بسمه تعالی

HW3- Deep Neural Networks

Supervisor:

Prof. Johari Majd

Student:

Morteza Bigdeli - 40261662001

Tarbiat Modares University

Spring 2024

Table of Contents

Table if Figures

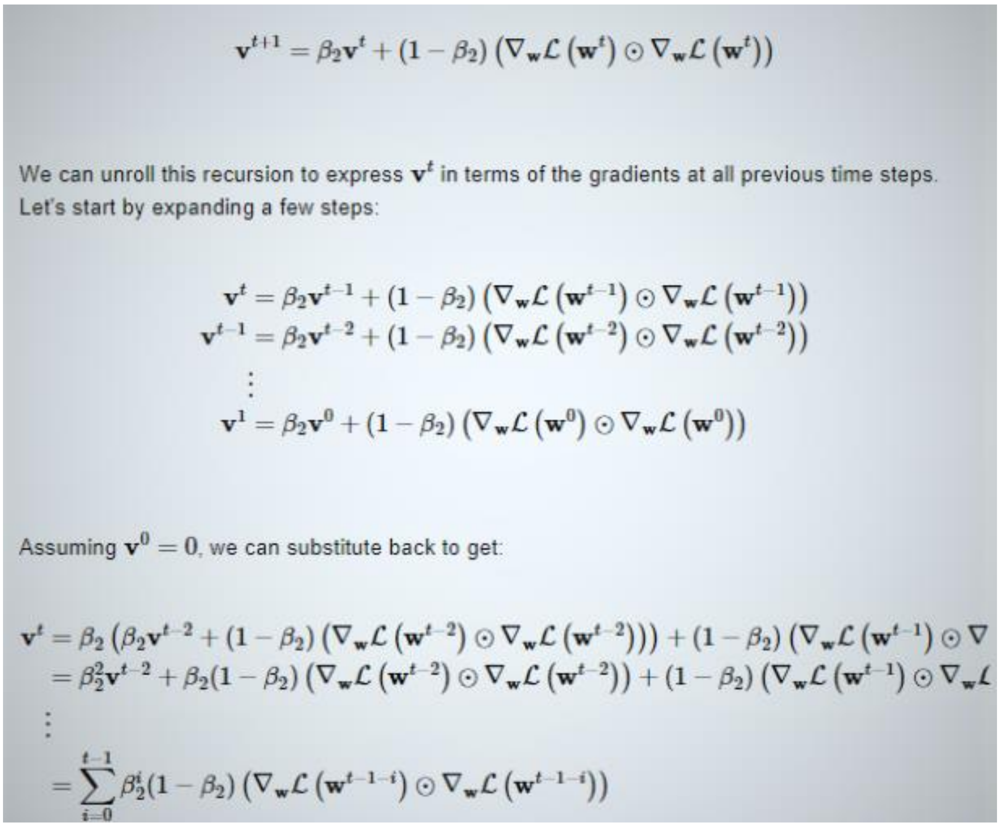
1. Pen and Paper Exercises

Derivation of the normalization term in Adam.

$$
\begin{aligned}
\mathbf{m}^{t+1} &= \beta_1 \mathbf{m}^t + (1-\beta_1)\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \\
\mathbf{v}^{t+1} &= \beta_2 \mathbf{v}^t + (1-\beta_2)\big(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)\big) \\
\hat{\mathbf{m}}^{t+1} &= \frac{\mathbf{m}^{t+1}}{1-\beta_1^{t+1}}, \hat{\mathbf{v}}^{t+1} = \frac{\mathbf{v}^{t+1}}{1-\beta_2^{t+1}} \\
\mathbf{w}^{t+1} &= \mathbf{w}^t - \eta \frac{\hat{\mathbf{m}}^{t+1}}{\sqrt{\hat{\mathbf{v}}_{t+1}} + \epsilon}
\end{aligned}
$$

a)

a) Derive the explicit form of $\mathbf{v}^t$ w.r.t. $\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)$. You can express uncentered variance $\mathbf{v}^t$ in the form of $\sum_t f^t(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t), \beta_2)$.



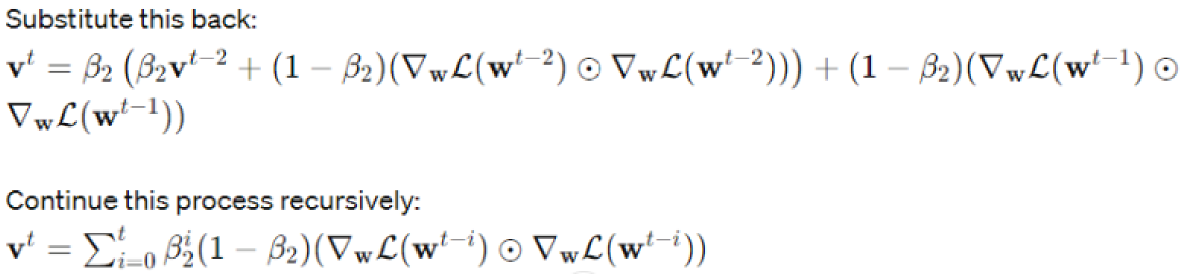
$$\mathbf{v}^{t+1} = \beta_2 \mathbf{v}^t + (1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^t\right) \odot \nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^t\right)\right)$$

We can unroll this recursion to express $\mathbf{v}^t$ in terms of the gradients at all previous time steps. Let's start by expanding a few steps:

$$
\begin{aligned}
\mathbf{v}^t &= \beta_2 \mathbf{v}^{t-1} + (1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-1}\right) \odot \nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-1}\right)\right) \\
\mathbf{v}^{t-1} &= \beta_2 \mathbf{v}^{t-2} + (1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-2}\right) \odot \nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-2}\right)\right) \\
&\vdots \\
\mathbf{v}^1 &= \beta_2 \mathbf{v}^0 + (1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^0\right) \odot \nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^0\right)\right)
\end{aligned}
$$

Assuming $\mathbf{v}^0 = 0$, we can substitute back to get:

$$
\begin{aligned}
\mathbf{v}^t &= \beta_2\left(\beta_2 \mathbf{v}^{t-2} + (1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-2}\right) \odot \nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-2}\right)\right)\right) + (1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-1}\right) \odot \nabla \right. \\
&= \beta_2^2 \mathbf{v}^{t-2} + \beta_2(1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-2}\right) \odot \nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-2}\right)\right) + (1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-1}\right) \odot \nabla_{\mathbf{w}}\mathcal{L} \right. \\
&\vdots \\
&= \sum_{i=0}^{t-1} \beta_2^i (1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-1-i}\right) \odot \nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}^{t-1-i}\right)\right)
\end{aligned}
$$



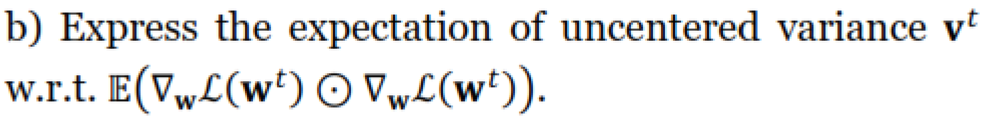
Substitute this back:

$\mathbf{v}^t = \beta_2\left(\beta_2\mathbf{v}^{t-2} + (1-\beta_2)(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^{t-2}) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^{t-2}))\right) + (1-\beta_2)(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^{t-1}) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^{t-1}))$

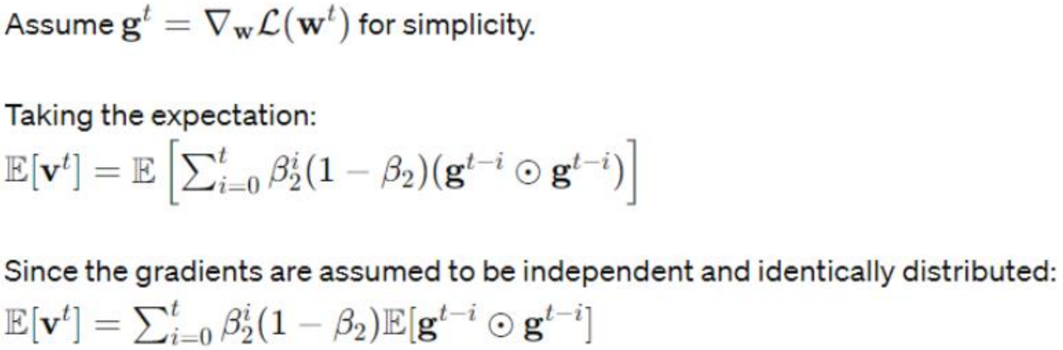
Continue this process recursively:

$\mathbf{v}^t = \sum_{i=0}^{t}\beta_2^i(1-\beta_2)(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^{t-i}) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^{t-i}))$

b)



b) Express the expectation of uncentered variance $\mathbf{v}^t$ w.r.t. $\mathbb{E}\big(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)\big)$.



Assume $\mathbf{g}^t = \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)$ for simplicity.

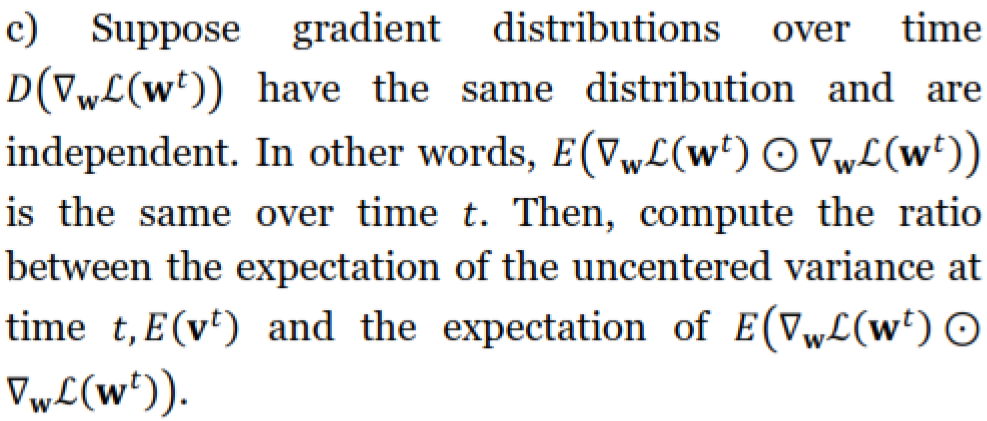
Taking the expectation:

$\mathbb{E}[\mathbf{v}^t] = \mathbb{E}\left[\sum_{i=0}^{t}\beta_2^i(1-\beta_2)(\mathbf{g}^{t-i} \odot \mathbf{g}^{t-i})\right]$

