Assignment 3 - Report

Pietro Alovisi

Gradient Check

To check the gradient I computed for each layer the maximum of the absolute difference of the gradients, as in the equations below:

$$max(|W_i - W_{i\ num}|)$$
$$max(|b_i - b_{i\ num}|)$$

Where $\cdot_{num}$ represents the numerical computed value. The

Table 1: Maximum absolute difference between the computed gradients for the parameter of each layer (first column) for different network configuration. The lists like $[10, 10, 10]$ mean that there are 3 hidden nodes with 10 nodes each. For all the networks λ was set to 1.0.

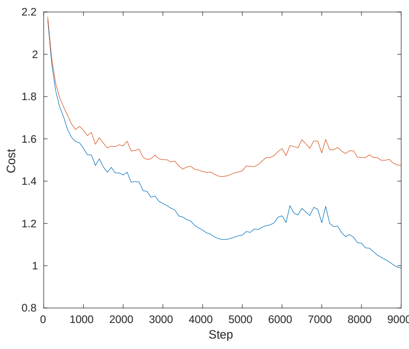
Inner nodes $\rightarrow$	$[50]$	$[20, 10]$	$[10, 10, 10]$
W_1	7.01e-10	8.33e-10	6.71e-10
b_1	3.49e-10	2.33e-10	3.18e-10
W_2	5.51e-10	5.48e-10	4.97e-10
b_2	3.12e-10	2.86e-10	3.18e-10
W_3	-	4.58e-10	5.77e-10
b_3	-	3.05e-10	1.80e-10
W_4	-	-	4.90e-10
b_4	-	-	3.36e-10

Test Multi-Layer

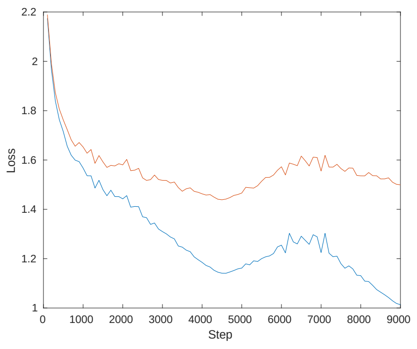
As suggested on the instructions I created a network with 2 hidden layers of 50 neurons each. Then I trained it on 45000 samples and validate it on 5000. The parameters I used were the one stated in the instructions : $eta_min = 1e-5$, $eta_max1e-1$, $n_s = 5 \cdot 450$ $\lambda = 0.0001$ (but I guess thre was a typo, n_s

was supposed to be $n_s = 2 \cdot 450$).

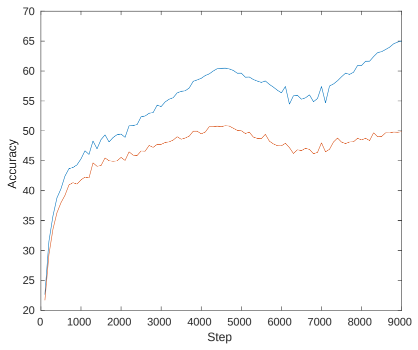
The loss and the accuracy throughout the 2 cycles of training are shown in Figure 1. The accuracy I got on the test data is 52.6%.



(a) Cost Fuction.



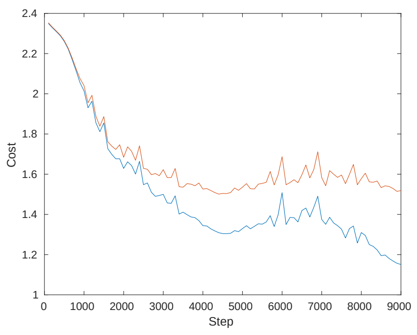
(b) Loss Fuction.



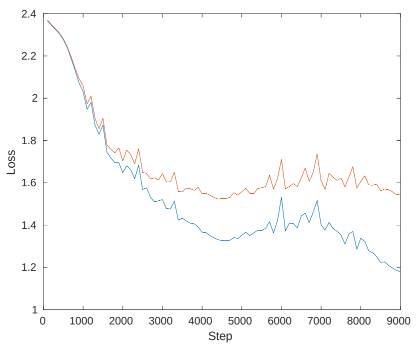
(c) Accuracy of the network (%).

Figure 1: Training evolution of cost loss and accuracy on test and validation sets for the 2 layer network, with parameters $eta_min = 1e-5$, $eta_max1e-1$, $n_s = 5 \cdot 450$ $\lambda = 0.0001$, run for 2 cycles.

