Programming Assignment 1 : MNIST Classification using Multilayer Perceptron

Akshit Kumar, EE14B127
EE6132 - Deep Learning for Imaging

August 29, 2017

1 Assignment Objective

In this assignment we do classification on the MNIST digits dataset using multilayer perceptrons. The architecture for the multilayer is as follows:

- Input Layer : (28×28) flattened into a 784 dimensional vector
- 3 Hidden Layers : $h_1(1000), h_2(500), h_3(250)$
- Output Layer : 10 units with one-hot encoding

We make use of the cross entropy loss function between $\hat{y_i}$ and y_i where y_i is the ground truth label and $\hat{y_i}$ is the predicted label. We make use of the following two activation functions for all the layers expect the output layer:

- Sigmoid
- ReLu

For the output layer, we have *softmax* activation function. Backpropagation algorithm is implemented from scratch.

2 Back-Propagation Algorithm Equations

We define our cross entropy loss function as follows:

$$L = -\sum_i y_i \log(\hat{y_i})$$

where $\hat{y_i} = \frac{e^{z_i}}{\sum_j e^{z_j}}$

Now we know that for any layer l,

$$\delta^l = \left(W^l\right)^T \delta^{l+1} \odot f^{'}(z^l)$$

Therefore, we first need to calculate the δ for the last layer L, let is be represented by δ^L.

$$\begin{aligned}
\delta_i^L &= \frac{\partial L}{\partial z_i^L} && (1)\\
&= -\sum_k y_k \frac{\partial \log(\hat{y_k})}{\partial z_i^L} && (2)\\
&= -\sum_k y_k \frac{1}{\hat{y_k}} \frac{\partial \hat{y_k}}{\partial z_i^L} && (3)\\
&= -y_i\left(1-\hat{y_i}\right) - \sum_{k\neq i} y_k \frac{1}{\hat{y_k}}\left(-\hat{y_k}\hat{y_i}\right) && (4)\\
&= -y_i + y_i\hat{y_i} + \sum_{k\neq i} y_k\hat{y_i} && (5)\\
&= \hat{y_i}\left(\sum_k y_k\right) - y_i && (6)\\
&= \hat{y_i} - y_i && (7)
\end{aligned}$$

Hence we can vectorize the calculation as $\delta^L = \hat{y} - y$. Now we can calculate the gradient of the weight matrices using

$$\frac{\partial L}{\partial W_{ij}^l} = \delta_i^{l+1} a_j^l$$

3 L_2 Regularization

In order to avoid overfitting of the data, we use the L_2-regularization and modify our cost function as follows :

$$\hat{L} = L + \frac{\lambda}{2}\sum_l\sum_i\sum_j \left(W_{ij}^l\right)^2$$

where L is the cross entropy function defined above. For the experimentations in the assignment, we fix $\lambda = 0.005$

4 Stochastic Gradient Descent with momentum

In order to update the weights, we make use of Stochastic Gradient Descent with momentum by making use of a mini-batch size of 64 images.For implementing momentum based acceleration we make use of following equations:

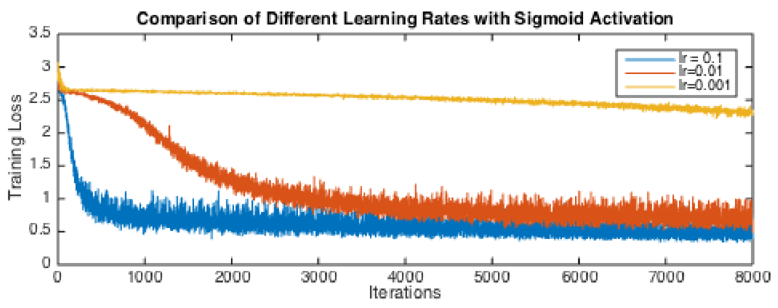
$$\begin{aligned}
v &= \alpha v - \epsilon\nabla_W\hat{L} \qquad (\nabla_W\hat{L} \text{ is the gradient}) && (8)\\
W &= W + v && (9)
\end{aligned}$$

