Deep learning

8.4. Networks for semantic segmentation

François Fleuret

https://fleuret.org/dlc/

Nov 24, 2020
The historical approach to image segmentation was to define a measure of similarity between pixels, and to cluster groups of similar pixels.
The historical approach to image segmentation was to define a measure of similarity between pixels, and to cluster groups of similar pixels. Such approaches account poorly for semantic content.
The historical approach to image segmentation was to define a measure of similarity between pixels, and to cluster groups of similar pixels. Such approaches account poorly for semantic content.

The deep-learning approach re-casts semantic segmentation as pixel classification, and re-uses networks trained for image classification by making them fully convolutional.
Shelhamer et al. (2016) proposed the FCN (“Fully Convolutional Network”) that uses a pre-trained classification network (e.g. VGG 16 layers).

The fully connected layers are converted to $1 \times 1$ convolutional filters, and the final one retrained for 21 output channels (VOC 20 classes + “background”).
Shelhamer et al. (2016) proposed the FCN ("Fully Convolutional Network") that uses a pre-trained classification network (e.g. VGG 16 layers).

The fully connected layers are converted to $1 \times 1$ convolutional filters, and the final one retrained for 21 output channels (VOC 20 classes + “background”).

Since VGG16 has 5 max-pooling with $2 \times 2$ kernels, with proper padding, the output is $1/2^5 = 1/32$ the size of the input.

This map is then up-scaled with a de-convolution layer with kernel $64 \times 64$ and stride $32 \times 32$ to get a final map of same size as the input image.
Shelhamer et al. (2016) proposed the FCN ("Fully Convolutional Network") that uses a pre-trained classification network (e.g. VGG 16 layers).

The fully connected layers are converted to $1 \times 1$ convolutional filters, and the final one retrained for 21 output channels (VOC 20 classes + “background”).

Since VGG16 has 5 max-pooling with $2 \times 2$ kernels, with proper padding, the output is $1/2^5 = 1/32$ the size of the input.

This map is then up-scaled with a de-convolution layer with kernel $64 \times 64$ and stride $32 \times 32$ to get a final map of same size as the input image.

Training is achieved with full images and pixel-wise cross-entropy, starting with a pre-trained VGG16. All layers are fine-tuned, although fixing the up-scaling de-convolution to bilinear does as well.
VGG without its last layer

2 × conv/relu + maxpool

2 × conv/relu + maxpool

3 × conv/relu + maxpool

3 × conv/relu + maxpool

3 × conv/relu + maxpool

2 × fc-conv/relu

3d

\( \frac{1}{2} \) 64d

\( \frac{1}{4} \) 128d

\( \frac{1}{8} \) 256d

\( \frac{1}{16} \) 512d

\( \frac{1}{32} \) 512d

\( \frac{1}{32} \) 4096d
$2 \times \text{conv/relu} + \text{maxpool}$

$\frac{1}{2} \times 64d$

$2 \times \text{conv/relu} + \text{maxpool}$

$\frac{1}{4} \times 128d$

$3 \times \text{conv/relu} + \text{maxpool}$

$\frac{1}{8} \times 256d$

$3 \times \text{conv/relu} + \text{maxpool}$

$\frac{1}{16} \times 512d$

$3 \times \text{conv/relu} + \text{maxpool}$

$\frac{1}{32} \times 512d$

$2 \times \text{fc-conv/relu}$

$\frac{1}{32} \times 4096d$

$\text{fc-conv}$

$\frac{1}{32} \times 21d$

$\text{deconv } \times 32$

$3d$

$21d$
Although the FCN achieved almost state-of-the-art results when published, its main weakness is the coarseness of the signal from which the final output is produced ($\frac{1}{32}$ of the original resolution).

Shelhamer et al. proposed an additional element, that consists of using the same prediction/up-scaling from intermediate layers of the VGG network.
Fully convolutional networks improve performance on PASCAL. The left column is the best network from Shelhamer et al. (2016). Here we detail a relationship between momentum and batch size that motivates heavy learning.

Table 8

<table>
<thead>
<tr>
<th>Mask Type</th>
<th>Mean IU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td>84.8</td>
</tr>
<tr>
<td>Reference-BG</td>
<td>19.8</td>
</tr>
<tr>
<td>FG-only keep</td>
<td>76.1</td>
</tr>
<tr>
<td>BG-only mask</td>
<td>37.8</td>
</tr>
<tr>
<td>FG BG</td>
<td>mean IU</td>
</tr>
</tbody>
</table>

Left column is the best network from Shelhamer et al. (2016).
Results with a network trained from mask only (Shelhamer et al., 2016).
The most sophisticated object detection methods achieve **instance segmentation** and estimate a segmentation mask per detected object.

Mask R-CNN (He et al., 2017) adds a branch to the Faster R-CNN model to estimate a mask for each detected region of interest.

![Diagram of Mask R-CNN](image)

Figure 1. The **Mask R-CNN** framework for instance segmentation.

(He et al., 2017)
Figure 5. More results of **Mask R-CNN** on COCO test images, using ResNet-101-FPN and running at 5 fps, with 35.7 mask AP (Table 1). (He et al., 2017)
It is noteworthy that for detection and semantic segmentation, there is an heavy re-use of large networks trained for classification.

The models themselves, as much as the source code of the algorithm that produced them, or the training data, are generic and re-usable assets.
The end
References