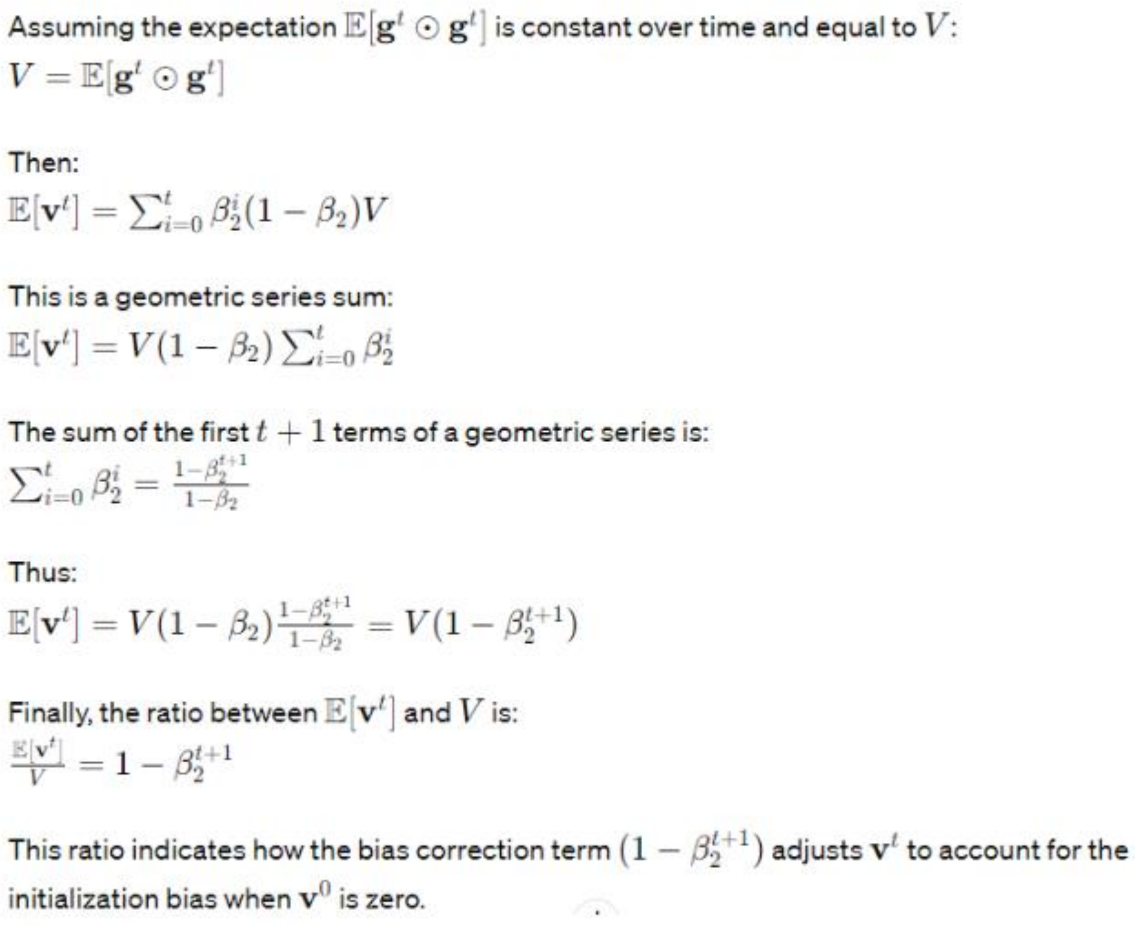
Since the gradients are assumed to be independent and identically distributed:

$\mathbb{E}[\mathbf{v}^t] = \sum_{i=0}^{t}\beta_2^i(1-\beta_2)\mathbb{E}[\mathbf{g}^{t-i} \odot \mathbf{g}^{t-i}]$

c)



c) Suppose gradient distributions over time $D\big(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)\big)$ have the same distribution and are independent. In other words, $E\big(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)\big)$ is the same over time t. Then, compute the ratio between the expectation of the uncentered variance at time $t, E(\mathbf{v}^t)$ and the expectation of $E\big(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)\big)$.



Assuming the expectation $\mathbb{E}[\mathbf{g}^t \odot \mathbf{g}^t]$ is constant over time and equal to V:

$V = \mathbb{E}[\mathbf{g}^t \odot \mathbf{g}^t]$

Then:

$\mathbb{E}[\mathbf{v}^t] = \sum_{i=0}^{t} \beta_2^i (1 - \beta_2) V$

This is a geometric series sum:

$\mathbb{E}[\mathbf{v}^t] = V(1 - \beta_2) \sum_{i=0}^{t} \beta_2^i$

The sum of the first $t + 1$ terms of a geometric series is:

$\sum_{i=0}^{t} \beta_2^i = \frac{1-\beta_2^{t+1}}{1-\beta_2}$

Thus:

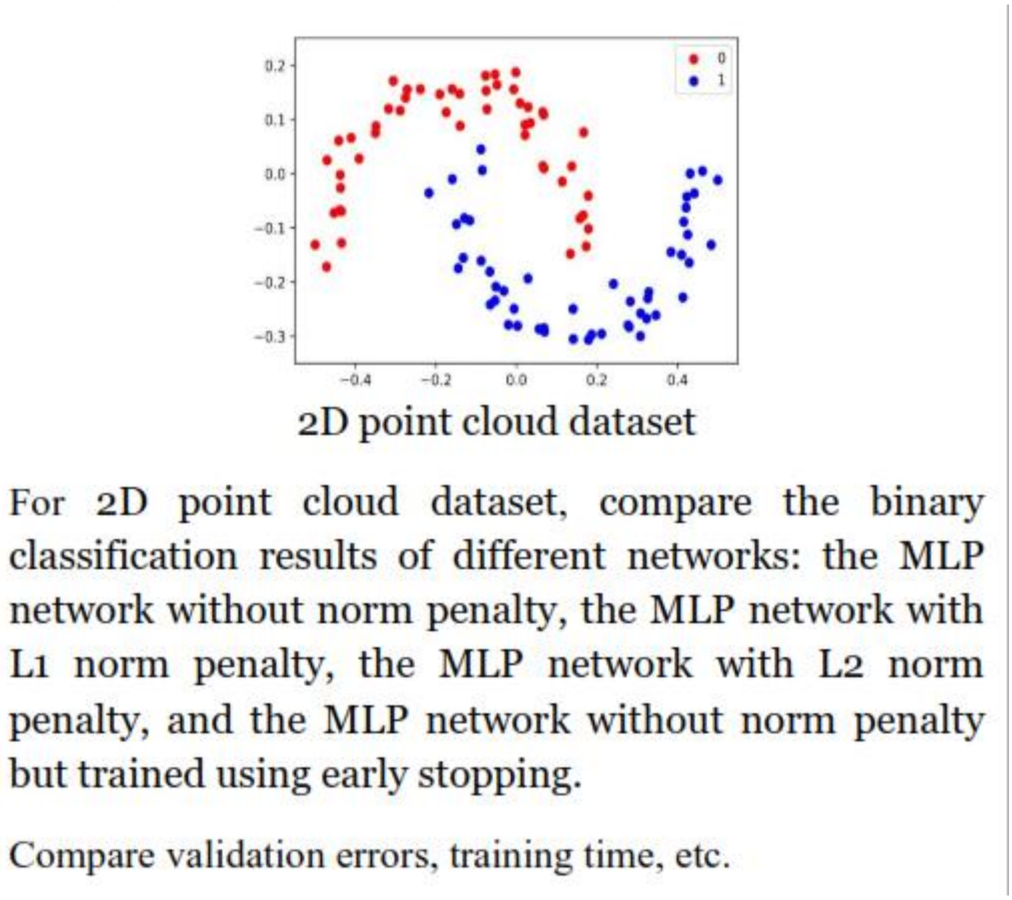
$\mathbb{E}[\mathbf{v}^t] = V(1 - \beta_2) \frac{1-\beta_2^{t+1}}{1-\beta_2} = V(1 - \beta_2^{t+1})$

Finally, the ratio between $\mathbb{E}[\mathbf{v}^t]$ and V is:

$\frac{\mathbb{E}[\mathbf{v}^t]}{V} = 1 - \beta_2^{t+1}$

This ratio indicates how the bias correction term $(1 - \beta_2^{t+1})$ adjusts $\mathbf{v}^t$ to account for the initialization bias when $\mathbf{v}^0$ is zero.

2. Binary Classification on 2D Point Cloud



2D point cloud dataset

For 2D point cloud dataset, compare the binary classification results of different networks: the MLP network without norm penalty, the MLP network with L1 norm penalty, the MLP network with L2 norm penalty, and the MLP network without norm penalty but trained using early stopping.

Compare validation errors, training time, etc.

2-1- Training with different approaches and comparing loss values

In this task, different approaches which has been mentioned in the question are implemented. Fig (1) shows the training losses for different approaches which are applied to the MLP network. Due to Fig (1), it is obvious that L2 norm training has the highest values, noting the equation for the calculation in the loss function. L2 loss performed better in training, however, the added values to the loss made that higher than normal and early stopping training. The patience is considered 5 in early stopping, so, it is stopped earlier due to validation errors.