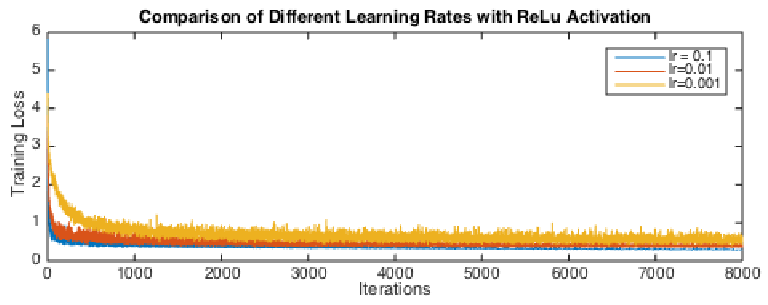
We can think of v as velocity,here α and ϵ are hyper parameters. For the purposes of this assignment, we fix $\alpha = 0.5$ and ϵ is kept both constant and scheduled. Results for both cases will be shown.

5 Experiments

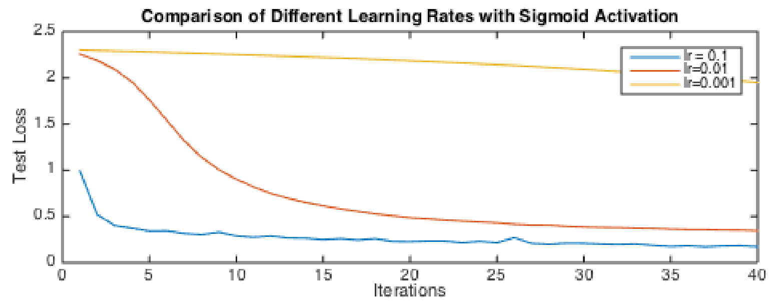
5.1 Comparison of different learning rates

