# CS725 - Foundations Of Machine Learning

Homework - 2


| | |
|---|---|
| Debdoot Roy Chowdhury | 23M0765 |
| Pranab Kumar Paul | 23M0800 |

# Contents

# 1 Simple Dataset

The Simple dataset contains 768 data points and each data point has two features and can be classified into 4 classes/labels. Here we have trained our model with two different numbers of hidden layers (layers except the input and output layer) i.e., 1 and 2 respectively. We have also given the data as well as the visualization for these layers w.r.t to various hyperparameters.

## 1.1 Learning Rate

A learning rate is a hyperparameter used in machine learning and optimization algorithms to control the step size at which a model's parameters (e.g., weights) are updated during training.

The primary goal of the majority of gradient descent-based optimisation techniques used in machine learning algorithms is to reduce the loss function by repeatedly altering model parameters or weights. One such hyperparameter, Learning Rate, controls how far we move in the direction of the negative gradient. The following parameters are updated with the aid of the learning rate:

$$w = w - \lambda \Delta L(w) \tag{1}$$

where $w$ is the parameters i.e., weights, $\lambda$ is the learning rate and $\Delta L(w)$ is the gradient of the loss function.

The observed effect of the learning rate on the loss and accuracy in our model is shown below:

### 1.1.1 Data

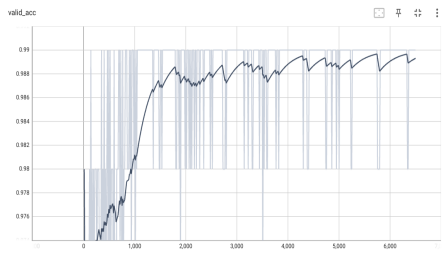
| **Learning Rate** | **Hidden Layer(s)** | **Tain Loss** | **Train Accuracy(%)** | **Validation Loss** | **Validation Accuracy(%)** |
|---|---|---|---|---|---|
| **1e-1** | 1 | 0.010 | 100.00 | 0.040 | 99.00 |
| **1e-2** | 1 | 0.009 | 100.00 | 0.040 | 99.00 |
| **1e-3** | 1 | 0.023 | 100.00 | 0.039 | 99.00 |
| **1e-4** | 1 | 0.051 | 100.00 | 0.126 | 98.00 |

Table 1: Table showing the observations for the model when No. of epochs = 500, seed = 1618(default), and No. of hidden layer = 1.

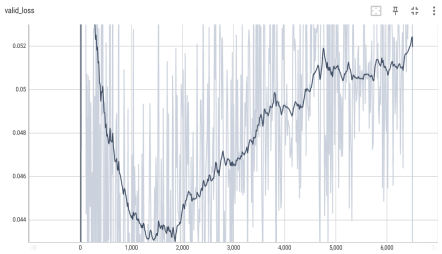
| **Learning Rate** | **Hidden Layer(s)** | **Tain Loss** | **Train Accuracy(%)** | **Validation Loss** | **Validation Accuracy(%)** |
|---|---|---|---|---|---|
| **1e-1** | 2 | 0.013 | 100.00 | 0.027 | 100.00 |
| **1e-2** | 2 | 0.002 | 100.00 | 0.043 | 99.00 |
| **1e-3** | 2 | 0.015 | 100.00 | 0.037 | 99.00 |
| **1e-4** | 2 | 0.038 | 100.00 | 0.187 | 98.00 |

Table 2: Table showing the observations for the model when No. of epochs = 500, seed = 1618(default), and No. of hidden layers = 2.

### 1.1.2 Visualization

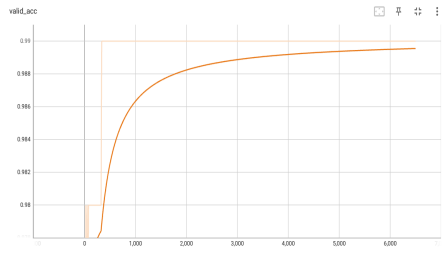



(a) Accuracy Plot

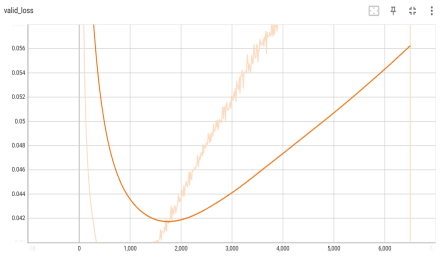



(b) Loss Plot

Figure 1: Learning Rate = $10^{-1}$

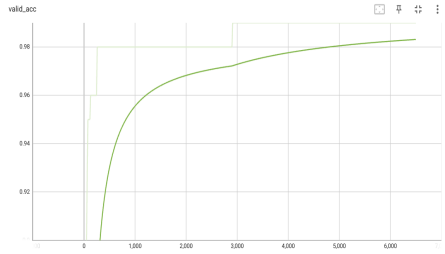



(a) Accuracy Plot

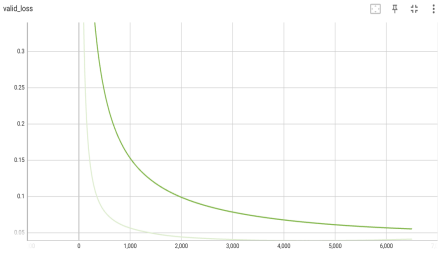



(b) Loss Plot

Figure 2: Learning Rate = $10^{-2}$

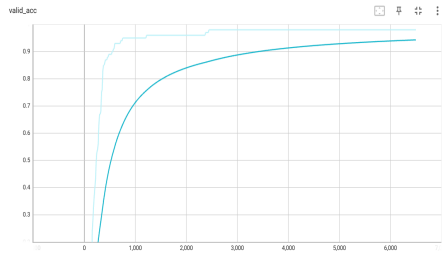



(a) Accuracy Plot

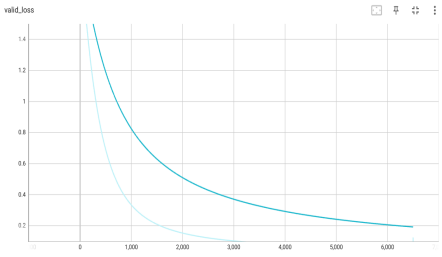



(b) Loss Plot

Figure 3: Learning Rate = $10^{-3}$




(a) Accuracy Plot



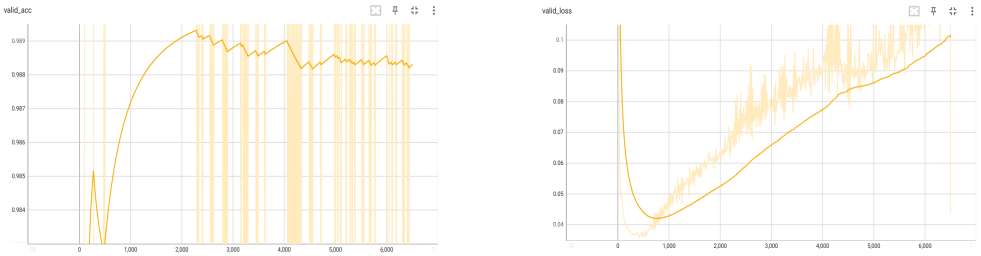

(b) Loss Plot

Figure 4: Learning Rate = $10^{-4}$

**Accuracy plot:** X-axis: Steps Y-axis: Accuracy **Loss plot:** X-axis: Steps Y-axis: Loss
These plots are for the number of hidden layer 1.

