# DD2424 A2

Harsha Holenarasipura Narasanna *harshahn@kth.se*

1. State how you checked your analytic gradient computations and whether you think that your gradient computations were bug free. Give evidence for these conclusions.

Gradient for the network parameter theta: *W* and *b* were implemented based on the analytical solution. Correctness of the computed analytical gradients are validated against numerically computed gradients(fast) measured in terms of relative error. Relative error is computed from the *rerr(ga, gn)* function. *ga*: Analytical gradient & *gn*: Numerical gradient.

$$\frac{|g_a - g_n|}{max(eps, |g_a| + |g_n|)}$$ where eps is a small number;

Results were obtained as follows:

For feature size = 20, n = 2 images, h = 1e-5

| **Relative error** | W1 | b1 | W2 | b2 |
|---|---|---|---|---|
| Lambda = 0 | 2.19e-06 | 6.31e-06 | 5.83e-07 | 1.99e-06 |
| Lambda = 0.1 | 9.64e-05 | 6.31e-06 | 1.18e-04 | 1.99e-06 |
| Lambda = 1 | 1.15e-04 | 6.28e-06 | 1.04e-04 | 1.99e-06 |

Relative error values are very low. Thus, the correctness of the implementation of analytical gradient is ensured. Even with significant value of hyperparameter *lamda* the relative error remained low.

2. The curves for the training and validation cost when using the cyclical learning rates with the default values, that is replicate figures 3 and 4. Also comment on the curves.

