Week6 Pre-Report: Overview of Semantic Segmentation - FCN and DilatedNet

Euijin Hong[1]
[1]Department of Electrical and Electronic Engineering, Yonsei University.

1 Introduction

Semantic segmentation is another challenge in computer vision along image classification. FCN[10] purposed a new model architecture of modifying and fine-tuning the existing image classification model(AlexNet[7], VGGNET[13], and GoogLeNet[14]), replacing the final fully-connected layers to convolutional layers, thus maintaining the positional information inferenced by former layers, learning end-to-end, pixels-to-pixels and whole-image-at-a-time. Also, they introduced the skip architecture, which combines semantic information from a deep & coarse layer with appearance information from a shallow & fine layer. By doing so, FCN could exceed the state-of-the-art in semantic segmentation at that time.

However, dense prediction problem (including semantic segmentation) was structurally different with image classification, but FCN was using almost the identical network as that used in image classification, without recognizing what aspects are truly necessary for the task. Also, the translation invariant property of convolutional layer turned out to be ineffective in dense prediction tasks. In DilatedNet[17], the redundant structures are discovered and removed from the network simplified as a frontend, and a context module introducing *dilated convolution* for systematically aggregating the multiscale contextual information without losing resolution, and exponentially expanding the receptive field without loss of resolution or coverage.

2 Fully Convolutional Networks(FCN)

2.1 Theory of Fully Convolutional Networks

In FCN, real-valued loss function defines a task, while SGD on l omputed over whole images is identical to SGD on l', having the entire final layer receptive fields as a minibatch. Here, feedforward progress and backpropagation are much more efficient when computing an entire image layer-by-layer than independently calculating patch-by-patch.

Adapting classifiers for dense prediction: The fully connected layers of existing classification models can be replaced as convolutional layers that cover the entire input regions, which outputs the spatial maps as classification maps. Also, the computation is highly reduced by overlapping regions of particular input patches, both for forward and backward passes. As the layer goes deeper, the dimension is reduced by subsampling, which coarsens the output.

Shift-and-stitch as filter rarefaction: Coarse outputs can yield dense predictions by stitching the shifted versions of the input together to form an output. This processes the downsampled output with factor of f, shifting the input horizontally x and vertically y, where $0 \leq x, y < f$. To prevent the computational cost to increase by f^2, we can use à trous algorithm[11], as following equation.

$$f'_{ij} \begin{cases} f_{i/s,j/s} & \text{if s divides both i and j;} \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

Decreasing subsampling has tradeoff between fineness of information and size of receptive field. By experiment, researchers choose not to use the trick in the model, using skip layer fusion instead.

Upsampling with deconvolution: Deconvolution with output stride of f upsamples the input with factor f, implementing in-network upsampling. The deconvolution filter can learn nonlinear upsampling.

Patchwise training as loss sampling: The efficiency of patchwise training and fully convolutional training depends on overlap and minibatch size. When each batch contains all the receptive fields of the units under the loss of an image, whole image fully convolutional training is identical to patchwise training. However, patchwise sampling is not used in the model since it does not give better results than whole image training.

2.2 Model Configuration of FCN

FCN is made from modifying and fine-tuning the following classifiers: AlexNet[7], VGG16[13] model, and the final loss layer of GoogLeNet[14] without the final layer. All classifiers are modified by converting the FC layers to convolution layers, appending 1×1 convolution layer for changing the channel length to number of output classes, and adding a deconvolution layer for bilinear upsampling coarse outputs to pixel-dense outputs.
Figure 1. shows the overall structure of FCN.

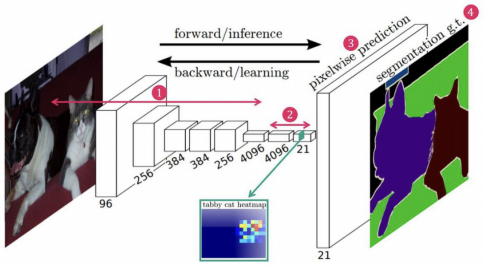

Figure 1: General structure of an FCN model.