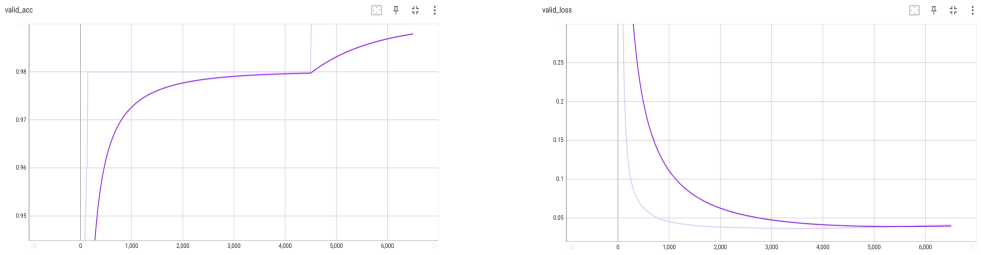

(a) Accuracy Plot (b) Loss Plot

Figure 5: Learning Rate = $10^{-1}$



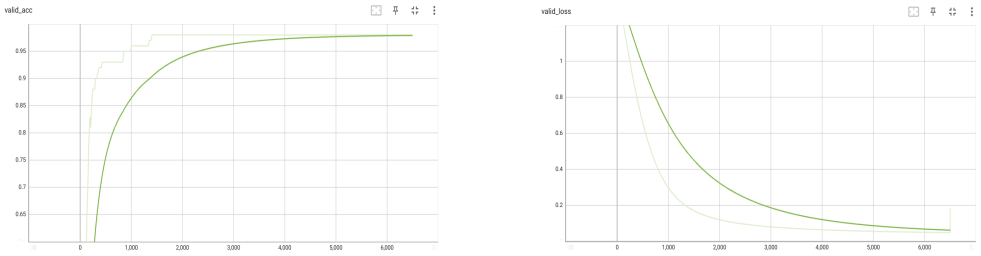

(a) Accuracy Plot (b) Loss Plot

Figure 6: Learning Rate = $10^{-2}$




(a) Accuracy Plot (b) Loss Plot

Figure 7: Learning Rate = $10^{-3}$




(a) Accuracy Plot (b) Loss Plot

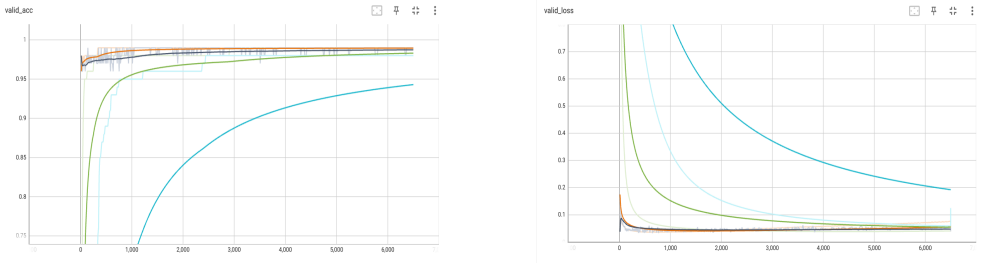
Figure 8: Learning Rate = $10^{-4}$

**Accuracy plot:** X-axis: Steps Y-axis: Accuracy **Loss plot:** X-axis: Steps Y-axis: Loss
These plots are for the number of hidden layer 2.

### 1.1.3 Observation

From the above plots i.e., [Fig. 1 - Fig. 4] We can see the change in validation accuracy and validation loss with respect to the learning rate. In [Fig. 1], as the learning rate is high i.e., $10^{-1}$ the validation loss increases after a certain point of time and converges much earlier and then overshoots. When the learning rate is too low i.e., $10^{-4}$ [Fig. 4] The model is not converging at all. This indicates that the model is overfitting for high learning rates. This is because as the learning rate is too high, the updates are getting too large and result in overshooting the minimum of the loss function, Because of this the training process becomes less stable and may oscillate. These things make it difficult to find a good set of parameters that generalize well on the given data. As the learning rate decreases the model overcomes the problem of overfitting as seen in the figures [Fig. 2 - Fig. 4].

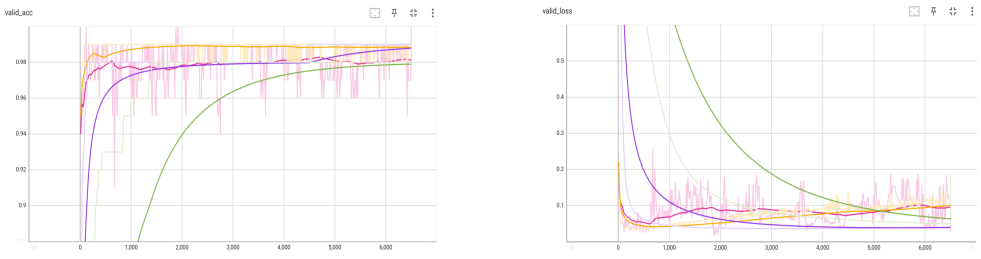
From the [Fig. 3] It's seen that our model is getting the highest validation accuracy and lowest validation loss. We have also given the plot i.e., [Fig. 9] where we can see the accuracy and loss plot for all the hyperparameters mentioned above. So, from the accuracy plot, we can easily observe that, as the learning rate increases with constant epochs and seed the model converges more quickly, and from the loss plot, we can also observe the same but here we can also get additional information about the overfitting of the model.




