DD2424 Assignment 2 Report

Student: Alberto Xamin xamin@kth.se

State how you checked your analytic gradient computations and whether you think that your gradient computations are bug free for your k-layer network with batch normalization.

The gradients were checked with a subset of the training data and with a network with 10 hidden nodes to speed up the comparison.

```
nnt = NeuralNet((train_X[:, 0:100], train_Y[:, 0:100], train_labels[:, 0:100]),
         mu, sigma, hidden_nodes=[10])
nnt.SanityCheck()

# where sanitycheck is defined as
def SanityCheck(self):
    x, y = self.data[0][:, 0:1], self.data[1][:, 0:1]
    anw, anb  = self.ComputeGradients(x, y, 0)
    print("an-done")
    numw, numb = self.ComputeGradsNum(x, y, 0, 1e-5)
    print("num-done")

    for i in range(len(anw)):
            print(f"W diff:{abs(np.mean(anw[i]) - np.mean(numw[i]))}\t
             b diff:{abs(np.mean(anb[i]) - np.mean(numb[i]))}")
```

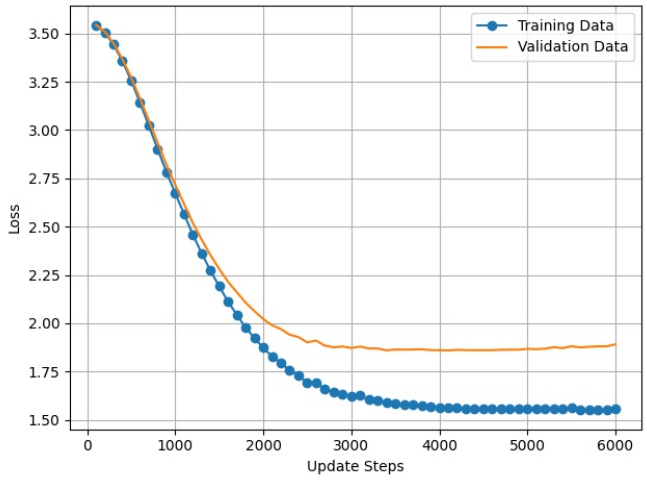
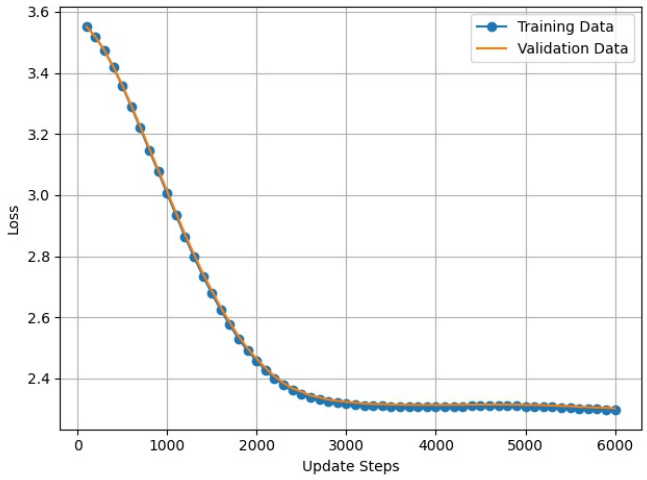
Which outputs

```
an-done
num-done
W diff:0.00020043066871582837    b diff:0.000804371882177268
W diff:4.4755865680201624e-18    b diff:1.9428902930940238e-17
```

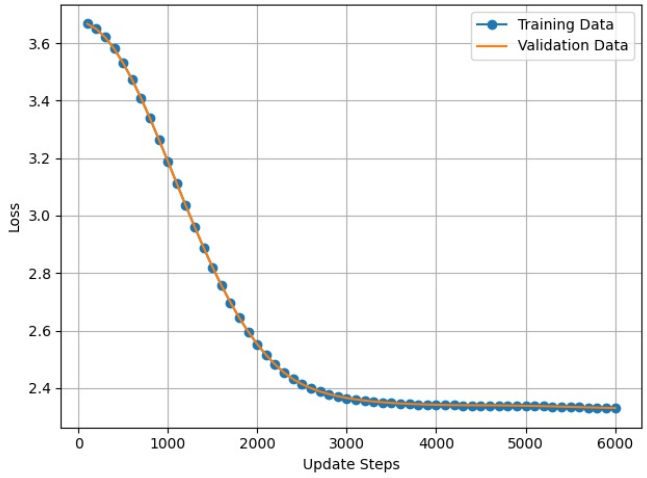
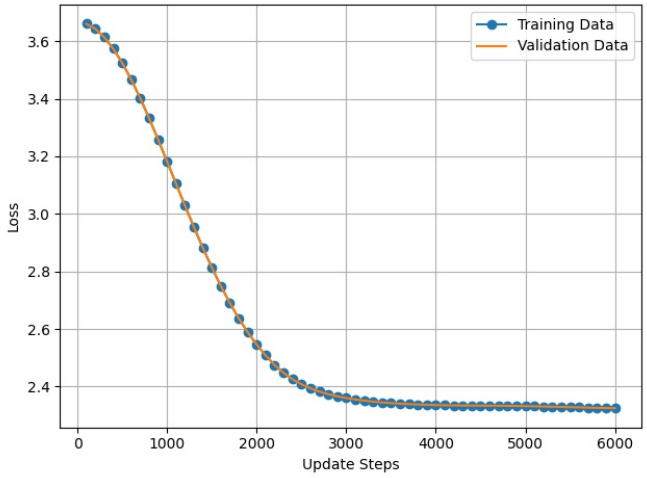
and suggests that the computations are correct.

