23M1074_23M1163

REPORT

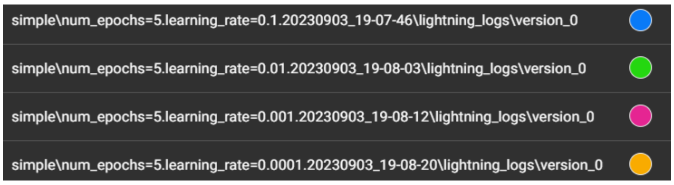
Analysis of FFNN Model

Classification on simple(4-class) dataset:

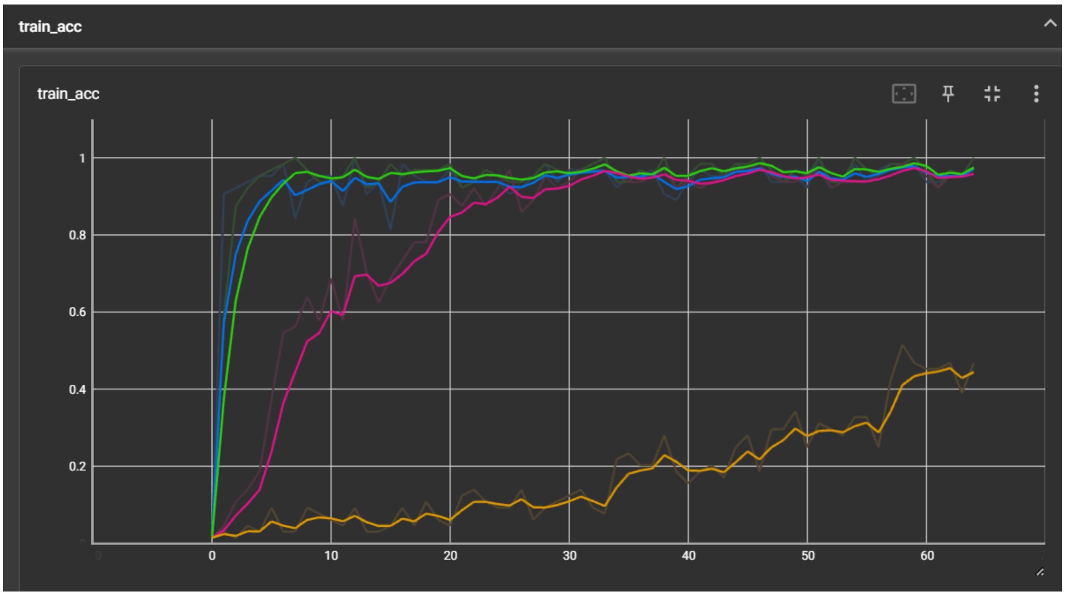
To analyse the effect of epoch, learning rate and seed.

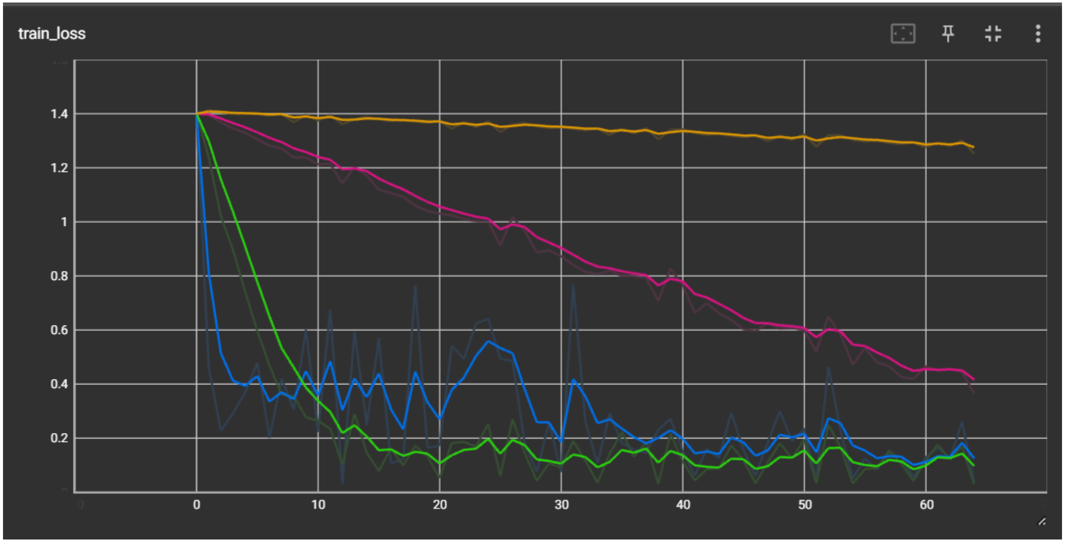
1. **Learning rate** -- Effect of the learning rate on the loss and accuracy.

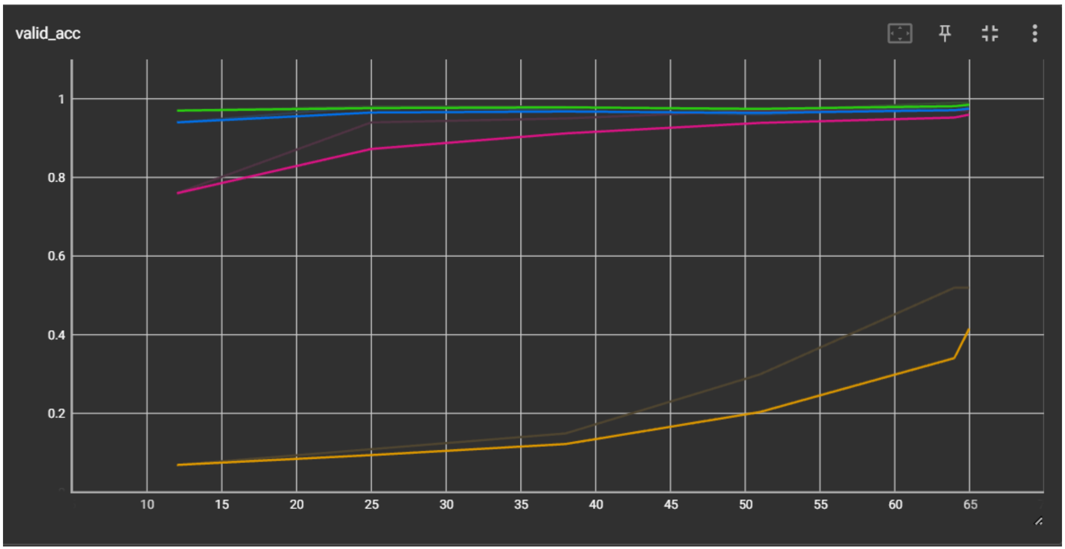
   For Simple dataset after completion of model selection we had conducted analysis to find the best hyperparameters. For best learning rate, we fixed the number of epochs to be 5, seed 1000 and analyse the loss and accuracy graphs for train and valid data with different learning rates. From the result in we got following graphs.

