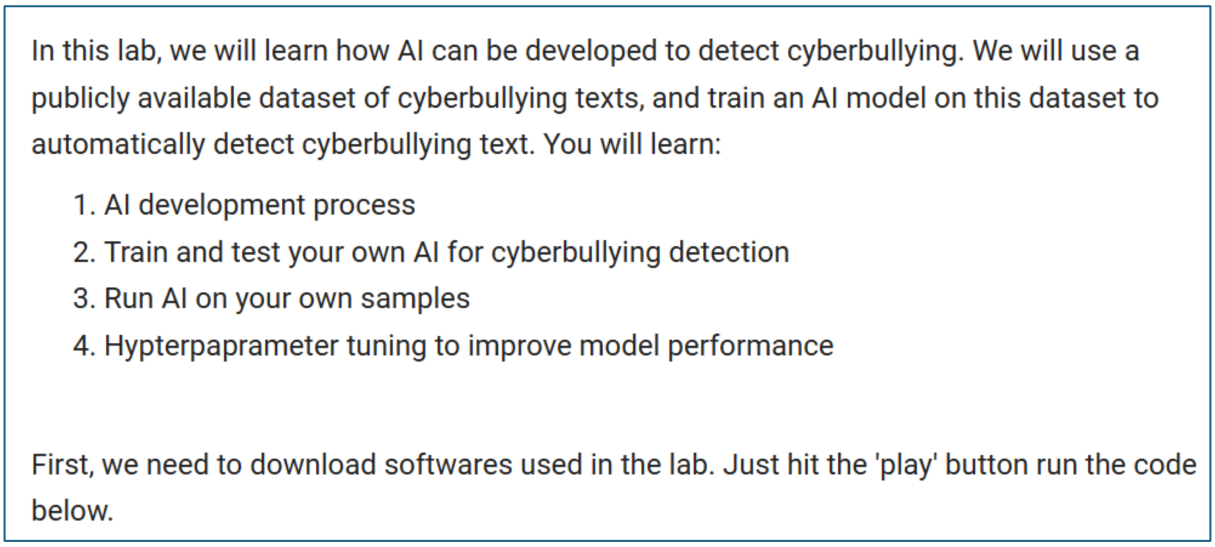
Partha Sai Madallapalli(pua528)

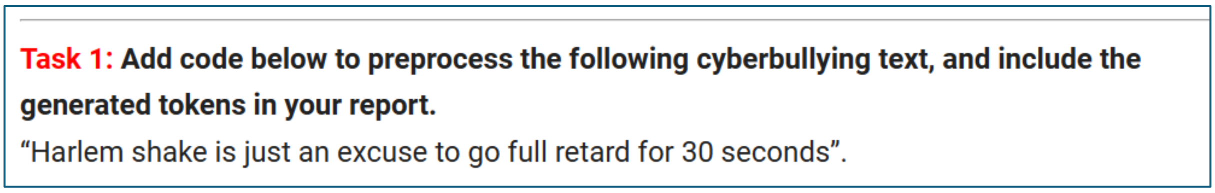
Lab 3: Cyberbullying Detection Using BERT



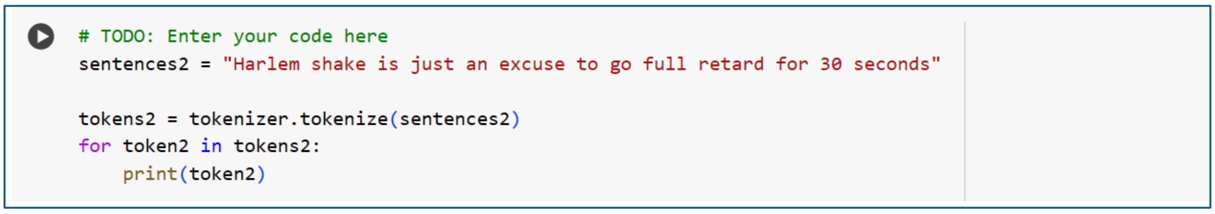
In this lab, we will learn how AI can be developed to detect cyberbullying. We will use a publicly available dataset of cyberbullying texts, and train an AI model on this dataset to automatically detect cyberbullying text. You will learn:

1. AI development process
2. Train and test your own AI for cyberbullying detection
3. Run AI on your own samples
4. Hypterpaprameter tuning to improve model performance

First, we need to download softwares used in the lab. Just hit the 'play' button run the code below.

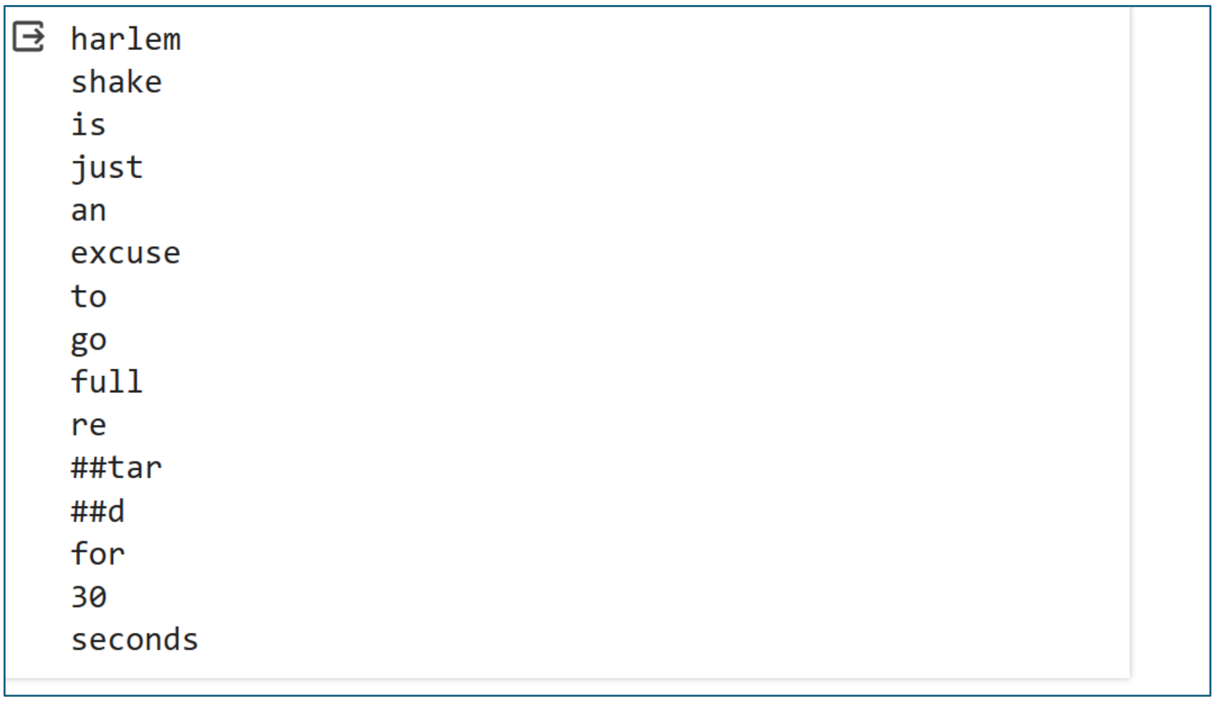


Task 1: **Add code below to preprocess the following cyberbullying text, and include the generated tokens in your report.**
“Harlem shake is just an excuse to go full retard for 30 seconds”.

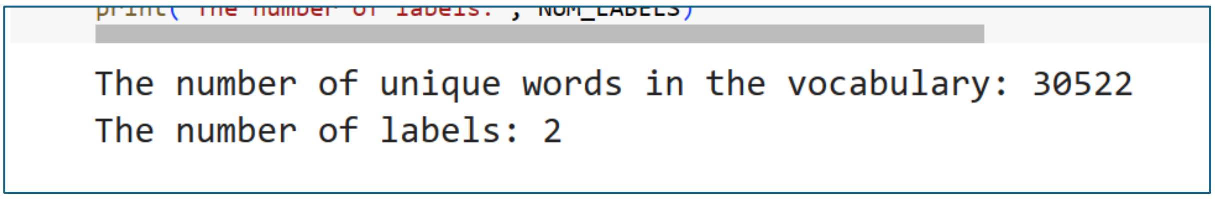


```
# TODO: Enter your code here
sentences2 = "Harlem shake is just an excuse to go full retard for 30 seconds"

tokens2 = tokenizer.tokenize(sentences2)
for token2 in tokens2:
    print(token2)
```

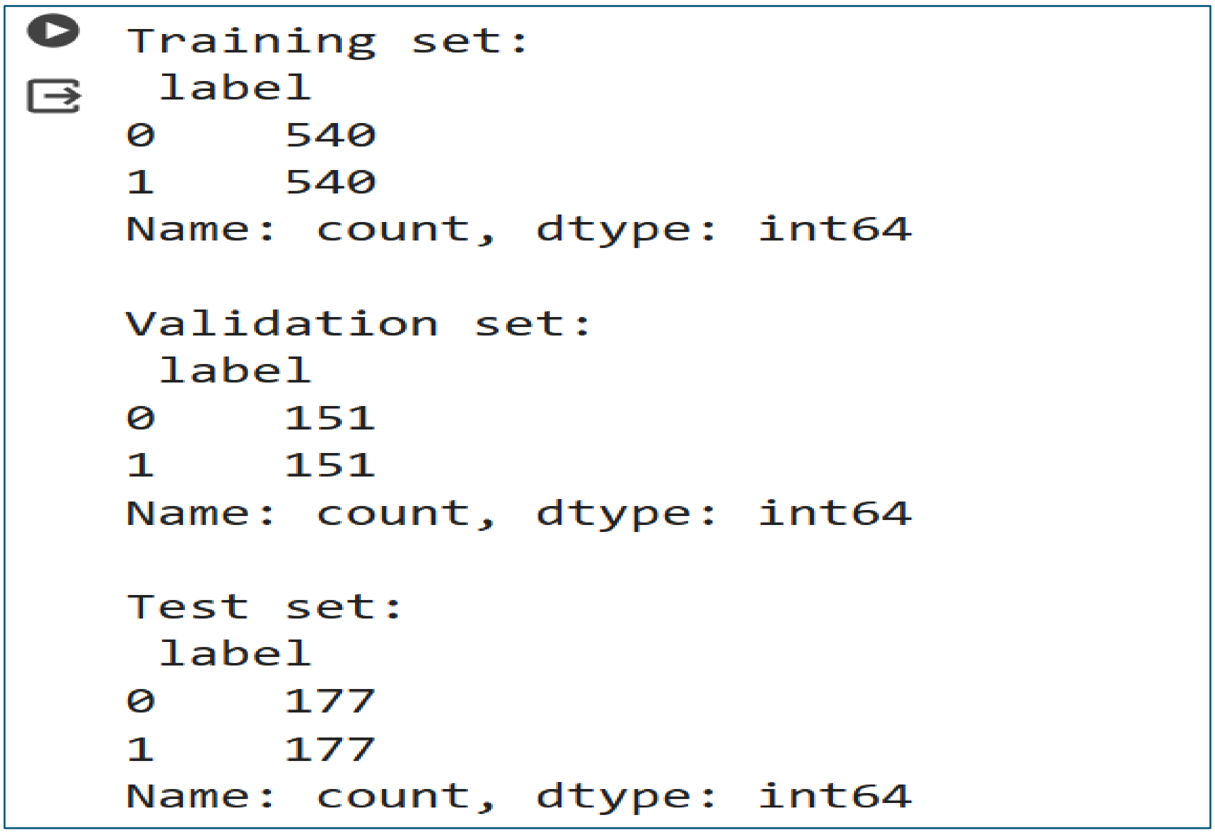
Partha Sai Madallapalli(pua528)



