DD2424 – Assignment 3

Deep Learning

May 4, 2017

Addi Djikic – TSCRM1
addi@kth.se
941028-5473

Abstract

For this assignment a K-layer network were trained with multiple outputs to classify the images from the given CIFAR-10 dataset. This was done mostly analogy as in Assignment 2 from DD2424 and was done successfully with generalizing it with K-layers. Seed with rng(400) was used throughout the assignment.

K-Layer Network Classifier

Gradient Computations

To test that the gradient is correctly computed we can compare it with function that use numerical estimation, therefor we will use `ComputeGradNumSlow` and `ComputeGradNum` from [1] to compare it. The absolute difference between the maximum error values of the slow center difference method, and the faster finite difference method can be seen in the Table below with batch normalization and a three layer network. (When just testing the K-layer network with 2 layers without batch normalization the errors were exactly identical to what we achieved in Assignment 2.)

Compared Errors	W_1	W_2	W_3	b_1	b_2	b_3
Central Diff.	6.11e-11	1.20e-04	3.87e-11	2.89e-19	2.31e-18	2.44e-11
Finite Diff.	6.65e-08	2.40e-04	1.38e-09	2.89e-19	2.31e-18	4.50e-07

This was tested with reduced input data, with the first three images, which can be seen in the Matlab code for this report. I believe the gradient computations were correct at this point due to similar computations as in Assignment 1 and 2. The low values of the errors indicate a good gradient estimation.

Evolution With and Without Batch Normalization

The below two plots shows the training and validation loss when we use batch normalization, as in fig. 1 and without batch normalization as in fig. 2. Both used the parameters `eta` = 0.01 and `lambda` = 0 along with 10 epochs. We can clearly see that we are basically unable to train anything without batch normalization with more than a two layer network, the loss could be adjusted by tweaking the eta parameter, but no good result were would unless we used batch normalization.

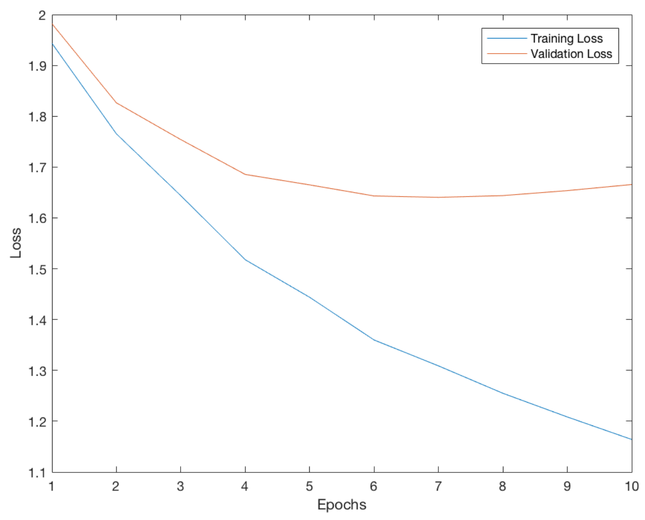

Figure 1: 10 epochs of training with batch normalization and moving average, an accuracy of 42.45% was achieved with parameters `eta = 0.01` and `lambda = 0`

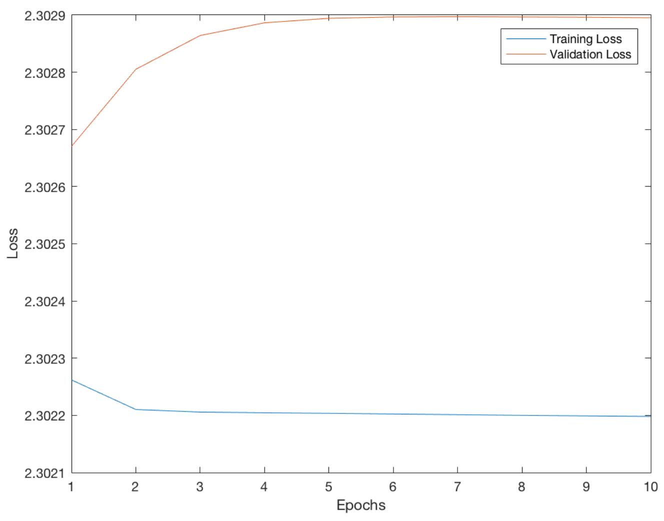

Figure 2: 10 epochs of training without batch normalization, an accuracy of around 10% was achieved with parameters `eta = 0.01` and `lambda = 0`