(a) Accuracy Plot (b) Loss Plot

Figure 9: Number of hidden layers: 1
Grey: $10^{-1}$ Orange: $10^{-2}$ Green: $10^{-3}$ Blue: $10^{-4}$

We have shown the data for the same hyperparameters but for the hidden layer 2 in the below figure [Fig. 10], here we are also observing the same phenomena as above. But here we can see some zig-zag patterns in the accuracy and loss plots. This might happen as we increase the number of hidden layers in the model and the model's capacity to fit the training data also increases and for this, the model also learns to fit the noise causing erratic behavior in the training process. Hence for small datasets less number of hidden layers are adequate to train the model.




(a) Accuracy Plot (b) Loss Plot

Figure 10: Number of hidden layers: 2
Grey: $10^{-1}$, Orange: $10^{-2}$, Green: $10^{-3}$, Blue: $10^{-4}$

In conclusion, from the above observations, we can infer that increasing the learning rate equally increases the convergence rate of the model but if the learning rate is too high then the model becomes overfit after a certain point of time.

## 1.2 Number of Epochs

In machine learning lingo Epochs represent the no. of times the entire training dataset is processed by a learning algorithm. It's an essential hyperparameter in the training of machine learning models. During each epoch, the model updates its parameters based on the training data to minimize a specific loss or error metric. This repetitive process helps the model learn patterns and relationships in the data.

The observed effect of the number of epochs on the loss and accuracy in our model is shown below:

### 1.2.1 Data

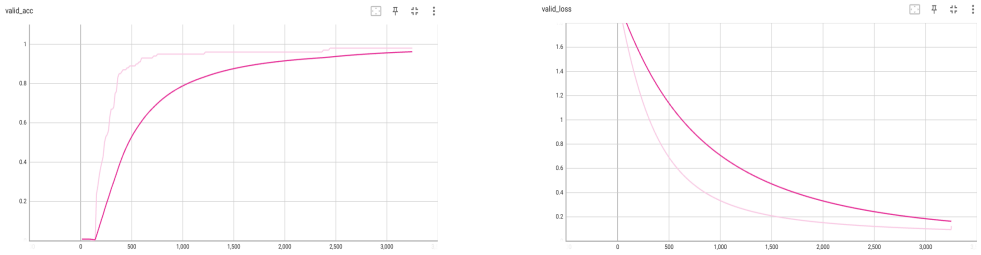
| **Number of Epochs** | **Hidden Layer(s)** | **Tain Loss** | **Train Accuracy(%)** | **Validation Loss** | **Validation Accuracy(%)** |
|---|---|---|---|---|---|
| **250** | 1 | 0.080 | 100.00 | 0.126 | 98.00 |
| **500** | 1 | 0.050 | 100.00 | 0.126 | 98.00 |
| **1000** | 1 | 0.036 | 100.00 | 0.125 | 98.00 |
| **1500** | 1 | 0.030 | 100.00 | 0.043 | 99.00 |

Table 3: Table showing the observation for the model when learning rate = $10^{-4}$, seed = 1618(default) and no. of hidden layers = 1.

| **Number of Epochs** | **Hidden Layer(s)** | **Tain Loss** | **Train Accuracy(%)** | **Validation Loss** | **Validation Accuracy(%)** |
|---|---|---|---|---|---|
| **250** | 2 | 0.042 | 100.00 | 0.113 | 98.00 |
| **500** | 2 | 0.030 | 100.00 | 0.137 | 98.00 |
| **1000** | 2 | 0.024 | 100.00 | 0.038 | 99.00 |
| **1500** | 2 | 0.020 | 100.00 | 0.037 | 99.00 |

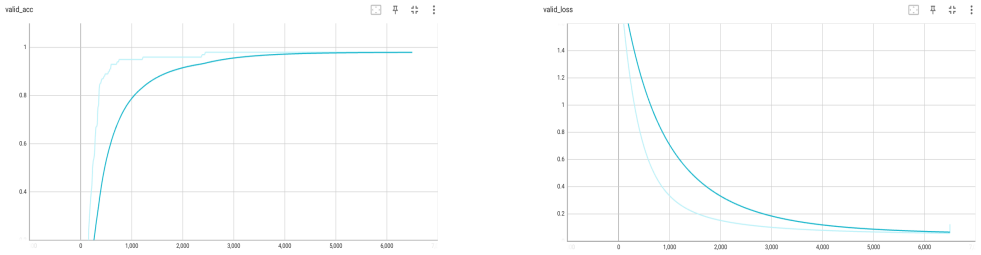
Table 4: Table showing the observation for the model when learning rate = $10^{-4}$, seed = 1618(default) and no. of hidden layers = 2.