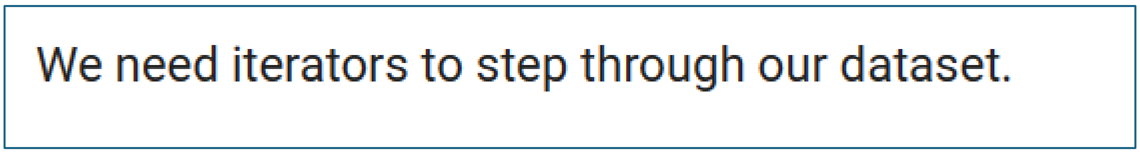
```
harlem
shake
is
just
an
excuse
to
go
full
re
##tar
##d
for
30
seconds
```



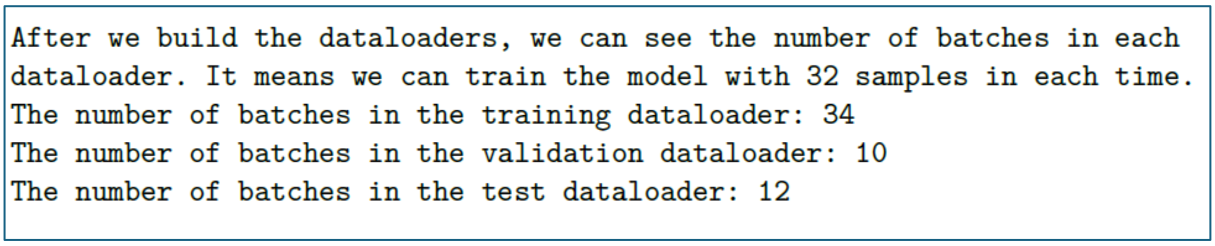
```
print("The number of labels: ", NUM_LABELS)

The number of unique words in the vocabulary: 30522
The number of labels: 2
```



```
Training set:
 label
0    540
1    540
Name: count, dtype: int64

Validation set:
 label
0    151
1    151
Name: count, dtype: int64

Test set:
 label
0    177
1    177
Name: count, dtype: int64
```

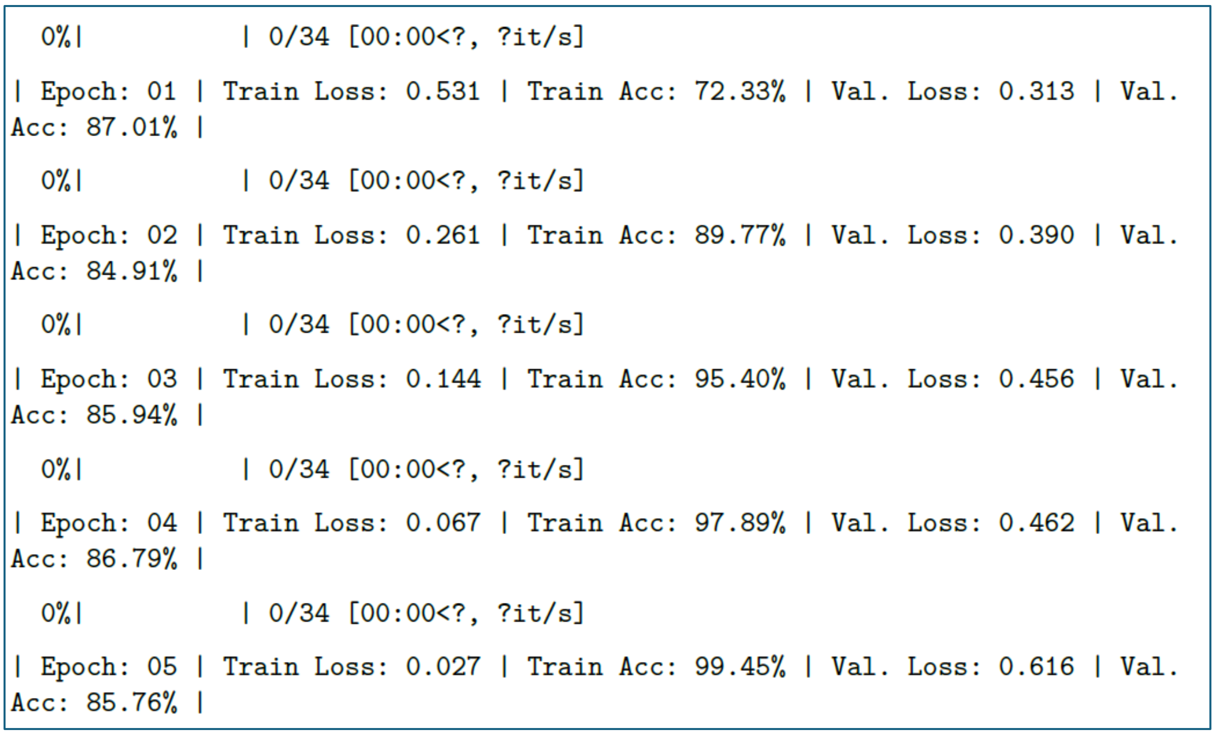


We need iterators to step through our dataset.



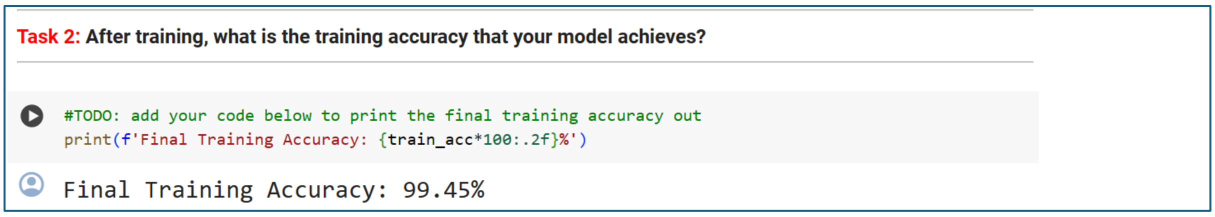
```
After we build the dataloaders, we can see the number of batches in each
dataloader. It means we can train the model with 32 samples in each time.
The number of batches in the training dataloader: 34
The number of batches in the validation dataloader: 10
The number of batches in the test dataloader: 12
```

Training Process



```
  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 01 | Train Loss: 0.531 | Train Acc: 72.33% | Val. Loss: 0.313 | Val.
Acc: 87.01% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 02 | Train Loss: 0.261 | Train Acc: 89.77% | Val. Loss: 0.390 | Val.
Acc: 84.91% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 03 | Train Loss: 0.144 | Train Acc: 95.40% | Val. Loss: 0.456 | Val.
Acc: 85.94% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 04 | Train Loss: 0.067 | Train Acc: 97.89% | Val. Loss: 0.462 | Val.
Acc: 86.79% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 05 | Train Loss: 0.027 | Train Acc: 99.45% | Val. Loss: 0.616 | Val.
Acc: 85.76% |
```

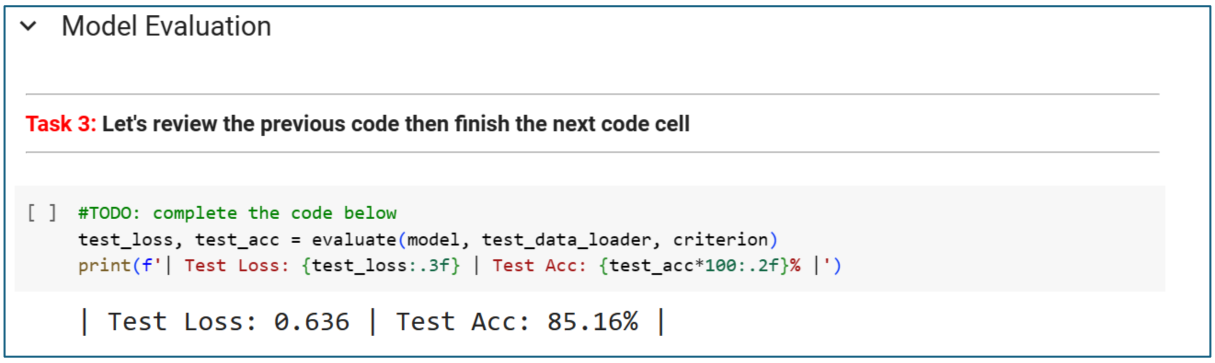
Partha Sai Madallapalli(pua528)



Task 2: After training, what is the training accuracy that your model achieves?

```
#TODO: add your code below to print the final training accuracy out
print(f'Final Training Accuracy: {train_acc*100:.2f}%')
```

```
Final Training Accuracy: 99.45%
```

