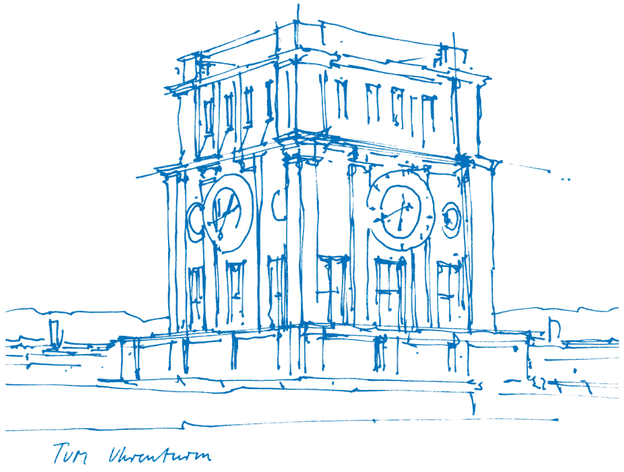
Computer Vision III:

MOT 02: Offline tracking (remainder)

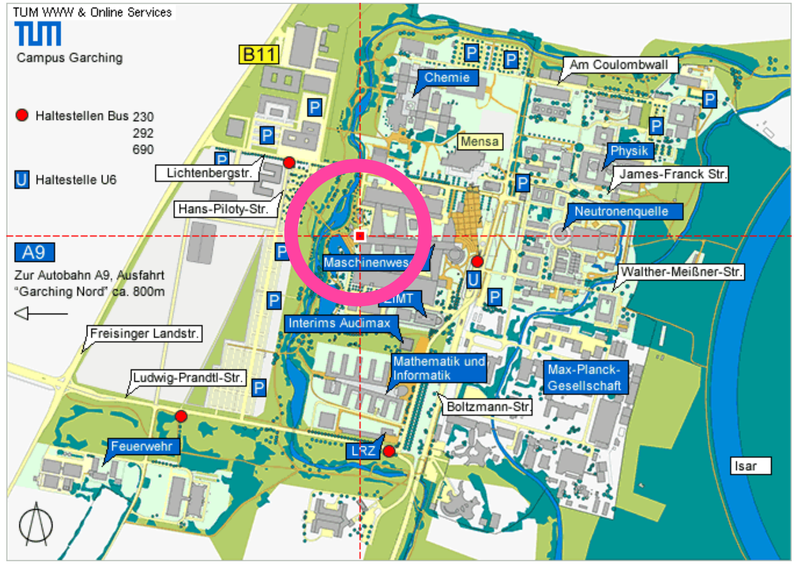
Dr. Nikita Araslanov
28.11.2023

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



Announcements

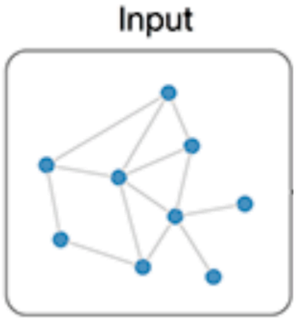
- Exam:
  - 01.03.2024, 11:00 - 12:30
  - Ernst-Schmidt-Hörsaal (5508.02.801)
  - Register until **January 15**



MOT
with Message Passing Networks

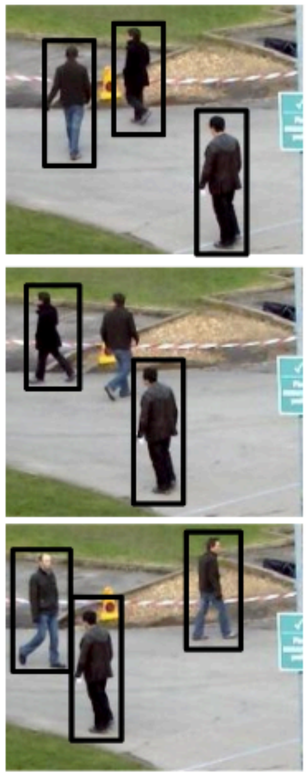
Recall our setup

- Input: task-encoding graph
  - nodes: detections encoded as feature vectors
  - edges: node interaction (e.g. inter-frame)



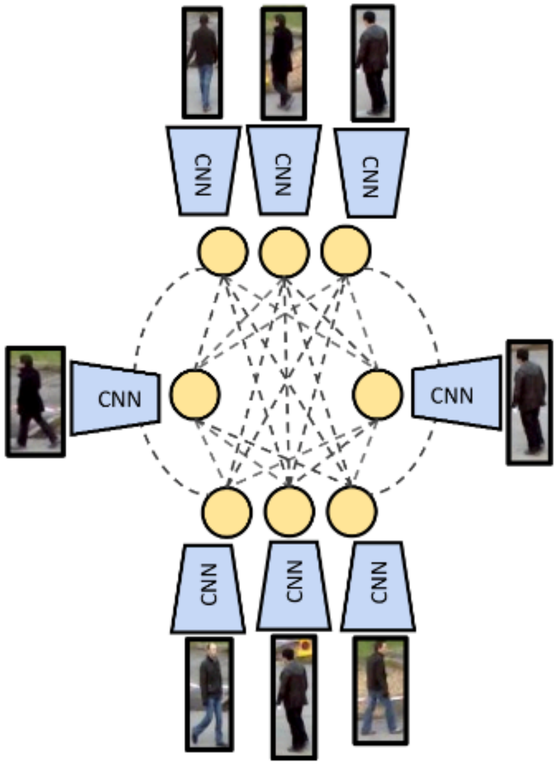
- Output: graph partitioning into (disjoint) trajectories
  - e.g. encoded by edge label (0,1)

Overview

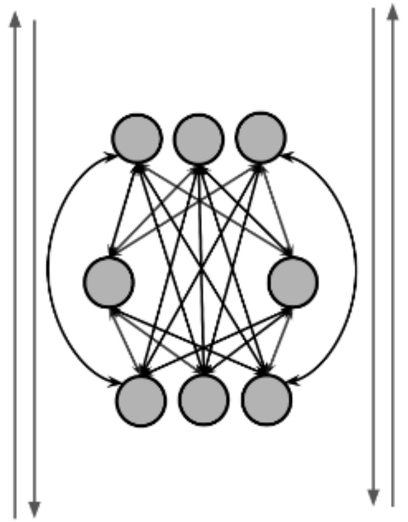


(a) Input

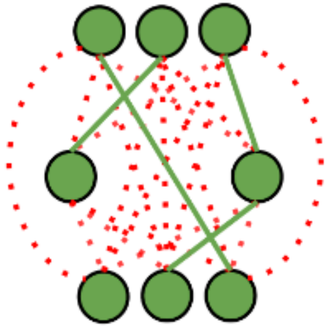
3 Frames



(b) Graph Construction + Feature Encoding



(c) Neural Message Passing



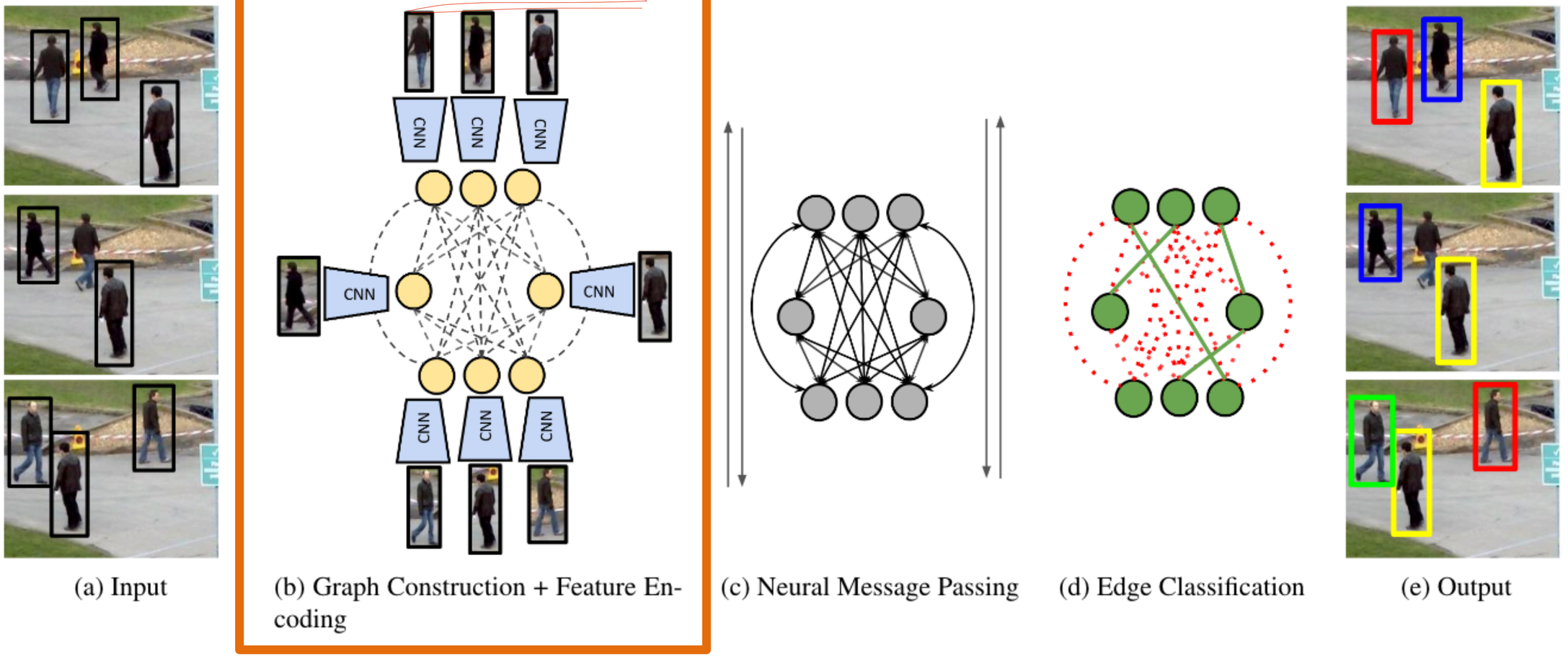
(d) Edge Classification

(e) Output

Brasó and Leal-Taixé. “Learning a Neural Solver for Multiple Object Tracking” (2019).

Overview

Encode appearance and scene geometry cues into node and edge embeddings

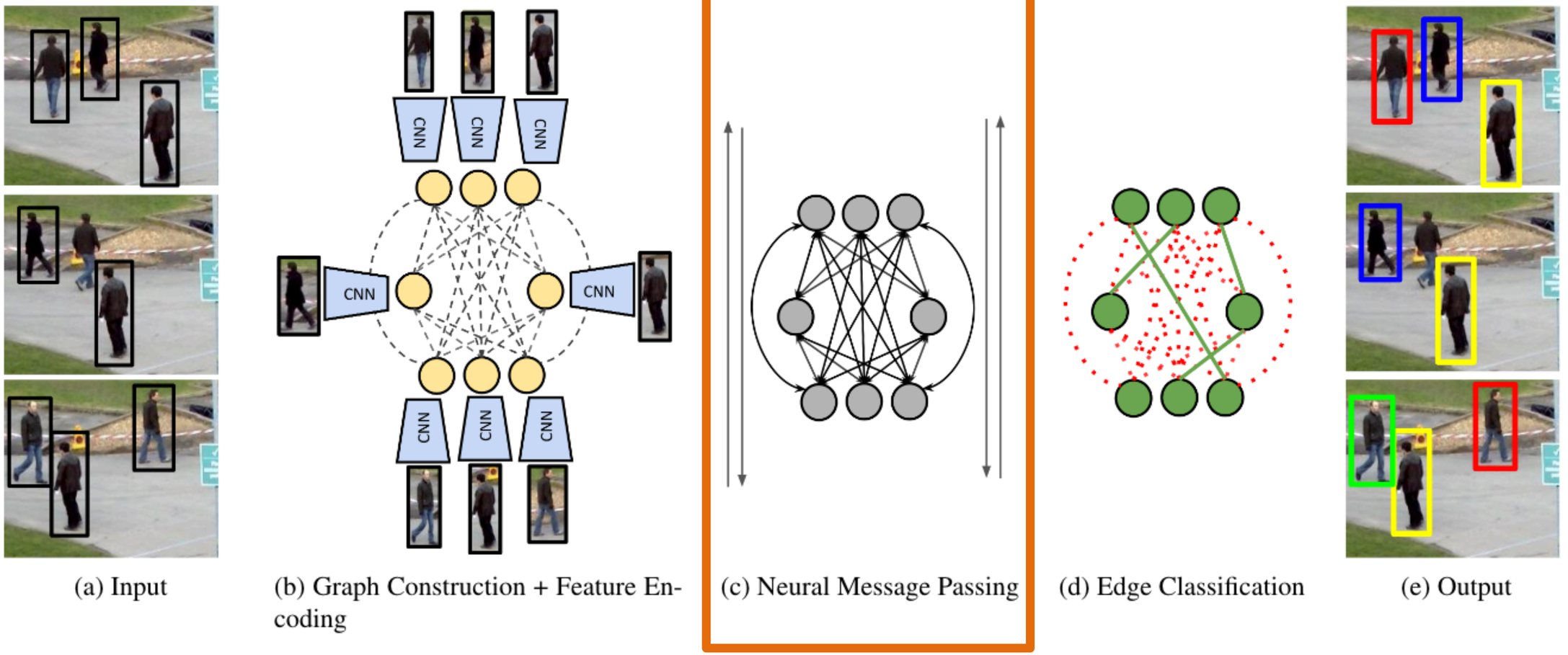


(a) Input (b) Graph Construction + Feature Encoding (c) Neural Message Passing (d) Edge Classification (e) Output

Brasó and Leal-Taixé. “Learning a Neural Solver for Multiple Object Tracking” (2019).

Overview

Propagate cues across the entire graph with neural message passing

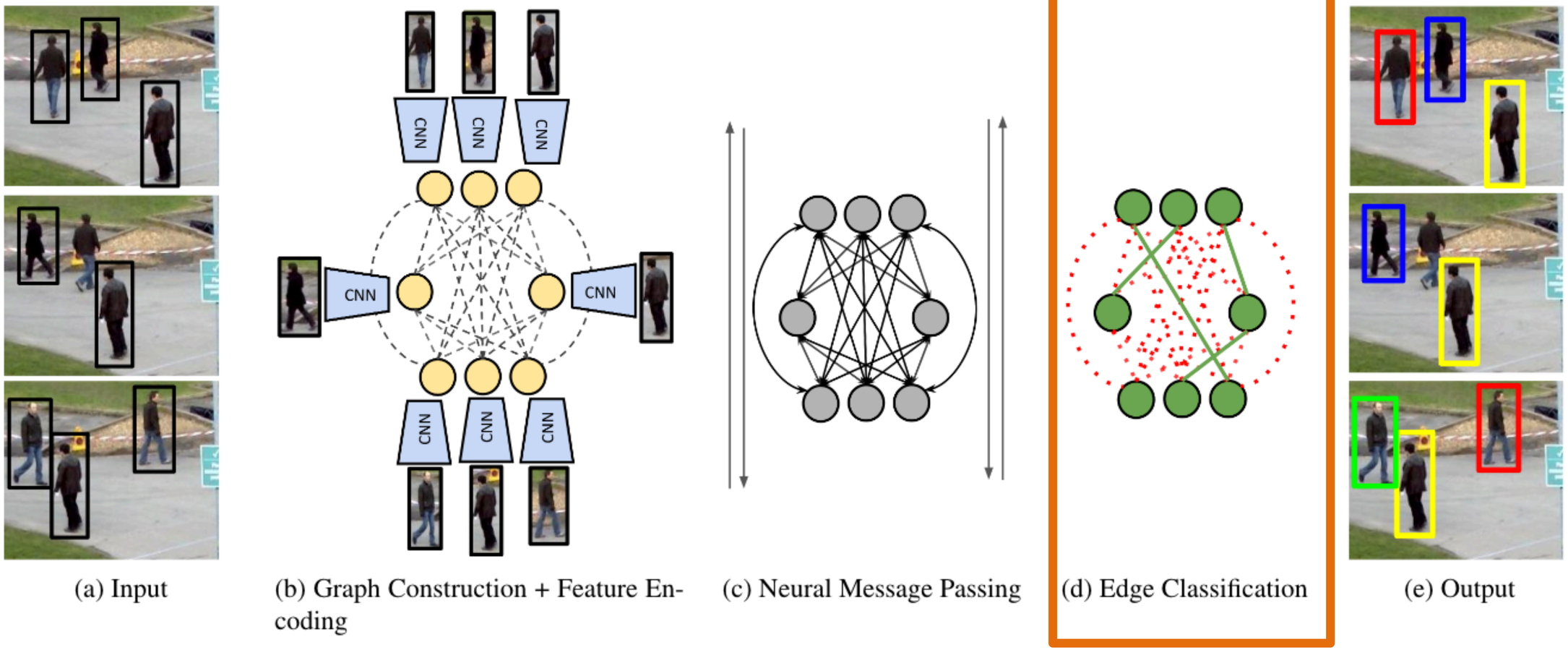


(a) Input (b) Graph Construction + Feature Encoding (c) Neural Message Passing (d) Edge Classification (e) Output

Brasó and Leal-Taixé. “Learning a Neural Solver for Multiple Object Tracking” (2019).

