Introduction

This time we will generalize a full connected k-layer network that trains on the CIFAR-10 dataset. And compare the performance between a network with and without batch normalization.

The assignment

The comparison of the numerical and my own implemented gradient was done by measuring the Euclidean distances between the vectors and matrices. Where a small value indicates that the gradient computation was done correctly.

Euclidean distances of:	$\nabla W1$	$\nabla W2$	$\nabla W3$	$\nabla b1$	$\nabla b2$	$\nabla b3$
2-layer 50 nodes	0.744*10^ -4	0.7439*10 ^-4		0.5575*10 ^-5	0.0000*10 ^-5	
3-layer 50-30 nodes	0.1238*10 ^-4	0.1462*10 ^-4	0.1344*10 ^-4	0.0003*10 ^-4	0.5455*10 ^-4	0.0000*10 ^-4

Table 1. Euclidean distances between the gradients from ComputeGradsNumSlow and my own.

The comparison was done for different layers with certain number of nodes at each layer, see table 1. And they showed relative small distances. Thus, the gradient computations seemed working well.

The evolution of the loss function for a 3 – layer network, with and without batch normalization are shown below. We can clearly see that with batch normalization gave better performance.

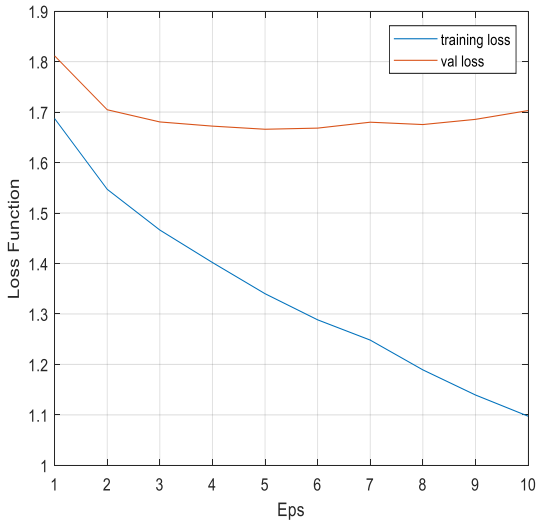

Fig 1. Loss with batch norm,
Train accuracy = 0.6306
Valid accuracy = 0.4224
Test accuracy = 0.4263

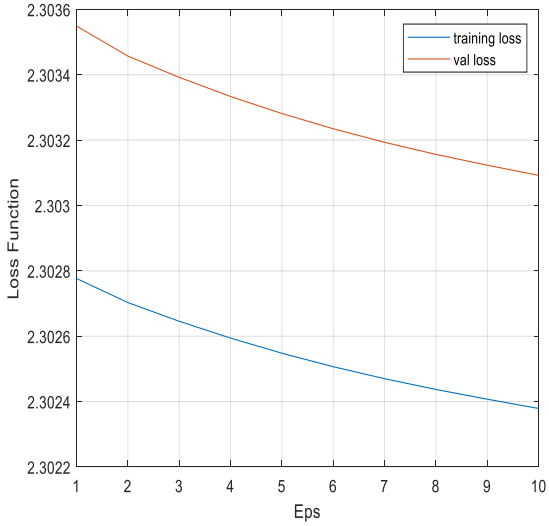

Fig 2. Loss without Batch norm,
Train accuracy = 0.1016
Valid accuracy = 0.0995
Test accuracy = 0.1000

With batch normalization on a 3-layer network with 50 and 30 nodes. For the coarse and fine search, I used 100 samples to evaluate the best performing model regarding the validation

accuracy. The range of the values used for coarse search: eta = [10^-3,10^1], lambda = [0, 10^-5]. And the range for the fine search by narrowing the range from the 2 best parameter combinations: eta = [0.03, 0.05], lambda = [7*10^-5, 5*10^-5].

The fine search gave the following best hyper-parameters: lambda = 5.9080e-05, eta = 0.0473. And the model showed the following performance by training on all the datasets with 10 epochs:

Training accuracy	Validation accuracy	Test accuracy
0.5818	0.5170	0.5044.

Table 2. Best accuracy from the 3-layer network.

With lambda = 1e-5, I plotted the training and validation loss for my 2-layer network with different eta values for with and without batch normalization.

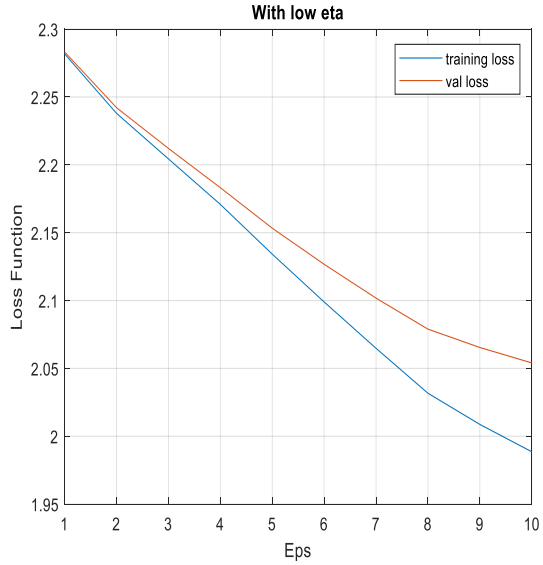

Fig 3. Loss with batch norm and eta = 10^-4.

