Lab 4: Semantic segmentation pre-report

CHI YEONG HEO[1],
[1]School of Computer Science and Engineering, Pusan National University, Busan 46241 Republic of Korea

1. Introduction

This report provides a summary of papers in the field of sematic segmentation: Fully Convolutional Networks for Semantic Segmentation[10] and Learning Deconvolution Network for Semantic Segmentation[12].

2. Fully Convolutional Networks for Semantic Segmentation

2.1. Overview

Fully Convolutional Networks (FCNs) are networks that take arbitrary size and produce spatially dense, correspondingly-sized output, making the suitable for tasks requiring pixel-level accuracy, like semantic segmentation. The paper adapts image classification networks into FCNs by using only convolutional layers and fine-tuning these networks to perform segmentation. The paper also proposes a novel architecture that combines coarse information from deeper layers and finer information from shallower layers.

2.2. Fully Convolutional Networks

Convolutional, pooling layers and activation functions operate on local input regions, and depend only on *relative* spatial coordinates.

Writing $\mathbf{x}_{ij}$ for the data vector at location (i, j) in a particular layer, and $\mathbf{y}_{ij}$ for the following layer, these functions compute outputs $\mathbf{y}_{ij}$ by

$$\mathbf{y}_{ij} = f_{ks}\left(\{\mathbf{x}_{si+\delta i, sj+\delta j}\}_{0\le\delta i,\delta j\le k}\right)$$

where k is called the kernel size, s is the stride or subsampling factor, and f_{ks} determines the layer type: a matrix multiplication for convolution or average pooling, a spatial max for max pooling, or an elementwise nonlinearity for an activation function, and so on for other types of layers.

This functional form is maintained under composition, with kernel size and stride obeying the transformation rule

$$f_{ks} \circ g_{k's'} = (f \circ g)_{k'+(k-1)s', ss'}.$$

FCN is a network with only layers of this from.

2.2.1. Adapting classifiers to FCNs

Typical classification networks take fixed-sized inputs and produce nonspatial outputs (class labels) since their fully connected layers have fixed dimensions and discard spatial coordinates. However, these fully connected layers can be replaced by convolutional layers, making the whole network FCN, which can take input of any size and output classification maps.

2.2.2. Shift-and-stitch and filter rarefaction

Shift-and-Stitch: When a network downsamples its output by a factor of f, it produces coarser spatial outputs since each pixel in the output corresponds to a larger receptive field in the input. The shift-and-stitch trick refine these coarse outputs by "shifting" the input and "stitching" the outputs, generating a denser prediction map without interpolation. For each position (x, y) within the downsampling factor, the input is shifted horizontally and vertically by a few pixels. Each of these outputs are interlaced, aligning predictions with the centers of the receptive fields.

Filter rarefaction: Instead of shift-and-stitch, filter rarefaction adjusts the network's filters and strides to mimic the effect. The filters are "rarefied" to match the upscaled output without losing spatial alignment. The rarefied filter f'_{ij} is created by:

$$f'_{ij} = \begin{cases} f_{i/s, j/s} & \text{if } s \text{ divides both } i \text{ and } j; \\ 0 & \text{otherwise,} \end{cases}$$

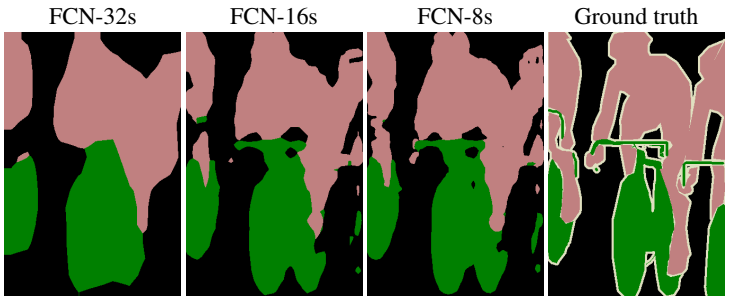

Figure 1: Refining fully convolutional nets by fusing information from layers with different strides improves segmentation detail. The first three images show the output from our 32, 16, and 8 pixel stride nets (see Figure 2).

2.2.3. Deconvolution

Deconvolution upsamples input by reversing the forward and backward passes of convolution. It can be performed in-network for end-to-end learning by backpropagation form the pixelwise loss. This method with skip layer fusion is more effective and efficient than shift-and-stitch and filter rarefaction.

2.3. Segmentation Architecture

This paper proposes a new FCN for segmentation that combines layers of the feature hierarchy and refines the spatial precision of the output. (See Figure 2).

Even though original classifiers can be fine-tuned to segmentation in FCN framework, their output is too coarse for pixel-wise task (see Figure 1).