Search for Optimal Parameters

When the first coarse search were implemented the following parameters were firstly set:

LambdaMax	LambdaMin	EtaMax	EtaMin	Epochs	Nbr of Runs	Decay	ρ
0.01	0.000001	0.6	0.001	10	60	0.7	0.9

Where the following best three accuracy's were achieved after the first search:

Lambda	Eta	Accuracy
5.8250e-5	0.0264	43.00%
1.0677e-5	0.0398	43.00%
1.4918e-5	0.0194	42.65%

When narrowing the span down even more for fine search the following three results were then achieved:

Lambda	Eta	Accuracy
1.7911e-4	0.2511	44.68%
5.4253e-4	0.2474	44.41%
4.8331e-4	0.2100	44.41%

Where we can take the average between these best parameters and get optimal `lambda = 4.0165e-4` and optimal `eta = 0.2361`

When the best hyper-parameter setting was used with batch normalization an accuracy of **52.90**% was achieved with all the data used, except for 1000 images for validation from the Cifar-10 dataset.

Result With all Training Data

The plots bellow show how different learning rates affect the training and validation cost when using both batch normalization and without batch normalization. The learning rates used were eta = 0.001, eta = 0.1 and eta = 0.6. Along with regularization set to lambda = 0.000001 as we used in the start. The first three plots is with batch normalization and two-layer (M = 50) while the others are without batch normalization.

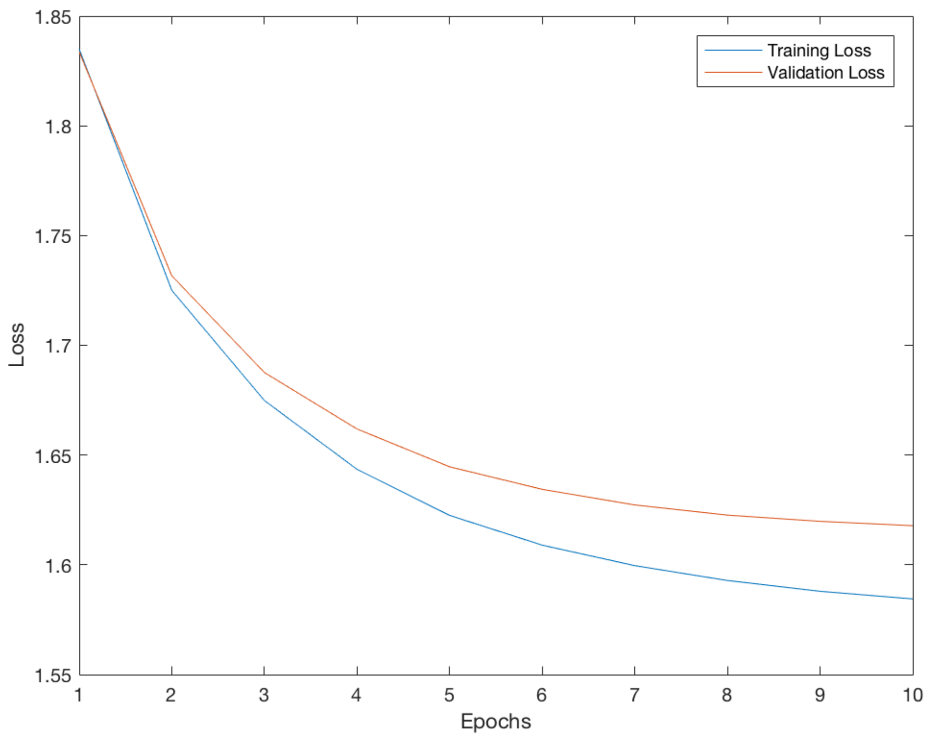

Figure 3: 10 epochs of training with all data when using low eta value and with batch normalization `eta = 0.001`, accuracy 43.7%