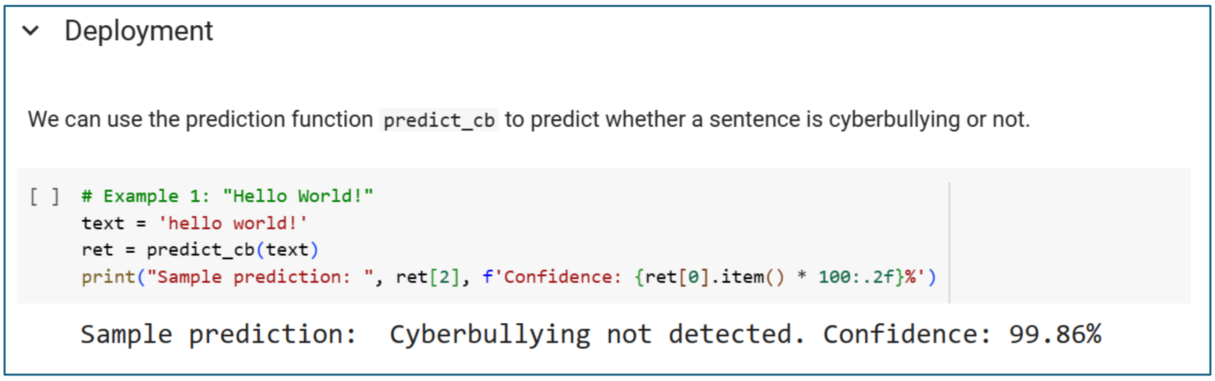


Model Evaluation

Task 3: Let's review the previous code then finish the next code cell

```
#TODO: complete the code below
test_loss, test_acc = evaluate(model, test_data_loader, criterion)
print(f'| Test Loss: {test_loss:.3f} | Test Acc: {test_acc*100:.2f}% |')
```

```
| Test Loss: 0.636 | Test Acc: 85.16% |
```

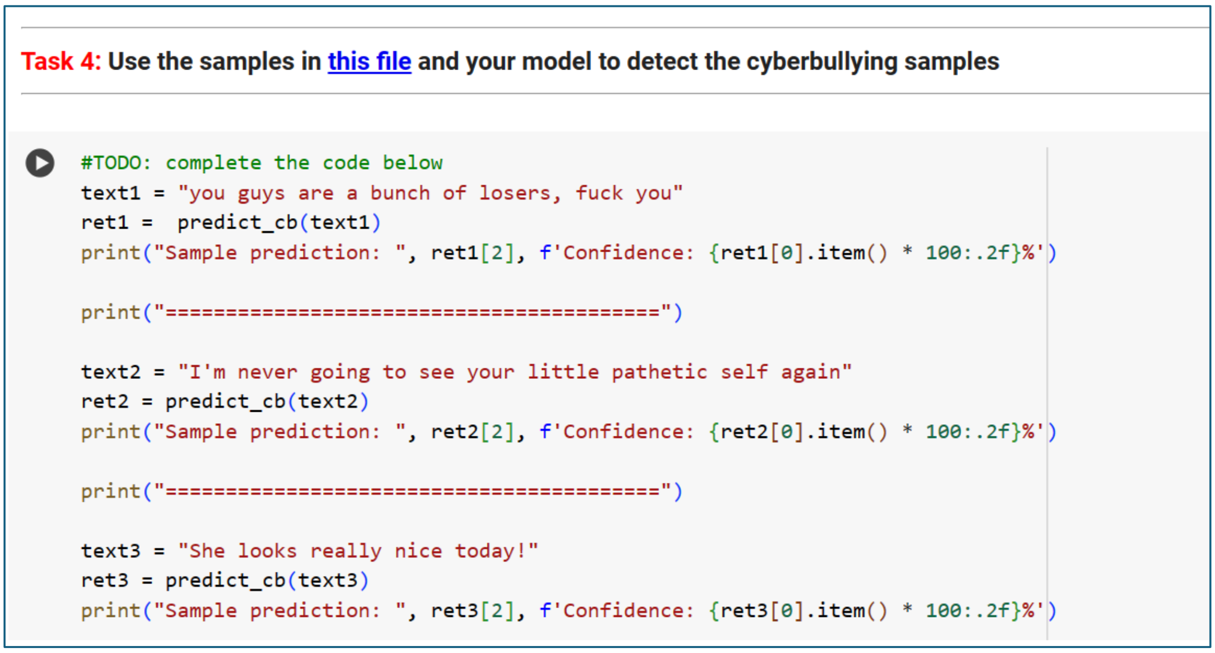


Deployment

We can use the prediction function `predict_cb` to predict whether a sentence is cyberbullying or not.

```
# Example 1: "Hello World!"
text = 'hello world!'
ret = predict_cb(text)
print("Sample prediction: ", ret[2], f'Confidence: {ret[0].item() * 100:.2f}%')
```

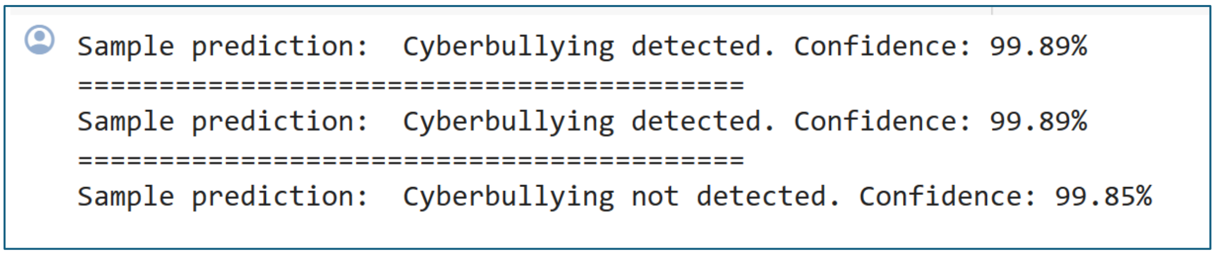
```
Sample prediction:  Cyberbullying not detected. Confidence: 99.86%
```



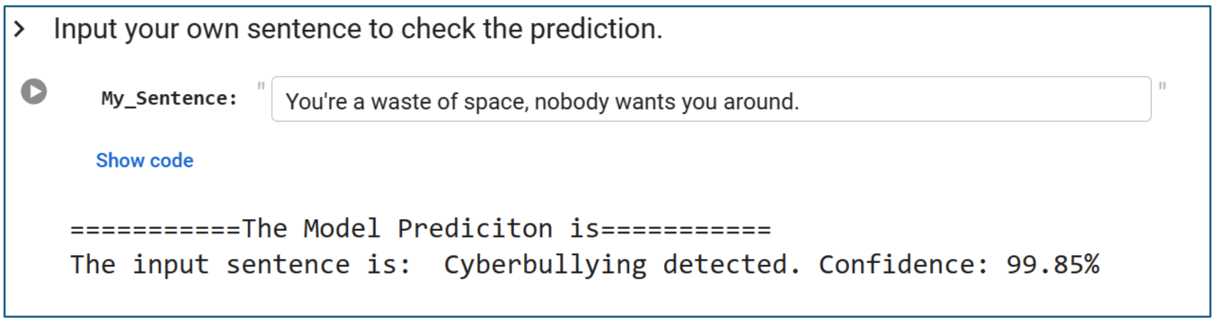
Task 4: Use the samples in this file and your model to detect the cyberbullying samples

```
#TODO: complete the code below
text1 = "you guys are a bunch of losers, fuck you"
ret1 =  predict_cb(text1)
print("Sample prediction: ", ret1[2], f'Confidence: {ret1[0].item() * 100:.2f}%')

print("=========================================")

text2 = "I'm never going to see your little pathetic self again"
ret2 = predict_cb(text2)
print("Sample prediction: ", ret2[2], f'Confidence: {ret2[0].item() * 100:.2f}%')

print("=========================================")

text3 = "She looks really nice today!"
ret3 = predict_cb(text3)
print("Sample prediction: ", ret3[2], f'Confidence: {ret3[0].item() * 100:.2f}%')
```

Partha Sai Madallapalli(pua528)



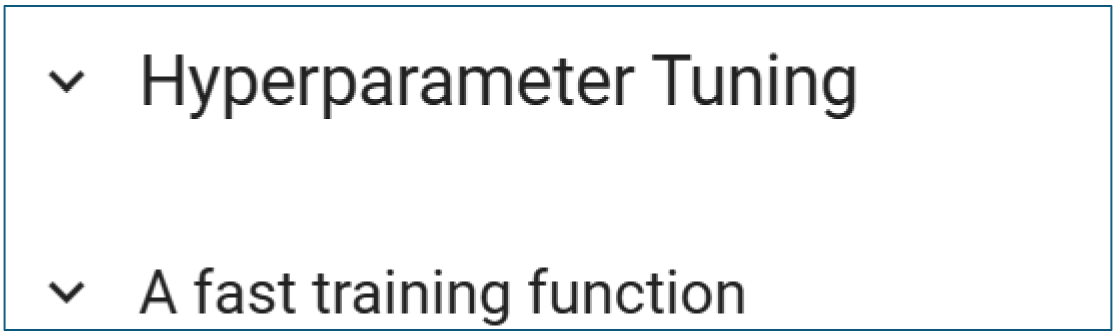
```
Sample prediction:  Cyberbullying detected. Confidence: 99.89%
==========================================
Sample prediction:  Cyberbullying detected. Confidence: 99.89%
==========================================
Sample prediction:  Cyberbullying not detected. Confidence: 99.85%
```



Input your own sentence to check the prediction.