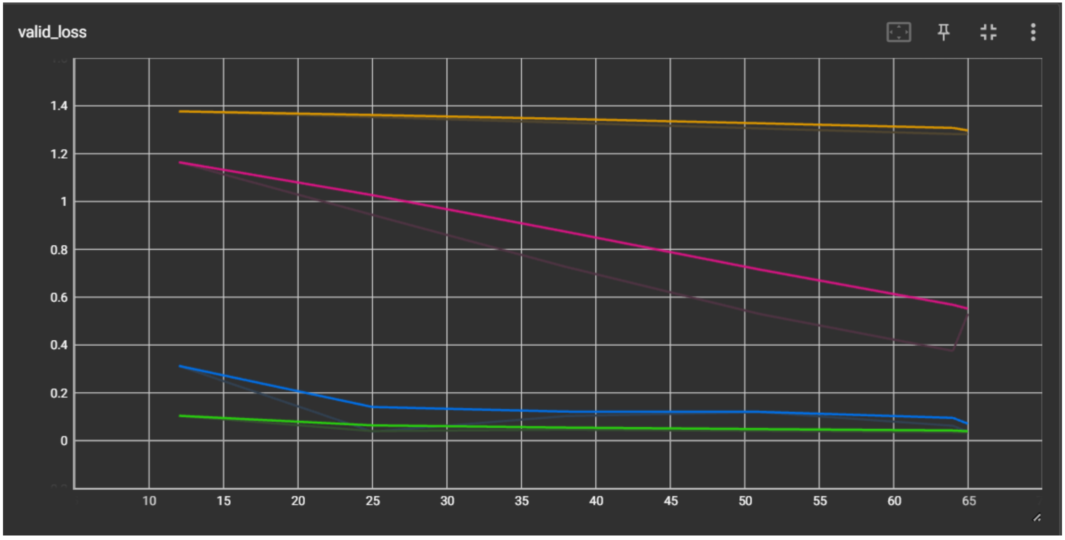


simple\num_epochs=5.learning_rate=0.1.20230903_19-07-46\lightning_logs\version_0	🔵
simple\num_epochs=5.learning_rate=0.01.20230903_19-08-03\lightning_logs\version_0	🟢
simple\num_epochs=5.learning_rate=0.001.20230903_19-08-12\lightning_logs\version_0	●
simple\num_epochs=5.learning_rate=0.0001.20230903_19-08-20\lightning_logs\version_0	🟠







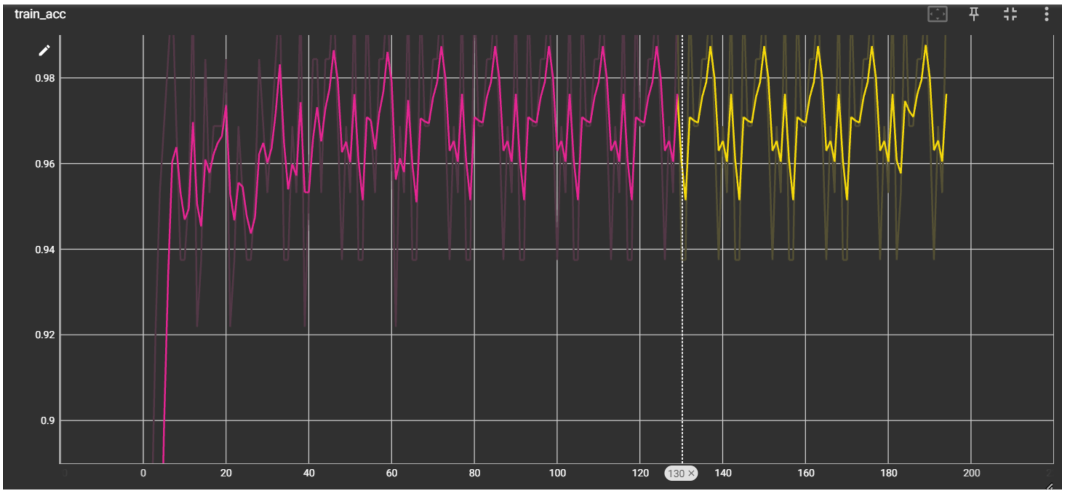


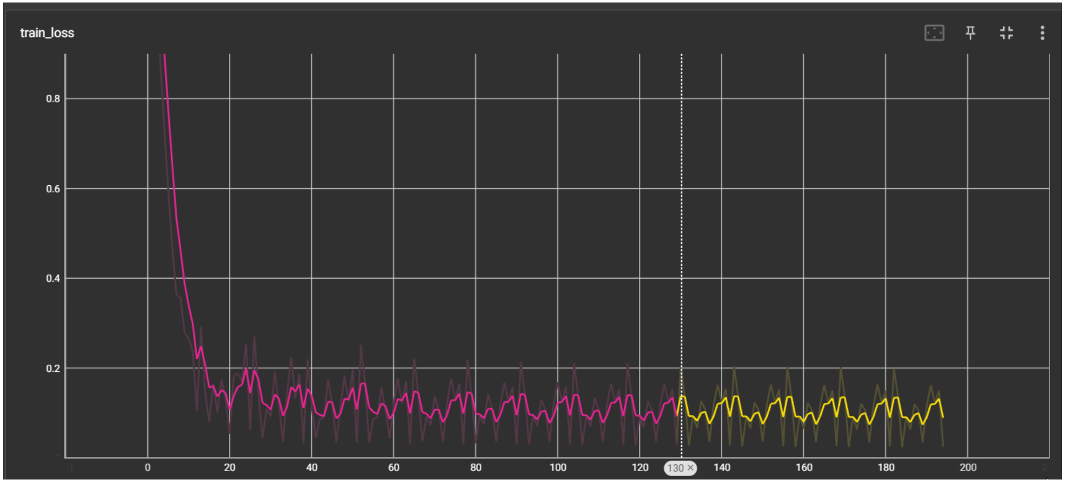
From all the 4 graphs we can say that learning rate 0.01(which Is in green) was giving the less loss and high accuracy in case of both training and valid datasets when compared to learning rates – 0.1, 0.001, 0.0001. Firstly, when learning rate was 0.1 it gave acceptable loss and accuracy as we decrease it to 0.01 then we can see slight variation in decreasing loss and increasing accuracy. Further if we are decreasing the learning rate by 0.001,0.0001 then we observed that as the step size was becoming small for 10 epoch we can see increase in loss and reduction in accuracy. Hence, we choose learning rate 0.01 as parameter.

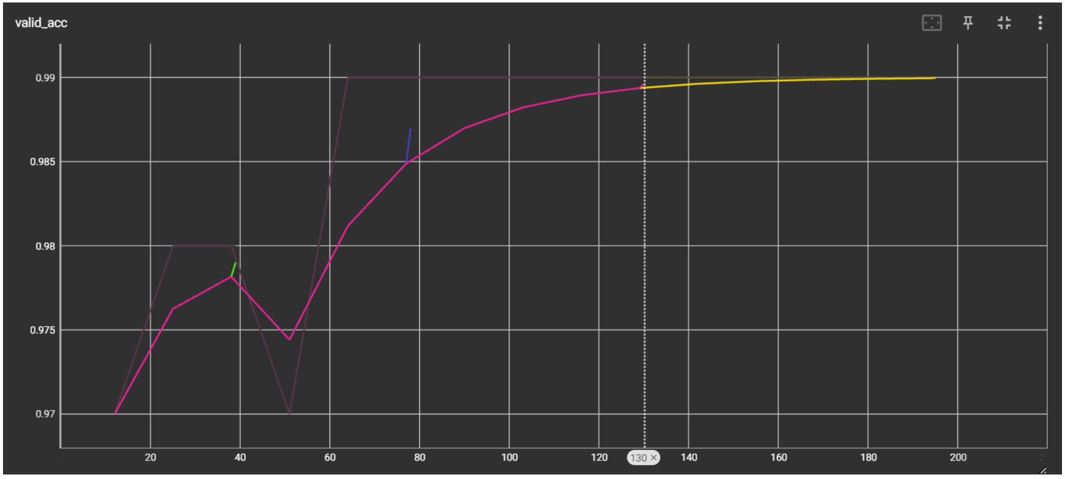
2. **Best epoch** -- Effect of the number of epochs on the loss and accuracy.