### 1.2.2 Visualization



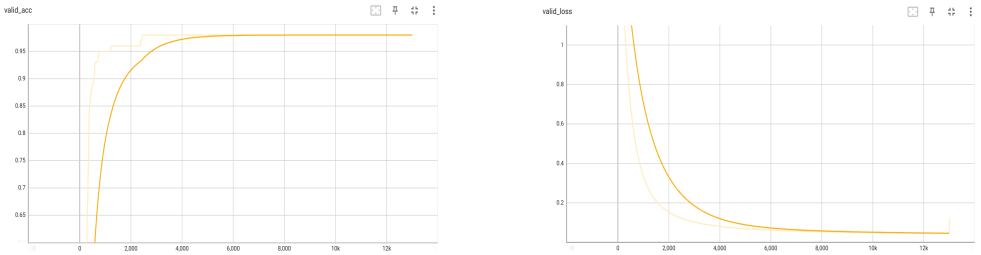

(a) Accuracy Plot (b) Loss Plot

Figure 11: Number of Epochs = 250



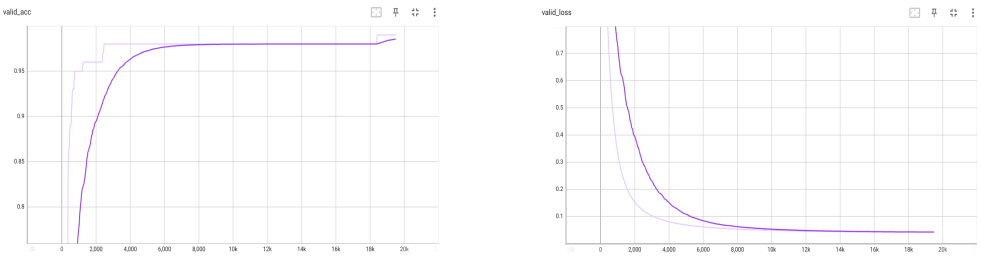

(a) Accuracy Plot (b) Loss Plot

Figure 12: Number of Epochs = 500




(a) Accuracy Plot (b) Loss Plot

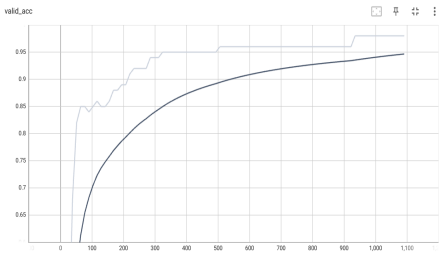
Figure 13: Number of Epochs = 1000



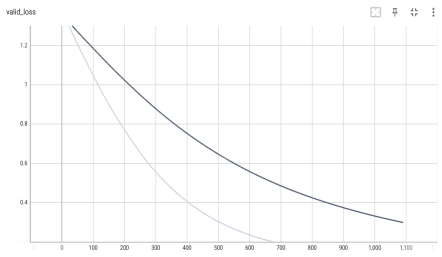

(a) Accuracy Plot (b) Loss Plot

Figure 14: Number of Epochs = 1500

**Accuracy plot:** X-axis: Steps Y-axis: Accuracy **Loss plot:** X-axis: Steps Y-axis: Loss
**Note:** These plots are for the number of hidden layers 1.

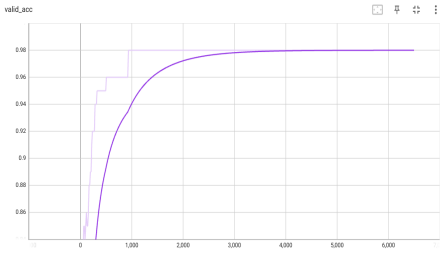



(a) Accuracy Plot

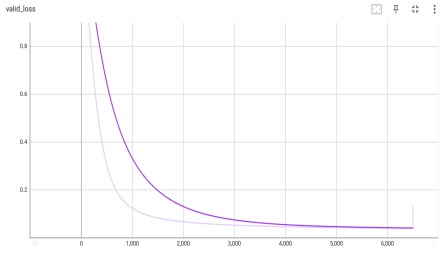



(b) Loss Plot

Figure 15: Number of Epochs = 250

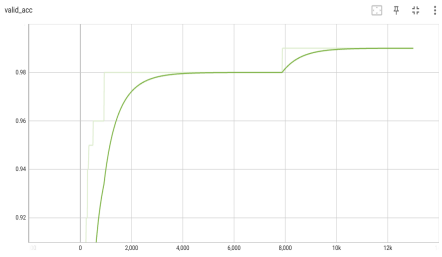



(a) Accuracy Plot

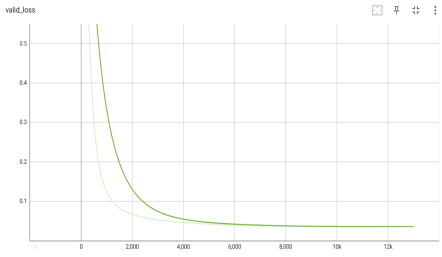



(b) Loss Plot

Figure 16: Number of Epochs = 500

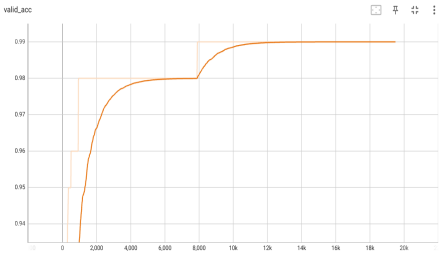



(a) Accuracy Plot

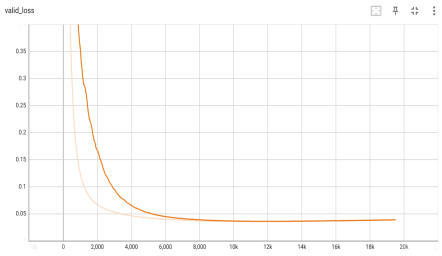



(b) Loss Plot

Figure 17: Number of Epochs = 1000




(a) Accuracy Plot




(b) Loss Plot

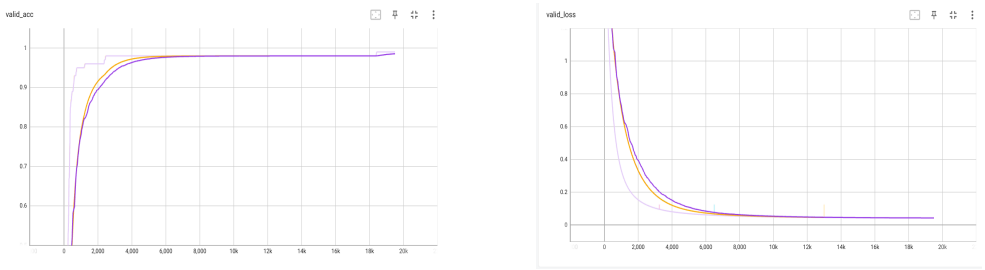
Figure 18: Number of Epochs = 1500

**Accuracy plot:** X-axis: Steps Y-axis: Accuracy **Loss plot:** X-axis: Steps Y-axis: Loss
**Note:** These plots are for the number of hidden layer 2.

### 1.2.3 Observation

In the above figures, we considered the learning rate to be constant at $10^-4$ and seed to be constant at default value i.e., 1618 in our experiment. From there we can easily observe that with the increase in the number of epochs for the specified learning rate the model is converging more accurately. In [Fig. 15] where the number of epochs is only 250 which is much smaller w.r.t the learning rate $10^{-4}$, the model is not converging at all and it needs more epochs to converge correctly as seen from the Loss plot. In contrast with that, in [Fig. 17] For the number of epochs 1000 the model is getting enough time to converge w.r.t the learning rate as we can see from the Loss plot in [Fig. 17].

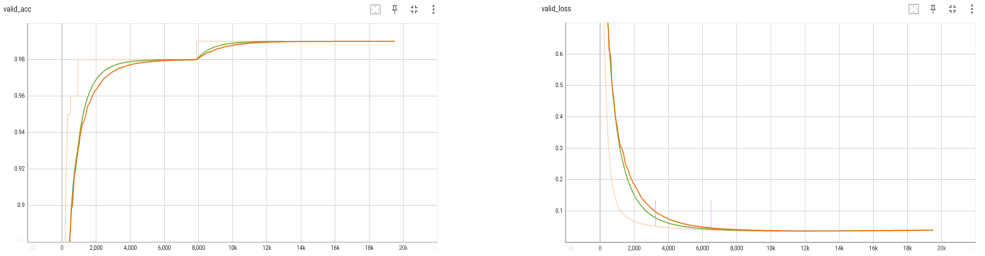
This relation between the accuracy and epochs exists because, as we are increasing the number of epochs we are letting the model more time to converge. That's why as the number of epochs increases the model gets an ample amount of time to train itself and gives a better loss and accuracy for the validation data as shown in [Fig. 19].