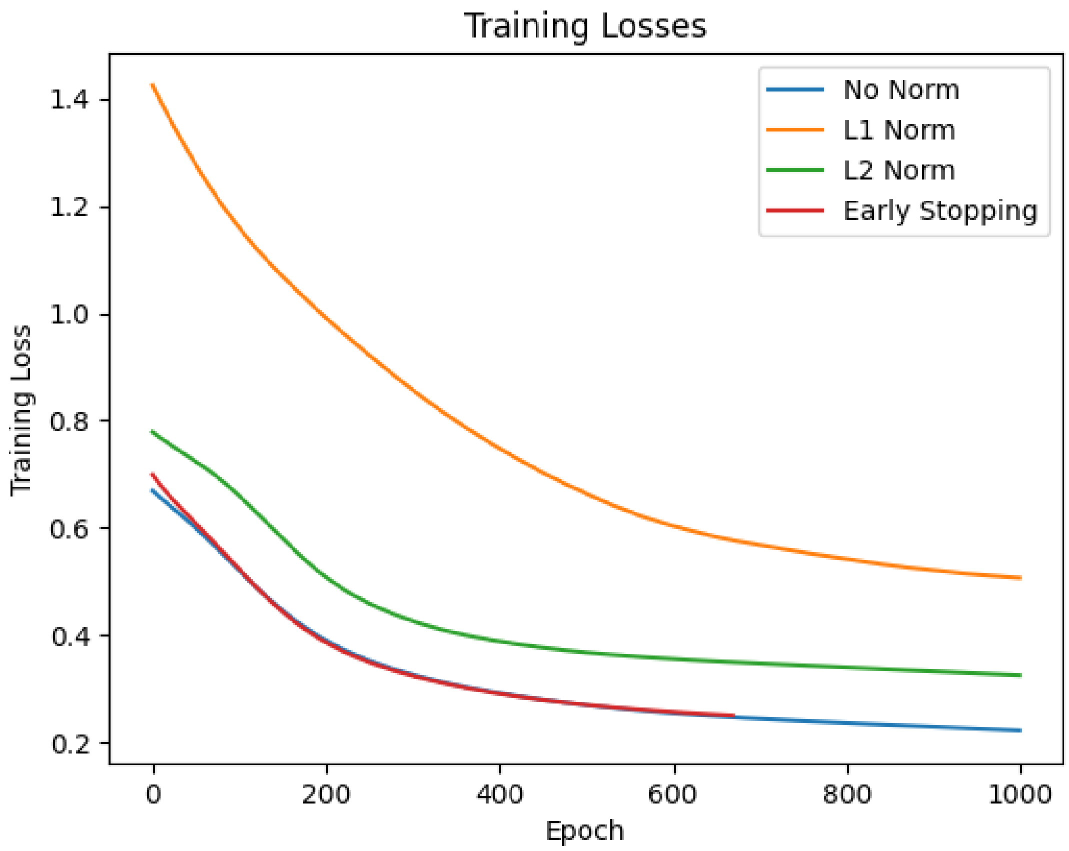


Figure 1. Training Losses over epochs for different approaches 2D data

2-2- Training with different approaches and comparing validation error

Fig (2) shows that validation errors for training with L2 norm became better in training after considerable epochs. The validation error shows that training for every method is obviously depends on the loss calculation. The most important point in Fig (2) might be that in the las steps, L2 norm regulation had the best value in validation set; therefore, the generalization should better by this point.

Fig (3) indicates the training time for each method. It can be seen that during the training the normal method with no norm had mostly had the lesser time due complexity and calculations

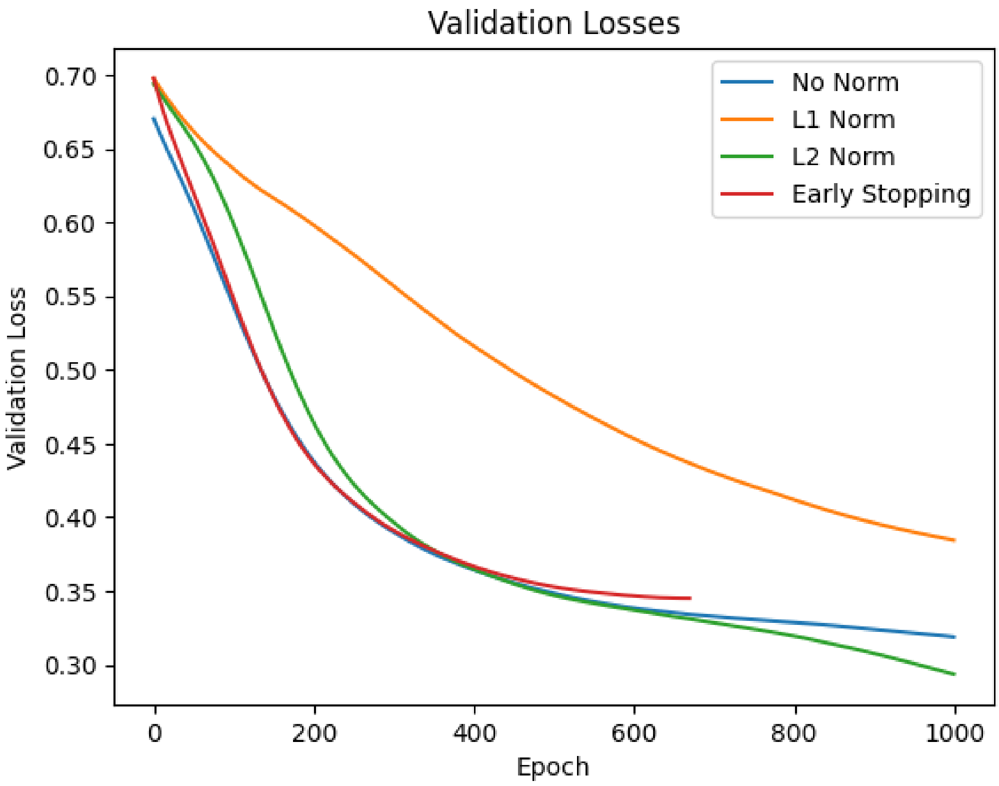


Figure 2. Validation Losses over epochs for different approaches 2D data

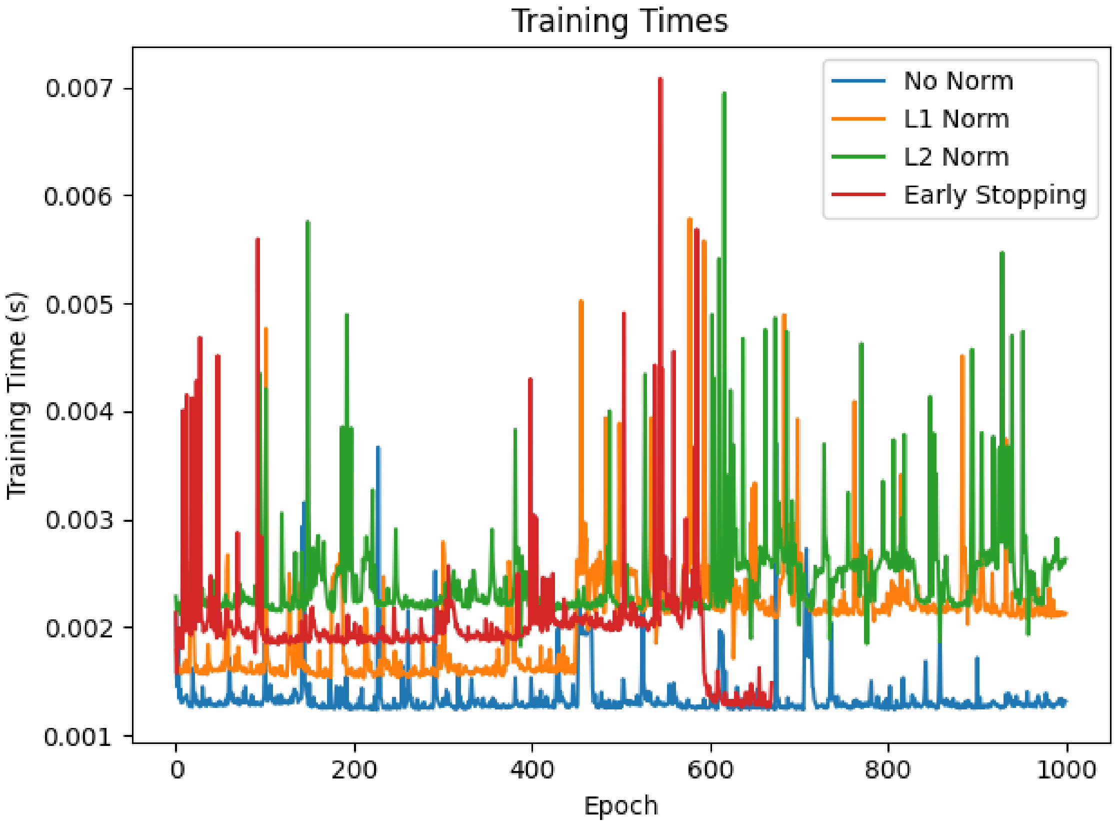
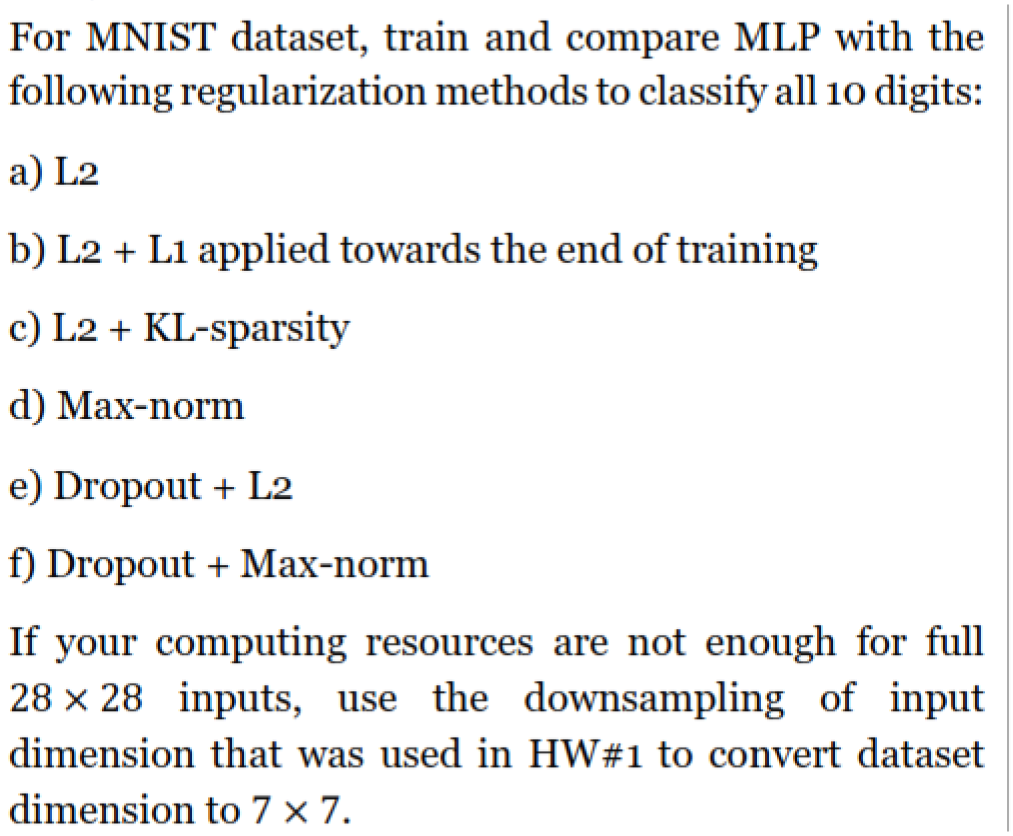


Figure 3. Training time for different approaches for 2D data

3. Image Classification on MNIST



For MNIST dataset, train and compare MLP with the following regularization methods to classify all 10 digits:

a) L2

b) L2 + L1 applied towards the end of training

c) L2 + KL-sparsity

d) Max-norm

e) Dropout + L2

f) Dropout + Max-norm

If your computing resources are not enough for full 28 × 28 inputs, use the downsampling of input dimension that was used in HW#1 to convert dataset dimension to 7 × 7.

a) L2 norm

Fig (4) shows the training loss values over iterations and Fig(5) during epochs, considering norm L2, the MLP network has one hidden layer 32 neurons using Relu and Softmax activation function in output layer. Considering 50 epochs for training, the loss values has been converged to under 1.6 and validation losses (Fig (6)) to about 1.52. By using this regularization method the test error is about 5.6%.

