Laboratory 5

Variant 3, Group 13

By Saumya Shah and Anas Tagui

Task

The task was to use a multilayer perceptron to classify the Fashion MNIST dataset, and explore the impact of using different learning rates, batch sizes, hidden layers, widths, and activation functions. The loss function was fixed to cross-entropy. The optimiser was fixed to Adam. Three values for each of these hyperparameters were tested, with a fixed number of epochs (10), and the accuracy was measured.

Results

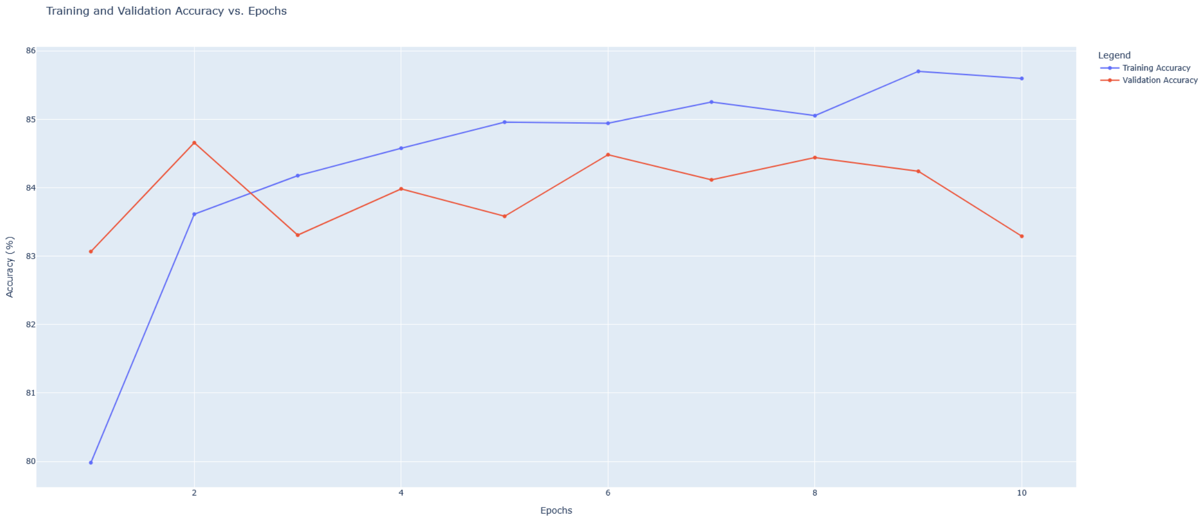
Learning Rates

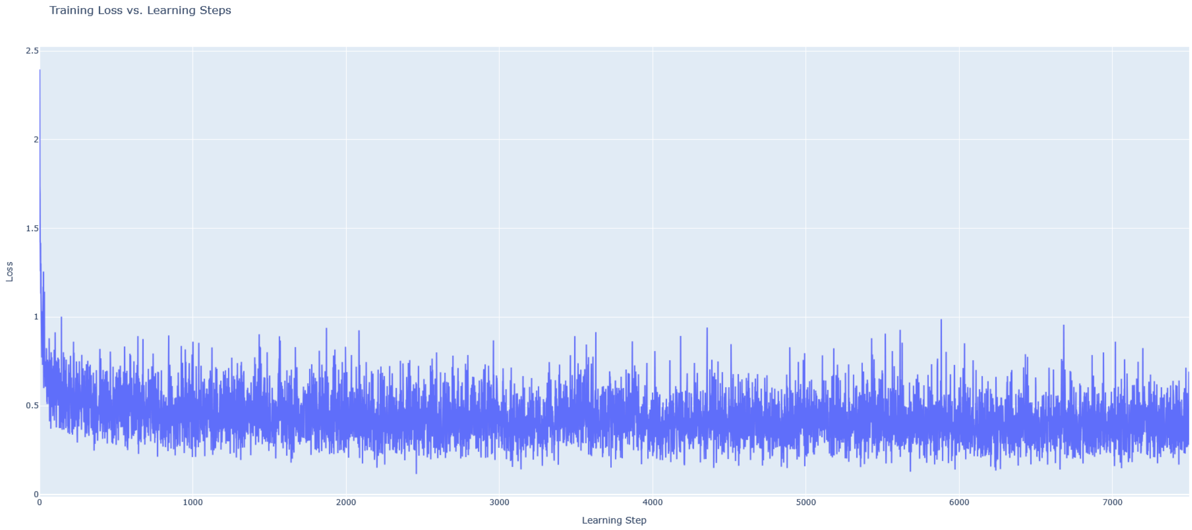
Batch Size: 64, **Number of Hidden Layers**: 1, **Hidden Layer Width**: 32, **Activation Function:** ReLU

Learning Rate	**Test Accuracy (%)**
0.01	81.89
0.1	10.01
1	9.01

Figures

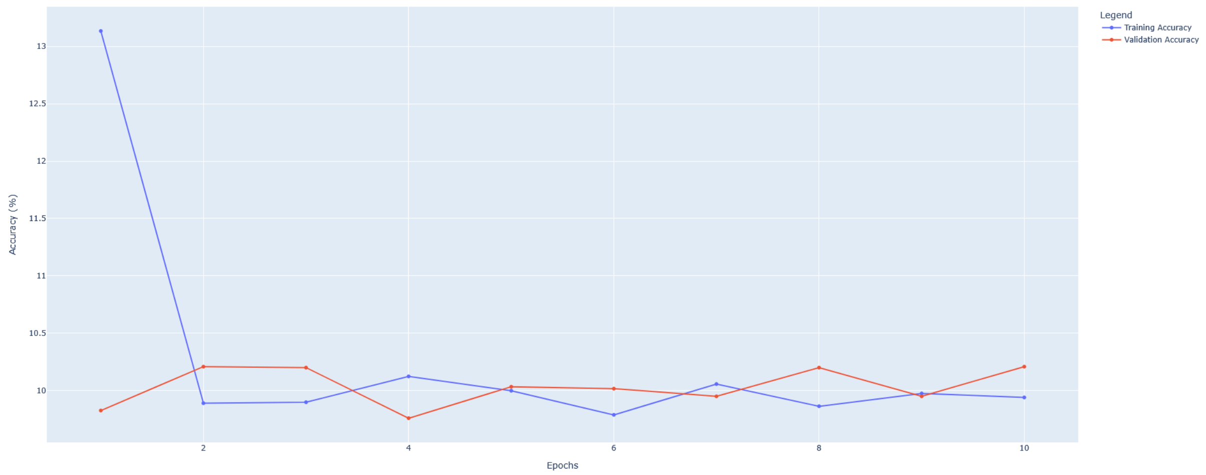
LR = 0.01


Training and Validation Accuracy vs. Epochs
Legend
Training Accuracy
Validation Accuracy
Accuracy (%)
Epochs
86
85
84
83
82
81
80
2
4
6
8
10