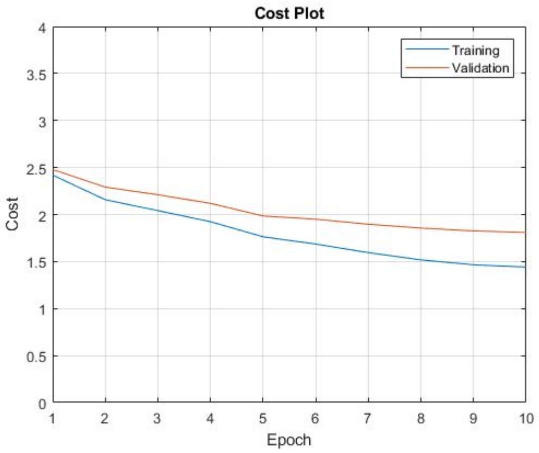

Figure a
Cost plot with cyclic learning rate strategy

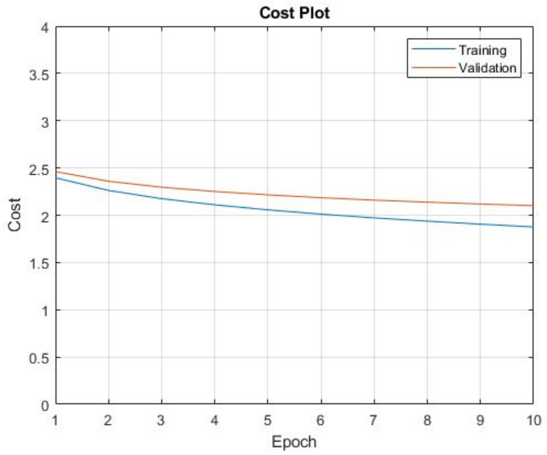

Figure b
Cost plot without cyclic learning rate strategy

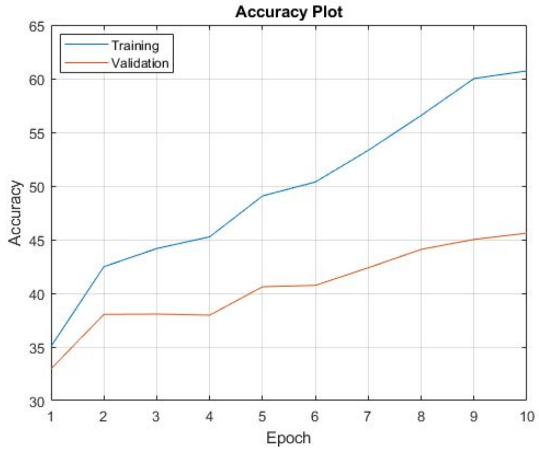

Figure c
Accuracy plot with cyclic learning strategy

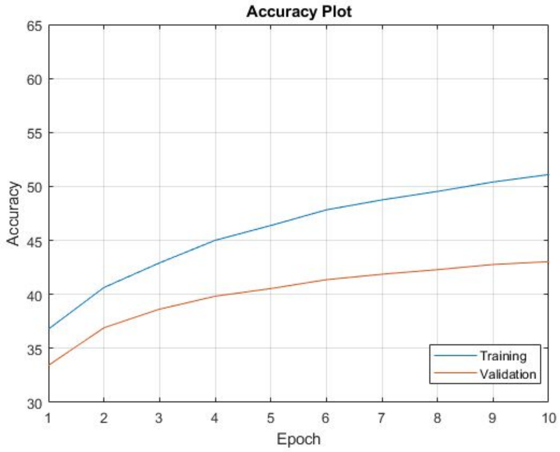

Figure d
Accuracy plot without cyclic learning strategy

Note: Instead of updates, epoch is considered for figure a & b, due to my PC not handle the load of cost evaluation at each update. Here each epoch corresponds to 100 updates. At each epoch the batches were shuffled. The hyper-parameter settings of the training algorithm are eta_min = 1e-5, eta_max = 1e-1, lambda=0.01 and n_s=500.

Training is the crucial part of neural network. Performance of this system is based on the convergence of the network parameters to its optimal values corresponding to minimal cost. Cyclical learning rate is a strategy which accelerates the rate of convergence.

From the above graphs it's evident that with the application of cyclical learning rate strategy, better accuracy & lower cost are achieved in the same limited time by huge margin. Graph also indicates the divergence in accuracy and cost amongst validation and training set, however, it is the question about regularization whose value *lambda* needs to be selected properly. Total time taken in either cases were ~ 227s.

Accuracy plot and eta values captured at each update is shown below. Total time ~ 420s.

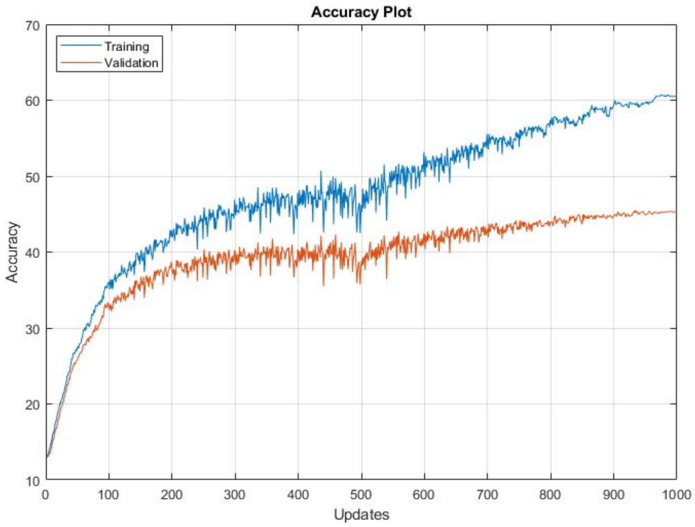

Figure e
Accuracy plot with cyclic learning strategy

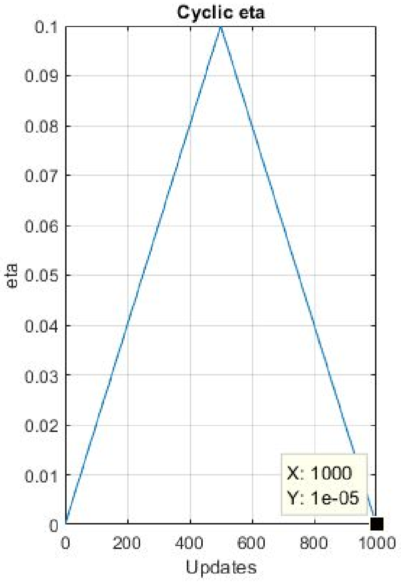

Figure f
eta values at each update

Similarly for the hyper-parameter settings eta_min = 1e-5, eta_max = 1e-1, lambda=0.01 and n_s=800, 3 cycles. The graphs are shown below.