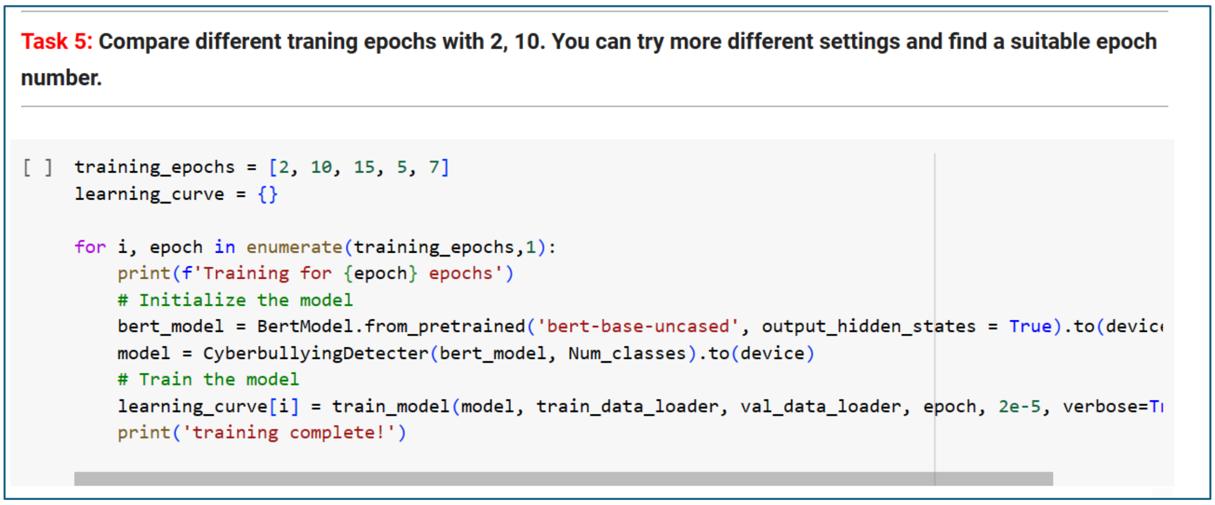
My_Sentence: " You're a waste of space, nobody wants you around. "

```
===========The Model Prediciton is===========
The input sentence is:  Cyberbullying detected. Confidence: 99.85%
```



Hyperparameter Tuning

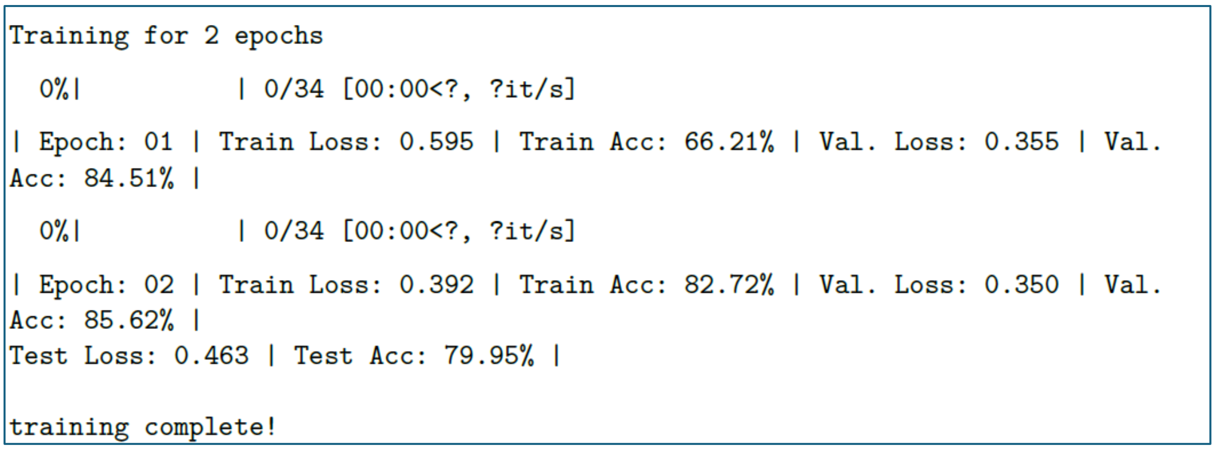
A fast training function



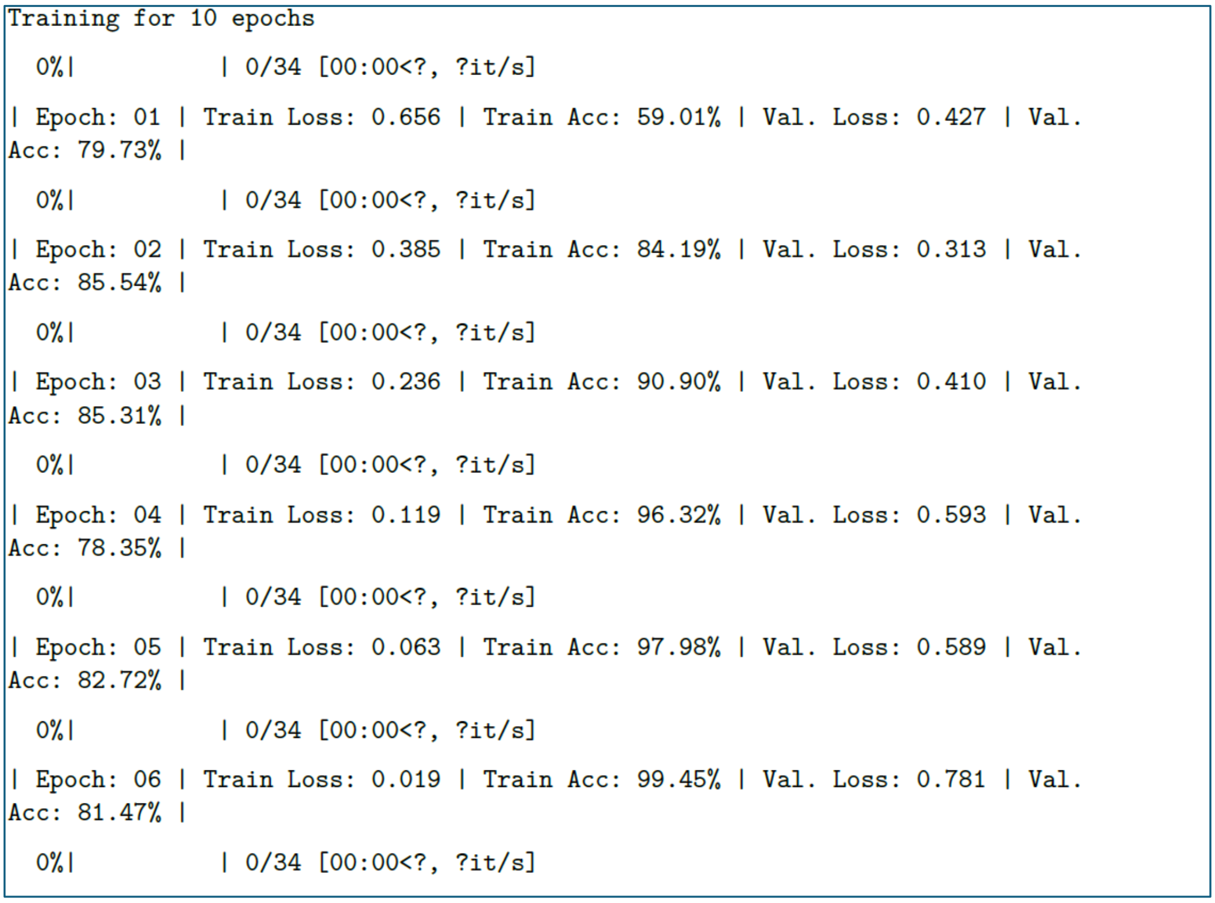
Task 5: Compare different traning epochs with 2, 10. You can try more different settings and find a suitable epoch number.

```
training_epochs = [2, 10, 15, 5, 7]
learning_curve = {}

for i, epoch in enumerate(training_epochs,1):
    print(f'Training for {epoch} epochs')
    # Initialize the model
    bert_model = BertModel.from_pretrained('bert-base-uncased', output_hidden_states = True).to(devic
    model = CyberbullyingDetecter(bert_model, Num_classes).to(device)
    # Train the model
    learning_curve[i] = train_model(model, train_data_loader, val_data_loader, epoch, 2e-5, verbose=T
    print('training complete!')
```

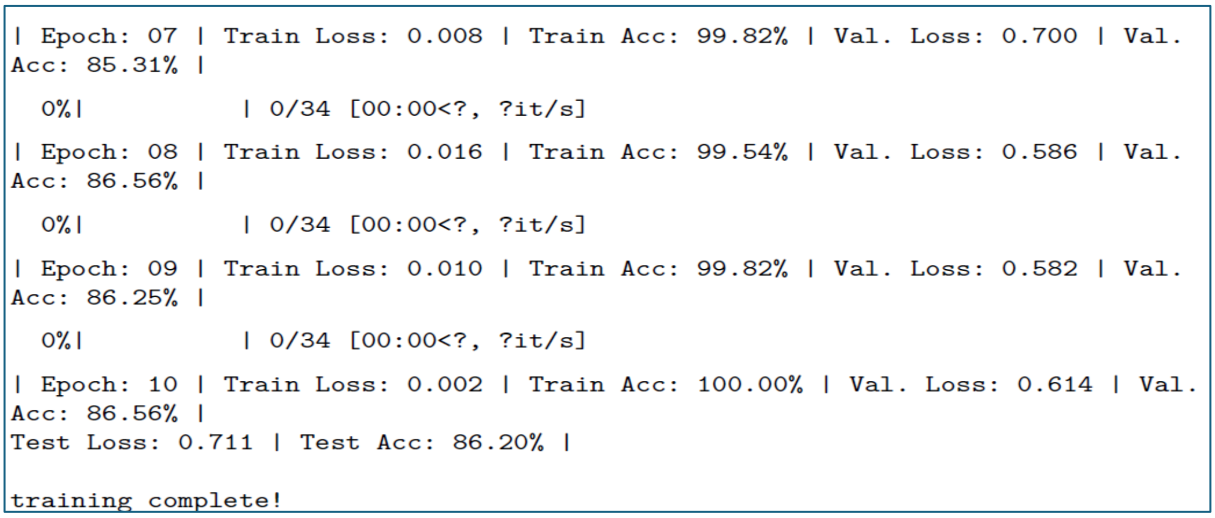
Partha Sai Madallapalli(pua528)