Then, we combine the semantic and locational prediction by DAG, having the edges that skip ahead from lower layers to higher ones. For instance, in FCN-16s, the output stride is divided by half, by predicting from a 16 pixel stride layer. Then, 1×1 convolution layer is added on top of `pool4`. Then we add the output with `conv7` predictions upsampled by $2\times$ upsampling layer, and upsample with factor $f = 16$. There were other attempts to refine predictions such as decreasing the stride of pooling layers, but skipping and layer fusion method was the most effective one. The detailed structure of DAG net (or skip FCN) is shown in Figure 2.

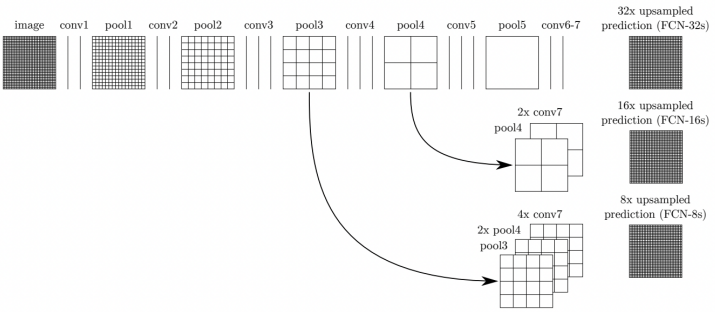

Figure 2: Detailed structure of DAG net of an FCN model.

2.3 Evalutation

The experimental framework was chosen to use SGD with momentum, fine-tuning with backpropagation, with using whole image training, without class balancing and augmentation, and initializing intermediate deconvolution layers as bilinear upsampling while fixing the final one. The metrics used are pixel accuracy, mean accuracy, mean IU, and frequency weighted IU.

By following the framework mentioned above, several evaluations of the model were undergone. As shown in Figure 3-(left), FCN-VGG16 achieved performance at 56.0 mean IU on validation set, while in Figure 3-(right) shows FCN-8s model yields the best result.

	FCN-AlexNet	FCN-VGG16	FCN-GoogLeNet[4]
mean IU	39.8	**56.0**	42.5
forward time	50 ms	210 ms	59 ms
conv. layers	8	16	22
parameters	57M	134M	6M
rf size	355	404	907
max stride	32	32	32

	pixel acc.	mean acc.	mean IU	f.w. IU
FCN-32s-fixed	83.0	59.7	45.4	72.0
FCN-32s	89.1	73.3	59.4	81.4
FCN-16s	90.0	75.7	62.4	83.0
FCN-8s	**90.3**	**75.9**	**62.7**	**83.2**

Figure 3: (left): Comparison of performance between FCN based on different classifier models (right): Comparison of performance of skip FCNs

Furthermore, researchers compared the FCN-8s model with previous state-of-the-art model R-CNN[3] and SDS[6], giving about 20% relative improvement. Also, the researchers created a two-stream net by "late-fusion" of RGB and HHA and compared it with existing model[4]. The result shows that the accuracy of FCN-16s-RGB-HHA model exceeds that of Gupta *et*

This is an extended abstract. The full paper is available at the Computer Vision Foundation webpage.

al[4]. Moreover, FCN also showed state-of-the-art performance in SIFT Flow dataset, compared with models of Liu *et al*[9], Tighe *et al*[15, 16], Farabet *et al*[2], and Pinheiro *et al*[12].

3 DilatedNet

3.1 Background

In dense prediction tasks such as semantic segmentation, pixelwise inference has to be done. In classification nets and FCN, the convolutional layers were translation invariant, which gives predictions regardless of the location in the feature map thus results coarse predictions in terms of spatial resolution. However, in tasks such as semantic segmentation or object detection, localization of the object is crucial to produce fine outputs with high accuracy. Therefore, translation invariant hampers the performance of the model, thus attempts to remove the property became necessary. Also, figuring out what structures are actually crucial in FCN to perform dense prediction has become an important issue in order to improve the accuracy further.