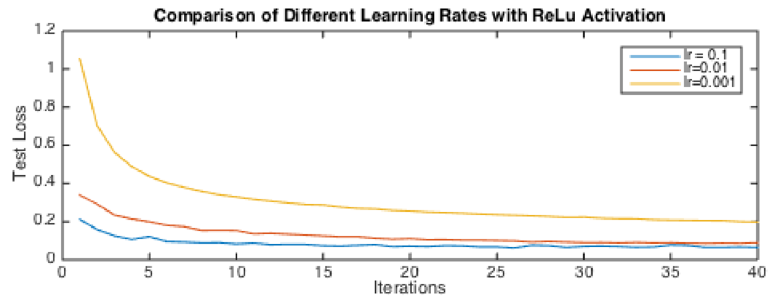
5.1.1 Training Loss Plot



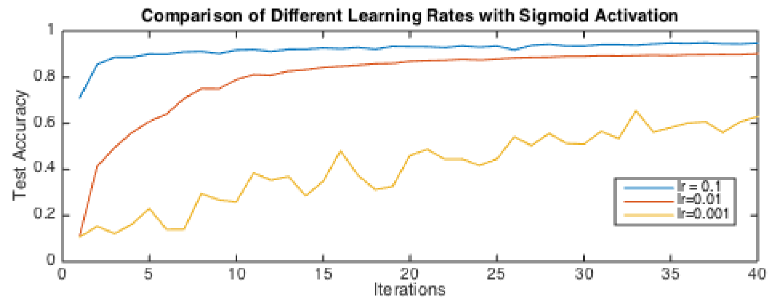


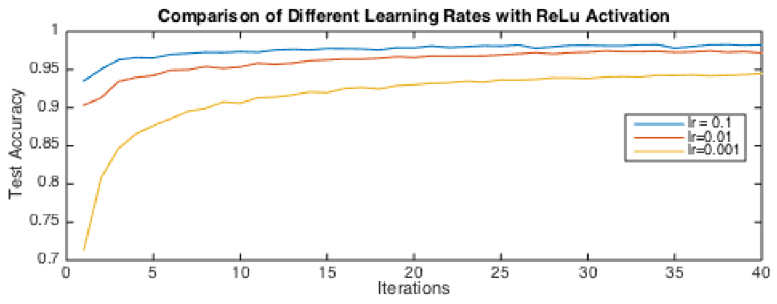
5.1.2 Test Loss Plot





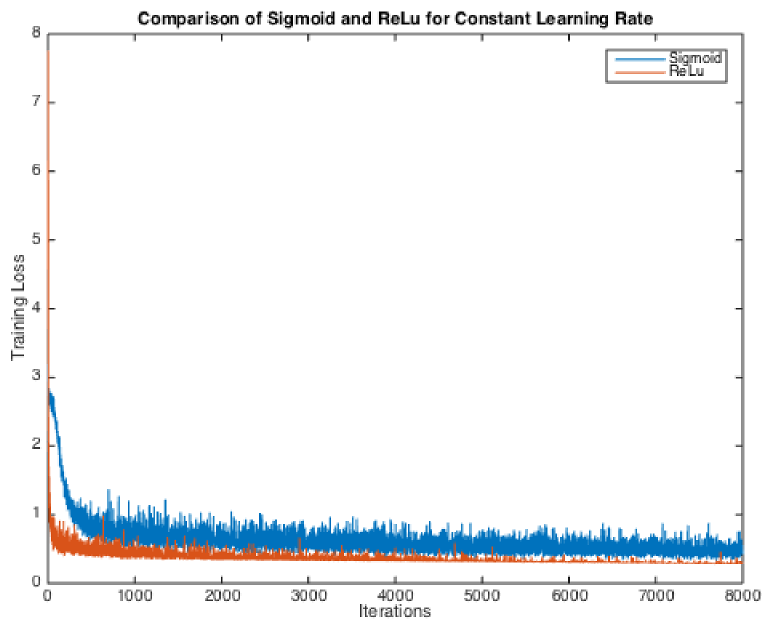
5.1.3 Test Accuracy Plot





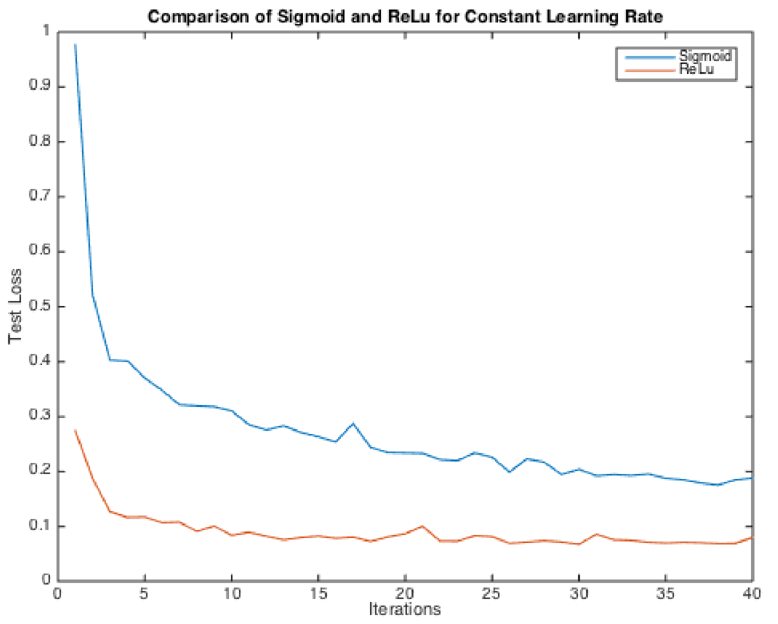
5.2 Comparison of Sigmoid and ReLu for constant learning rate

