Semantic segmentation
Task definition: semantic segmentation

Classify the main object in the image.

No objects, just classify each pixel.
Semantic Segmentation

- Every pixel in the image needs to be labelled with a category label.

- Do not differentiate between the instances (see how we do not differentiate between pixels coming from different cows).
Fully Convolutional Networks
Fully convolutional neural networks

- A FCN is able to deal with any input/output size

Fully convolutional neural networks

1. Replace FC layers with convolutional layers.
2. Convert the last layer output to the original resolution.
3. Do softmax-cross entropy between the pixelwise predictions and segmentation ground truth.
4. Backprop and SGD
“Convolutionalization”
Recall: Convolutions on Images

Image 5x5

Kernel 3x3

Output 3x3

\[
\begin{bmatrix}
-5 & 3 & 2 & -5 & 3 \\
4 & 3 & 2 & 1 & -3 \\
1 & 0 & 3 & 3 & 5 \\
-2 & 0 & 1 & 4 & 4 \\
5 & 6 & 7 & 9 & -1 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
0 & -1 & 0 \\
-1 & 5 & -1 \\
0 & -1 & 0 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
6 \\
\end{bmatrix}
\]

\[
5 \cdot 3 + (-1) \cdot 3 + (-1) \cdot 2 + (-1) \cdot 0 + (-1) \cdot 4 = 15 - 9 = 6
\]
1x1 Convolution

What is the output size?

<table>
<thead>
<tr>
<th>-5</th>
<th>3</th>
<th>2</th>
<th>-5</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>-3</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>-2</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>-1</td>
</tr>
</tbody>
</table>

Image 5x5

Kernel 1x1

2
1x1 Convolution

Image 5x5

Kernel 1x1

\[-5 \times 2 = -10\]
# 1x1 Convolution

## Image 5x5

<table>
<thead>
<tr>
<th></th>
<th>-5</th>
<th>3</th>
<th>2</th>
<th>-5</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>-3</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>-1</td>
<td></td>
</tr>
</tbody>
</table>

## Kernel 1x1

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

## Calculation

\[-1 \times 2 = -2\]
# 1x1 Convolution

<table>
<thead>
<tr>
<th>Image 5x5</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-5</td>
<td>3</td>
<td>2</td>
<td>-5</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>-3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>-1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-10</th>
<th>6</th>
<th>4</th>
<th>-10</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>-6</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>6</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>-4</td>
<td>0</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>14</td>
<td>18</td>
<td>-2</td>
</tr>
</tbody>
</table>

- **1x1 kernel:** keeps the dimensions and scales input
1x1 Convolution

- Same as having a 3 neuron fully connected layer
• As always we use more convolutional filters
Network in Network

As always, we use more convolutional filters.

[Li et al. 2013]
Using 1x1 Convolutions

- Use it to shrink the number of channels
- Further adds a non-linearity → one can learn more complex functions

```
200 32 32 Conv 1x1x200 + ReLU
```

CV3DST | Prof. Leal-Taixé
Semantic Segmentation (FCN)

- Fully Convolutional Networks for Semantic Segmentation

How do we upsample?

Network's architecture

Predict the segmentation mask from high level features

32x upsampled prediction (FCN-32s)
Network's architecture

Predict the segmentation mask from high level features

Predict the segmentation mask from mid-level features
Network's architecture

Predict the segmentation mask from high level features

Predict the segmentation mask from mid-level features

Predict the segmentation mask from low-level features
Network's architecture

Hierarchical training where the network is initially trained only based on high level features and then fine-tuned based on middle and low-level features.
This is important because it allows the network to also learn the mid and low-level details of the image, in addition to high level ones.
Qualitative results

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

Good | Better | Best
Qualitative results

SDS is an R-CNN-based method, i.e., it uses object proposals. In general, FCN outperforms significantly (both qualitatively and quantitatively) pre-deep learning and quasi-deep learning methods and is recognized as the AlexNet of semantic segmentation.
Autoencoder-style architecture
SegNet