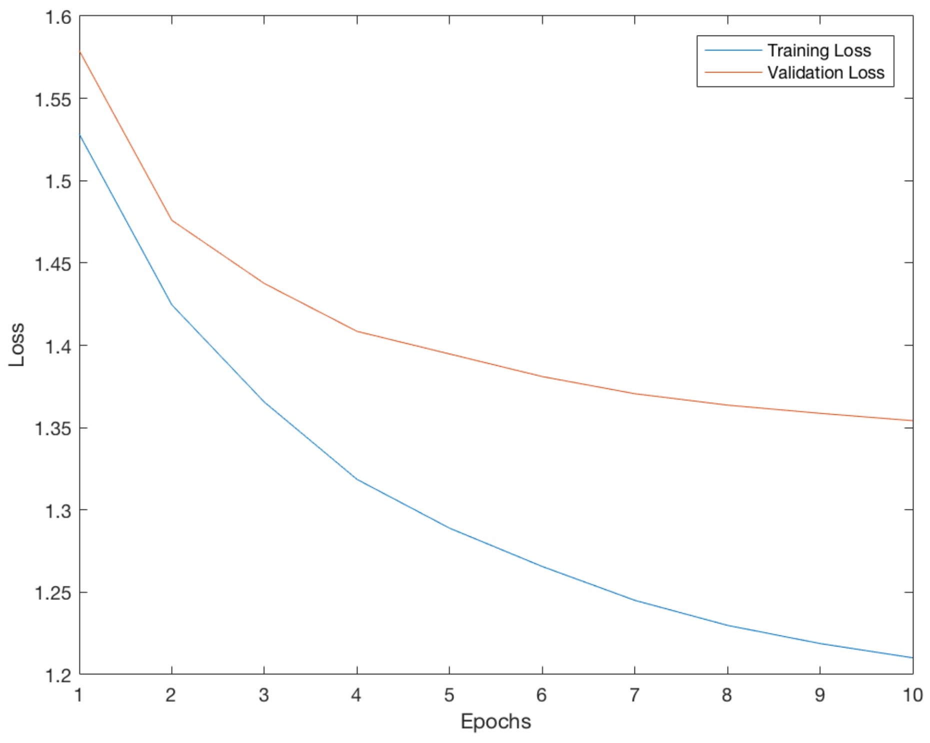

Figure 4: 10 epochs of training with all data when using low eta value and with batch normalization `eta = 0.1`, accuracy 51.7%

