A Review of Neural Network based Semantic Segmentation for Scene Understanding in Context of the self driving Car

J. Niemeijer[1], P. Pekezou Fouopi[2], S. Knake-Langhorst[2], and E. Barth[3]

[1] Medizinische Informatik, Universität zu Lübeck, Joshua.Niemeijer@student.uni-luebeck.de
[2] German Aerospace Center, Braunschweig, {Paulin.Pekezou,Sascha.Knake-Langhorst}@dlr.de
[3] Institute of Neuro- and Bioinformatics, Universität zu Lübeck, barth@inb.uni-luebeck.de

Abstract

This paper tackles the challenge of scene understanding in context of automated driving. To react properly to the conditions given by the surrounding scene, the car has to understand its environment. Further the real time capability of a method solving this task is essential. For scene understanding the car has to detect and classify its surrounding objects. For this purpose a semantic segmentation can be employed to assign a class label to every pixel. In this paper we evaluate the state of the art methods for the semantic segmentation and perform tests on the FCN-8 architecture. Due to hardware limitations, we train the FCN-8 on a downscaled version of the Cityscapes Dataset, containing urban traffic scenes. The evaluation of the results shows, the necessity to train the FCN-8 on the original size City Scapes Dataset. We conclude that we need to purchase a better hardware.

1 Introduction

Scene understanding is an important task in the context of automated driving. In order to react properly to the conditions given by the surrounding scene, the self driving car has to understand its environment. For example information about the state of other traffic participants or objects are needed to predict their behaviour and avoid collisions. Besides the quality of scene understanding, the real time capability of a method solving this task is essential, because of the fastly changing traffic conditions. Amongst other things the self driving car perceives its environment through a camera. By performing a semantic segmentation on the recorded images, one receives labels for every pixel, describing the object it is part of. Hence an understanding of the present objects and their location is created. These information in turn are the basis of further scene analysis. In this paper we review a variety of pixel wise semantic segmentation Methods based on fully convolutional neural networks. In addition we show the results of training and testing the FCN-8 architecture from the paper [1] on the Cityscapes Dataset [2].

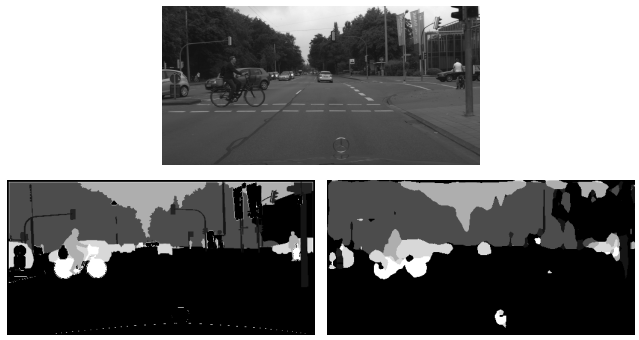

Figure 1: On top the downscaled version of the original Cityscapes validation image can be seen. The image on the left side is the ground truth and the image on the right is the result of the network trained on the Cityscapes Dataset.

2 Material and Methods

Semantic segmentation can be understood as the pixel wise labeling of an image, to declare which class-label each pixel belongs to. An example can be observed in Fig. 1. The top picture shows the image picturing a traffic scene to be segmented, the image on the left side is the groud truth semantic segmentation and the image on the right side is the semantic segmentation computed by a neural network. The different gray values indicate the classes the pixels belong to (e.g. car or road etc.). There are many algorithms to tackle this task, however recently artificial neural networks have offered the best results. In general there are two main architectures of artificial neural networks to cope with the task. Bounding Box approaches first detect the object and then classify all the pixels belonging to this object. However recent challenges showed that methods which are based on the pixel wise segmentation architecture proposed in [1], perform generally better. Hence in this paper we take a closer look at these methods.

The general architecture, as it is presented in [1] can be ob-

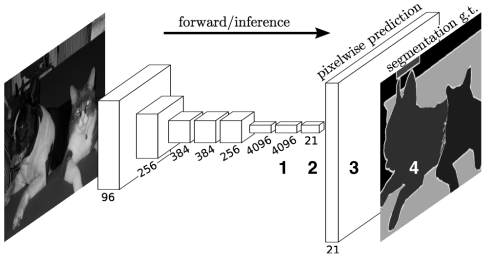

Figure 2: The general architecture of pixel wise segmentation is presented. The image information is processed to a feature map of the shape (1). From this feature map a score map for each of the (in this case 21) classes is derived (2). These score maps are upsampled to the original image size (3) and for every pixel the class with the highest score is chosen (4). The image was taken and modified from [1].