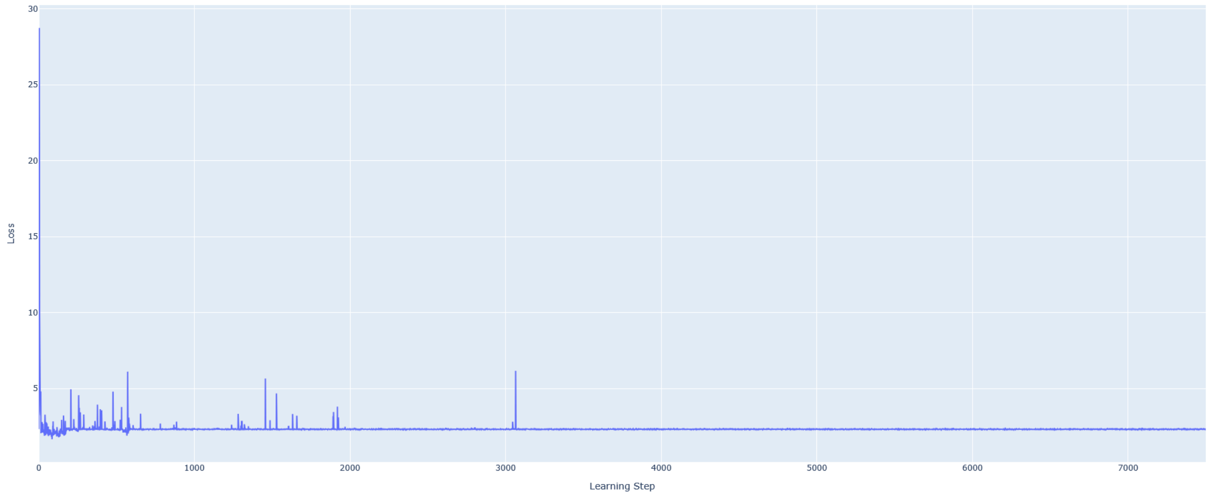

Training Loss vs. Learning Steps
Loss
Learning Step
2.5
2
1.5
1
0.5
0
1000
2000
3000
4000
5000
6000
7000