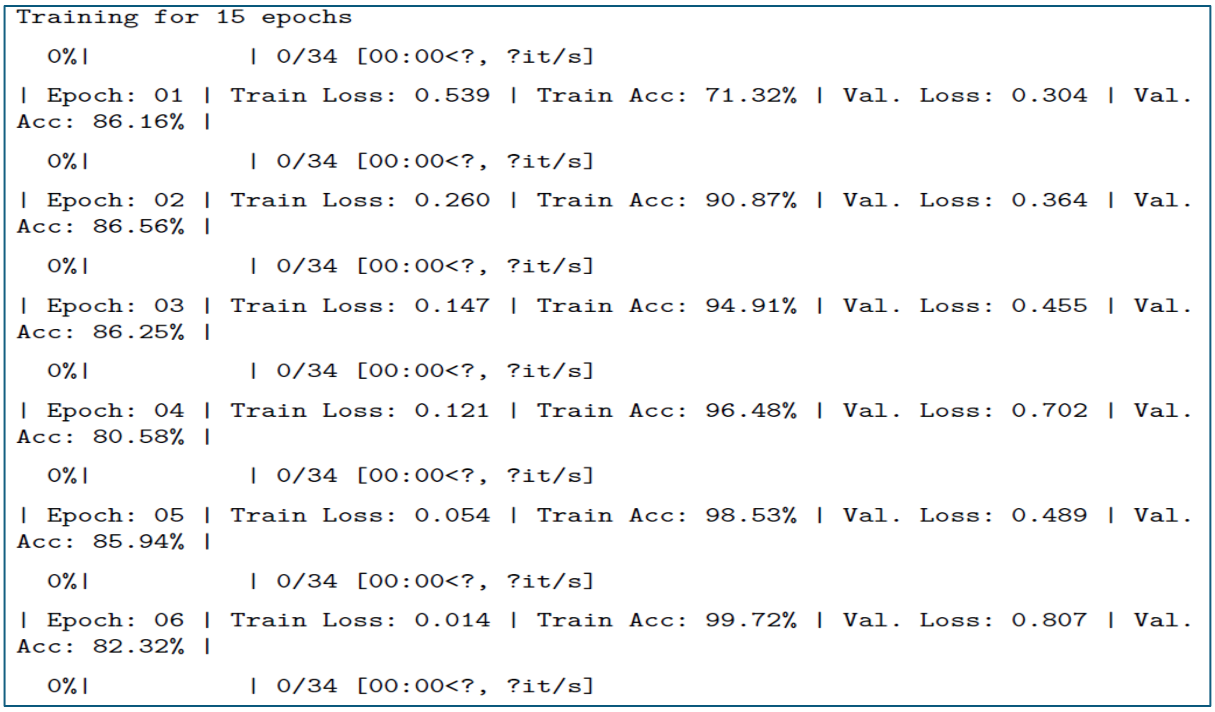
```
Training for 2 epochs
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 01 | Train Loss: 0.595 | Train Acc: 66.21% | Val. Loss: 0.355 | Val.
Acc: 84.51% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 02 | Train Loss: 0.392 | Train Acc: 82.72% | Val. Loss: 0.350 | Val.
Acc: 85.62% |
Test Loss: 0.463 | Test Acc: 79.95% |

training complete!
```



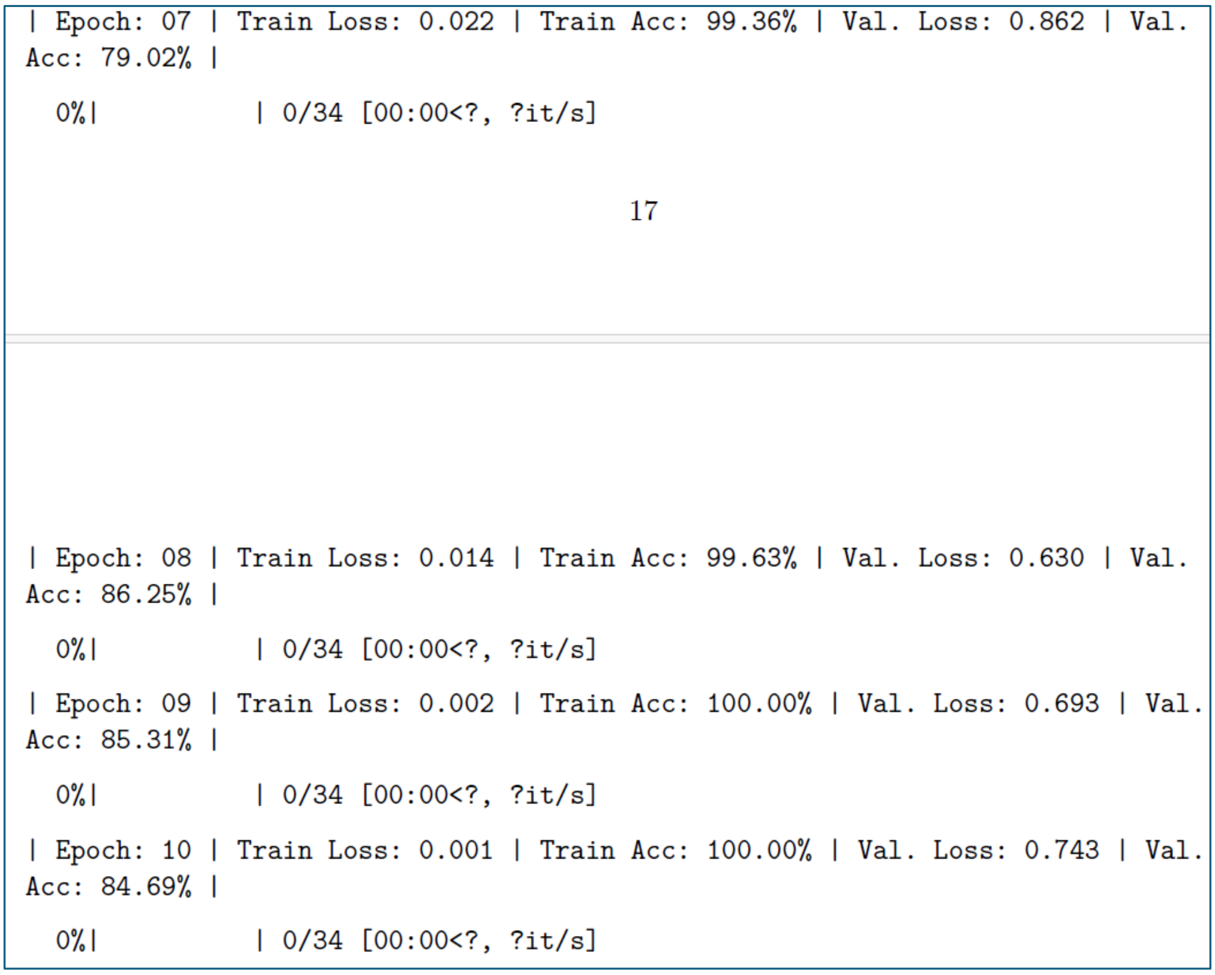
```
Training for 10 epochs
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 01 | Train Loss: 0.656 | Train Acc: 59.01% | Val. Loss: 0.427 | Val.
Acc: 79.73% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 02 | Train Loss: 0.385 | Train Acc: 84.19% | Val. Loss: 0.313 | Val.
Acc: 85.54% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 03 | Train Loss: 0.236 | Train Acc: 90.90% | Val. Loss: 0.410 | Val.
Acc: 85.31% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 04 | Train Loss: 0.119 | Train Acc: 96.32% | Val. Loss: 0.593 | Val.
Acc: 78.35% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 05 | Train Loss: 0.063 | Train Acc: 97.98% | Val. Loss: 0.589 | Val.
Acc: 82.72% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 06 | Train Loss: 0.019 | Train Acc: 99.45% | Val. Loss: 0.781 | Val.
Acc: 81.47% |
  0%|          | 0/34 [00:00<?, ?it/s]
```



```
| Epoch: 07 | Train Loss: 0.008 | Train Acc: 99.82% | Val. Loss: 0.700 | Val.
Acc: 85.31% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 08 | Train Loss: 0.016 | Train Acc: 99.54% | Val. Loss: 0.586 | Val.
Acc: 86.56% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 09 | Train Loss: 0.010 | Train Acc: 99.82% | Val. Loss: 0.582 | Val.
Acc: 86.25% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 10 | Train Loss: 0.002 | Train Acc: 100.00% | Val. Loss: 0.614 | Val.
Acc: 86.56% |
Test Loss: 0.711 | Test Acc: 86.20% |

training complete!
```



```
Training for 15 epochs
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 01 | Train Loss: 0.539 | Train Acc: 71.32% | Val. Loss: 0.304 | Val.
Acc: 86.16% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 02 | Train Loss: 0.260 | Train Acc: 90.87% | Val. Loss: 0.364 | Val.
Acc: 86.56% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 03 | Train Loss: 0.147 | Train Acc: 94.91% | Val. Loss: 0.455 | Val.
Acc: 86.25% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 04 | Train Loss: 0.121 | Train Acc: 96.48% | Val. Loss: 0.702 | Val.
Acc: 80.58% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 05 | Train Loss: 0.054 | Train Acc: 98.53% | Val. Loss: 0.489 | Val.
Acc: 85.94% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 06 | Train Loss: 0.014 | Train Acc: 99.72% | Val. Loss: 0.807 | Val.
Acc: 82.32% |
  0%|          | 0/34 [00:00<?, ?it/s]
```

