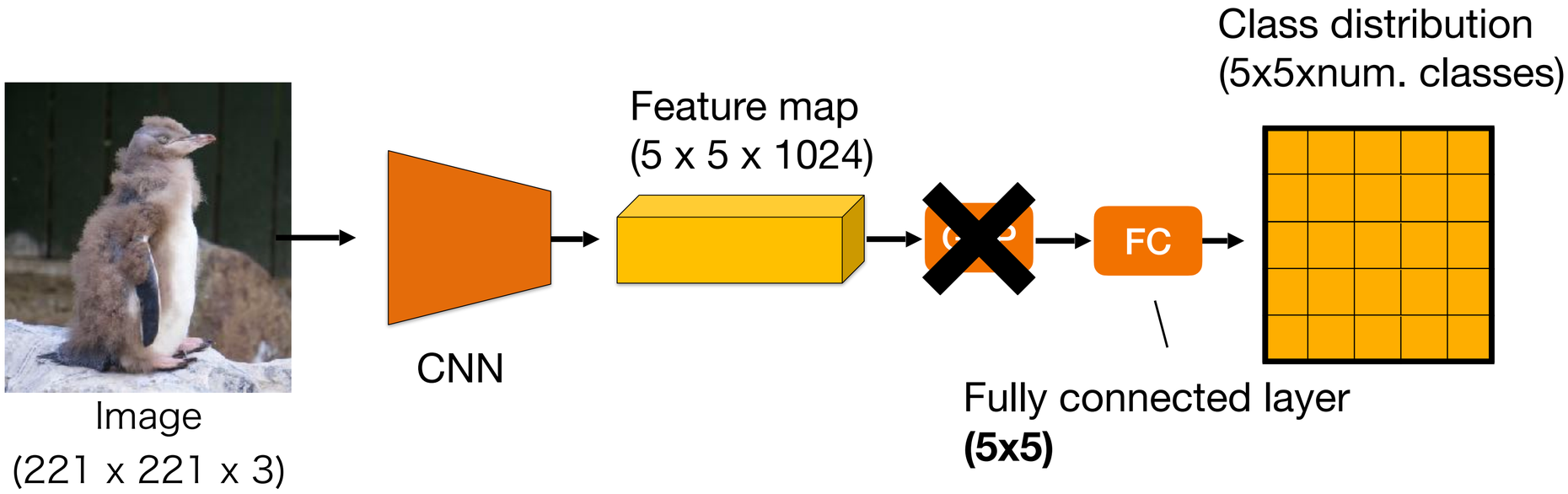
Deep networks for classification

- How can we extend this to segmentation?



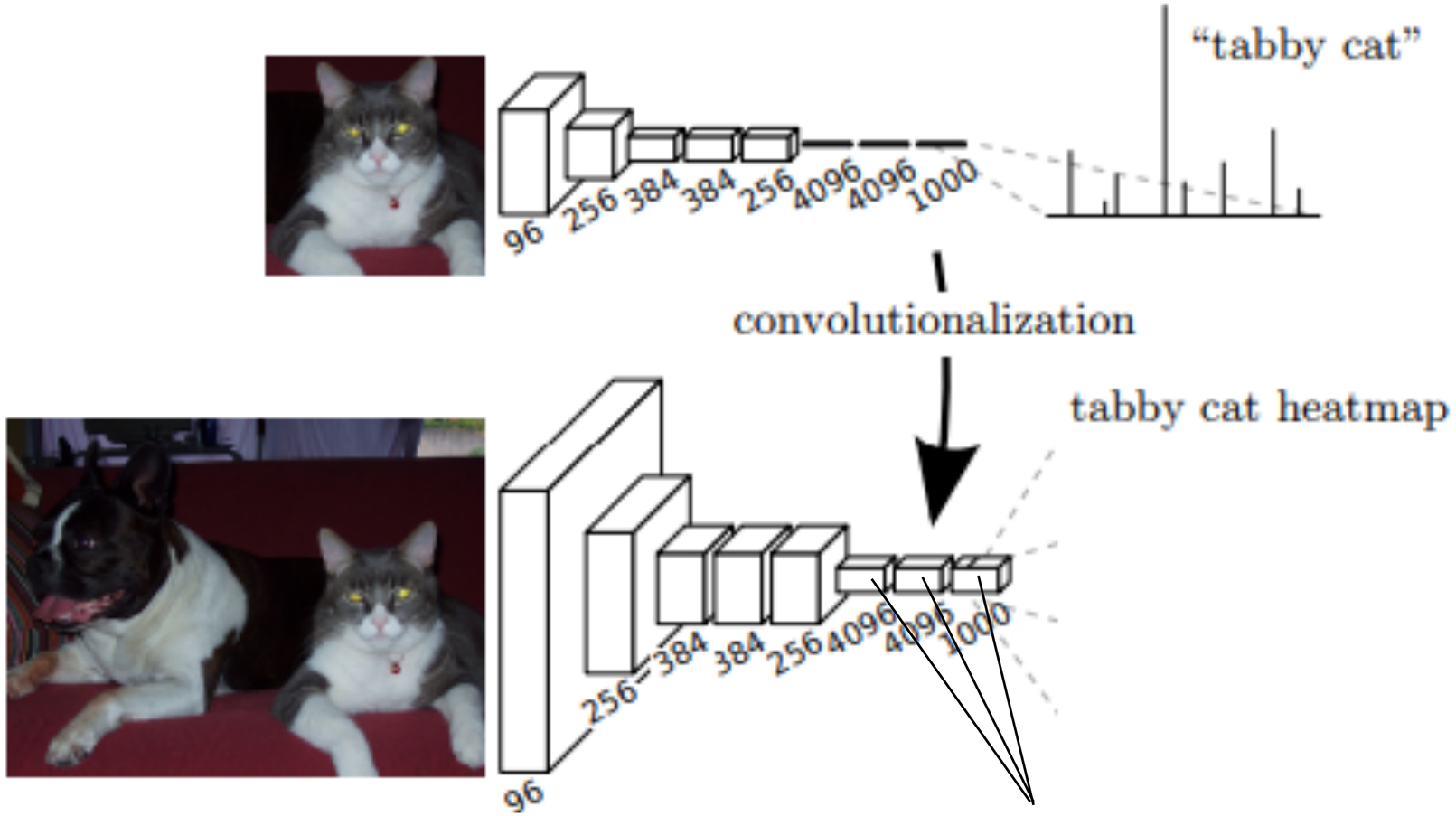
Deep networks for classification

- Let us remove GAP:



– parameter increase; fixed input size only; no translation invariance.

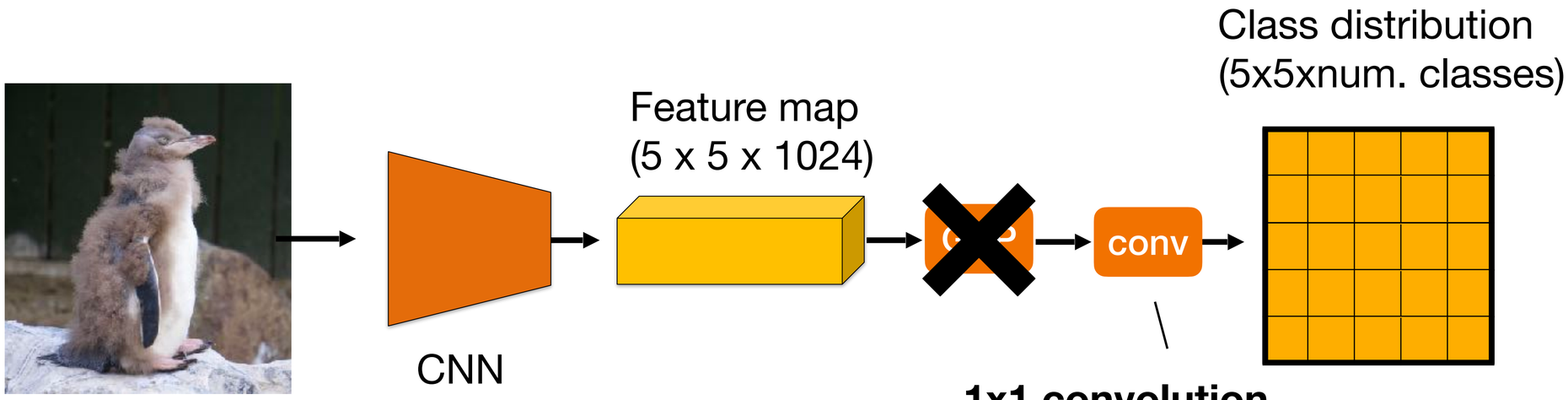
“Convolutionalization”



1x1 Convolutions

Deep networks for classification

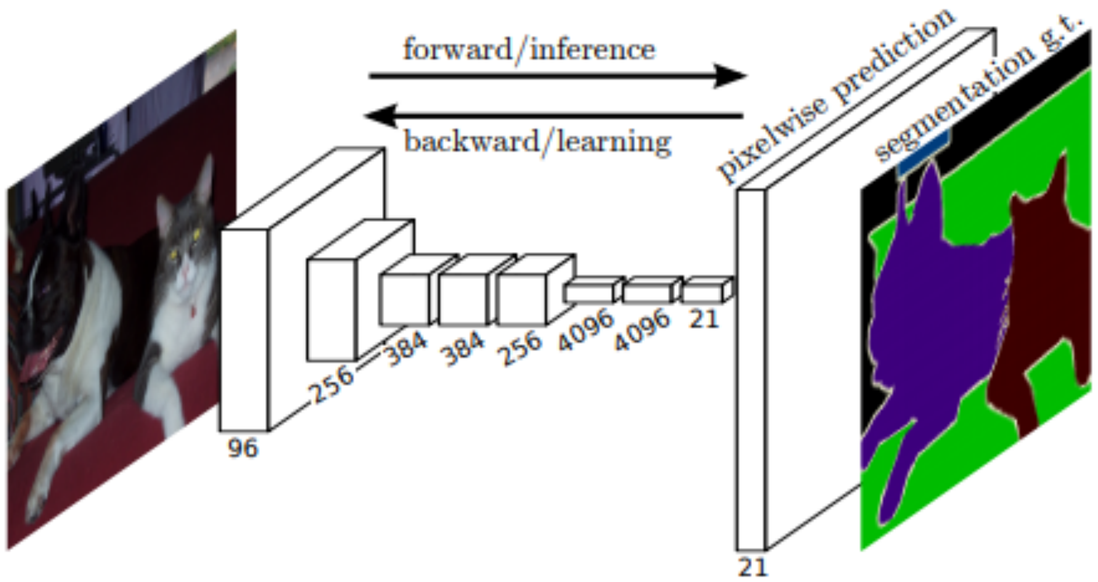
- Replace fully connected layer with convolution:



QUIZ: How many parameters do we have here?