(a) Accuracy Plot (b) Loss Plot

Figure 19: Number of hidden layers: 1
Grey: 250, Orange: 500, Green: 1000, Blue: 1500

As we increase the number of hidden layers from 1 to 2, we do not get any significant improvement in the accuracy as well as loss as seen in [Fig. 20]. This suggests that for this type of small (with respect to dimension) dataset, only one hidden layer is sufficient to fit the model accurately. There is no need to add more hidden layers as it is not giving any fruitful results as seen in comparison with [Fig. 19].




(a) Accuracy Plot (b) Loss Plot

Figure 20: Number of hidden layers: 2
Grey: 250, Orange: 500, Green: 1000, Blue: 1500

In conclusion, a larger number of epochs gives the model time to converge to an optimal set of model parameters that minimize the chosen loss function and thus provide better accuracy. However, a very high number of epochs can also make the model overfit and reduce the accuracy of the validation set.

# 2 Digits Dataset

The Digits dataset contains 1024 data points and each data point has 64 features and can be classified into 10 classes/labels. Here we have trained our model with two different numbers of hidden layers (layers except the input and output layer) i.e., 2 and 3 respectively. We have also given the data as well as the visualization for these layers w.r.t to various hyperparameters.

## 2.1 Learning Rate

A learning rate is a hyperparameter used in machine learning and optimization algorithms to control the step size at which a model's parameters (e.g., weights) are updated during training.

By iteratively altering model parameters, such as weights, the main goal of the majority of gradient descent-based optimisation techniques used in machine learning algorithms is to minimise the loss function. The magnitude of the step we take in the direction of the negative gradient is determined by one of the hyperparameters known as Learning Rate. The following parameters are updated with the aid of the learning rate:

$$w = w - \lambda \Delta L(w) \tag{2}$$

where $w$ is the parameters i.e., weights, $\lambda$ is the learning rate and $\Delta L(w)$ is the gradient of the loss function. The observed effect of the learning rate on the loss and accuracy in our model is shown below:

### 2.1.1 Data

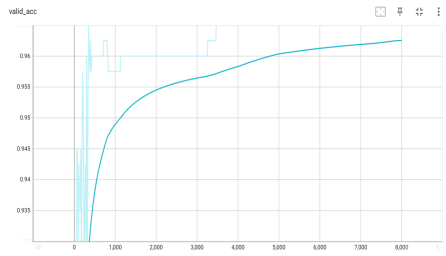
| **Learning Rate** | **Hidden Layer(s)** | **Tain Loss** | **Train Accuracy(%)** | **Validation Loss** | **Validation Accuracy(%)** |
|---|---|---|---|---|---|
| **1e-2** | 2 | 0.003 | 100.00 | 0.196 | 96.75 |
| **1e-3** | 2 | 0.011 | 100.00 | 0.148 | 97.00 |
| **1e-4** | 2 | 0.001 | 100.00 | 0.139 | 97.00 |

Table 5: Table showing the observations for the model when No. of epochs = 500, seed = 1618(default) and number of hidden layers = 2.

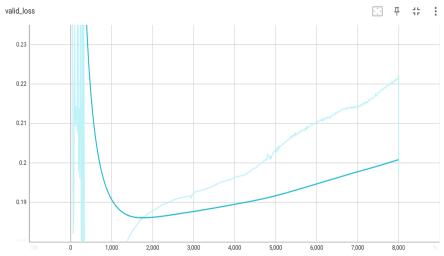
| **Learning Rate** | **Hidden Layer(s)** | **Tain Loss** | **Train Accuracy(%)** | **Validation Loss** | **Validation Accuracy(%)** |
|---|---|---|---|---|---|
| **1e-2** | 3 | 0.023 | 100.00 | 0.180 | 97.75 |
| **1e-3** | 3 | 0.000 | 100.00 | 0.183 | 97.25 |
| **1e-4** | 3 | 0.055 | 100.00 | 0.148 | 97.25 |

Table 6: Table showing the observations for the model when no. of epochs = 500, seed = 1618(default) and no. of hidden layers = 3.

### 2.1.2 Visualization

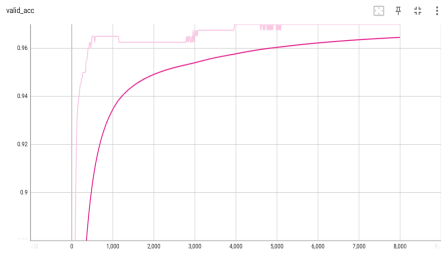



(a) Accuracy Plot

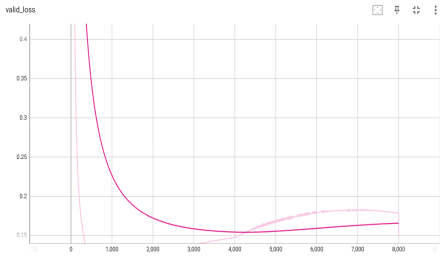



(b) Loss Plot

Figure 21: Learning Rate = $10^{-2}$

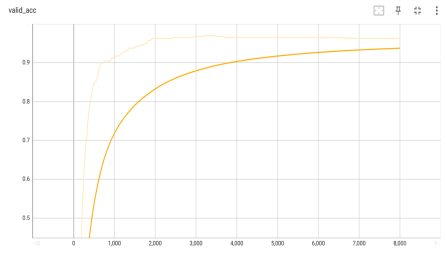



(a) Accuracy Plot

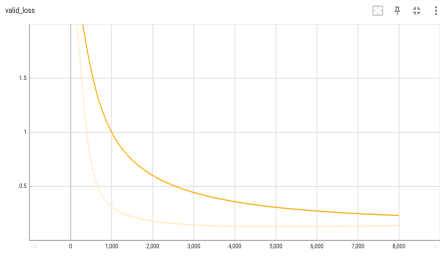



(b) Loss Plot

Figure 22: Learning Rate = $10^{-3}$



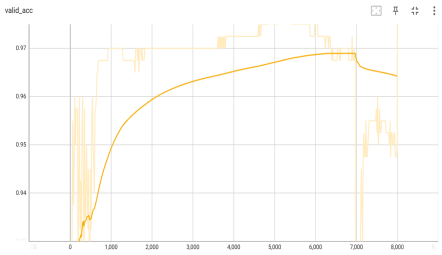

(a) Accuracy Plot



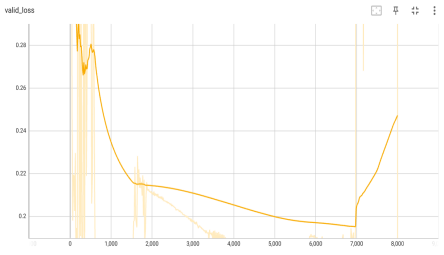

(b) Loss Plot

Figure 23: Learning Rate = $10^{-4}$

**Accuracy plot:** X-axis: Steps Y-axis: Accuracy **Loss plot:** X-axis: Steps Y-axis: Loss
**Note:** These plots are for the number of hidden layers 2.