served in Fig.2. The architecture is divided up into four stages. In the first stage the image data is processed, by extracting the most important features. The output of this data processing is a downscaled feature map, in which every pixel represents a receptive field in the original image and hence holds the information of this area. Based on this information in the second stage a score map Sc is computed for every class. So the output of this layer, the score map (Sc), is of the shape $lengthFeatureMap \times widthFeatureMap \times numberClasses$. The score values can be interpreted as a probability of a pixel, to belong to a semantic class. In the third stage, the now coarse score map is interpolated to the size of the original image. For interpolation commonly the bilinear interpolation is used. Here an upsampled score value is weighted by the inverse distance to its four closest neighbours in the lower-resolution score map. Now for every pixel in the original image for every class there is a score. To determine the class label of an pixel (x, y), in the fourth stage the class with the maximum value over all score values is computed, making it the pixel's class label CL.

$$\forall(x, y)\ CL = \arg\max_{z} Sc(x, y, z) \tag{1}$$

Approaches that are based on this architecture try to improve the data processing, the creation of the score layer or the interpolation stage. For example the approach presented in [3], tries to improve the interpolation stage through an architecture based on the Laplacian Pyramid. In this approach a low-resolution score map is refined by using higher frequency details derived from higher-resolution feature maps. The idea is based on the assumption that lower-resolution feature maps have bigger receptive fields and hence more context information, leading to better predictions in the score map. Whereas higher resolution feature maps contain more information about local structures. So refining a low-resolution score map, which is computed from a lower-resolution feature map, with features derived from a high resolution score map saves details, hence creating confident predictions with a high level of details.

Conditional random fields are used in the creation of the score map. The approach in [4] is to capture the spatial context of a pixel and thus improving its classification, by using conditional random fields. The conditional random fields is thereby constructed, so that for every spatial position in the coarse feature map, there is a node. The pair wise connections between a node and all other nodes are drawn within a certain range around the node. By choosing the range spatial relations like above or under can be modelled. An other possibility is, to improve the creation of the feature map. In [5] a new shallower architecture of the residual network is used to create better features and thus improve the creation of the score map. Residual networks are neural nets that include neurons that map their own input to their output (residual units). The advantage is that these nets are easier to optimize. This new approach is based on the finding that paths in the residual nets that are deeper than the effective depth (number of residual units) aren't trained fully end to end. So by reducing the effective depth the net becomes fully end to end trainable. Empirical results show that these networks outperform deeper residual networks [5]. The approach in [6] also tries to improve the feature map. A pyramid scene parsing network is proposed, which combines global context information and local cues to improve the pixel wise predictions. However its architecture is slightly different from the baseline approach. In this case not the score map, which is created by a residual network, is upsampled, but directly the feature map. The pyramid pooling module is applied to the upsampled feature map and fuses features under four different scales. It thus separates the feature map into different sub regions and forms pooled representation for different locations. Then these new features are upsampled to the size of the original feature map and concatenated with it. Subsequently the pixel wise predictions are made by an convolutional layer.

Table 1: Model performance on the Cityscapes Dataset [2]

Model	mean class IoU	mean class iIoU
ResNet-38 [5]	80.6	57.8
PSPNet [6]	80.2	58.1
LRR-4x [3]	71.8	47.9
Adelaide-context [4]	71.6	51.7
FCN 8s [1]	65.3	41.7

Table 1 shows an overview of the presented methods. As it can be seen the FCN-8 performs worse than the methods inspired by its architecture (information about the dataset and the metric in Results and Discussion). The main reason we decided to use the FCN-8 was that at the time it was already available as a Caffe model. In addition to this the models from [5] and [6] were published only recently and developing a new architecture was not the goal of this work.

2.1 FCN-8

Fig. 3 shows the architecture of the FCN-8, which we used to perform experiments on the Cityscapes Dataset. This network is a fully convolutional network. Thereby the spatial information of the image is preserved. By employing a fully

connected layer this information would be destroyed [1]. The creation of the feature map is done in the layers Conv1 to Conv6-7. This network is a neural network for classification trained on the same dataset as the semantic segmentation without the classification layer. In this case the network is the VGG-16 [7]. The FCN-8 combines the output of fine and coarse layers, as it can be observed in Fig. 3. This is done by first scoring the output of these layers after pool3, pool4 and conv6-7 and then aligning the score maps through upsampling (interpolation) and cropping. After the score maps are brought to the original image size, they are merged by a 1×1 convolution, summing the score maps. By combining the information of coarse and fine layers the model makes local predictions with respect to global structures. The interpolation is initialized as a bilinear interpolation. During training the whole model is fine tuned end to end.