3.2 Theory and Architecture

Dilated Convolution: The key idea of DilatedNet is usage of *dilated convolution*, which does not make loss of resolution nor requires additional parameters, but expands the receptive field. The equation (2) is a mathematical expression of dilated convolution $*_l$, where $F : \mathbb{Z}^2 \to \mathbb{R}$ is a discrete function, let $\Omega_r = [-r,r]^2 \cap \mathbb{Z}^2$, $k : \Omega_r \to \mathbb{R}$ a discrete filter with size $(2r+1)^2$, and l to be a dilation factor.

$$(F *_l k)(p) = \sum_{s+lt=p} F(s)k(t) \tag{2}$$

Instead of creating a separate "dilated filter", dilated convolution operators can apply the same filter at different ranges by changing dilation factors. Although in FCN[10], dilation of filters (shift-and-stitch) were considered but not used, in DilatedNet it is systematically used as dilated convolution.

Dilated convolution can exponentially expand the receptive fields without giving up resolution or coverage, as we can see in Figure 4, where the receptive field increases exponentially while the number of parameters increases linearly.

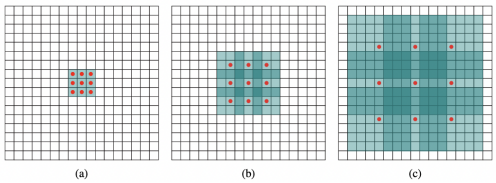

Figure 4: (a) F_1 produced by F_0 from 1-dilated convolutional filter, with receptive field size of 3×3. (b) F_2 produced by F_1 from 2-dilated convolutional filter, having receptive field size of 7×7. (c) F_3 produced by F_2 from 4-dilated convolutional filter, with receptive field 15×15.

By using dilated convolution, we can expand the receptive field without using maxpool or stride-2 convolutional layer, thus can preserve the spatial information and maintain the calculation amount.

Multi-scale Context Aggregation: DilatedNet also proposes a *context module*. The context module aggregates the multi-scale contextual information and improves the performance of dense prediction. The number of channels of input feature map and output feature map identical as C since each layer has C channels, which allows the context module to be used in various dense prediction network architectures even if feature maps are not normalized or no loss function is defined. The basic context modlue consists of 7 $3 \times 3 \times C$ convolution layers with different dilation factors followed by a final $1 \times 1 \times C$ convolution layer that produces the output of the model, as depicted in Figure 5. Truncation, which means the usage of activation function after the convolution layer, is applied for every dilated convolution layers except the final one.

Layer	1	2	3	4	5	6	7	8
Convolution	3×3	3×3	3×3	3×3	3×3	3×3	3×3	1×1
Dilation	1	1	2	4	8	16	1	1
Truncation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Receptive field	3×3	5×5	9×9	17×17	33×33	65×65	67×67	67×67
Output channels								
Basic	C	C	C	C	C	C	C	C
Large	$2C$	$2C$	$4C$	$8C$	$16C$	$32C$	$32C$	C

Figure 5: Context network architecture aggregating contextual information with continuously increasing the scale, while maintaining resolution and processing C feature map.

The initialization of context module was a newly designed form of identity initialization by following the work in recurrent networks [8], which can be expressed as Equation 3, where a is an index of the input feature map and b is an index of output map. Researchers found that backpropagation in this initialization reliably harvests a contextual information provided by the network.

$$k^b(t,a) = 1_{[t=0]} 1_([a=b]) \tag{3}$$

Moreover, larger context module using larger number of feature maps in deeper layers is introduced as shown in Figure 5. The initialization scheme is based on difference between number of feature maps in different layers as in Equation 4, where ε N$(0,\sigma^2)$ and $\sigma << C/c_{i+1}$.

$$k^b(t,a) = \begin{cases} \frac{C}{c_{i+1}} & t = 0 \text{ and } [\frac{aC}{c_i}] = [\frac{bC}{c_i}] \\ \varepsilon & \text{otherwise} \end{cases} \tag{4}$$

