**Parametric Linear Regression using Deep Learning with concrete dropout for uncertainty estimation**

In the summary 10092019.docx we considered cases where a point cloud created from

for various intercepts and slopes for different noise levels .

The concrete dropout procedure from Gal 2018 makes it possible to adjust the dropout parameter while training. After 10000 epochs of training on 10000 different lines, the dropout rate converged to a specific rate for each layer. On an independent test set of 100 samples by using the dropout rate as determined while training 50 Monte Carlo samples were drawn by repeatedly predicting the parameters and with varying dropout masks. From these repeated trials we generate the standard deviation of the estimated parameters and .

However, we found out, that this standard deviation did not increase with increasing data noise . Instead, the standard deviation is dependent on the dropout rate itself. We therefore suggest holding the noise fix and train different nets for different noise levels .

**Determination of the optimal dropout rate**

While Gal 2018 adapts the dropout rate during training by adding a dropout regularization to the loss function in the backpropagation algorithm of neural network training, we derive an optimal dropout rate with an alternative interpretation.

We create three different datasets for training different neural nets in 1000 epochs given a noise . Further two independent testsets and . All neural nets were trained with a constant dropout rate of , which basically means no dropout.

The trained neural network is taken to predict the parameters   .

Together with the ground truth we can compute the covariance matrix

which states the variation of the neural network output.

Now we try to sample the predictions on testset with different dropout rate . By sampling 50 times the parameters for each sample in dataset individually. Over the different dropout samples, we compute the covariance matrix for each case in .

The final covariance matrix

Is achieved by calculating the mean covariance matrix over all individual matrices for the whole dataset .

We finally chose the dropout rate where the Frobeniusnorm

is minimal.

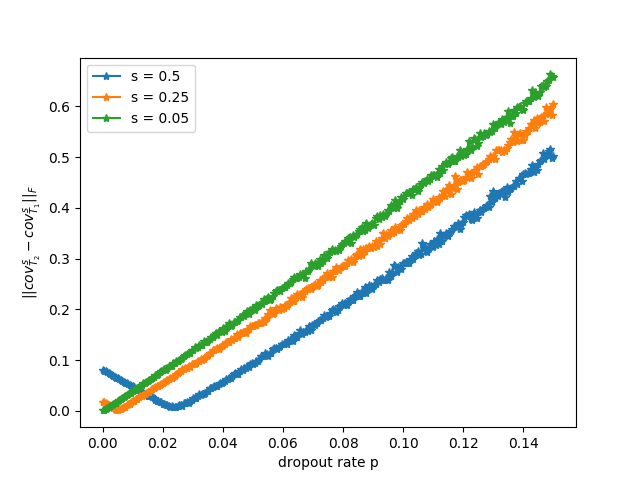
In Figure 1 the value of the Frobeniusnorm is plotted against the dropout parameter for three different noise scenarios. For each noise level , a different net was trained and subsequently tested on two different testsets as described above. The dropout rate was taken from an equidistant grid between and . For 300 different rates in this interval the results are determined and displayed.

Figure Frobeniusnorm of the differences of the covariance matrices for different noise levels in the data. For each level s a different net was learnt and subsequently applied to predict covariances in T1 and T2.

Table Minimum of the Frobeniusnorm of the two different covariance matrices for different noise levels in the data.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

**Training with optimized dropout rate**

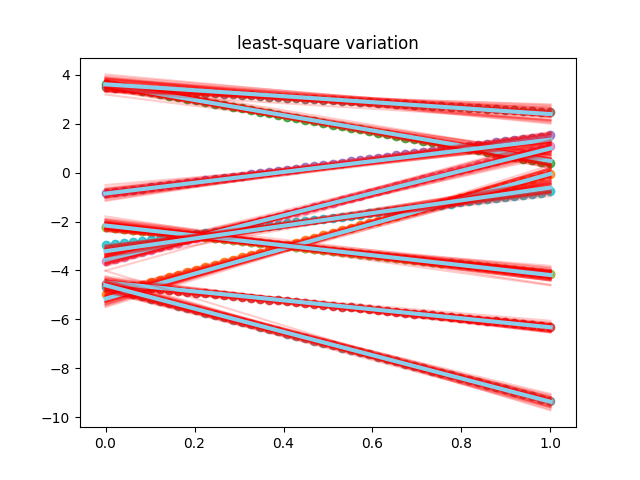
After determining the optimal learning rate as described above, we train a neural network on a dataset of a given noise level . We chose and a training set size of 10000 samples. According to the results we chose as dropout rate. After learning for 10000 epochs we tested the trained net and it’s variation on an independent testset. The results are shown in Figures 2 and 3.

Figure Predicted mean (skyblue) and variation (red) for 50 dropout samples of 10 testlines (scatter point) with noise 0.5 of the least-square procedure.

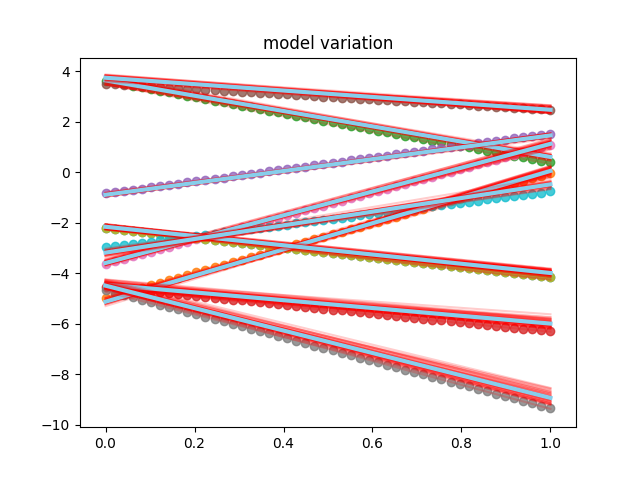
****

Figure Predicted mean (skyblue) and variation (red) for 50 dropout samples of 10 testlines (scatter point) with noise 0.5 of the trained neural network with a dropout rate of 0.024.

**Gal Concrete Dropout for each layer**

The original paper of Gal 2018 suggests learning a different dropout rate for each layer. We train with a noise level . The size of the training set is 10000 and we trained a multilayer perceptron with three hidden layers of 1024 neurons each. We start with a dropout rate of in every layer. After the training the following dropout rates were learnt:

Table Automatic determination of the dropout rate p according to Gal 2018 after training for 10000 epochs.

|  |  |
| --- | --- |
|  | Dropout rate p |
| Input -> Hidden 1 | 1.1e-1 |
| Hidden 1 -> Hidden 2 | 2.0e-2 |
| Hidden 2 -> Hidden 3 | 1.2e-1 |
| Hidden 3 -> b0 | 9.3e-3 |
| Hidden 3 -> b1 | 8.2e-3 |