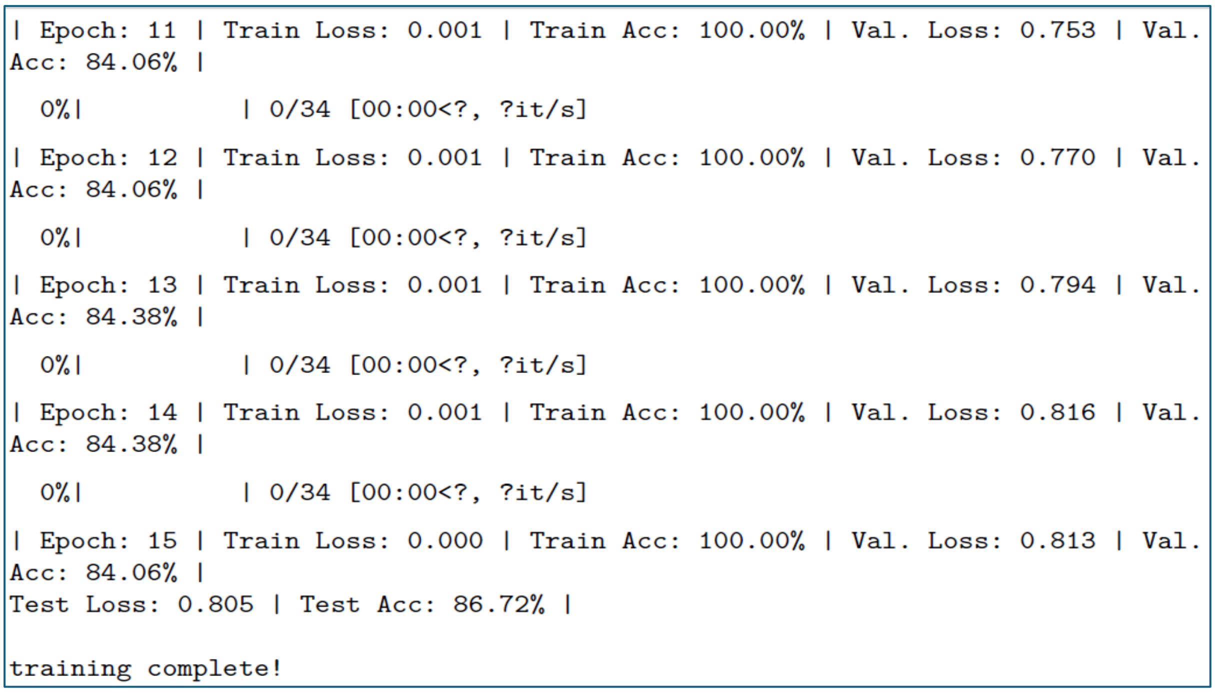


```
| Epoch: 07 | Train Loss: 0.022 | Train Acc: 99.36% | Val. Loss: 0.862 | Val.
Acc: 79.02% |
  0%|          | 0/34 [00:00<?, ?it/s]
```

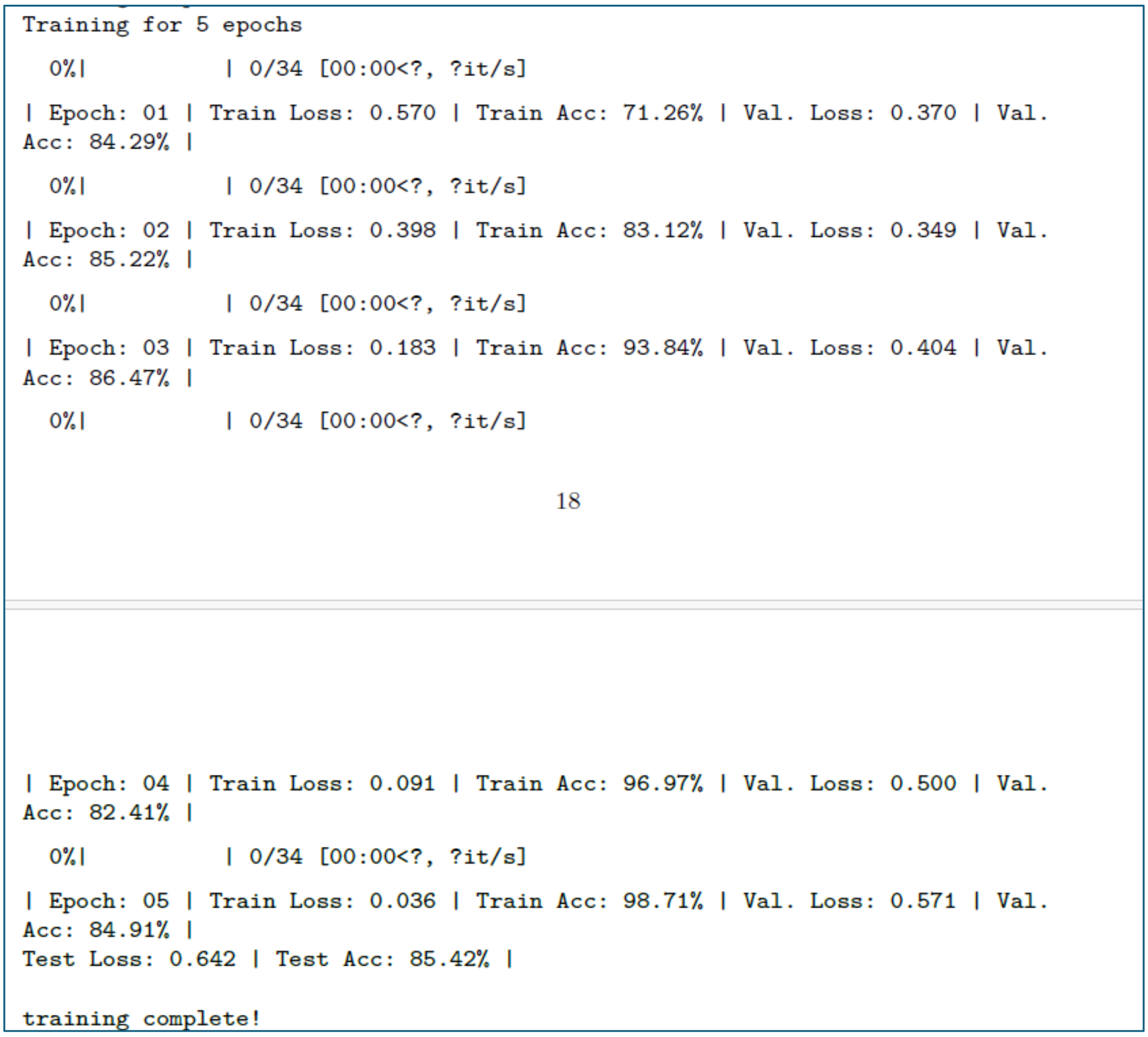
17

```
| Epoch: 08 | Train Loss: 0.014 | Train Acc: 99.63% | Val. Loss: 0.630 | Val.
Acc: 86.25% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 09 | Train Loss: 0.002 | Train Acc: 100.00% | Val. Loss: 0.693 | Val.
Acc: 85.31% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 10 | Train Loss: 0.001 | Train Acc: 100.00% | Val. Loss: 0.743 | Val.
Acc: 84.69% |
  0%|          | 0/34 [00:00<?, ?it/s]
```

Partha Sai Madallapalli(pua528)



```
| Epoch: 11 | Train Loss: 0.001 | Train Acc: 100.00% | Val. Loss: 0.753 | Val.
Acc: 84.06% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 12 | Train Loss: 0.001 | Train Acc: 100.00% | Val. Loss: 0.770 | Val.
Acc: 84.06% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 13 | Train Loss: 0.001 | Train Acc: 100.00% | Val. Loss: 0.794 | Val.
Acc: 84.38% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 14 | Train Loss: 0.001 | Train Acc: 100.00% | Val. Loss: 0.816 | Val.
Acc: 84.38% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 15 | Train Loss: 0.000 | Train Acc: 100.00% | Val. Loss: 0.813 | Val.
Acc: 84.06% |
Test Loss: 0.805 | Test Acc: 86.72% |

training complete!
```

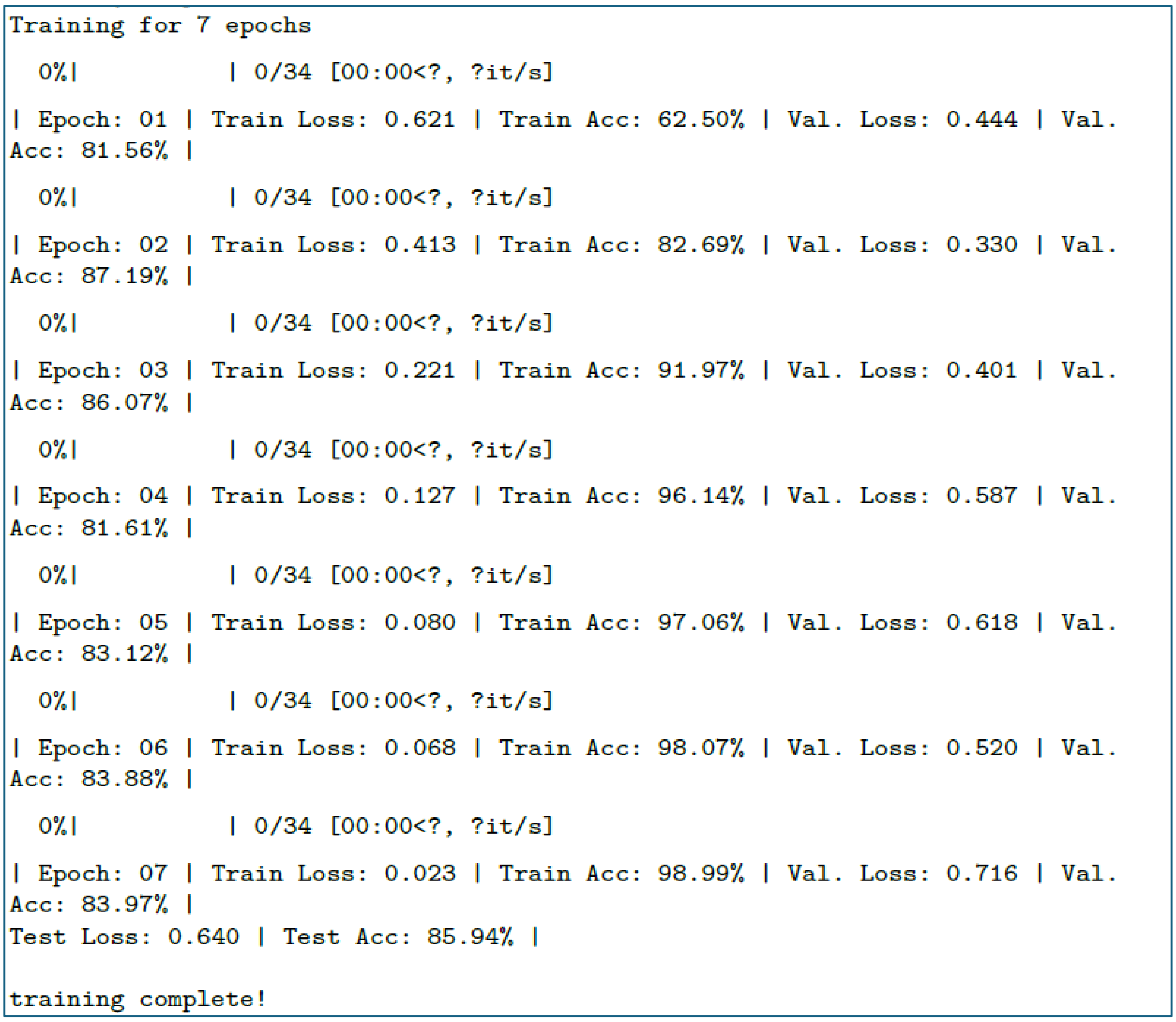


```
Training for 5 epochs

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 01 | Train Loss: 0.570 | Train Acc: 71.26% | Val. Loss: 0.370 | Val.
Acc: 84.29% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 02 | Train Loss: 0.398 | Train Acc: 83.12% | Val. Loss: 0.349 | Val.
Acc: 85.22% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 03 | Train Loss: 0.183 | Train Acc: 93.84% | Val. Loss: 0.404 | Val.
Acc: 86.47% |

  0%|          | 0/34 [00:00<?, ?it/s]
```