5.2.1 Training Loss Plot

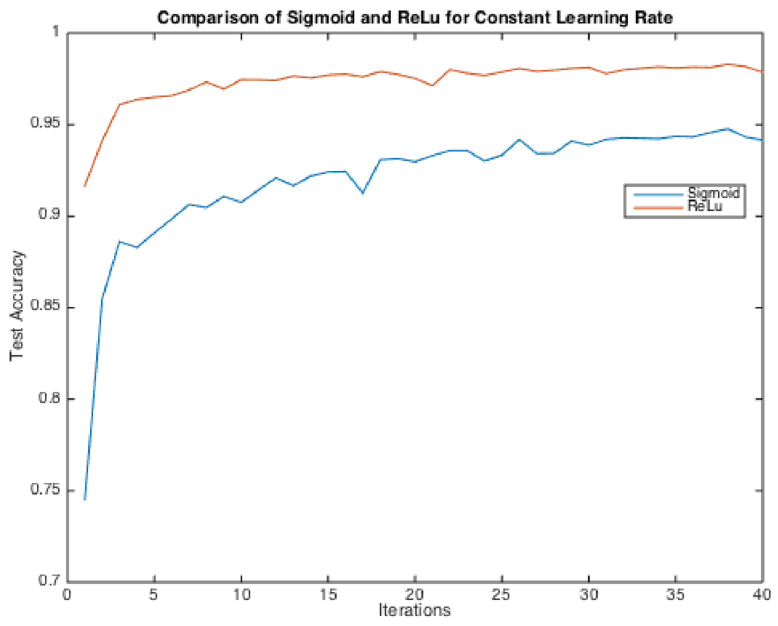


From this graph, we infer that ReLu activation function converges faster compared to the Sigmoid activation function which could probably be attributed to vanishing gradients in the case of sigmoid activation function.