<u>LR = 0.1</u>

Training and Validation Accuracy vs. Epochs

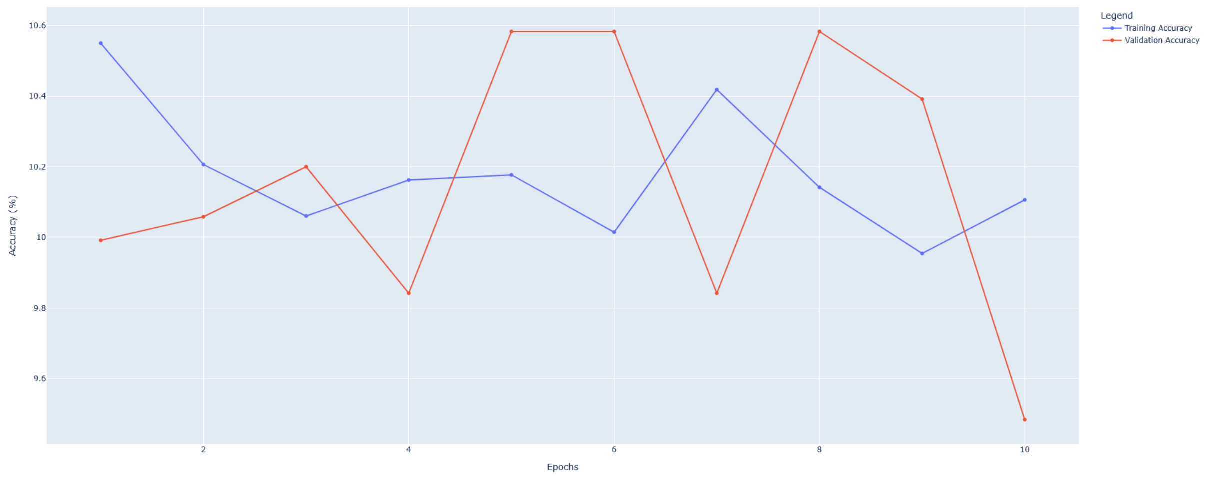


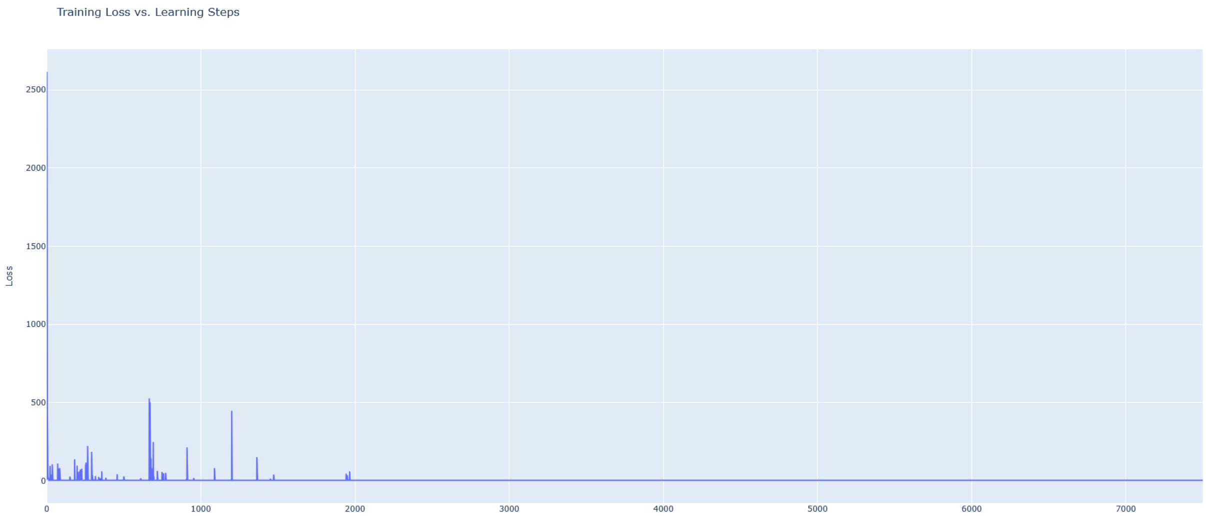
Training Loss vs. Learning Steps



LR = 1

Training and Validation Accuracy vs. Epochs





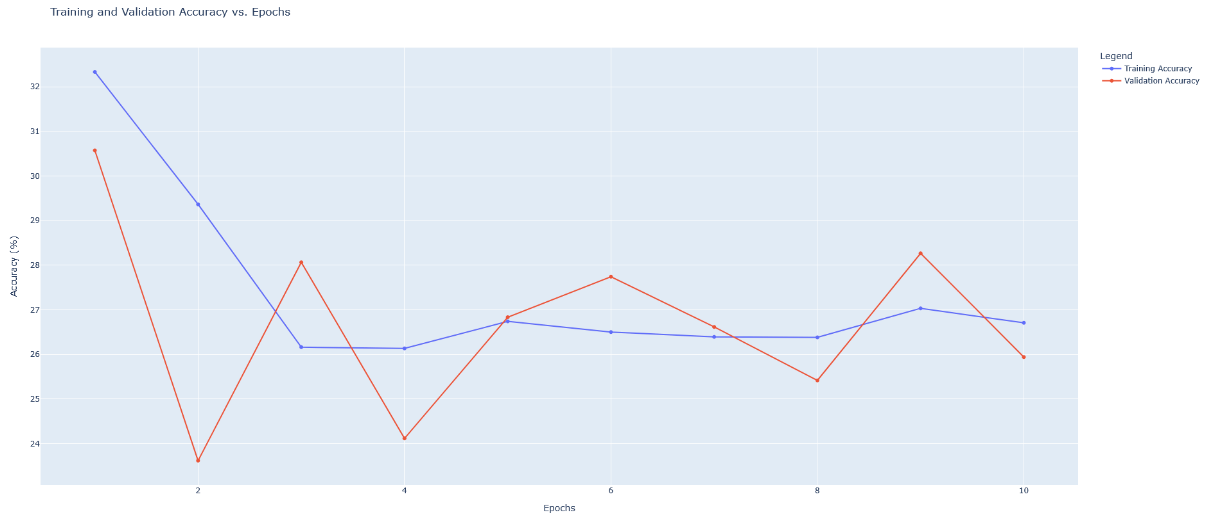
Batch Size

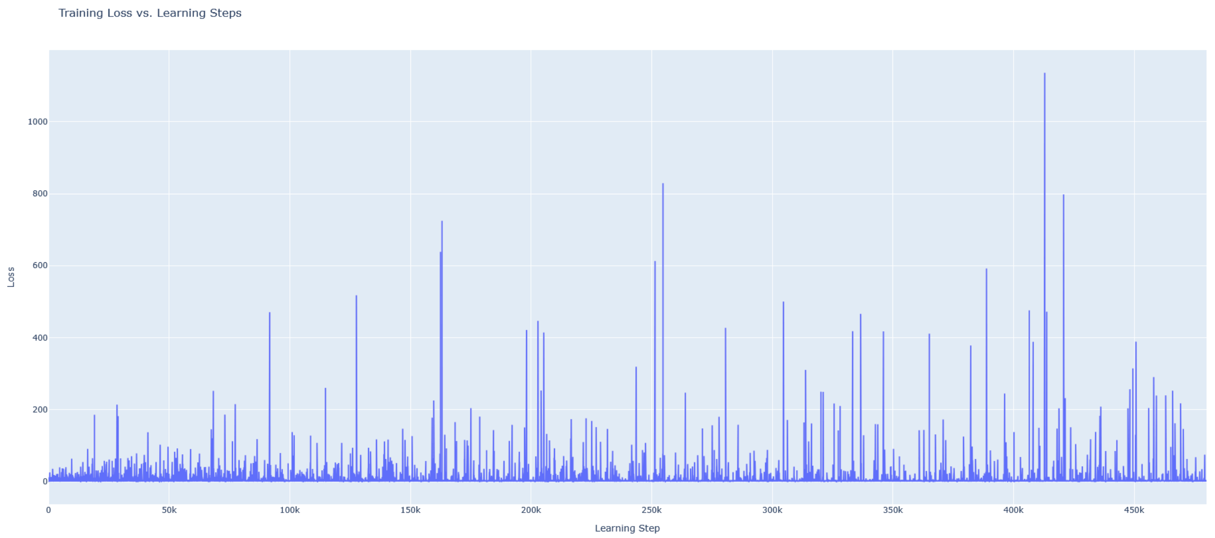
Learning Rate: 0.01, **Number of Hidden Layers**: 1, **Hidden Layer Width**: 32, **Activation Function**: ReLU