18

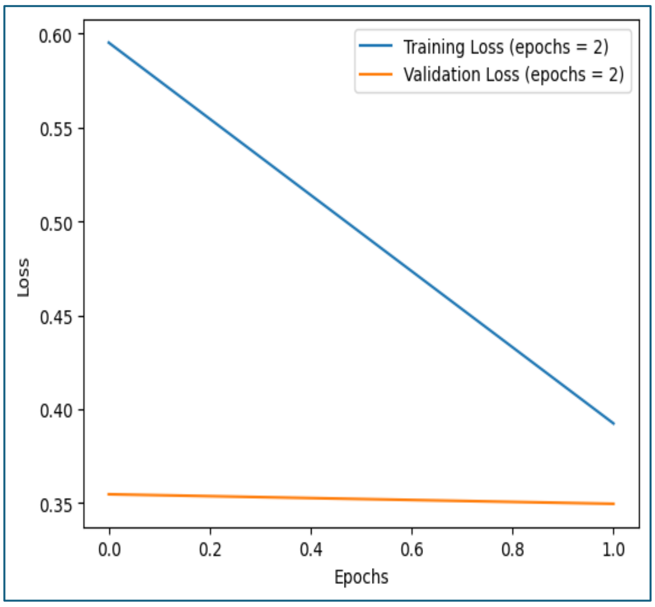
```
| Epoch: 04 | Train Loss: 0.091 | Train Acc: 96.97% | Val. Loss: 0.500 | Val.
Acc: 82.41% |

  0%|          | 0/34 [00:00<?, ?it/s]

| Epoch: 05 | Train Loss: 0.036 | Train Acc: 98.71% | Val. Loss: 0.571 | Val.
Acc: 84.91% |
Test Loss: 0.642 | Test Acc: 85.42% |

training complete!
```

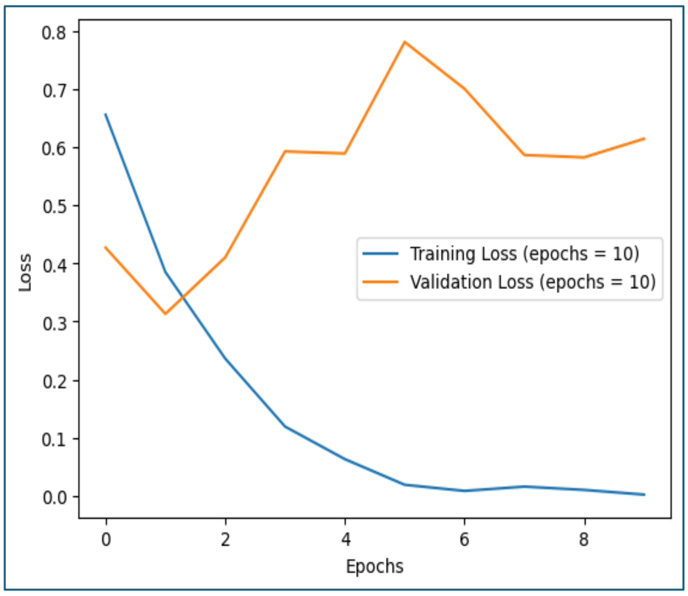
Partha Sai Madallapalli(pua528)



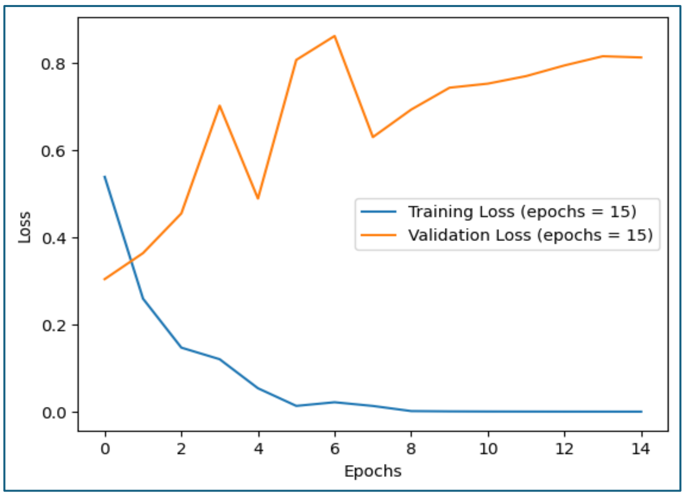
```
Training for 7 epochs
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 01 | Train Loss: 0.621 | Train Acc: 62.50% | Val. Loss: 0.444 | Val.
Acc: 81.56% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 02 | Train Loss: 0.413 | Train Acc: 82.69% | Val. Loss: 0.330 | Val.
Acc: 87.19% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 03 | Train Loss: 0.221 | Train Acc: 91.97% | Val. Loss: 0.401 | Val.
Acc: 86.07% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 04 | Train Loss: 0.127 | Train Acc: 96.14% | Val. Loss: 0.587 | Val.
Acc: 81.61% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 05 | Train Loss: 0.080 | Train Acc: 97.06% | Val. Loss: 0.618 | Val.
Acc: 83.12% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 06 | Train Loss: 0.068 | Train Acc: 98.07% | Val. Loss: 0.520 | Val.
Acc: 83.88% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 07 | Train Loss: 0.023 | Train Acc: 98.99% | Val. Loss: 0.716 | Val.
Acc: 83.97% |
Test Loss: 0.640 | Test Acc: 85.94% |

training complete!
```

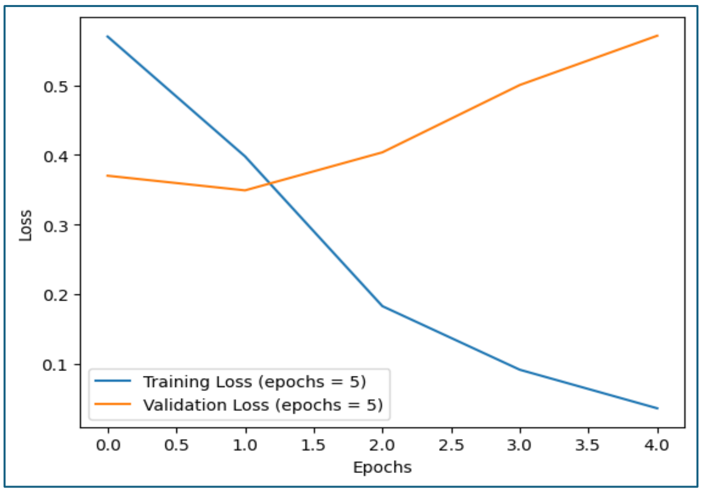
Plotting Curves for different epochs vs training ~ validation loss:

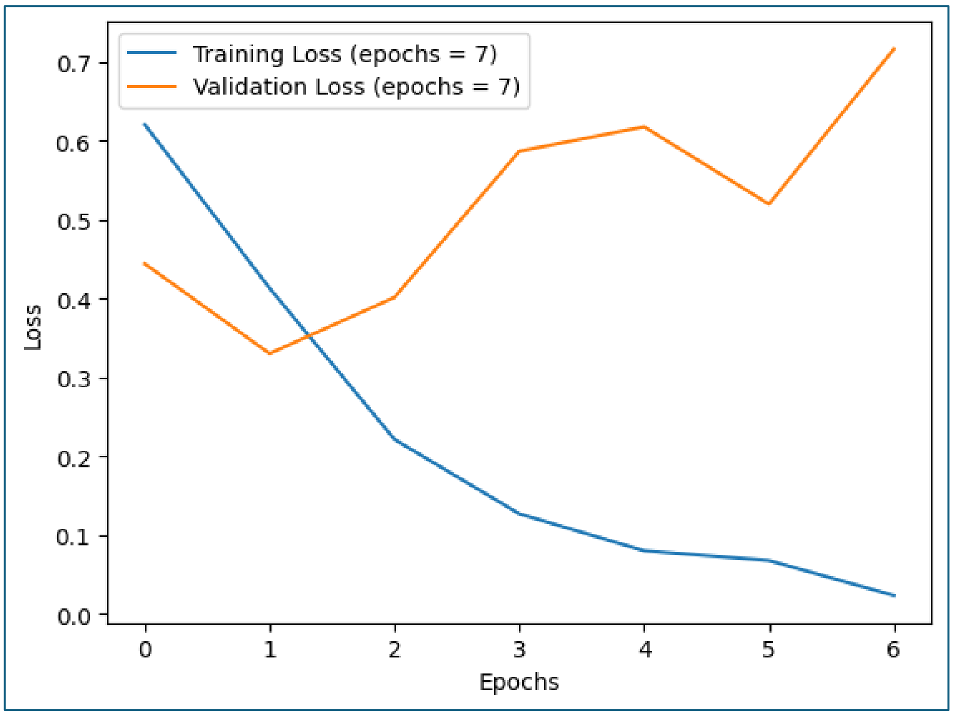




Partha Sai Madallapalli(pua528)





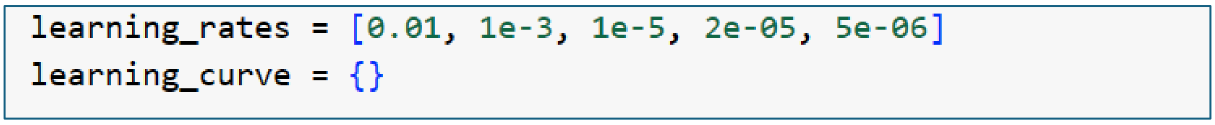


Interpretation: From the above graph it is understandable that, as the number of epochs increases the validation loss also increases at the same time at this point the model will not get converged, as the converged model look for minimized validation loss. Hence the model would be converged at lowest epoch in this scenario which is epochs of 2 with minimum validation loss or decreased validation loss.