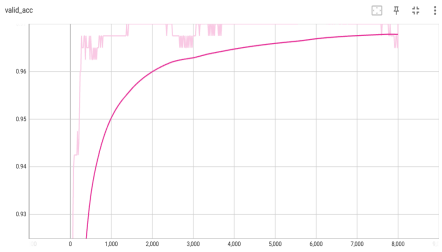



(a) Accuracy Plot

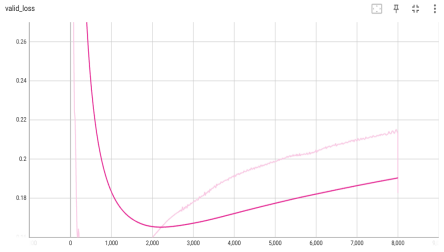



(b) Loss Plot

Figure 24: Learning Rate: $10^{-2}$

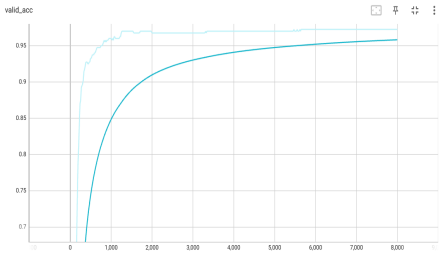



(a) Accuracy Plot

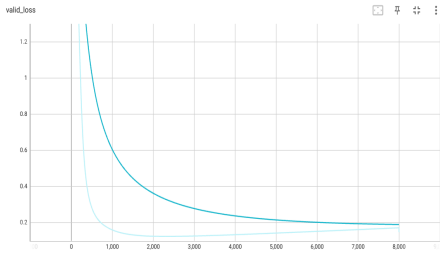



(b) Loss Plot

Figure 25: Learning Rate: $10^{-3}$




(a) Accuracy Plot




(b) Loss Plot

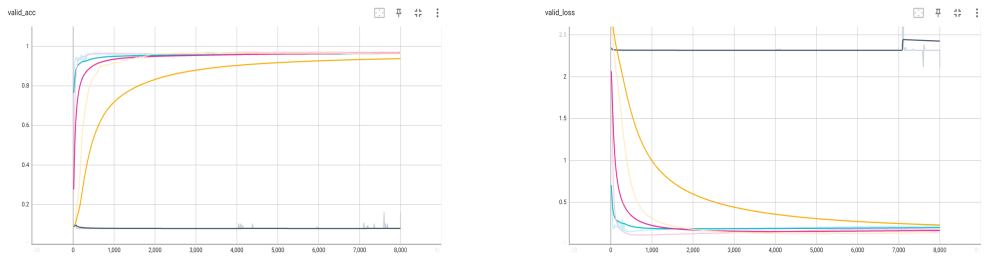
Figure 26: Learning Rate: $10^{-4}$

**Accuracy plot:** X-axis: Steps Y-axis: Accuracy **Loss plot:** X-axis: Steps Y-axis: Loss
**Note:** These plots are for the number of hidden layer 3.

### 2.1.3 Observation

From the above plots [Fig. 21 - Fig. 24], for the number of hidden layers 2, we can see the variations of accuracy and loss of the model with respect to various learning rates. From the loss plot of [Fig. 21(b)] we can see that the model is over-fitted for the learning rate $10^{-2}$ as the loss increases after a certain point of time. But from figure [Fig. 22(b)] we can observe slight overfitting and lastly for the figure [Fig. 23(b)] we can see that there is no overfitting as the curve decreases smoothly.

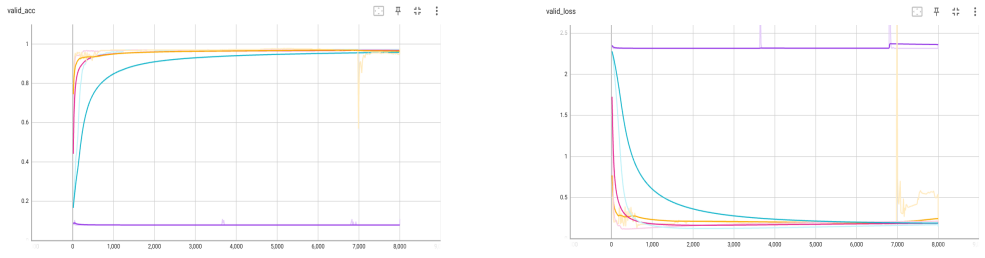
From the above observation, it can be inferred that if the learning rate is too high the model becomes over-fitted, and as the learning rate decreases the overfitting problem also vanishes. As the learning rate is low it allows the model to train properly and if the learning rate is too high then it causes the model to overshoot the loss function and over-trained the weight parameters, which causes oscillations in the plot.




(a) Accuracy Plot (b) Loss Plot

Figure 27: Number of hidden layers: 2

Similarly for the number of hidden layers 3 there are no significant changes in the results. This is because one hidden layer is enough to fit the model. But in [Fig. 24] There are some unusual patterns which might be because of the use of batch gradient descent and the Adam optimizer which we have used in our model.




(a) Accuracy Plot (b) Loss Plot

Figure 28: Number of hidden layers: 3

In conclusion, as the learning rate decreases the model can train its parameters accurately hence giving better accuracy and loss on the validation data.

## 2.2 Number of Epochs

In the language of machine learning the number of epochs indicates how frequently a learning algorithm has gone over the full training dataset. This hyperparameter is crucial for training machine learning models. A specified loss or error measure is minimised throughout each epoch by the model, which modifies its parameters depending on training data. The model can discover patterns and connections in the input data.

