Semantic Segmentation based on Deeplabv3

Ming Cui **Haoran Yao** **Gongchang Chu**

Abstract

Deeplabv3 is a new method to realize the semantic image segmentation task. In this paper, we implemented the Deeplabv3 model using Pytorch, based on previous network structure with some modifications. Specifically, we replaced the dataset in original implementation "VOC 2012" with "cityscape". According to the test, Deeplabv3 worked perfectly on "cityscape" dataset and the modifications. A modification of the basebone network "ResNet101" is also proposed. Then we made a comparision between the new network and the original network.

1 Problem Statement and Motivations

Image semantic segmentation is an important researching area in computer vision field. Basically, semantic segmentation is aimed to make a classification on the image pixel-by-pixel. With an input image, we could get the annotation of the image. Different from the object detection, semantic segmentation requires much more accuracy on the edge of one object.
Image semantic segmentation is applied on many fields nowadays, such as self-driving, medical image processing, robotic vision. We are motivated by this large usage of semantic segmentation and its novel theory such as up-sampling and Deconvolution layer. Consequently, we decided to implement the image semantic segmentation task.

2 Background

Image semantic segmentation is a significant branch of scene understanding in computer vision. With the development of deep learning, Deep Convolution Neural Network(DCNN) has become the most powerful method to realize image semantic segmentation. However, during the process of traditional CNN, the size of the image becomes smaller and smaller due to the Polling layer which is not suitable of semantic segmentation task. The main point is that, an output annotation image with the same size of the original input image is required. To tackle this problem, Jonathan and his group introduced Fully Convolutional Networks(1) to achieve image semantic segmentation. This FCN network structure is regarded as the milestone of this area.
In short, FCN utilizes an up-sampling layer after the last convolution layer. The up-sampling process recovers the last feature map into the same size of the input layer. Consequently, a prediction towards every pixel of the image is realized.
After the FCN structure, other methods such as U-net(2), Segnet(3) are created. Among these new methods, the most powerful is Deeplabv3(4) which is developed by Google. Deeplabv3 is based on the previous work Deeplabv1 and Deeplabv2(5), which firstly introduced the concept of Atrous convolution. Deeplabv3 also utilizes the Atrous convolution method, but it creates the "atrous spatial pyramid pooling"(ASPP) model, which utilizes four atrous convolution kernels with different rates. According to the data from the paper(4), Deeplabv3 improves the pixel accuracy to 85.7%.
More information of Deeplabv3 will be introduced in Section 3.

3 Methodology

To efficiently implement the semantic image segmentation, we build our network Deeplabv3+ with ResNet(Residual Neural Network) and improved ASPP(Atrous Spatial Pyramid Pooling) and encoder-decoder structure. The core idea of ResNet is introducing so-called “identity shortcut connections” which is also introduced by Highway network. So ResNet can be considered as a special case of Highway Network but performs better. Actually, Deeplabv3+ network is a combination of Deeplabv3 and a simple decoder, the Deeplabv3 acts as a encoder in Deeplabv3+ network.

3.1 Using ASPP

ASPP is introduced in Deeplabv2 which implements four different atrous convolutions with different atrous rates in parallel. For common pooling layer, the feature map will be compressed in order to simplify the complexity of computation while reducing the resolution. ASPP is effective to resample features at different scales for accurately and effectively classifying regions of an arbitrary scale.

3.2 The process of ASPP

As described above, ASPP with four different atrous rates in parallel can capture multi-scale information. In this project, our ASPP consists of one 1×1 convolution and three 3×3 convolutions with Atrous rates = (6,12,18) and batch normalization layer. We feed the resulting image-level feature into the 1×1 convolution and then bilinearly upsample the feature to the desired spatial dimension (by going through several convolution layers). Next our ASPP will work by doing the convolution in four branches in parallel and then concatenate the results resulting from four different branches.

3.3 Proposing a simple decoder

Actually, in initial Deeplabv3 network, the feature are bilinearly unsampled by factor of 16(output-stride=16), which can be considered as a simple decoder, but this decoder has one problem that it can not effectively recover object segmentation details. So in out project, we implement a simple decoder to refine the segmentation results along object boundaries.

