# Exercise 3 deep learning lab 2018

Max Fuchs
*Matriculation number: 4340529*
*maxfuchs@gmx.de*

## 1. Introduction

The task in exercise 3 was to setup an Encoder – Decoder network for semantic segmentation. The architecture was trained and tested on the CamVid semantic segmentation data set with 11 class classification.

## 2. Implementation

The network was implemented using tensorflow.
The net was optimized using Adam.
This report will document the results from 4 different decoder configurations. They build on each other and refine the resulting performance from 1 → 4.
Configuration 1 up samples the data directly after the encoder network with a up sampling rate of x16.
The 2nd to 4th configuration are illustratet in figure1[1] with DB being the skip connection used in the decoder from the associated encoder layer.

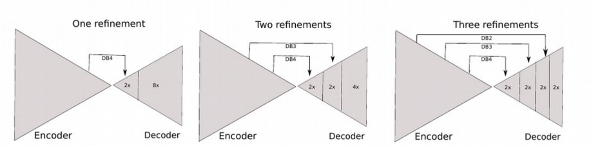

*figure 1: configurations 2,3,4 implementation*

## 3. Fully convolutional networks

The presented results use the Intersection over union metric which compares the bounding box of an image segment with the ground truth bounding box.
Where the final IoU score is calculated from all the single IoU values for the 11 different classes. The 11 classes are :
CamVid_classes: {Sky,Building,Pole,Road, Pavement, Tree, SignSymbol, Fence, Car, Pedestrian, Bicyclist, Unlabeled}
With the 12th class “Unlabeled” not used for IoU calculation.

| configuration | maximum IoU |
|---|---|
| 1 | 0.038 |
| 2 | 0.083 |
| 3 | 0.203 |
| 4 | 0.391 |

*table 1: maximum IoU Test values for all configurations*

| | Config1 | Config2 | Config3 | Config4 |
|---|---|---|---|---|
| Sky | 0.0310 | 0.1285 | 0.5123 | 0.8644 |
| Building | 0.0247 | 0.0998 | 0.3488 | 0.6237 |
| Pole | 0.0020 | 0.0315 | 0.0733 | 0.1254 |
| Road | 0.2743 | 0.2951 | 0.4012 | 0.8165 |
| Pavement | 0.0154 | 0.0890 | 0.1808 | 0.5536 |
| Tree | 0.0222 | 0.0844 | 0.2681 | 0.5285 |
| SignSymbol | 0.0018 | 0.0185 | 0.0729 | 0.1093 |
| Fence | 0.0011 | 0.0096 | 0.0382 | 0.0394 |
| Car | 0.0202 | 0.0737 | 0.1895 | 0.3847 |
| Pedestrian | 0.0101 | 0.0352 | 0.0895 | 0.1674 |
| Bicyclist | 0.0063 | 0.0249 | 0.0412 | 0.0813 |

*table 2: IoU per class for different configurations*

### 3.1. Configuration 1

Figure 2 plots the test results over the training epochs.
The learning curve increases in the first couple of epochs for about 0.3 %. From there on the test results vary around a mean value of 3.6 % IoU. The learning curve is varying due to a single image training, which leads to a non smooth learning curve. This could have been avoided with an increased batch size. In this implementation it was not possible, as the higher batch size would have lead to performance issues.

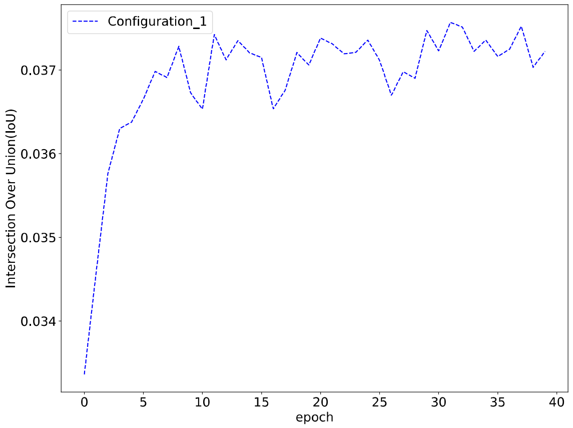

*figure 2: configuration 1; test result over learning epochs*

The segmentation with configuration 1 is only very limited usable for detailed image segmentation. From table 2 can bee seen that configuration 1 is only able to segment big classes like road decently well. For small details like feces this network is not producing any usable results.

## 3.2 Configuration 2

Configuration 2 is illustrated in figure 1, it consists of one skip connection that is processed in the decoder after increasing the spacial dimension from the last encoder layer by a factor of 2.