- same num. of parameters; variable input size; translation invariance!

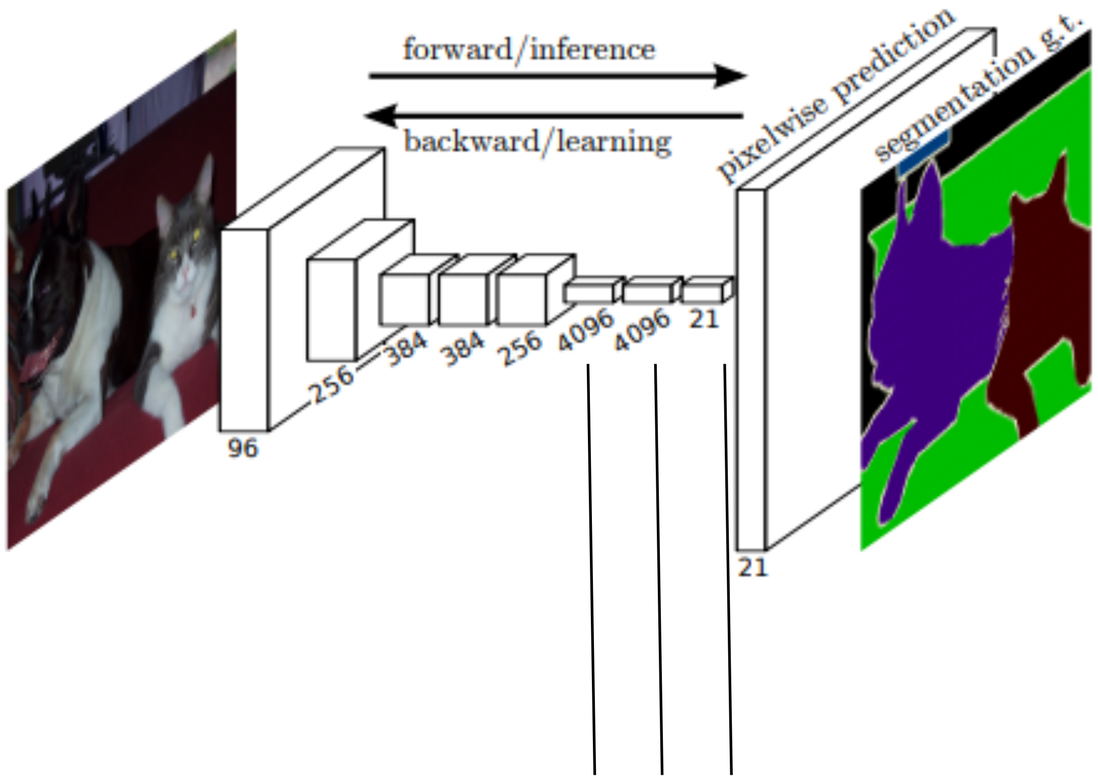
Fully convolutional neural networks



Long, Shelhamer, Darrell. “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015, PAMI 2016.

Fully convolutional neural networks

- Replace FC layers with convolutional layers.
- Upsample the last layer output to the original resolution.
- Can be trained with pixelwise cross-entropy with SGD.



Long, Shelhamer, Darrell. “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015, PAMI 2016.

Recall: Convolutions on Images

Image 5x5

-5	3	2	-5	3
4	3	2	1	-3
1	0	3	3	5
-2	0	1	4	4
5	6	7	9	-1

Kernel 3x3

0	-1	0
-1	5	-1
0	-1	0

Output 3x3

6		

$$5 \cdot 3 + (-1) \cdot 3 + (-1) \cdot 2 + (-1) \cdot 0 + (-1) \cdot 4 =$$
$$15 - 9 = 6$$

1x1 convolution

Image 5x5

-5	3	2	-5	3
4	3	2	1	-3
1	0	3	3	5
-2	0	1	4	4
5	6	7	9	-1

Kernel 1x1

2

1x1 convolution

Image 5x5

-5	3	2	-5	3
4	3	2	1	-3
1	0	3	3	5
-2	0	1	4	4
5	6	7	9	-1

Kernel 1x1

2

-10				

$$-5 * 2 = -10$$

1x1 convolution

Image 5x5

-5	3	2	-5	3
4	3	2	1	-3
1	0	3	3	5
-2	0	1	4	4
5	6	7	9	-1

-10	6	4	-10	6
8	6	4	2	-6
2	0	6	6	10
-4	0	2	8	8
10	12	14	18	-2

Kernel 1x1

2

$$-1 * 2 = -2$$

1x1 convolution

Image 5x5

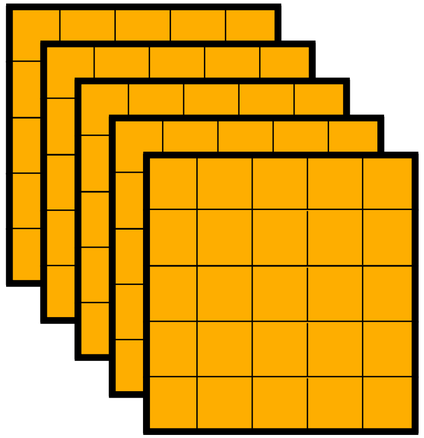
-5	3	2	-5	3
4	3	2	1	-3
1	0	3	3	5
-2	0	1	4	4
5	6	7	9	-1

-10	6	4	-10	6
8	6	4	2	-6
2	0	6	6	10
-4	0	2	8	8
10	12	14	18	-2

- 1x1(x1) kernel: keeps the dimensions and scales input

1x1 convolution

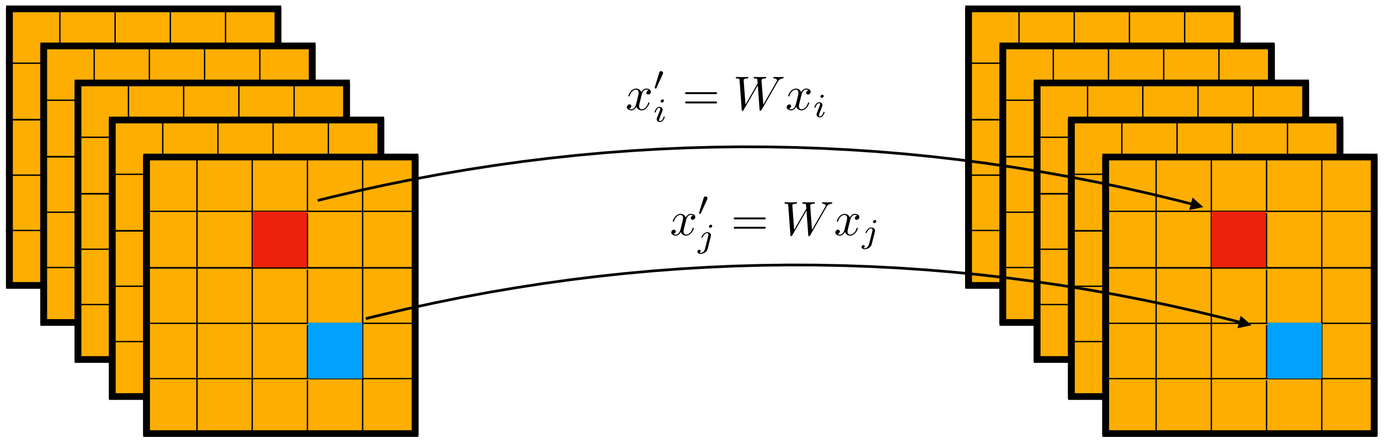
- Simply scaling the input? How is this useful?
- Every (feature) pixel is a multi-dimensional feature:



(h x w x D)

1x1 convolution

- Simply scaling the input? How is this useful?
- Every (feature) pixel is a multi-dimensional feature:



- 1x1 convolution is equivalent to applying a **shared** fully connected layer to every pixel!

