TEK5040 - Assignment 1

Road segmentation

Segmentation is an important aspect of many autonomous platforms. Road segmentation can obviously be very helpful, for autonomous cars. After completing this exercise you will get a basic overview of how to do segmentations with deep neural networks.

Your goal is to create a deep learning model that can visibly segment road at a basic level. We use the Kitti road dataset, consisting of 289 labelled images.

Get started

You can base your solution on the train.py script, we have included in TEK5040_Assignment1.zip. Do not however copy and paste code from the web. You can of course be inspired by others work, but do cite where you got the **inspiration**.

When you unzip TEK5040_Assignment1.zip, you will find a **data_road** folder containing all the image data and a script **train.py**.

train.py have the basics for reading the image data and training a very simple neural network. What **train.py** are lacking is some visualizations, a good model and an appropriate loss function.

Tip: If your low on computational resources, you can reduce the size of your images to e.g. 224.

Your tasks

Visualize different metrics

You should make summaries of different metrics, so they can be displayed in Tensorboard. We want you to summaries: **Accuracy**, **Precision**, **Recall** and **F-measure**.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

$$F\text{-measure} = (1 + \beta^2)\,\frac{\text{Precision Recall}}{\beta^2\,\text{Precision} + \text{Recall}} \tag{3}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{4}$$

More on these metrics can be found in here:
http://www.cvlibs.net/publications/Fritsch2013ITSC.pdf

Also visualize your **loss**. All these measures should be visualized for both training and validation data.
Next visualize both the **images** and **segmentations** in Tensorboard. Then visualize the **predicted segmentation** of your network. Finally visualize the images with **predicted segmentation** superimposed as a transparent color. Similar to this:

 ,

Extra: Visualize the gradients of your updates as histograms.

Improve the model

Build a **better performing** network than the simple one in **train.py.** Change the loss function to a more appropriate one, and see if the metrics change.

Try data augmentation and regularization

Try random cropping, flipping and color changes to the images, and see if this affects the training and test results.

Test l2 regularization on your weight, and see if this improves your result.

Write a small report

The report should contain:

- your reasoning behind the choice of model
- how and why you think the results changed based on model augmentation and regularization
- The graph visualizations of your metrics

Deliver

Zip you **code**, **report** and the **log folder** for you best results (not the image data).
Send the zip-file to sigmund.rolfsjord@gmail.com

Good luck!