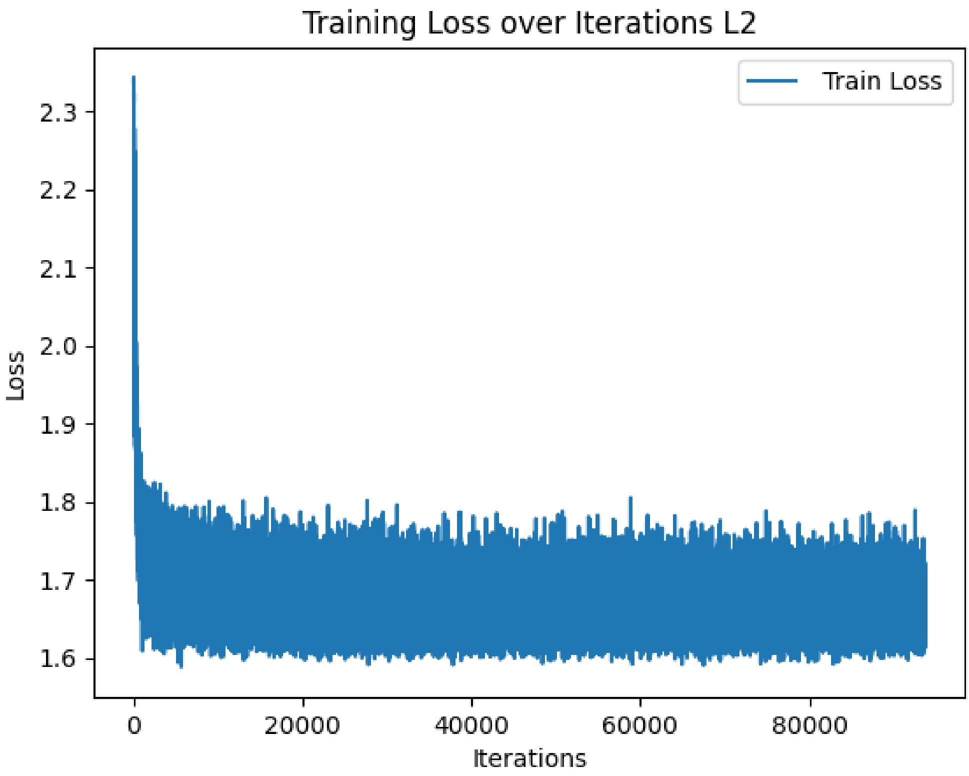


Figure 4. Training Loss values over iterations L2 for mnist data

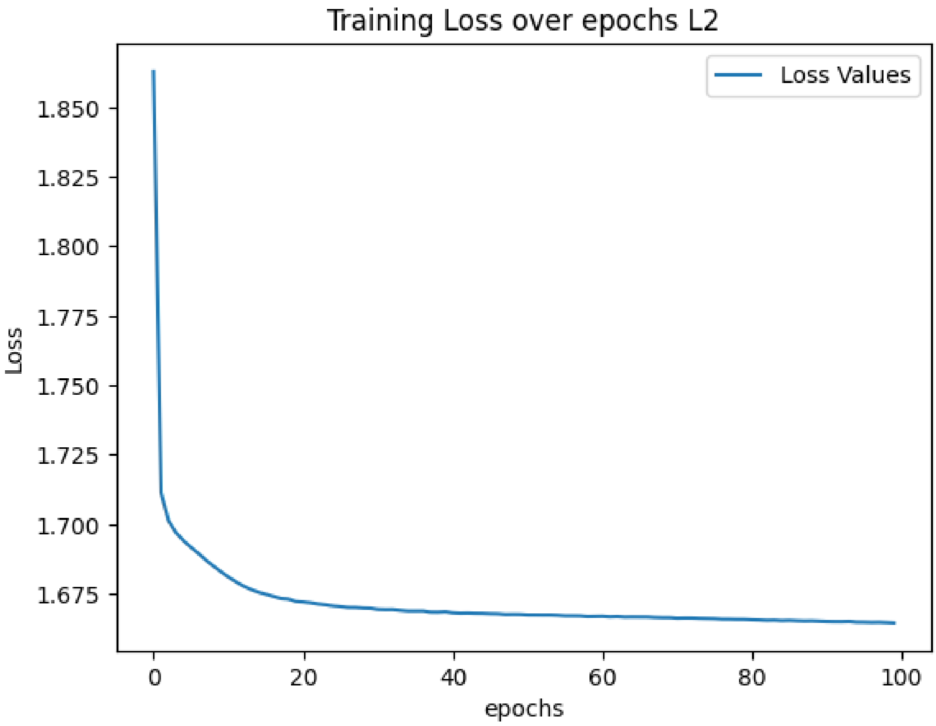


Figure 5. Training Loss values over epochs L2 for mnist data

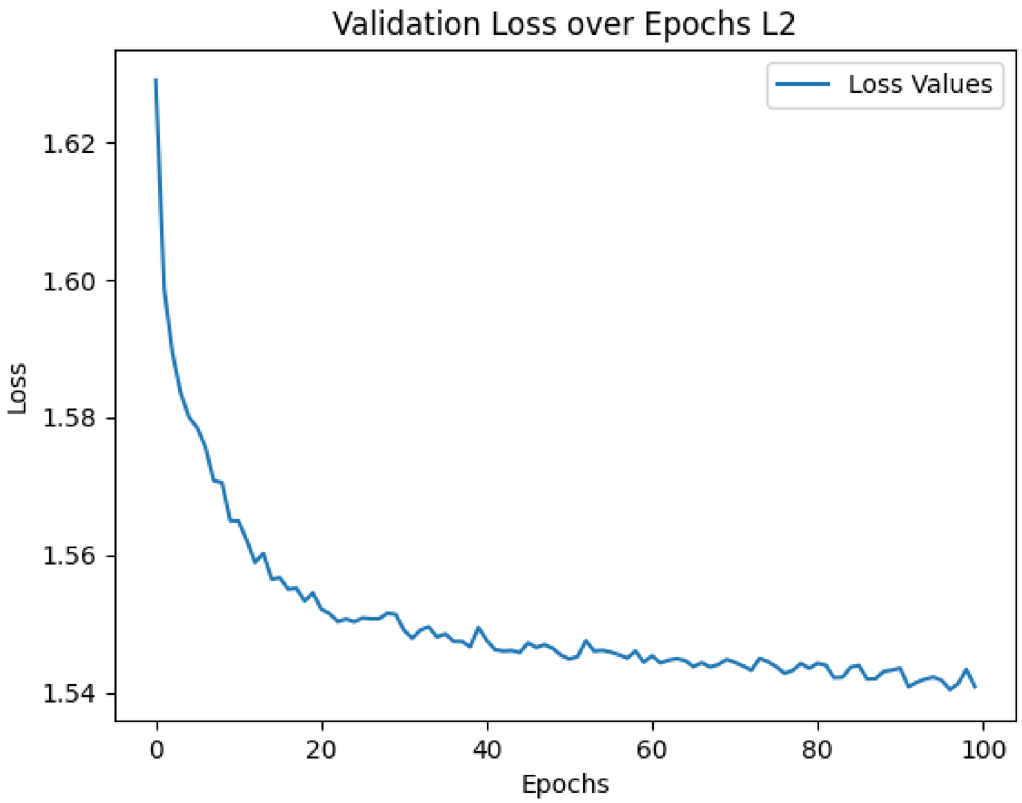


Figure 6. Validation Loss values over epochs L2 for mnist data

b) L2+L1applied towards the end of training

By applying L2 norm for training, the training over iterations (Fig (7)) is the same with last part; however, the loss values during iterations shows a jump and then start to converge again but training time has been excluded. The loss values over epochs for training in Fig (8) and loss values for validation (Fig(9)) are also depicts the same jump and continue to reduce. Nevertheless, it is worth to mention that by this training method the test error rise to about 35%!