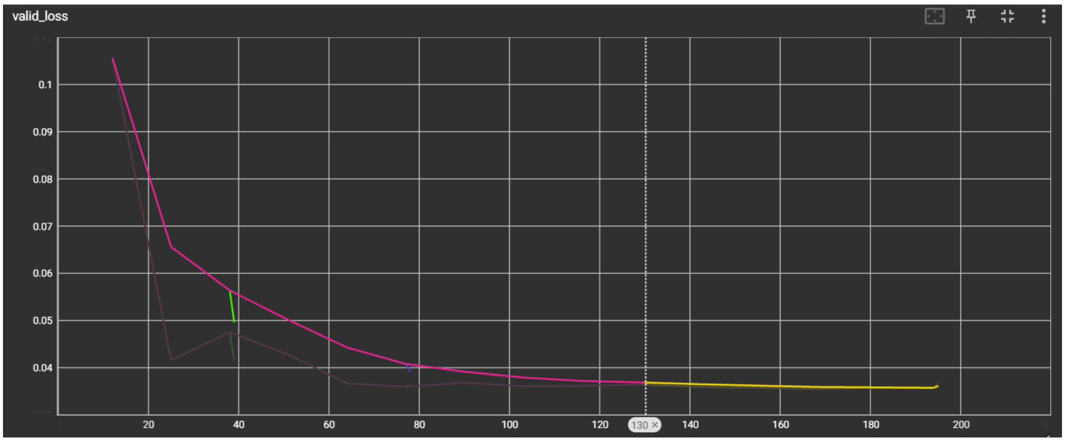
In this part we kept the learning rate to 0.01, seed to 1000 and changed epoch values and analysed the loss and accuracy graphs with different epochs i.e., 3,6,10 and 15.By observation we got below graphs.

23M1074_23M1163





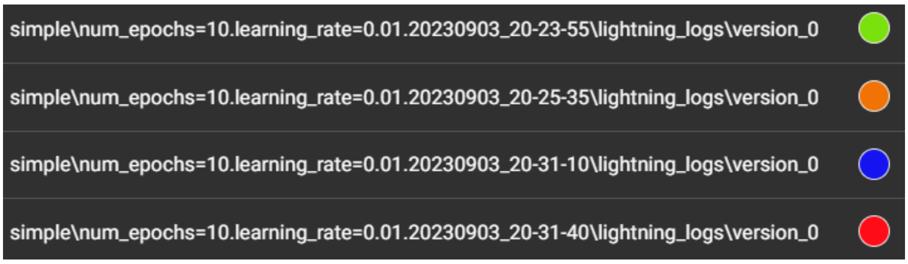




From the graphs we can say that loss was reducing with respect to epochs and also accuracy was increasing with epoch. But after certain epoch values accuracy and loss remain same in valid data set. So based on that observation we took the best low epoch count that gives best accuracy and loss. Even we can observe that there is no much variation after that epoch.

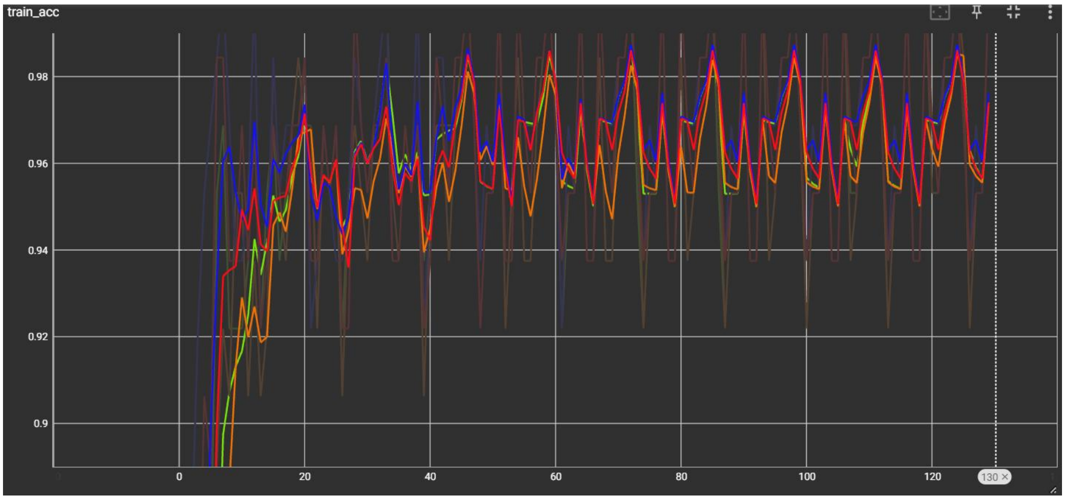
3. **Best Seed**

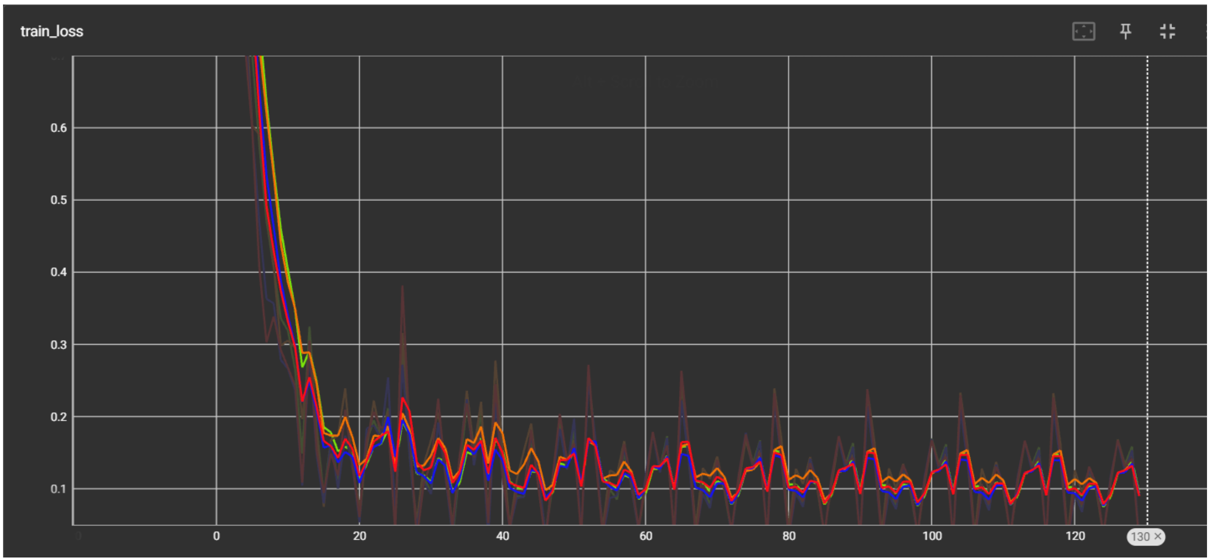
In this part we kept the learning rate to 0.01, epoch to 10 and changed seed values and analysed the loss and accuracy graphs with different seed i.e., 100, 400,700 and 1500.By observation we got below graphs.