3.4 The process of whole Deeplabv3+ network

For the whole network, we have four main layers behind the initial 11 convolution. The first three layers are the common convolution layers with batch normalization, the output of each layer will be put as an input of the next layer. The fourth layer is different, we implement it by using the atrous convolution with a 3×3 convolution and rate equals to 2. After the image goes through these four convolution layers, we make the block pass through the ASPP. We concatenate the resulting features generated by four different branches into a image-level feature and make it pass through another 1×1 convolution to finally generate logits. We also apply a 1×1 convolution on the low-level features to reduce the number of channels and concatenate the result with the result unsampled by a factor of 4 from the encoder before feed it into a 3×3 convolution to get the prediction result. This process can be illustrated in Figure 1.

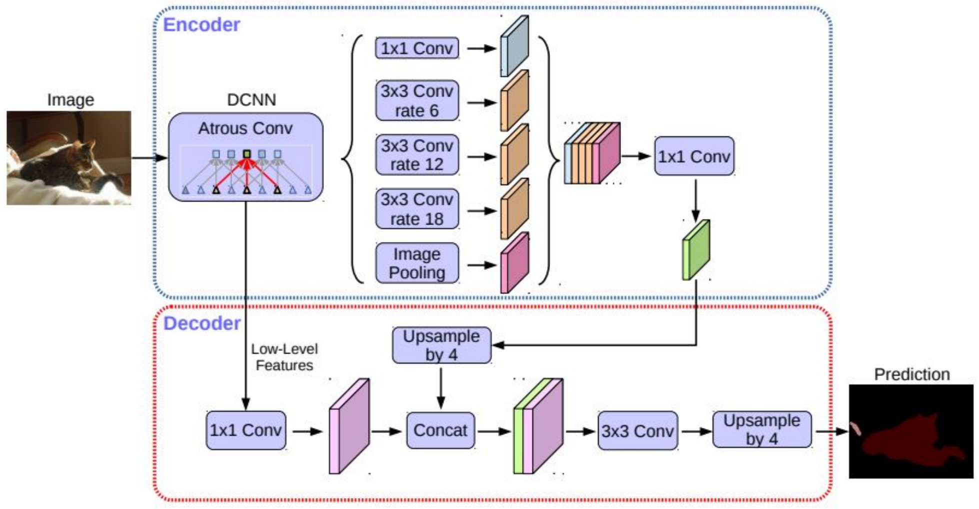

Figure 1: The whole process of Deeplabv3+. (Picture from the original paper(6))

4 Experiment

According to the methodology mentioned in Section 3, we built our network based on previous implementation of deeplabv3 but with some modifications. We firstly followed the paper (4) to built our network with ResNet101 backbone and four different Atrous Spatial Pyramid Pooling kernel. The kernel size and dilation value of these four kernel is shown in table 1. We trained this network on the Google Clout Platform with NVIDIA Tesla K80 GPU. The whole training consists of 40 epochs and we utilizes the tensorboard plugin to monitor the training process. It takes almost 13 hours to train this network. According to the paper, ResNet101 performs better than ResNet50. Consequently, we modified the backbone ResNet101 into ResNet156 to train the model again in order to get a better result. The training for ResNet156 backbone network takes 16 hours. The description of the datasets is shown in Subsection 4.1. The method we used to evaluate the network is shown in Subsection 4.2. The result of this experiment is shown in Section 5.

Table 1: ASPP parameters

Kernel Number	Kernel Size	Dilation Value
1	1	1
2	3	6
3	3	12
4	3	18

4.1 Datasets

The datasets we used is different from the previous implementation using Pytorch. We choose to use Cityscapes dataset(7). Cityscapes Dataset is a dataset for image semantic understanding which focus on the urban street scenes. This dataset contains 30 classes which is more complecated than VOC 2012 dataset. We used the images with fine annotations. These 5000 images are taken from 50 different cities across the world. Specifically, this dataset contains 2975 images for training, 500 images for validating, 1525 images for testing. For simplicity, the images are reshaped to 513×513 pixels.