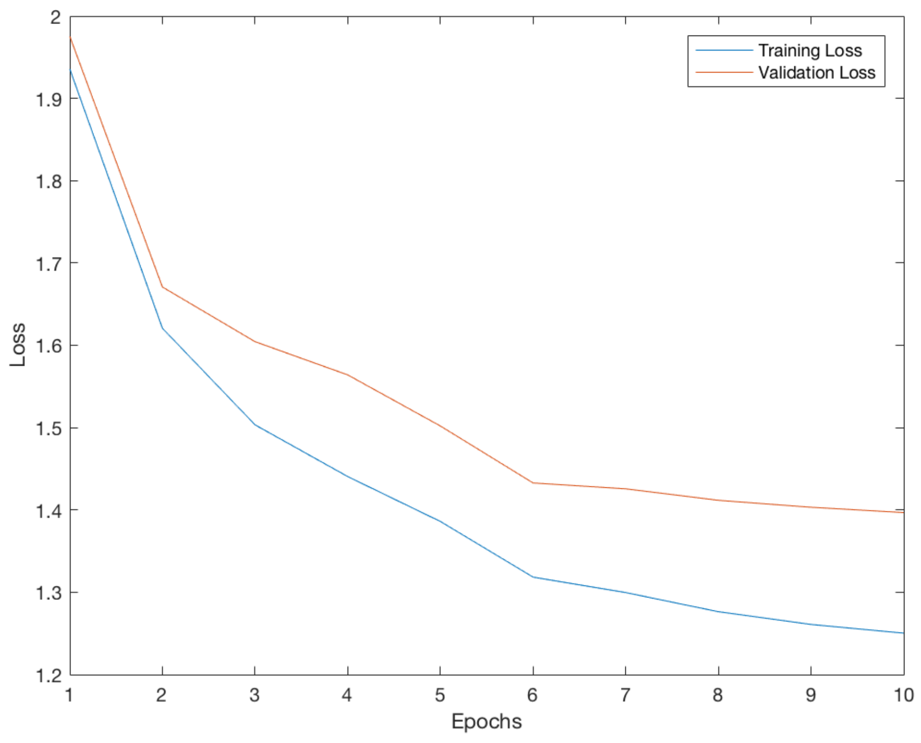

Figure 5: 10 epochs of training with all data when using low eta value and with batch normalization `eta = 0.6`, accuracy 51.6%

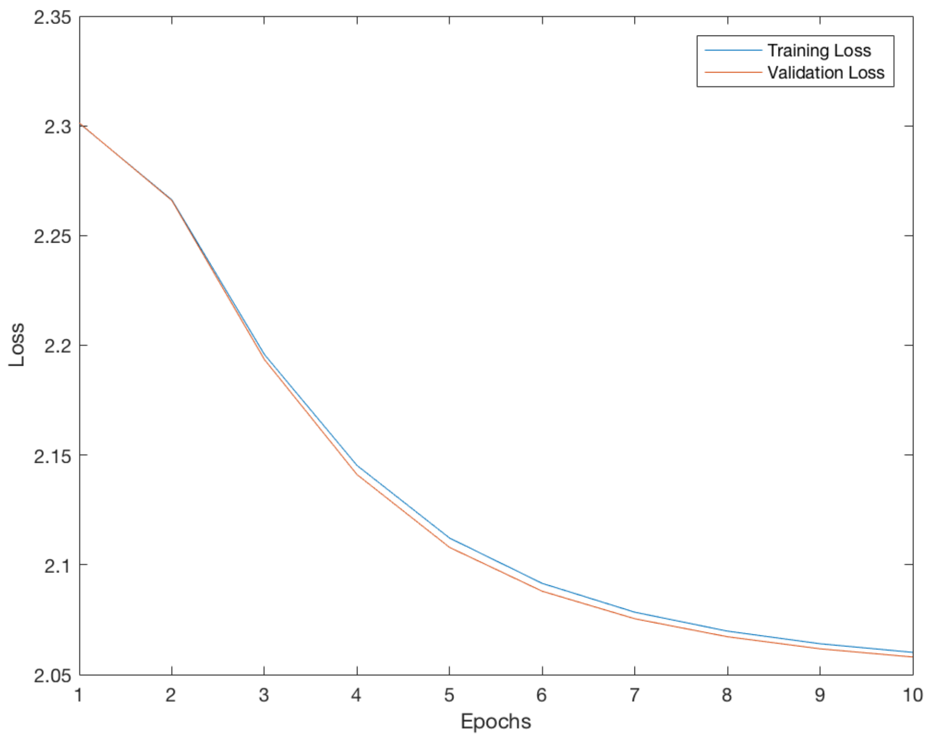

Figure 6: 10 epochs of training with all data when using low eta value and without batch normalization `eta = 0.001`, accuracy 26.1%