5.2.2 Test Loss Plot

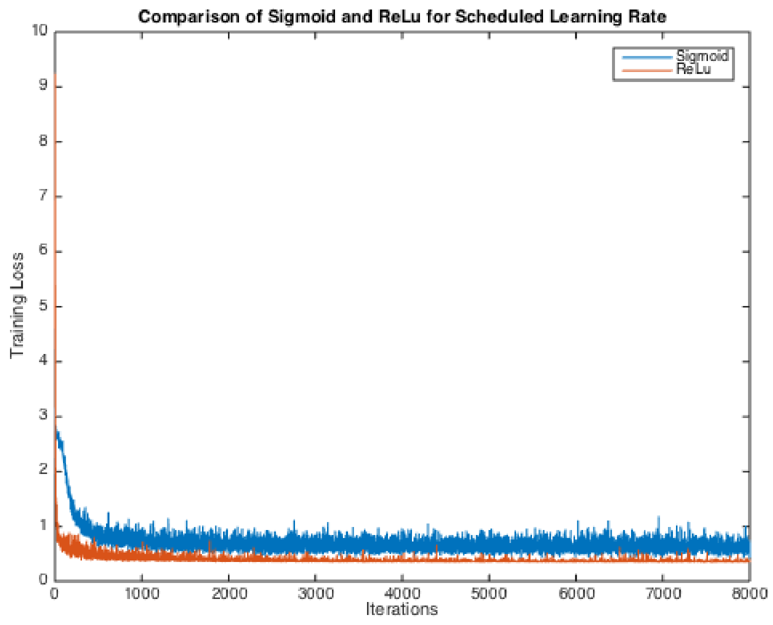


5.2.3 Test Accuracy Plot

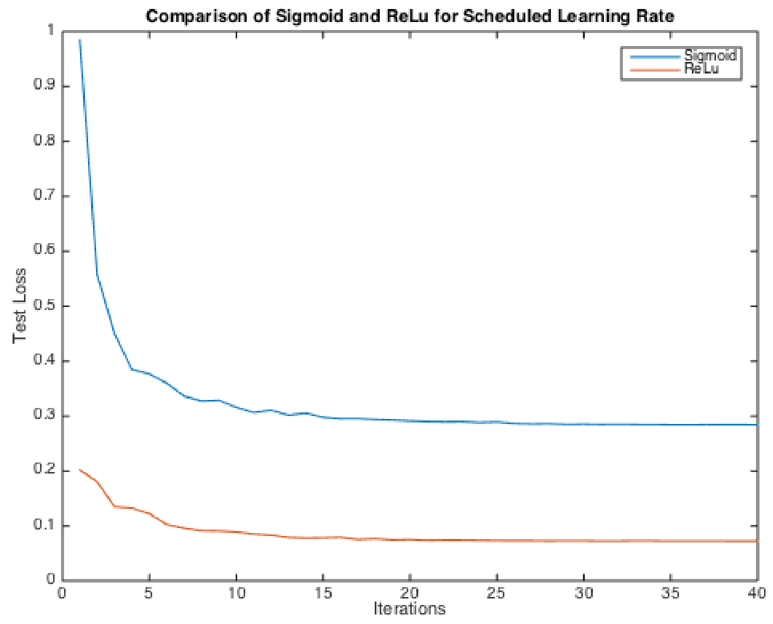


5.3 Comparison of Sigmoid and Relu for scheduled learning rate

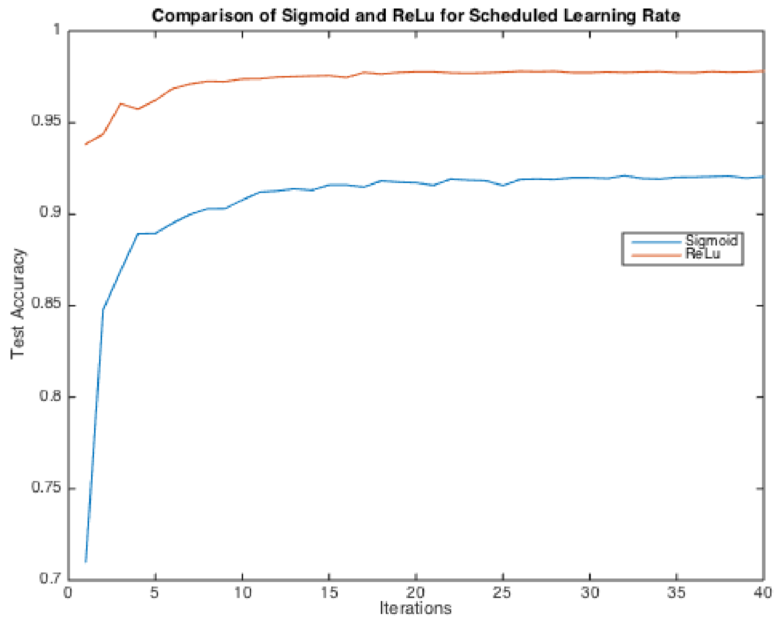
5.3.1 Training Loss Plot



5.3.2 Test Loss Plot

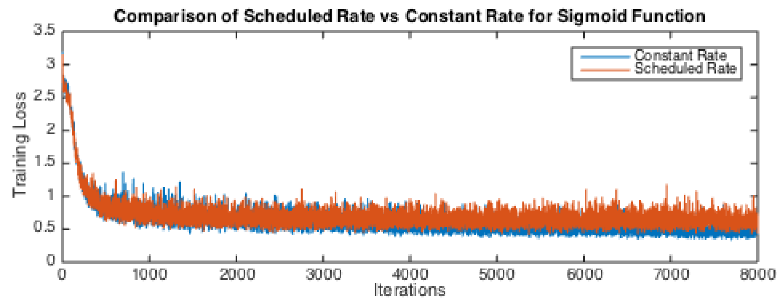


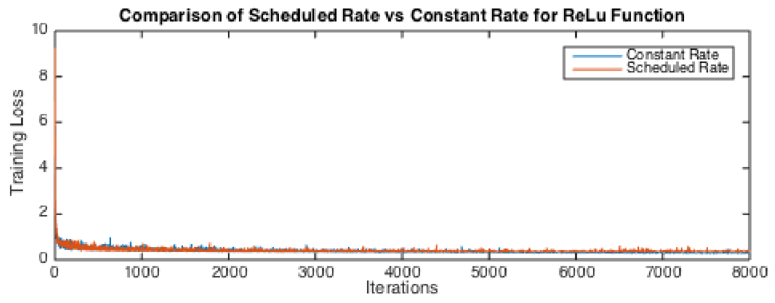
5.3.3 Test Accuracy Plot