Overview

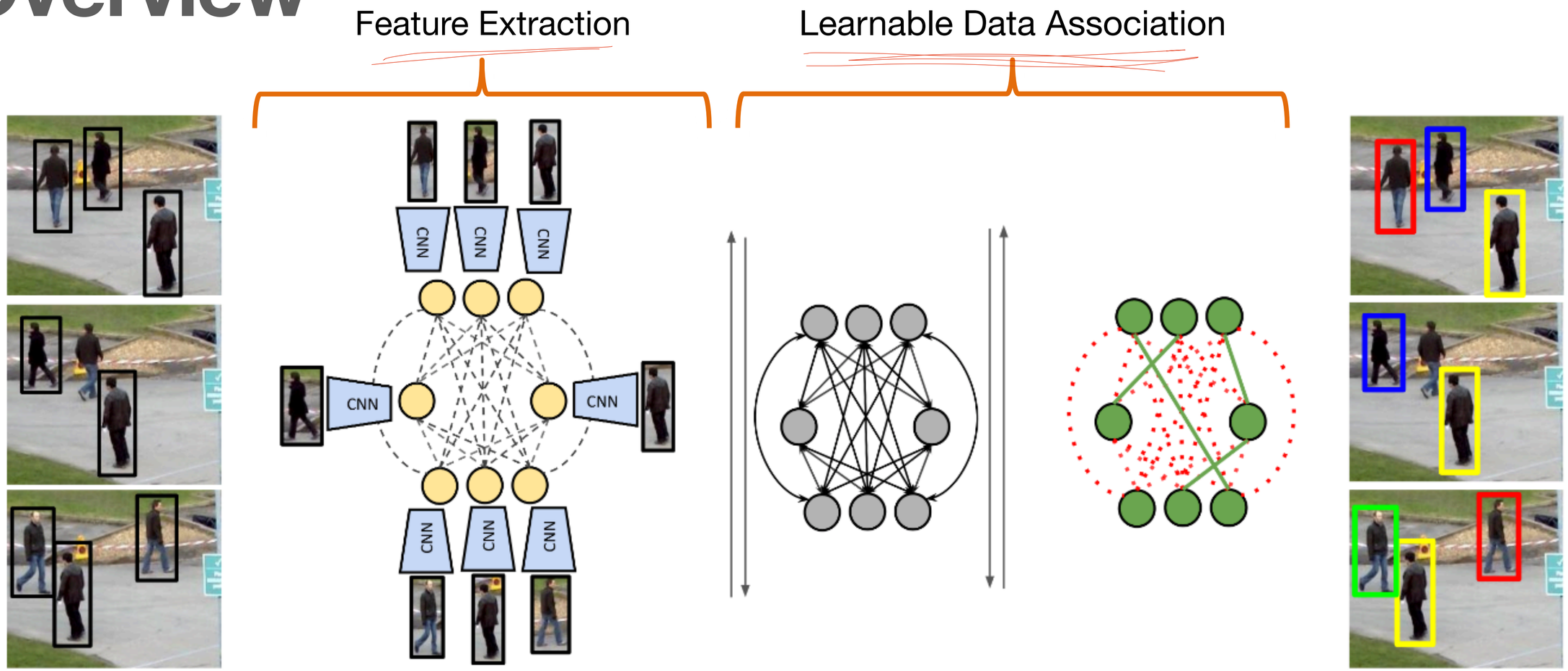
Learn to directly predict solutions of the Min-Cost Flow problem by classifying edge embeddings



(a) Input (b) Graph Construction + Feature Encoding (c) Neural Message Passing (d) Edge Classification (e) Output

Brasó and Leal-Taixé. “Learning a Neural Solver for Multiple Object Tracking” (2019).

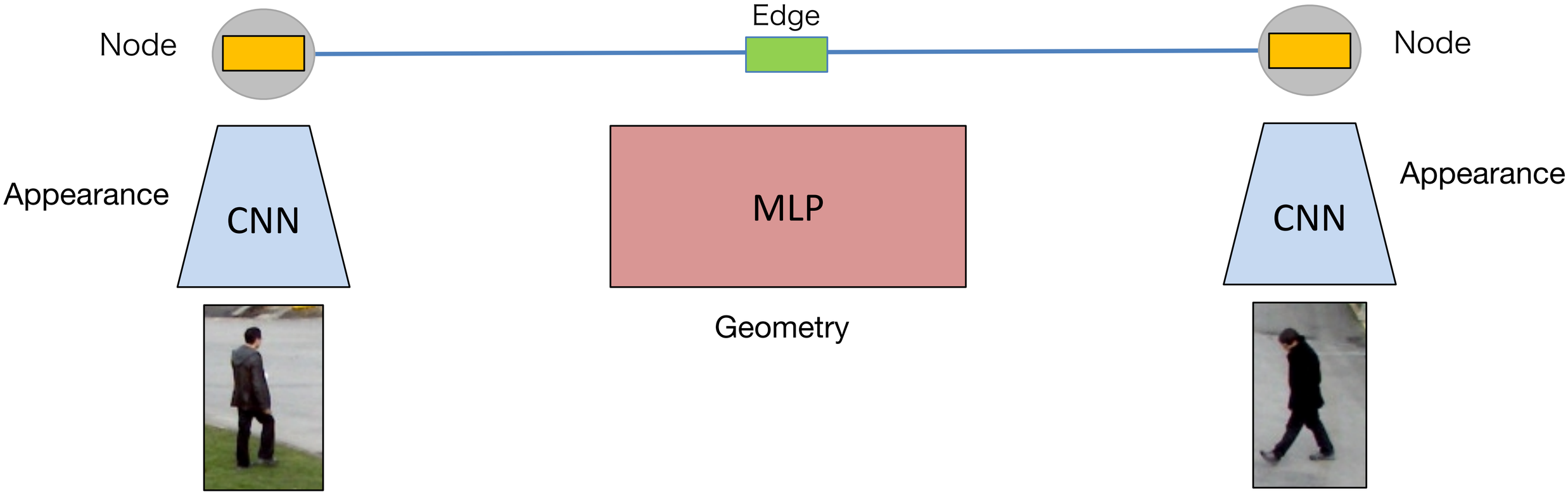
Overview



Brasó and Leal-Taixé. “Learning a Neural Solver for Multiple Object Tracking” (2019).

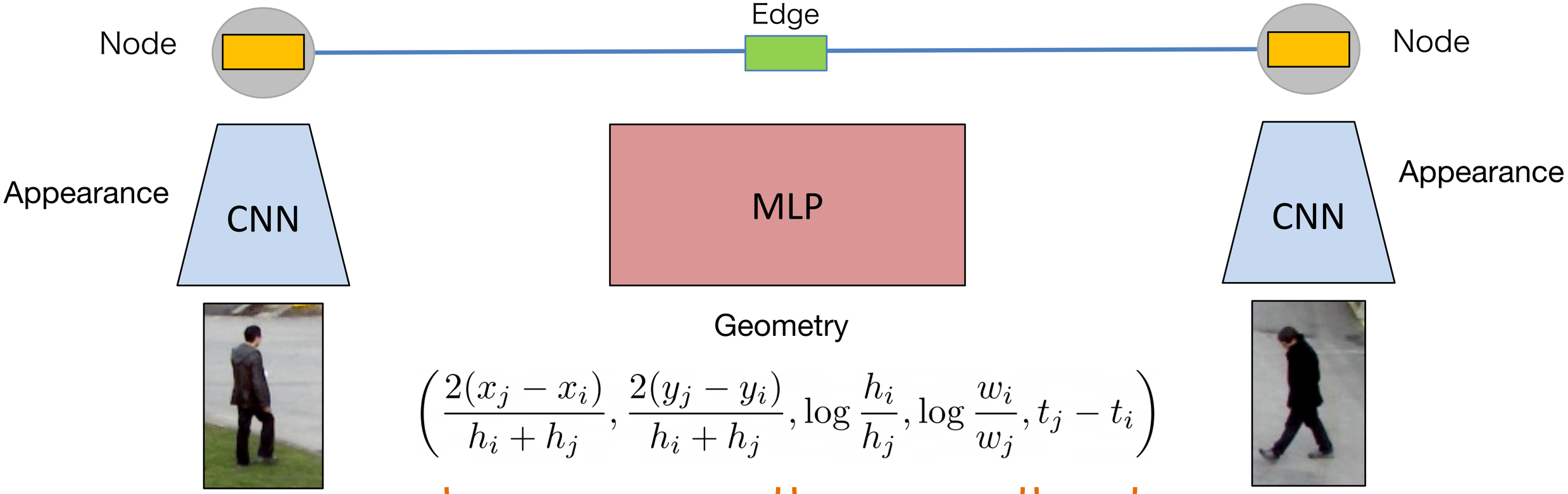
Feature encoding

- Appearance and geometry encodings



Feature encoding

- Appearance and geometry encodings



Feature encoding

- Contrast:
  - earlier: defining pairwise and unary costs
  - now:
    - feature vectors associated to nodes and edges
    - use message passing to aggregate context information into the features

Classifying edges

- After several iterations of message passing, each edge embedding contains context information about detections
- We feed the embeddings to an MLP that predicts whether an edge is active/inactive

$$\mathcal{L} = \frac{-1}{|E|} \sum_{l=l_0}^{l=L} \sum_{(i,j) \in E} w \cdot y_{(i,j)} \log(\hat{y}_{(i,j)}^{(l)}) + (1 - y_{(i,j)}) \log(1 - \hat{y}_{(i,j)}^{(l)})$$

Sum over the last steps

Weight to balance active / inactive edges

Edge predictions (w. sigmoid) at iteration l

Binary cross-entropy

Obtaining final solutions

- After classifying edges, we get a prediction between 0 and 1 for each edge in the graph.
- Directly thresholding solutions does not guarantee flow conservation constraints.
- In practice, around 98% of constraints are automatically satisfied.
- Lightweight post-processing (rounding or linear programming)
  - (see Appendix B in Brasó and Leal-Taixé (2019))
- The overall method is fast (~12 fps) and achieves SOTA in the MOT challenge by a significant margin

Summary

- No strong assumptions on the graph structure
  - handling occlusions
- Costs can be can learned from data
- Accurate and fast (for an offline tracker).
- (Almost) End-to-end learning approach
  - some post-processing required

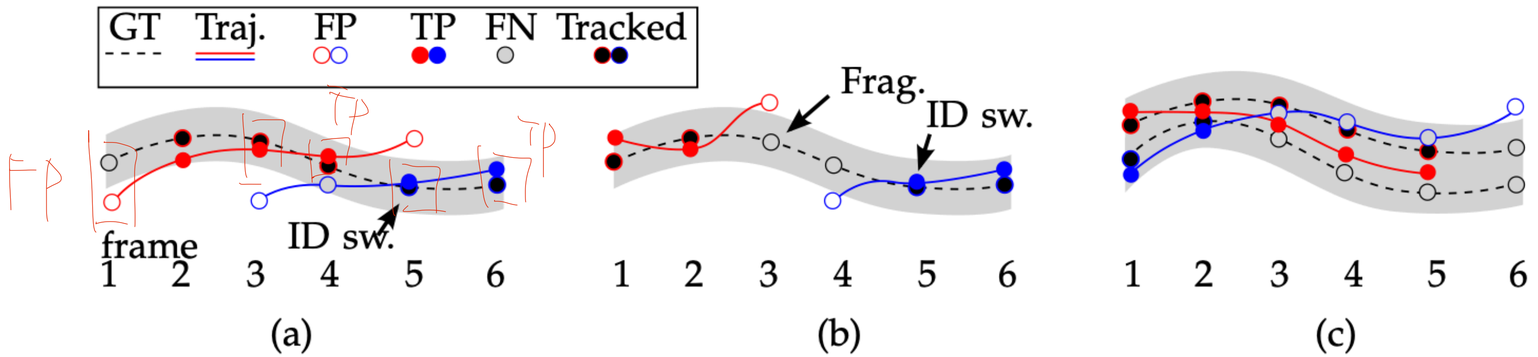
MOT evaluation

Evaluation metrics

- Compute a set of measures per frame
  - Perform matching between predictions and ground truth
    - Hungarian algorithm
  - FP = false positives
  - FN = false negatives
  - IDSW: “identity switches”

Evaluation metrics

- How do we compute ID switches?



(a) An ID switch is counted because the ground truth track is assigned first to red, then to blue.

(b) Count both an ID switch (red and blue both assigned to the same ground truth), but also a fragmentation (Frag.) because the ground truth coverage was cut.

(c) Identity is preserved. If two trajectories overlap with a ground truth trajectory (within a threshold), the one that forces least ID switches is chosen (the red one).

Leal-Taixé et al.. MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking (2015).

Evaluation metrics

- Compute a set of measures per frame
  - Perform matching between predictions and ground truth
    - Hungarian algorithm
  - FP = False positives
  - FN = False negatives (missing detections)
  - IDSW: “identity switches”

“Accuracy”?

Multi-object tracking accuracy

$$\mathrm{MOTA} = 1 - \frac{\sum_t (\mathrm{FN}_t + \mathrm{FP}_t + \mathrm{IDSW}_t)}{\sum_t \mathrm{GT}_t},$$

Ground truth

Evaluation metrics

- Multi-object tracking accuracy (MOTA):

$$\mathrm{MOTA} = 1 - \frac{\sum_t (\mathrm{FN}_t + \mathrm{FP}_t + \mathrm{IDSW}_t)}{\sum_t \mathrm{GT}_t},$$

Ground truth

- F_1-Score: $\mathrm{IDF1} = \frac{2\sum_t \mathrm{TP}_t}{\sum_t 2\mathrm{TP}_t + \mathrm{FP}_t + \mathrm{FN}_t}$

- Multi-object tracking precision (MOTP): $\mathrm{MOTP} = \frac{\sum_{t,i} \mathrm{IoU}_{t,i}}{\sum_t \mathrm{TP}}$

Datasets

- MOTChallenge: www.motchallenge.net (people)
- KITTI benchmark: http://www.cvlibs.net/datasets/kitti/ (vehicles)
- UA-Detrac: http://detrac-db.rit.albany.edu (vehicles)

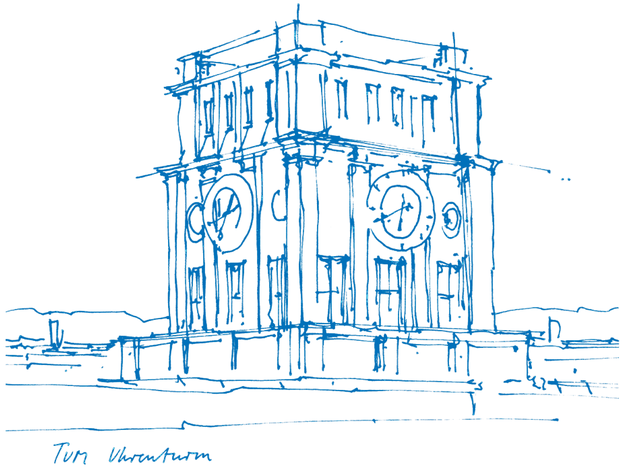
- Each benchmark may have a unique set challenges
- Unbiased view:
  - high/competitive accuracy on all datasets is desirable

Computer Vision III:

Image segmentation 1

Dr. Nikita Araslanov
28.11.2023

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



Task definition

Label each pixel:

Segmentation method
(e.g. CNN)

?

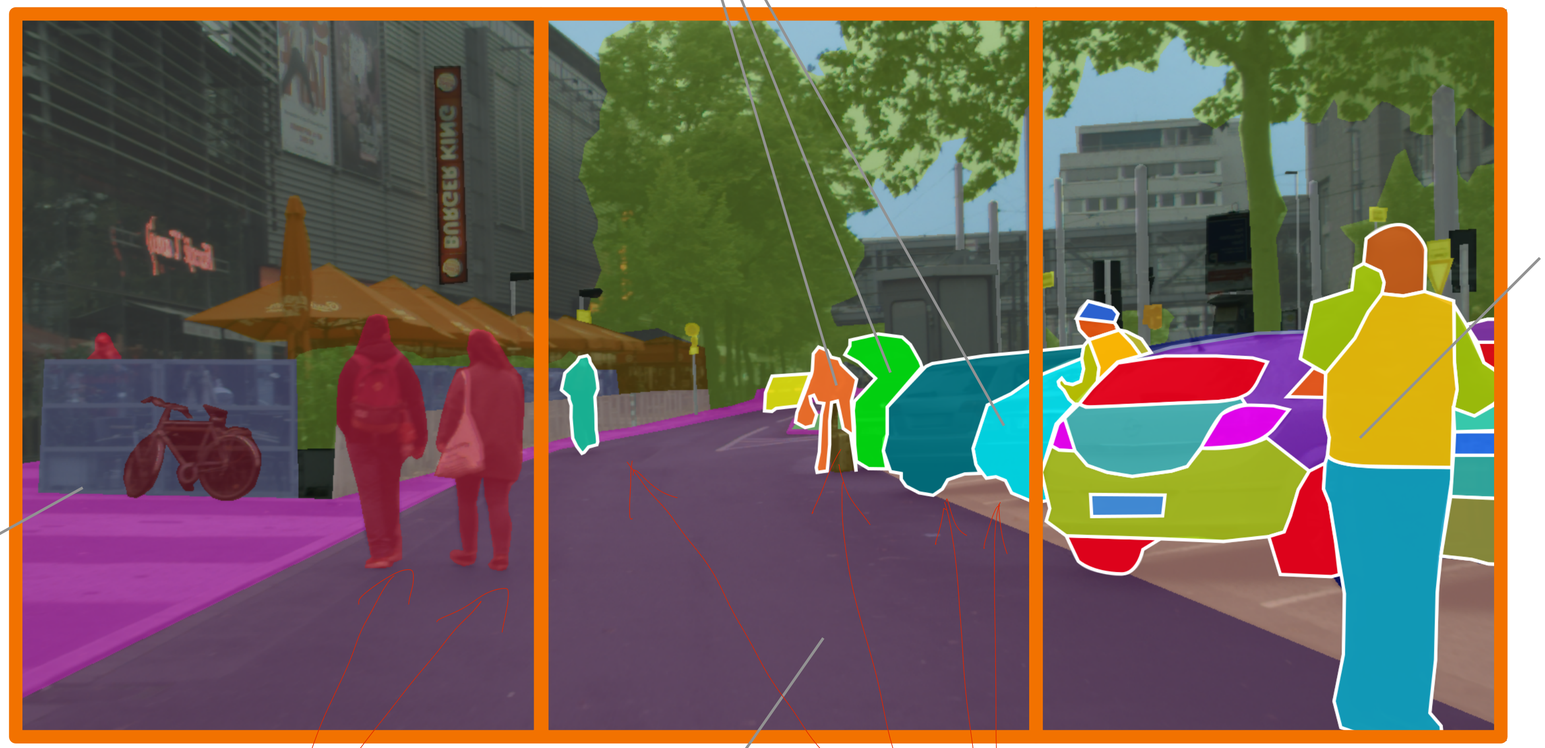
Flavours of image segmentation

- **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”)



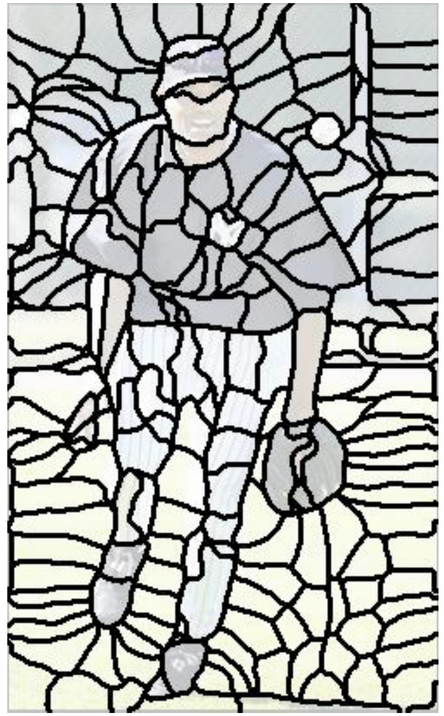
Part segmentation
(typically one object)

Semantic segmentation

Panoptic segmentation

Superpixels

- Local class-agnostic pixel grouping:



[Mori et al., 2004; Stefan Roth]

Image segmentation with...