1x1 convolution facts

- 1x1 convolution is a pixel-wise linear projection:

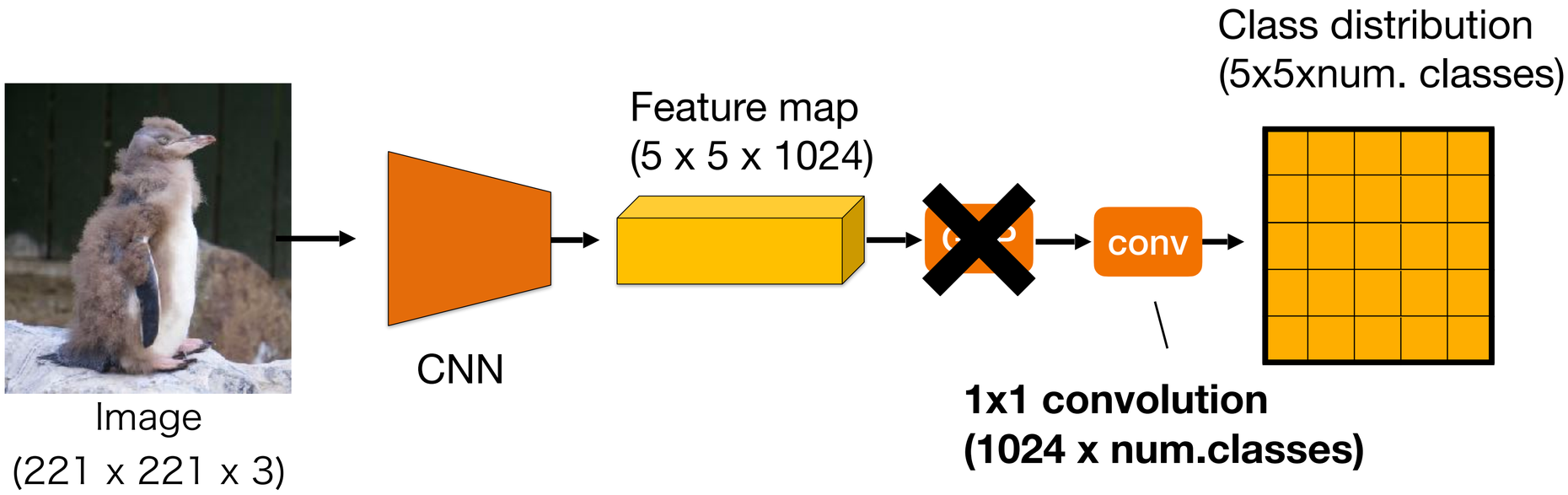
$$X' := W X$$

$[D', \text{pixels}] \quad [D', D] \quad [D, \text{pixels}]$

- each pixel is treated the same (shared parameters)
  - e.g. contrast this to 3x3 convolution;
- the output is treated as in fully connected layers:
  - followed by normalisation, non-linearity (except for the output layer)

Deep networks for classification

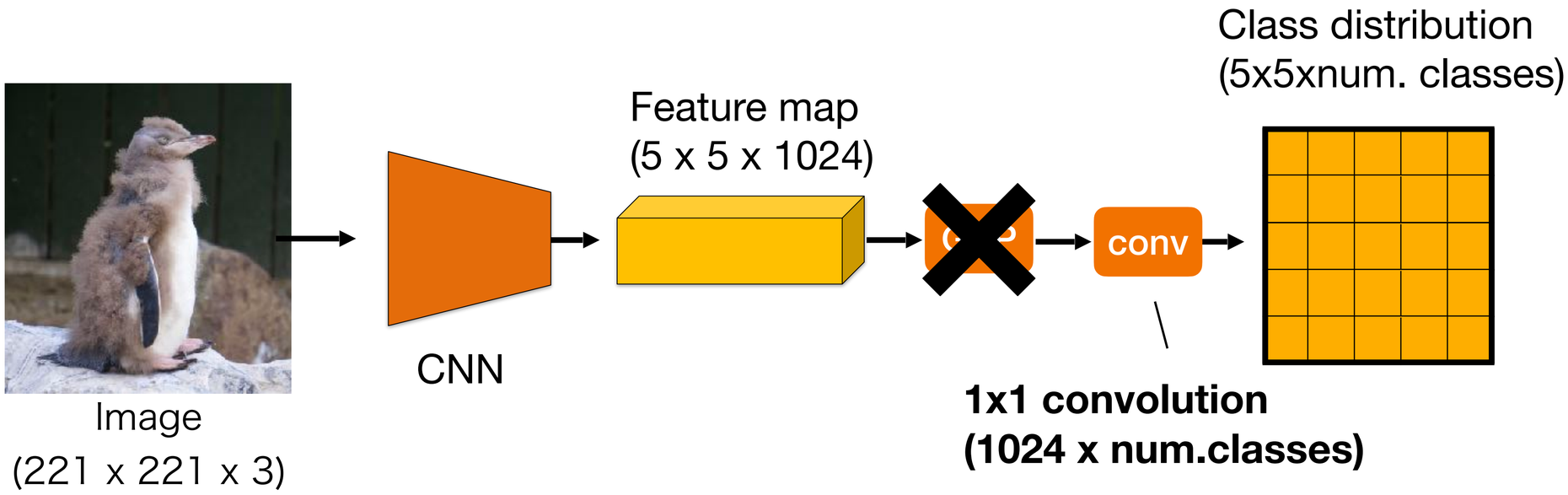
- Recall: we produce only a coarse grid



- How to produce segmentation result for each image pixel?

Deep networks for classification

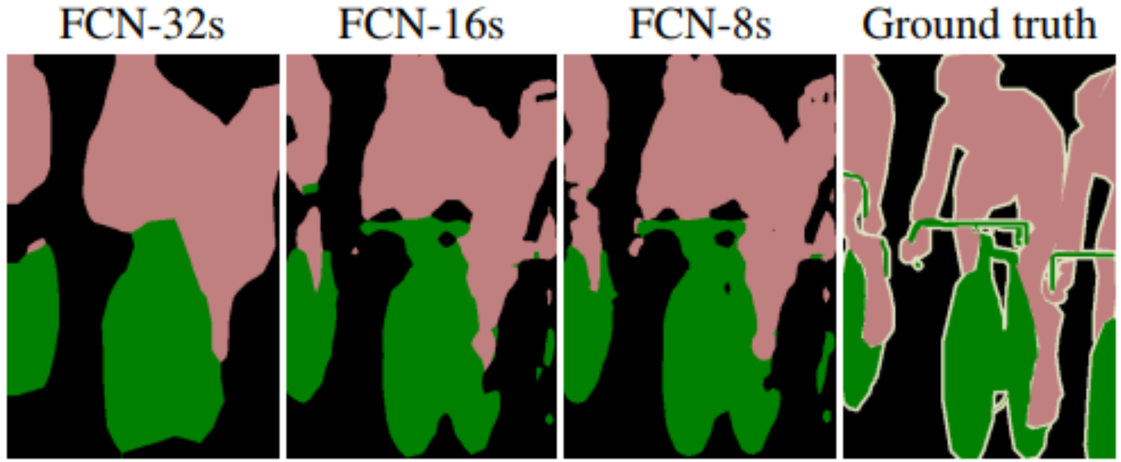
- Recall: we produce only a coarse grid



- How to produce segmentation result for each image pixel?

Qualitative results

- We may try to maintain original image resolution in the encoder:



- Decreasing the output stride improves segmentation accuracy.
- Quiz: Why not keep feature resolution high in all network layers?

Long, Shelhamer, Darrell. “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015, PAMI 2016.

Feature resolution

Two problems:

- receptive field size:
  - removing an operation with stride 2 reduces the area of the receptive field by a factor of ~4;
  - limited access to context information.
- computational footprint (both memory and FLOPs):
  - e.g. feature tensor size increases 4 times for each remove stride-2 operation.

Dilated convolutions

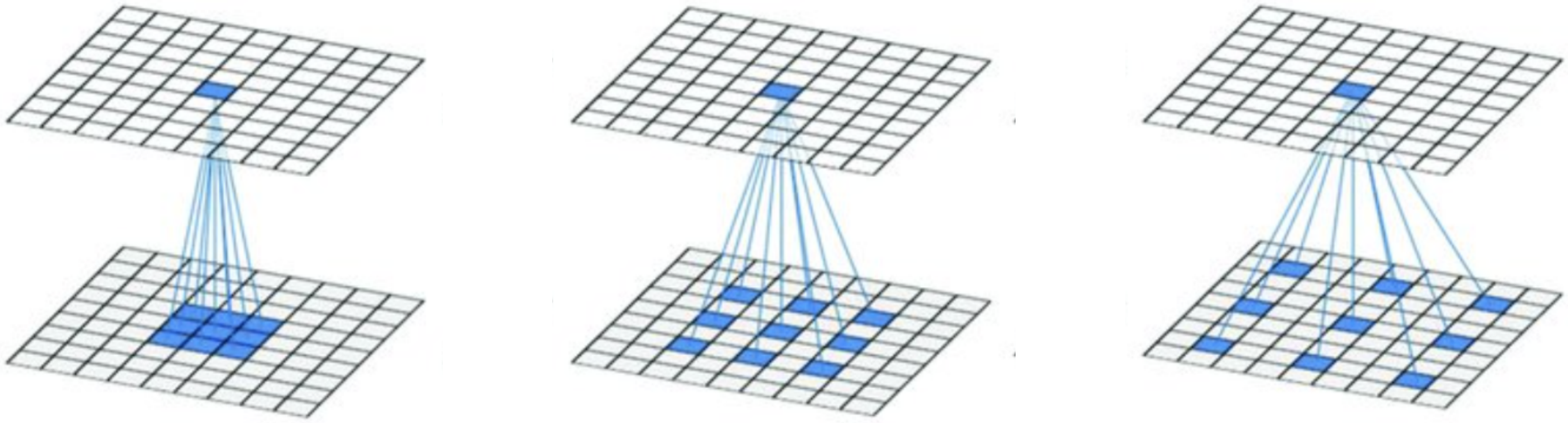
How to maintain the receptive field size?

- we can replace a stride-2 operation with multiple stride-1 operations
- deeper networks → harder to train
- impractical: custom architecture for a desired output feature resolution.

- a better alternative: dilated convolutions.