4.2 Evaluation Method

In order to evaluate the network we built, we choose these methods to judge the performance of the network. Firstly, we measure the total loss during the training (in Subsection 5.1). As it is a classification task, we choose to use the cross-entropy loss to represent the difference between the prediction labels and the true labels. Then we measured the pixel accuracy which represents the rate of correct classification of the pixels in the image(in Subsection 5.2). What's more, we also measured the Mean Intersection over Union (MIoU) value of the network(in Subsection 5.3). Assuming that p_{ij} stands for a pixel of class i and is predicted to class j, then:

$$pixel\ accuracy = \frac{\sum_i p_{ii}}{\sum_i \sum_j p_{ij}} \tag{1}$$

$$MIoU = \frac{1}{n} \times \sum_i \frac{p_{ii}}{\sum_j p_{ij} + \sum_j (p_{ji} - p_{ii})} \tag{2}$$

At last, we input some original images into the trained network and observed the output. Then, we compared the predicted annotation images with the ground-truth annotation images.(in Subsection 5.5)

5 Result and Analysis

In this section, we listed the results as we mentioned in Section 4.2. We used tensorboard to vitalize the results. We firstly run the network with ResNet101 backbone, then we run the network with ResNet156 backbone. The analysis for these results will be shown in each corresponding subsection.

5.1 Loss Curve

In this subsection, we will demonstrate the loss curves for the training set and the validation set. The loss curve for the training set is shown in Firgure 2 and the loss curve for the validation set is shown in Figure 3. In these figures, the red curve illustrates the result for the ResNet101 network and the blue curve illustrates the result for the ResNet105 network. As is shown in these figures, the loss of the training set and validation set declines smoothly during the training which means that the difference between the ground-truth annotation labels and the prediction labels are getting smaller and smaller. However, the loss for ResNet156 network does not perform better than ResNet101 network, which is not as expected.

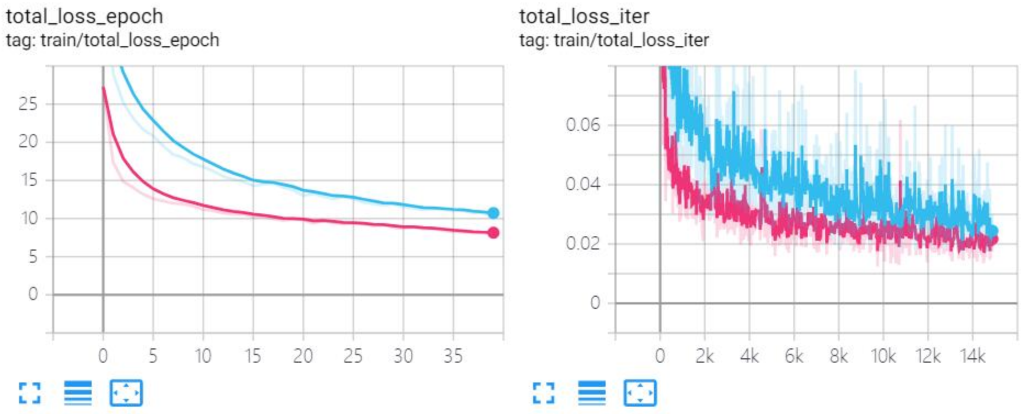

Figure 2: The loss value for the training set (blue curve for ResNet156 and red curve for ResNet101)

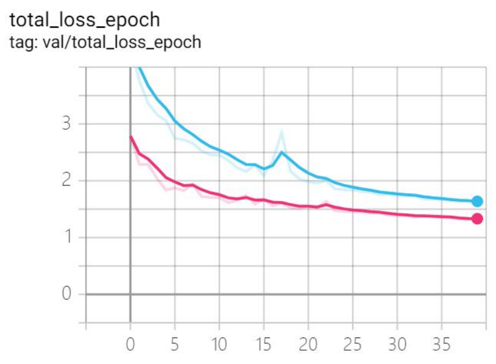

Figure 3: The loss value for the validation set (blue curve for ResNet156 and red curve for ResNet101)

5.2 Pixel Accuracy

The curve for pixel accuracy is demonstrated in Figure 4. In details, the pixel accuracy for the ResNet101 and ResNet156 reaches 94.86% and 93.86% respectively after training 40 epochs. We have to admit that the performance of ResNet101 network is better.

