Model Pruning: Weekly Report 5

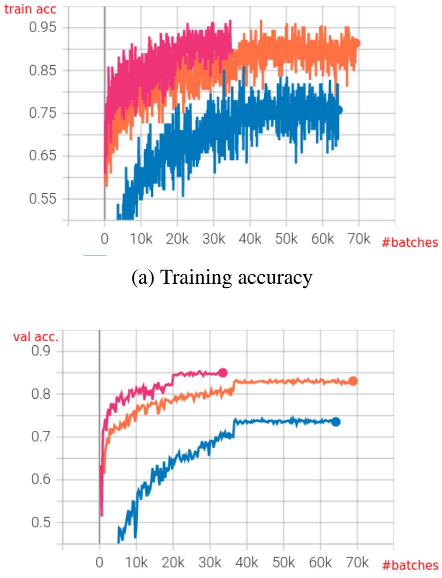
Patricia Gschoßmann

1. Weekly Progress

In this week I continued working on the unfinished classification task from last week. The experiment is nearly finished, i.e. the model currently reaches a validation accuracy of 85% at $\alpha = 0$.

2. Results

Last week's experiments were continued from $\alpha = 0.4$. Since an exponential scheduler with a decay rate of 0.05 was used until then, the following α-decays would have been extremly small. To speed up training, the maximum number of epochs was reduced to 100 and the decay rate was increased to 0.5 from this point on until $\alpha = 0.018$. Since all of these models maintained a stable performance, I decided to reduce the training duration even more by manually decreasing α to zero. With a learning rate of 0.001 the model was able to reach a validation accuracy of $\approx 73\%$, based on the model, which was trained with $\alpha = 0.05$ (see fig. 1, blue curve). To overcome this local minima, the initial learning rate was increased - at the beginning to 0.01 and then to 0.1 - whenever the training stagnated. In this way, a validation accuracy of $\approx 85\%$ was achieved. Currently a further iteration of this approach is running, to overcome the 85% accuracy bound. Figure 1 shows the corresponding training and validation accuracy of each iteration until now.



(a) Training accuracy

(b) Validation accuracy

Figure 1: Train- and validation accuracy for $\alpha = 0.0$, based on the previously trained model with $\alpha = 0.05$. Blue: First iteration with a learning rate of 0.001. Orange: Second iteration; first training was resumed with a lr of 0.01. Pink: Third iteration; resumed second training with a lr of 0.1.

3. Plan

Once the model reached a validation accuracy of nearly 90%, I am going to execute additonal tests to further compare the performances of the original and pruned model and determine other benefits besides the reduced storage space. The tests focus on time and average memory consumption and are already implemented.

I additionally plan to determine a more suitable α-schedule for the VGG16 on CIFAR10. The fact, that I was able to change the exponential decay-rate during scheduling and even set $\alpha = 0$ much sooner, than the specified scheduler would have, shows, that the used exponential decay main not be an ideal schedule for this task. A schedule with larger α-decays at the beginning and smaller at the end still makes sense, however, the decay should not become as small as with an exponential schedule. Probably a multiplicative or time-based approach would be suitable.
The changes I applied to the learning rate additionally indicate, that a cosine annealing learning rate scheduler would help to solve the problem. This type of lr scheduler was already used in previous experiments, but replaced by one, that reduces the learning rate once the validation loss stops decreasing. The change was made, when the initial learning rate was decreased to prevent loss fluctuactions.

Furthermore, I aim to apply the pruning approach to other use-cases.

4. Summary of new scripts

- `test.py`: Implements comparison between original and pruned model regarding time and memory consumption (GPU and CPU) on the test set.