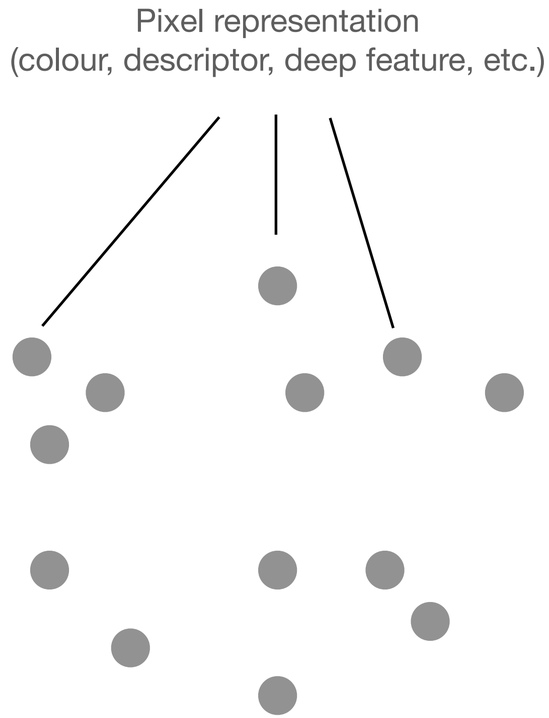
- Clustering, e.g.
  - K-means;
  - mean shift (Comaniciu and Meer, 2002);
  - spectral clustering, etc.
- Normalised cuts (Ncut).
- Energy-based models
  - Conditional Random Fields (CRFs).
- Deep learning
  - Fully convolutional networks.

Image segmentation with...

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

Segmentation as clustering

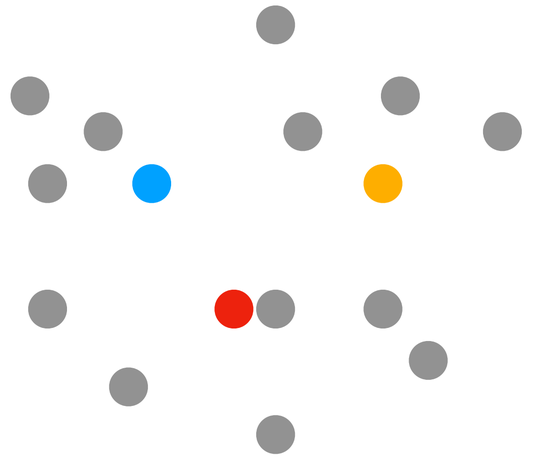
- K-means



Segmentation as clustering

- K-means

1. Initialise K cluster centres

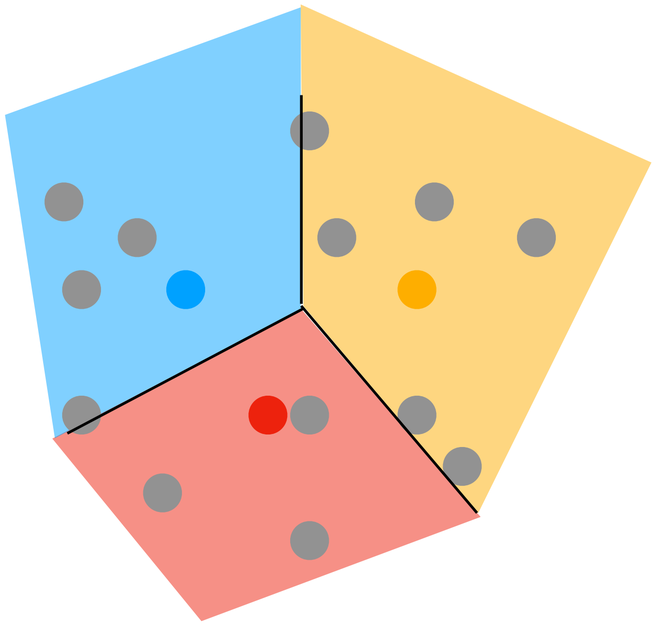


Segmentation as clustering

- K-means

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

   using a distance metric



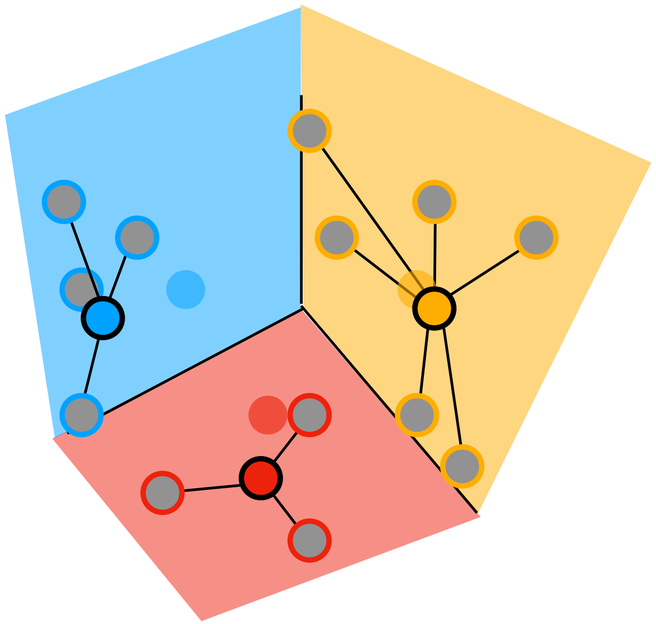
Segmentation as clustering

- 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

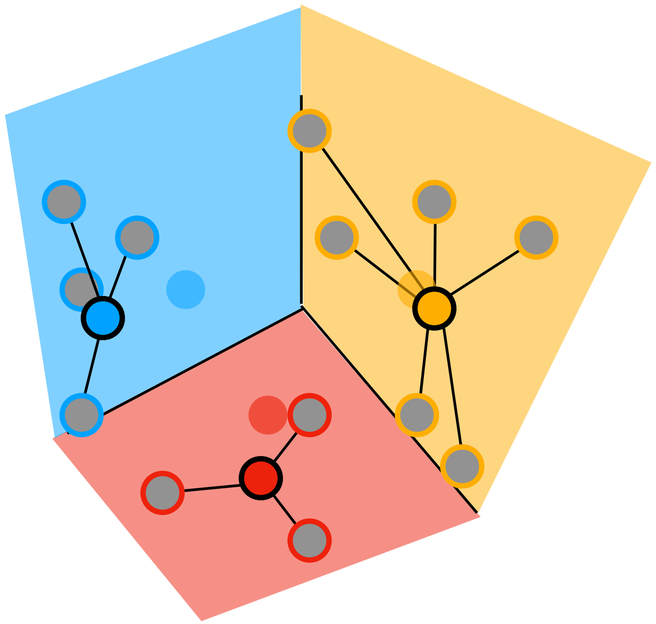


Segmentation as clustering

- 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

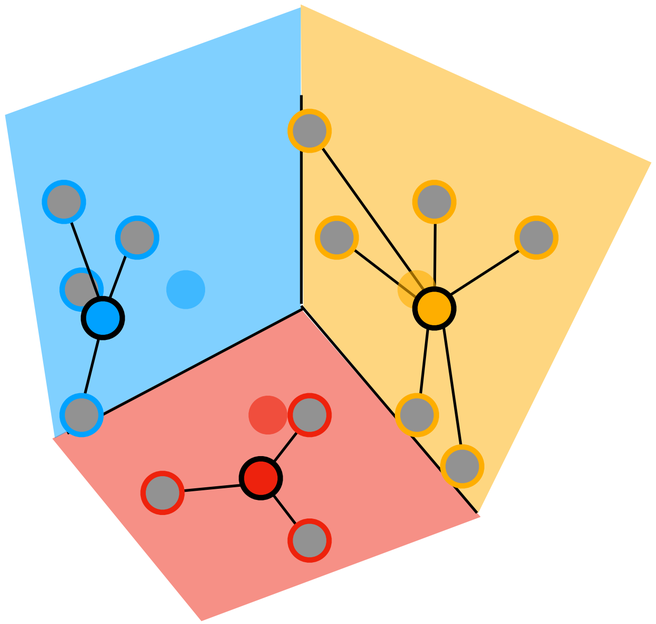


Segmentation as clustering

- K-means

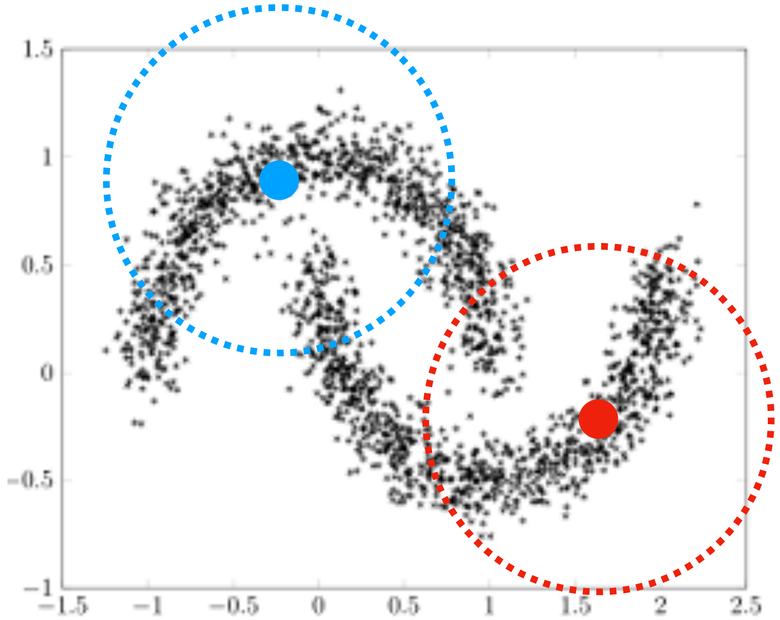
1. Initialise (randomly) K cluster centres
2. Assign points to clusters

   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?

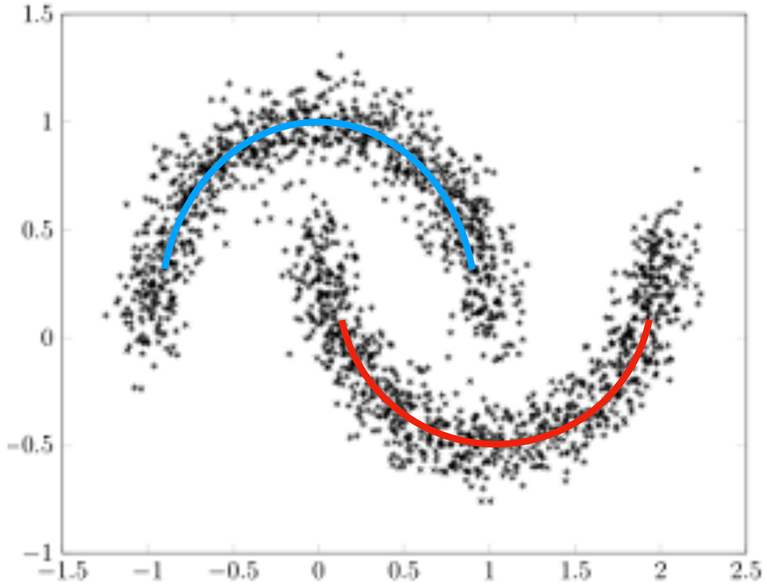


[Mathworks]

- Euclidean distance here may not be as useful here.

Segmentation as clustering

- K-means
  - does it always work?



[Mathworks]

Connected components

- Euclidean distance here may not be as useful here.
- Instead, we want to cluster based on the notion of connectedness.

Spectral clustering

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

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

- $a_{i,j} \geq 0$ 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

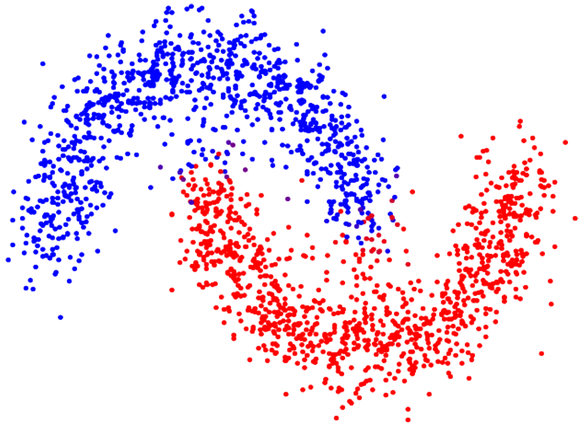
- Intuitively, if x is an eigenvector with 0 eigenvalue, then

$$x^T L x = \frac{1}{2} \sum_{i,j} A_{i,j} (x_i - x_j)^2 = 0$$

- This implies that
  - if $x_i \neq x_j$ (if i and j are in different clusters), then $A_{i,j} = 0$
  - if $A_{i,j} > 0$ (i and j are similar), then $x_i = x_j$ (same cluster)
- In practice, it's all approximate. Spectral clustering finds a global solution satisfying most of the conditions above.

Spectral clustering

Example



[Mathworks]

- 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

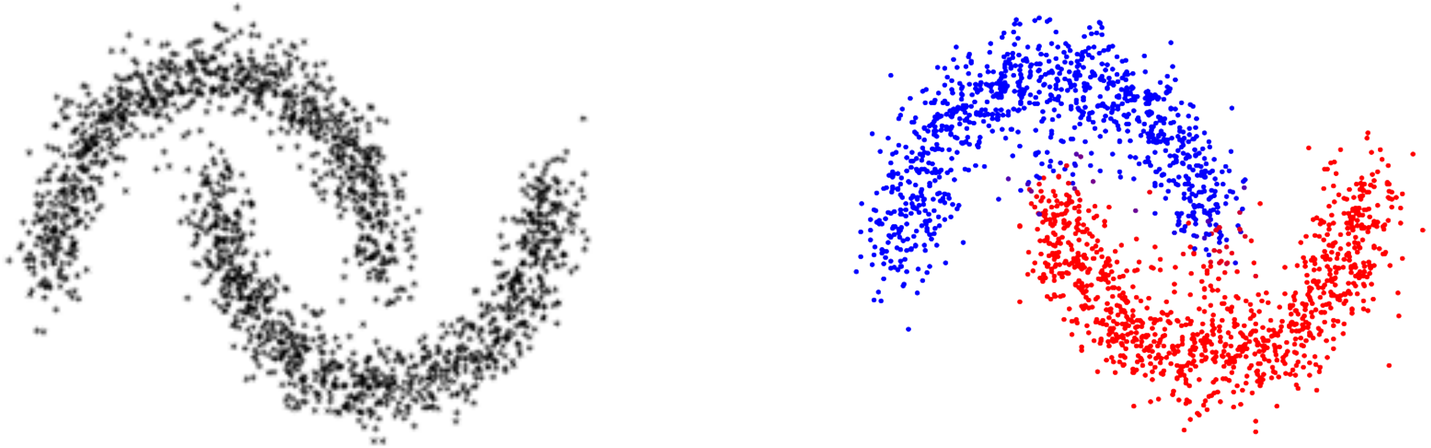
In nutshell:

- **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



[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...