The authors create skip connections that link the final prediction layer with intermediate layers that retain finer spatial resolution. These connections allow the model to incorporate localized features from shallower layers and contextual information from deeper layers, forming a directed acyclic graph (DAG) instead of a linear topology.

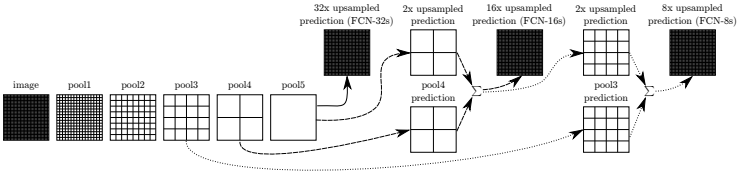

Figure 2: Only pooling and prediction layers are shown; Solid line (FCN-32s): upsamples stride 32 predictions back to pixels in a single step. Dashed line (FCN-16s): Combining predictions from both the final layer and the `pool4` layer, at stride 16, lets the net predict finer details, while retaining high-level semantic information. Dotted line (FCN-8s): Additional predictions from `pool3`, at stride 8, provide further precision.

2.4. Comparison

Table 1: FCN-8s gives a 20% relative improvement over the state-of-the-art on the PASCAL VOC 2011 and 2012 test sets, and reduces inference time.

	mean IU VOC2011 test	mean IU VOC2012 test	inference time
R-CNN [5]	47.9	-	-
SDS [6]	52.6	51.6	$\sim$ 50 s
FCN-8s	**62.7**	**62.2**	$\sim$ **175 ms**

3. Learning Deconvolution Network for Semantic Segmentation

3.1. Overview

This paper proposes a novel semantic segmentation algorithm by learning a deconvolution network. It applied to each proposal of an input image, and combines the results from all proposals to construct the final sematic segmentation map. Integrating deep deconvolution network and proposal-wise prediction mitigates the limitations of the prior methods based on FCNs.

3.2. System Architecture

The network is composed of two parts-convolution and deconvolution networks. The convolution network is a feature extractor that transforms the input to multidimensional feature. The deconvolution network is a shape generator that produces object segmentation from the feature from convolution network. The final output is a probability map in the same size to input image, indicating probability of each pixel that belongs to one of the classes.

The convolution network is constructed with VGG-16 network [14], without last classification layer. The deconvolution network is a mirrored version of the convolution network, and has multiple series of unpooling, deconvolution, and rectification layers.

3.3. Deconvolution Network for Segmentation

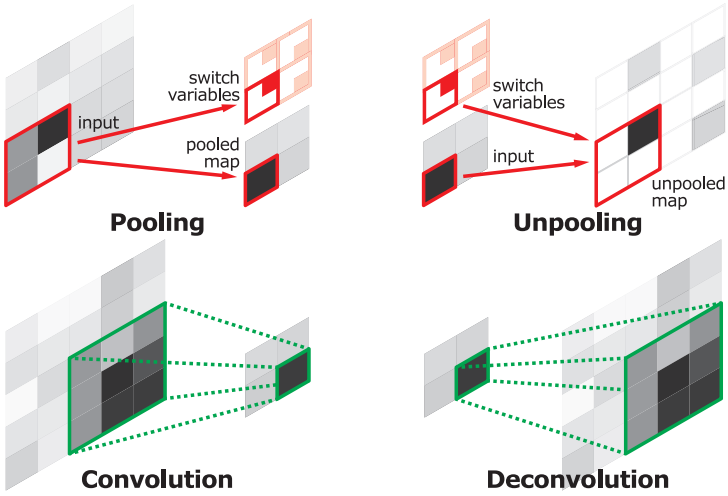

Figure 3: Illustration of deconvolution and unpooling operations.

3.3.1. Unpooling

Pooling layer is used to filter noisy activations in a lower layer by abstracting activations in a receptive field with a single value. However, the spatial information within a receptive field is lost during pooling, which may be critical for precise localization that is required for semantic segmentation.

Unpooling layers in deconvolution network performs the reverse operation of pooling and reconstruct the original size of activations. For the unpooling operation, the locations of activations selected during pooling are recorded in switch variables and used to place each activation back to its original location, as illustrated in Figure 3.

3.3.2. Deconvolution

The deconvolution layers densify the spare activations obtained by unpooling. A single input activation generates multiple outputs, as illustrated in Figure 3. The enlarged activation map is cropped to keep its size identical to the preceding unpooling layer.

The filters in lower layers capture overall shape of an object while higher layers capture the class-specific fine-details. This hierarchical structure allows the network takes class-specific shape information for semantic segmentation.

3.4. Instance-wise Segmentation

This algorithm addresses semantic segmentation by processing individual image proposals, each potentially containing an object instance. Outputs from these proposals are aggregated to form a complete segmentation map in the original image space, effectively capturing objects at multiple scales and fine details that fixed-size receptive field approaches often miss. This method reduces the prediction search space, easing training complexity and memory demands.