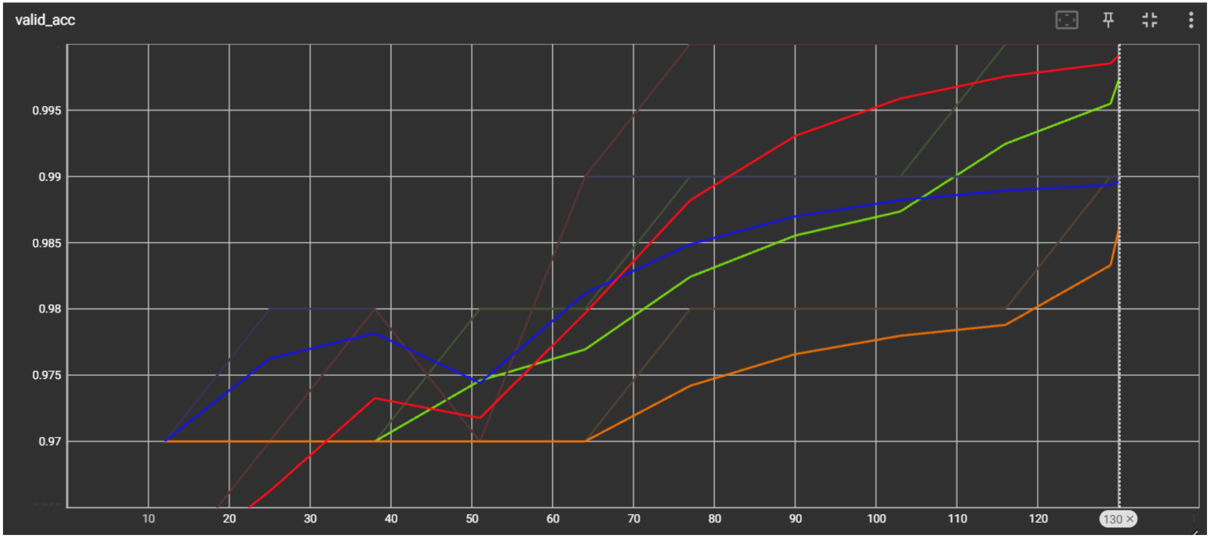


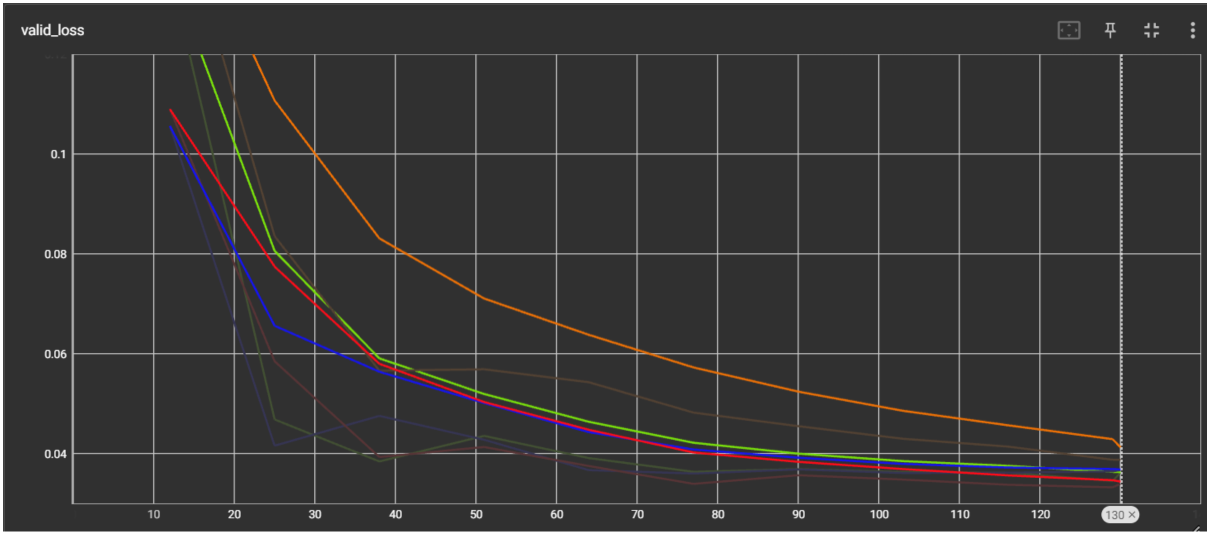
. . .

simple\num_epochs=10.learning_rate=0.01.20230903_20-23-55\lightning_logs\version_0	🟢	. 400
simple\num_epochs=10.learning_rate=0.01.20230903_20-25-35\lightning_logs\version_0	🟠	100
simple\num_epochs=10.learning_rate=0.01.20230903_20-31-10\lightning_logs\version_0	🔵	. 700
simple\num_epochs=10.learning_rate=0.01.20230903_20-31-40\lightning_logs\version_0	🔴	. 1500









Seed number 1500 was giving the better accuracy and loss in valid and train data. Hence, we choose 1500. Final Hyperparameters were epoch – 10, learning rate – 0.01 and seed – 1500

In order to prevent overfitting:

In this question when we are increasing the no of epochs or having too much or too low learning rate, we incurring larger Val loss and training loss, the same is same for If we increase the no of layers or the no of neurons in a layer and by carefully modifying these things we reached at our optimal settings (according to us).

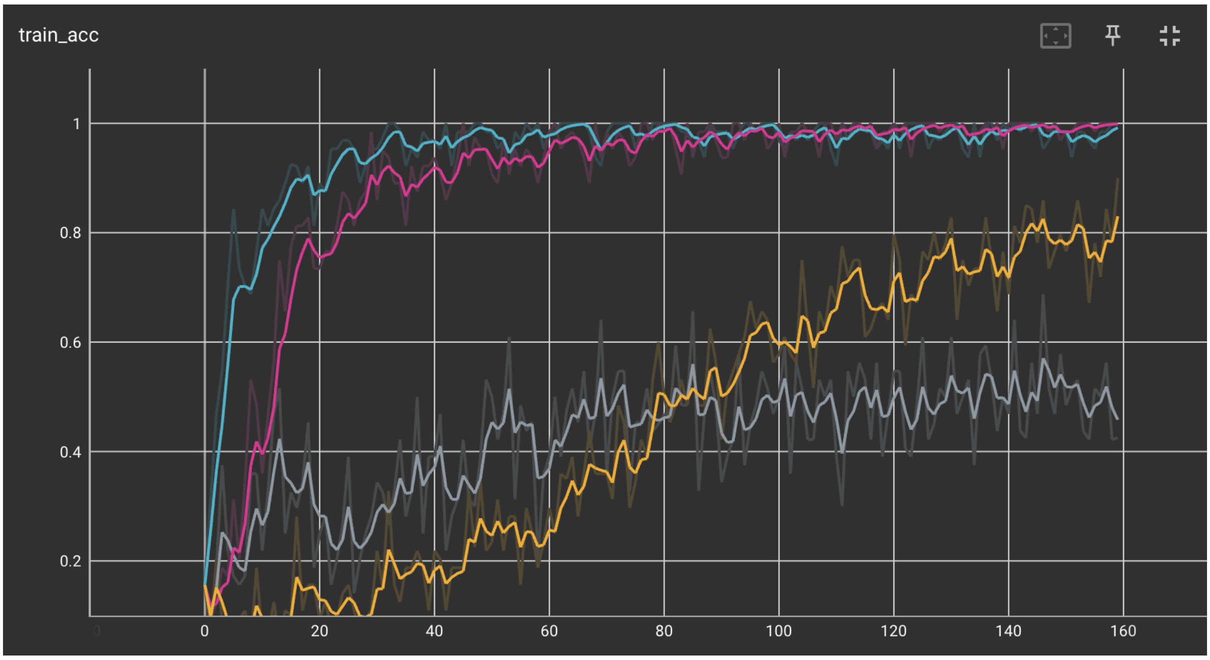
Classification on digits dataset:

1. First with fixed Epoch and Seed for best **Learning Rate**:

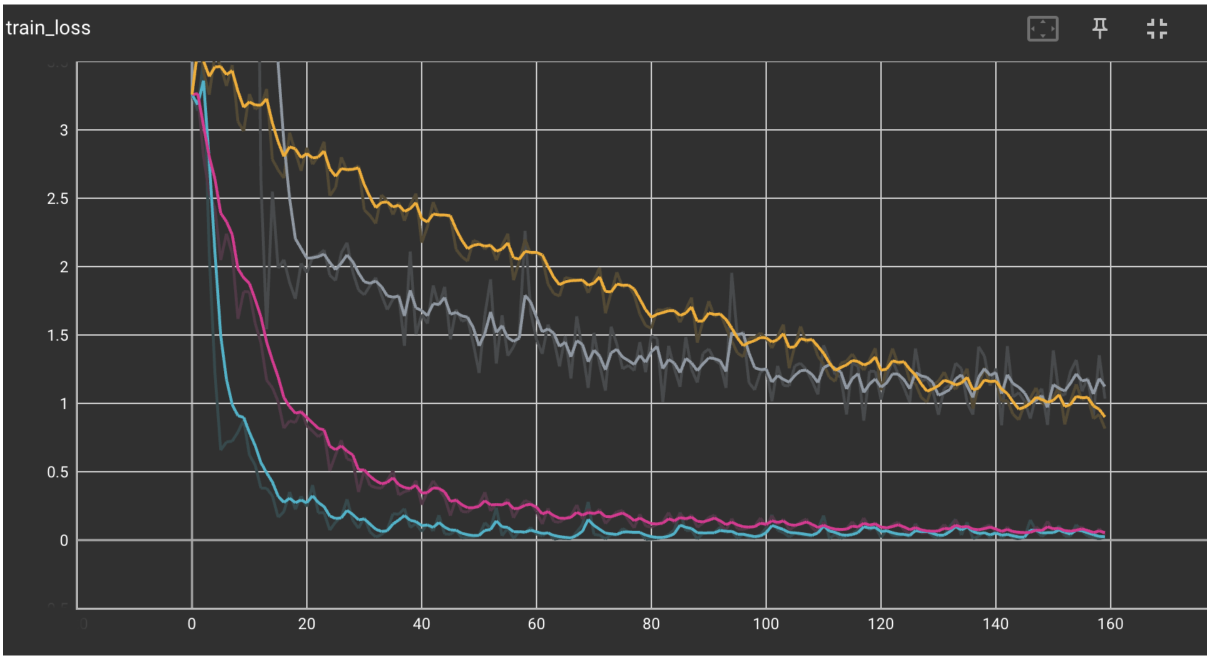
epoch(default) = 10

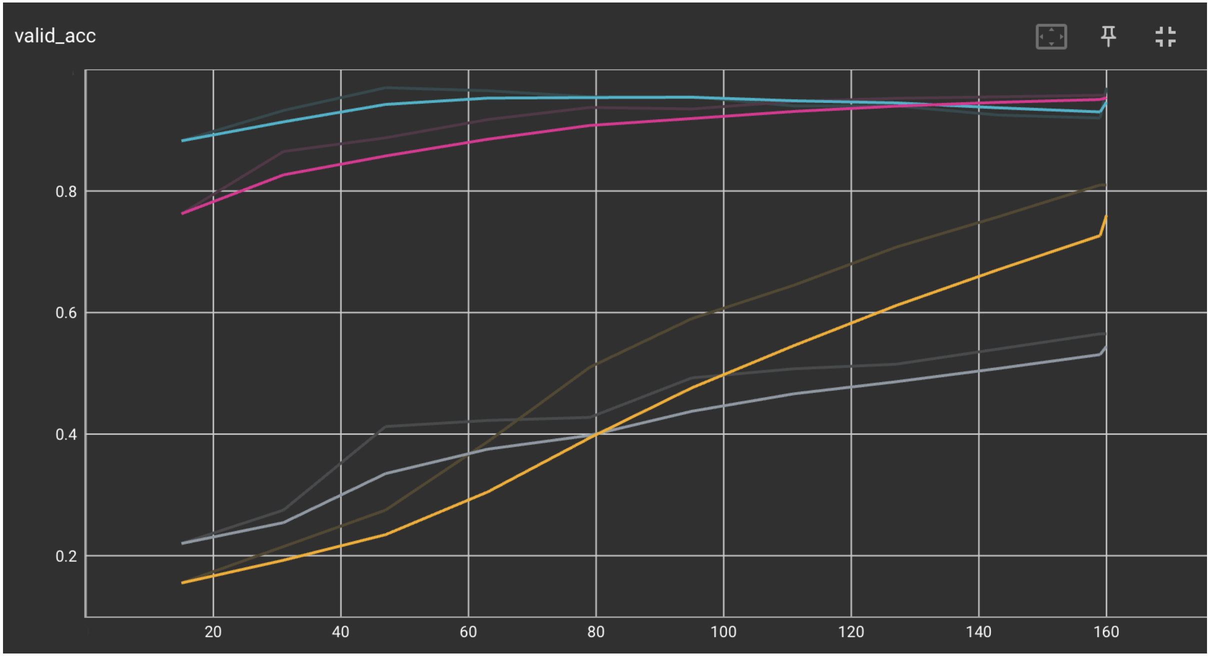
Seed = 1000

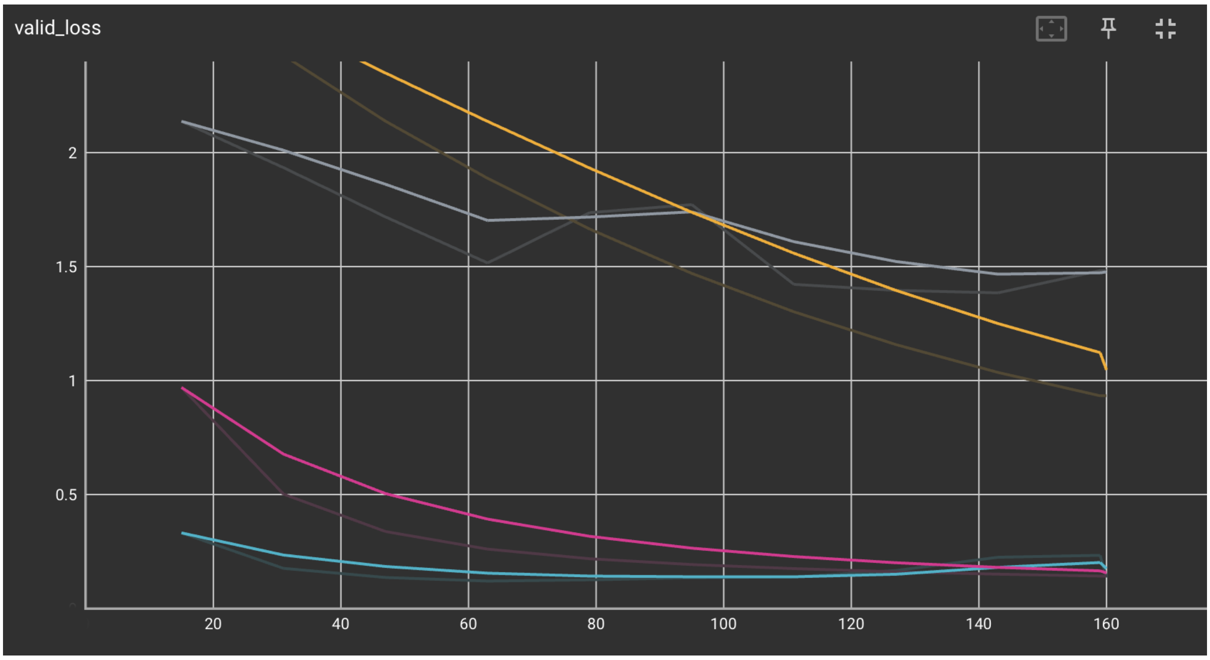
Learning Rate = 0.0001(Yellow), 0.001(Pink), 0.01(Blue), 0.1(Gray)