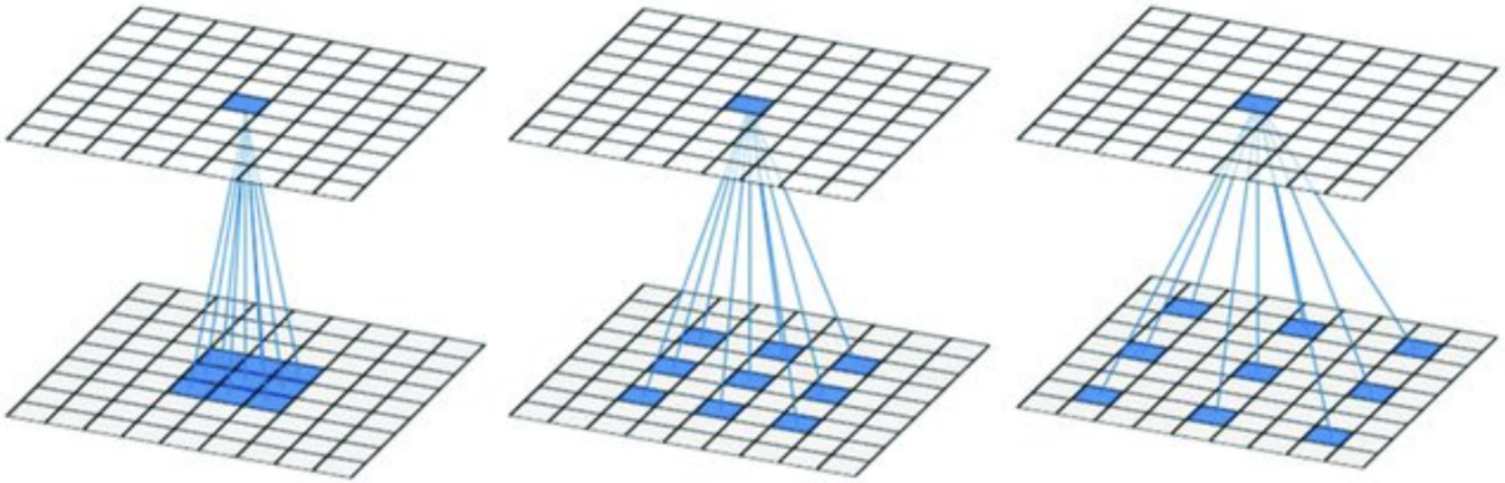
Dilated convolutions 2D

- Consider convolution as a special case with “dilation” = 1;
- For dilation N, the kernel “skips” N-1 pixels in-between:



dilation = 1 dilation = 2 dilation = 3

Dilated convolutions 2D



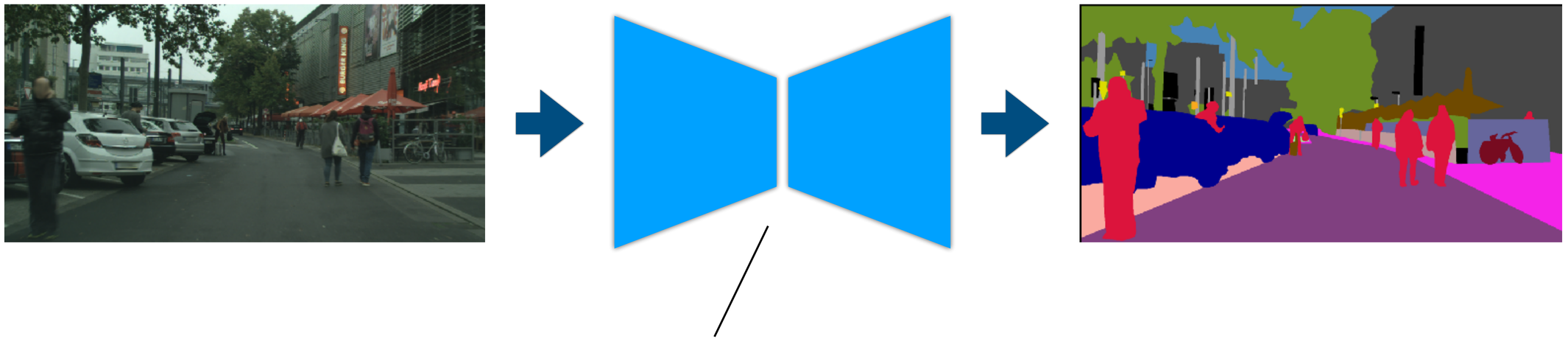
Note:

- The number of parameters remains the same;
- The receptive field size of kernel size K and dilation D:

$$D(K-1)+1$$

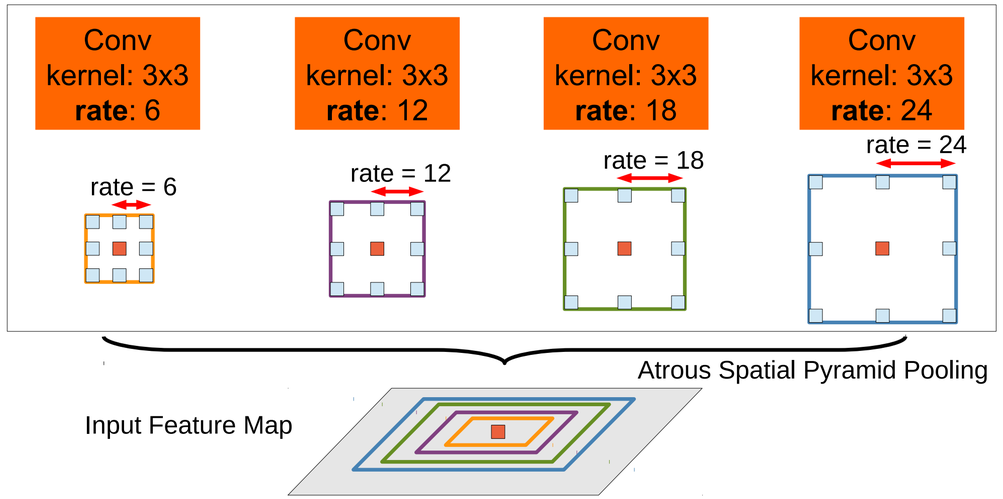
Dilated convolutions 2D

- Dilation improves scale invariance
  - we can use multiple dilations in the same layer:

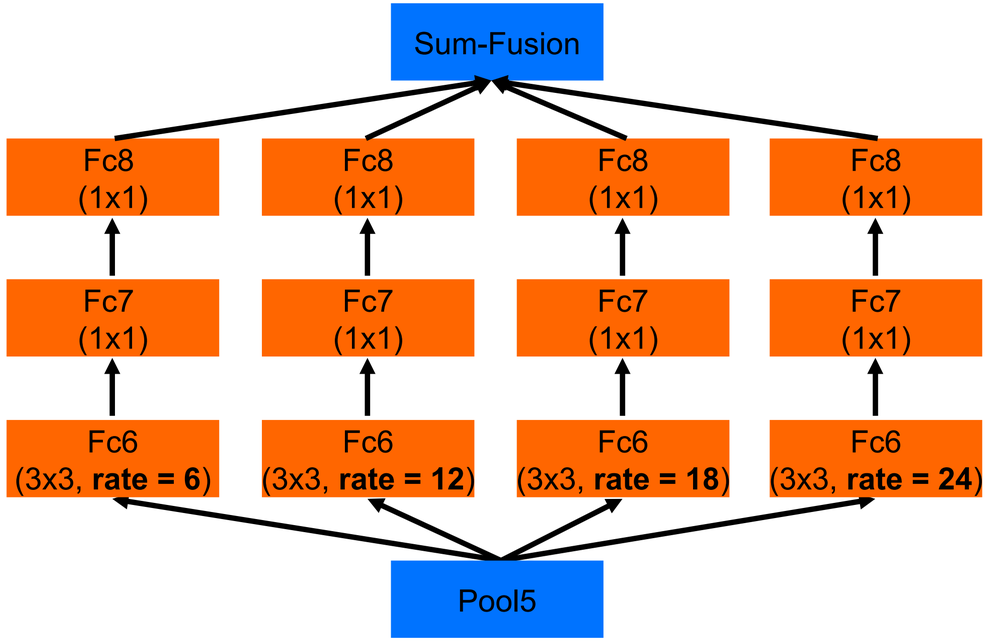


Typically at the “bottleneck” section

ASPP



Atrous Spatial Pyramid Pooling



DeepLab-ASPP

Chen et al., “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs” (2016).

Feature resolution

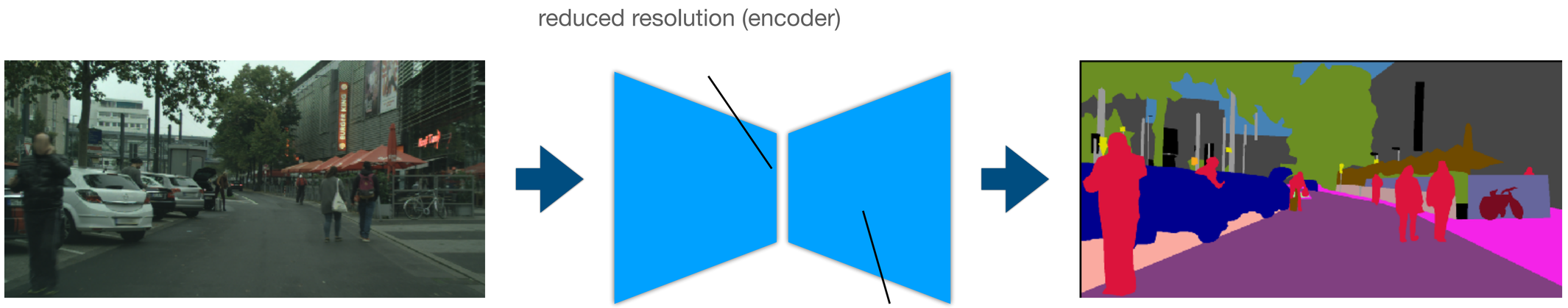
Two problems:

- receptive field size:
  - removing an operation with stride 2 reduces the area of the receptive field by a factor of 4;
  - less context information may get used.
- computational footprint (both memory and FLOPs):
  - e.g. feature tensor size increases 4 times for each remove stride-2 operation.

