DD2424 - Assignment 2

Timo Nicolai

April 29, 2019

I verified my implementation of the analytical gradient by computing the relative error with respect to the simple gradient calculated via forward difference. I reimplemented this numerical gradient calculation in Python.

Figure 1 shows the maximum relative errors for the gradient of both weight matrices and bias vectors, all of them are reasonably small so I assume that my gradient implementation is correct.

$\nabla W1$

Data Dimensions	Batch Size	λ	$\epsilon_{rel,max}$
20	1	0	3.66e-6
20	1	0.5	1.12e-3
20	20	0	1.13e-4
20	20	0.5	9.92e-4

$\nabla W2$

Data Dimensions	Batch Size	λ	$\epsilon_{rel,max}$
20	1	0	2.54e-7
20	1	0.5	7.92e-4
20	20	0	2.30e-4
20	20	0.5	7.72e-4

$\nabla b1$

Data Dimensions	Batch Size	λ	$\epsilon_{rel,max}$
20	1	0	4.72e-7
20	1	0.5	7.09e-7
20	20	0	8.48e-6
20	20	0.5	1.76e-6

$\nabla b2$

Data Dimensions	Batch Size	λ	$\epsilon_{rel,max}$
20	1	0	2.41e-7
20	1	0.5	2.58e-7
20	20	0	5.70e-6
20	20	0.5	3.55e-5

Figure 1: Maximum relative gradient error between analytical and numerical gradient. Maximum taken over all elements of a gradient matrix resulting from the same random initialization. Numerical gradient obtained via simple forward difference with $h = 1e - 6$.

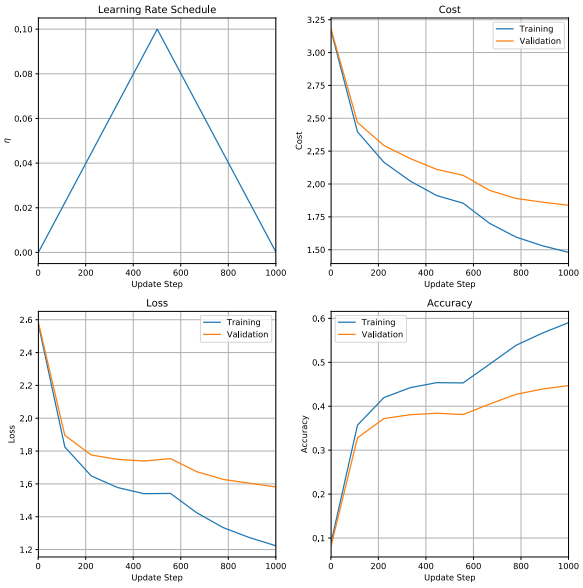

Figure 2: Learning rate schedule and learning curves for $\eta_{min} = 1e-5$ $\eta_{max} = 1e-1$, $\lambda = 0.01$ and $n_s = 500$. 10 datapoints per cycle.

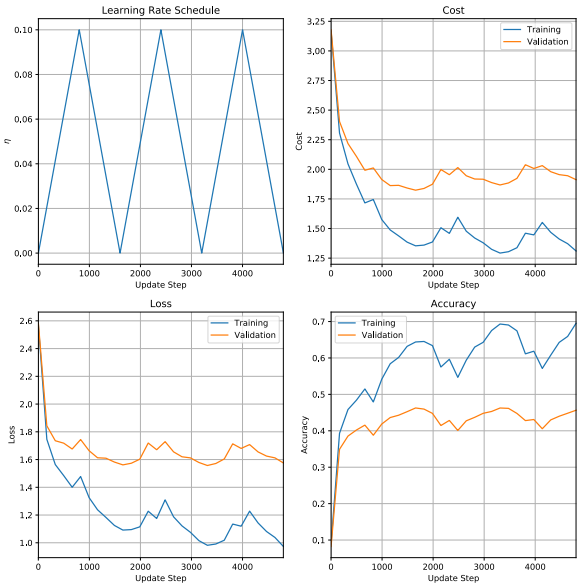

Figure 3: Learning rate schedule and learning curves for $\eta_{min} = 1e-5$ $\eta_{max} = 1e-1$, $\lambda = 0.01$ and $n_s = 800$. 10 datapoints per cycle.

Figure 2 and Figure 3 show reproductions of the learning curves given in the assignment. The peaks in the loss and cost curves roughly correspond to the peaks in the learning rate schedule. Intuitively, I would assume that the learning process tends to reach something akin to a local minimum around the time the learning rate reaches one of its minima and that increasing the learning rate then allows the training to "break out" of this minimum. I.e. a small increase in training cost is incurred with the potential to find a better local minimum during the next cycle and so on. Figure 3 shows this very well, we first reach a minimum in the cost curve at around 1500 updates and then a slightly better minimum at around 3500 updates.

To find a good value for λ, I initially took 20 random samples s_i, $i \in \{1, \ldots, 20\}$ from the uniform distribution over the interval $[-5, -1]$ and then calculated 10^{s_i} for each one of them. I trained with batch size 100 and the recommended step size (which works out to 900 in this case) for two cycles.

Note that random search does not make a whole lot of sense when searching over a single parameter but 20 samples were nevertheless enough to get an impression of the overall influence of λ on the validation performance.

Figure 4 shows an overview of the sampled values of λ (here and throughout the code named *alpha*) and the corresponding validation set costs and accuracies after two cycles of training. Sadly, it seems that as in assignment one, there is no clear benefit to using weak regularization over no regularization at all. The best achieved validation accuracies and the corresponding values of λ are tabulated in Figure 5.