23M1074_23M1163







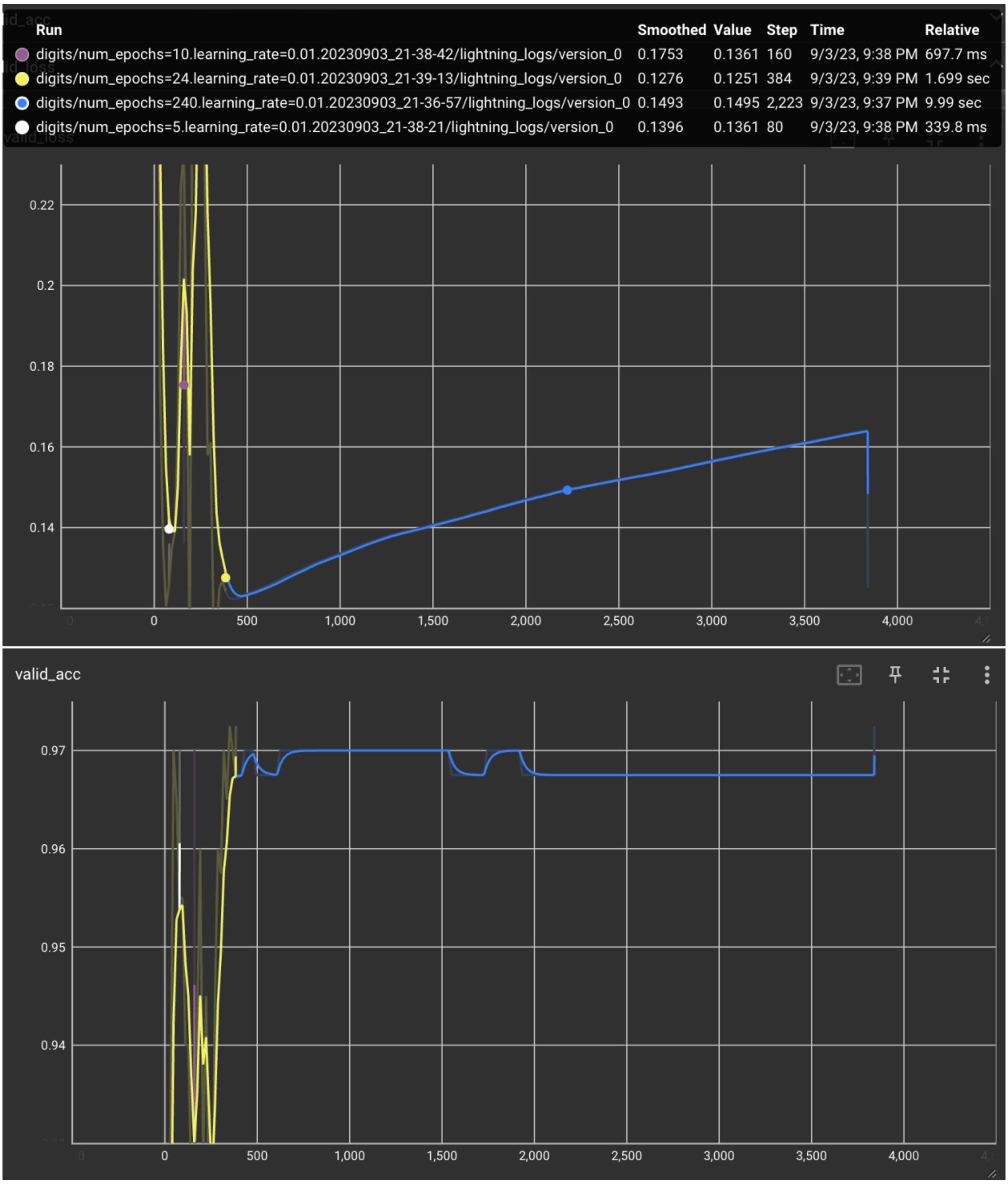
By keeping the epochs and seed fixed (10,1000) we get best accuracy for learning rate = 0.01

2. Finding **best # of epochs** by fixing the learning rate and speed:

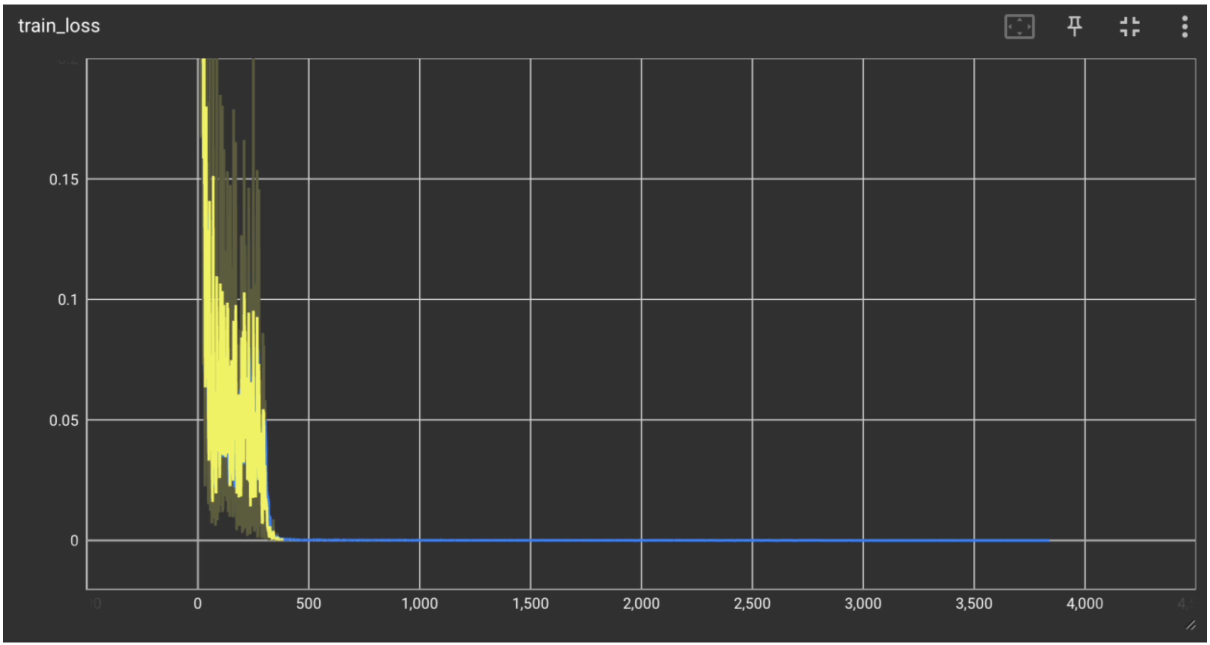
Learning rate = 0.01(calculated previously)