Replace stride-2 OPs with dilation-2 OPs

Feature resolution

- We have to make computation tractable by reducing feature resolution.
  - even though the receptive field size may not be an issue anymore.
- Perhaps, we can develop effective upsampling strategies.

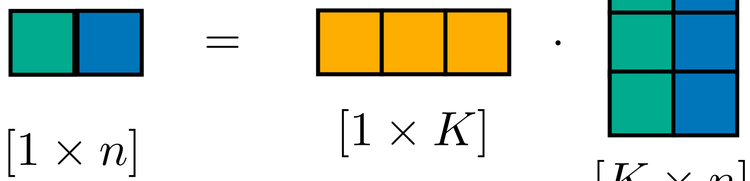


Upsampling

Transposed convolution

- Can we use convolution to increase the output resolution?
- Recall convolution (as matrix multiplcation):

$$X' = WX \longrightarrow$$



$[1 \times n]$ $[1 \times K]$ $[K \times n]$

- We can obtain the opposite effect (increase the output size) by

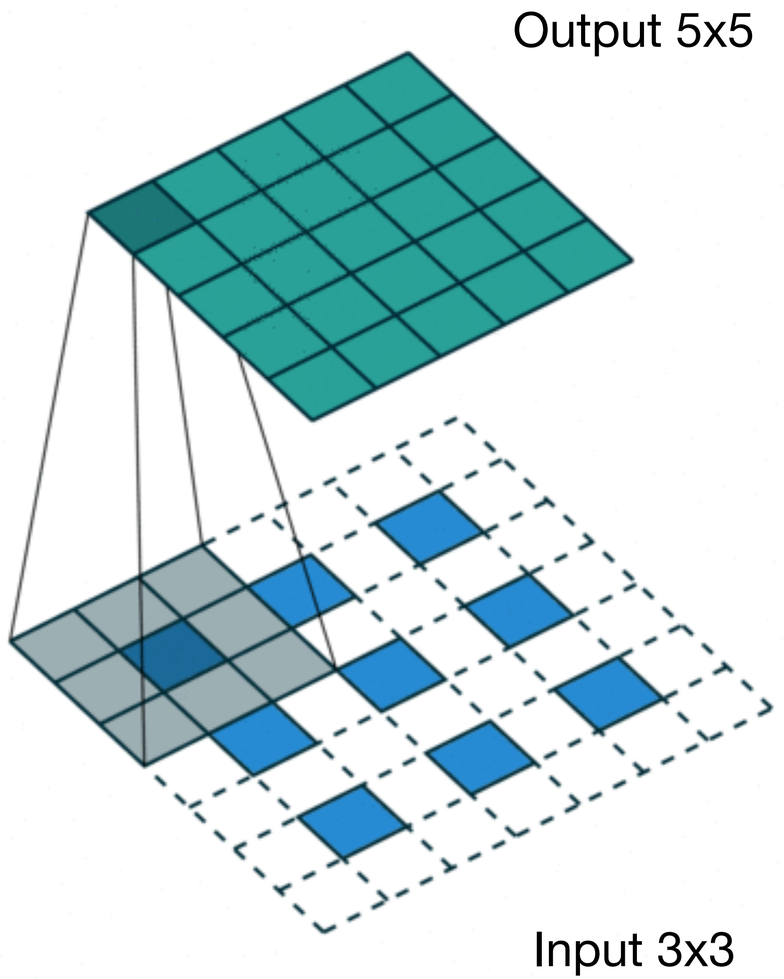
$$X = W^T X' \qquad \text{“broadcasting”}$$

$[K \times n]$ $[K \times 1]$ $[1 \times n]$

Transposed convolution

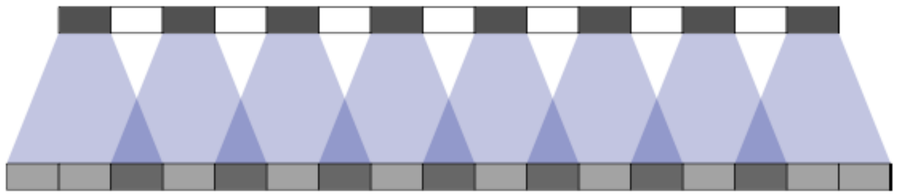
- Transposed convolution
  - also called “up-convolution”
  - or “deconvolution” (incorrect)

- Pad each pixel in the input (e.g. zeros)
- Convolve with a kernel (e.g. 3x3)
- The amount of padding and stride determines the output resolution



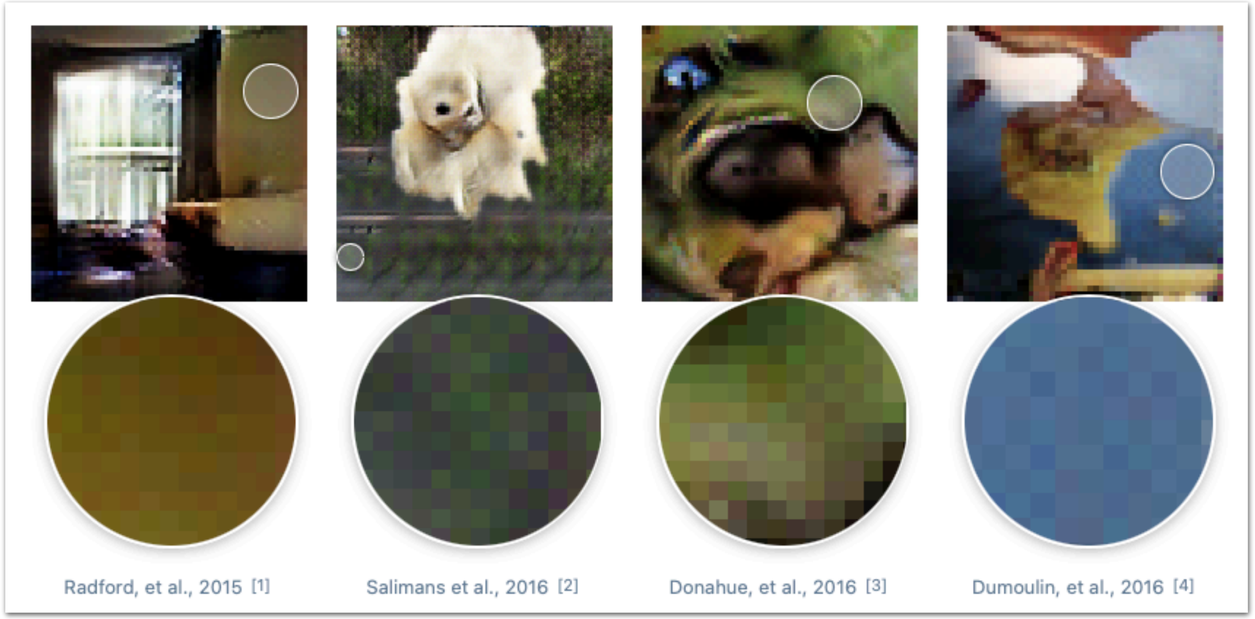
Transposed convolution

- Equivalent implementation without padding:



- Issue: checkboard artefacts

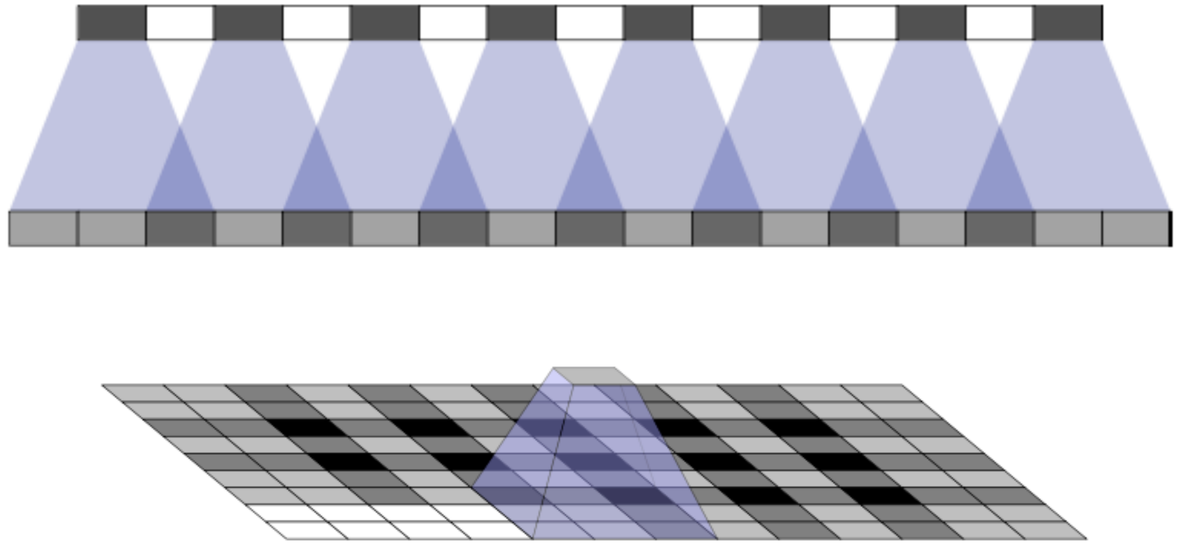
https://distill.pub/2016/deconv-checkerboard/