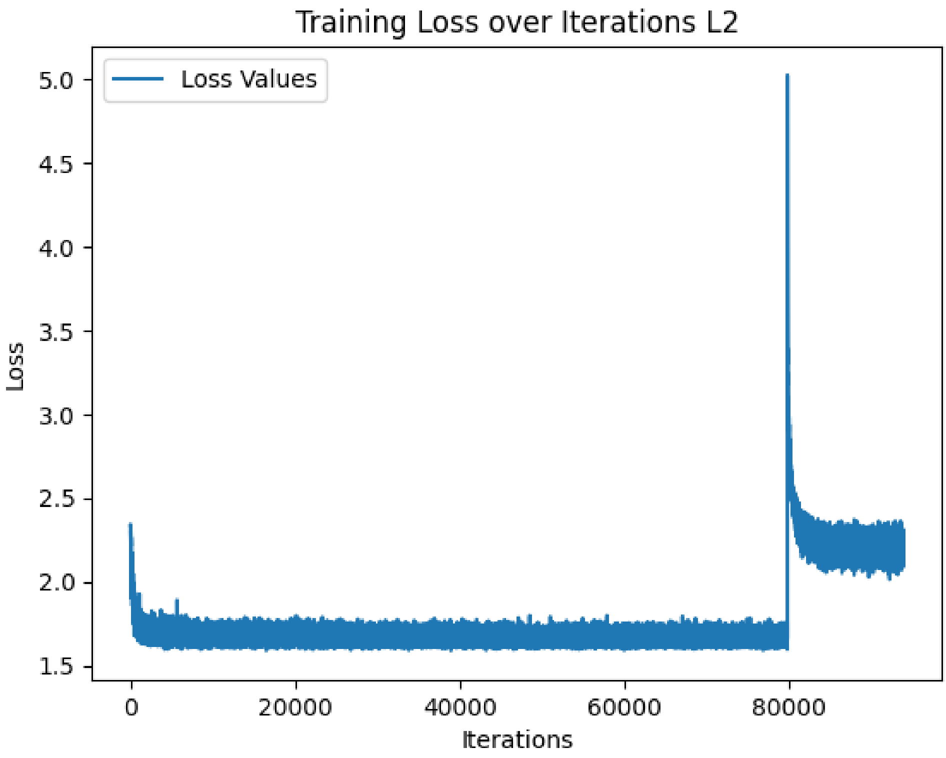


Figure 7. Training Loss values over iterations L2+L1 for mnist data

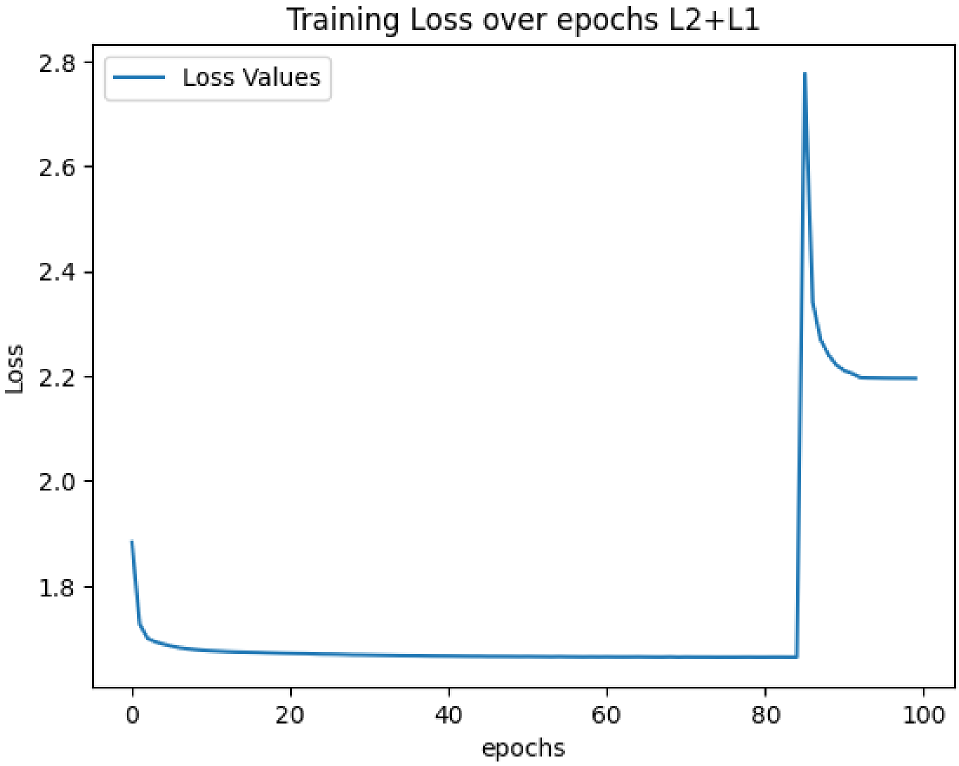


Figure 8. Training Loss values over epochs L2+L1 for mnist data

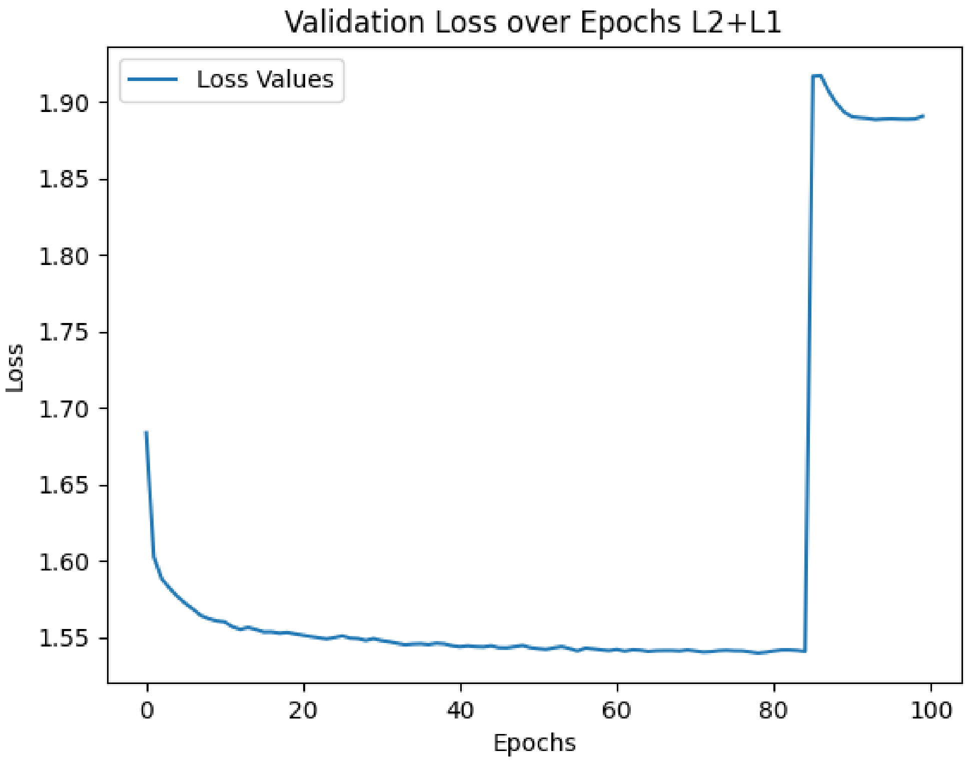


Figure 9. Validation Loss values over epochs L2+L1 for mnist data

c) L2+KL sparsity

Comparing to L2 method, this one by adding KL to loss values behave more smooth in training over epochs which has been shown in Fig (11). However, loss values over iteration in Fig (10) and validation loss (Fig (12)) might not so differ from L2 approach. The test error is also 6.1%.

