SEMANTIC IMAGE SEGMENTATION WITH DEEP CONVOLUTIONAL NETS AND FULLY CONNECTED CRFS

Paper by Chen, Papandreou, Kokkinos, Murphy, Yuille

Slides by Josh Kelle (with graphics from the paper)
Semantic Segmentation

Goal: Partition the image into semantically meaningful parts, and classify each part.
Main Idea

1. Use CNN to generate a rough prediction of segmentation (smooth, blurry heat map)

2. Refine this prediction with a conditional random field (CRF)
Why are CNNs insufficient?

Too much invariance. Good for high-level vision tasks like classification, bad for low level tasks like segmentation.

• Problem: subsampling  
  Solution: ‘atrous’ algorithm (hole algorithm)

• Problem: spatial invariance (shared kernel weights)  
  Solution: fully connected CRF
Example

![Image](image)

![Ground Truth](ground_truth)

- DCNN output
- CRF 1 iteration
- CRF 2 iteration
- CRF 10 iteration
Part 1: CNN
CNNs for Dense Feature Extraction

- Construct “DeepLab” by modifying VGG-16 (a 16-layer CNN pre-trained on ImageNet, publicly available).

- Convert the fully-connected layers of VGG-16 into convolutional layers.

- Skip subsampling after the last two max-pooling layers.
Hole Algorithm

- How to skip max pooling, but keep learned kernels the same?
- Could introduce zeros into the kernels, but that’s slow.
- The hole algorithm is faster.
Image Resolution

• CNN shrinks the image. We need image at original resolution.

• Skipping the last two phases of max pooling helps, but the CNN output is still 8x too small.

• Since the score maps are smooth, just use bi-linear interpolation to grow the image.
Part 2: CRF
Fully Connected CRF

- Traditionally, short range CRFs are used to smooth noisy segmentation.

- CNN output is already very smooth. Short range CRF would make it worse.

- Use a fully connected CRF. The graphical model has every pixel connected to every other pixel.
CRF Energy Function

\[ E(\mathbf{x}) = \sum_i \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) \]

where \( x_i \) is assignment of pixel \( i \)

\[ \theta_i(x_i) = -\log P(x_i) \]

\[ P(x_i) = \text{label assignment probability computed by CNN} \]
CRF Energy Function

\[ \theta_{ij}(x_i, x_j) = \mu(x_i, x_j) \sum_{m=1}^{K} w_m \cdot k^m(f_i, f_j) \]
CRF Energy Function

$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j) \sum_{m=1}^{K} w_m \cdot k^m(f_i, f_j)$$

$$\mu(x_i, x_j) = 1 \text{ if } x_i \neq x_j, \text{ and zero otherwise}$$

indicator function
CRF Energy Function

\[
\theta_{ij}(x_i, x_j) = \mu(x_i, x_j) \sum_{m=1}^{K} w_m \cdot k^m(f_i, f_j)
\]

(\(\mu(x_i, x_j) = 1\) if \(x_i \neq x_j\), and zero otherwise)

\[
\sum_{m=1}^{K} w_m \cdot k^m(f_i, f_j) = w_1 \exp \left( - \frac{||p_i - p_j||^2}{2\sigma^2_{\alpha}} \right) - \frac{||I_i - I_j||^2}{2\sigma^2_{\beta}} \\
+ w_2 \exp \left( - \frac{||p_i - p_j||^2}{2\sigma^2_{\gamma}} \right)
\]

(p = pixel position, I = pixel color intensities)

2 Gaussian kernels

(w and \(\sigma\) are hyper parameters fit with cross validation)
Full Pipeline
“DeepLab-CRF”

Input

Deep Convolutional Neural Network

Coarse Score map

Bi-linear Interpolation

Fully Connected CRF

Aeroplane

Final Output
Comparison to state-of-the-art

<table>
<thead>
<tr>
<th>Method</th>
<th>mean IOU (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSRA-CFM</td>
<td>61.8</td>
</tr>
<tr>
<td>FCN-8s</td>
<td>62.2</td>
</tr>
<tr>
<td>TTI-Zoomout-16</td>
<td>64.4</td>
</tr>
<tr>
<td>DeepLab-CRF</td>
<td>66.4</td>
</tr>
<tr>
<td>DeepLab-MSc-CRF</td>
<td>67.1</td>
</tr>
<tr>
<td>DeepLab-MSc-CRF-LargeFOV</td>
<td>71.6</td>
</tr>
</tbody>
</table>
Comparison to state-of-the-art

- image
- ground truth
- FCN-8s
- DeepLab-CRF
Comparison to state-of-the-art
Success Cases

image  ground truth  DeepLab  DeepLab-CRF
Failure Cases

image | ground truth | DeepLab | DeepLab-CRF
Conclusion

• Modify the CNN architecture to become less spatially invariant.

• Use the CNN to compute a rough score map.

• Use a fully connected CRF to sharpen the score map.