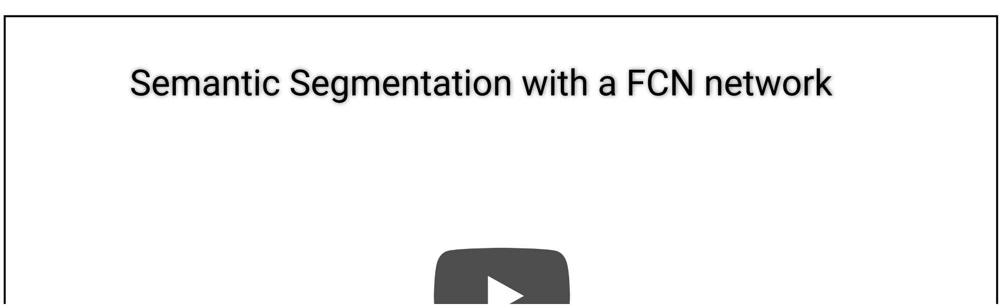
Review: FCN — Fully Convolutional Network (Semantic Segmentation)

Sik-Ho Tsang Oct 6, 2018 · 4 min read

In this story, **Fully Convolutional Network (FCN) for Semantic Segmentation** is briefly reviewed. Compared with classification and detection tasks, segmentation is a much more difficult task.

- **Image Classification**: Classify the object (Recognize the **object class**) within an image.
- **Object Detection**: Classify and detect the object(s) within an image with bounding box(es) bounded the object(s). That means we also need to know the **class, position and size of each object**.
- **Semantic Segmentation**: Classify the **object class for each pixel** within an image. That means there is **a label for each pixel**.

An example for semantic segmentation is as below:



An example of Semantic Segmentation

Original Image (Leftmost), Ground Truth Label Map (2nd Left), Predicted Label Map (2nd Right), Overlap Image and Predicted Label (Rightmost)

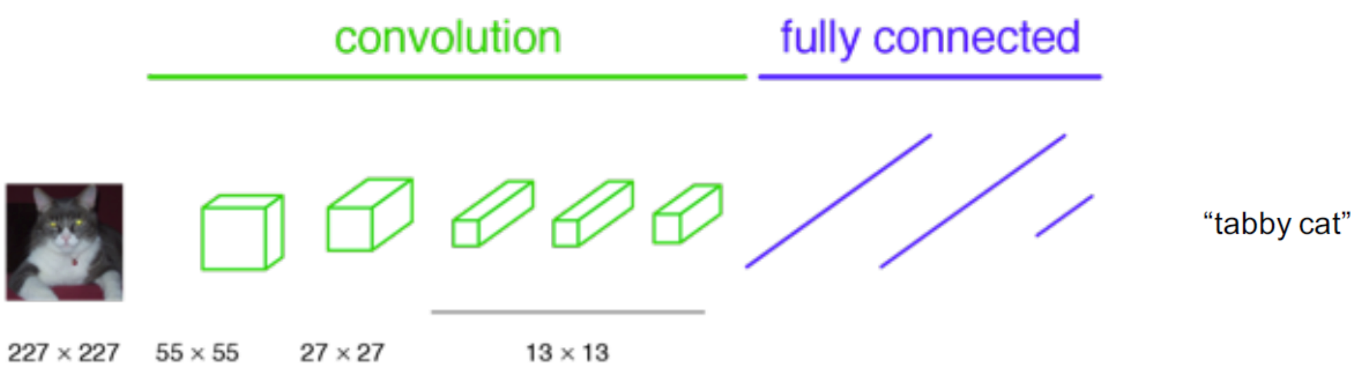
It has been published in **2015 CVPR** [1] and **2017 TPAMI** [2] with **citations more than 6000** while I was writing this story. Thus, it is also one of the most basic papers for semantic segmentation using FCN. (Sik-Ho Tsang @ Medium)

What Are Covered

1. **From Image Classification to Semantic Segmentation**
2. **Upsampling Via Deconvolution**
3. **Fusing the Output**