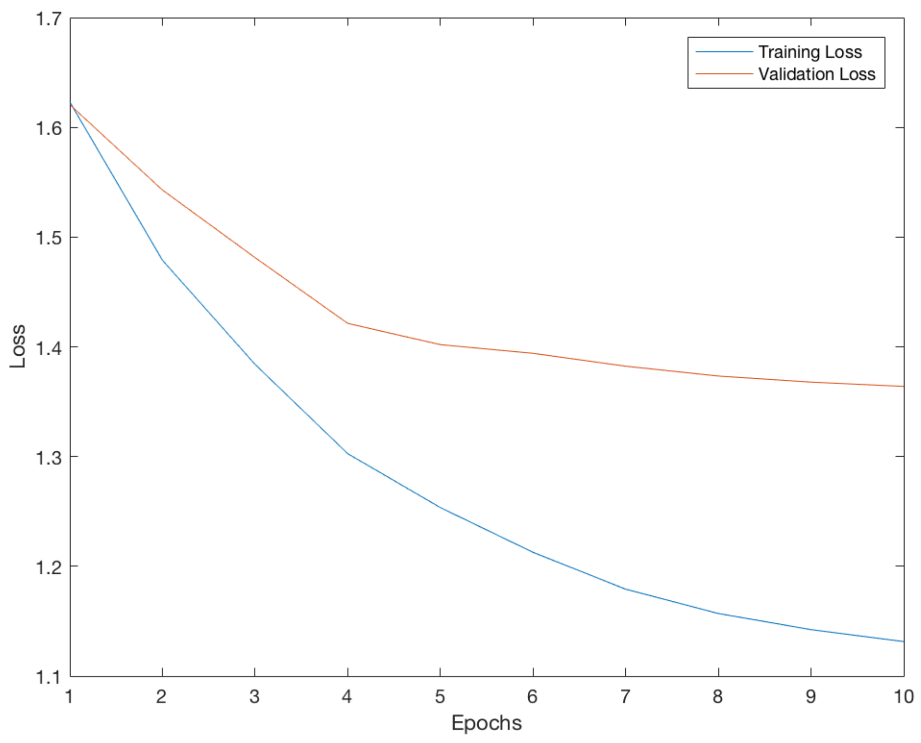

Figure 7: 10 epochs of training with all data when using low eta value and without batch normalization `eta = 0.1`, accuracy 52.7%

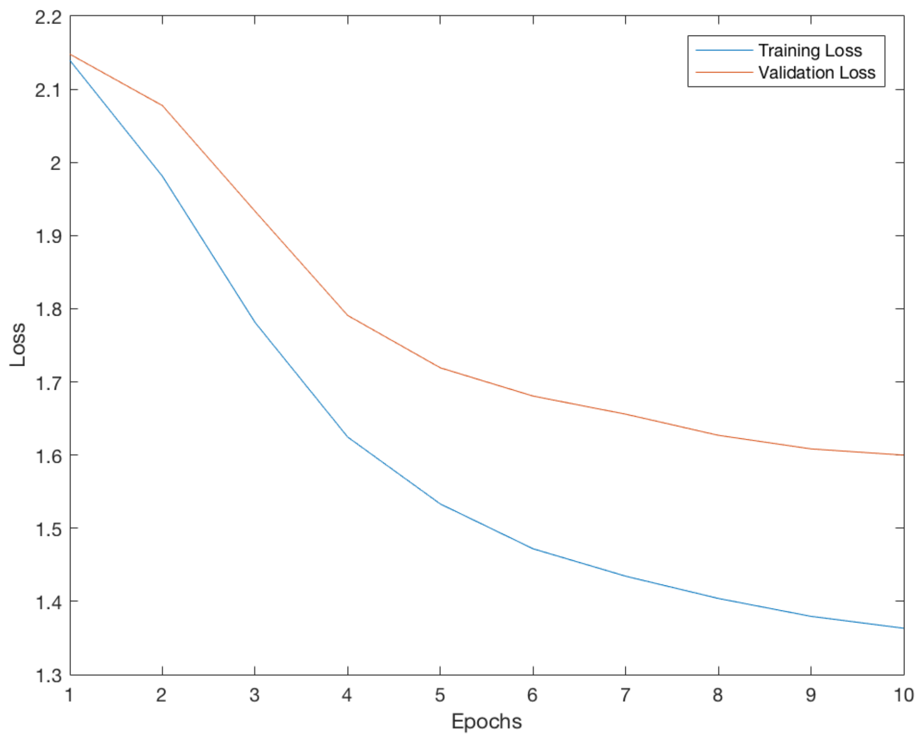

Figure 8: 10 epochs of training with all data when using low eta value and without batch normalization `eta = 0.3` (Which is not same as in with batch norm in this case due to avoid NaN values), accuracy 44.9%

References

[1] Assignment3 DD2424 Instructions paper:
https://www.kth.se/social/files/5901c393f2765417bcf31c93/Assignment3.pdf

[2] Lecture notes by Josephine Sullivan on DD2424 KTH: https://www.kth.se/social/course/DD2424/subgroup/vt-2017-958/page/lectures-164/