Extension to the previous case, learning was carried out until 3 cycles. Significant rise from 60% training accuracy to 72% for two extra cycle is huge computational leap. End of training, 47.02% of accuracy on test data was achieved.

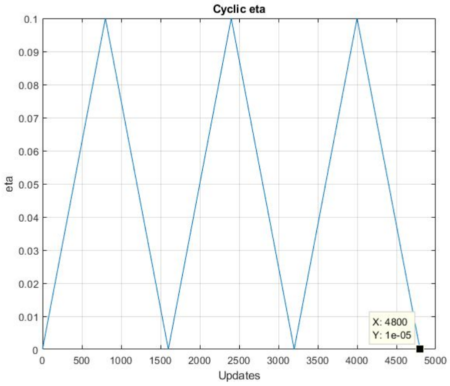

Figure e
eta values at each update

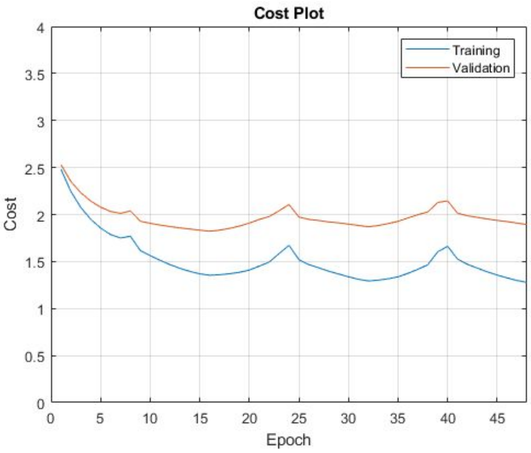

Figure a
Cost plot with cyclic learning rate strategy

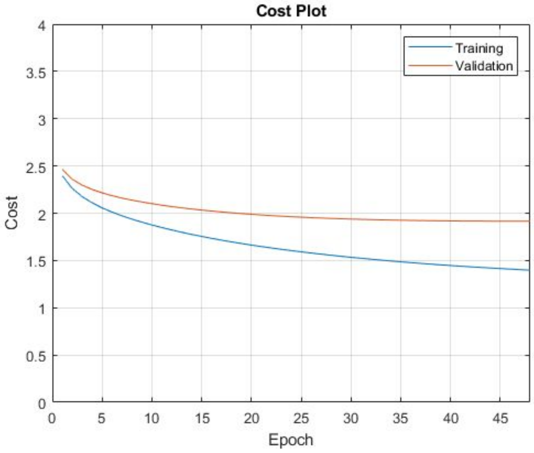

Figure b
Cost plot without cyclic learning rate strategy

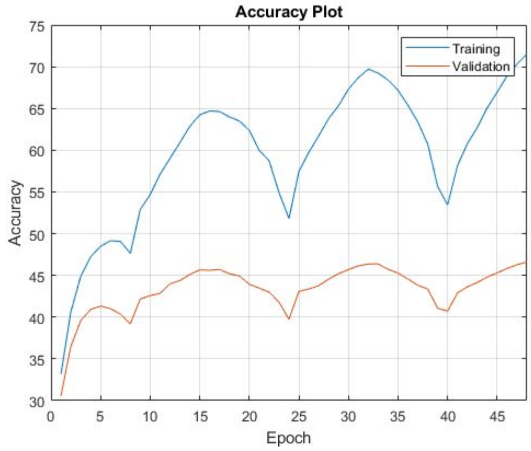

Figure c
Accuracy plot with cyclic learning strategy

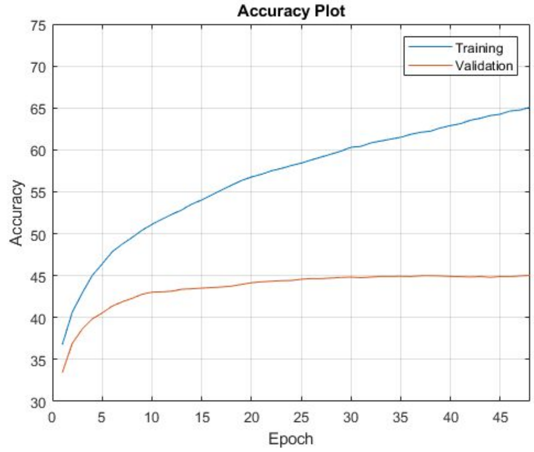

Figure d
Accuracy plot without cyclic learning strategy

3. State the range of the values you searched for lambda, the number of cycles used for training during the coarse search and the hyper-parameter settings for the 3 best performing networks you trained.

Range of *lambda* 1e-5 to 1e-1.5, range of *eta* 1e-5 to 1e-1 cyclic, 2 cycles, batch_size = 100, ns = 800, Layers = 2, Hidden nodes = 50. Graph below is a plot of *Accuracy* vs. *lambda*