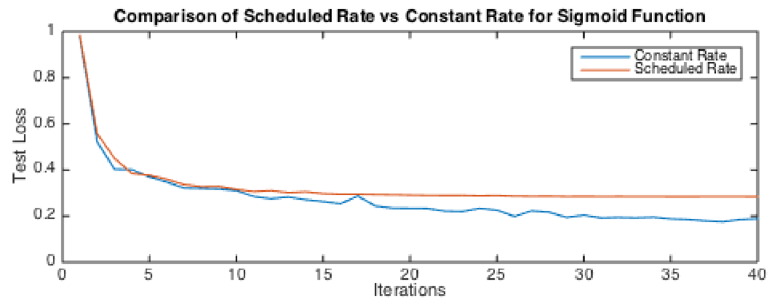
5.4 Comparison of Scheduled Learning Rate to Constant Learning Rate

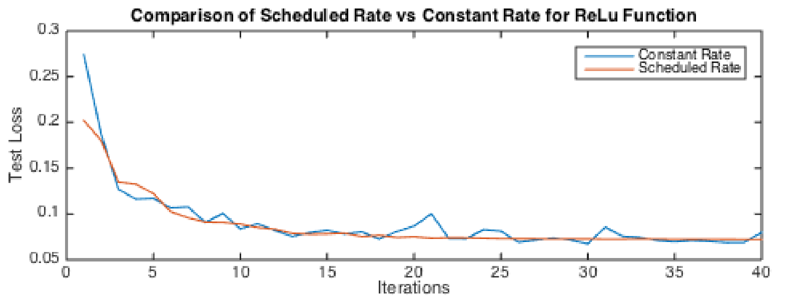
5.4.1 Training Loss Plot



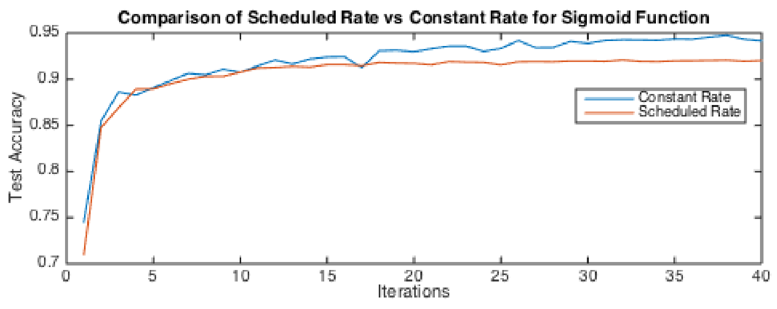


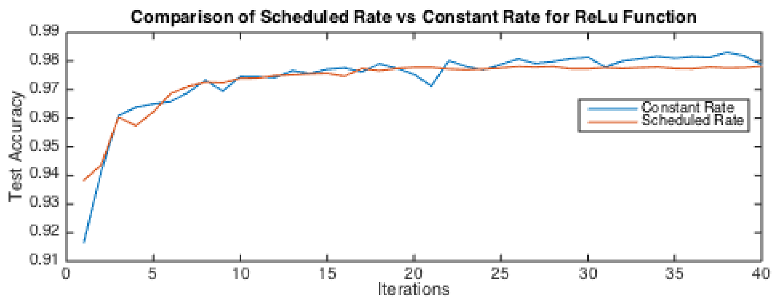
5.4.2 Test Loss Plot





5.4.3 Test Accuracy Plot





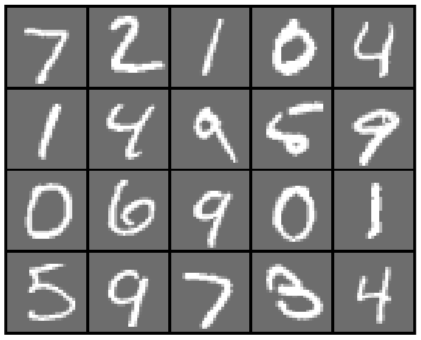
6 Final Test Accuracies

Following are the test accuracies achieved:

Activation	Initial Learning Rate	Scheduling	Final Test Accuracy
Sigmoid	0.1	No	94.16 %
Sigmoid	0.1	Yes	92.04 %
ReLu	0.1	No	97.87 %
ReLu	0.1	Yes	97.82 %

7 Top 3 Guess Estimates for Sigmoid and ReLu

7.1 20 Test Images



7.2 Representation of Images

Let the 20 images be represented by this table

I-1	I-2	I-3	I-4	I-5
I-6	I-7	I-8	I-9	I-10
I-11	I-12	I-13	I-14	I-15
I-16	I-17	I-18	I-19	I-20

7.3 Estimates for Sigmoid Activation

7.3.1 Constant Learning Rate

In this scenario, the weights were learnt using the following parameters:

- ϵ (Fixed Learning Rate) = 0.1
- The learning rate was not decayed

Image	Guess 1 / Probability	Guess 2 / Probability	Guess 3 / Probability
I-1	7 / 0.999190	3 / 0.000348	2 / 0.000342
I-2	2 / 0.989174	3 / 0.003896	6 / 0.002250
I-3	1 / 0.992020	2 / 0.004603	7 / 0.001199
I-4	0 / 0.999664	7 / 0.000095	5 / 0.000066
I-5	4 / 0.993679	9 / 0.003306	2 / 0.001156
I-6	1 / 0.991659	7 / 0.006660	2 / 0.001107
I-7	4 / 0.989119	5 / 0.006356	8 / 0.002833
I-8	9 / 0.992484	4 / 0.003214	7 / 0.002266
I-9	6 / 0.976888	5 / 0.012997	4 / 0.003485
I-10	9 / 0.921045	4 / 0.039969	7 / 0.035453
I-11	0 / 0.998769	5 / 0.000954	2 / 0.000242
I-12	6 / 0.982632	8 / 0.006860	2 / 0.005027
I-13	9 / 0.977623	7 / 0.013483	4 / 0.007654
I-14	0 / 0.998434	9 / 0.000832	5 / 0.000339
I-15	1 / 0.999089	3 / 0.000380	8 / 0.000166
I-16	5 / 0.996141	3 / 0.002638	8 / 0.000988
I-17	9 / 0.961589	4 / 0.024884	7 / 0.011969
I-18	7 / 0.998695	3 / 0.000740	2 / 0.000458
I-19	3 / 0.732751	8 / 0.101390	2 / 0.064843
I-20	4 / 0.998097	5 / 0.000754	9 / 0.000742

7.3.2 Scheduled Learning Rate

In this scenario, the weights were learnt using the following parameters:

- ϵ_0 (Initial Learning Rate) = 0.1
- The learning rate was gradually decayed

Image	Guess 1 / Probability	Guess 2 / Probability	Guess 3 / Probability
I-1	7 / 0.996803	3 / 0.001573	9 / 0.001104
I-2	2 / 0.924212	6 / 0.025138	3 / 0.025006
I-3	1 / 0.983606	2 / 0.007285	7 / 0.002901
I-4	0 / 0.998297	5 / 0.001227	2 / 0.000247
I-5	4 / 0.950882	9 / 0.037692	7 / 0.005001
I-6	1 / 0.982689	7 / 0.007171	2 / 0.003847
I-7	4 / 0.974769	9 / 0.010388	5 / 0.010015
I-8	9 / 0.913269	4 / 0.034508	7 / 0.016969
I-9	6 / 0.975984	4 / 0.014237	2 / 0.004873
I-10	9 / 0.941090	7 / 0.032582	4 / 0.022834
I-11	0 / 0.961688	5 / 0.032782	2 / 0.003213
I-12	6 / 0.884358	2 / 0.047780	8 / 0.040796
I-13	9 / 0.930547	7 / 0.039155	4 / 0.026913
I-14	0 / 0.988783	5 / 0.007936	8 / 0.001547
I-15	1 / 0.993824	3 / 0.003248	8 / 0.001416
I-16	5 / 0.831020	3 / 0.096807	8 / 0.062968
I-17	9 / 0.897692	7 / 0.058613	4 / 0.039194
I-18	7 / 0.992020	3 / 0.006610	9 / 0.000486
I-19	3 / 0.816720	2 / 0.085775	8 / 0.039881
I-20	4 / 0.982669	9 / 0.013542	5 / 0.002058