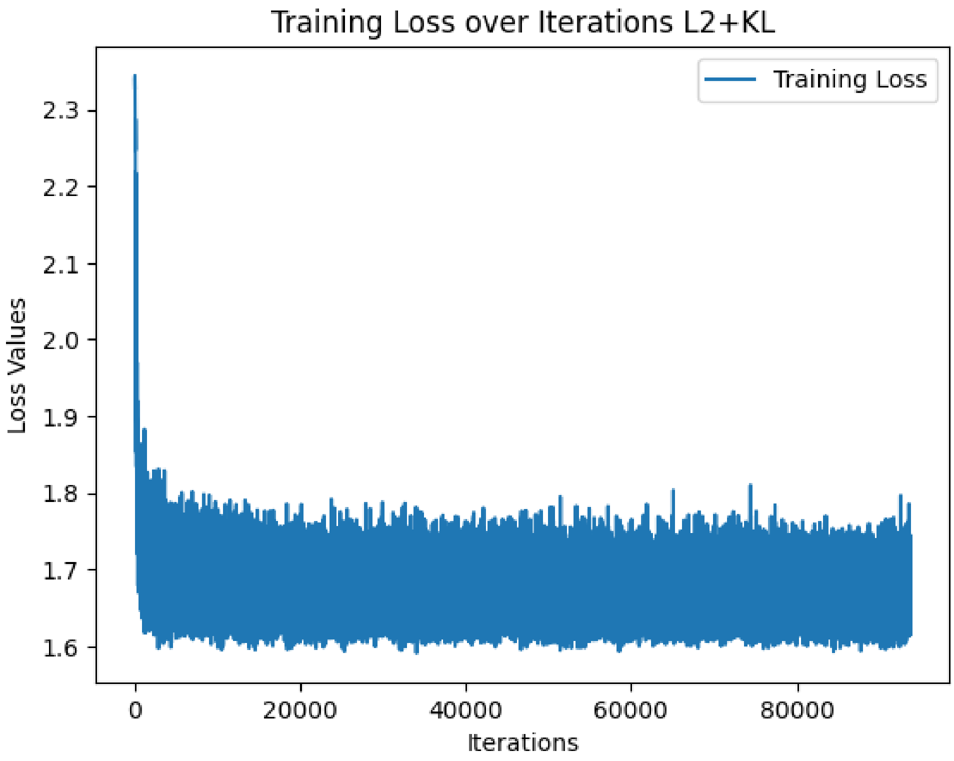


Figure 10. Training Loss values over iterations L2+KL for mnist data

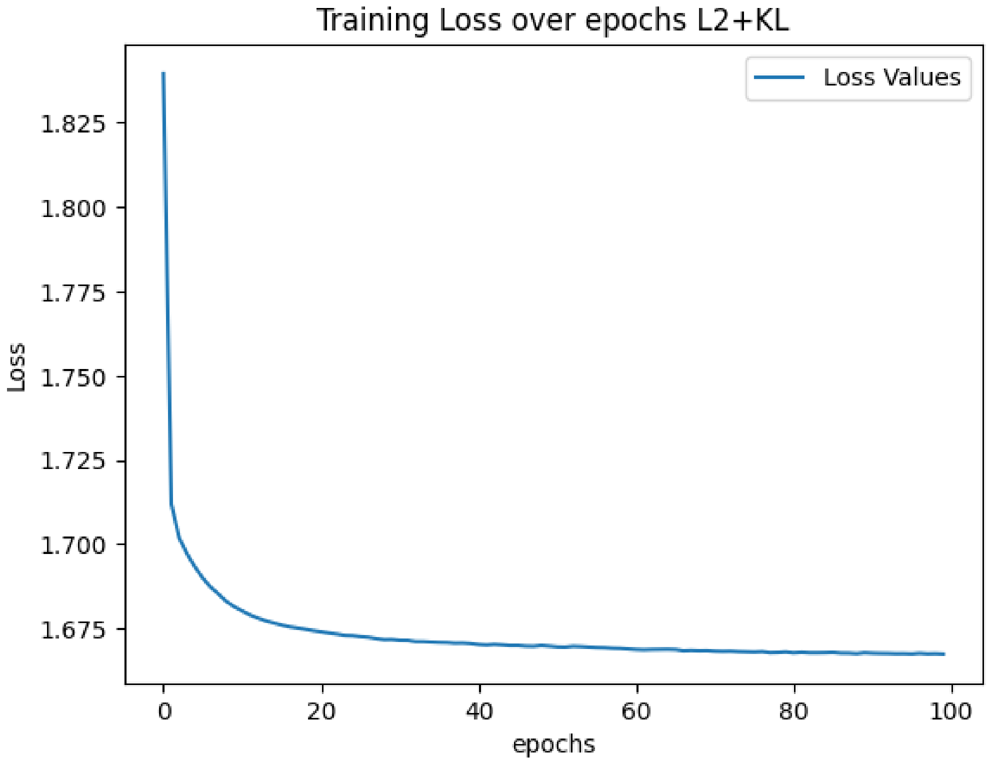


Figure 11. Training Loss values over epochs L2+KL for mnist data

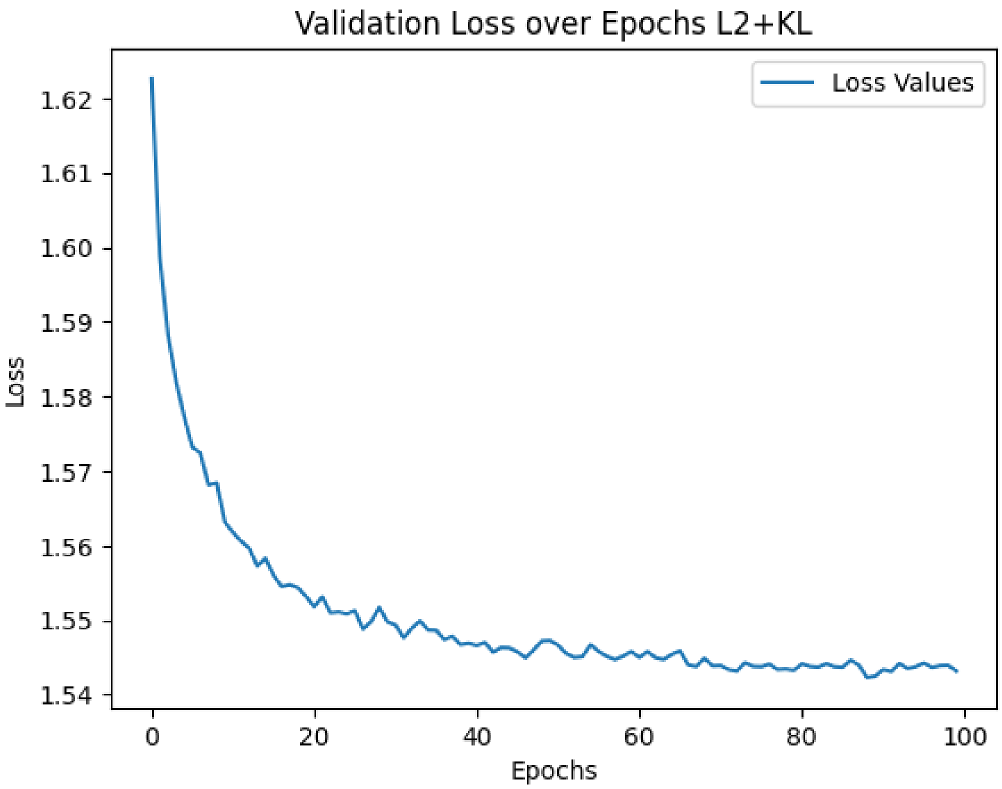


Figure 12. Validation Loss values over epochs L2+KL for mnist data

d) Max-Norm

In this approach, there is clamp rate which has been set to 3 for training. Fig (13) shows the loss error over iterations for training which is obvious that has been clamped at first steps and forced to be less by this approach. Moreover, the training loss in Fig (14) and validation loss in Fig (15) depicts the smooth outputs by clamping in training. The test error is also lesser that last part and has been obtained 4.6%.

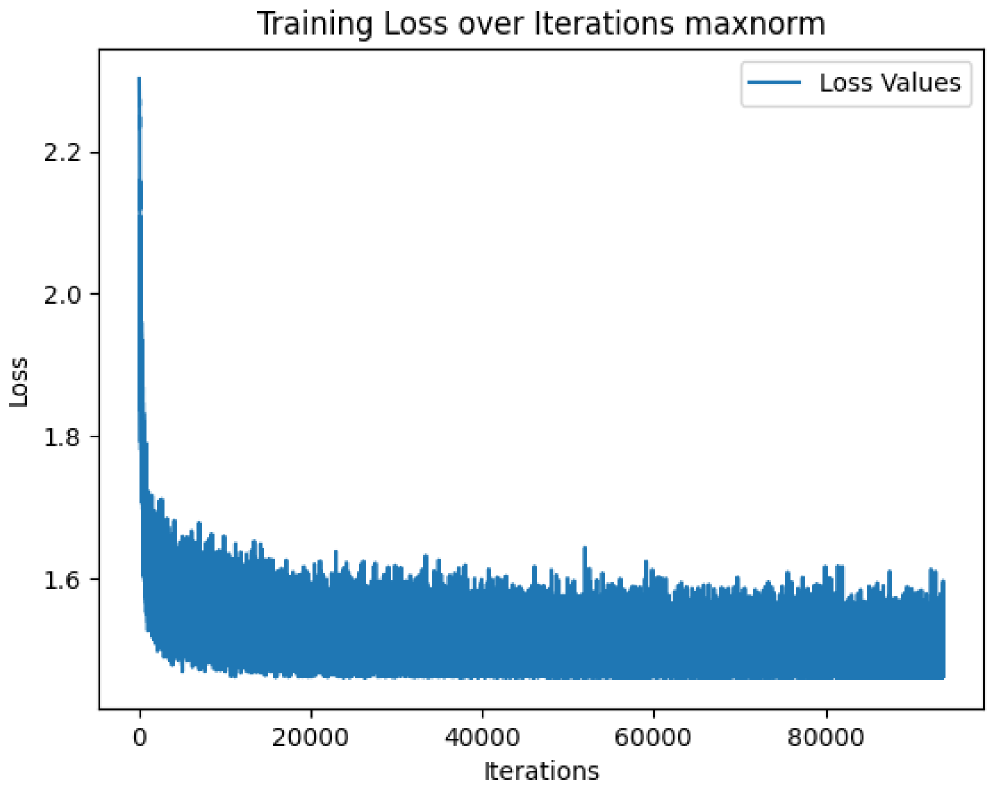


Figure 13. Training Loss values over iterations Max-Norm for mnist data

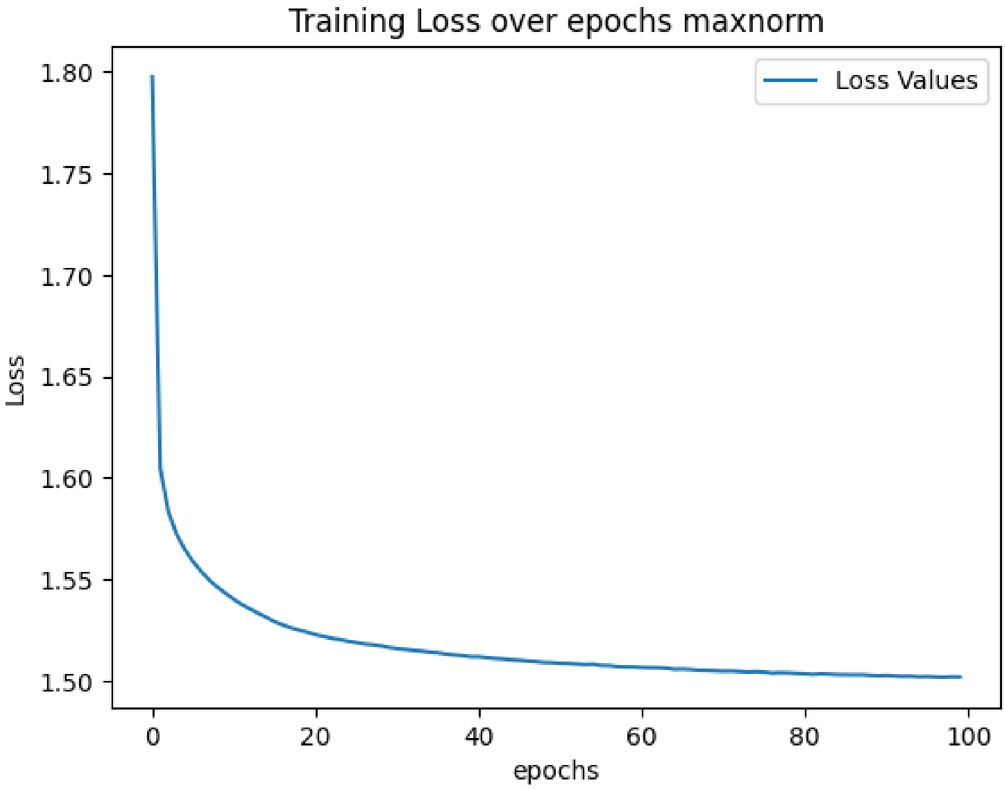


Figure 14. Training Loss values over epochs Max-Norm for mnist data

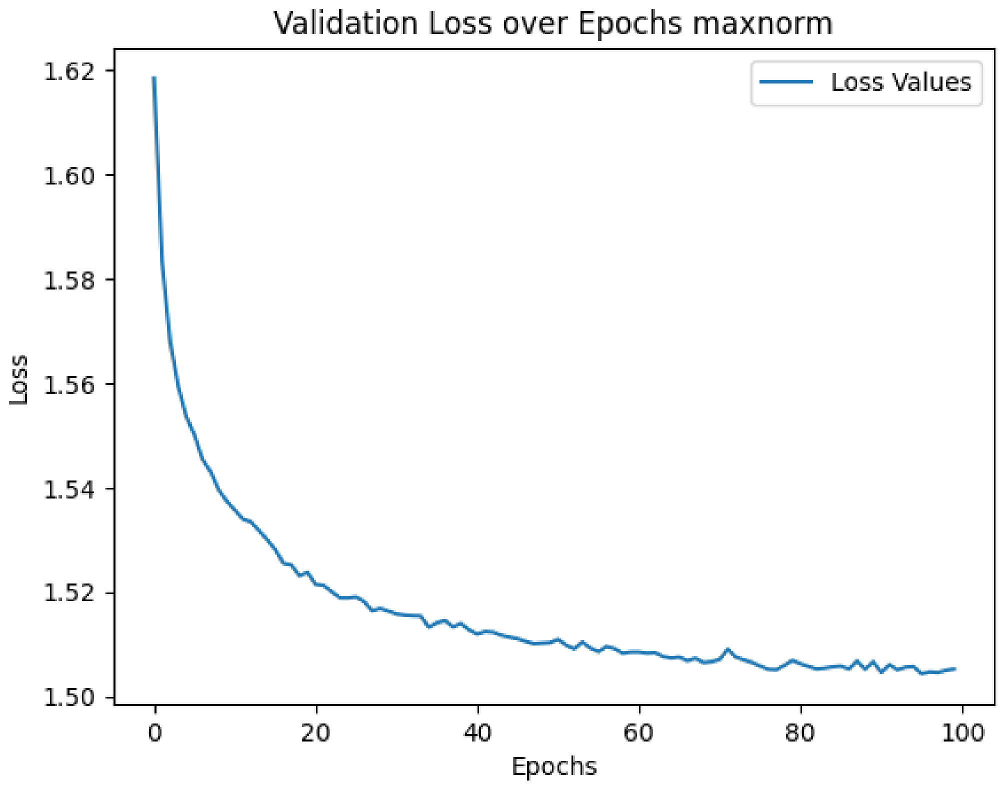


Figure 15. Validation Loss values over epochs Max-Norm for mnist data

e) Dropout+L2

In this part Dropout method has been applied in the network with p=0.5, and L2 loss has been considered for calculating loss. The loss values over iterations has been shown in Fig (16) and over epochs (Fig (17)) and validation losses in Fig (18). The results show that considering dropout with L2 might not perform well and it is worse than L2 approach.