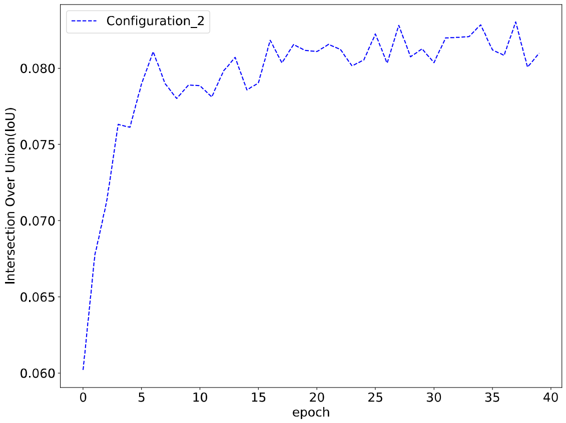

*figure 3: configuration 2; test result over learning epochs*

The Test accuracy in compared to configuration 1 doubles, with the learning cure looking very similar. The per class IoU from table 2 increases very similar in all classes, with performance peaks in classifying small object classes like SignSymbols and fences, where it can improve the IoU up to 10 times.

## 3.3. Configuration 3

The Configuration 3 adds a second skip layer in the encoder-decoder network, for the architecture see figure 1. The learning cure is slightly more smooth in the first 10 epochs but behaves like figure 3 and 4. Nevertheless this configuration can as well significantly increase the total IoU to upto 20%.

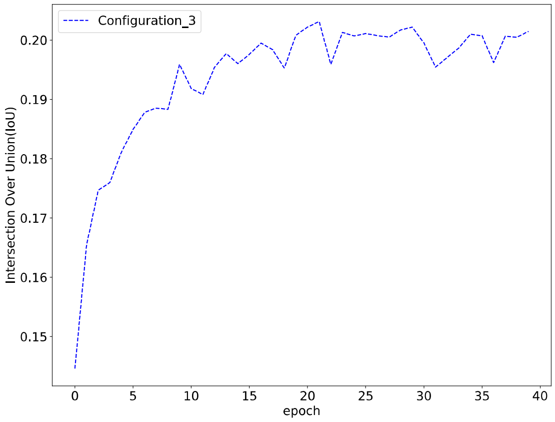

*figure 4: configuration 3; test result over learning epochs*

Comparing the per class performance from conf. 3 with the one from conf. 2, it appears that the maximum performance growth is ones more seen in small classes like fences.

## 3.4. Configuration 4

Table1 shows the dimensionality of the single stages of encoder and decoder part for configuration 4. Decoder values for each stage are documented after: transposed convolution → cropping → convolution.

| Layer number | dimension(h x w x d) |
|---|---|
| Encoder 1 | 300 x 300 x 3 |
| Encoder 2 | 150  x 150 x 24 |
| Encoder 3 | 75 x 75 x 32 |
| Encoder 4 | 38 x 38 x 96 |
| Encoder 5/decoder 1 | 19 x19 x 160 |
| Decoder 2 | 38 x 38 x 256 |
| Decoder 3 | 75 x 75 x 160 |
| Decoder 4 | 150 x 150 x 96 |
| Decoder 5 | 300 x 300 x 120 |

*table 3: Dimensionality of encoder and decoder network stages; configuration 4*

From table 1 it can be seen that configuration 4 performs about 10 times better than a configuration without skip connections. The Learning curve in figure 4 looks very similar to the one from the other configurations, but starting at around 24 % IoU.

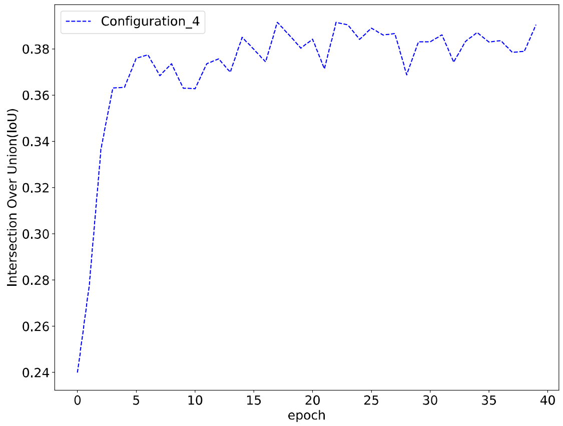

*figure 5: configuration 4; test result over learning epochs*

Interestingly the 4th configuration performs better in those classes having a rather high spatial dimension in the picture. It can more than double the classification for road and pavement. Whereas it only minor improves the classification for fences and SignSymbols.

## 6. Conclusion

Fully convolutional networks are a powerful tool for image segmentation. Where skip connections in lower resolution layers significantly increase the IoU of smaller objects in the image and skip connections in higher layers, regarding spacial dimension, can significantly improve the IoU of large area objects.

## 10. References

[1] Gabriel Leivas Oliveira, Tonmoy Saikia, Deep Learning Lab Course exercise sheet number 3