7.4 Estimates for ReLu Activation

7.4.1 Constant Learning Rate

In this scenario, the weights were learnt using the following parameters:

- ϵ (Fixed Learning Rate) = 0.1
- The learning rate was not decayed

Image	Guess 1 / Probability	Guess 2 / Probability	Guess 3 / Probability
I-1	7 / 1.000000	3 / 0.000000	9 / 0.000000
I-2	2 / 0.999999	1 / 0.000001	3 / 0.000000
I-3	1 / 0.999297	7 / 0.000627	2 / 0.000046
I-4	0 / 0.999998	2 / 0.000001	7 / 0.000000
I-5	4 / 0.999858	9 / 0.000141	7 / 0.000001
I-6	1 / 0.998474	7 / 0.001524	4 / 0.000001
I-7	4 / 0.986769	8 / 0.013081	9 / 0.000106
I-8	9 / 0.999689	3 / 0.000301	7 / 0.000006
I-9	5 / 0.974093	6 / 0.025399	8 / 0.000374
I-10	9 / 0.999316	4 / 0.000682	7 / 0.000002
I-11	0 / 1.000000	2 / 0.000000	7 / 0.000000
I-12	6 / 1.000000	0 / 0.000000	8 / 0.000000
I-13	9 / 1.000000	4 / 0.000000	7 / 0.000000
I-14	0 / 0.999997	7 / 0.000002	2 / 0.000000
I-15	1 / 1.000000	7 / 0.000000	8 / 0.000000
I-16	5 / 0.999746	3 / 0.000254	1 / 0.000000
I-17	9 / 0.999998	7 / 0.000001	4 / 0.000000
I-18	7 / 1.000000	3 / 0.000000	9 / 0.000000
I-19	3 / 0.999512	8 / 0.000411	2 / 0.000071
I-20	4 / 1.000000	9 / 0.000000	7 / 0.000000

7.4.2 Scheduled Learning Rate

In this scenario, the weights were learnt using the following parameters:

- ϵ_0 (Initial Learning Rate) = 0.1
- The learning rate was gradually decayed

Image	Guess 1 / Probability	Guess 2 / Probability	Guess 3 / Probability
I-1	7 / 0.999930	3 / 0.000045	2 / 0.000016
I-2	2 / 0.999860	1 / 0.000130	3 / 0.000006
I-3	1 / 0.996079	7 / 0.002889	2 / 0.000466
I-4	0 / 0.999686	2 / 0.000293	9 / 0.000007
I-5	4 / 0.999516	9 / 0.000397	7 / 0.000061
I-6	1 / 0.999104	7 / 0.000867	4 / 0.000019
I-7	4 / 0.997550	8 / 0.002350	9 / 0.000071
I-8	9 / 0.990573	3 / 0.008724	7 / 0.000524
I-9	5 / 0.900104	6 / 0.051146	8 / 0.031192
I-10	9 / 0.999841	4 / 0.000133	8 / 0.000016
I-11	0 / 0.999994	2 / 0.000005	9 / 0.000000
I-12	6 / 0.999960	0 / 0.000025	8 / 0.000007
I-13	9 / 0.999938	4 / 0.000042	3 / 0.000016
I-14	0 / 0.999964	9 / 0.000023	2 / 0.000005
I-15	1 / 0.999996	8 / 0.000002	7 / 0.000001
I-16	5 / 0.983181	3 / 0.015039	8 / 0.001102
I-17	9 / 0.999746	7 / 0.000097	8 / 0.000094
I-18	7 / 0.999862	3 / 0.000079	2 / 0.000045
I-19	3 / 0.981717	8 / 0.014619	2 / 0.003481
I-20	4 / 0.999994	7 / 0.000004	9 / 0.000002