DD2424 - Assignment 1

Timo Nicolai

April 11, 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.

∇W

data dimensions	batch size	λ	$\epsilon_{rel,max}$
20	1	0	1.31e-7
20	1	0.5	4.69e-5
20	20	0	5.24e-5
20	20	0.5	5.14e-3

∇b

data dimensions	batch size	λ	$\epsilon_{rel,max}$
20	1	0	2.25e-7
20	1	0.5	2.25e-7
20	20	0	2.38e-5
20	20	0.5	2.38e-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 weight matrix initialization. Numerical gradient obtained via simple forward difference with $h = 1e - 6$.

Figure 1 shows the maximum relative error over all gradient elements for different batch sizes and regularization parameters. All of these are reasonably small so I assume my implementation is correct.

Figure 2 shows learning curves for the requested hyperparameter settings. The first set of curves clearly shows that if the learning rate is chosen too large, loss (and thus accuracy) may not converge smoothly or at all. While we initially still observe a downwards trend in the loss for $\eta = 0.1$, the learning curve fluctuates strongly and it seems unlikely that training will reach a good local minimum.

Increasing λ decreases the distance between training and validation set error. Intuitively, regularization makes it harder for the training algorithm to overfit on the training set. Another consequence of this is that the training error does not converge as quickly. Ideally the first of these effects should dominate the other so that regularization results in an overall decrease in validation error. Here however we observe that higher regularization leads both to higher training and validation errors, possibly because the model is not very complex and the training set is not that large.

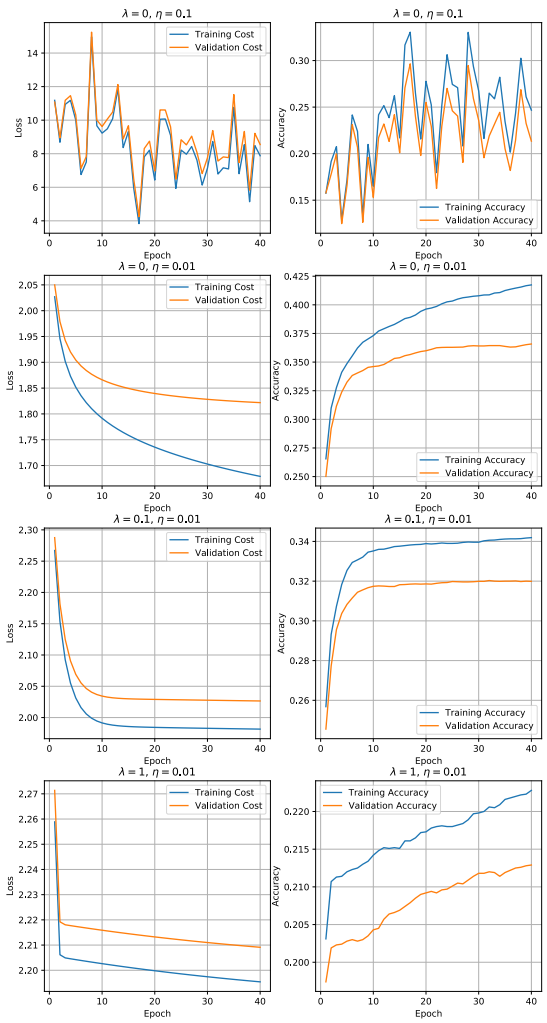

Figure 2: Learning curves for different learning rates and regularization parameters.

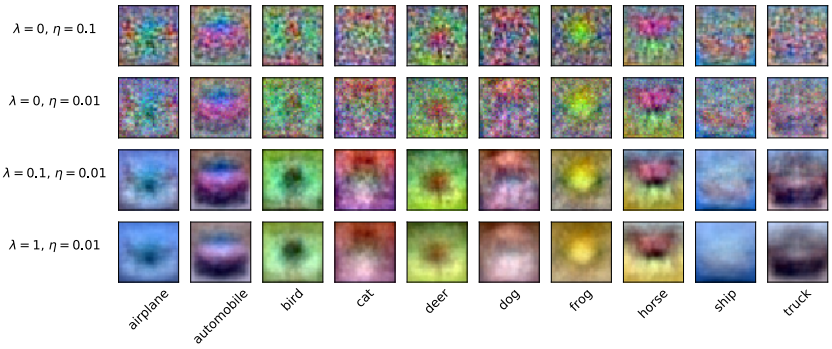

Figure 3: Image representations of weight matrices of different networks trained for 40 epochs each.

Figure 3 shows the weights learned by the different networks. Observe how increasing the amount of regularization "smoothes out" the weights.

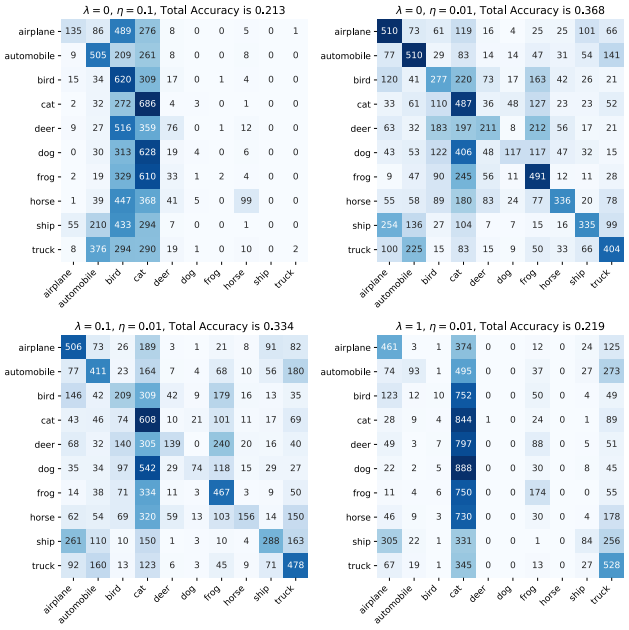

Figure 4: Confusion matrices and total test set accuracies of different networks trained for 40 epochs each.

Figure 4 displays confusion matrices and total test set accuracies for all four networks. The best performance (with an accuracy of 0.367) could be achieved by setting η to 0.01 and λ to 0.