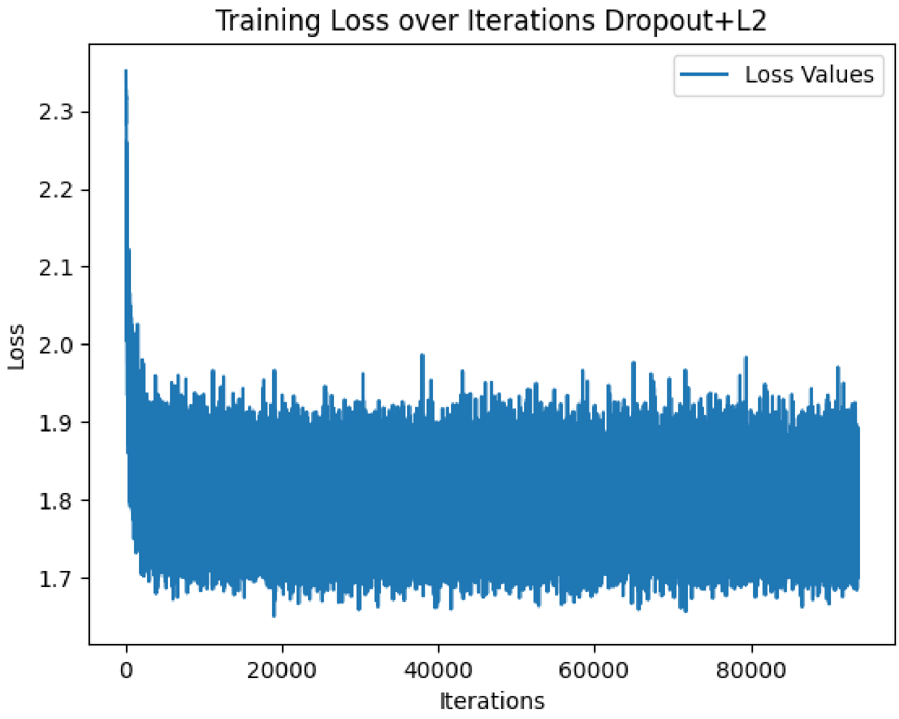


Figure 16. Training Loss values over iterations Droput+L2 for mnist data

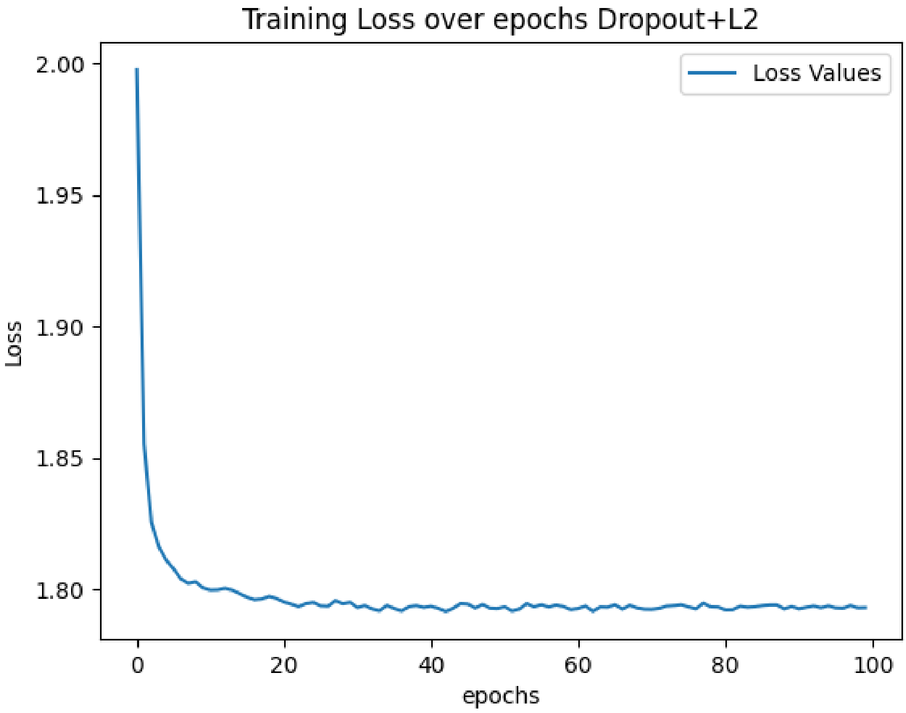


Figure 17. Training Loss values over epochs Dropout+L2 for mnist data

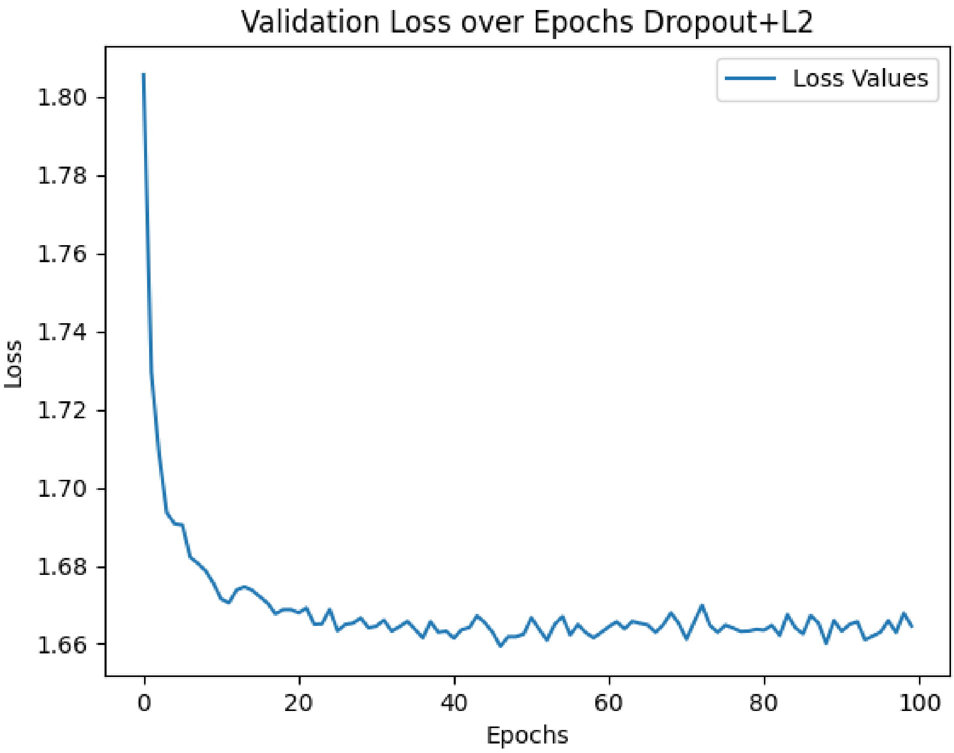


Figure 18. Validation Loss values over epochs Dropout+L2 for mnist dat

f) Dropou+Max-Norm

In one the last parts, max nom clamping showed better performance than other methods. Now by applying dropout the training loss over iteration has been shown in Fig (19), over epochs in Fig (20), and validation loss in Fig (21). Comparing all results to max norm, it seems the max norm approach performed better solely; however, it can be concluded that dropout method with max norm performs better than dropout with L2 norm.

