# DD2424 - Assignment 2


## Summary

I have implemented the cyclical learning rate with a coarse $\lambda$ search, followed by a finer $\lambda$ search, for which I achieved a final testing accuracy of 51.80% when training the network for 5 cycles with all of the training observations (except a small subset for validation) and `ns` $= 500$. For the finer $\lambda$ search, I used $\lambda_{\text{min}} = -5$ and $\lambda_{\text{max}} = -3$ when sampling the exponent for calculating $\lambda$. Additionally, I have also completed the bonus problems.

The code for the assignment has been written in `python`. I have implemented the neural network as a class. For the hand-in, all of the code has been put toghether in a main file with all the functions and the class declared at the top. For the hand-in, I have also commented out the saving of generated figures and results in JSON files, as well as omitting some of the case-specific testing and gradient testing.



Oskar Stigland

DD2424
Spring 2023

## Checking gradients

In order to validate the anaytic gradient computations, I compared the analytically computed gradients with numerically computed gradients

(a) a subset of the weights of a large subset of the training data, and

(b) all of the weights on a small subset of the training data

using $h = 10^{-5}$. For both (a) and (b), I got that

$$||\nabla_{\boldsymbol{w}} \boldsymbol{J} - \nabla_{\boldsymbol{w}} \boldsymbol{J}^{\text{numerical}}|| < 10^{-10}, \quad \forall \boldsymbol{w} \in \boldsymbol{W}$$

where $\boldsymbol{W}$ is the full set of weights, including also the weights in the bias vector. Then, I also attempted to train the network on a small subset of the training data with $|D| = 100$ and 500 epochs, not using any regularization. For this procedure, I got the following results:

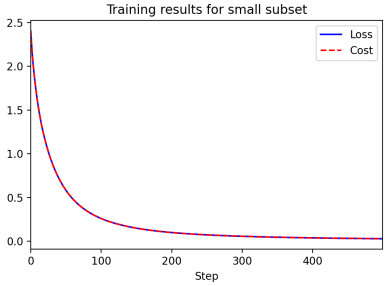


Figure 1: Loss and cost for $\eta = 0.01$, $\lambda = 0.0$, on $D \subset \boldsymbol{X}$, $|D| = 100$

Thus, I am confident the analytically computed gradients are indeed correct. They are identical to the numerically computed gradients up to an acceptable accuracy and the model clearly converges to a solution when trained on a small subset of the data, indicating that the gradient calculations are stable.

## Training with cyclical learning rates

For the initial parameter settings, i.e. using `nt`= 500 for 1 cycle and then `nt`= 800, and 3 cycles, the network achieves an accuracy on the testing data of 47.09% and 48.34%, respectively. The results ara shown in Figure 2 and 3. Overall, there are clear cyclical components in all metrics and the results seem to agree with those shown in Figure 3 and 4 of the assignment description.

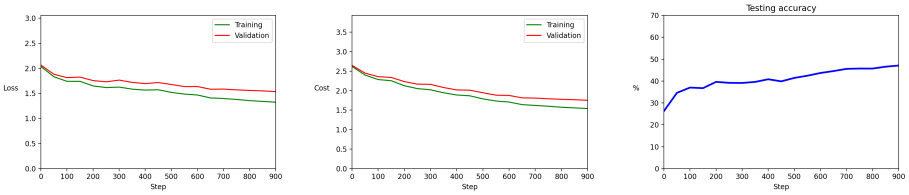


Figure 2: Loss, cost and accuracy for `etaMin`= $10^{-5}$, `etaMax`= $10^{-1}$, `nt`= 500, and 1 cycle.

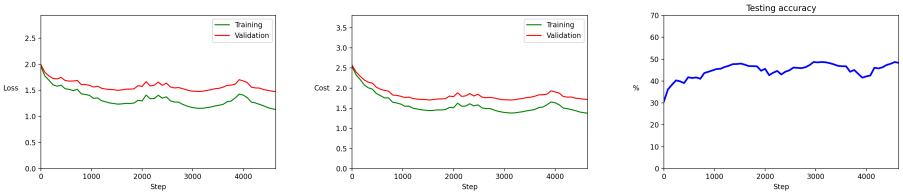


Figure 3: Loss, cost and accuracy for `etaMin`= $10^{-5}$, `etaMax`= $10^{-1}$, `nt`= 800, and 3 cycles.

## Searching for $\lambda$

When searching for an optimal value for the regularization parameter, $\lambda$, I computed `nt` based on the size of the training data, which was $|D| = 45000$, and `nBatch`= 100, such that `nt`= 450. I then trained for 2 cycles for each value of $\lambda$, which was calculated using

$$\lambda = 10^l, \quad l = \lambda_{\min} + (\lambda_{\max} - \lambda_{\min}) \cdot \boldsymbol{u}, \quad \boldsymbol{u} \sim \text{Uniform}(0, 1)$$

where $\lambda_{\min} = -5$ and $\lambda_{\max} = -1$. The search was implemented with an initial coarse search, where 20 different values of $\lambda$ were sampled. Then, the results from the coarse search were used to conduct a finer search. For the coarse search, the network seems to have performed worse with a higher $\lambda$, i.e. $\lambda > 0.01$. I have summarized the top-performing $\lambda$ settings in the following table. The same seed was used for all iterations.

| $\lambda$ | Accuracy, % |
|---|---|
| 0.0005 | 51.16 |
| 0.0003 | 51.15 |
| 0.0026 | 51.14 |

Hence, for the fine search I instead set $\lambda_{\min} = -5$ and $\lambda_{\max} = -3$. The result for the three top-performing networks are shown in the table below. The performance gains are somewhat modest, with the best accuracy found in the fine search beats the one from the coarse search with only +0.15%. Finally, I trained a network for 5 cycles, with `nt`= 500 and $\lambda = 0.0263$, for which the final accuracy on the test data was 51.80%. The results are plotted below in Figure 4.

| $\lambda$ | Accuracy, % |
|---|---|
| 0.00263 | 51.31 |
| 0.0009 | 51.23 |
| 0.002 | 51.18 |

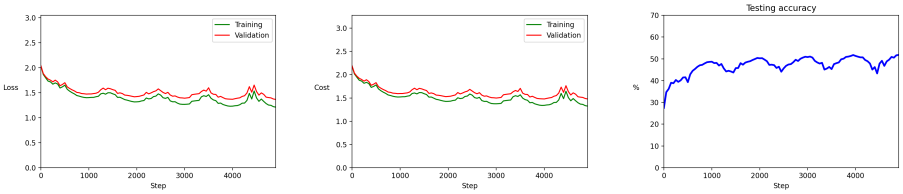


Figure 4: Loss, cost and accuracy for $\lambda = 0.0263$, `etaMin`= $10^{-5}$, `etaMax`= $10^{-1}$, `nt`= 500, and 5 cycles.