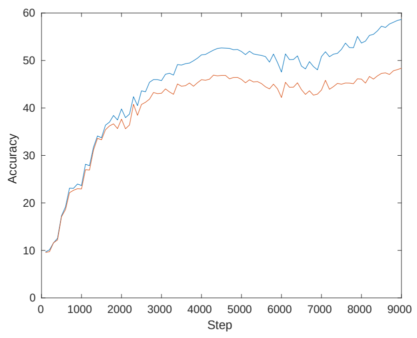
Then I tested with the same hyperparameters the 9 layer network described by the sequence of hidden states $[50, 30, 20, 20, 10, 10, 10, 10]$. The performance I get is 43.27% (even though various rounds show a lot of variability in the result ranging from 40% to 45%). For completeness I also plot in Figure 2 the cost, loss and accuracy.



(a) Cost Fuction.



(b) Loss Fuction.



(c) Accuracy of the network (%).

Figure 2: Training evolution of cost loss and accuracy for the 9-layer network.

Batch Normalization

Gradient Computation

I repeated the process described above for comparing the numerical and the computed gradients. What I found was that both the absolute value and the relative error was in the order of $1e-2$ for every parameter. To check the correctness I just tried to train the network and see the evolution of the loss function. What I found was that the loss was decreasing and that the performance of the network was fine.

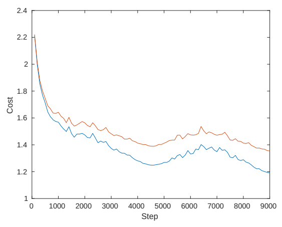
I still double checked the code but I could not find any error. The only concern I have is that the sample mean and variance is not constant in practice (as assumed in the formulas), so this is what could cause such variations in the gradient.

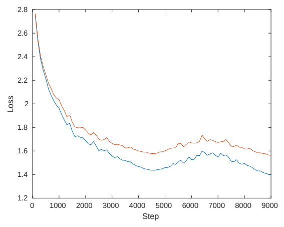
Testing the Network

Using the same parameters as for the multi-layer network tested before, I got the following results:

Network	Test Accuracy
3 layer	53.37%
9 layer	46.98%

For completeness I also reported in Figure 3 and Figure 4 the loss, cost, and accuracy for both the networks. As we can see batch normalization helps mantaining performance even for deep networks.


(a) Cost Fuction.


(b) Loss Fuction.
an accuracy of 51.81%

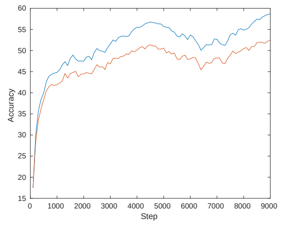
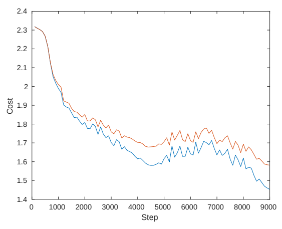
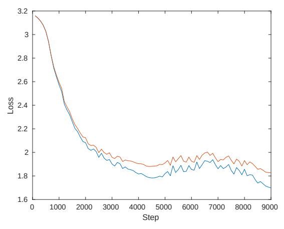

(c) Accuracy of the network (%).

Figure 3: Training evolution of cost loss and accuracy on test and validation sets for the 2 layer network with batch normalization.


(a) Cost Fuction.


(b) Loss Fuction.

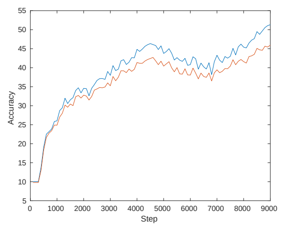

(c) Accuracy of the network (%).

Figure 4: Training evolution of cost loss and accuracy on test and validation sets for the 9 layer network with batch normalization.

Search for λ

I searched for the best value of λ to optimize the 3 layer network. I tested around 40 values between $1e-5$ and $1e-1$, and the search is depicted in Figure 5.

