TEK5040 Assignment 1

Tonje Viddal Sandanger

07.09.23

1 Answers

1. **You may observe that the accuracy is quite high already after one epoch, what is the main reason for this?**
   The main reason that the accuracy is quite high already after one epoch is because the picture mostly consists of "NO ROAD" pixels. So if our program classifies every pixel as "NO ROAD" it would have an almost 70 percent accuracy. So it's actually not classifying things correctly it's just making assumptions that almost none of the pixels are road.

2. **The training loss should be decreasing from the start, but it might take some time before the accuracy increases after the first epoch, why do you think this is?**
   The training loss should be decreasing from the start, but it might take some time before the accuracy increases after the first epoch, because even if the loss decreases it might not predict correctly and for computation of accuracy we only use if it's classified correctly. So even if the loss decreases from 0.2 to 0.4, it's still not classified correctly and therefore not have any change in the accuracy but in the loss.

3. **How many training steps are there per epoch?**
   There are 70 training steps per epoch.

4. **How many training steps are there in total?**
   There are 810 training steps in total.

5. **Can you think of any cases where you can get good accuracy result without the model having learned anything clever?**
   An example of cases where you get good accuracy without the model having learned anything is for example when we overfit. When we overfit the accuracy can be very high and classify all the data correctly but it hasn't learned anything. It just learned to memorize the data. Therefor it generalizes badly to other data.

6. **Can you think of any other metrics that might shed some additional light on the performance of your model?**
   Precision and recall are good metrics to show how your model actually performs. These are both based on relevance. Precision for example looks at how many true positives

7. **Briefly describe transposed convolution also known as deconvolution. (Hint: It may be easier to consider 1D-case.)**
   Transposed convolution, also known as upsampling, is a way to increase the spacial resolution of a feature map i NN.

8. **How many trainable parameters do your model have?**
   I have 11217 trainable parameters in my model. This accounts for (43.82 KB) in total.

9. **Do you expect your model to behave differently under training and test models (i.e. for a given input do you get the same output from the model in train and test modes)? State the reason for your answer.**
   No, in this example the model shouldn't behave any differently under training and test models.

10. **If your task was to perform segmentation with more than two (eg: four classes), how would you the activation function and number of output channels?**
    If our task was to perform segmantation with more than two classes we would have to change the activation function so that if fit a multiclass segmentation. We would also need to make sure that the number of output channels matches the number of classes. We would also probably have to use a CE loss function and also change the way we classify the pixels. We could for example use one-hot encoding. The activation would have to be softmax.

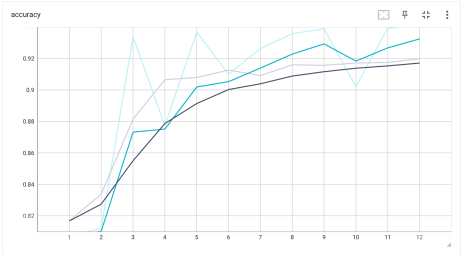
11. **Briefly explain why skip connections, shown in gray arrows in Figure 1, are useful in segmentation.**
    Skip connections are valuable in U-Net segmentation because they bridge the semantic gap. By merging low-level and high-level features, they enable the model to capture fine details and information, maintain spatial resolution, enhance localization, ensure gradient flow, and stabilize training, resulting in more accurate and detailed image segmentation.

12. **If your task is classification of a given image, how do you modify the model architecture and the loss function?**
    You would have to add fully connected layers in the end and a final classification sigmoid.

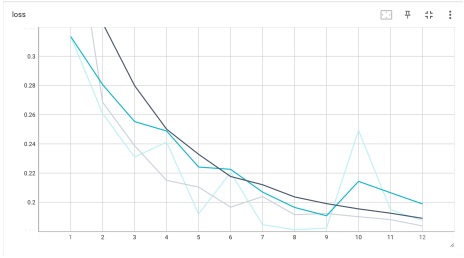
13. **If your task is to transform each input image to an image similar to another given image, how do you modify the loss function?**
If my task was to transform each input image to be similar to another given target image, we should modify the loss function to encourage the model to generate images that are close to the target image in terms of content and style. We could do this with content loss which measures the likeness between a generated image and the target image. We should use MSE to calculate this.

14. **Did you notice any improvements over the original model?**
Yes, the first thing I notices was that it took way less time for the algorithm to find good solutions with low loss. It only took the program 2 epochs to get down to 0.20 while it took the other program a lot more. I noticed it does run significantly slower. It took the simple segmentation about 3-4 minutes and the u-net segmentation takes a around 12 minutes. However, we could have cut the runtime shorter and only ran 3-4 epochs and that would save a lot of time. After a while it also produced solutions with lower loss than the simple segmentation. It also classified the road/no road much more consistently and better.

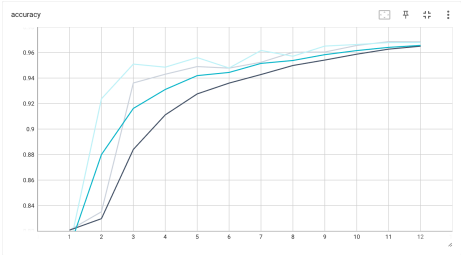
2 Loss and Accuracy Graphs



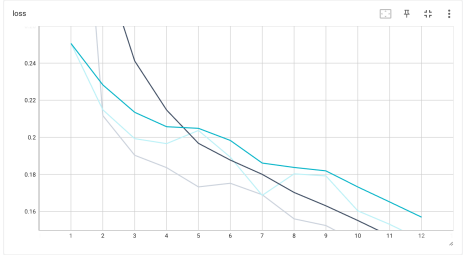
(a) Accuracy Simple



(b) Loss Simple



(c) Accuracy U-Net



(d) Loss U-net