Paper Implementation Report

Fully Convolutional Networks for Semantic Segmentation

Ⅰ. Implementation of FCN's key contribution

The key concept of this paper is that they extend the existing classification nets to segmentation by **replacing fully-connected layers with fully-convolutional layer** and improved not only segmentation performance but also the speed of learning and inference **through multi-resolution layer combinations (skip connections)**. To demonstrate these contributions, I conducted two major experiments. The first was to change existing classification models (AlexNet, VGG, GoogLeNet) to fully-conventional layers to evaluate segmentation performance and visualize the score map of VGG16. The next experiment was to reproduce the results of FCN32s, FCN16s, and FCN8s using the val dataset of VOC2011 and VOC2012.

I tried to royally follow the implementation details in this paper as far as possible. As noted in the paper, the batch size: 20, momentum: 0.9, weight decay: 0.0005, and SGD was used as an optimizer. In this case, the all deconvolutions (transposed conv) were initialized to function as bilinear interpolation, and made them learnable, except the final deconv layer. Also, I applied zero initialization to all the score layers, and used dropout the fc layers. Furthermore, I doubled the learning rate at the bias term in each layer and didn't apply the weight decay to them, as the author did. All models were trained up to 200 epoch.

Ⅱ. Experiment results

ⅰ. Adapting classifiers for dense prediction (From classifier to dense FCN)

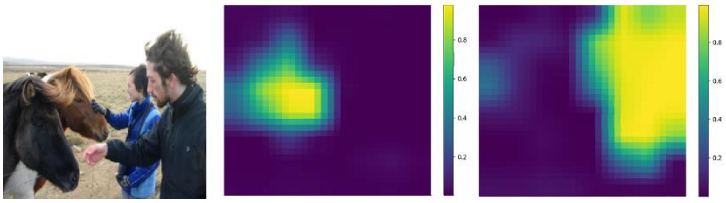

Figure 1. Visualization of the each score map. (middle: horse, right: person)

	FCN-AlexNet	FCN-VGG16	FCN-GoogLeNet
Mean IoU (%)	26.4	**38.9**	33.6
Forward time	0.8 ms	16.9 ms	8.0 ms

Table 1. Results of the FCN-Classifiers.

First, I changed the classifier part of AlexNet, VGG and GoogLeNet to fully convolutional layers with a 1x1 kernel size, and fine-tuned all layers by backpropagation through the whole net. For AlexNet, the layers were somewhat different from the ones the author used, because I used a pre-trained model in *torchvision* library. As Table1 shows, mean IoU was smaller than the figures in the paper, b**ut the tendency was similar.** And you can see that it was faster for forward time, I guess it is because I experimented with *Titan V*, unlike the author. Figure 1 is a visualization of the corresponding parts of each horse and person in the score map of **VGG16**, and shows that **they are activated for the actual part.**

ⅱ. Comparison of skip FCNs on a subset of PASCAL VOC 2011 validation dataset

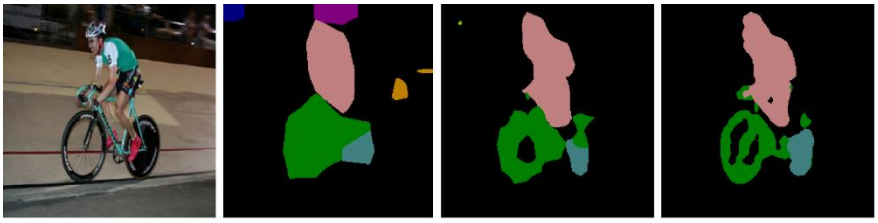

Figure 2. Each FCN's segmentation results. (origin, FCN32s, FCN16s, FCN8s)

	pixel acc.	mean acc.	mean IoU	f.w. IoU
FCN-32s-fixed	82.9	40.5	31.6	71.4
FCN-32s	82.3	**57.4**	39.0	73.0
FCN-16s	84.9	52.6	**40.3**	75.5
FCN-8s	**85.4**	50.2	39.0	**75.7**

Table 2. Performance results of each FCN.

Next, I checked the segmentation result of each FCN model with the added skip connection. FCN32s were initialized with the pre-trained parameters of VGG16 to learn with a learning rate of 10e-4. FCN16s and FCN8s were immediately initialized with top-level learned course FCNs, and they were selected as the best performers by learning 10e-3, 10e-4, 10-e-5 and 10-e-6, respectively. Table2 shows that FCN8s had high values for ***pixel acc.*** and ***f.w. IoU***, but ***mean acc.*** and ***mean IoU*** had the highest values for FCN32s and FCN16s, respectively. However, Figure2 shows that from FCN32s to FCN8s, namely, **the more course layers are combined, the more detailed the segmentation result is.**

Ⅲ. Discussion

Through this experiment, we can not only forward images regardless of image size but also easily expand task from classification to segmentation by using a fully conventional layer. Also, it was possible to demonstrate that more detailed segmentation results could be obtained by combining information from the courses layer at the front of the network with information from the fine layer at the back. However, the above results showed that the overall mean IoU results were lower than the figures presented in the paper. **To look back at the reasons for this,** the author used image size 227x227, but I used 256x256, and author local response normalization (LRN) in conv1, conv2. With the advent of BatchNorm, recently LRN has been deprecated, so I didn't even apply BatchNorm to the experiment. That might make the problems with internal covariance shift. Or because the results of each FCN have very subtle differences, as can be seen in the above, the results may vary depending on some random initialized values. In addition, given that the results of FCN8s which captured the very details, the performance in details seems to have an effect on the pixel acc. and frequency weighted IoU.