Report:Programming Assignment I

Subhajit Mondal

a. Backpropagation equations for all the layers in terms of matrix notation.

Network Structure:
Our neural network have three hidden layers beside input and output layers.
Input layer take 784 dimensional vector as input followed by h1(1000) , h2(500) and h3(250) hidden layers with hidden units shown in brackets. Output layer got 10 units.

(a) Hidden layer use sigmoid or ReLu as activation function.

(b) Output layers activation function is linear.

(c) For classification we use softmax function and we took cross-entropy as error function.

Now the backpropagation equations for the network is,

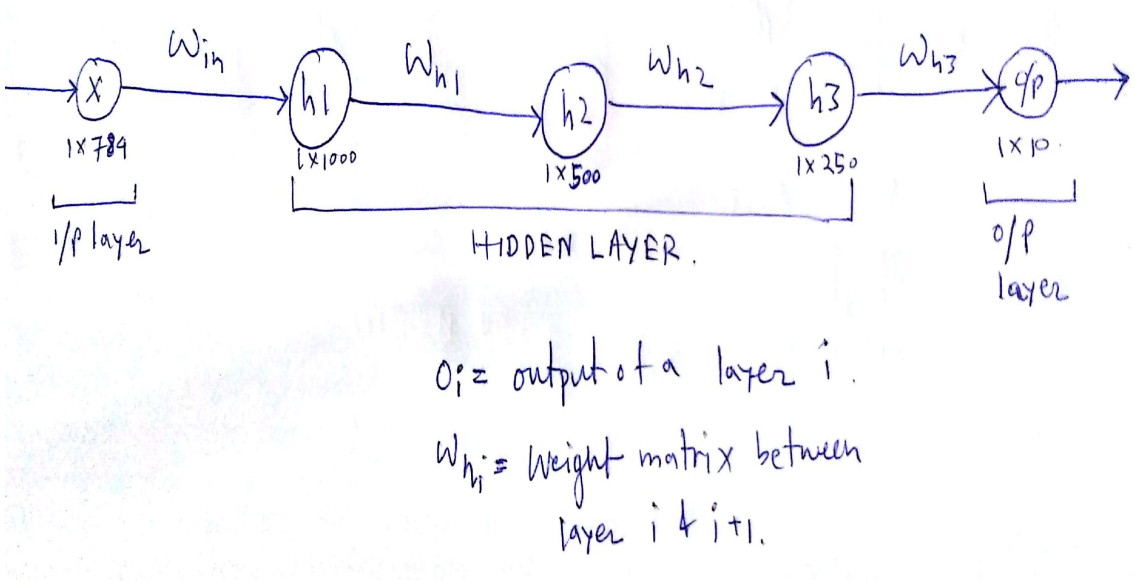

Figure 1: our network

Delta for output layer:

$$\delta_o = (\hat{y} - y), \text{ where } \delta_o, \hat{y}, y \in \mathbb{R}^{10} \tag{1}$$

where $\hat{y}$ is output(from the forward propagation) of the neural network for a specific input. y is the given label for the specific input. For the cross entrophy error function with softmax give us the δ_o as in the above form.

Delta for 3^{rd} hidden layer:

$$\delta_3 = (\delta_o W_{h3}^T) * \frac{d(f(in_{L3}))}{d(in_{L3})} \text{ ;where } \delta_o \in \mathbb{R}^{1*10}, \delta_3 \in \mathbb{R}^{1*250}, W_{h3} \in \mathbb{R}^{250*10} and \frac{d(f(in_{L3}))}{d(in_{L3})} \in \mathbb{R}^{1*250} \tag{2}$$

Where, W_{h3} is the weight matrix between output layer and h3 . in_{L3} is input to h3. $f(*)$ is activation function for hidden layer. $\star$ means element wise multiplication.

Delta for 2^{nd} hidden layer:

$$\delta_2 = (\delta_3 W_{h2}^T) * \frac{d(f(in_{L2}))}{d(in_{L2})} \text{ ;where } \delta_3 \in \mathbb{R}^{1*250}, \delta_2 \in \mathbb{R}^{1*500}, W_{h2} \in \mathbb{R}^{500*250} and \frac{d(f(in_{L2}))}{d(in_{L2})} \in \mathbb{R}^{1*500} \quad (3)$$

Where, W_{h2} is the weight matrix between h3 and h2.
in_{L2} is input to h2.