Batch Size	Test Accuracy (%)
1	25.44
64	81.89
256	85.87

Figures

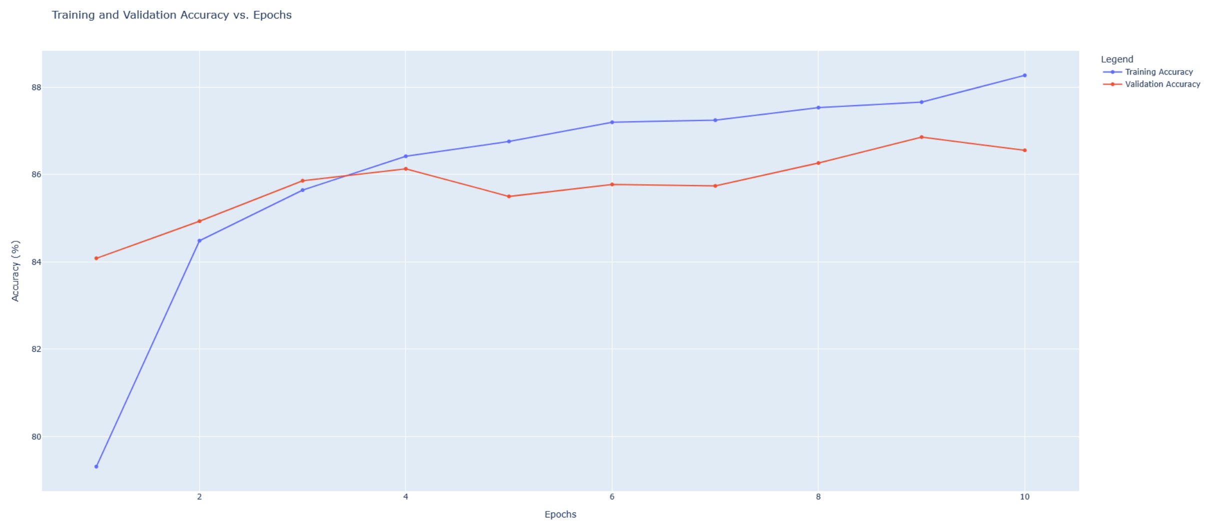
Batch Size = 1

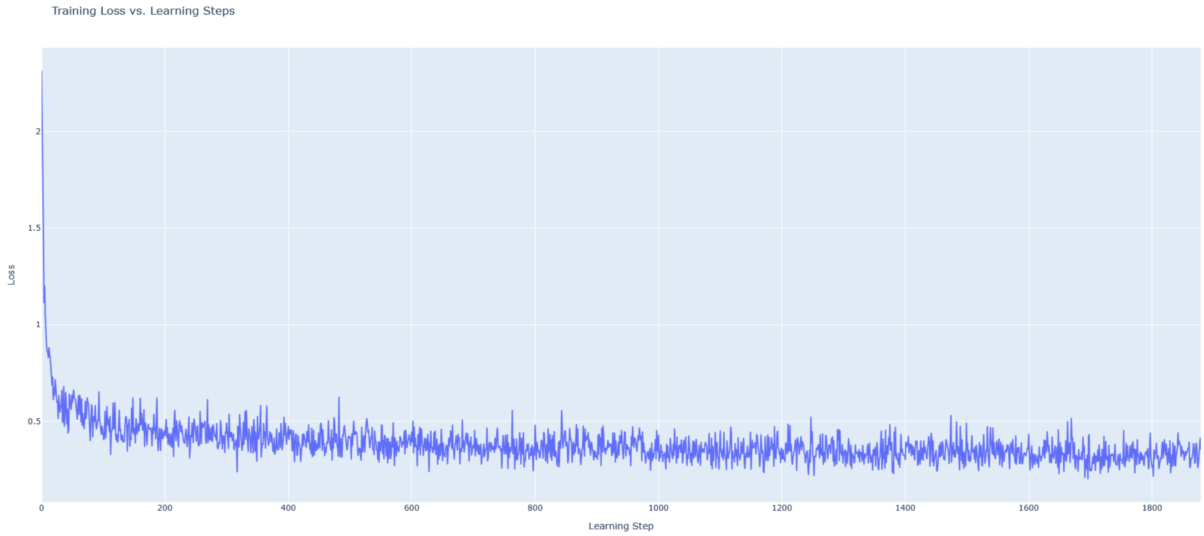




<u>Batch Size = 64</u>: see figures for LR = 0.01 above

<u>Batch Size = 256</u>





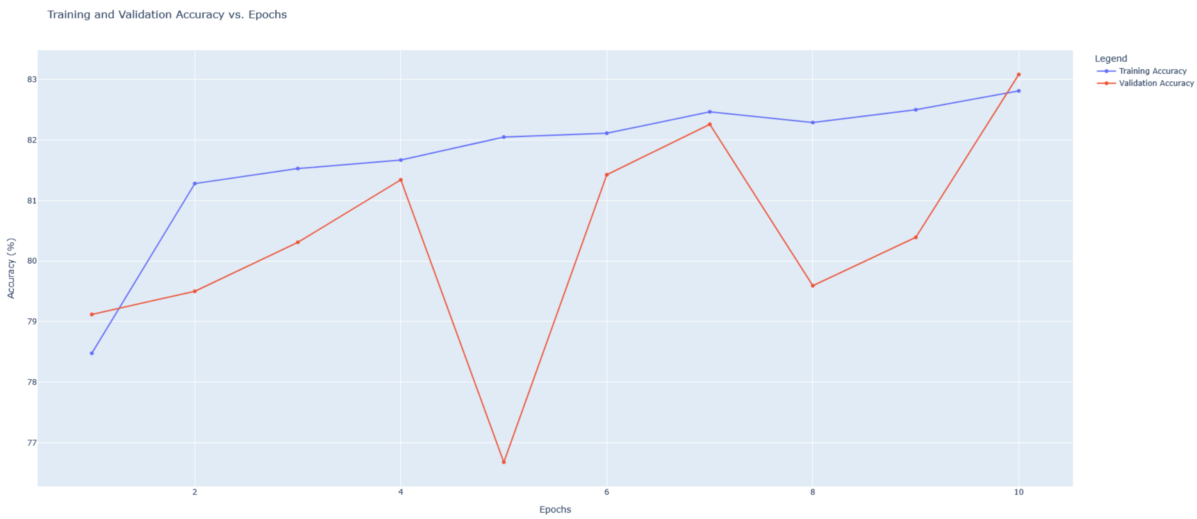
Number of Hidden Layers

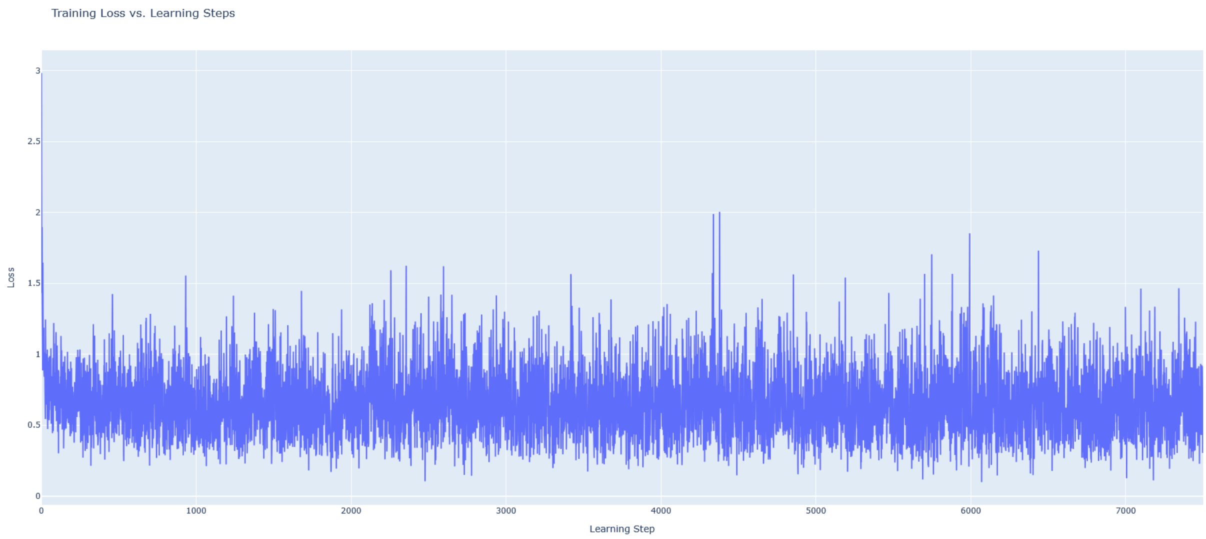