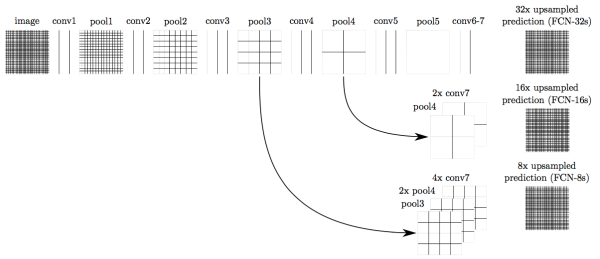

Figure 3: The figure (from [1]) shows the architecture of the FCN-8. (Conv = convolutional layer/pool=pooling layer)

3 Experiments

3.1 Dataset and quality metric

The Cityscapes Dataset contains 5000 images of street scenes with fine annotations for the purpose of semantic segmentation. An example for these images is pictured in Fig. 1. The images are taken in 50 cities under different conditions, seasons and daytimes. Annotations include 30 classes, which are divided up into the categories flat, human, vehicle, construction, object, nature, sky and ground. The images have got a HD resolution of 2048×1024 pixels. The training set includes 2974 images, the validation set includes 501 images and the test set includes 1525 images. Due to the fact that we do not have the ground truth images of the test set and hence can not evaluate results on this test images, we performed the tests on the validation set.
To evaluate the results of the experiments we use the quality metric IoU and iIoU, as is done in the cityscapes benchmark. These metrics are scalar values $\in [0, 1]$ computed per class over the whole test set to evaluate the segmentation. IoU (intersection-over-union) is defined as follows:

$$IoU = TP/(TP + FP + FN) \tag{2}$$

In which per class TP stands for true positive, FP for false positive and FN for false negative. This could also be interpreted as dividing the intersection of the segmentation with the ground truth by the union of the segmentation and the ground truth. IoU is biased toward object instances that cover large image area. To address this problem $iIoU$(instance-level intersection-over-union) is used:

$$iIoU = iTP/(iTP + FP + iFN) \tag{3}$$

iTP and iFN are computed by weighting the contribution of each pixel by the ratio of the class' average instance size to the size of the respective ground truth instance.

3.2 Training

Training the FCN-8 on the Cityscapes Dataset was done using a server with a Tesla K-20 GPU. This GPU has got 5 GB of RAM, which limited the maximum size of the images due to the fact that the image information and the size of the derived feature maps increase with the size of the image. Accordingly we downsampled the images from 2048×1024 pixels to 512×256 pixels, which offered a good RAM Capacity utilization. We already had a caffe model (`https://github.com/shelhamer/fcn.berkeleyvision.org.git`) of the FCN-8 trained on the Pascal Dataset to initialize the FCN-8 with for finetuning. We made the assumption that the features derived from the Pascal Dataset are well suited for the Cityscapes dataset. This can be assumed due to the fact that these dataset are similar and in general the features derived are mostly some kind of edge detection. We also adopted the interpolation from the given model, assuming it would be well suited due to the fact that the changes of the interpolation parameters are very small and the datasets are similar. Thus we retrained the layers that create the class-scores. For training we used the train IDs from the Cityscapes Dataset, encoding the 19 classes that can be observed in table 1. The training was done for 100000 iterations with a learning rate of 1e-14 and a batch size of three (again due to the low RAM).

3.3 Results

Fig. 1 shows example of the semantic segmentation performed on the rescaled validation set. On top the image to be segmented is pictured, on the left side the ground truth and on the right side the result of the FCN-8 on the rescaled image. It can be observed that in general the main structures of the picture, as cars, the street and the vegetation are well captured. In general these structures tend to occupy more space in the segmentation, than they actually occupy in the ground truth image. However the model struggles to segment finer structures as e.g. the rider.
In Table 2 the results of the test on the rescaled validation data are displayed relating to the classes (rescaled). On the left side of these scores the scores of the FCN-8 trained on the full sized Cityscapes Dataset (orig) can be seen. These scores are computed on the full size Cityscapes Test Set.
In general it can be observed that the FCN-8 trained on the rescaled Cityscapes Dataset performs on the validation dataset worse than the FCN-8 trained on the full sized

Table 2: Performance of the FCN-8 on the Cityscapes Dataset. orig. FCN-8 trained on the full size Dataset; rescaled FCN-8 trained on the rescaled dataset