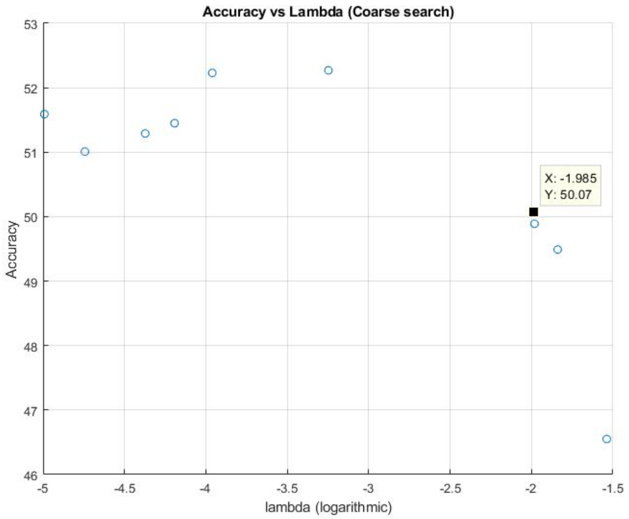

Fig 2. Coarse search: Accuracy vs. Lambda Plot.

Data set: Train set = 40,000 (PC hardware limitations), Validation set = 5000, Test set = 10,000. Best performance were observed for *lambda* = 1e-3.25, 1e-4 and 1e-4.25 with accuracy above 51% with peak of 52.22% around 1e-3.25 for the validation set.

4. State the range of the values you searched for lambda, the number of cycles used for training during the fine search, and the hyper-parameter settings for the 3 best performing networks you Trained.

Range of *lambda* chosen in the vicinity of peak accuracy observed in the coarse search i.e., about 1e-3.25 and thus, the fine search bandwidth is set as 1e-3.75 to 1e-2.75, range of *eta* 1e-5 to 1e-1 cyclic, 2 cycles, batch_size = 100, ns = 800, Layers = 2, Hidden nodes = 50. Graph below is a plot of *Accuracy* vs. *lambda*

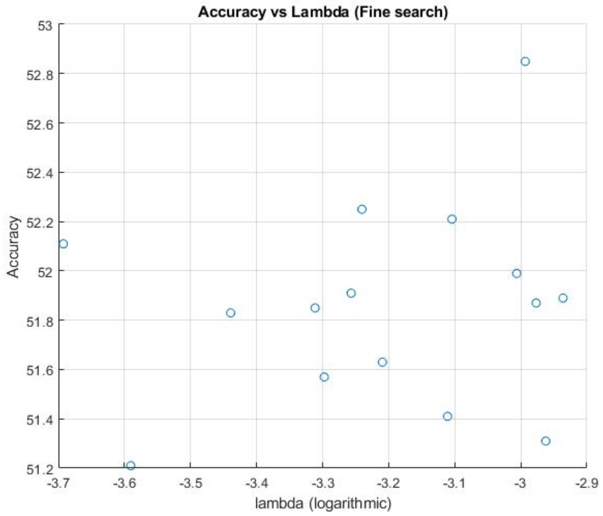

Fig 3. Fine search: Accuracy vs. Lambda Plot.