Learning Rate: 0.01, **Batch Size:** 64, **Hidden Layer Width**: 32, **Activation Function**: ReLU

Number of Hidden Layers	Test Accuracy (%)
0	81.62
1	81.89
2	83.71

Figures

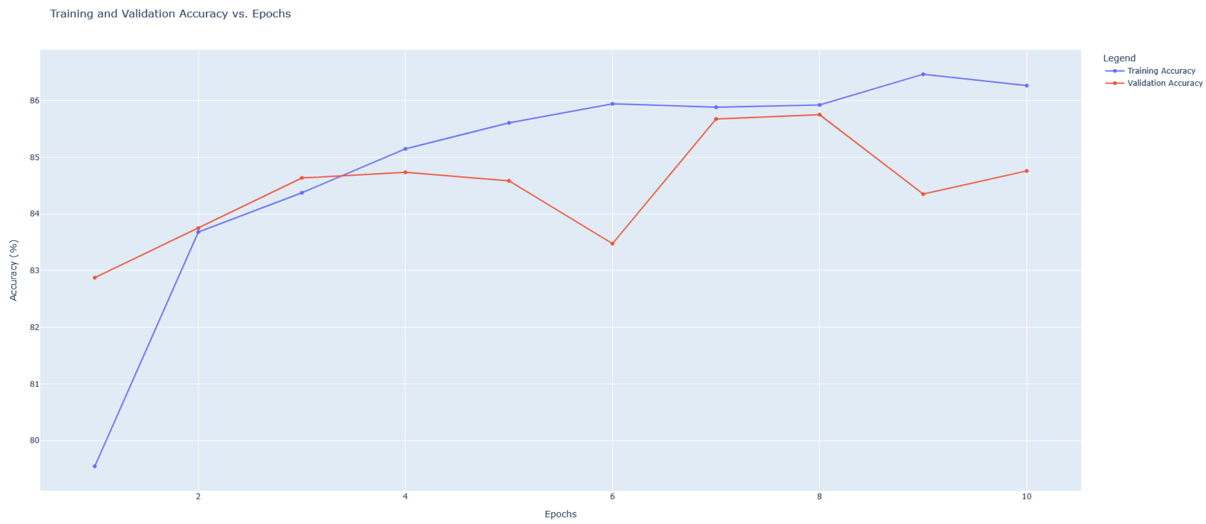
<u>Num of Hidden Layers = 0</u>

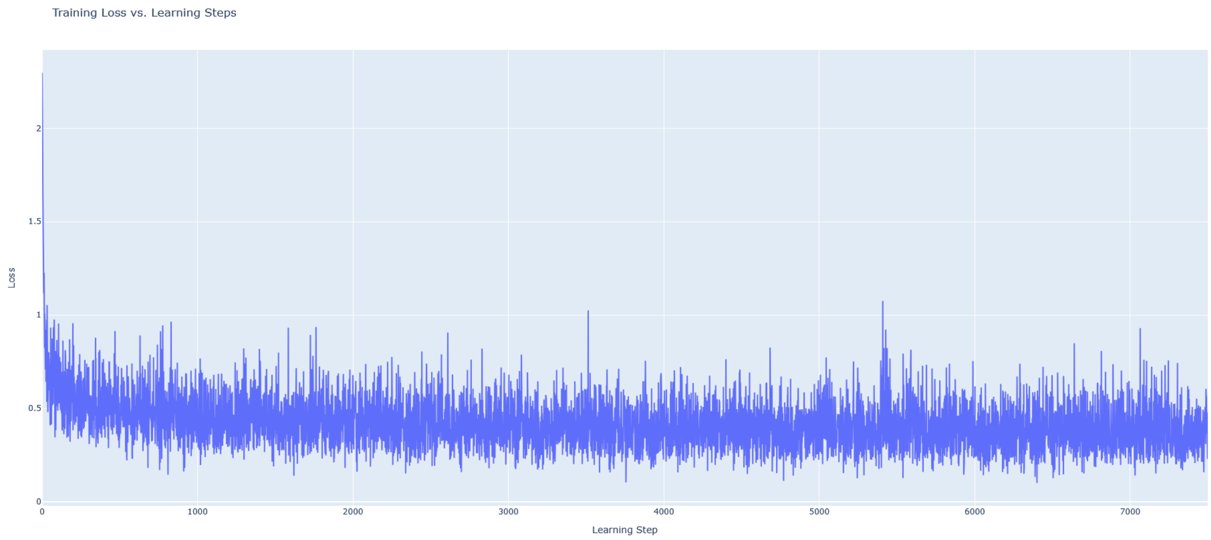




Num of Hidden Layers = 1: See figures for LR = 0.01.