- Step-wise upsampling

SegNet

- **Encoder**: normal convolutional filters + pooling

- **Decoder**: Upsampling + convolutional filters

SegNet

- **Encoder**: normal convolutional filters + pooling
- **Decoder**: Upsampling + convolutional filters

SegNet

- **Encoder**: normal convolutional filters + pooling
- **Decoder**: Upsampling + convolutional filters
- The convolutional filters in the decoder are learned using backprop and their goal is to refine the upsampling

Transposed convolution

- Transposed convolution
  - Unpooling
  - Convolution filter (learned)
  - Also called up-convolution (never deconvolution)
Upsampling
Types of upsampleings

• 1. Interpolation
Types of upsamplings

- 1. Interpolation
  - Nearest neighbor interpolation
  - Bilinear interpolation
  - Bicubic interpolation

Original image \( \times 10 \)

Image: Michael Guerzhoy
Types of upsamplings

• 1. Interpolation

    Few artifacts
Types of upsamplings

- 2. Fixed unpooling

A. Dosovitskiy, “Learning to Generate Chairs, Tables and Cars with Convolutional Networks”. TPAMI 2017
Types of upsamplings

- 3. Unpooling: “à la DeconvNet”

Keep the locations where the max came from.

Zeiler and Fergus. "Visualizing and understanding convolutional neural networks". ECCV 2014
Types of upsamplings

• 3. Unpooling: “à la DeconvNet”

Keep the details of the structures
Skip connections
(U-Net)
Skip Connections

• U-Net

Recall ResNet

Skip Connections

• U-Net: zoom in

Skip Connections

- Concatenation connections
DeepLab
DeepLab

Input

Deep Convolutional Neural Network

Aeroplane Coarse Score map

Fully Connected CRF

Final Output

Bi-linear Interpolation
Semantic Segmentation: 3 challenges

- Reduced feature resolution
  - Proposed solution: Atrous convolutions

- Objects exist at multiple scales
  - Proposed solution: Pyramid pooling, as in detection.

- Poor localization of the edges
  - Proposed solution: Refinement with Conditional Random Field (CRF)
Semantic Segmentation: 3 challenges

• Reduced feature resolution
  – Proposed solution: Atrous convolutions

• Objects exist at multiple scales
  – Proposed solution: Pyramid pooling, as in detection.

• Poor localization of the edges
  – Proposed solution: Refinement with Conditional Random Field (CRF)
Wish: no reduced feature resolution

pixels in
width x height x RGB

Just convs & activations

Fully Convolutional Network

pixels out
width x height x classes

Super expensive!
Alternative: Dilated (atrous) convolutions

Sparse feature extraction with standard convolution on a low resolution input feature map.
Alternative: Dilated (atrous) convolutions

Sparse feature extraction with standard convolution on a low resolution input feature map.

Dense feature extraction with atrous convolution with rate $r=2$, applied on a high resolution input feature map.
Dilated (atrous) convolutions 1D

(a) Sparse feature extraction with standard convolution on a low resolution input feature map.

(b) Dense feature extraction with atrous convolution with rate $r = 2$, applied on a high resolution input feature map.
Dilated (atrous) convolutions in 2D

Standard convolution has dilation 1

An analogy for dilated conv is a conv filter with holes

```python
class torch.nn.Conv2d (in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=2)

class torch.nn.ConvTranspose2d (in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=2)
```
Dilated (atrous) convolutions 2D

(a) the dilation parameter is 1, and each element produced by this filter has receptive field of 3x3.

(b) the dilation parameter is 2, and each element produced by it has receptive field of 7x7.

(c) the dilation parameter is 4, and each element produced by it has receptive field of 15x15.
Dilated (atrous) convolutions 2D

Each layer has the same number of parameters, but the receptive field grows exponentially while the number of parameters grows linearly.
Semantic Segmentation: 3 challenges

• Reduced feature resolution
  – Proposed solution: Atrous convolutions

• Objects exist at multiple scales
  – Proposed solution: Pyramid pooling, as in detection.

• Poor localization of the edges
  – Proposed solution: Refinement with Conditional Random Field (CRF)
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 \( j \)
Effect of number of iterations of CRF

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

(a) Image

(b) G.T.