Data set: Train set = 40,000 (PC hardware limitations), Validation set = 5000, Test set = 10,000. Best performance were observed for *lambda* = 1e-2.98, 1e-3.07 and 1e-3.23 with accuracy all above 52% with peak of 52.81% at 1e-2.98 for the validation set.

5. For your best found lambda setting , train the network on all the training data, except for 1000 examples in a validation set, for 3 cycles. Plot the training and validation loss plots and then report the learnt network's performance on the test data.

Value of *lambda* chosen to be 1e-2.98, range of *eta* 1e-5 to 1e-1 cyclic, 3 cycles, batch_size = 100, ns = 800, Layers = 2, Hidden nodes = 50. Graph below is a plot of *Accuracy* vs. *lambda.*

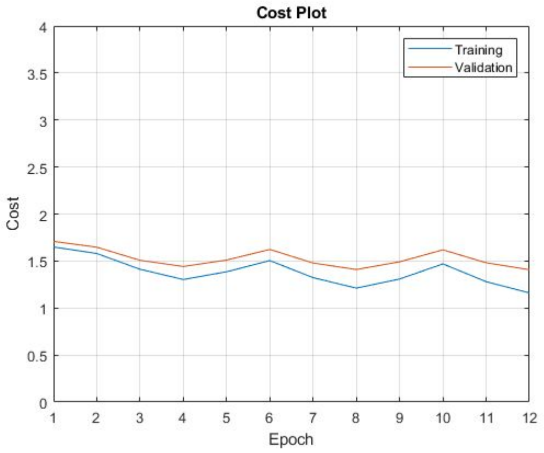

Figure a
Cost plot

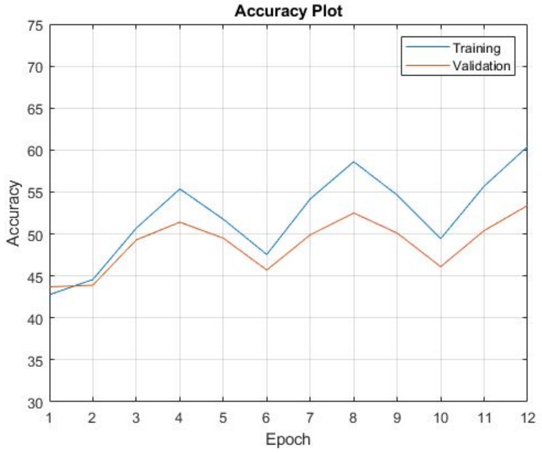

Figure b
Accuracy plot

Involved Epochs: 12, Itr: 1, Cycles: 3, Updates: 5880, Training data size: 49,000 images, Validation data size: 1,000. Test data size: 10,000. Final accuracy on test set: 52.28% and total time taken 941.33s

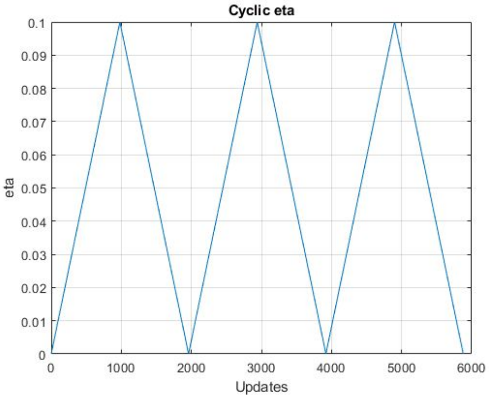

Figure c
eta values at each update