Num of Hidden Layers = 2





Width of Hidden Layers

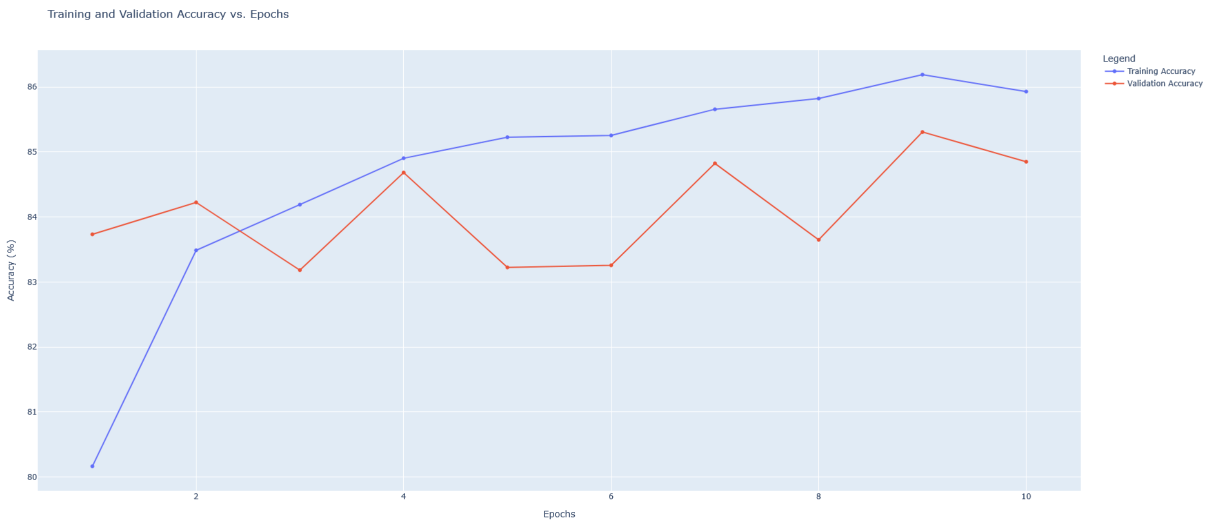
Learning Rate: 0.01, **Batch Size:** 64, **Number of Hidden Layers**: 1, **Activation Function**: ReLU

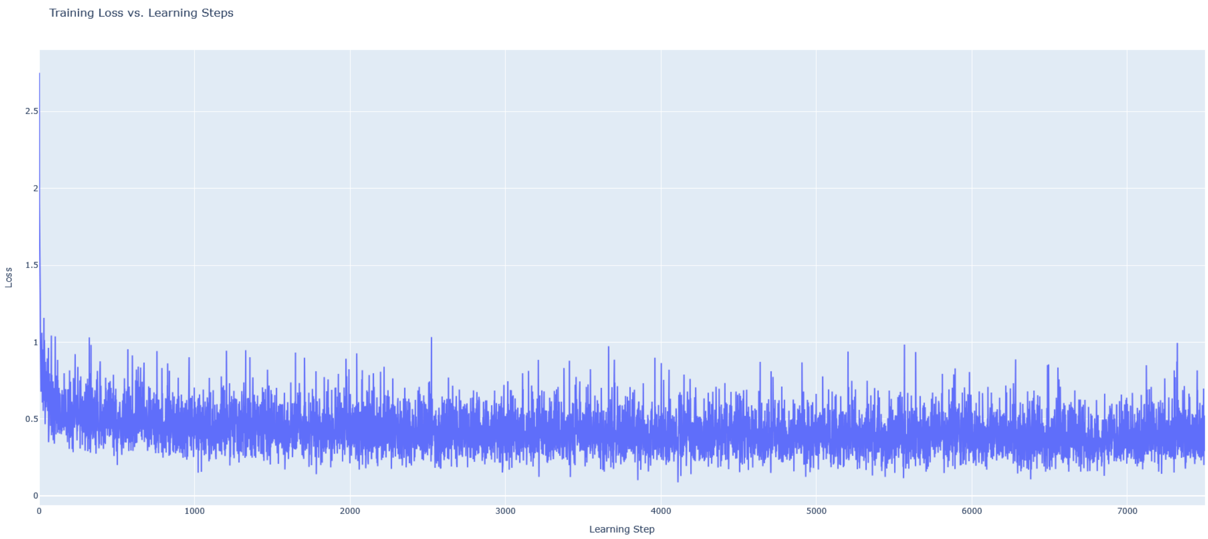
Hidden Layer Width	**Test Accuracy (%)**
32	81.89
64	83.92
128	84.13

Figures

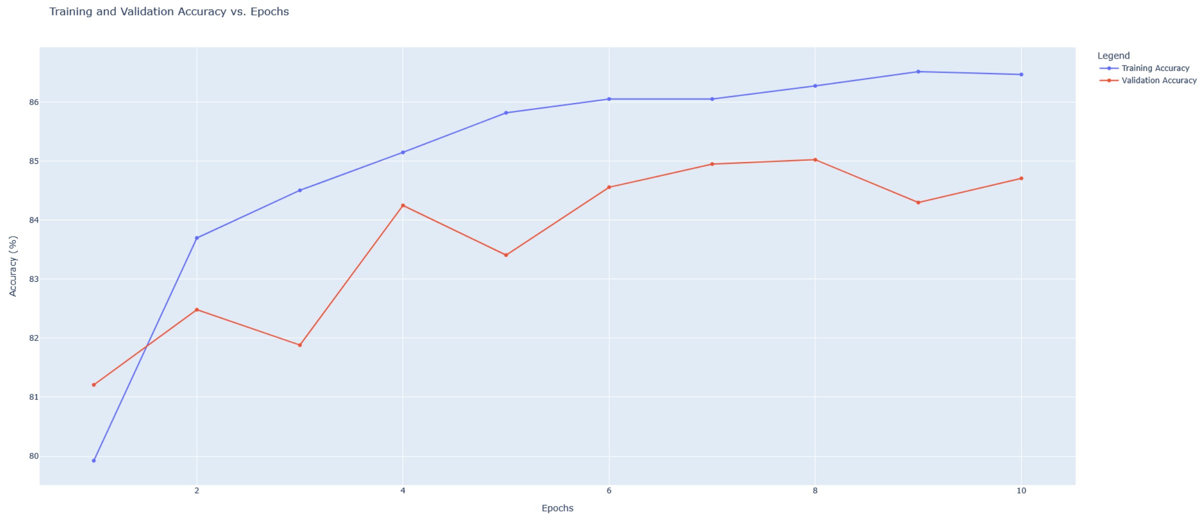
Hidden Layer Width = 32: See figures for LR = 0.01.