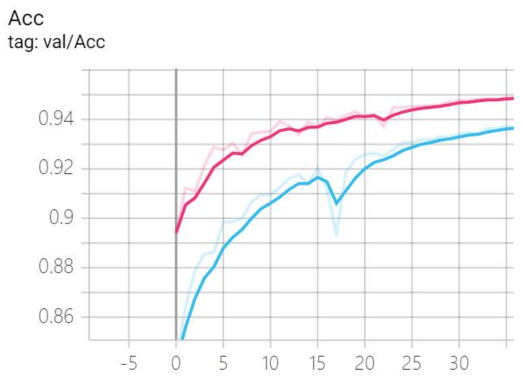

Figure 4: The pixel accuracy curve (blue curve for ResNet156 and red curve for ResNet101)

5.3 MIoU Value

The curve for the MIoU value is demonstrated in Figure 5. In details, the MIoU value for the ResNet101 and ResNet156 reaches 64.66 and 50.1 respectively after training 40 epochs. It is the same case as the result in Subsection 5.2.

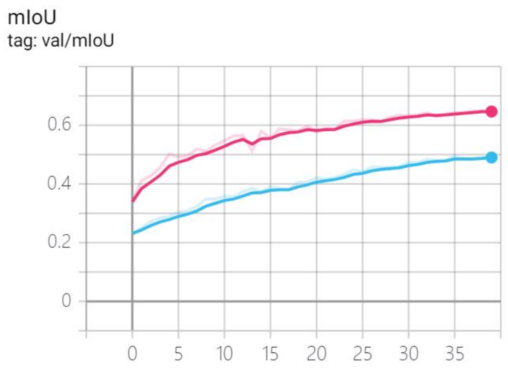

Figure 5: The MIoU value curve (blue curve for ResNet156 and red curve for ResNet101)

5.4 Analysis for Pixel Accuracy and MIoU Value curves

As we can see from the Subsection 5.2 and Subsection 5.3, the performance of the ResNet156 is worse. we looked into these curves and found that, the initial value of ResNet156 curves in far below the initial value of ResNet101 curves. Especially in the pixel accuracy curves, the ResNet156 curve starts from 85% while the ResNet101 curve starts from 90%. I think it may bacause the initialization for the ResNet156 is not as good as ResNet101. However, the first derivative of the ResNet156 pixel accuracy curve is larger than the ResNet101 curve which means that the pixel accuracy of ResNet156 network may go beyond the ResNet101 network if it is trained with more epochs. The time for training is also a disadvange of ResNet156 backbone, as it is much more complecated than ResNet101 backbone. It is still unclear why the performance of ResNet101 is better than ResNet50. We think it may because of the vital overfit in the backbone natwrok. The ResNet101 network performs beteer because of it complexity and ability to gain more information from the orignal images. And the amount of information is enough for ASPP network.

5.5 Output Images

There is no doubt that the output images is the most exciting part. After getting the annotations for each pixel, simple coloring method are used to generated these annotations images. It is obvious that different classes are colored differently. The left sub-figure shows the ouput of ResNet101 and the right sub-figure shows the output of ResNet156. Basically, the output predicted annotations images are almost the same as the ground-truth annotations images.