Transposed convolution

- Issue: checkboard artefacts. Why?
- “uneven overlap” when kernel size is not divisible by stride:

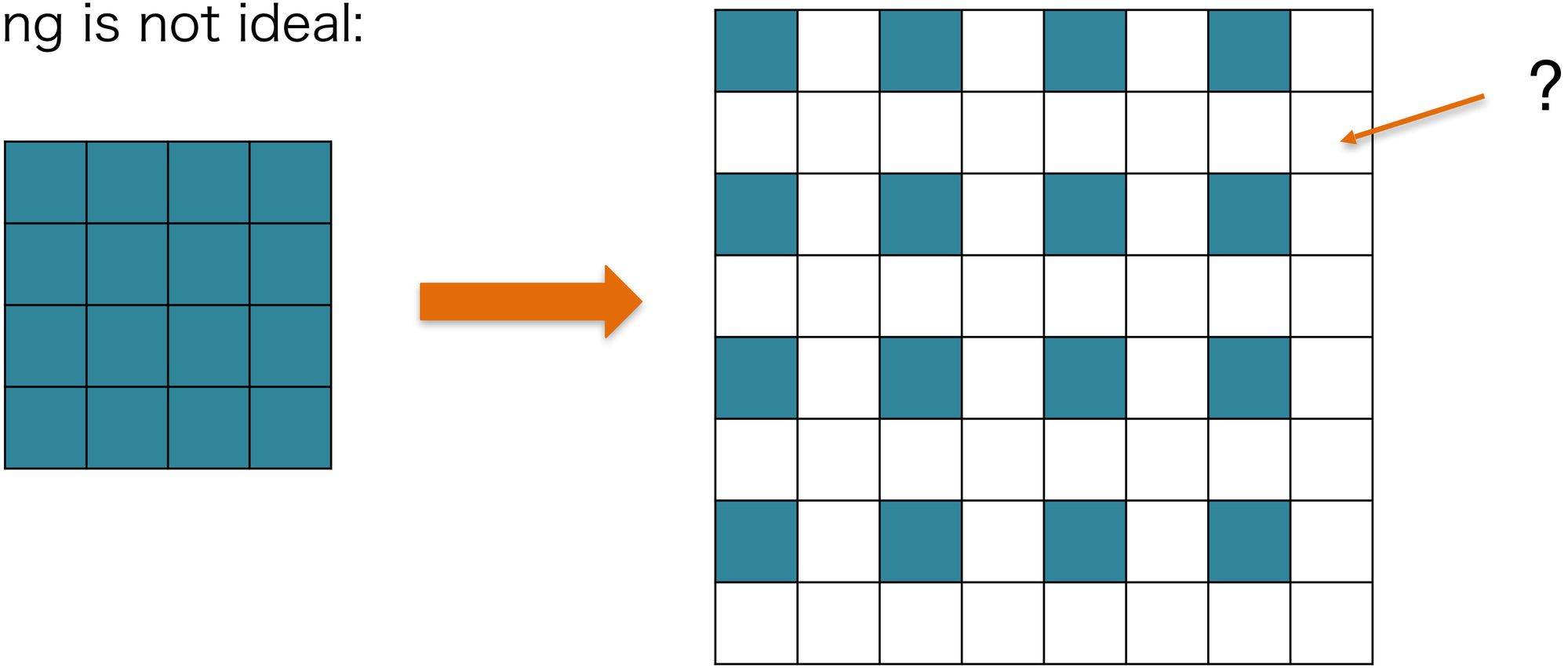
Kernel size: 3
Stride: 2



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

Upsampling: Interpolation

- Padding is not ideal:



- Solution: interpolation (e.g. bilinear) followed by standard convolution

Interpolation

- Features can be interpolated just as an image:

Original image

x 10

Nearest neighbor interpolation

Bilinear interpolation

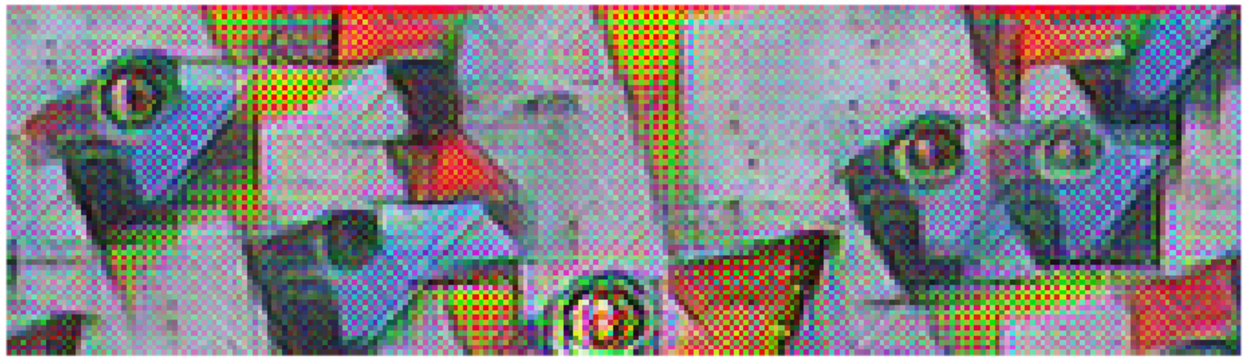
Bicubic interpolation

Resize-convolution

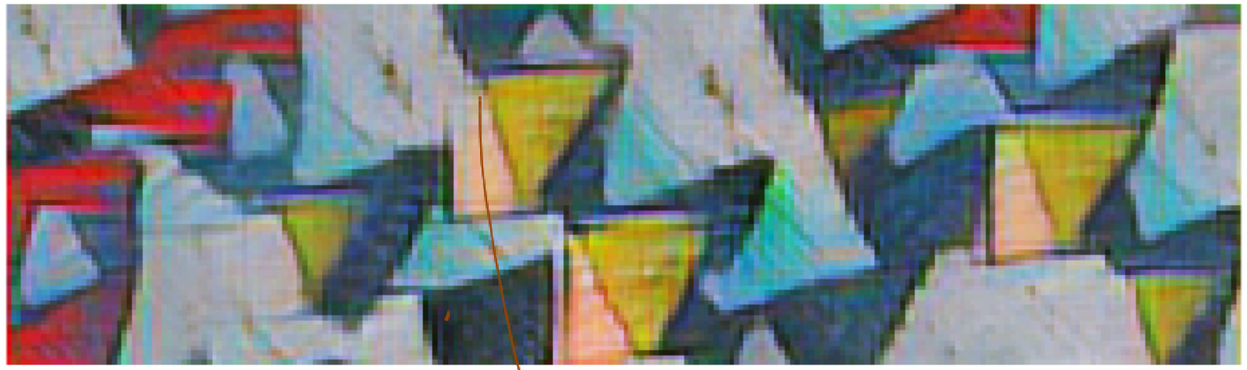
- Interpolate + convolution:

Image reconstruction example

Using transposed convolution



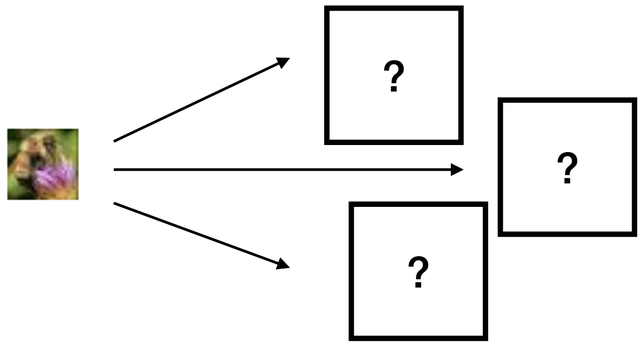
Resize-convolution



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

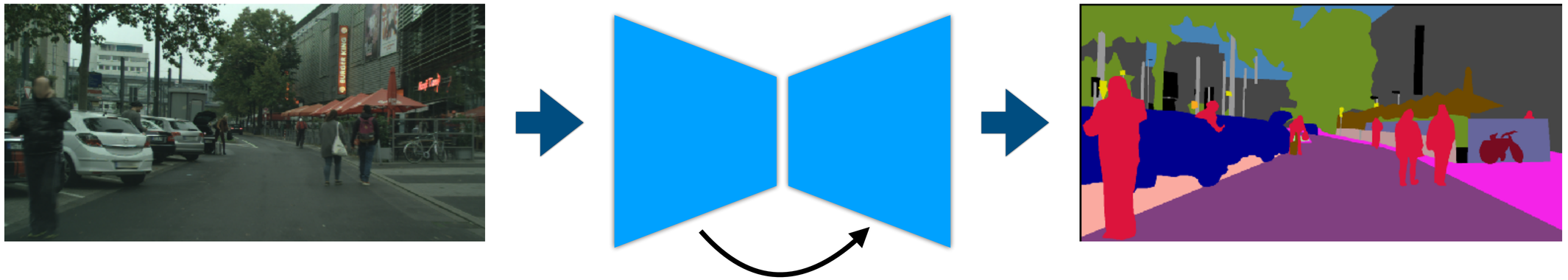
Resize-convolution

- Transposed convolution produces checkerboard artefacts
  - can be resolved by a careful choice of the kernel/stride.
- “Out-of-the-box” solution: interpolate, then convolve
- Issue: We still lost some information due to downsampling:
- In general, there are multiple plausible results of upsampling