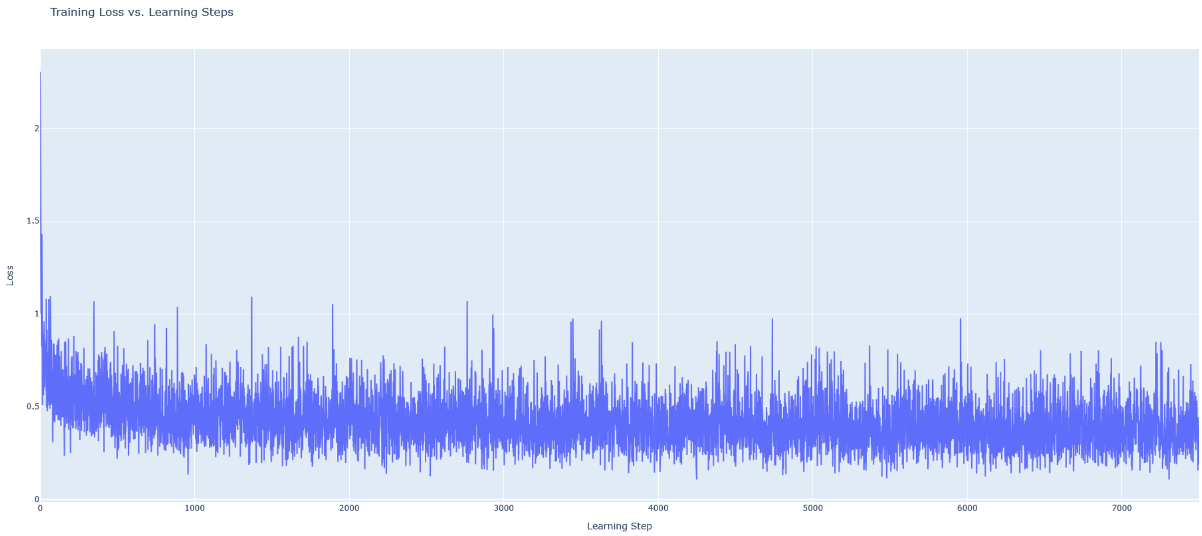
Hidden Layer Width = 64:





Hidden Layer Width = 128:





Activation Function

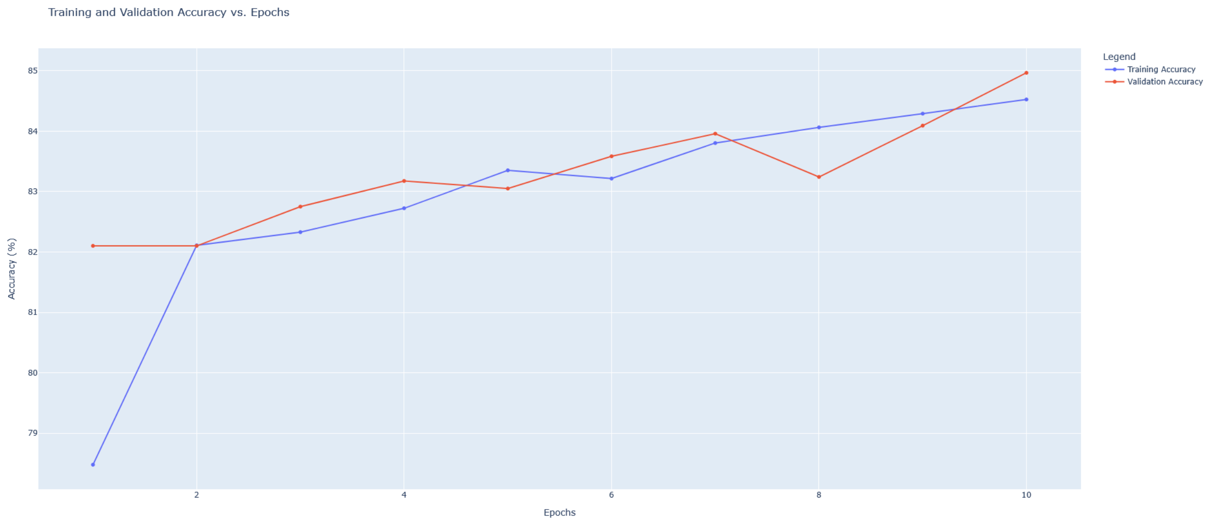
Learning Rate: 0.01, **Batch Size:** 64, **Number of Hidden Layers**: 1, **Hidden Layer Width:** 32

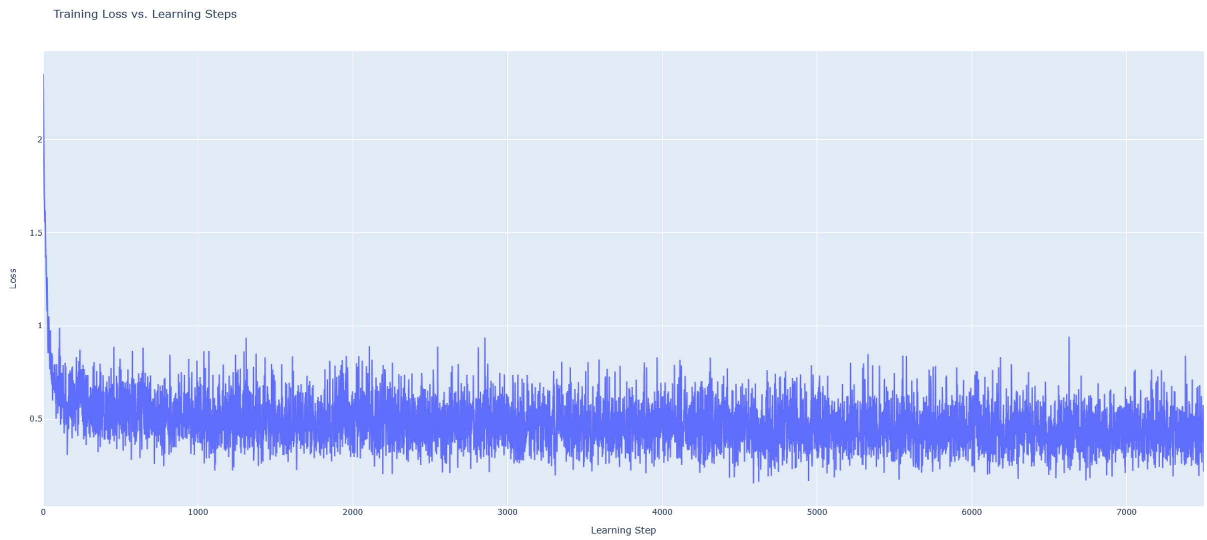
Activation Function	**Test Accuracy (%)**
ReLU	81.89
Sigmoid	83.65
Tanh	81.28