Delta for 1^{st} hidden layer:

$$\delta_1 = (\delta_2 W_{h1}^T) * \frac{d(f(in_{L1}))}{d(in_{L1})} \text{ ;where } \delta_2 \in \mathbb{R}^{1*500}, \delta_1 \in \mathbb{R}^{1*1000}, W_{h1} \in \mathbb{R}^{784*1000} and \frac{d(f(in_{L1}))}{d(in_{L1})} \in \mathbb{R}^{1*1000} \quad (4)$$

Where, W_{h1} is the weight matrix between h2 and h1 .
in_{L1} is input to h1.

Gradient of W_{h3}:

$$dW_{h3} = O_3^T \delta_o \text{;where } .dW_{h3} \in \mathbb{R}^{250*10}, O_3 \in \mathbb{R}^{1*250} \quad (5)$$

dW_{h3} is gradient of W_{h3} and O_3 is output of hidden layer h3.
Gradient of W_{h2}:

$$dW_{h2} = O_2^T \delta_3 \text{;where } .dW_{h2} \in \mathbb{R}^{500*250}, O_2 \in \mathbb{R}^{1*500} \quad (6)$$

dW_{h2} is gradient of W_{h2} and O_2 is output of hidden layer h2.
yer h3.
Gradient of W_{h1}:

$$W_{h1} = O_1^T \delta_2 \text{;where } .dW_{h1} \in \mathbb{R}^{1000*500}, O_1 \in \mathbb{R}^{1*1000} \quad (7)$$

dW_{h1} is gradient of W_{h1} and O_1 is output of hidden layer h1.
yer h3.
Gradient of W_{in}:

$$dW_{in} = X^T \delta_1 \text{;where } .dW_{in} \in \mathbb{R}^{784*1000}, X \in \mathbb{R}^{1*784} \quad (8)$$

dW_{in} is gradient of W_{in} which is weight matrix between input layer and h1 . X is input vectors.

b. Plots for testing and training loss for sigmoid activation function for diffrent learning rates:

The neural networks are trained using sigmoid activation function,with learning rate schedular (by 0.85 every 250 iteration) and momentum (momentum parameter 0.9) update method.
Learning rate:0.01