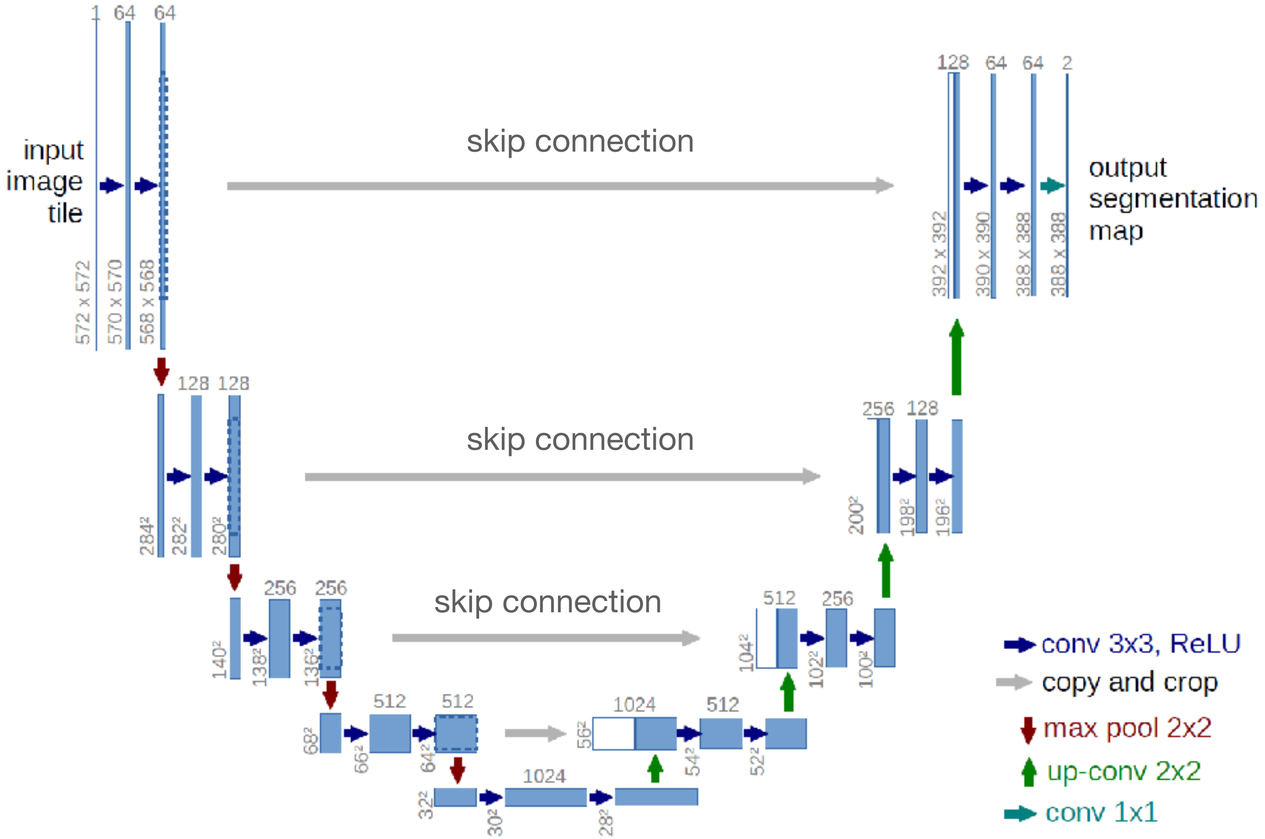
Mitigating information loss

- Where can we obtain high-resolution information?
  - first layers in the encoder via a skip connection:



Skip Connections

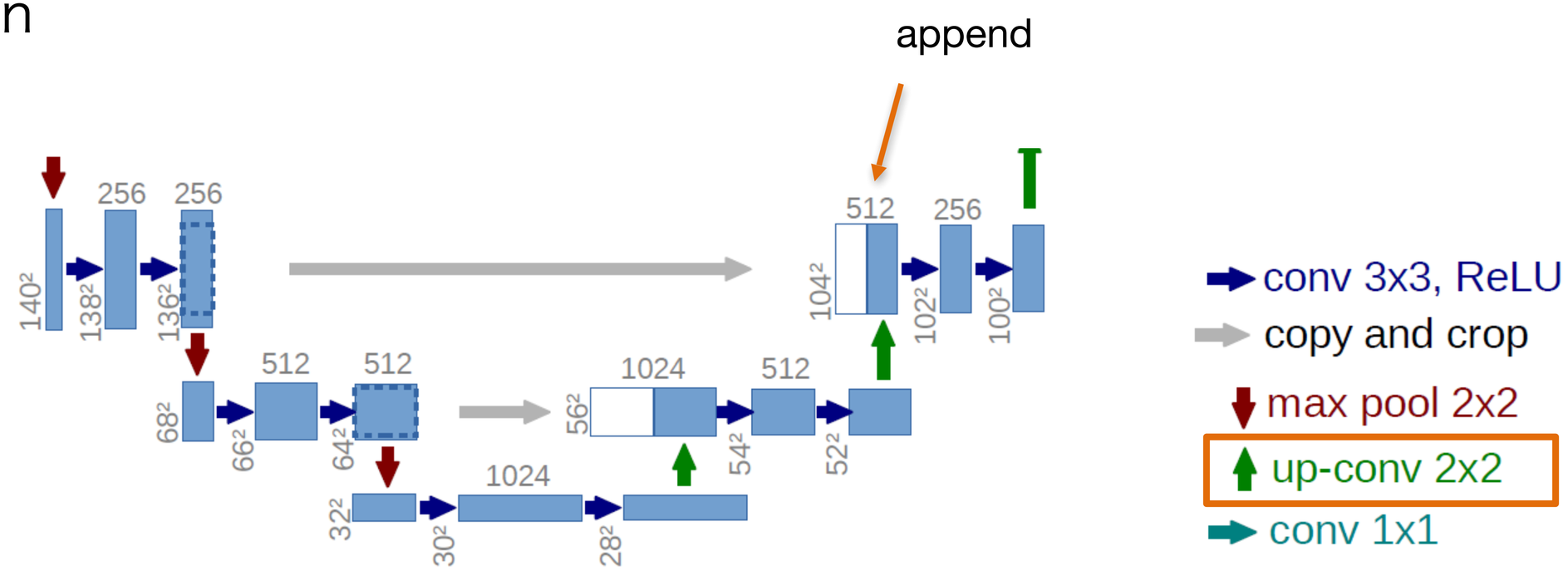
- U-Net
- QUIZ: Another example we have already seen?