Figures

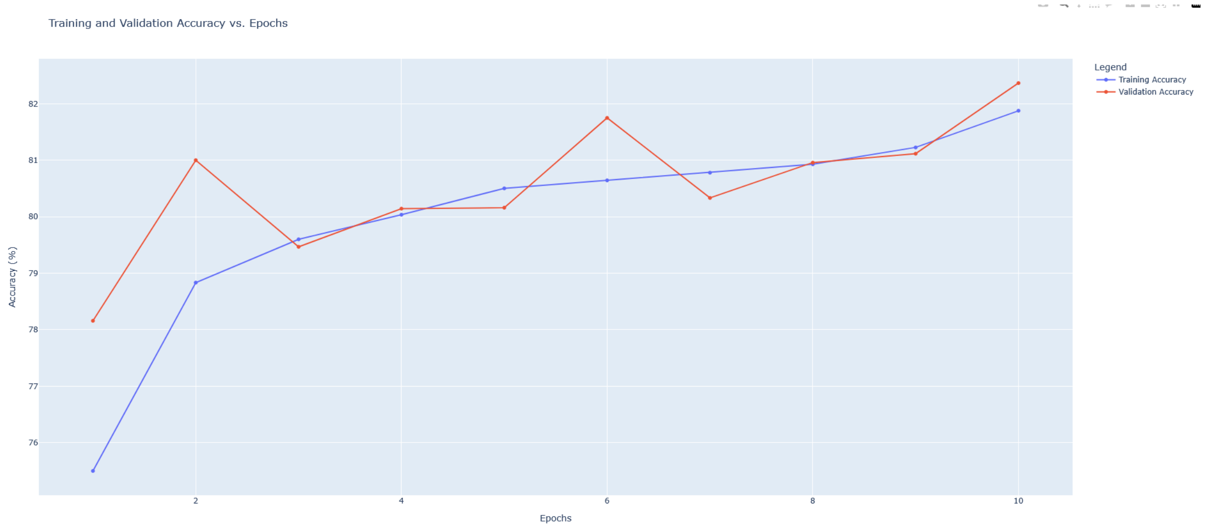
<u>Activation Function = ReLU:</u> See figures for LR = 0.01.

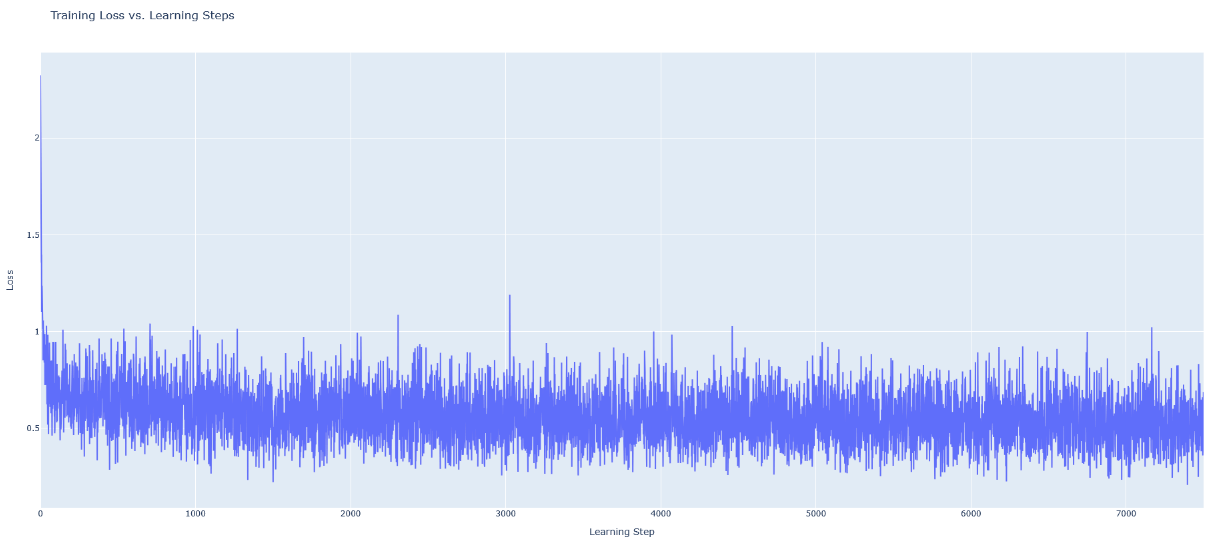
<u>Activation Function = Sigmoid:</u>





Activation Function = Tanh





Evaluation

Learning Rate: We notice that a higher learning rate not only leads to more chaotic loss curves, it also leads to lower accuracy. This is because a high learning rate causes the optimisation algorithm to take excessively large steps in the parameter space. Instead of smoothly converging towards the minimum of the loss function, these large steps can cause the optimisation to overshoot the minimum, bounce around erratically, or even diverge.

Batch Size: As we see, a batch size of 1 (equivalent to stochastic gradient descent) gives us extremely chaotic loss values with a low testing accuracy. This shows that the gradient descent is unstable and erratic. Higher batch sizes give us smoother descent with greater accuracy and a consistent decline in loss. This is because the averaging process in higher batch sizes reduces variance and noise in the gradient estimate, leading to smoother and more reliable gradient descent.

Number of Hidden Layers: We observe slight increases in test accuracy as the number of hidden layers increases. This may be because more hidden layers may allow the model to learn more complex non-linear patterns by composing features from the previous layers.

Hidden Layer Width: We observe that increasing the width of hidden layers slightly improves test accuracy. This is because wider layers increase the model's capacity, allowing it to learn more features and patterns in the data.

Activation Functions: We see that Sigmoid performed the best (83.65% accuracy), followed by ReLU (81.89%), and then Tanh (81.28%) in this configuration. However, the difference in performance is not very pronounced. Different activation functions introduce non-linearities in distinct ways, allowing them to learn complex patterns, leading to variations in performance. In this specific shallow network setup, Sigmoid proved most effective.