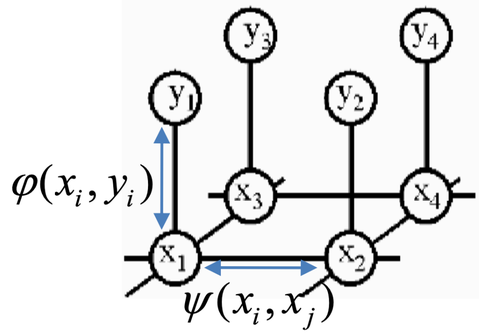
- Clustering, e.g.
  - K-means;
  - mean shift (Comaniciu and Meer, 2002);
  - spectral clustering, etc.
- Normalised cuts (Ncut)
- Energy-based models
  - Conditional Random Fields (CRFs).
- Deep learning
  - Fully convolutional networks.

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

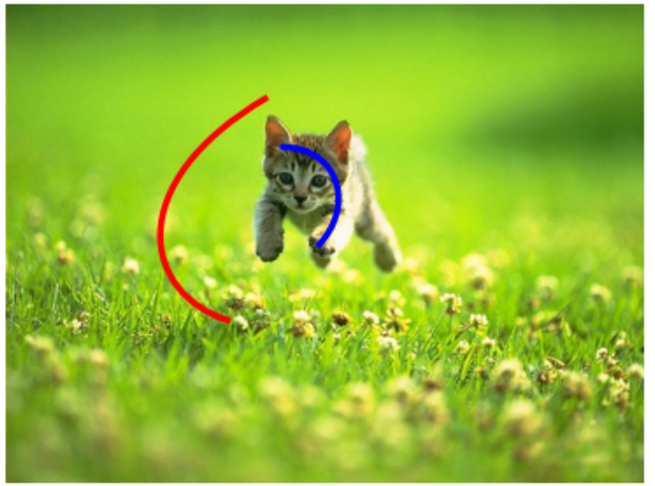
Slide credit: Bastian Leibe

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

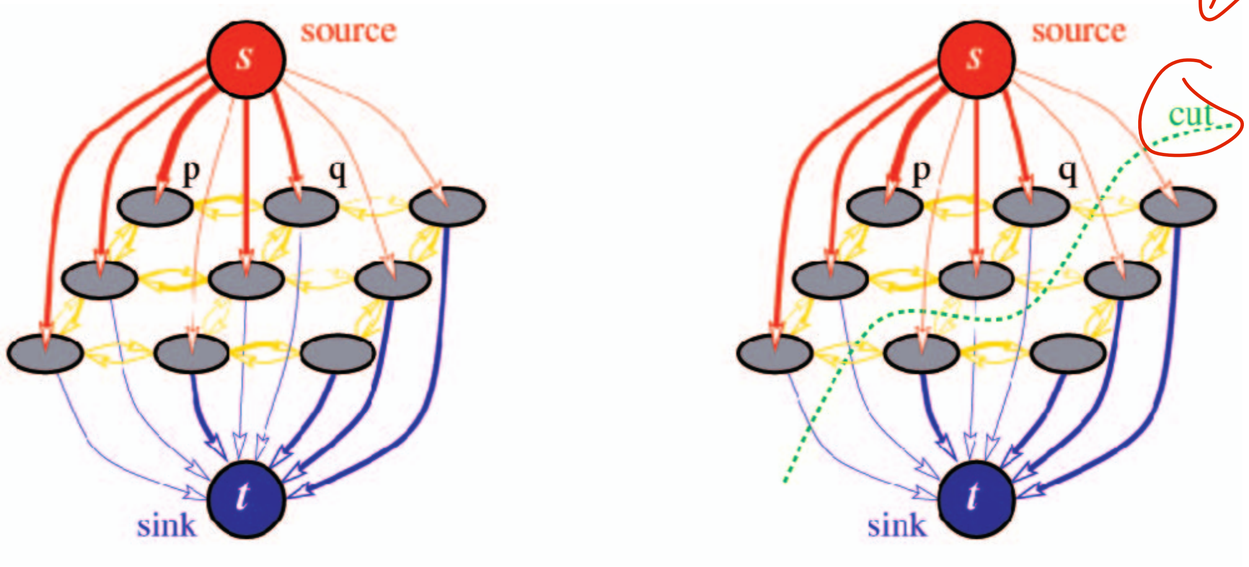


Slide credit: Philipp Krahenbuhl

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

Energy Minimisation

- Optimisation with graph cuts:



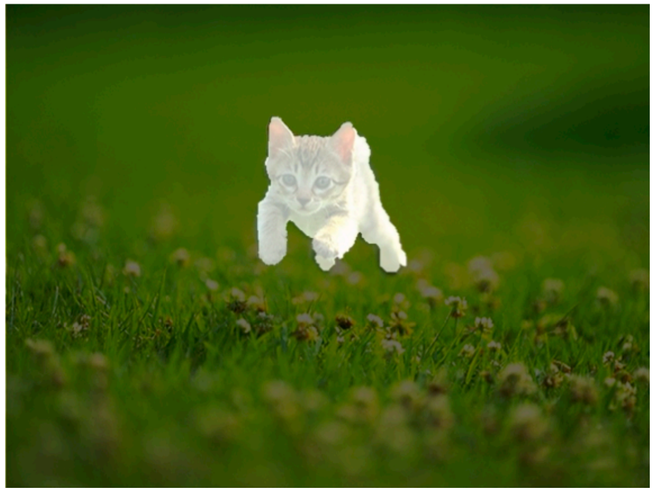
Find

Max-flow min-cut theorem:
The maximum value of an s-t flow is equal to the minimum capacity over all s-t cuts.

Boykov and Kolmogorov. An Experimental Comparison of Min-Cut/ Max-Flow Algorithms for Energy Minimization in Vision. T-PAMI 2004

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

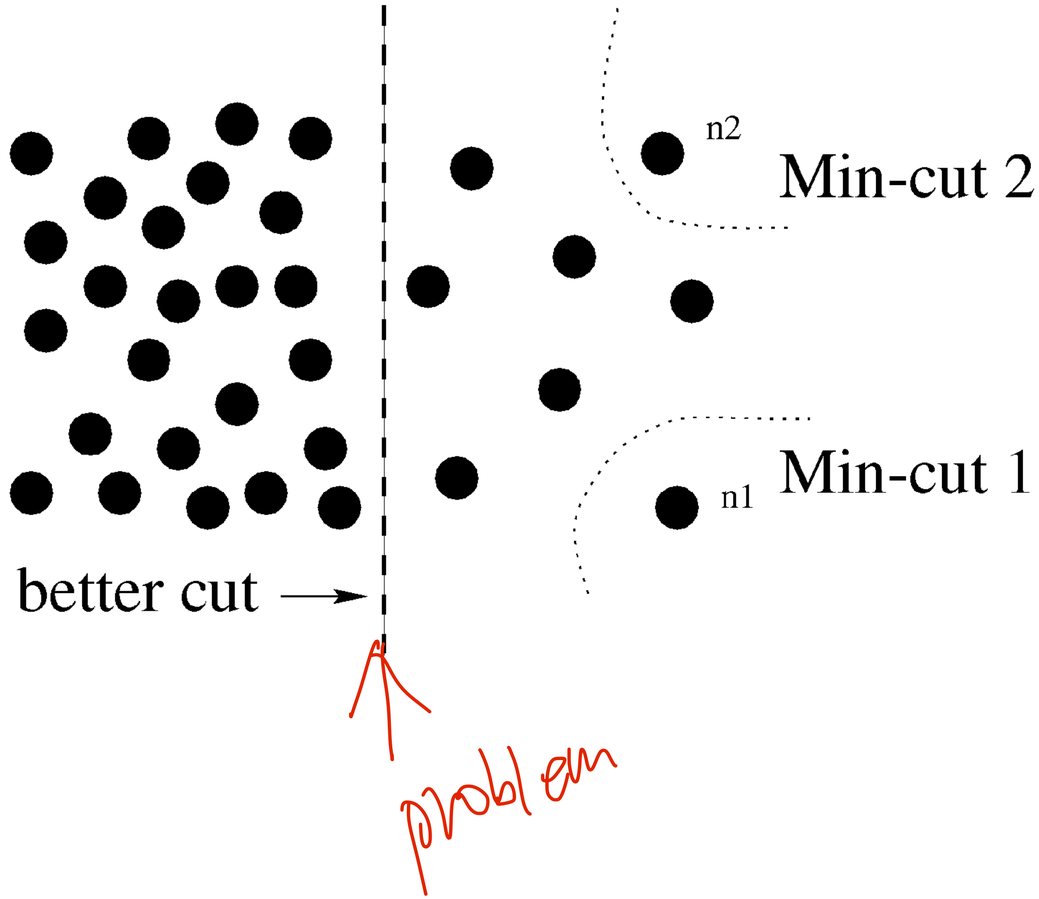
Boykov and Jolly “Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images” (2001).

Image segmentation with...

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

Normalised cuts

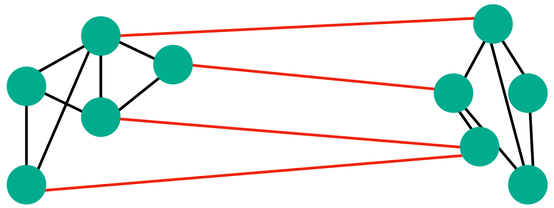
- Problem with min cuts – favour unbalanced isolated clusters:



[Shi and Malik, 2001; Shi, 2009]

Normalised cuts

- Balanced cut



$$\mathrm{Ncut}(A, B) = \mathrm{cut}(A, B)(\mathrm{vol}^{-1}(A) + \mathrm{vol}^{-1}(B))$$

The cost of cutting sets A and B:

$$\mathrm{cut}(A, B) = \sum_{i \in A, j \in B} w_{i,j}$$

Total edge cost in set A/B:

$$\mathrm{vol}(V) = \sum_{i,j \in V} w_{i,j}$$

- Intuitively – **minimise** the similarity between groups A and B, while **maximising** the similarity within each group.

Adapted from [Shi. 2009]

Ncut

- Define a graph $G := (V, E)$
  - V is the set of nodes representing pixels
  - E defines similarity of two nodes
- Solve a generalised eigenvalue problem $(D - W)x = \lambda Dx$

The Laplacian matrix

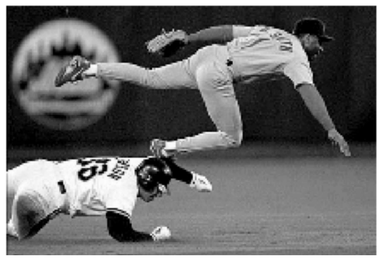
$$d(i) = \sum_j w_{ij}$$

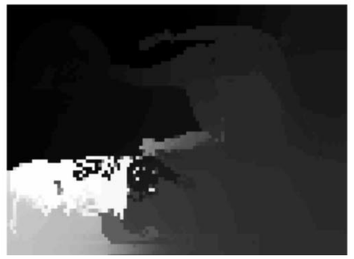
- Use the eigenvector with the 2nd smallest eigenvalue to cut the graph
- Recurse if needed (i.e. subdivide the two node groups).

[Shi and Malik, 2001; Shi, 2009]

Ncut

Eigenvectors





2

3

4

5

6

7

8

9

[Shi and Malik, 2001; Shi, 2009]

Ncut

- Define a graph $G := (V, E)$
  - V is the set of nodes representing pixels
  - E defines similarity of two nodes
- Solve a generalised eigenvalue problem $(D - W)x = \lambda Dx$

The Laplacian matrix

$$d(i) = \sum_j w_{ij}$$

- Use the eigenvector with the 2nd smallest eigenvalue to cut the graph
- Recurse if needed (i.e. subdivide the two node groups).

Wait, why second?!

[Shi and Malik, 2001; Shi, 2009]

Ncut

- Why second smallest eigenvalue/eigenvector?
- We're solving a relaxation of the following problem:

$$\min_y \frac{y^T(D-W)y}{y^T D y}$$ – Rayleigh quotient

- The smallest eigenvalue solution of $(D-W)y = \lambda D y$

is indeed the minimum, but ...

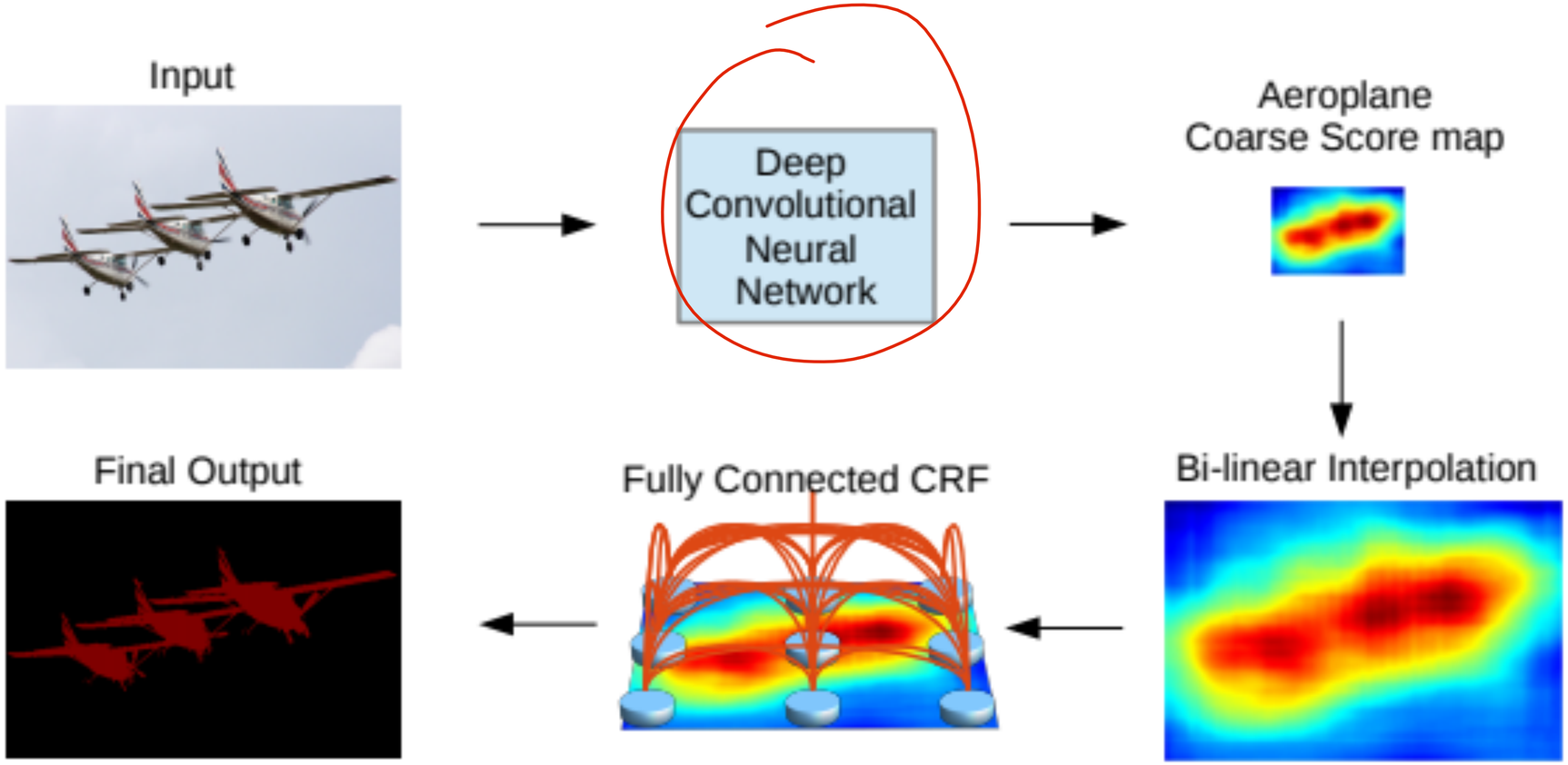
- (QUIZ): Is it an interesting one? (Hint: Find non-zero y making LHS zero)
- More details: Sec. 2.1 in [Shi and Malik, 2000].

[Shi and Malik, 2000; Shi, 2009]

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.