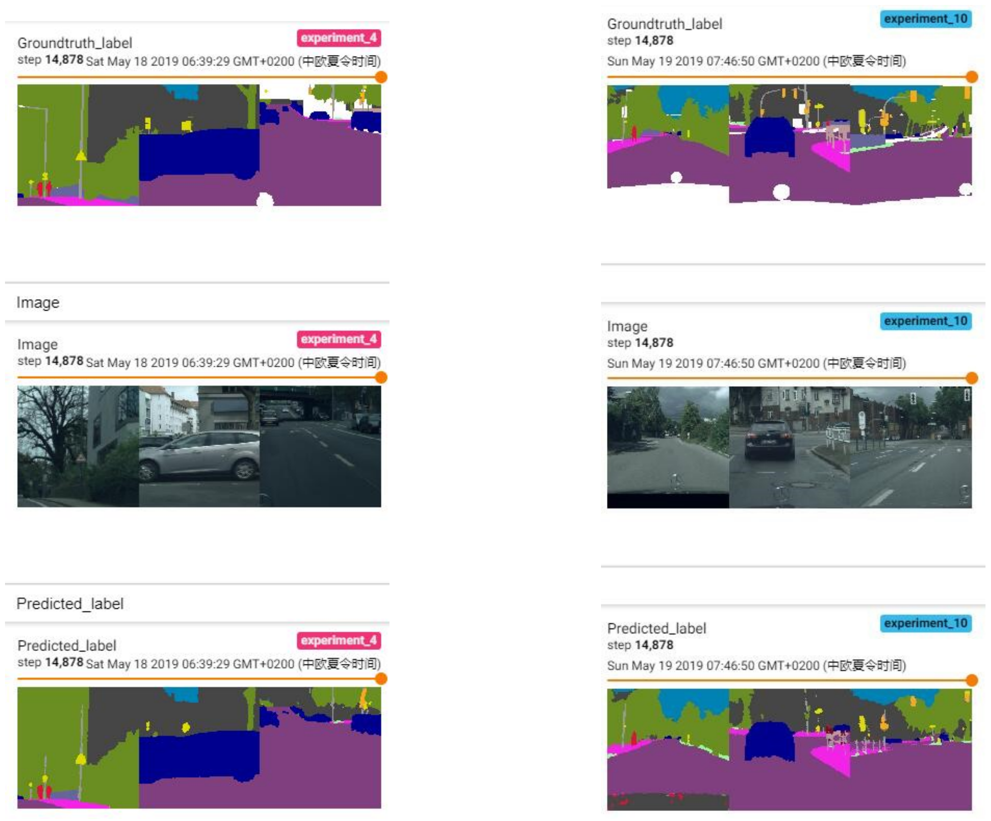
Groundtruth_label
experiment_4
step 14,878 Sat May 18 2019 06:39:29 GMT+0200 (中欧夏令时间)
Image
Image
experiment_4
step 14,878 Sat May 18 2019 06:39:29 GMT+0200 (中欧夏令时间)
Predicted_label
Predicted_label
experiment_4
step 14,878 Sat May 18 2019 06:39:29 GMT+0200 (中欧夏令时间)
Groundtruth_label
experiment_10
step 14,878
Sun May 19 2019 07:46:50 GMT+0200 (中欧夏令时间)
Image
experiment_10
step 14,878
Sun May 19 2019 07:46:50 GMT+0200 (中欧夏令时间)
Predicted_label
experiment_10
step 14,878
Sun May 19 2019 07:46:50 GMT+0200 (中欧夏令时间)

Figure 6: The outputs and corresponding ground-truth annotations

6 Discussion and Conclusion

In this project, based on the paper about Deeplabv3+(6) and some previous implementations, we built the network which realized the images semantic segmentation. We firstly built the network according to the paper and trained the network with cityscapes dataset. We modified the previous backbone network from ResNet101 to ResNet156 and compared the results. However, it turns out that the more complexity of the backbone cannot improve the performance of the network. We think that it might due to the overfit in the backbone network. We planned to change the dropout value to reduce this influence in the future. We also plan to modify other parameters of the network such as the ASPP kernels in order to get a better performance.

References

[1] J. Long, E. Shelhamer, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation,” *arXiv e-prints*, p. arXiv:1411.4038, Nov 2014.

[2] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” *CoRR*, vol. abs/1505.04597, 2015. [Online]. Available: http://arxiv.org/abs/1505.04597

[3] A. Kendall, V. Badrinarayanan, , and R. Cipolla, “Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding,” *arXiv preprint arXiv:1511.02680*, 2015.

[4] L. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” *CoRR*, vol. abs/1706.05587, 2017. [Online]. Available: http://arxiv.org/abs/1706.05587

[5] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” *CoRR*, vol. abs/1606.00915, 2016. [Online]. Available: http://arxiv.org/abs/1606.00915

[6] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in *The European Conference on Computer Vision (ECCV)*, September 2018.

[7] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.