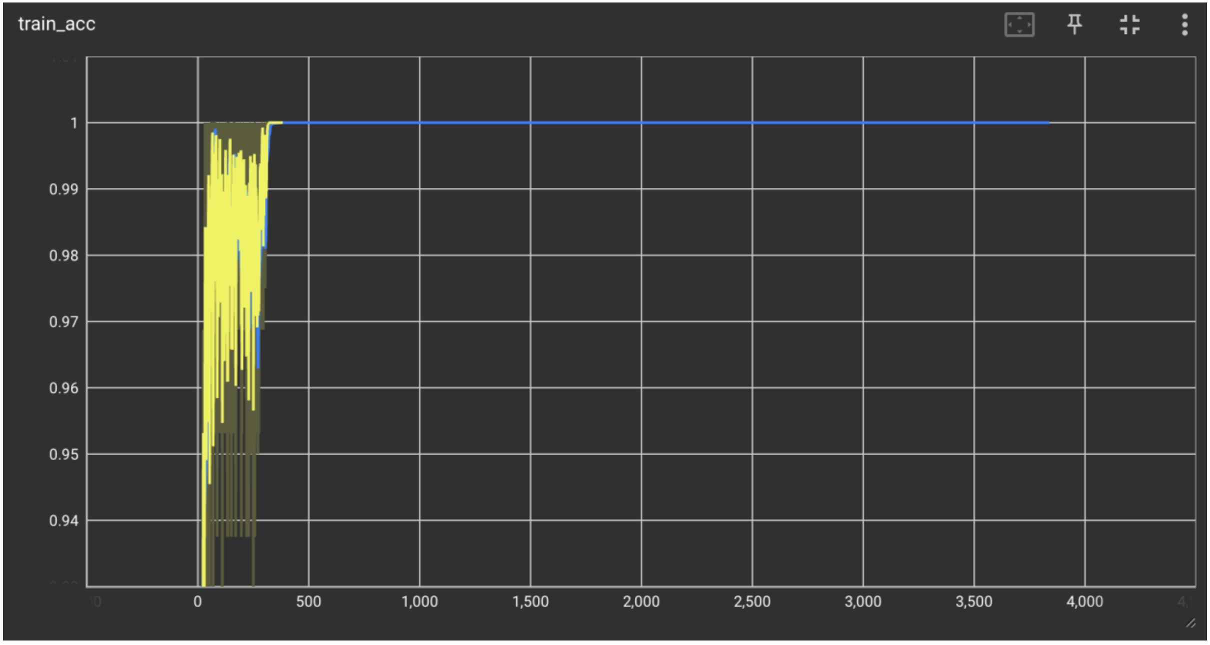
Seed = 1000 (arbitrary)

Valid loss:



	Run	Smoothed	Value	Step	Time	Relative
●	digits/num_epochs=10.learning_rate=0.01.20230903_21-38-42/lightning_logs/version_0	0.1753	0.1361	160	9/3/23, 9:38 PM	697.7 ms
●	digits/num_epochs=24.learning_rate=0.01.20230903_21-39-13/lightning_logs/version_0	0.1276	0.1251	384	9/3/23, 9:39 PM	1.699 sec
●	digits/num_epochs=240.learning_rate=0.01.20230903_21-36-57/lightning_logs/version_0	0.1493	0.1495	2,223	9/3/23, 9:37 PM	9.99 sec
●	digits/num_epochs=5.learning_rate=0.01.20230903_21-38-21/lightning_logs/version_0	0.1396	0.1361	80	9/3/23, 9:38 PM	339.8 ms





We get the best accuracy for epochs somewhat above 24 epochs; anything above it is resulting in overfitting.

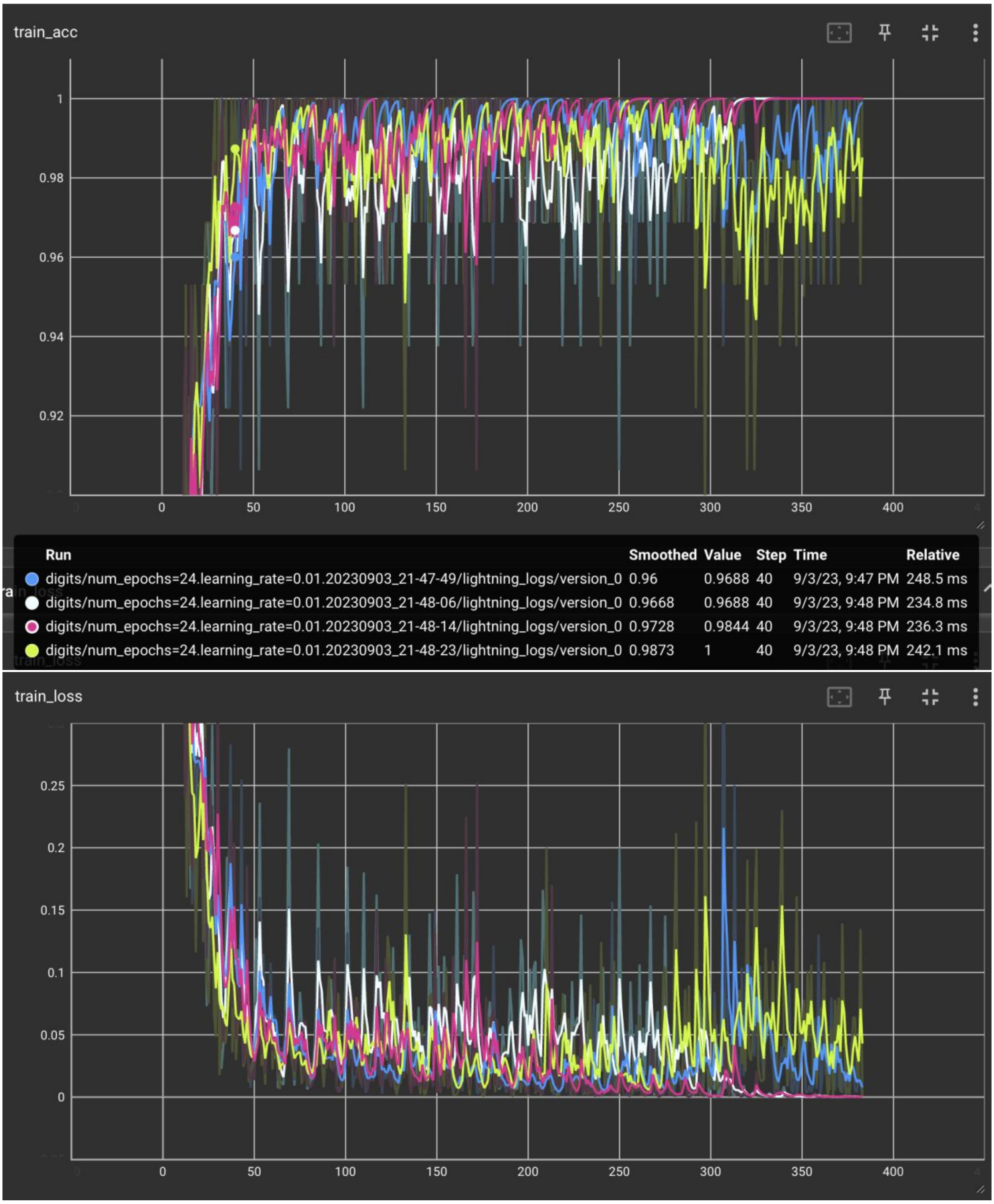
But even after getting the best values from these hyperparameters we also changed the seed values as we were curious and we got to see that seed values also affect the accuracy of our model.