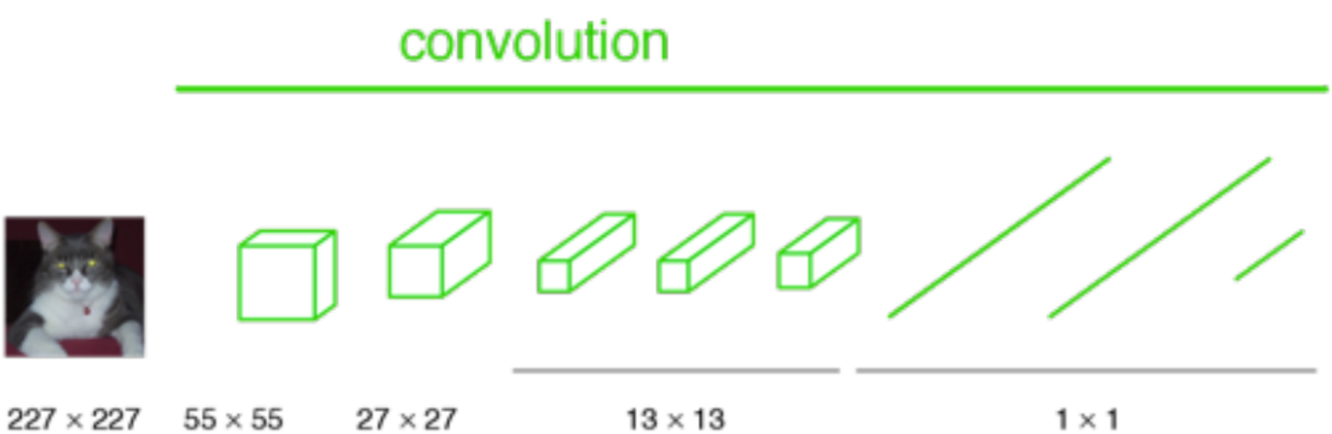
1. From Image Classification to Semantic Segmentation

In classification, conventionally, an input image is downsized and goes through the convolution layers and fully connected (FC) layers, and output one predicted label for the input image, as follows:



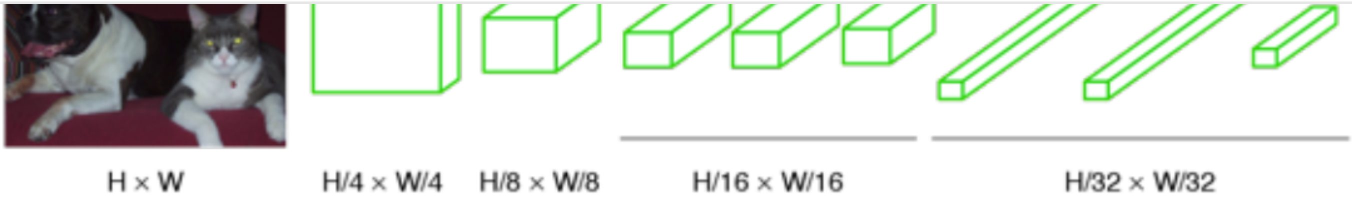
Classification

Imagine we turn the FC layers into 1×1 convolutional layers:



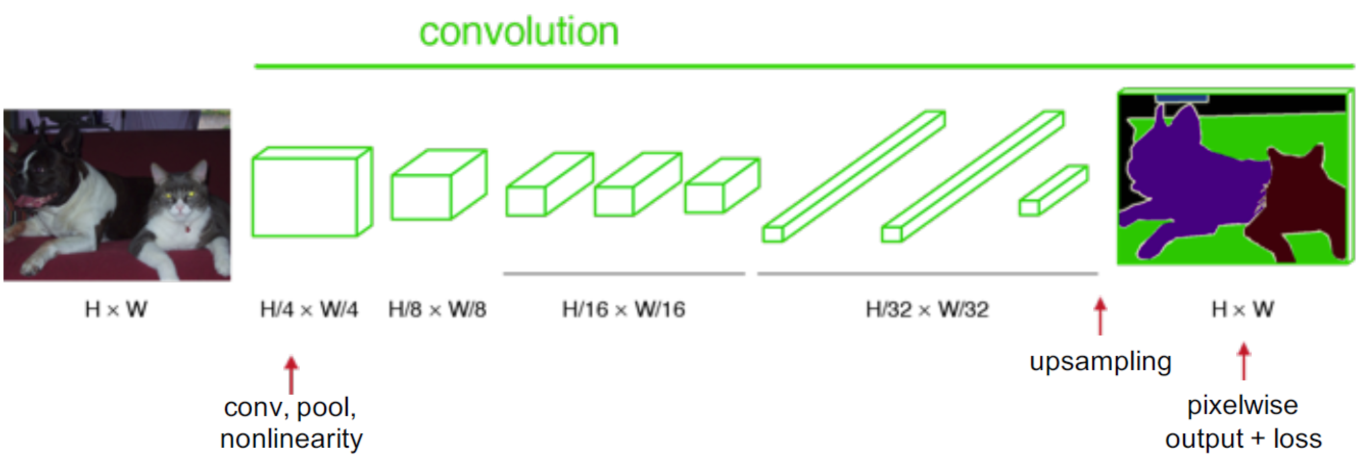
All layers are convolutional layers

And if the image is not downsized, the output will not be a single label. Instead, the output has a size smaller than the input image (due to the max pooling):

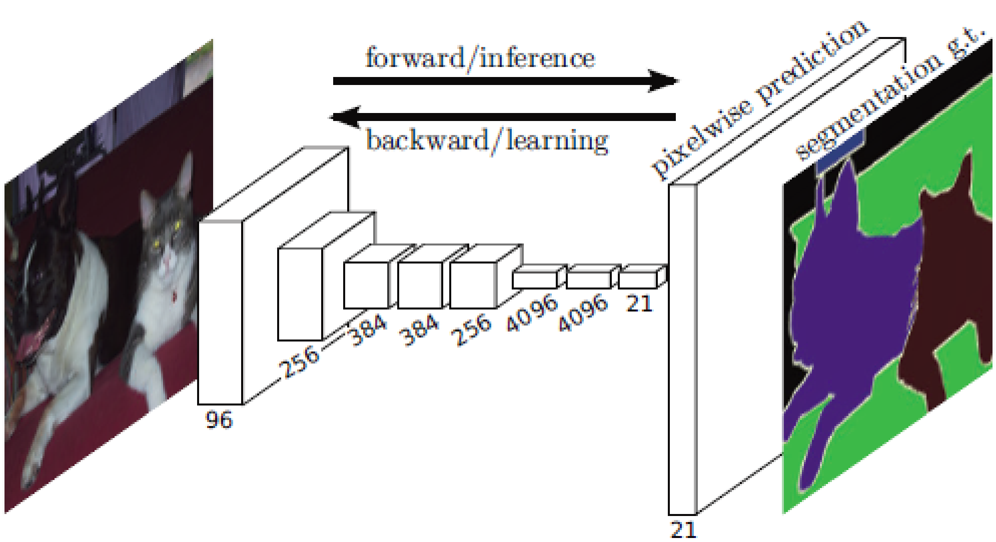


All layers are convolutional layers

If we upsample the output above, then we can calculate the pixelwise output (label map) as below:

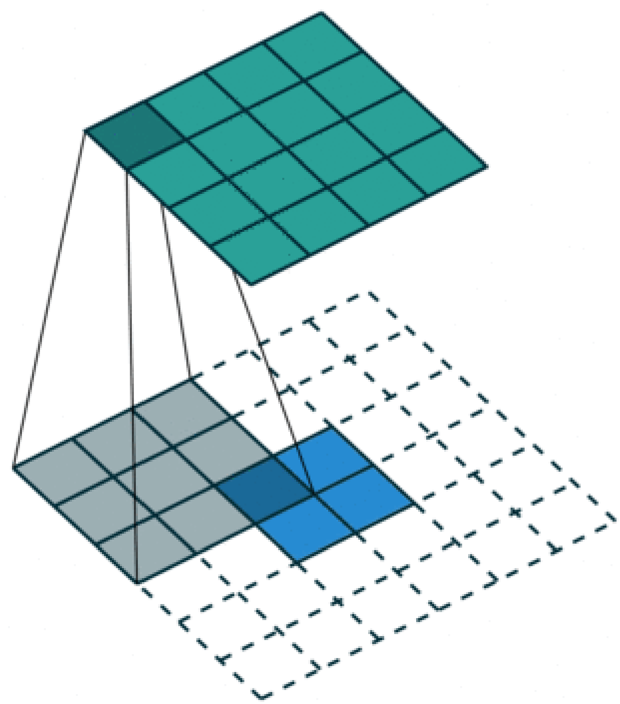


Upsampling at the last step



Feature Map / Filter Number Along Layers

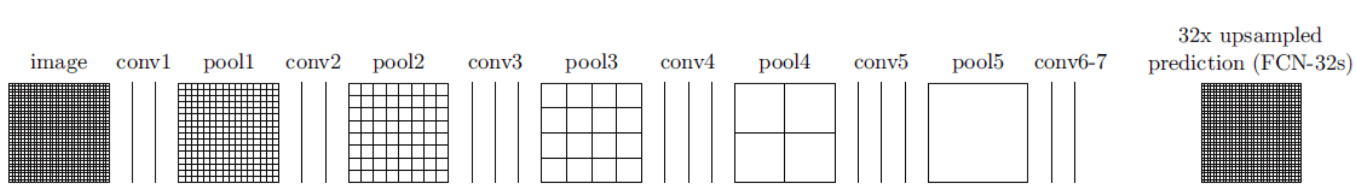
is coming from when we want to have upsampling to get the output size larger. (But the name, deconvolution, is misinterpreted as reverse process of convolution, but it is not.) And it is also called, **up convolution, and transposed convolution.** And it is also called **fractional stride convolution** when fractional stride is used.



Upsampling Via Deconvolution (Blue: Input, Green: Output)

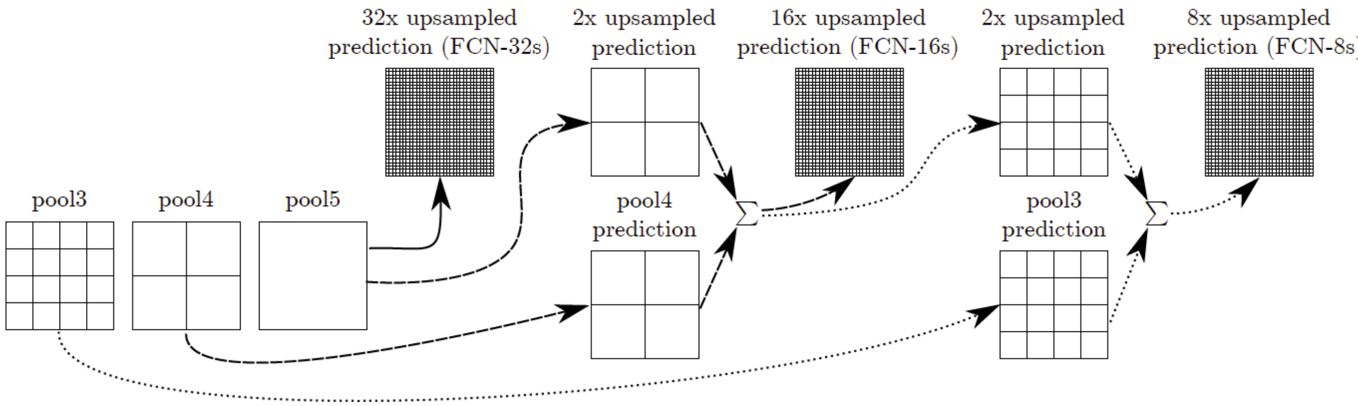
3. Fusing the Output

After going through conv7 as below, the output size is small, then 32× upsampling is done to make the output have the same size of input image. But it also makes the output label map rough. And it is called **FCN-32s**:



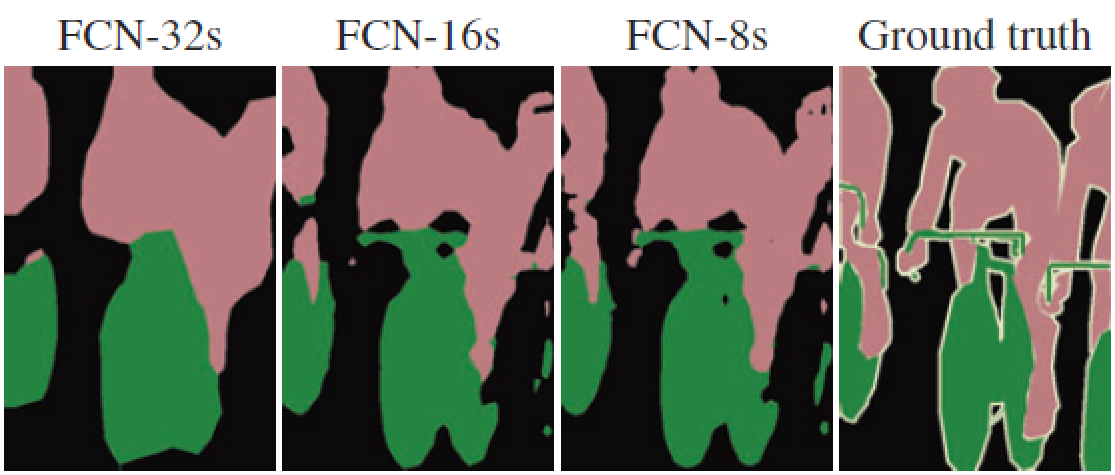
This is because, **deep features can be obtained when going deeper, spatial location information is also lost when going deeper.** That means output from shallower layers have more location information. If we combine both, we can enhance the result.

To combine, we **fuse the output (by element-wise addition)**:



Fusing for FCN-16s and FCN-8s

FCN-16s: The output from pool5 is 2× upsampled and fuse with pool4 and perform 16× upsampling. Similar operations for **FCN-8s** as in the figure above.



Comparison with different FCNs

FCN-32s result is very rough due to loss of location information while FCN-8s has the best result.

make the prediction more accurate. But in this case, it is done for each pixel, and they are added from the results of different layers within a model.