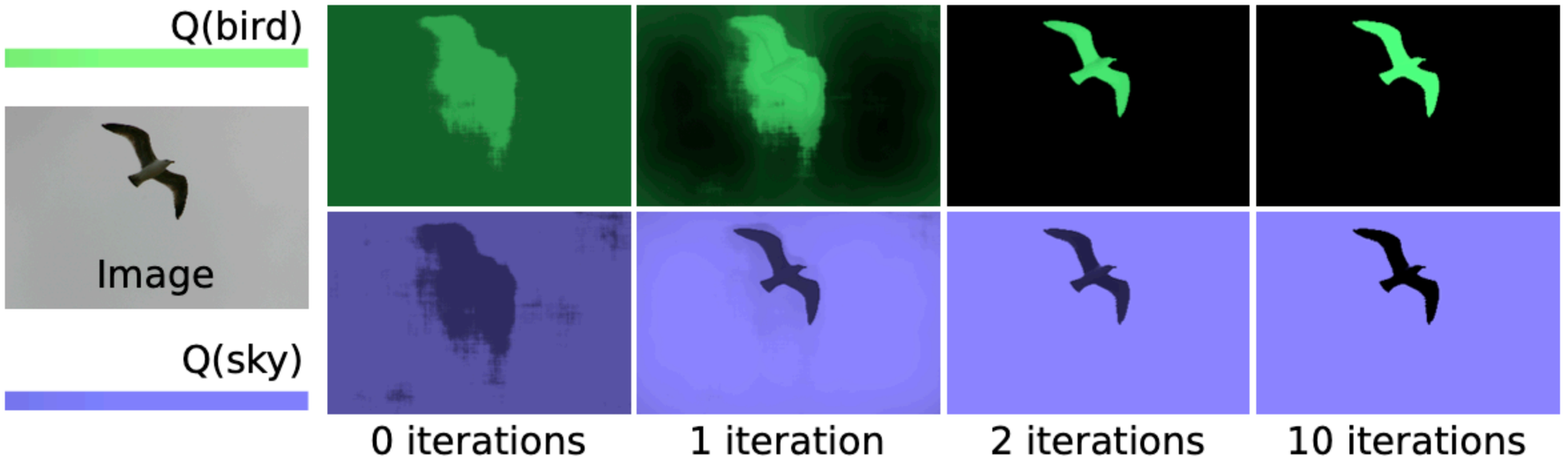
DeepLab



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)

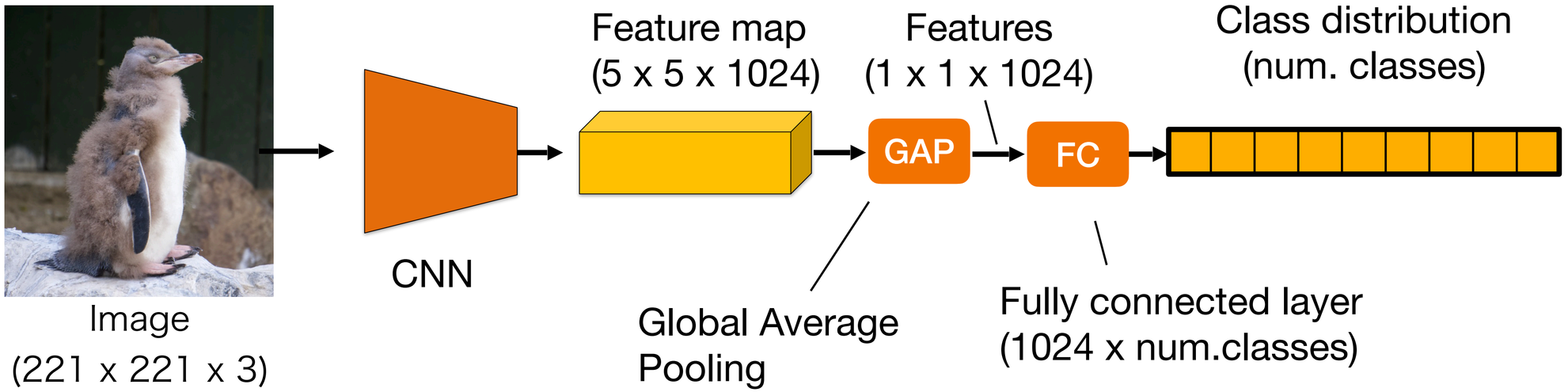
Krähenbühl & Koltun, 2011

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.

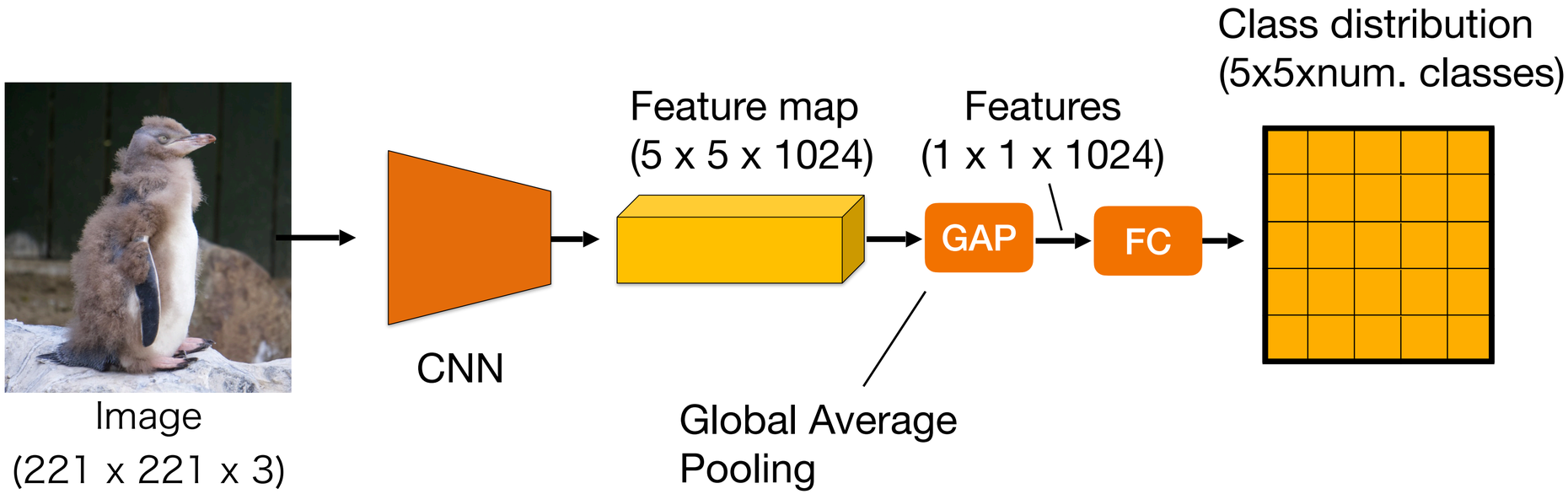
Deep networks for classification

- Recall deep networks for classification



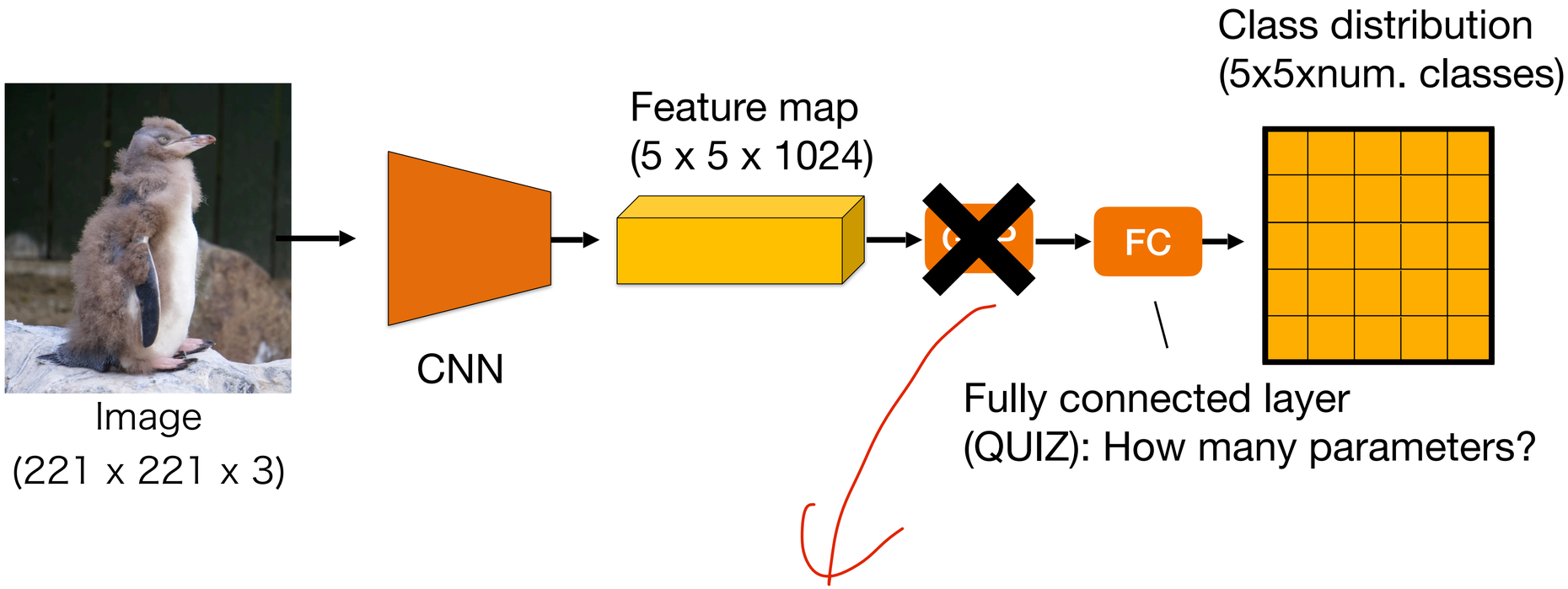
Deep networks for classification

- How can we extend this to segmentation?



Deep networks for classification

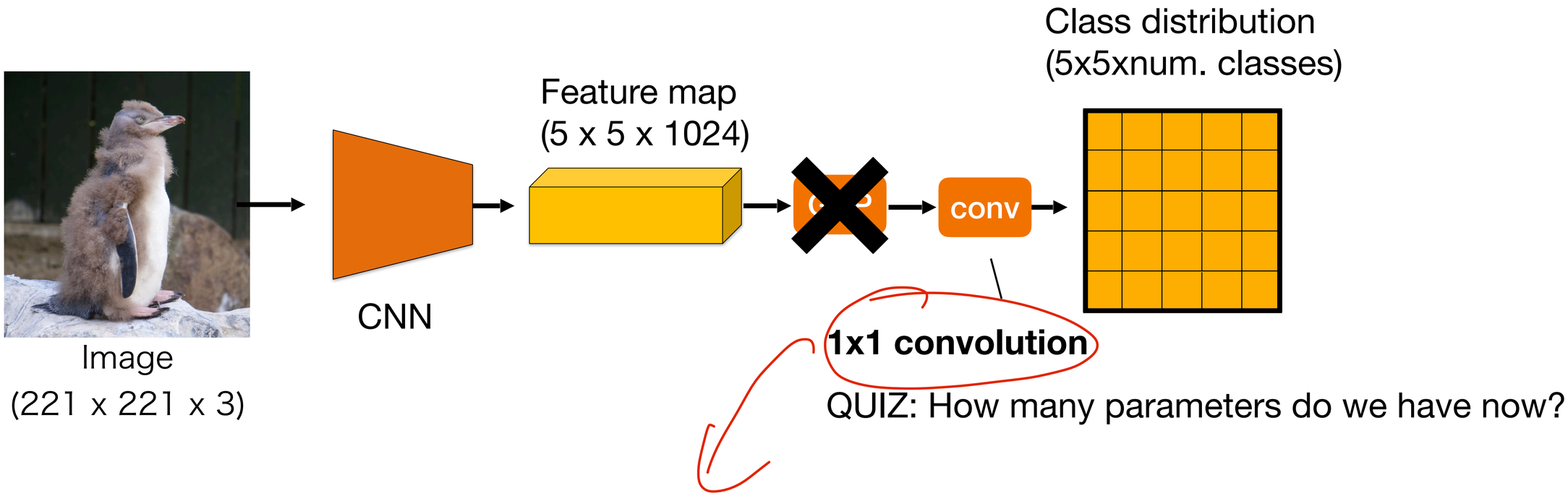
- Let us remove GAP:



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

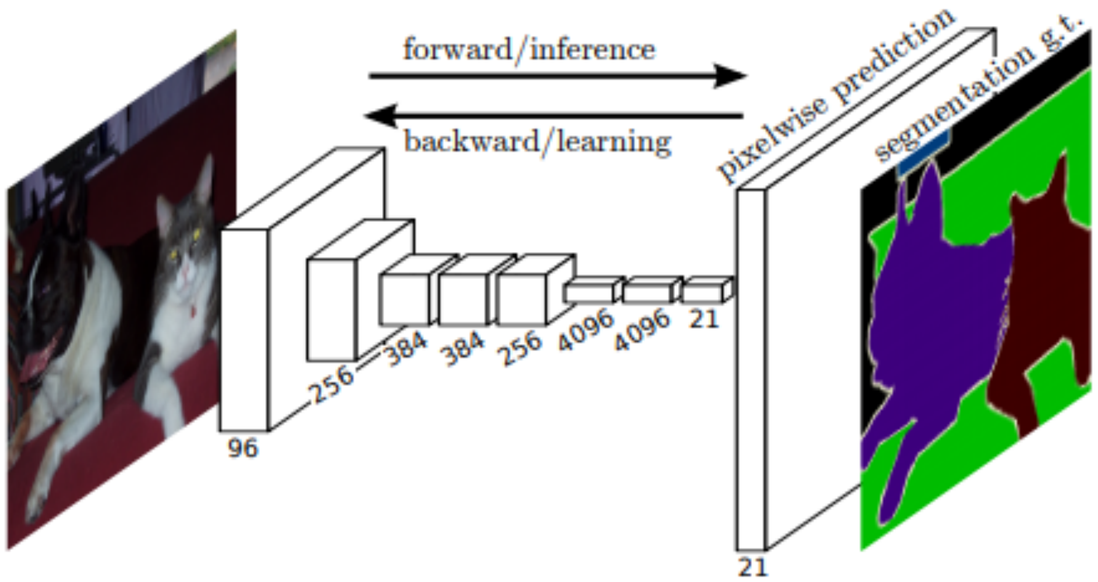
Deep networks for classification

- Replace fully connected layer with convolution:



- few parameters; variable input size; translation invariance!

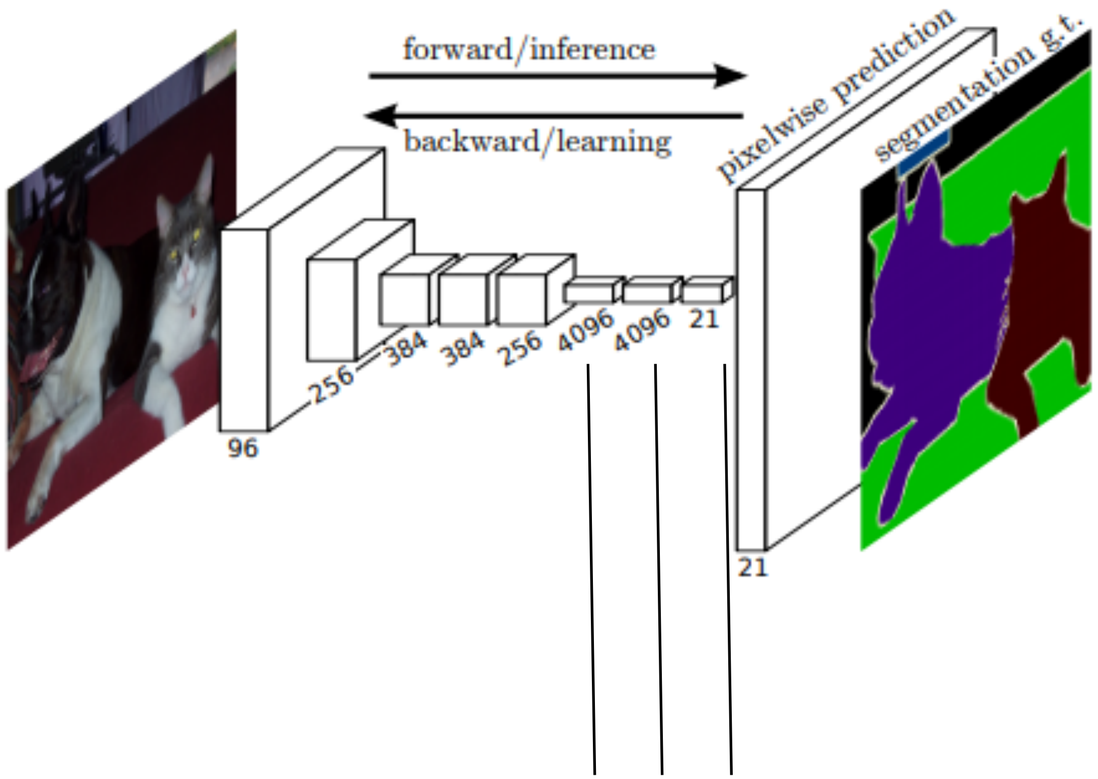
Fully convolutional neural networks



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

Fully convolutional neural networks

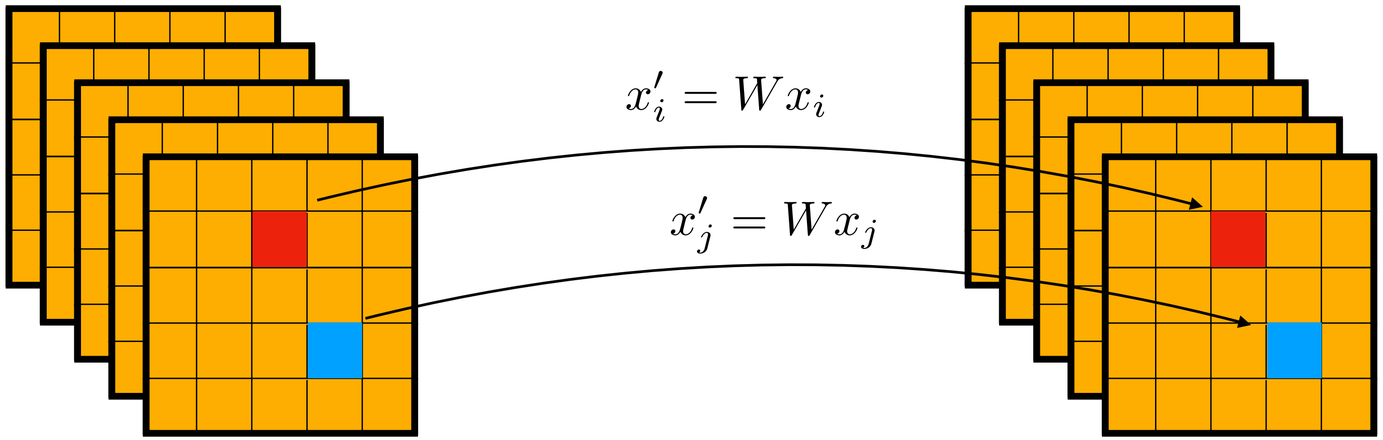
- Replace FC layers with convolutional layers.
- Upsample the last layer output to the original resolution.
- Can be trained with pixelwise cross-entropy with SGD.