O. Ronneberger et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. MICCAI 2015

Skip Connections

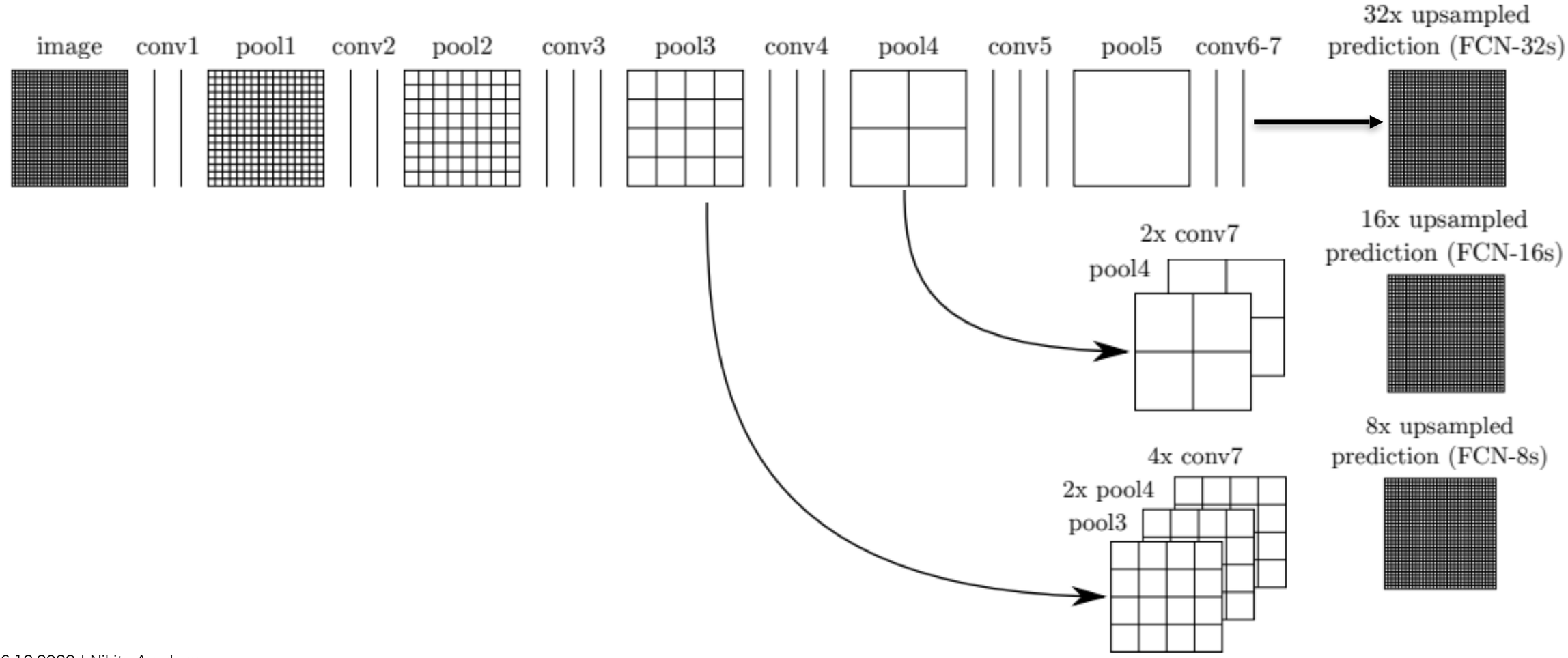
U-Net: zoom in



O. Ronneberger et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. MICCAI 2015

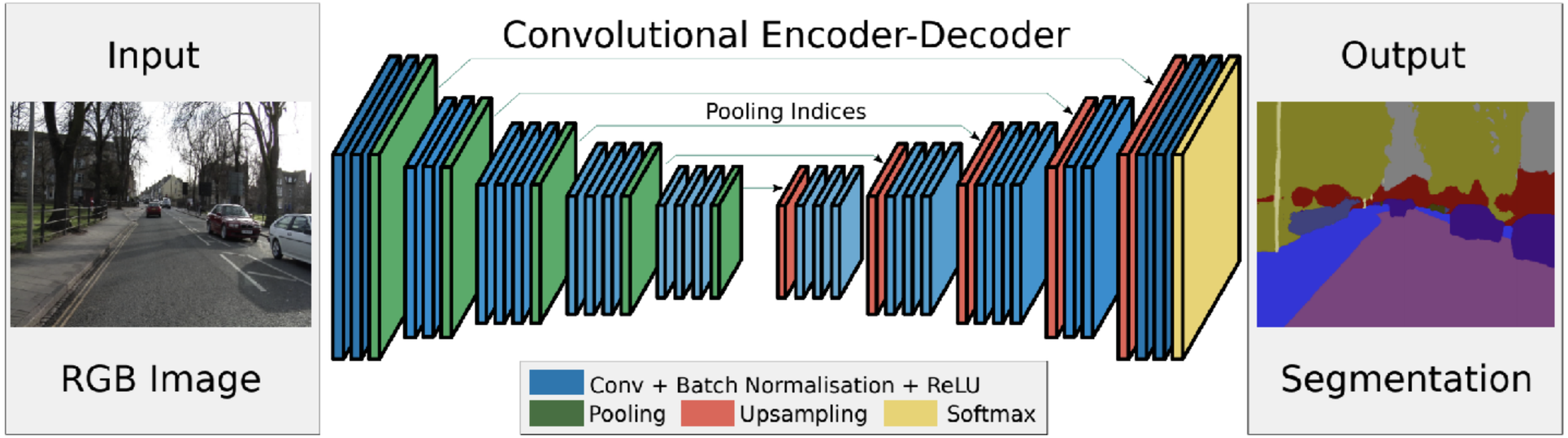
Network architecture

Similarly in FCNs, but re-use the features after pooling:



SegNet

- Recovering pooling indices: “unpooling”



Badrinarayanan et al. „SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation“. TPAMI 2016

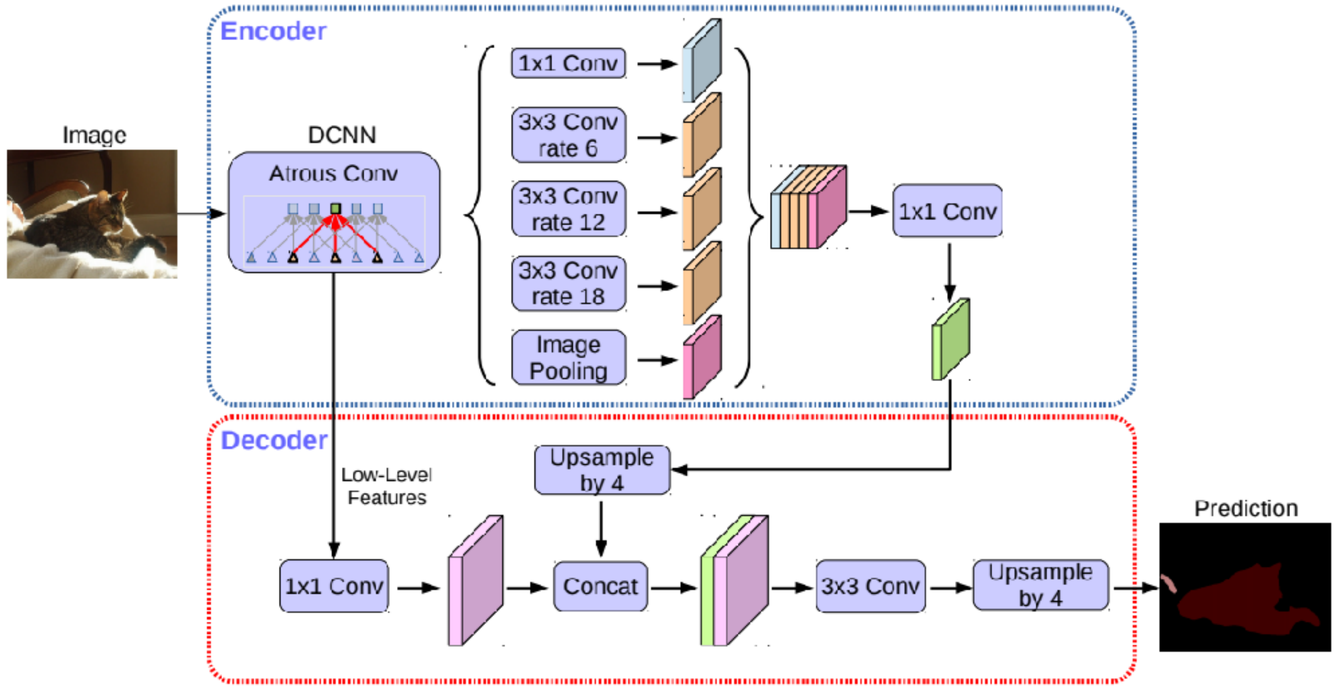
Summary

- Transposed convolutions
  - upsampling with learnable parameters; may produce artefacts.
- More practical: resize-convolution:
  - interpolate, then convolve.
- Skip connections:
  - connecting first (high-resolution) layers.
  - may require stage-wise training (train deeper layers first).

- Decoder architecture is still an active field of research.

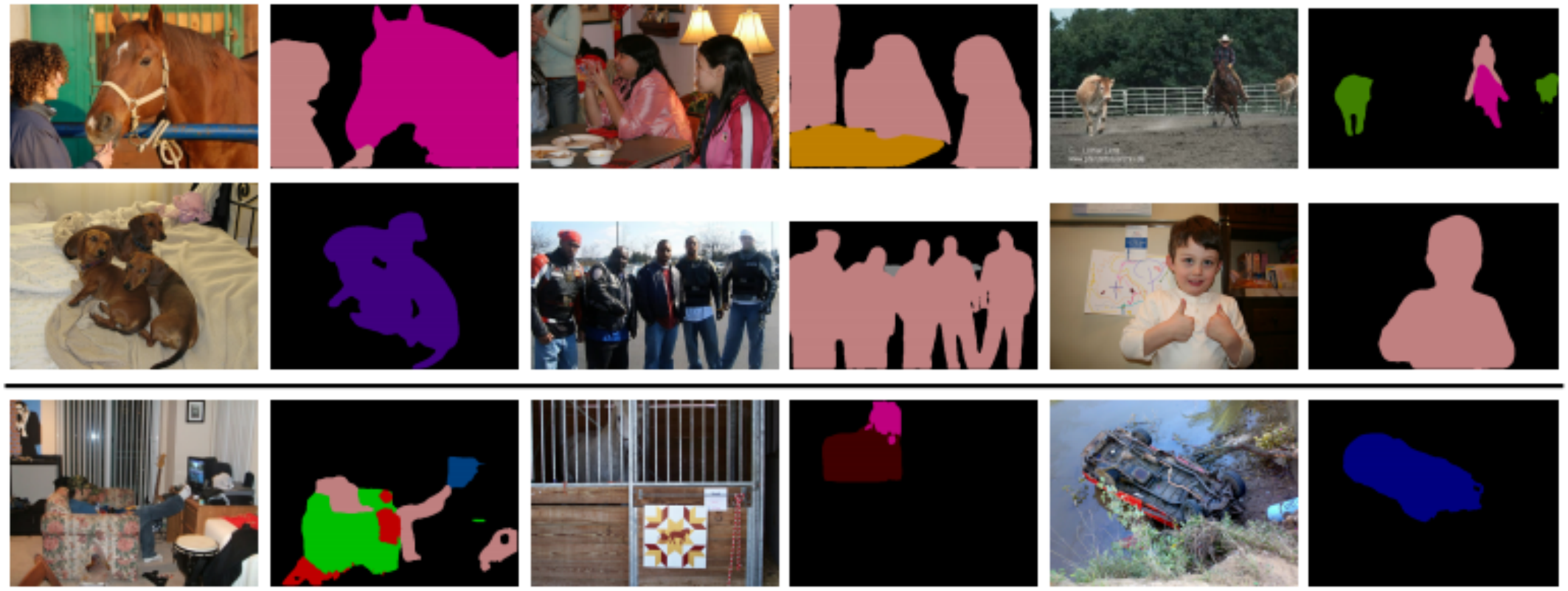
Delving deeper into DeepLabv3+

- State-of-the architectures for image segmentation can be rather complicated...



Chen et al., “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation” (2018).

DeepLabv3+: qualitative results



Still considered as SOTA!

Chen et al., “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation” (2018).

Best practice tips

- Keep feature resolution high
  - output stride 8 is typical
- Use dilated convolution to keep the receptive field size high
  - context information is important
- Use skip connections between the encoder and decoder
  - improves upsampling quality
- Use image-adaptive post-processing such as CRFs
  - improves segmentation boundaries

Additional reading

- Comaniciu et al. “Mean Shift: A Robust Approach Toward Feature Space Analysis” (2001)
- Shi and Malik “Normalized Cuts and Image Segmentation” (2000)
- von Luxburg “A tutorial on spectral clustering” (2007)
- Ronneberger et al. “U-net: Convolutional networks for biomedical image segmentation” (2015)
- Chen et al. “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs” (2016)
- Chen et al. “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation” (2018)
- Zhao et al. “Pyramid scene parsing network” (2016)
- Lin et al. “RefineNet: Multi-path refinement networks for high-resolution semantic segmentation" (2017)