Class	IoU orig.|rescaled	iIoU orig.|rescaled
building	89.2 | 69.4	/ | /
fence	44.2 | 2.8	/ | /
pole	47.4 | 10.9	/ | /
road	97.4 | 81.9	/ | /
sidewalk	78.4 | 34.1	/ | /
sky	93.9 | 70.1	/ | /
terrain	69.3 | 19.5	/ | /
traffic light	60.1 | 3.6	/ | /
traffic sign	65.0 | 15.5	/ | /
vegetation	91.4 | 74.6	/ | /
wall	34.9 | 0.8	/ | /
truck	35.3 | 1.1	22.2 | 1.1
train	46.5 | 1.0	26.7 | 3.54e-04
rider	51.4 | 0.3	33.4 | 0.4
person	77.1 | 30.7	55.9 | 29.9
motorcycle	51.6 | 0.3	31.1 | 0.1
car	92.6 | 65.4	83.9 | 50.1
bus	48.6 | 1.7	30.8 | 0.5
bicycle	66.8 | 28.7	49.6 | 23.7
average	65.3 | 27.0	41.7 | 13.2

Cityscapes Dataset and tested on the test dataset (65.3 vs. 27.0 IoU and 41.7 vs. 13.2 iIoU). Especially those classes which make up a low portion of the rescaled training data have a much lower IoU and iIoU scores. For example the classes truck and train have a prior portion of 0.0046 and 0.0016 and the IoU scores fall from 35.3 to 1.1 and from 46.5 to 1.0. Whereas the class car has a prior portion of 0.052 and the IoU scores only fall from 92.6 to 65.4.

The deployment time for both Models is 0.5 seconds or lower. The FCN-8 orig. was deployed on the TitanX (16 GB RAM) and the FCN-8 rescaled on the Tesla K20 GPU.

3.4 Discussion

The evaluation of FCN-8 trained on the rescaled dataset on its own training dataset showed similar results as the evaluation on the rescaled validation dataset (29.2mean IoU/13.35mean iIoU). It can be assumed, that no Overfitting took place but possibly Underfitting.Although the loss function converged it was still high, leading to the assumption that training for more than 100000 iterations might bring an improvement.

That the FCN-8 trained on the rescaled Dataset offers results, that are a lot worse is caused by the fact, that the information available for training the network is only 1/16 of the original information. The rescaling of the images leads in particular to the fact, that classes that make a small portion of the original training data, are not well enough represented for FCN-8 to offer good results. If the FCN-8 is trained on the Cityscapes Dataset in its original size, this model doesn't fulfil the real time requirements (runtime less than 100 ms) since it takes around 0.5 seconds to segment an image. However deploying the model on dedicated hardware like Nvidia DPX2 (`http://www.nvidia.com/object/drive-px.html`) can considerably reduce the run time.

4 Conclusion

The results of this work show that training the FCN-8 on the rescaled Cityscapes Dataset leads to a low detection rate, while deploying the FCN-8 on the full size Cityscapes Dataset doesn't fulfil the real time requirements. However for deploying the FCN-8 in a self driving car, a good detection rate is necessary. Hence we need to train the the FCN-8 on the full sized City Scapes dataset. For that we will need a stronger GPU with enough RAM, like the Nvidia Titan X for training or the Nvidia DPX2 for the deployment of the model. Another goal should be to improve the semantic segmentation by employing better architectures and training the models to distinguish between object instances.

The methods of semantic segmentation presented in this paper, aren't limited to the use in context of self driving cars, but can also be used in a medical context. An application example is the detection and monitoring of tumors.

Acknowledgement

The work has been carried out at the German Aerospace Center, Braunschweig (Institute of Transportation Systems) and was supervised by the institute of Neuro- and Bioinformatics at the Universität zu Lübeck.

5 References

[1] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, pp. 1–1, 2016.

[2] 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," *CoRR*, vol. abs/1604.01685, 2016.

[3] G. Ghiasi and C. C. Fowlkes, *Laplacian Pyramid Reconstruction and Refinement for Semantic Segmentation*, pp. 519–534. Cham: Springer International Publishing, 2016.

[4] G. Lin, C. Shen, A. van den Hengel, and I. D. Reid, "Exploring context with deep structured models for semantic segmentation," *CoRR*, vol. abs/1603.03183, 2016.

[5] Z. Wu, C. Shen, and A. van den Hengel, "Wider or deeper: Revisiting the resnet model for visual recognition," *CoRR*, vol. abs/1611.10080, 2016.

[6] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," *ArXiv e-prints*, Dec. 2016.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.