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

1x1 “convolution”

- Every (feature) pixel is a multi-dimensional feature:



- 1x1 convolution is equivalent to applying a **shared** fully connected layer to every pixel feature!
  - Actually, no need to use `conv` OP in deep learning frameworks (though often done).

1x1 convolution facts

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

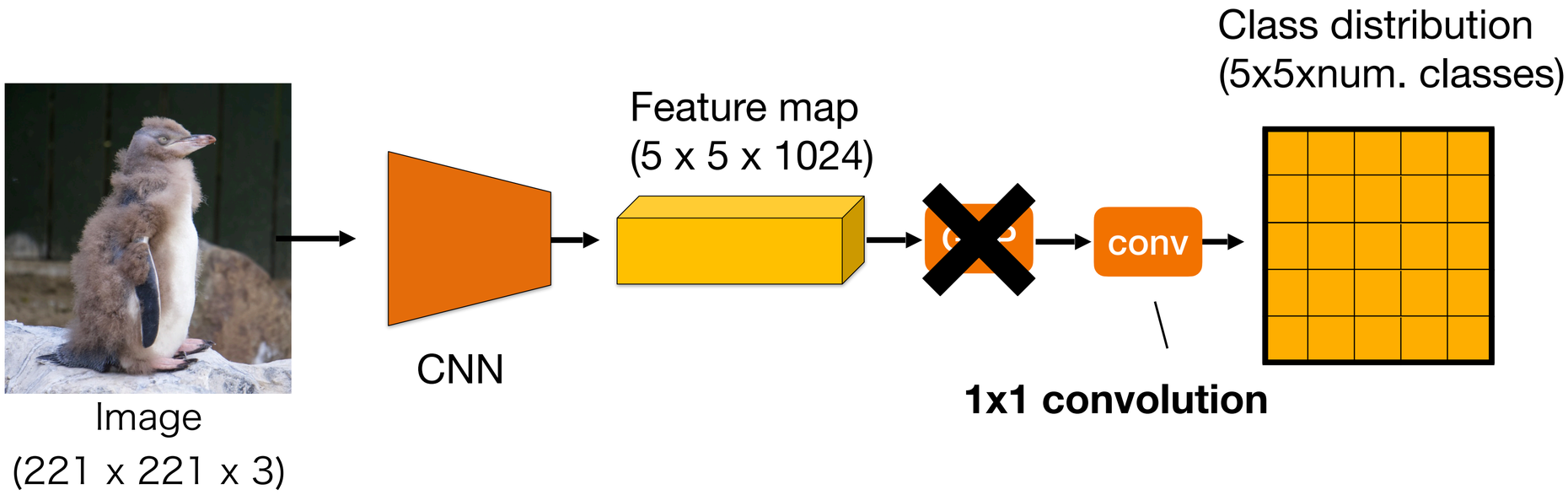
$$X' := W X$$

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

- 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

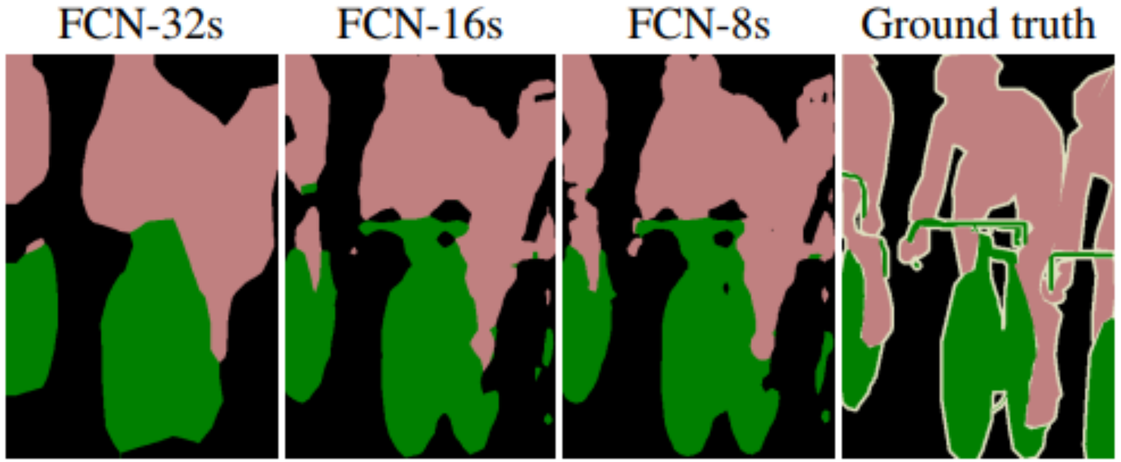
- Recall: we produce only a coarse grid



- Does the output resolution (here, 5x5) matter?

Qualitative results

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



Output stride:
input resolution / output resolution
(typically of an 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

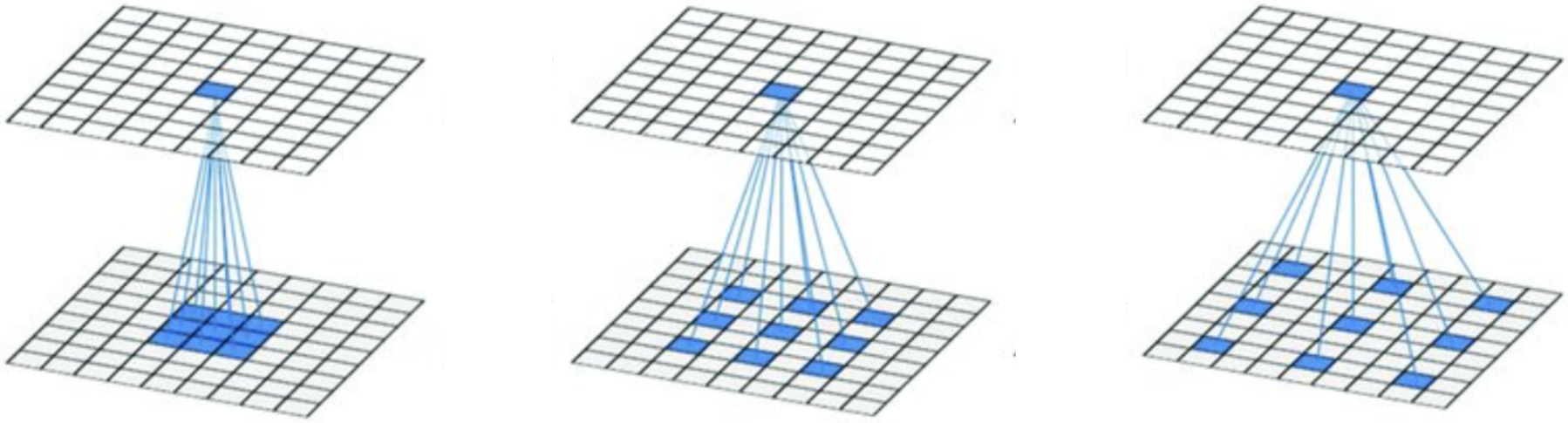
How to maintain the receptive field size?

- we can replace a stride-2 operation with multiple stride-1 operations
- deeper networks → harder and more expensive to train

- a better alternative – dilated convolutions:
  - maintains the size of the receptive field, while using the stride of 1.

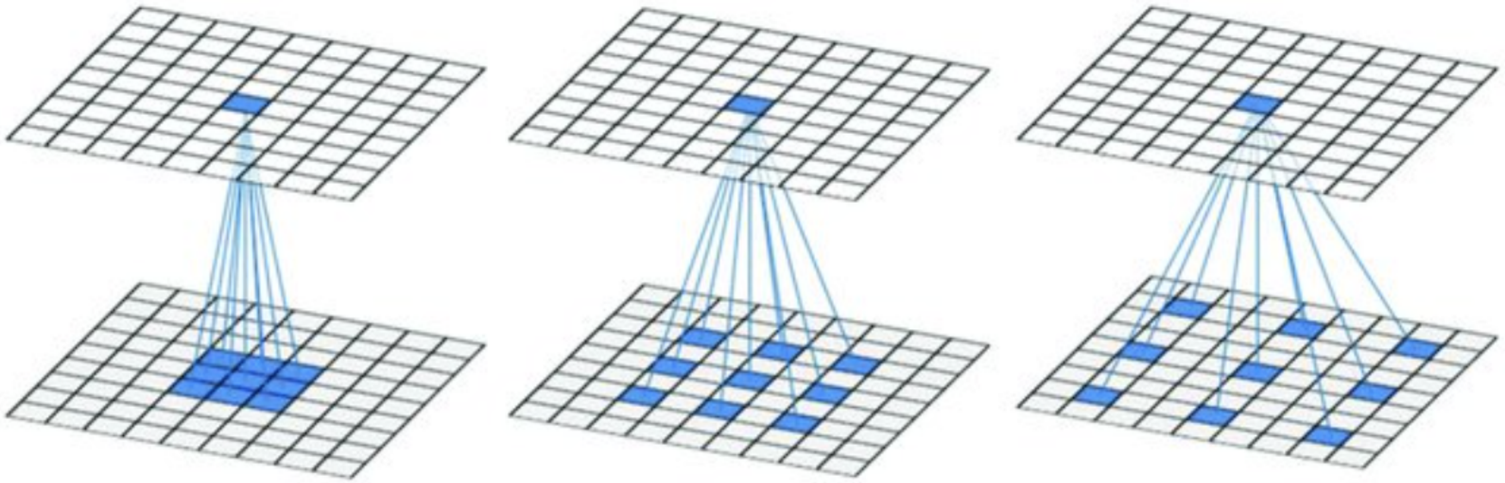
Dilated convolutions 2D

- 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



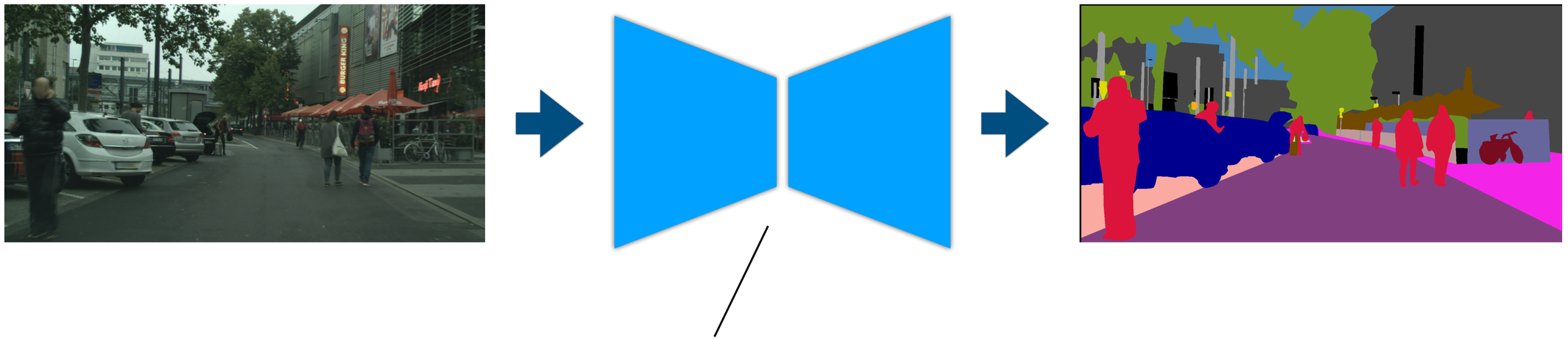
Note:

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

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

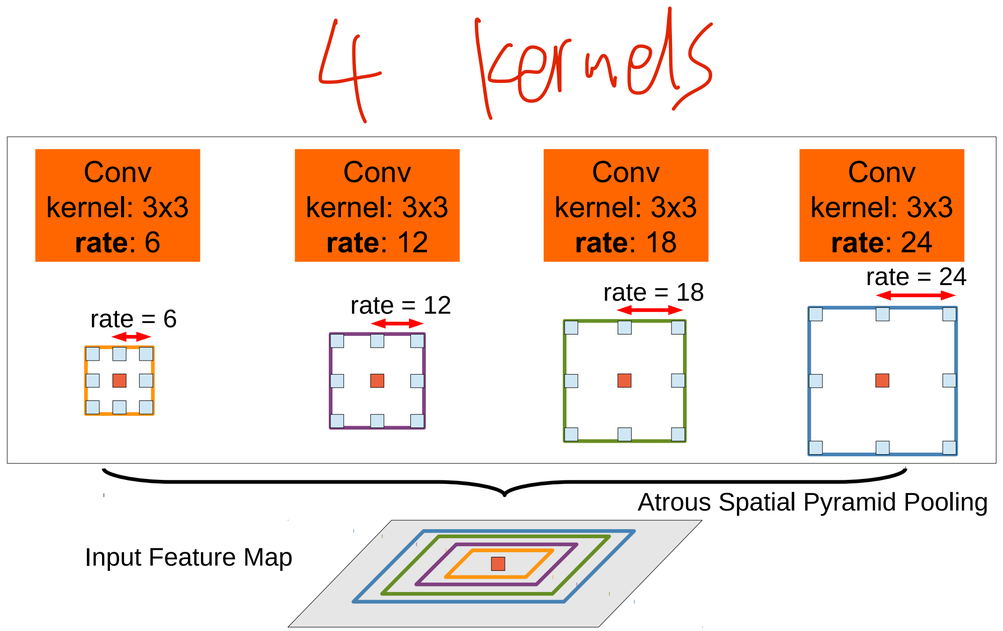
Dilated convolutions 2D

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

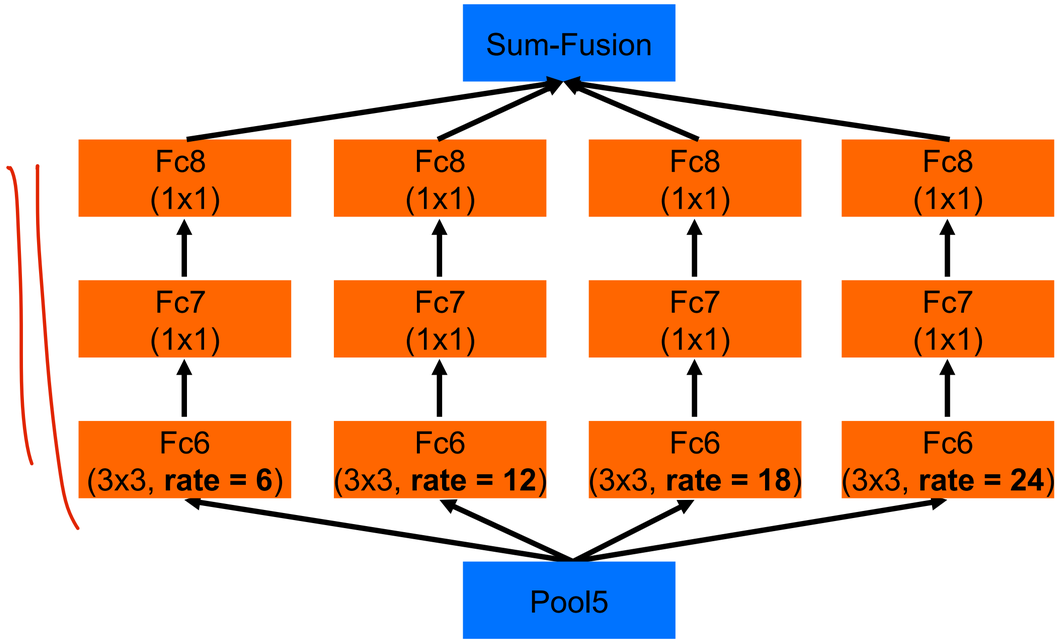


Typically at the “bottleneck” section

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:

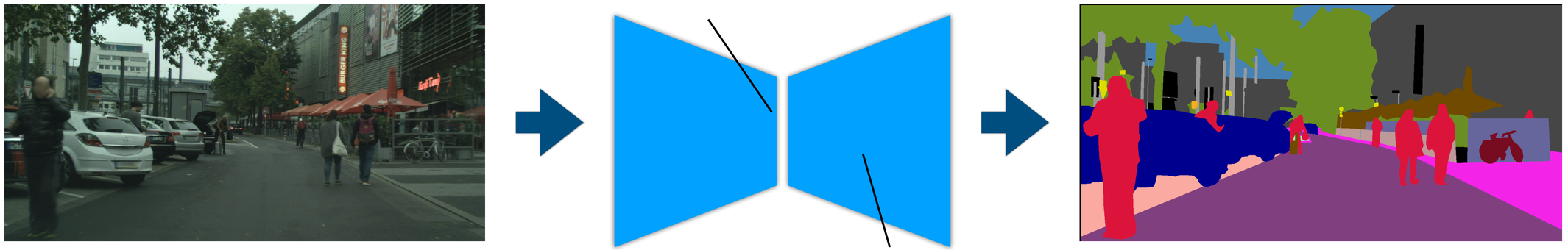
- receptive field size:
  - removing an operation with stride 2 reduces the area of the receptive field by a factor of 4;
  - less context information may get used.
- computational footprint (both memory and FLOPs):
  - e.g. feature tensor size increases 4 times for each remove stride-2 operation.