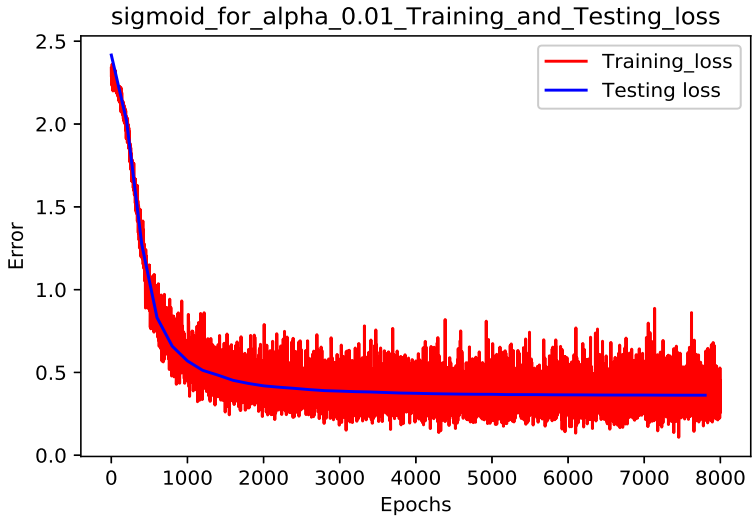

Figure 2: iteration vs training and testing error for learning rate:0.01

Learning rate:0.05

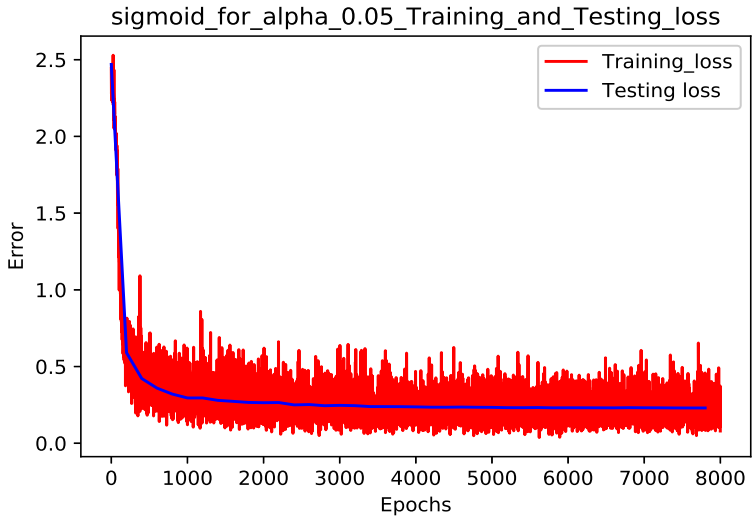

Figure 3: iteration vs training and testing error for learning rate:0.05

Learning rate:0.1

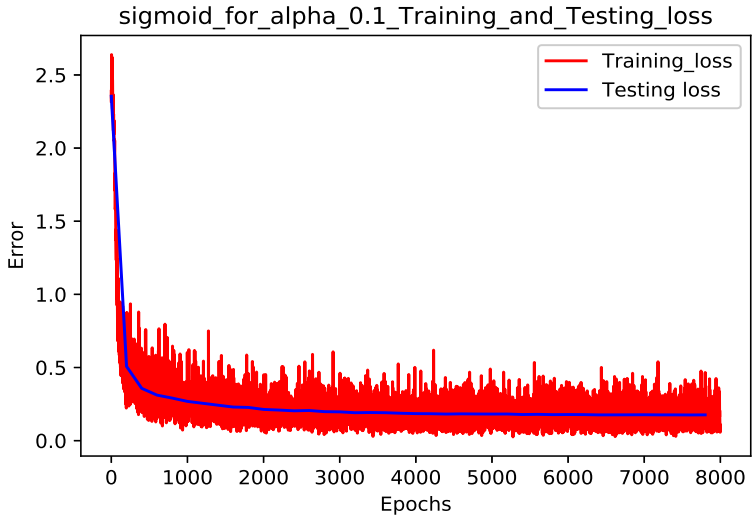

Figure 4: iteration vs training and testing error for learning rate:0.1

We can see from the figures that errors are convergeing at a faster rate with the increase of learning rate.For the 0.1 learning rate the error converge quickly than others.

c. Sigmoid activation function with and without learning rate scheduling

Now we are training the model without LR scheduling(previously all training done using LR scheduling) and plot the training and test error plot with iteration.
Learning rate:0.5(without scheduling)

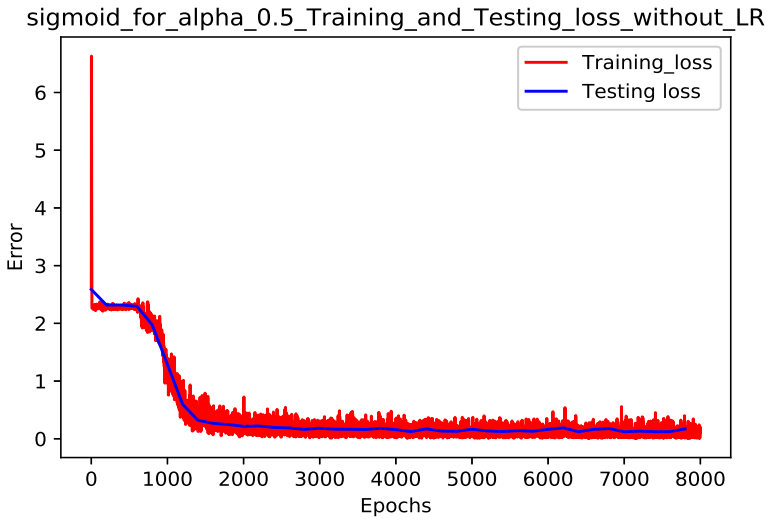

Figure 5: iteration vs training and testing error for learning rate:0.5 without scheduling

Learning rate:0.5(with scheduling)

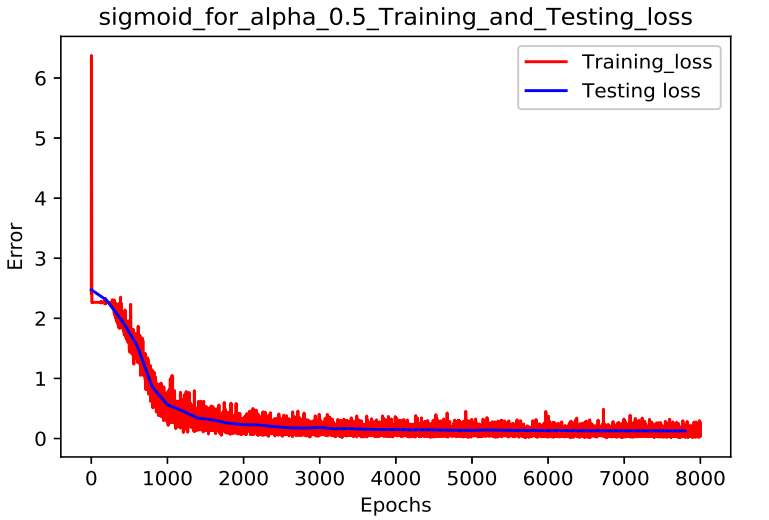

Figure 6: iteration vs training and testing error for learning rate:0.5 with scheduling by 0.85 every 250 iteration