The observed effect of the number of epochs on the loss and accuracy in our model is shown below:

### 2.2.1 Data

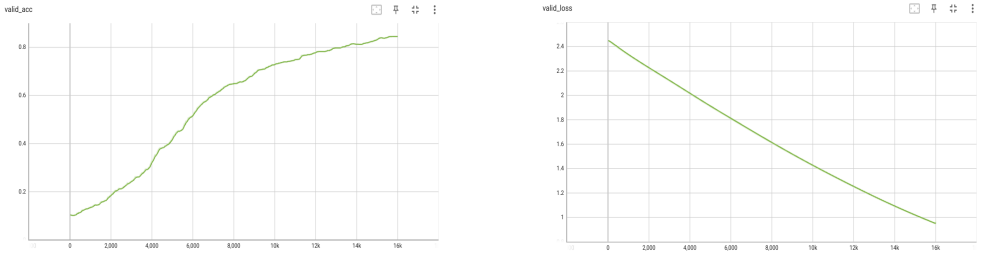
| Number of Epochs | Hidden Layer(s) | Tain Loss | Train Accuracy(%) | Validation Loss | Validation Accuracy(%) |
|---|---|---|---|---|---|
| **1000** | 2 | 0.831 | 87.50 | 0.981 | 84.50 |
| **2000** | 2 | 0.240 | 97.5 | 0.3643 | 90.75 |
| **4000** | 2 | 0.051 | 100.00 | 0.143 | 97.25 |
| **6000** | 2 | 0.016 | 100.00 | 0.105 | 98.00 |

Table 7: Table showing the observations for the model when Learning Rate: $10^{-6}$, seed: 1618 and number of hidden layers: 2

| Number of Epochs | Hidden Layer(s) | Tain Loss | Train Accuracy(%) | Validation Loss | Validation Accuracy(%) |
|---|---|---|---|---|---|
| **100** | 3 | 0.340 | 97.50 | 0.459 | 92.25 |
| **250** | 3 | 0.046 | 100.00 | 0.168 | 96.25 |
| **500** | 3 | 0.006 | 100.00 | 0.153 | 96.75 |
| **1000** | 2 | 0.006 | 100.00 | 0.110 | 97.50 |

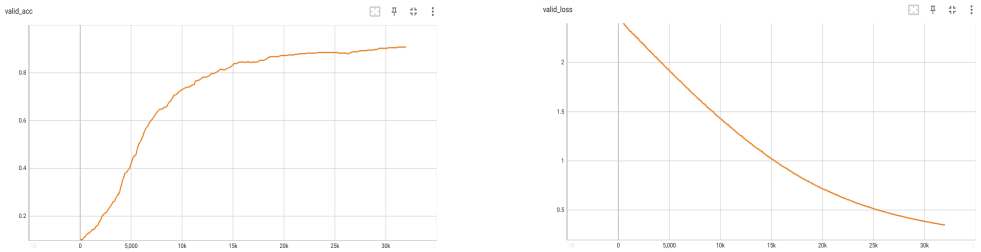
Table 8: Table showing the observations for the model when Learning Rate: $10^{-5}$, seed: 1618 and number of hidden layers: 3

### 2.2.2 Visualization



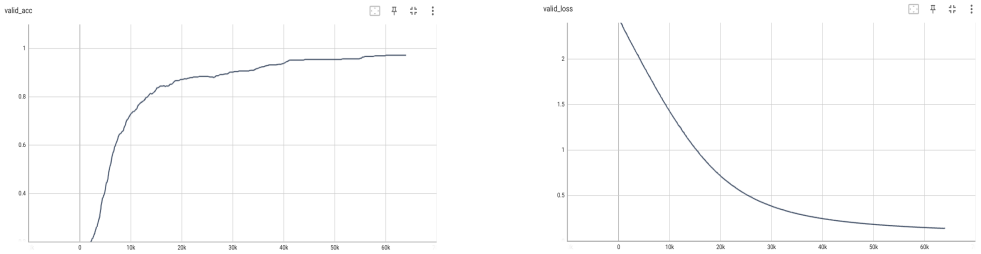

(a) Accuracy Plot (b) Loss Plot

Figure 29: Number of Epochs: 1000



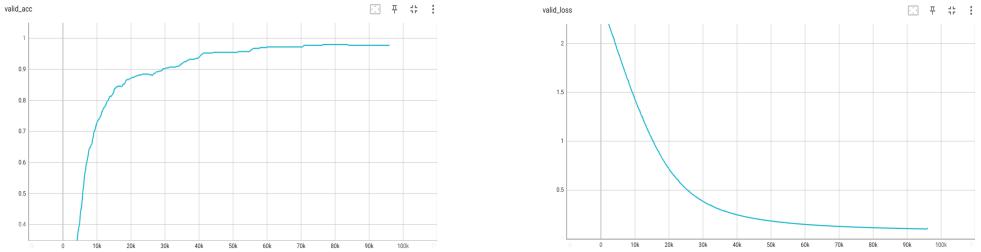

(a) Accuracy Plot (b) Loss Plot

Figure 30: Number of Epochs: 2000




(a) Accuracy Plot (b) Loss Plot

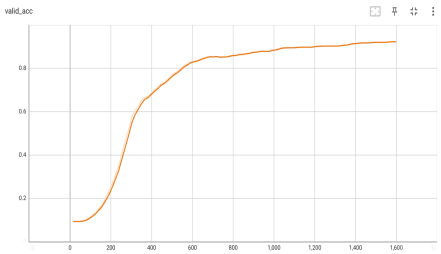
Figure 31: Number of Epochs: 4000



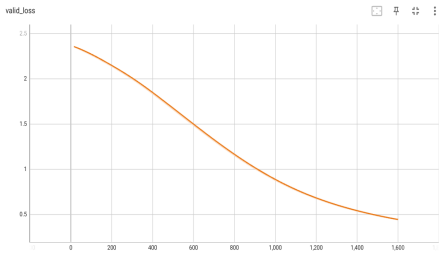

(a) Accuracy Plot (b) Loss Plot

Figure 32: Number of Epochs: 6000

**Accuracy plot:** X-axis: Steps Y-axis: Accuracy **Loss plot:** X-axis: Steps Y-axis: Loss
These plots are for the number of hidden layer: 2.