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.

reduced resolution (encoder)



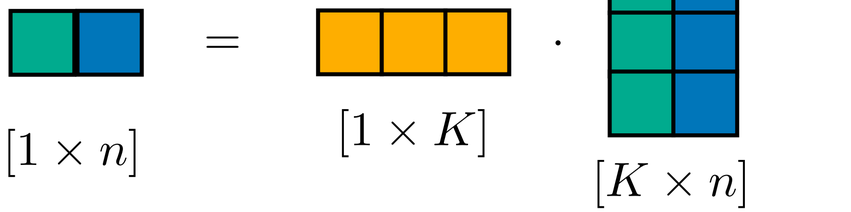
upsampling (decoder)

Upsampling

Transposed convolution

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

$$X' = WX \longrightarrow$$



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

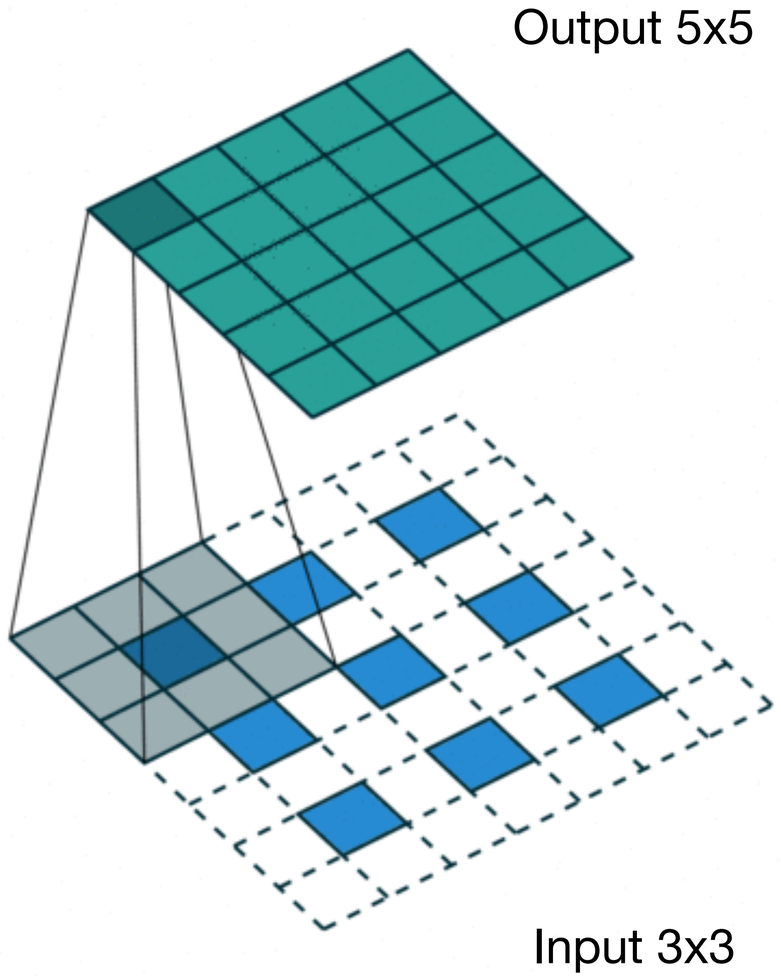
$$X = W^T X' \qquad \text{“broadcasting”}$$

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

Transposed convolution

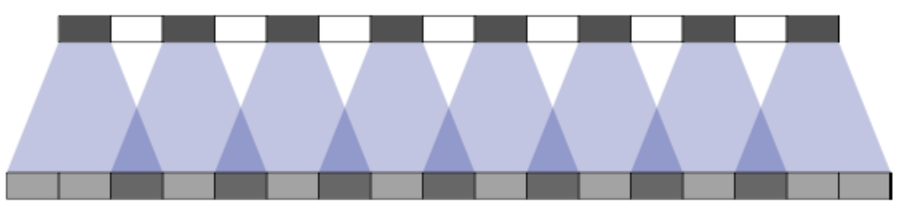
- Transposed convolution
  - also called “up-convolution”
  - or “deconvolution” (incorrect)

- 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



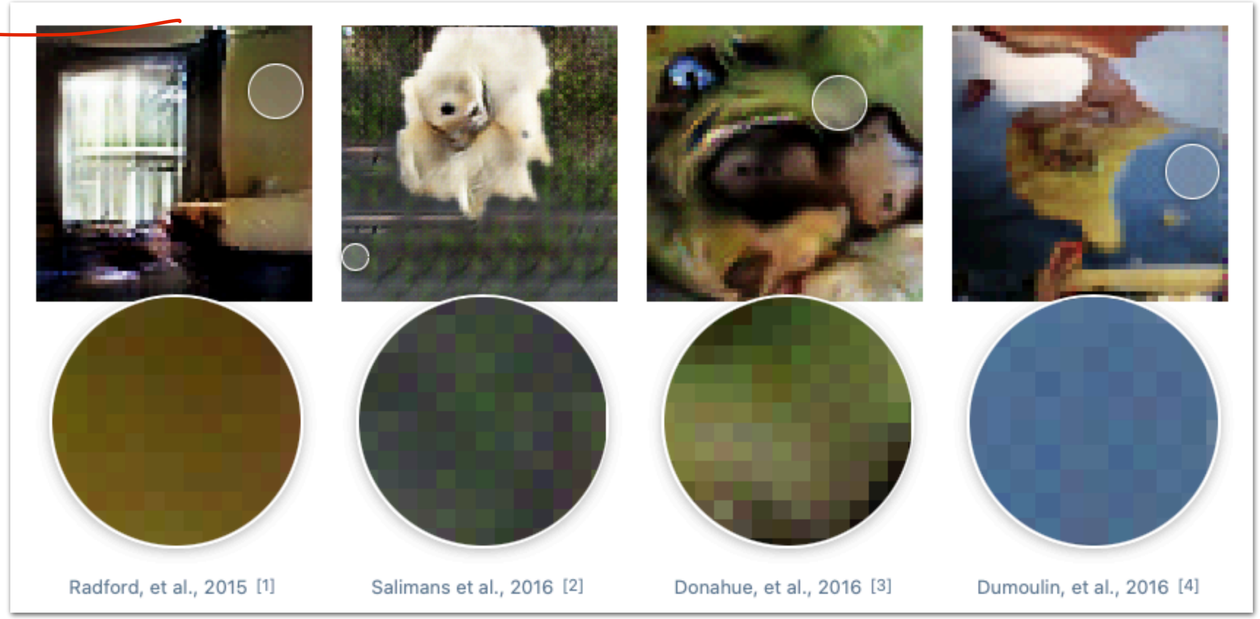
Transposed convolution

- Equivalent implementation without padding:



- Issue: checkboard artefacts

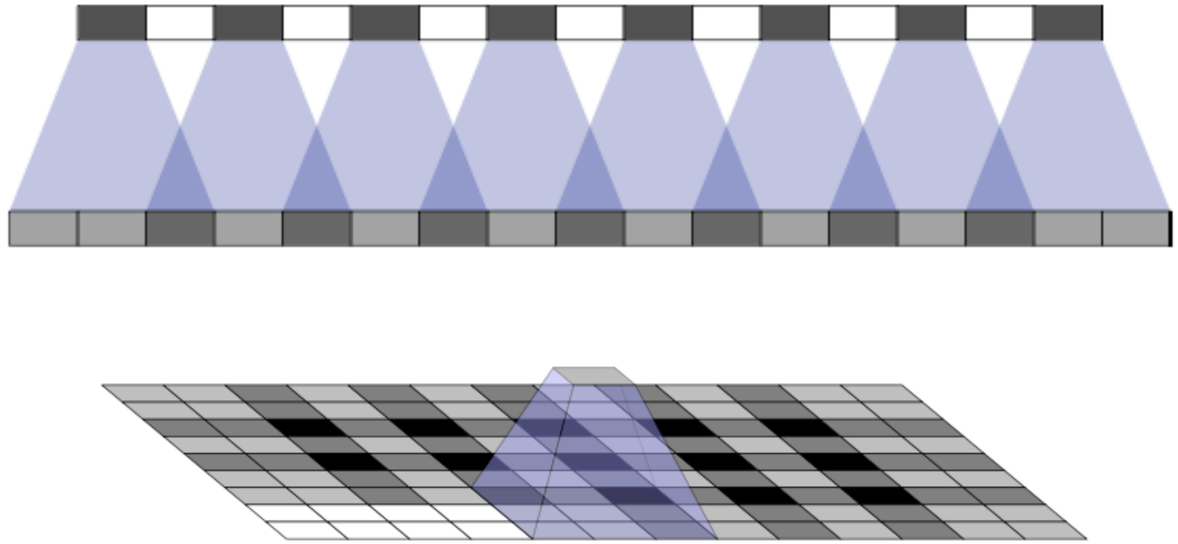
https://distill.pub/2016/deconv-checkerboard/



Transposed convolution

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

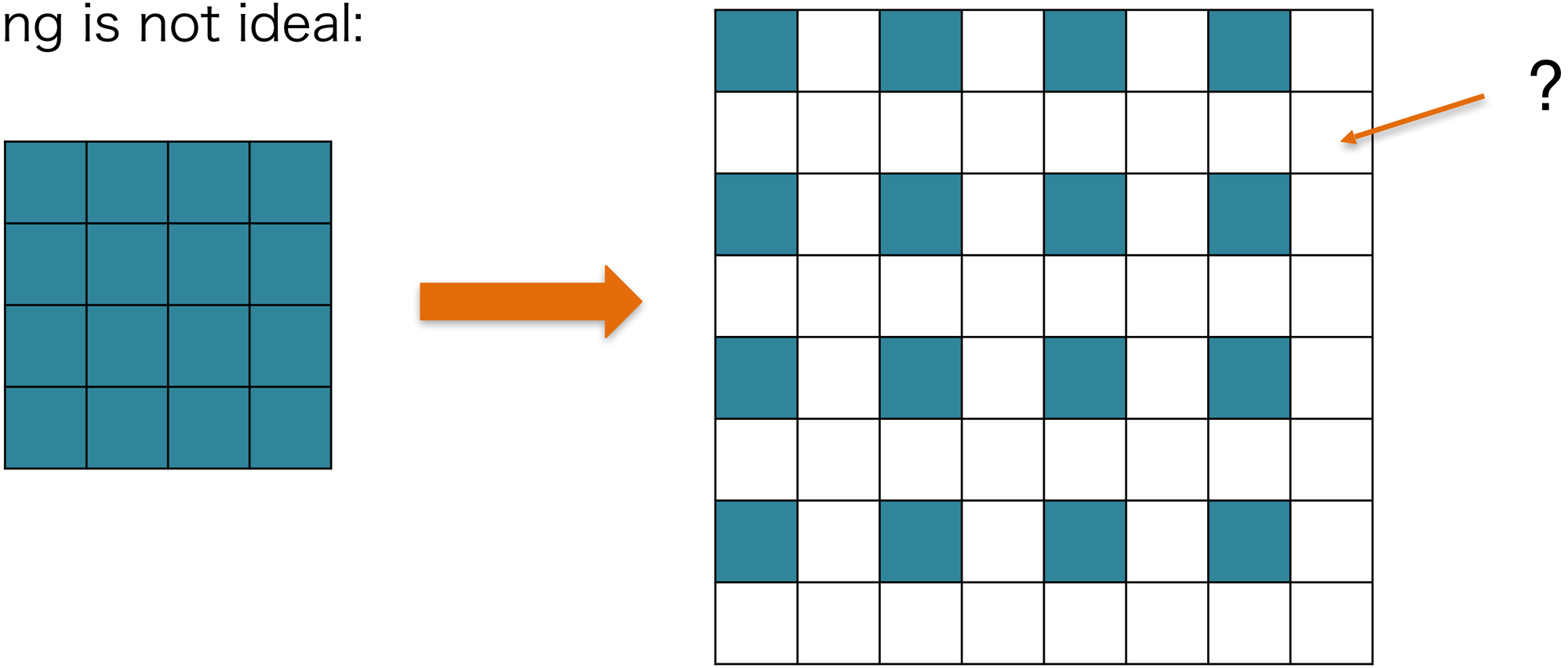
Kernel size: 3
Stride: 2



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

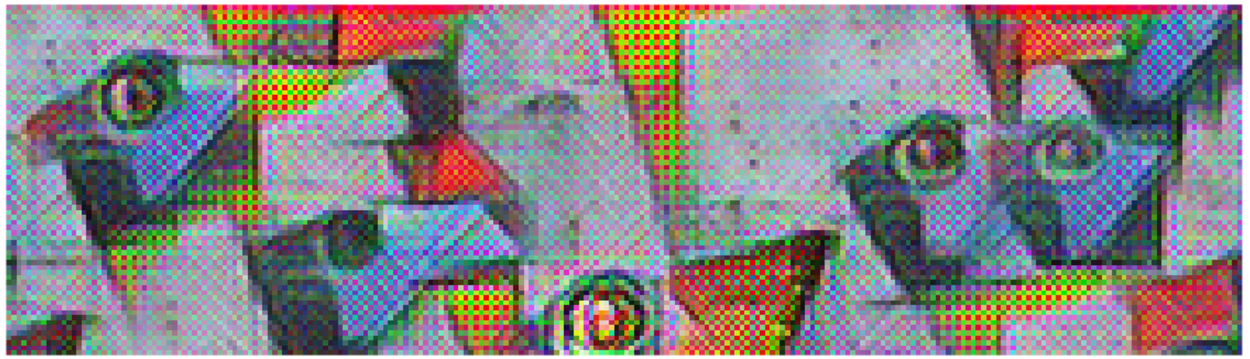
Bicubic interpolation

Resize-convolution

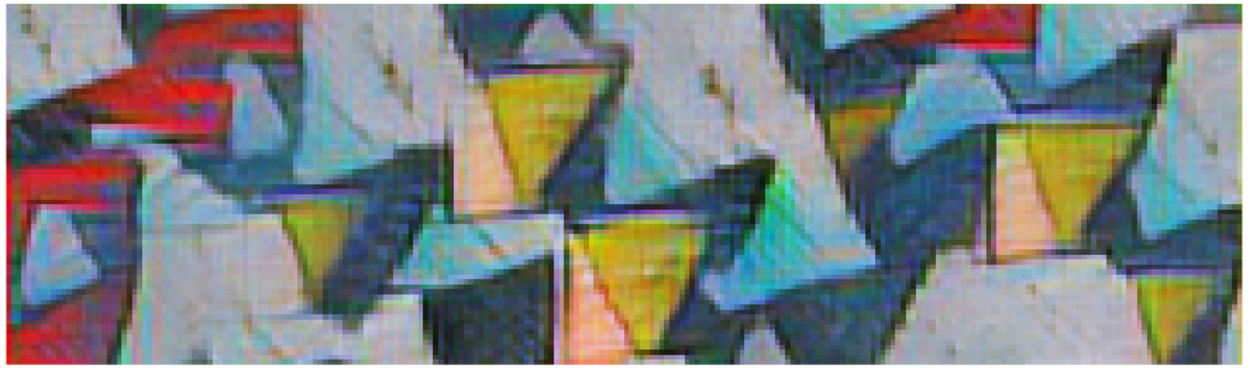
- Interpolate + convolution:

Image reconstruction example

Using transposed convolution



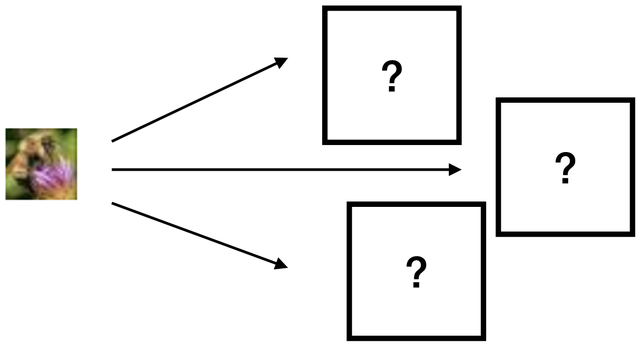
Resize-convolution



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

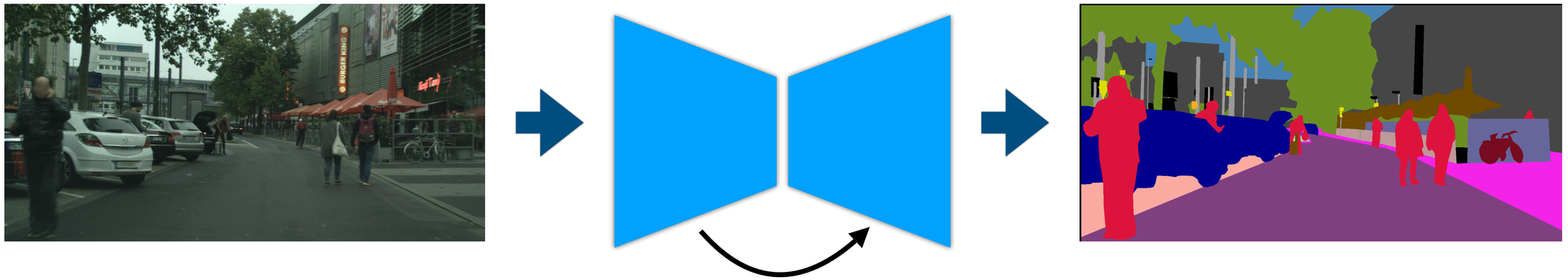
Resize-convolution

- 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

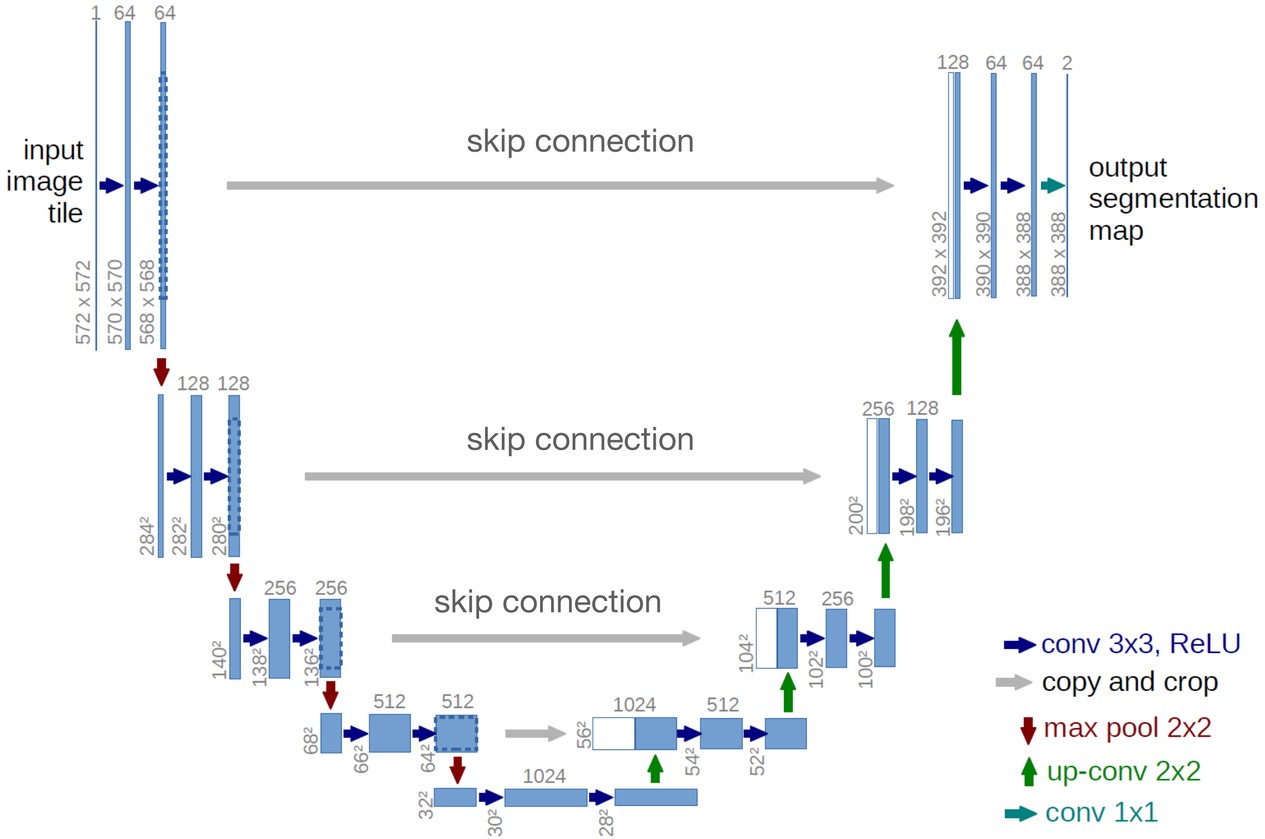
- Where can we obtain high-resolution information?
  - first layers in the encoder via a skip connection:



skip connection

Skip Connections

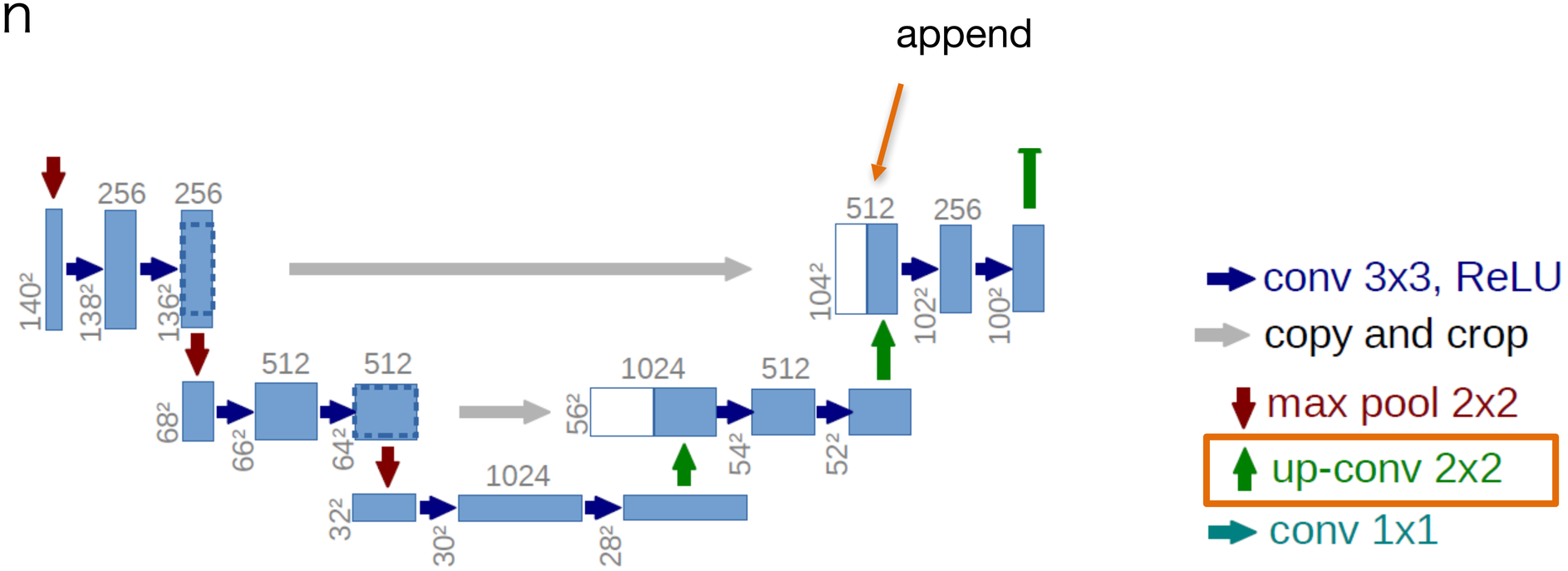
- U-Net
- QUIZ: Another example we have already seen?



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

Skip Connections

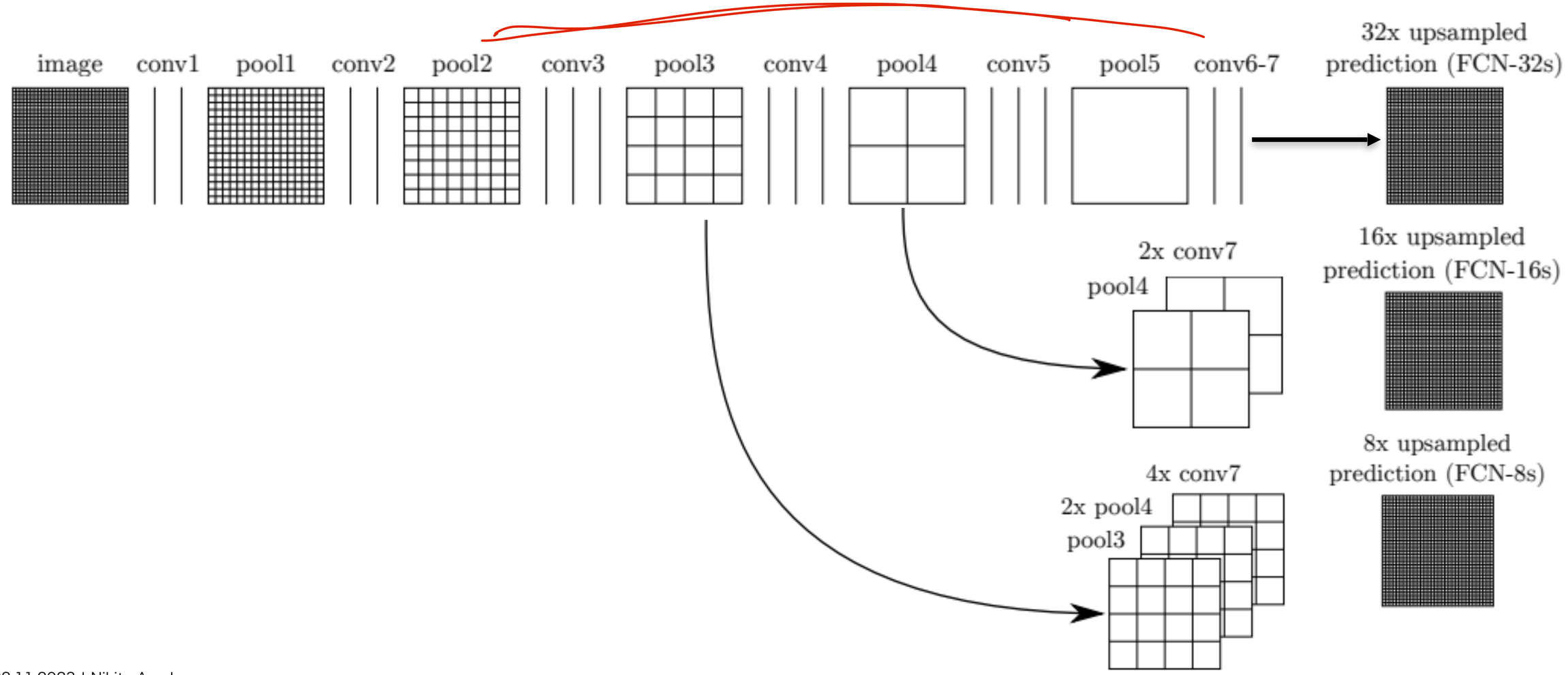
U-Net: zoom in



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

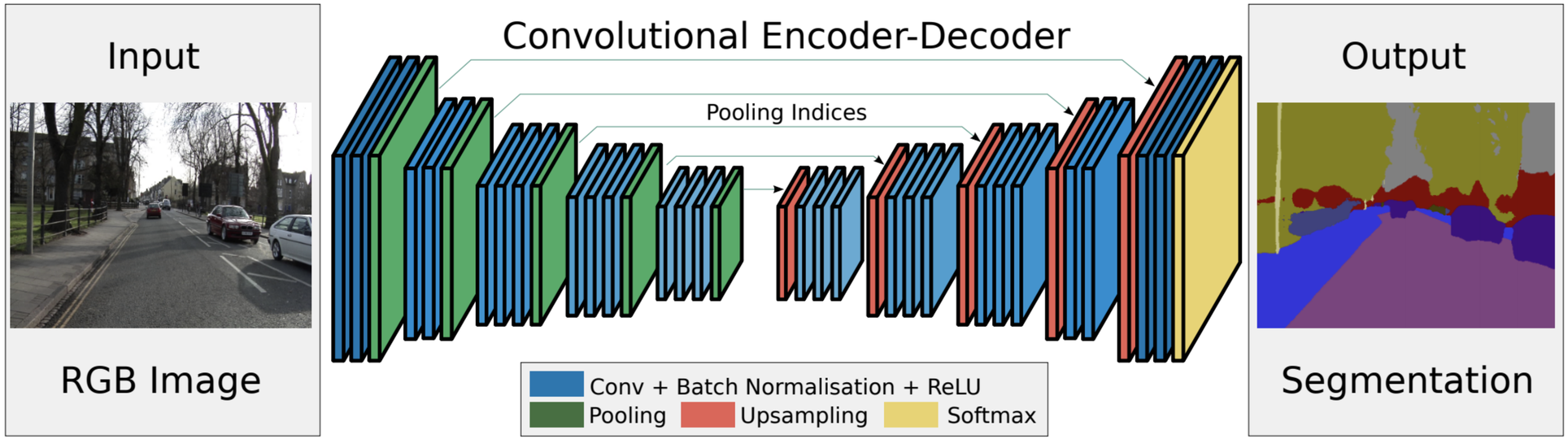
Network architecture

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

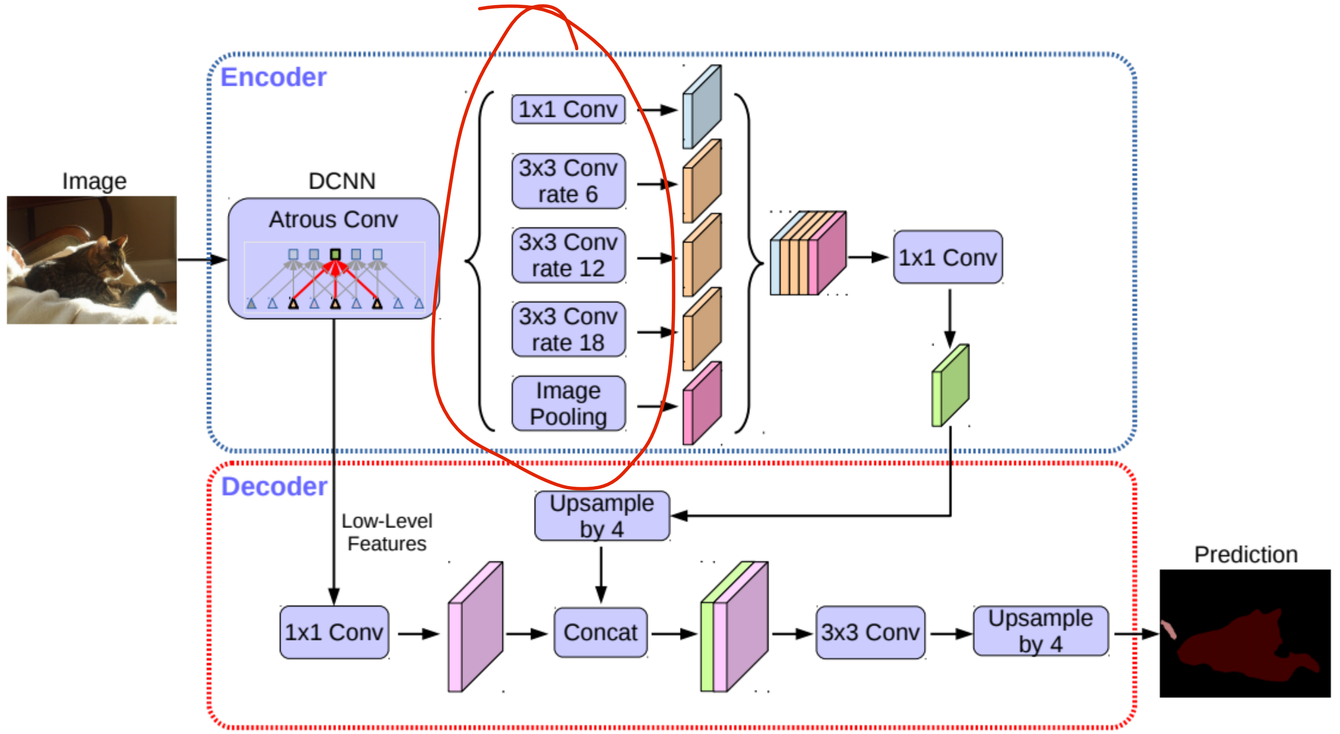
Summary

- 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.

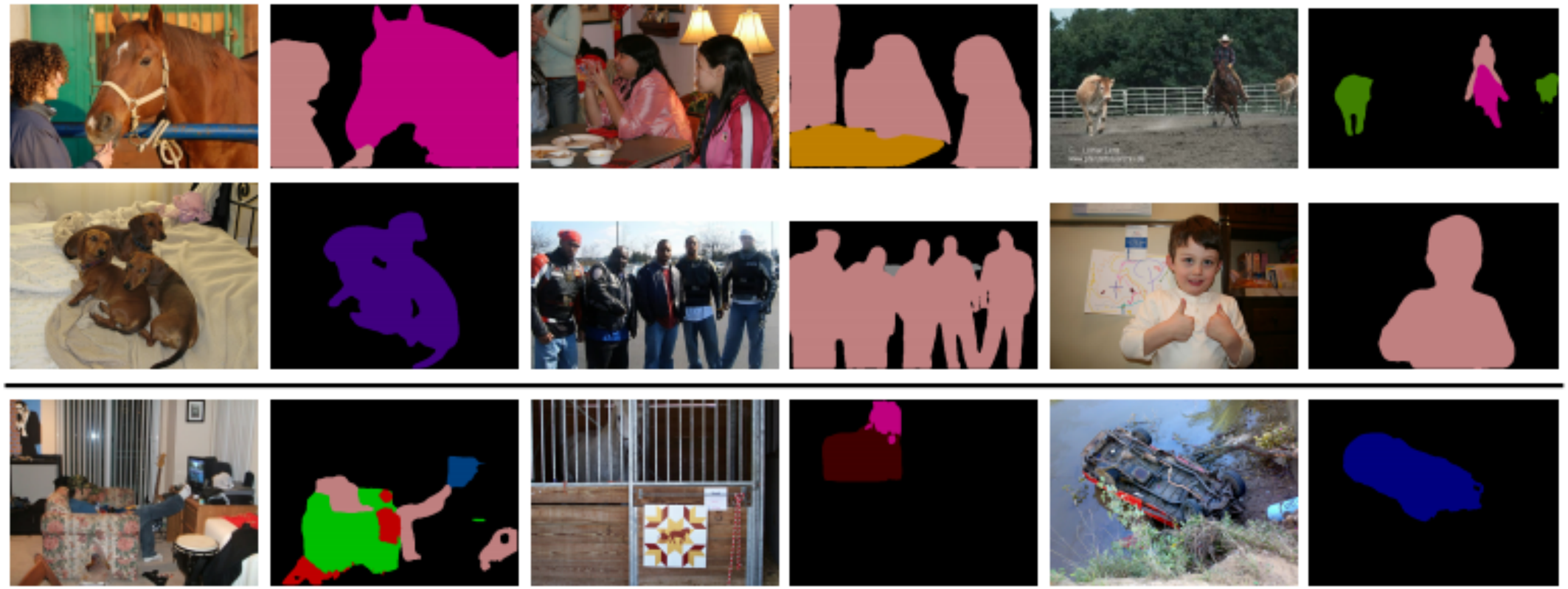
Delving deeper into DeepLabv3+

- 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)