Here by seed we meant the initialization of our weights, random data splicing and random noise augmentation.

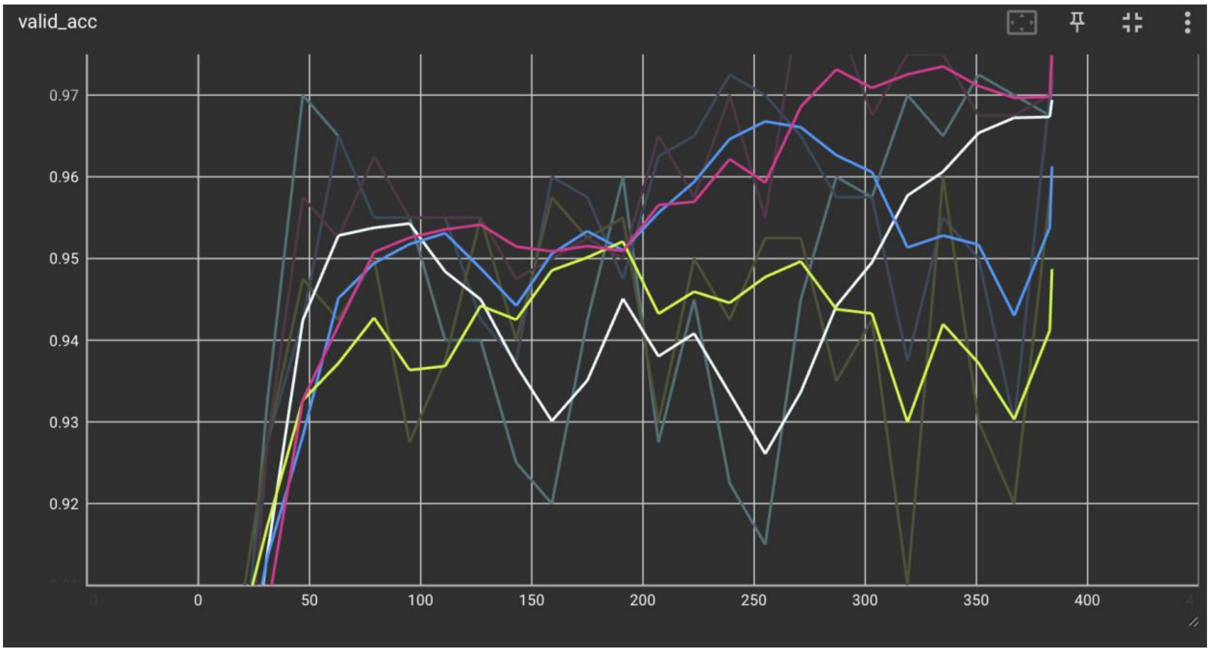
3. Fixed Learning Rate and epochs with variable **Seed values**:

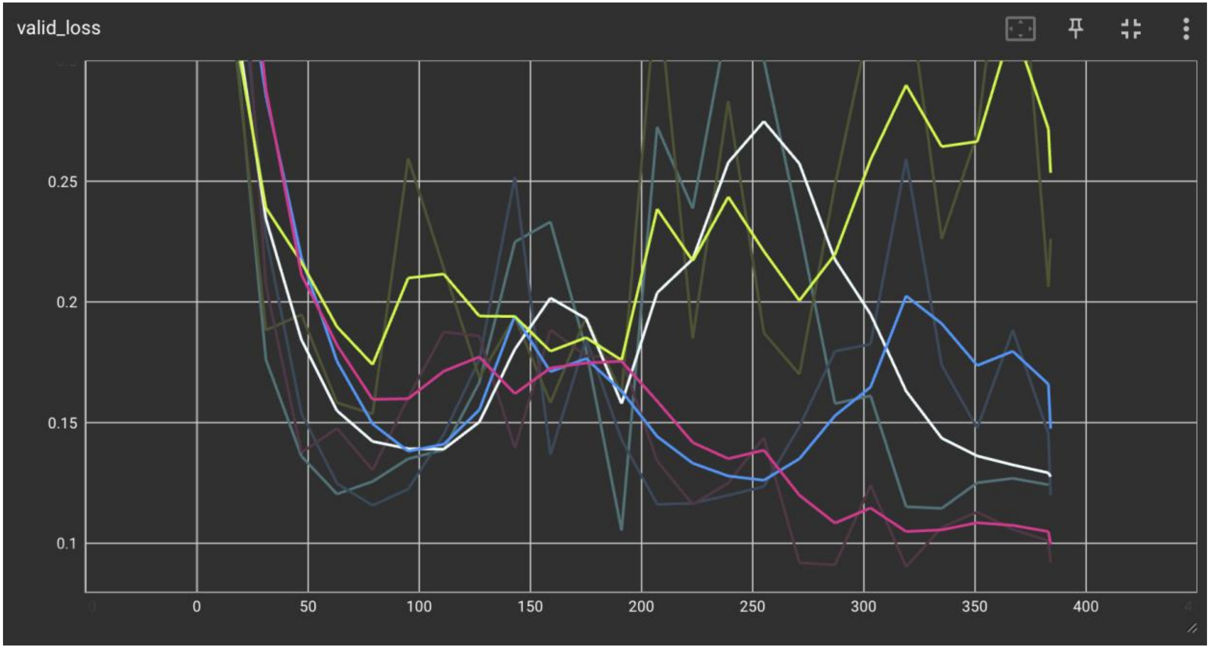
Learning Rate = 0.01

Epochs = 24



Run	Smoothed	Value	Step	Time	Relative
● digits/num_epochs=24.learning_rate=0.01.20230903_21-47-49/lightning_logs/version_0	0.96	0.9688	40	9/3/23, 9:47 PM	248.5 ms
○ digits/num_epochs=24.learning_rate=0.01.20230903_21-48-06/lightning_logs/version_0	0.9668	0.9688	40	9/3/23, 9:48 PM	234.8 ms
● digits/num_epochs=24.learning_rate=0.01.20230903_21-48-14/lightning_logs/version_0	0.9728	0.9844	40	9/3/23, 9:48 PM	236.3 ms
● digits/num_epochs=24.learning_rate=0.01.20230903_21-48-23/lightning_logs/version_0	0.9873	1	40	9/3/23, 9:48 PM	242.1 ms





Seeds values used = 900(yellow) , 1000(blue), 1100(pink), 1600(blue)

The best accuracy that we got and we've submitted is for these hyperparameters

Epochs = 24

Learning rate = 0.01

Seed = 1100

23M1074_23M1163

How to avoid Overfitting 😡:

For avoiding overfitting first in this question, we tried with larger no layers having very few to medium no of neurons and as we increased the no. of layers the “validation accuracy keeps on decreasing” if we decrease the no of neurons in layers the “ validation loss keeps on increasing”.

Then we remembered the theorem that any function can be approximated with a single layer of larger no of neurons and we tried to use it and come up with these settings for our digits model:

1 hidden layer of 150 neurons and this was giving best training and validation accuracy and the least training and validation loss.