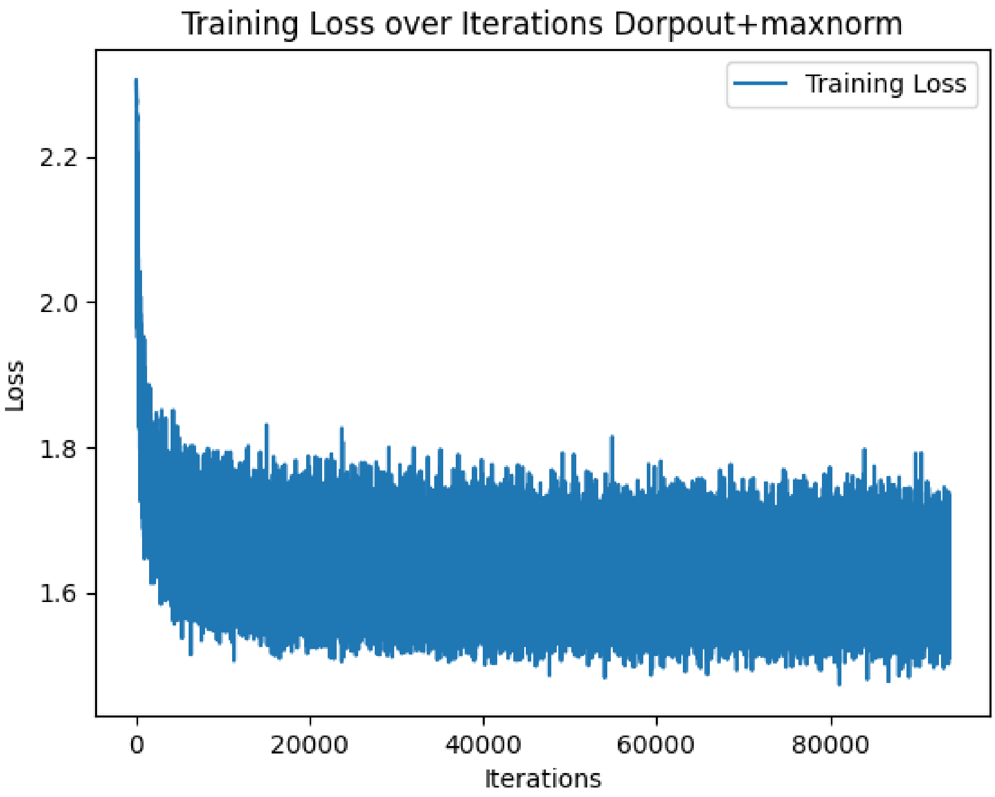


Figure 19. Training Loss values over iterations Dropout+Max-Norm for mnist data

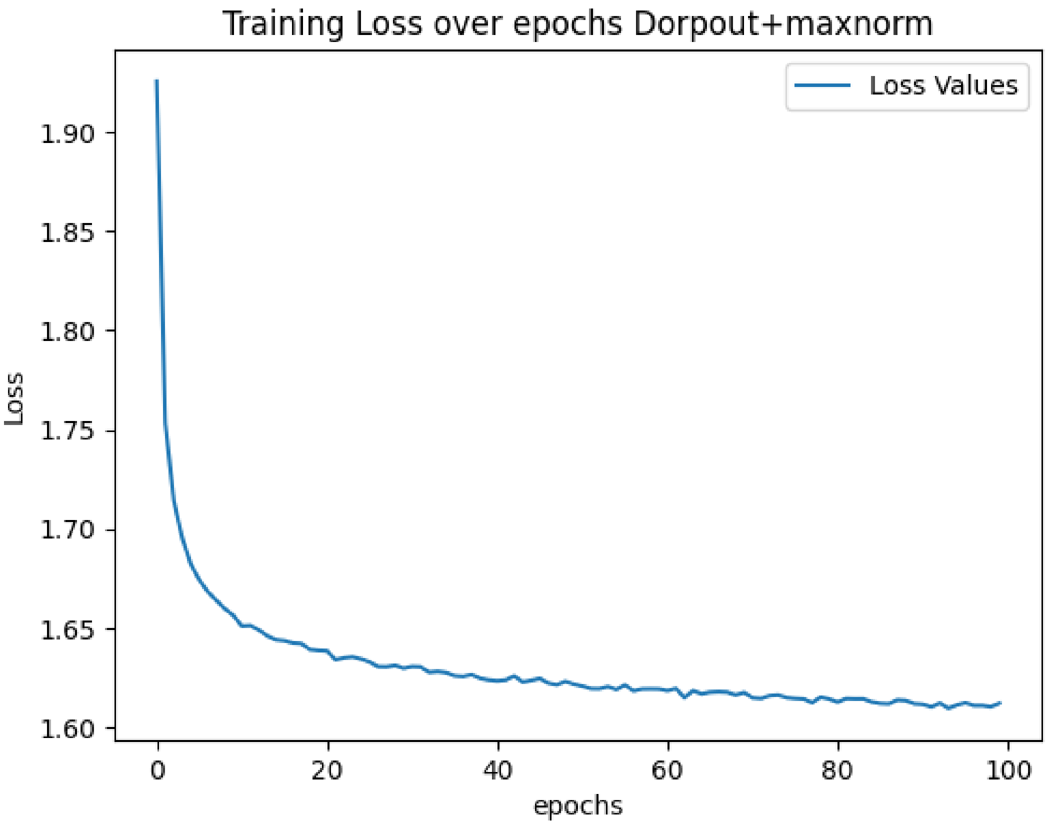


Figure 20. Training Loss values over epochs Dropout+Max-Norm for mnist data

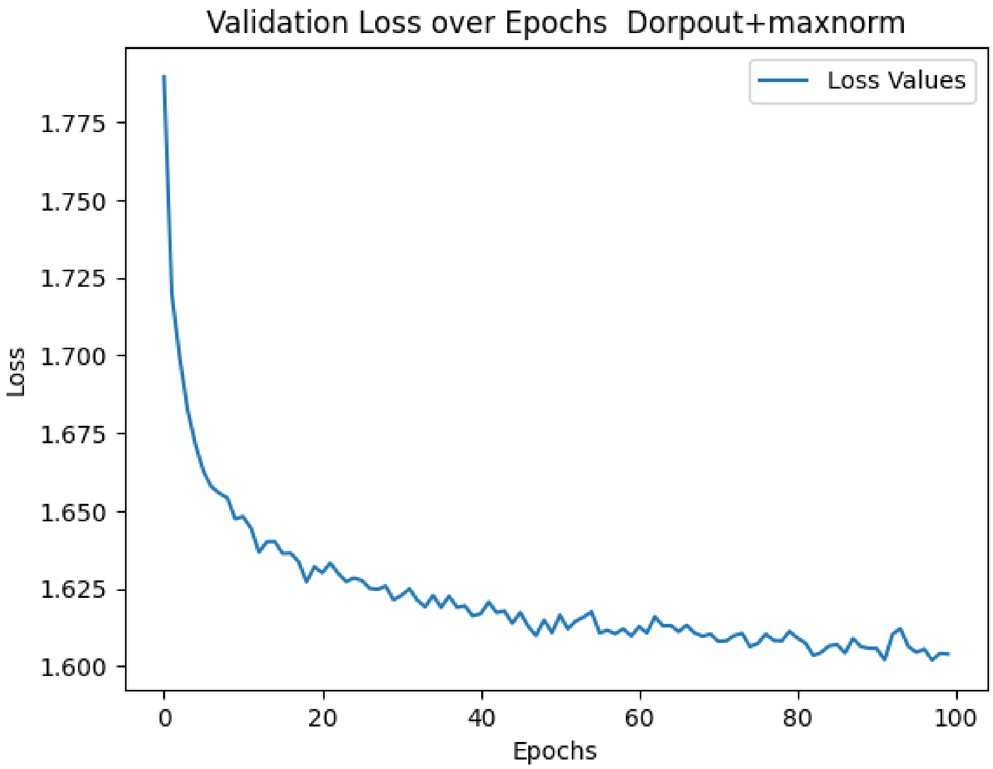


Figure 21. Validation Loss values over epochs Dropout+Max-Norm for mnist dat

3-1- comparing all methods training, validation loss and test errors

Fig (22) shows all training losses with different approaches, comparing all results together shows that the worst training loss over epochs belongs to dropout +L2 with excluding the L1 part of L2+L1. The best performance is also for max norm training. In validation sets, the same scenario can be seen; however, the validation loss of L2+KL is better than droput+maxnorm which is the opposite in training loss values. Test errors are also shown in Fig (23) which shows the performance od the network in testing.

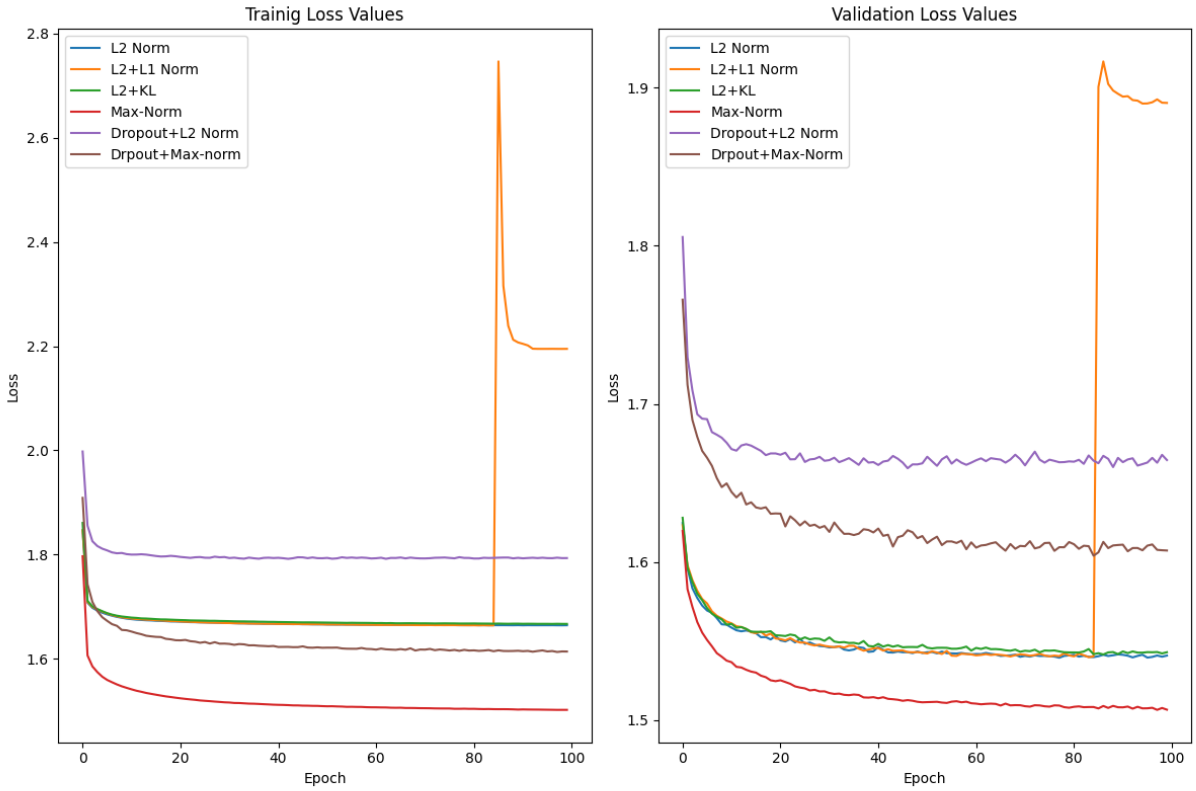


Figure 22.Training and validation losses with different approaches comparison

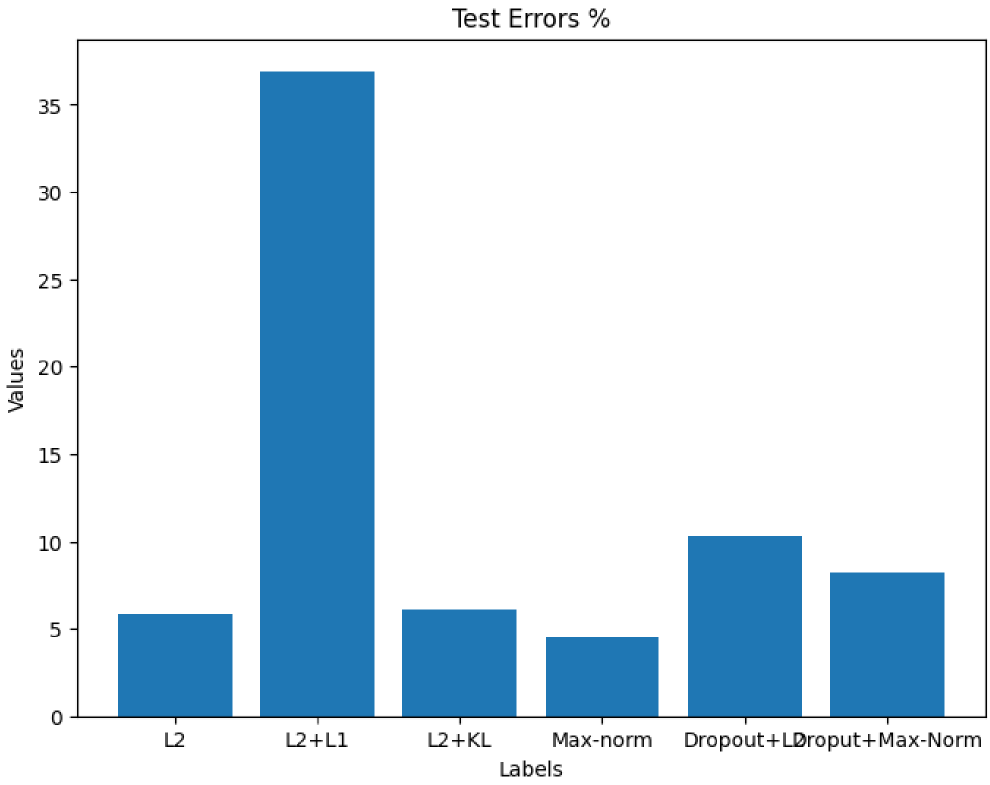


Figure 23.Test Errors for different approaches