From figures we can clearly see that with Learning rate Scheduling the error converge faster than non scheduling case.

d. Sigmoid activation function with vs ReLu activation function with scheduling using learning rate:0.1

Sigmoid activation function

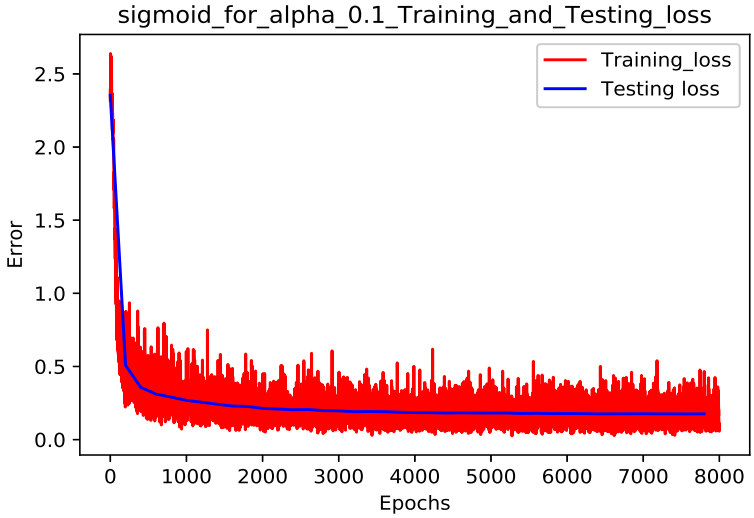

Figure 7: iteration vs training and testing error for Sigmoid activation function

Using ReLu activation function

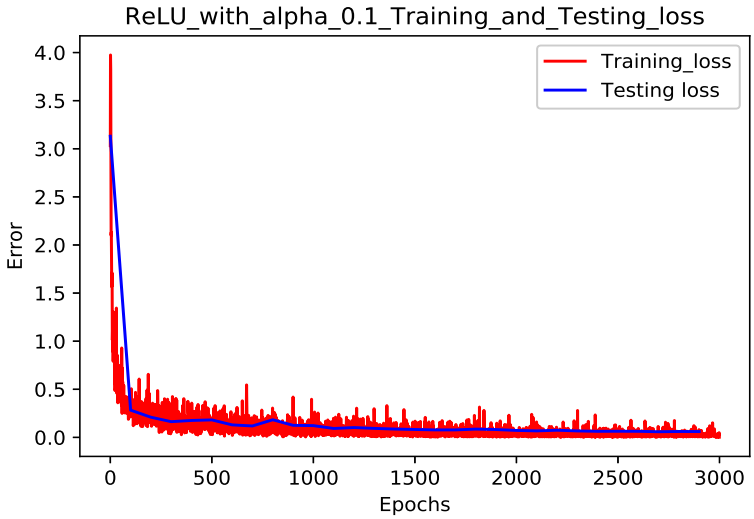

Figure 8: iteration vs training and testing error for ReLu activation function

We can see from above figures that if we use ReLu activation function,the errors converge very faster than Sigmoid activation function.For Relu we have taken only 3000 iteration but we can see that error already reduced than Sigmoid function.
Final Accuracy for Sigmoid 94.84 percent.
Final ReLu for Sigmoid 98.15 percent.

e. Prediction for a fixed set of 20 images

For ReLu:

Image	True	1st Prediction	2nd prediction	3rd prediction
4387	5	5	9	3
8320	2	2	7	3
4077	1	1	8	4
4751	4	4	6	1
4668	2	2	3	8
6191	0	0	2	9
8500	4	4	7	1
5164	9	9	5	3
2918	2	2	3	4
229	7	7	3	9
4988	9	9	7	3
5586	8	8	2	0
9736	6	6	5	8
6960	7	7	3	2
1546	2	2	3	7
9009	7	2	3	7
5308	2	2	3	7
846	7	7	9	0
9373	6	6	4	0
8817	9	9	3	5

For sigmoid:

Image	True	1st Prediction	2nd prediction	3rd prediction
4387	5	5	8	3
8320	2	2	7	3
4077	1	1	3	8
4751	4	6	2	4
4668	2	2	3	1
6191	0	0	2	5
8500	4	4	9	7
5164	9	9	5	4
2918	2	2	3	4
229	7	7	9	3
4988	9	9	7	4
5586	8	8	0	9
9736	6	6	5	2
6960	7	7	3	2
1546	2	2	3	8
9009	7	2	3	7
5308	2	2	3	6
846	7	7	9	0
9373	6	6	4	2
8817	9	9	4	3