Include graphs of the evolution of the loss function when you train the 3-layer network with and without batch normalization with the given default parameter setting.

lambda=0, n epochs=40, n batch=100, eta=.1

Without	With
	

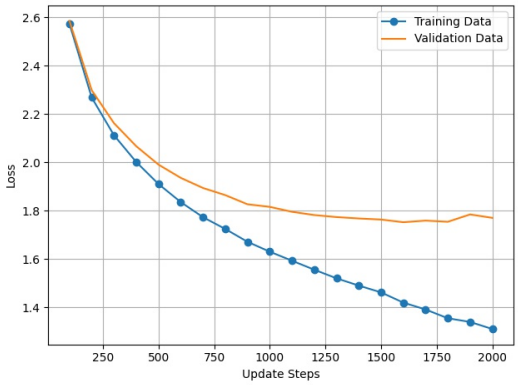
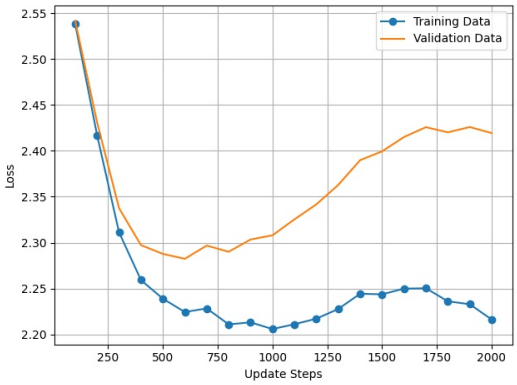
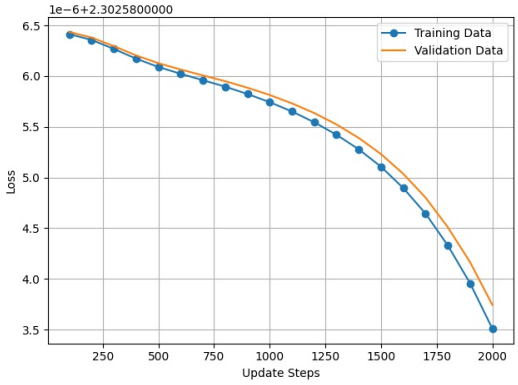
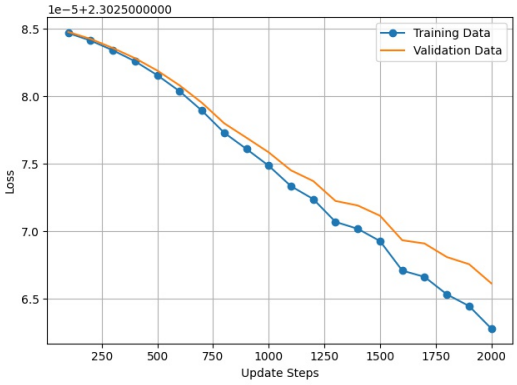
Include graphs of the evolution of the loss function when you train the 9-layer network with and without batch normalization with the given default parameter setting.

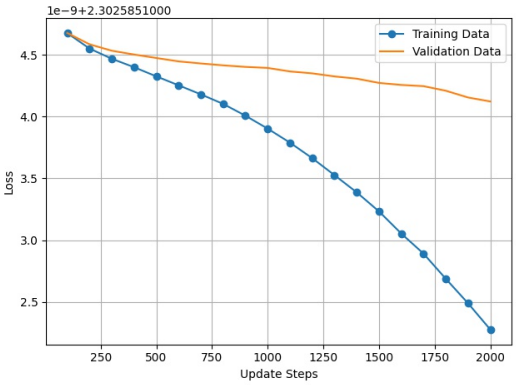
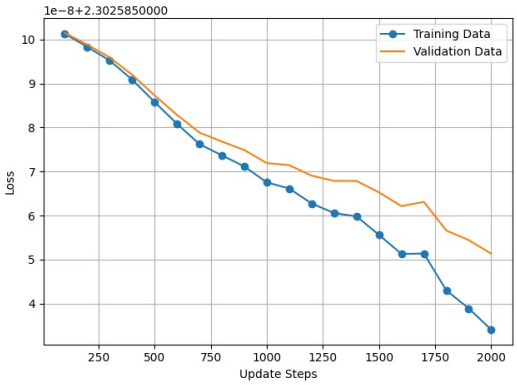
Without	With
	

State the range of the values you searched for lambda when you tried to optimize the performance of the 3-layer network trained with batch normalization, and the lambda settings for your best performing 3-layer network. Also state the test accuracy achieved by this network.

Include the loss plots for the training with Batch Norm Vs no Batch Norm for the experiment related to Sensitivity to initialization and comment on your experimental findings.

In the following table are reported the graphs of the loss function of the same network with 2 hidden layers of size 50 and the following parameters, but with different sigmas. `lambda=0.0000075, epochs=20, n_batch=100, eta={ 'min': 1e-5, 'max': 1e-1, 'step_size': 2250, 'l': 0 }`

Sigma	Without Batch norm	With Batch Norm
1e-1		
1e-3		

Sigma	Without Batch norm	With Batch Norm
1e-4		

The results seem to indicate that even if the loss is higher when using batch normalization, batch normalization keeps the final performance (in terms of accuracy) of the network almost the same even for different initialized parameters. Whereas without batch normalization the performance is highly sensible of the initialization.