(c) Before CRF

(d) After CRF
Problems with CRF

- The network is not trained end-to-end. The FCN and the CRF are trained independently from each other.
- This makes the training both slow and arguably suboptimal.

Solution: Formulate CRF as an Recurrent Neural Network

Zheng et al., Conditional Random Fields as Recurrent Neural Networks, ICCV 2015
Replacing CRF with an RNN

Zheng et al., Conditional Random Fields as Recurrent Neural Networks, ICCV 2015
CRF-RNN: qualitative results
CRF-RNN: qualitative results
Why do we need the CRF?

• To properly localize the masks, i.e., get the contours correctly

• We need to process information at the original (image) resolution for this. We need to look at the pixels. → CRF is conditioned on the RGB image.

• What if we use attention?
Attention
The problem

- For very long sentences, the score for machine translation really goes down after 30-40 words.

Bahdanau et al. 2014. Neural machine translation by jointly learning to align and translate.
Basic structure of a RNN

- We want to have notion of “time” or “sequence”
Basic structure of a RNN

- We want to have notion of “time” or “sequence”

\[
A_t = \theta_c A_{t-1} + \theta_x x_t
\]

Parameters to be learned
Basic structure of a RNN

• We want to have notion of “time” or “sequence”

Output

Hidden state

\[ A_t = \theta_c A_{t-1} + \theta_x X_t \]

\[ h_t = \theta_h A_t \]

Same parameters for each time step = generalization!
Basic structure of a RNN

- Unrolling RNNs

$A$  
$h_t$  
$X_t$  

$A$  
$h_0$  
$X_0$  

$A$  
$h_1$  
$X_1$  

$A$  
$h_2$  
$X_2$  

...  

$A$  
$h_t$  

Hidden state is the same

Image: Christopher Olah - Understanding LSTMs
Basic structure of a RNN

• Unrolling RNNs
Long-term dependencies

I moved to Germany ... so I speak German fluently
Attention: intuition

ATTENTION: Which hidden states are more important to predict my output?

I moved to Germany ... so I speak German fluently.
Attention: intuition

I moved to Germany ... so I speak German fluently
Attention: architecture

- A decoder processes the information.
- Decoders take as input:
  - Previous decoder hidden state
  - Previous output
  - Attention
Attention

• $\alpha_{1,t+1}$ indicates how much the word in the position 1 is important to translate the word in position $t + 1$

• The context aggregates the attention

\[ c_{t+1} = \sum_{k=1}^{t+1} \alpha_{k,t+1} a_k \]

• **Soft** attention: All attention masks alpha sum up to 1
Computing the attention mask

- We can train a small neural network

\[
\alpha_{1,t+1} = \frac{\exp f_{1,t+1}}{\sum_{k=1}^{t+1} \exp f_{k,t+1}}
\]

Previous state of the decoder

Hidden state of the encoder

\( d_t \)

\( a_1 \)

\( f_{1,t+1} \)

\( \text{NN} \)
How do we translate?
First read the whole sentence in language 1.
Afterwards, translate the whole sentence in language 2.
Seq2Seq + Attention?

- If the sentence is very long, we might have forgotten what was said at the beginning.

- Solution: take “notes” of keywords as we read the sentence in language 1.

- Use attention!
Seq2Seq + Attention

Animation from: https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3
Seq2Seq + Attention

Animation from: https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3
Seq2Seq + Attention

Animation from: https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3
Seq2Seq + Attention

Animation from: https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3
Seq2Seq + Attention

Animation from: https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3
Seq2Seq + Attention

Animation from: https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3
Transformers

• What if we could get rid of the recurrent architecture and use only attention?
• All the memory problems of RNNs could disappear
• No RNN, no CNN, just attention!

• Current state-of-the-art in NLP!
Transformers

Translation?
Sentiment?
Next word?
Part-of-speech tags?
Transformers

- Wait, what does this remind you of?