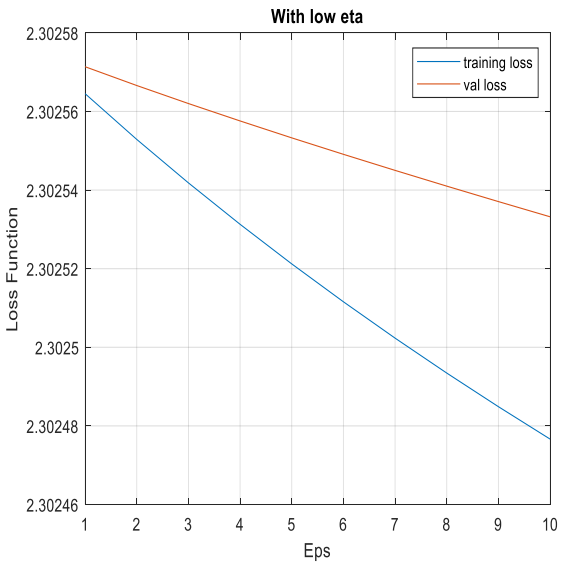

Fig 4. Loss without batch norm and eta = 10^-4.

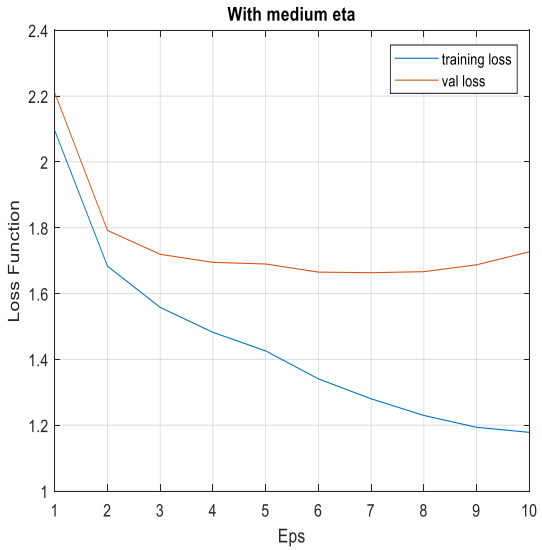

Fig 5. Loss with batch norm and eta = 10^-2.

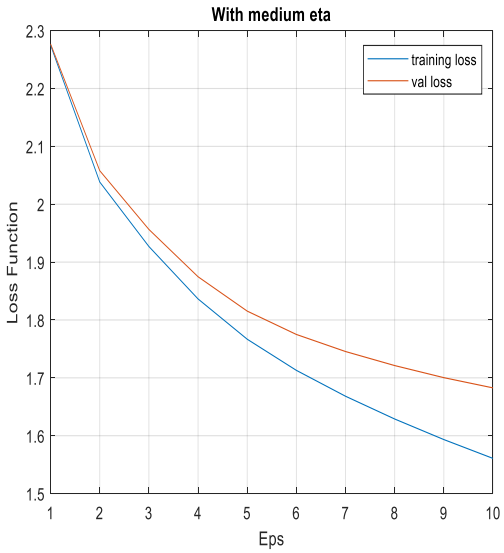

Fig 6. Loss without batch norm and eta = 10^-2.

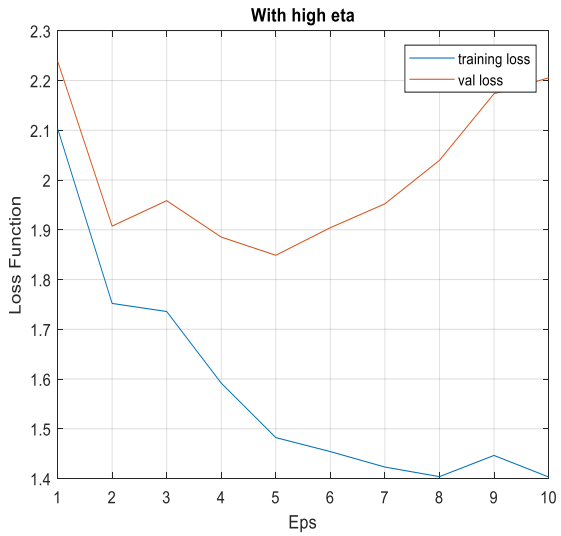

Fig 7. Loss with batch norm and eta = 1.

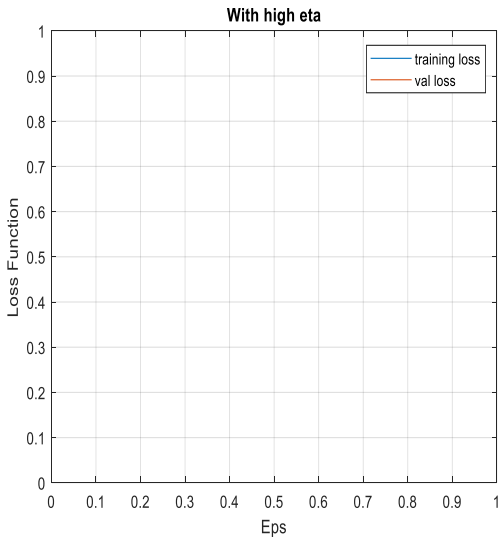

Fig 8. Loss without batch norm and eta = 1.

A conclusion that can be drawn by comparing fig 3-8, is that batch normalization increases the performance of the network which result in lower loss values. Last, batch normalization can handle higher learning rate where fig 7 shows the loss but compare to fig 8 which resulted in NaN.