Model Pruning: Weekly Report 9

Patricia Gschoßmann

1. Weekly Progress

In last week's experiments the α-decay from 0.1 to 0.0 was too large to maintain the model's performance. In this week the experiments were continued with several different approaches in order to improve the results:

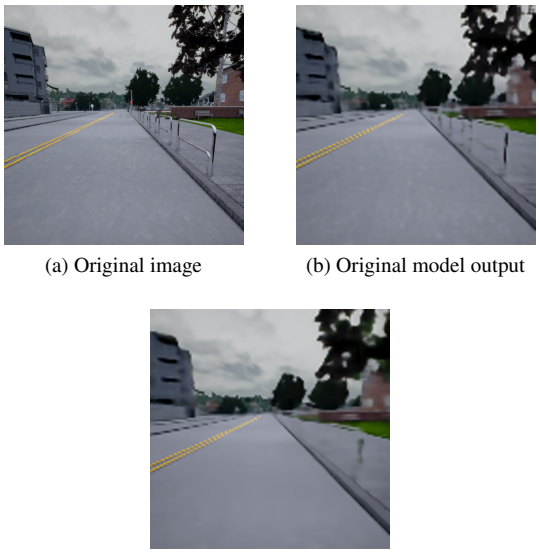
- Increase the learning rate at $\alpha = 0$ (to 0.01 and 0.1)
- Restart pruning from $\alpha = 0.1$ using an exponential α-schedule with a decay-rate of 0.3 (and an initial learing rate of 0.001)
- Analyze the weights (i.e. "How big are they compared to the original model's weights?")
- Add weight decay of 1e-4

After multiple experiments, I came to the conclusion that I was not able to reduce the validation loss. Unfortunately, I then discovered a bug in my `inference_rgb.py` used to test the models: The models' α-values were not updated accordingly (i.e. $\alpha = 0.1$ was used instead of $\alpha = 0.0$), which is why the models' outputs looked not as expected. The correct output of last week's model with $\alpha = 0.0$ is depicted in fig. 1. One can observe, that the output is a bit more blurred than in the original model output, however, the most prominent objects are still easily recognizable. Unfortunately, this issue was only noticed at the end of this week's pruning process, which is why not many other results can be shown in this report.

Since last week's final model produced reasonable reconstructions, I tried to multiply the weights to shrink the model. However, the implementation seems still to be incorrect as the smaller model only produces black images. I suppose the mistake lies in the transposed convolution, as the shape of the weights is different to standard convolutions: The dimensions for the in- and output channels are switched. I am going to investigate this issue in the following week.

2. Addendum to report week 7: GPU test results

As the GPU test results for the pruned VGG were rather unexpected, the implementation was revised. It turned out, that I measured the wrong parameter (i.e., GPU utilization vs. used memory/free memory). The following updated results were obtained: On average, the original model took $\approx$0.055 seconds for each batch, while the pruned model took $\approx$0.053 seconds.[1] The smaller model had an average unit consumption of 31.02%, whereas the original model needed 41.41% on average. These results are much more in line with the expected impact of pruning.



(a) Original image

(b) Original model output

(c) Correctly reconstructed image data for last week's model at $\alpha = 0$.

Figure 1: Original and reconstructed image data

3. Plan

- Multiply weights correctly to obtain smaller model
- If successfull: Prune corresponding decoder based on smaller model
- Test if it is possible to prune encoder and decoder at the same time

[1] The duration measurement has not been changed, but is listed for the sake of completeness.