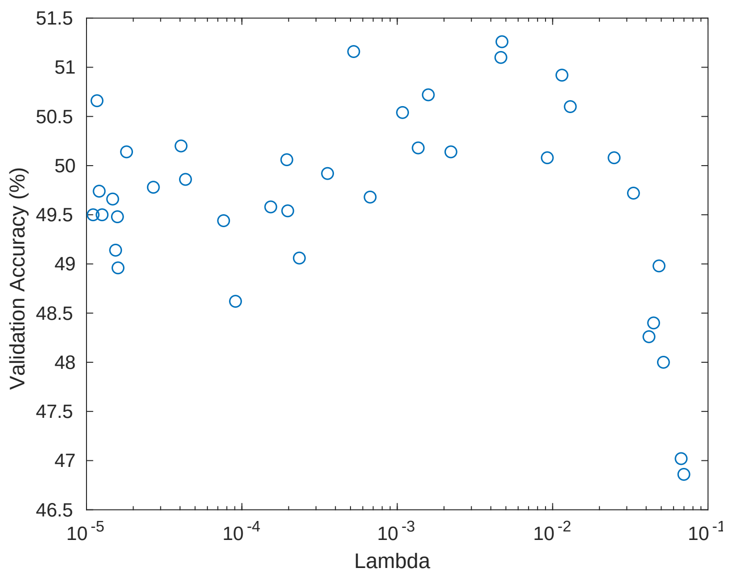


Figure 5: All the performances on the validation test for all the trials I made on the different values of λ.

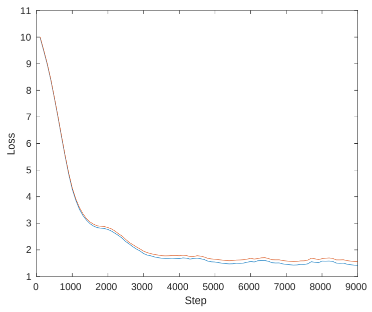
The best value for λ is 0.004718 which corresponds to an accuracy of 51.81% on the test set, and an accuracy of 51.26% on the validation set. The value is very close to the one given by the assignment instructions. Of course this search does not account for variability in the results, which can lead to higher performance scores.

Sensitivity to initialization

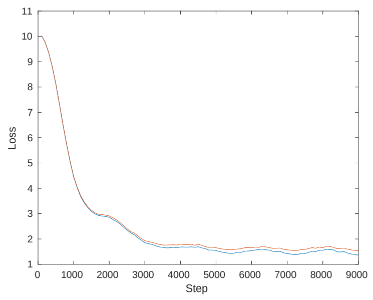
I thested the 3 layer networks with different standard deviation for the initial-izaiton parameter. The result for the different values of σ are reported in the table below, and the loss plots are shown in Figure 6.

σ	Multi-Layer	Batch Normalized
0.1	52.3%	51.1%
0.001	10.0%	50.45%
0.0001	10.0%	10.0%

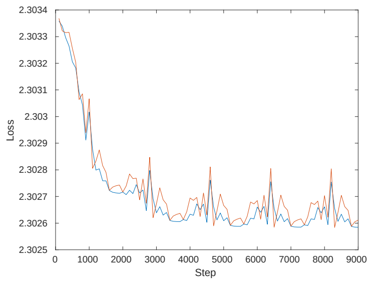
It is obvious how a very small value of σ nullifies the network, by having a random performance on the test set. Also we can see the effect of the initializaion in the losses which, as σ goes down, they get more spiky and more caotic, and enhancing the peakes in the plot caused by the cyclic learning rate.



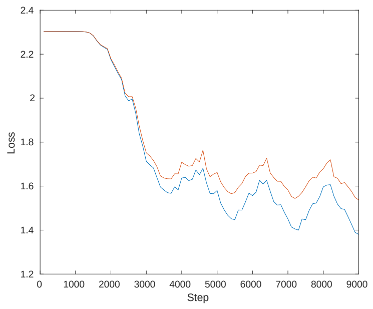
(a) Multi-Layer Network, $sig = 1e-1$.



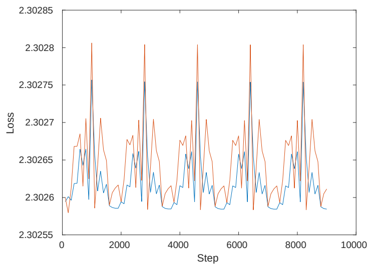
(b) Batch Normalization, $sig = 1e-1$.



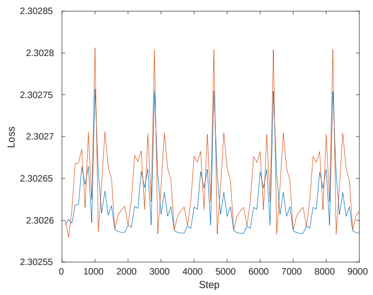
(c) Multi-Layer Network, $sig = 1e-3$.



(d) Batch Normalization, $sig = 1e-3$.



(e) Multi-Layer Network, $sig = 1e-4$.



(f) Batch Normalization, $sig = 1e-4$.

Figure 6: Loss plots for different values of sig and for the two types of network.