Translation?
Sentiment?
Next word?
Part-of-speech tags?
Transformers

• Broadly speaking, Transformers are based on Graph Attention Networks (GAT)

• GAT replace the aggregation operation of GNN (usually a summation) by a weighted sum, i.e., an attention mechanism
Why do we need attention?

• We use the whole image to make the classification

• Are all pixels equally important?
Why do we need attention?

• Wouldn’t it be easier and computationally more efficient to just run our classification network on the patch?
Attention for semantic segmentation

The attention model learns to put different weights on objects of different scales.

For example, the model learns to put large weights on the small-scale person (green dashed circle) for features from scale = 1, and large weights on the large-scale child (magenta dashed circle) for features from scale = 0.5. We jointly train the network component and the attention model.
• Do we even need these blocks which include the global information (CRF, RNN, attention)?

Spoiler alert: Not necessarily.
DeepLabv3+

Combine atrous convolutions and spatial pyramid pooling with an encoder-decoder module.
Delving deeper into DeepLabv3+

1) Encode multi-scale contextual information by applying atrous convolution at multiple scales.
Delving deeper into DeepLabv3+

1) Encode multi-scale contextual information by applying atrous convolution at multiple scales.

2) Refine the segmentation results along object boundaries.
Delving deeper into DeepLabv3+

1) Encode multi-scale contextual information by applying atrous convolution at multiple scales.

2) Refine the segmentation results along object boundaries.

3) Use depthwise separable convolutions.
Depth-wise separable convolutions

```
12 12 3

5 5 3

8 8 256
```

Normal convolutions act on all channels.
Depth-wise separable convolutions

Filters are applied only at certain depths of the features. Normal convolutions have groups set to 1, the convolutions used in this image have groups set to 3.

```python
class torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0, groups=3)

class torch.nn.ConvTranspose2d(in_channels, out_channels, kernel_size, stride=1, padding=0, groups=3)
```
Depth-wise separable convolutions

```
class torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0, groups=3)
class torch.nn.ConvTranspose2d(in_channels, out_channels, kernel_size, stride=1, padding=0, groups=3)
```

But the depth size is always the same!
Depth-wise separable convolutions

Solution: 1x1 convs!
But why?

Original convolution
256 kernels of size 5x5x3

Multiplications:
256x5x5x3 x (8x8 locations) = 1.228.800
But why?

Original convolution
256 kernels of size 5x5x3

Multiplications:
$256 \times 5 \times 5 \times 3 \times (8 \times 8 \text{ locations}) = 1,228,800$

Depth-wise convolution
3 kernels of size 5x5x1

Multiplications:
$5 \times 5 \times 3 \times (8 \times 8 \text{ locations}) = 4,800$

Less computations!

1x1 convolution
256 kernels of size 1x1x3

Multiplications:
$256 \times 1 \times 1 \times 3 \times (8 \times 8 \text{ locations}) = 49,152$
DeepLabv3+: qualitative results

Still considered as SOTA!

Chen et al., Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation, ECCV 2018
DeepLab is great…

• …but there are other important architectures

• Recommended to read
  – RefineNet
  – PSPNet
Many building blocks but the goal is the same: use convolutional layers to refine the information coming from different scales.

Lin et al., RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation, CVPR 2017
Similar idea to RefineNet (fuse information from multiple scales), but the features here are shared (and the multi-scaling comes from pooling). The method is simpler than RefineNet and performs slightly better.
Datasets and metrics
Datasets

Pascal VOC 2012:
9993 natural images divided into 20 classes.

Cityscapes:
25K urban-street images divided into 30 classes.

ADE20K:
25K (20 stands for 20K training) scene-parsing images divided into 150 classes.

Mapillary Vistas:
25K street level images, divided into 152 classes.

Models are often pre-trained in the large MS-COCO dataset, before finetuned to the specific dataset.
Metrics: intersection over union (IoU)
Metrics: intersection over union (IoU)

\[
\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}
\]
Metrics: mean intersection over union (mIoU)

\[
\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}
\]

MIoU simply computes the IoU for each class and then computes the mean of those values.

Another widely used metric is the pixel accuracy (ratio of pixels classified correctly).
Semantic segmentation