3.4.1. Aggregating Instance-wise Segmentation Map

To counter segmentation errors from object misalignment or background clutter, noise is suppressed during aggregation. The pixel-wise maximum or average across score maps for each class creates robust segmentation results:

$$P(x,y,c) = \max_i G_i(x,y,c), \quad \forall i, \tag{1}$$

or

$$P(x,y,c) = \sum_i G_i(x,y,c), \quad \forall i. \tag{2}$$

Applying softmax to the aggregated maps yields class-conditional probabilities, which are refined using a fully connected CRF [9] for final pixel-wise labeling.

3.5. Training

The network contains a lot of parameters since the network is twice deeper than VGG-16. The number of training examples for semantic segmentation is relatively small than the size of the network. Training a deep network with a limited number of examples is not trivial.

Batch Normalization [8]: A batch normalization layer is added to the output of every convolutional and deconvolutional layer. As batch normalization reduce the internal-covariate-shift by normalizing input distributions to the standard Gaussian distribution, it is critical to optimize the network.

Two-stage Training: The network is trained with easy examples first and fine-tuned with more challenging examples later. To construct training examples for the first stage training, the object instances are cropped using ground-truth annotations so that an object is centered at the bounding box. It limits the variations in object location and size, reducing the search space for semantic segmentation. In the second stage, object proposals are used to construct more challenging examples. It makes the network robust to the misalignment of proposals in testing, but makes training more hard as the location and scale of an object may be significantly different across training examples.

3.6. Comparison

The network's performance is evaluated on the PASCAL VOC 2012 benchmark [4], which includes 1,456 test images across 20 object categories. For evaluation, the comp6 protocol is adopted, which assesses segmentation accuracy based on the Intersection over Union (IoU) between predicted and ground-truth segmentations. The network achieves state-of-the-art performance on the PASCAL VOC 2012 segmentation benchmark, outperforming other methods trained without the use of external data.

Table 2: Evaluation results on PASCAL VOC 2012 test set. Only a subset of classes (aero, bike, bird, boat, and bottle) are shown due to space constraints. (asterisk (∗) denotes algorithms trained with additional data.)

Method	areo	bike	bird	boat	bottle	mean
Hypercolumn [7]	68.4	27.2	68.2	47.6	61.7	59.2
MSRA-CFM [2]	75.7	26.7	69.5	48.8	65.6	61.8
FCN8s [10]	76.8	34.2	68.9	49.4	60.3	62.2
TTI-Zoomout-16 [11]	81.9	35.1	78.2	57.4	56.5	64.4
DeepLab-CRF [1]	84.4	**54.5**	81.5	63.6	65.9	71.6
DeconvNet	85.9	42.6	78.9	62.5	66.6	69.6
DeconvNet+CRF	87.8	41.9	80.6	63.9	67.3	70.5
EDeconvNet	88.4	39.7	79.0	63.0	67.7	71.7
EDeconvNet+CRF	**89.9**	39.3	79.7	63.9	**68.2**	**72.5**
* WSSL [13]	85.3	36.2	**84.8**	61.2	67.5	70.4
* BoxSup [3]	86.4	35.5	79.7	**65.2**	65.2	71.0

[1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs, 2016. URL https://arxiv.org/abs/1412.7062.

[2] Jifeng Dai, Kaiming He, and Jian Sun. Convolutional feature masking for joint object and stuff segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2015. doi: 10.1109/cvpr.2015.7299025. URL http://dx.doi.org/10.1109/CVPR.2015.7299025.

[3] Jifeng Dai, Kaiming He, and Jian Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation, 2015. URL https://arxiv.org/abs/1503.01640.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[5] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, 2014. URL https://arxiv.org/abs/1311.2524.

[6] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation, 2014. URL https://arxiv.org/abs/1407.1808.

[7] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization, 2015. URL https://arxiv.org/abs/1411.5752.

[8] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015. URL https://arxiv.org/abs/1502.03167.

[9] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials, 2012. URL https://arxiv.org/abs/1210.5644.

[10] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation, 2015. URL https://arxiv.org/abs/1411.4038.

[11] Mohammadreza Mostajabi, Payman Yadollahpour, and Gregory Shakhnarovich. Feedforward semantic segmentation with zoom-out features, 2014. URL https://arxiv.org/abs/1412.0774.

[12] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation, 2015. URL https://arxiv.org/abs/1505.04366.

[13] George Papandreou, Liang-Chieh Chen, Kevin Murphy, and Alan L. Yuille. Weakly- and semi-supervised learning of a dcnn for semantic image segmentation, 2015. URL https://arxiv.org/abs/1502.02734.

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. URL https://arxiv.org/abs/1409.1556.