4. Results

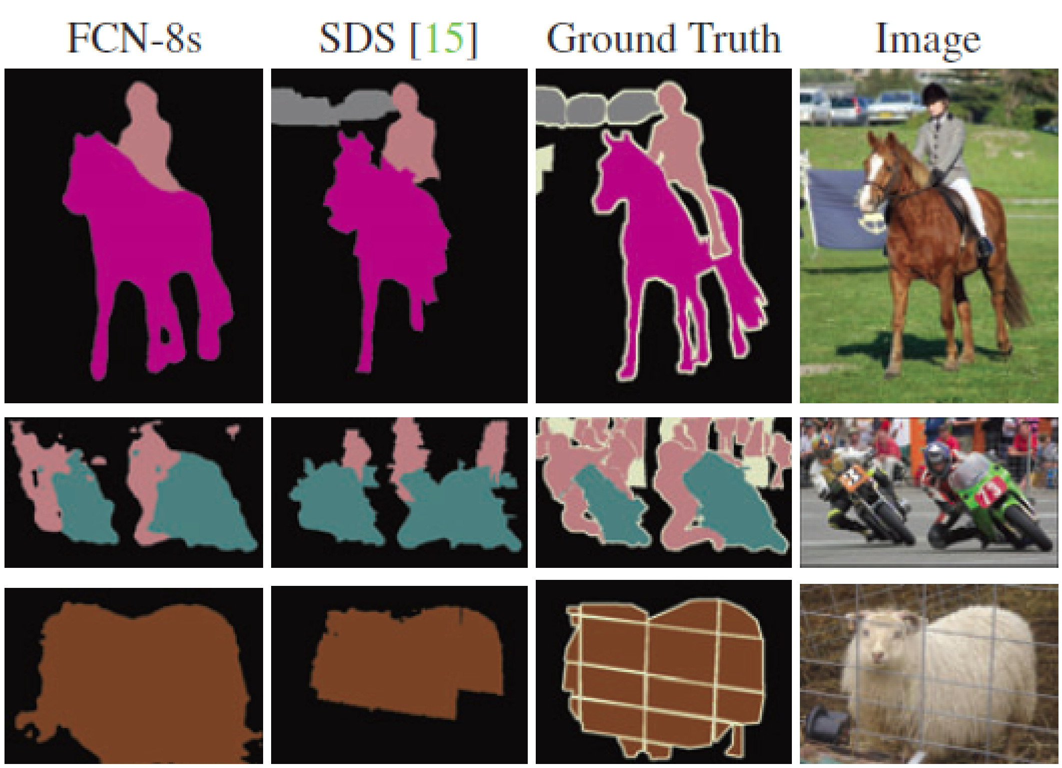
	pixel acc.	mean acc.	mean IU	f.w. IU
FCN-32s-fixed	83.0	59.7	45.4	72.0
FCN-32s	89.1	73.3	59.4	81.4
FCN-16s	90.0	75.7	62.4	83.0
FCN-8s	**90.3**	**75.9**	**62.7**	**83.2**

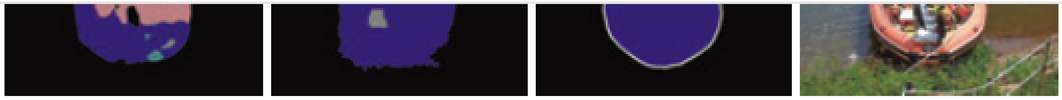
	pixel acc.	mean acc.	mean IU	f.w. IU
Gupta *et al.* [14]	60.3	-	28.6	47.0
FCN-32s RGB	60.0	42.2	29.2	43.9
FCN-32s RGBD	61.5	42.4	30.5	45.5
FCN-32s HHA	57.1	35.2	24.2	40.4
FCN-32s RGB-HHA	64.3	44.9	32.8	48.0
FCN-16s RGB-HHA	**65.4**	**46.1**	**34.0**	**49.5**

	pixel acc.	mean acc.	mean IU	f.w. IU	geom. acc.
Liu *et al.* [23]	76.7	-	-	-	-
Tighe *et al.* [33]	-	-	-	-	90.8
Tighe *et al.* [34] 1	75.6	41.1	-	-	-
Tighe *et al.* [34] 2	78.6	39.2	-	-	-
Farabet *et al.* [8] 1	72.3	50.8	-	-	-
Farabet *et al.* [8] 2	78.5	29.6	-	-	-
Pinheiro *et al.* [28]	77.7	29.8	-	-	-
FCN-16s	**85.2**	**51.7**	39.5	76.1	**94.3**

Pascal VOC 2011 dataset (Left), **NYUDv2 Dataset (Middle), SIFT Flow Dataset (Right)**

- FCN-8s is the best in Pascal VOC 2011.
- FCN-16s is the best in NYUDv2.
- FCN-16s is the best in SIFT Flow.





Visualized Results Compared with [Ref 15]

The fourth row shows a failure case: the net sees lifejackets in a boat as people.

I hope I can review more about deep learning techniques for semantic segmentation in the future.

References

1. [2015 CVPR] [FCN]
   Fully Convolutional Networks for Semantic Segmentation
2. [2017 TPAMI] [FCN]
   Fully Convolutional Networks for Semantic Segmentation

Machine Learning Deep Learning Artificial Intelligence Neural Networks Convolutional Network