Front End: The front end module is a sort of a backbone connected to the front of the context module for feature extraction. It receives a 3-channel input image and produces a 21-channel output feature map which serves as an input of the context module. It uses a basic structure of the FCN[10][1] based on VGG16 net[13]. However, researchers removed the last two pooling and striding layers thus making the output dimension 64×64. Also, they replaced the convolutional layers of *conv5 block* into 2-dilated convolution layer, and the *fc1* layer into 4-dilated convolution layer. This produces high-resolution output, even though initialization can be done with parameters of original classification network. Reflection padding(filling the buffer zone by reflecting the image of each edge) is also used, while paddings of the intermediate feature map was removed.

3.3 Evalutation

Researchers first evlauated the performance of their front end module, by comparing it with results of FCN-8s[10] and DeepLab.[1] The module was trained under Pascal VOC2012 training set augmented as [5], with SGD, mini-batch size 14, learning rate 10^{-3}, and momentum 0.9 for 60k iterations. Furthermore, they compared the test results of the plain front end model, context model-added network, and CRF/RNN-added network. The test result is shown in Figure 6.

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean IoU
FCN-8s	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1	62.2
DeepLab	72	31	71.2	53.7	60.5	77	71.9	73.1	25.2	62.6	49.1	68.7	63.3	73.9	73.6	50.8	72.3	42.1	67.9	52.6	62.1
DeepLab-Msc	74.9	34.1	72.6	52.9	61.0	77.9	73.0	73.7	26.4	62.2	49.3	68.4	64.1	74.0	75.0	51.7	72.7	42.5	67.2	55.7	62.9
Our front end	**82.2**	**37.4**	**72.7**	**57.1**	**62.7**	**82.8**	**77.8**	**78.9**	**28**	**70**	**51.6**	**73.1**	**72.8**	**81.5**	**79.1**	**56.6**	**77.1**	**49.9**	**75.3**	**60.9**	**67.6**

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean IoU
Front end	86.3	38.2	76.8	**66.8**	63.2	87.3	78.7	82	33.7	76.7	53.5	73.7	76	76.6	83	**51.9**	77.8	44	79.9	**66.3**	69.8
Front + Basic	86.4	37.6	78.5	66.3	64.1	89.9	79.9	84.9	**36.1**	79.4	**55.8**	77.6	81.6	79	83.1	51.2	81.3	43.7	82.3	65.7	71.3
Front + Large	**87.3**	**39.2**	**80.3**	65.6	**66.4**	**90.2**	**82.6**	**85.8**	34.8	**81.9**	51.7	**79**	**84.1**	**80.9**	**83.2**	51.2	**83.2**	**44.7**	**83.4**	65.6	**72.1**
Front end + CRF	89.2	38.8	80	**69.8**	63.2	88.8	80	85.2	33.8	80.6	55.5	77.1	80.8	77.3	84.3	**53.1**	80.4	45	80.7	**67.9**	71.6
Front + Basic + CRF	89.1	38.7	81.4	67.4	65	91	81	86.7	**37.5**	81	**57**	79.6	83.6	79.9	**84.6**	52.7	83.3	44.3	82.6	67.2	72.7
Front + Large + CRF	**89.6**	**39.9**	**82.7**	66.7	**67.5**	**91.1**	**83.3**	**87.4**	36	**83.3**	52.5	**80.7**	**85.7**	**81.8**	84.4	52.6	**84.4**	**45.3**	**83.7**	66.7	**73.3**
Front end + RNN	88.8	38.1	80.8	**69.1**	65.6	89.9	79.6	85.7	36.3	83.6	57.3	77.9	83.2	77	**84.6**	**54.7**	82.1	**46.9**	80.9	66.7	72.5
Front + Basic + RNN	89	38.4	82.3	67.9	65.2	91.5	80.4	87.2	**38.4**	82.1	**57.7**	79.9	85	79.6	84.5	53.5	84	45	82.8	66.2	73.1
Front + Large + RNN	**89.3**	**39.2**	**83.6**	67.2	**69**	**92.1**	**83.1**	**88**	**38.4**	**84.8**	55.3	**81.2**	**86.7**	**81.3**	84.3	53.6	**84.4**	45.8	**83.8**	**67**	**73.9**