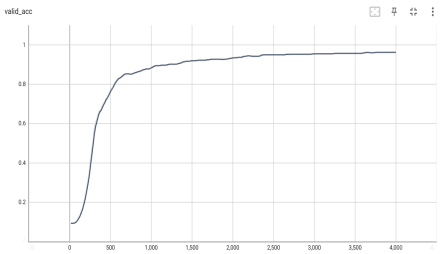



(a) Accuracy Plot

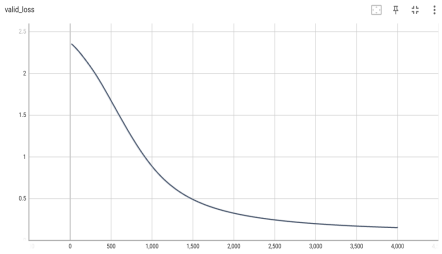



(b) Loss Plot

Figure 33: Number of Epochs: 100

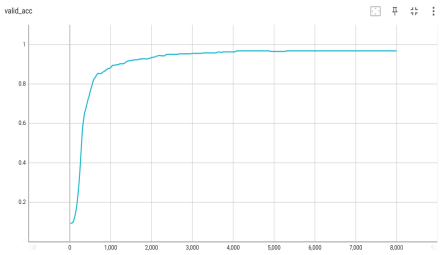



(a) Accuracy Plot

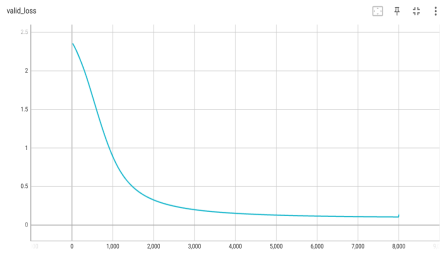



(b) Loss Plot

Figure 34: Number of Epochs: 250

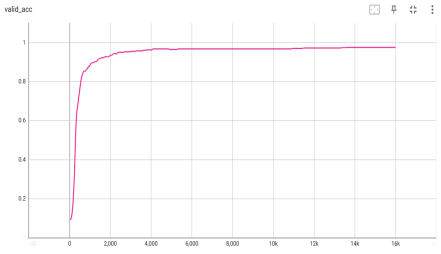



(a) Accuracy Plot

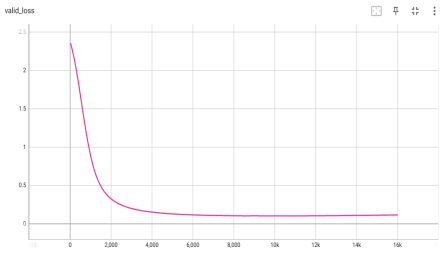



(b) Loss Plot

Figure 35: Number of Epochs: 500




(a) Accuracy Plot



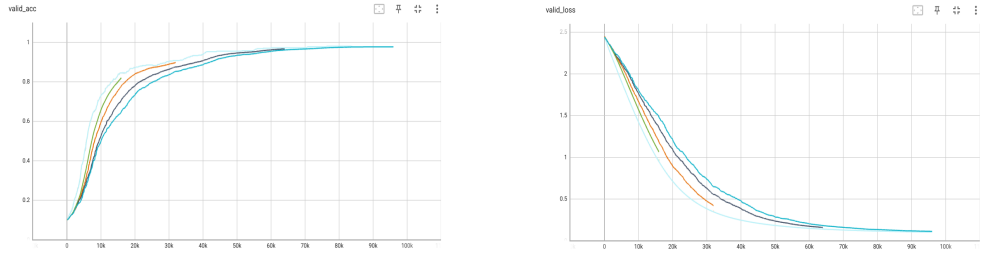

(b) Loss Plot

Figure 36: Number of Epochs: 1000

**Accuracy plot:** X-axis: Steps Y-axis: Accuracy **Loss plot:** X-axis: Steps Y-axis: Loss
These plots are for the number of hidden layer: 3.

### 2.2.3 Observation

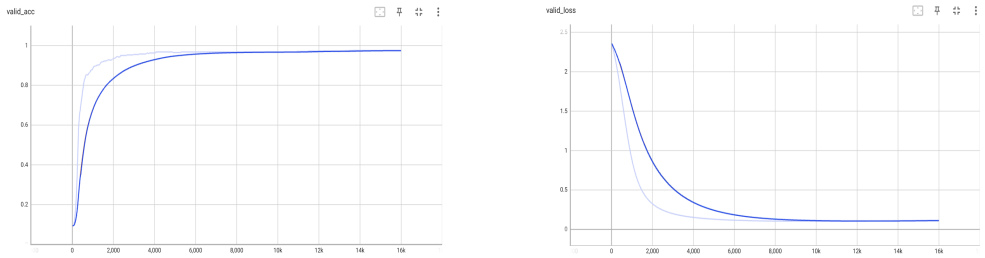
We can observe from the above figures where we considered the learning rate to be constant at $10^{-5}$, seed to be constant at 1618 in our experiment. From [Fig. 29] We can observe that our model has yet to converge and more epochs are needed to get good accuracy, as the number of epochs increases from 1000 to 6000 we can see that the model is converging with more accuracy and less loss [Fig. 37].




(a) Accuracy Plot (b) Loss Plot

Figure 37: Number of hidden layers: 2

Similarly for the number of hidden layers 3 there is no significant improvement in the accuracy but here we are getting the more accurate solution in a much smaller number of epochs as shown in the above figures [Fig. 33 - Fig. 36]. From the figure [Fig. 38], It is clearly seen that all the loss and accuracy plots overlap with each other. As the number of epochs increases the convergence is happening in the same path but for the smaller number of epochs, the convergence is stopped in earlier positions.




(a) Accuracy Plot (b) Loss Plot

Figure 38: Number of hidden layers: 3

In summary, increasing the number of epochs allows the model more time to converge toward an optimal parameter configuration that minimizes the selected loss function, leading to improved accuracy. However, excessive epochs can lead to overfitting, diminishing the accuracy of the validation set.

# 3 Best Epoch

## 3.1 Simple Dataset

The best accuracy for the simple data set is observed when the **number of epochs**: 500, **learning rate**: $10^{-3}$, **seed**: 1618 (default) and **hidden layers**: 1.

## 3.2 Digits Dataset

The best accuracy for the digits data set is observed when the **number of epochs**: 6000, **learning rate**: $10^{-6}$, **seed**: 1618 (default) and **hidden layers**: 2.

# 4 Data Preprocessing

Here in our model, we are implementing data processing in the following way:

- **Step 1:** Removing the outliers present in the data by replacing them with the mean of the total input data.
- **Step 2:** Standardize the input data by the following method:

$$\lambda = \frac{X - \hat{X}}{\sigma} \tag{3}$$

where $\lambda$ is the standardize input value, $X$ is the input data (outlier), $\hat{X}$ is the mean of the input values and $\sigma$ is the standard deviation of the input data.

# 5 Overfitting Prevention

We have used weight decay as L2 regularization to prevent overfitting of the model. It is a regularization technique that keeps the weight parameter small and avoids an exploding gradient. We have used $10^{-5}$ as the value of weight decay.