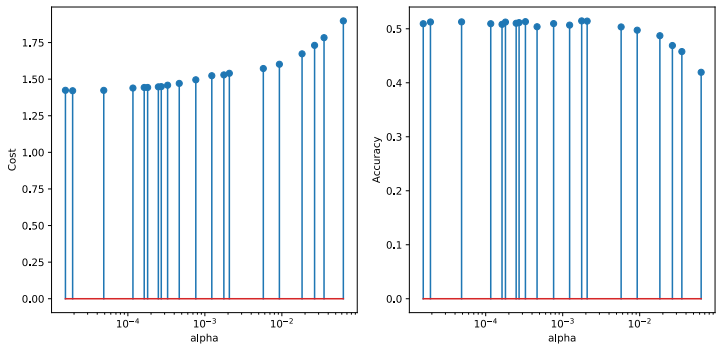

Figure 4: Coarse search results.

rank	λ	acc_{val}
1	0.00176	$\approx 51.5\%$
2	0.00208	$\approx 51.4\%$
3	3.281e-4	$\approx 51.3\%$

Figure 5: Coarse search best validation accuracies and corresponding values of λ.

I decided to perform the fine tuned search over the logarithmic range $[-3.5, -2, 5]$ because the two best results from the coarse search both fall within this range (although they are not significantly better then those achieved for very low values of λ). Figure 6 again shows the sampled values of λ and the corresponding achieved costs and accuracies and Figure 7 shows the best achieved validation accuracies and corresponding values of λ.

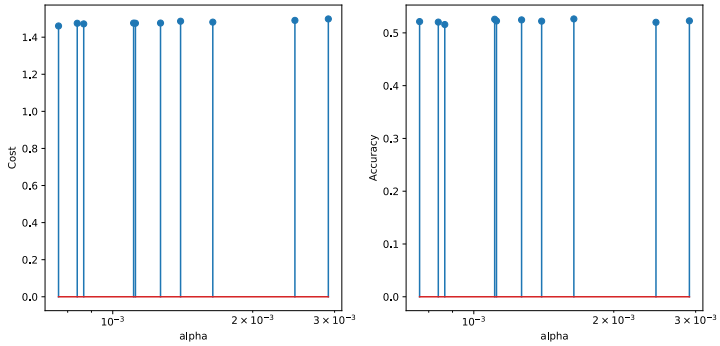

Figure 6: Fine grained search results.

rank	λ	acc_{val}
1	0.00164	$\approx 52.6\%$
2	0.00111	$\approx 52.6\%$
3	0.00127	$\approx 52.5\%$

Figure 7: Fine tuned search best validation accuracies and corresponding values of λ.

Using the best value of λ (≈ 0.00164) I trained on 49.000 training samples with step size 980 for three cycles. Figure 8 shows the corresponding learning curves and Figure 9 shows the trained networks performance on the test set. At 52% it's already around 10% higher than the best network trained in the last assignment.

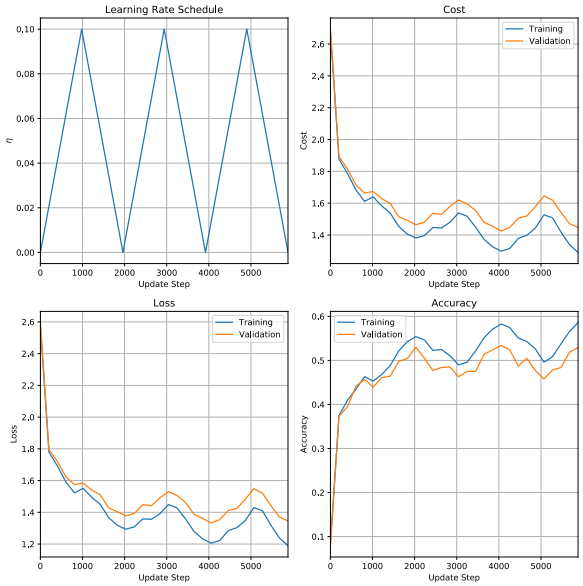

Figure 8: Learning curves for the final network with optimized regularization parameter trained on 49.000 training samples.

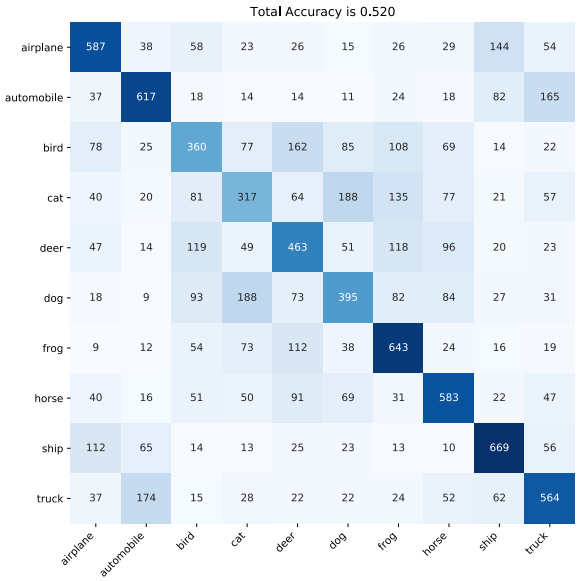

Figure 9: Test set performance of the final network with optimized regularization parameter trained on 49.000 training samples.