DD2424 Deep Learning in Data Science
Assignment 2

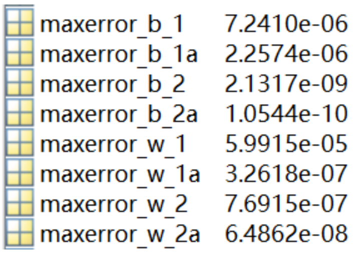
Lingxi Xiong
lingxi@kth.se

7th April, 2019

1 Sanity check

Following the same error computing function in assignment, the maximum values in difference array/matrix of analytic gradient and numerical gradient of the example data set $trainX(1:20,1:2), trainY(:,1:2), W1(:,1:20), W2$ are shown in Figure 1, where the first variable stands for the maximum error of analytic gradient of b compared with the faster numerical gradient computing method, the second one is the maximum error of analytic gradient of b compared with the more accurate version of numerical gradient computing method, and so on. The maximum value of them is sufficiently small.

What's more, from the Figure 2 we can see that the training accuracy of the network with a fraction of the data, i.e. the first 100 samples in the training data, can achieve 100% by setting learning rate to 0.01 and number of epochs to 200. The training loss can also be close to zero at the same time. In consideration of the above two approach, we can safely draw the conclusion that the gradient computations and mini-batch gradient descent algorithm are okay.



maxerror_b_1	7.2410e-06
maxerror_b_1a	2.2574e-06
maxerror_b_2	2.1317e-09
maxerror_b_2a	1.0544e-10
maxerror_w_1	5.9915e-05
maxerror_w_1a	3.2618e-07
maxerror_w_2	7.6915e-07
maxerror_w_2a	6.4862e-08

Figure 1: Analytic and numerical gradient computing comparison

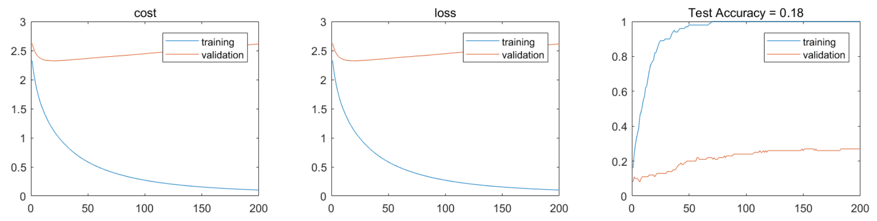


Figure 2: Overfitting on a smaller dataset

2 Cyclical learning rate

By setting min_eta = 1e-5, max_eta = 1e-1, n_s = 500, batch size = 100, lambda = 0.01 and train for one cycle, the Figure 3 we get has almost the same curves of training and validation cost, loss and accuracy with the ones in the tutorial.

By changing the number of step n_s of above setting to 800 and number of cycles to 3, we get the training curves shown in Figure 4. Here we see clearly the humps/peaks in cost and loss curves and the dips in accuracy curves. And both training set and validation set have the same trend.

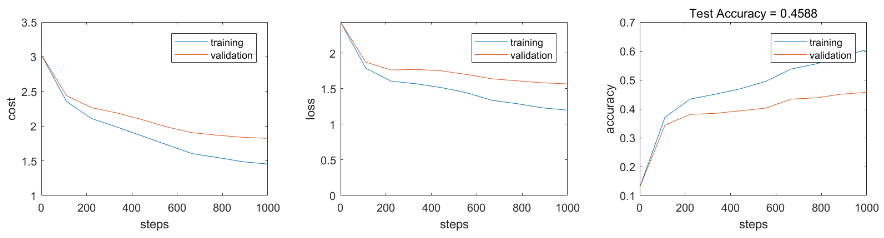


Figure 3: Training curves (cost, loss, accuracy) for one cycle of training

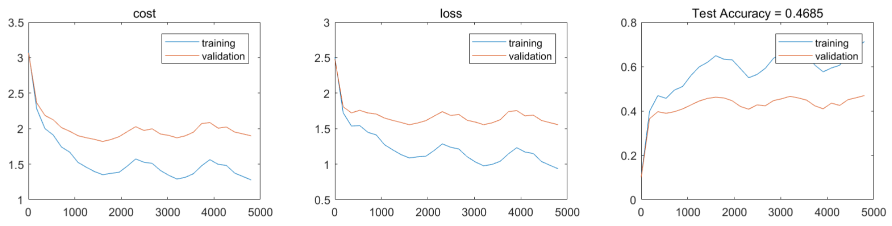
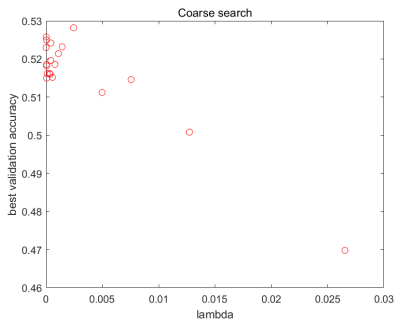


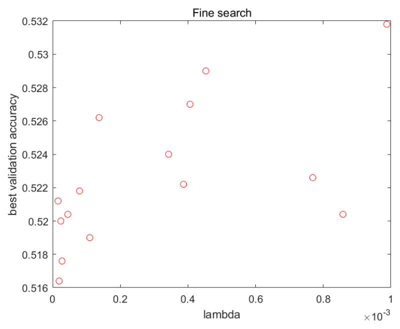
Figure 4: Training curves (cost, loss, accuracy) for three cycles of training

3 Coarse search

During the coarse search, the min and max eta of 1e-6 and 1e-1 are applied. And in all of the 20 times of random search, 2 cycles and 900 step number and 100 batch size keep the same. Figure 5(a) shows the distribution of the result. Among these, the best 3 lambda are 0.0024, 1.1174e-05 and 3.9162e-05, whose best accuracy on validation set are 0.5282, 0.5258 and 0.5250, respectively.



(a) Coarse search



(b) Fine search

Figure 5: Result of coarse-to-fine search

4 Fine search

Learned from the coarse search, the range of lambda in fine search is 1e-5 to 1e-3. Except for increase the number of cycles used for training to 3, the other hyper-parameters setting remain the same. The same plot is shown in Figure 5(b), where lambda value of 9.8774e-04, 4.5284e-04 and 4.0604e-04 achieve the best accuracy result of 0.5318, 0.5290 and 0.5270 respectively.

5 Train network with the best setting

Figure 6 plots the training and validation loss. In this case, the network is trained on all the training data (all the batch data), except for 1000 examples in a validation set, for 3 cycles, where the data set is slightly larger than the above task. Therefore, I round the best lambda a bit down to 9.87e-4 for the final training. The learnt network has test accuracy of 51.57% on test set.

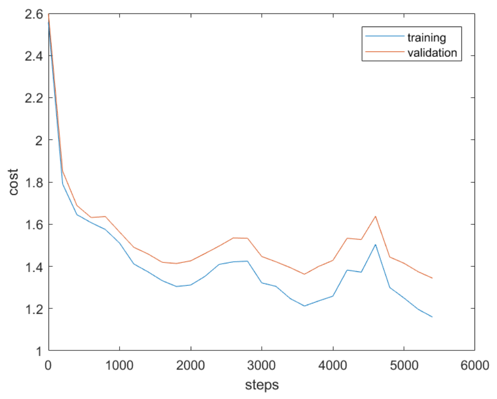


Figure 6: Training and validation loss