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean IoU
DeepLab++	89.1	38.3	88.1	63.3	69.7	87.1	**83.1**	85	29.3	76.5	56.5	79.8	77.9	85.8	82.4	57.4	84.3	54.9	80.5	64.1	72.7
DeepLab-MSc++	89.2	46.7	88.5	63.5	68.4	87.0	81.2	86.3	32.6	80.7	62.4	81.0	81.3	84.3	82.1	56.2	84.6	58.3	76.2	67.2	73.9
CRF-RNN	90.4	**55.3**	88.7	**68.4**	69.8	88.3	82.4	85.1	32.6	78.5	**64.4**	79.6	81.9	**86.4**	81.8	**58.6**	82.4	53.5	77.4	**70.1**	74.7
Front end	86.6	37.3	84.9	62.4	67.3	86.2	81.2	82.1	32.6	77.4	58.3	75.9	81	83.6	82.3	54.2	81.5	50.1	77.5	63	71.3
Context	89.1	39.1	86.8	62.6	68.9	88.2	82.6	87.7	33.8	81.2	59.2	81.8	87.2	83.3	83.6	53.6	84.9	53.7	80.5	62.9	73.5
Context + CRF	91.3	39.9	**88.9**	64.3	69.8	88.9	82.6	89.7	34.7	82.7	59.5	83	88.4	84.2	85	55.3	86.7	54.4	**81.9**	63.6	74.7
Context + CRF-RNN	**91.7**	39.6	87.8	63.1	**71.8**	**89.7**	82.9	**89.8**	**37.2**	**84**	63	**83.3**	**89**	83.8	**85.1**	56.8	**87.6**	**56**	80.2	64.7	**75.3**

Figure 6: (top): Comparison of semantic segmentation results of FCN-8s, DeepLab, and front-end. (middle): Semantic segmentation performances of front-end, front-end + context module(basic/large), and front-end + context module + CRF / RNN (bottom): Comparison of results of 'DeepLab++' (DeepLab-CRF-COCOLargeFOV), 'DeepLab-MSc++' (DeepLab-MSc-CRF-LargeFOV-COCO-CrossJoint)[1], CRF-RNN[18], and large context module with and without CRF-RNN.

4 Conclusion

In conclusion, the two models respectively exceeded the former state-of-the-art results in semantic segmentation(for DilatedNet, one of them were FCNs), by introducing new model structures. In FCN, it was basically for convolutional layers replacing the fully connected layers of classification model, and skip architectures introduced for achieving finer results. In DilatedNet, it was applying additional context module functioning with dilated convolution while removing pooling or expanding layers, and optimizing the model by refining the frontend FCN model structure.

[1] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L Yuille. Attention to scale: Scale-aware semantic image segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3640–3649, 2016.

[2] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2012.

[3] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[4] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13*, pages 345–360. Springer, 2014.

[5] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *2011 international conference on computer vision*, pages 991–998. IEEE, 2011.

[6] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13*, pages 297–312. Springer, 2014.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[8] Quoc V Le, Navdeep Jaitly, and Geoffrey E Hinton. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*, 2015.

[9] Ce Liu, Jenny Yuen, and Antonio Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):978–994, 2010.

[10] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[11] Stéphane Mallat. *A wavelet tour of signal processing*. Elsevier, 1999.

[12] Pedro Pinheiro and Ronan Collobert. Recurrent convolutional neural networks for scene labeling. In *International conference on machine learning*, pages 82–90. PMLR, 2014.

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[14] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[15] Joseph Tighe and Svetlana Lazebnik. Finding things: Image parsing with regions and per-exemplar detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3001–3008, 2013.

[16] Joseph Tighe and Svetlana Lazebnik. Superparsing: scalable nonparametric image parsing with superpixels. *International Journal of Computer Vision*, 101:329–349, 2013.

[17] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[18] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537, 2015.