Task 6: Compare different learning rates with 0.1, 1e-3 and 1e-5. You can try with your own settings and find the best learning rate.

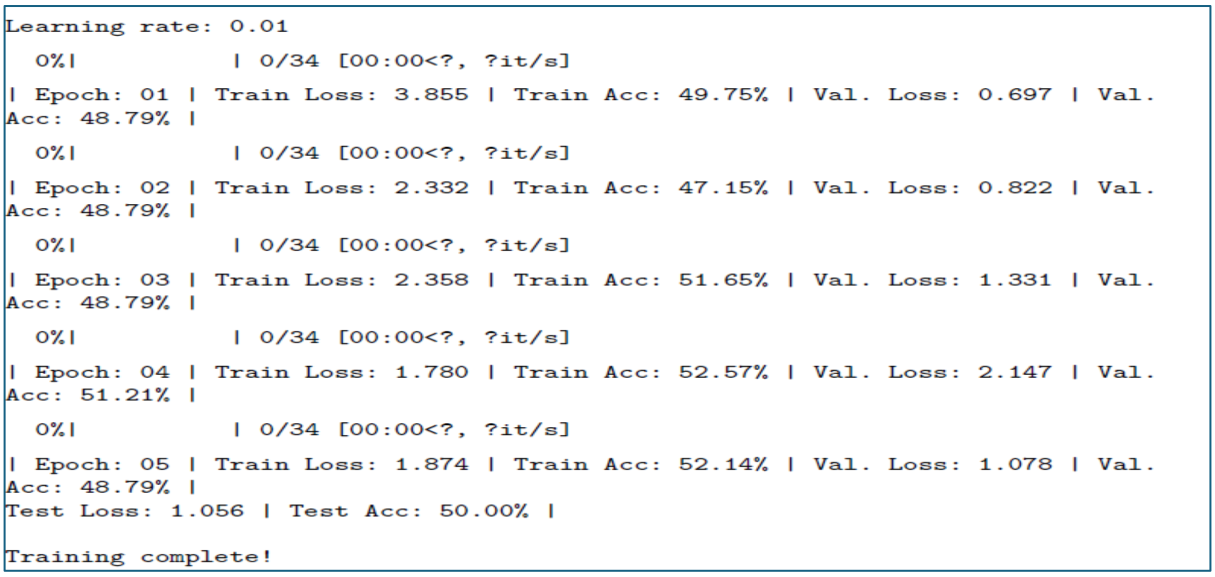
Tried with different settings for the best learning rate:

Partha Sai Madallapalli(pua528)

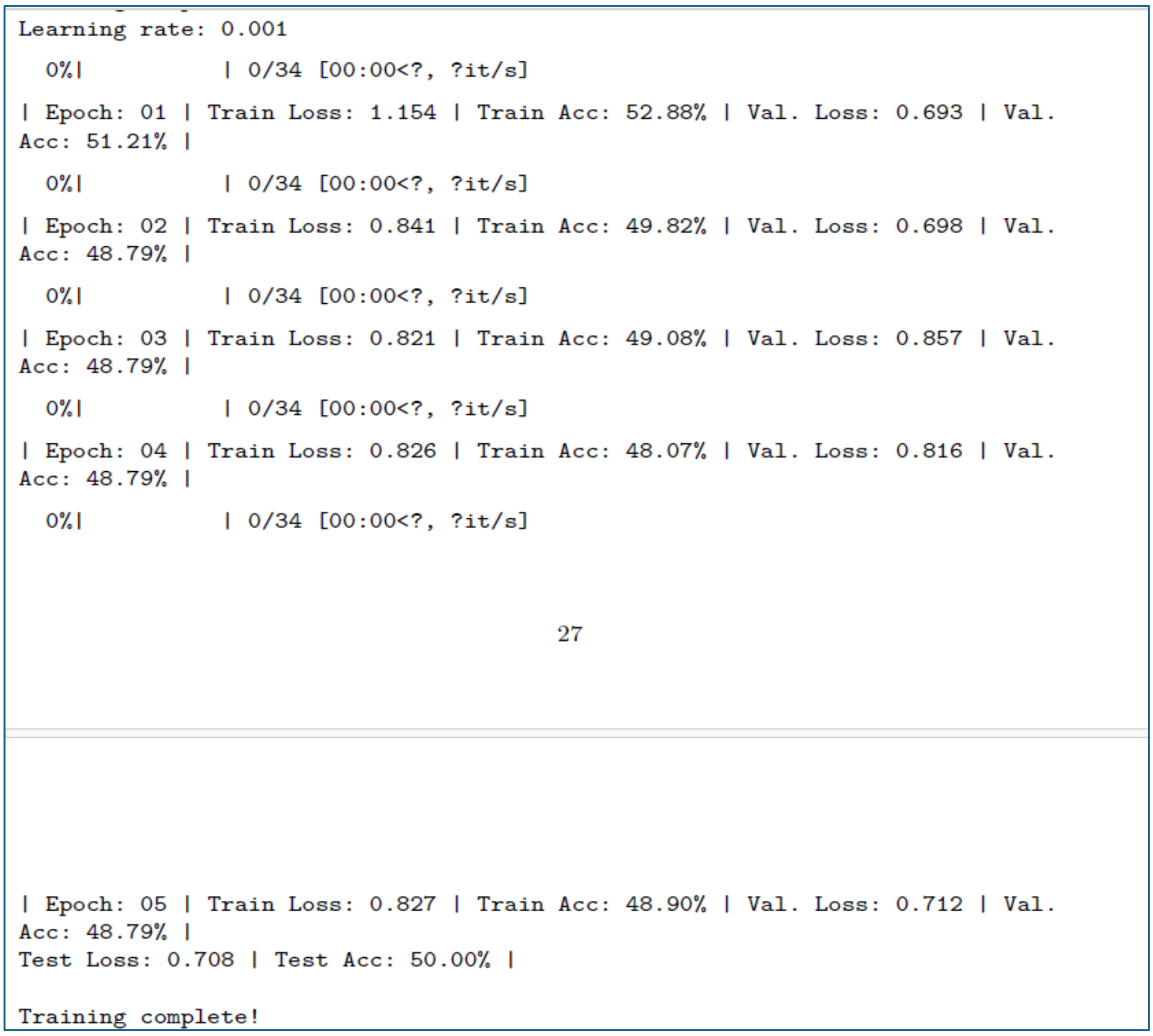


```
learning_rates = [0.01, 1e-3, 1e-5, 2e-05, 5e-06]
learning_curve = {}
```

Partha Sai Madallapalli(pua528)

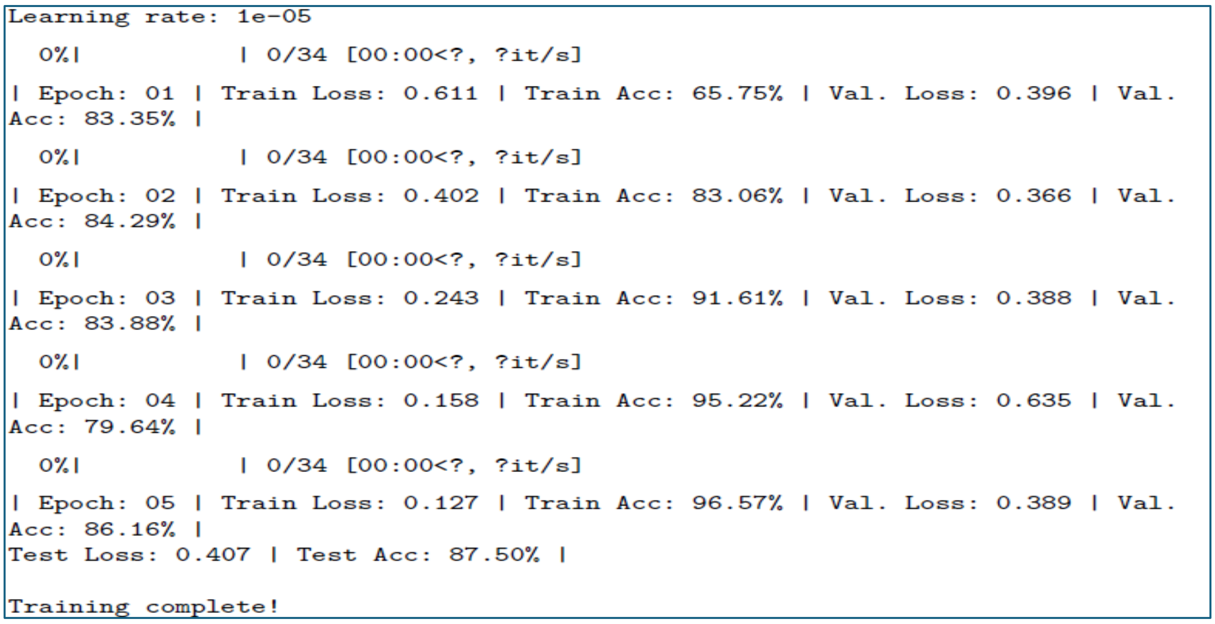


```
Learning rate: 0.01
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 01 | Train Loss: 3.855 | Train Acc: 49.75% | Val. Loss: 0.697 | Val.
Acc: 48.79% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 02 | Train Loss: 2.332 | Train Acc: 47.15% | Val. Loss: 0.822 | Val.
Acc: 48.79% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 03 | Train Loss: 2.358 | Train Acc: 51.65% | Val. Loss: 1.331 | Val.
Acc: 48.79% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 04 | Train Loss: 1.780 | Train Acc: 52.57% | Val. Loss: 2.147 | Val.
Acc: 51.21% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 05 | Train Loss: 1.874 | Train Acc: 52.14% | Val. Loss: 1.078 | Val.
Acc: 48.79% |
Test Loss: 1.056 | Test Acc: 50.00% |

Training complete!
```

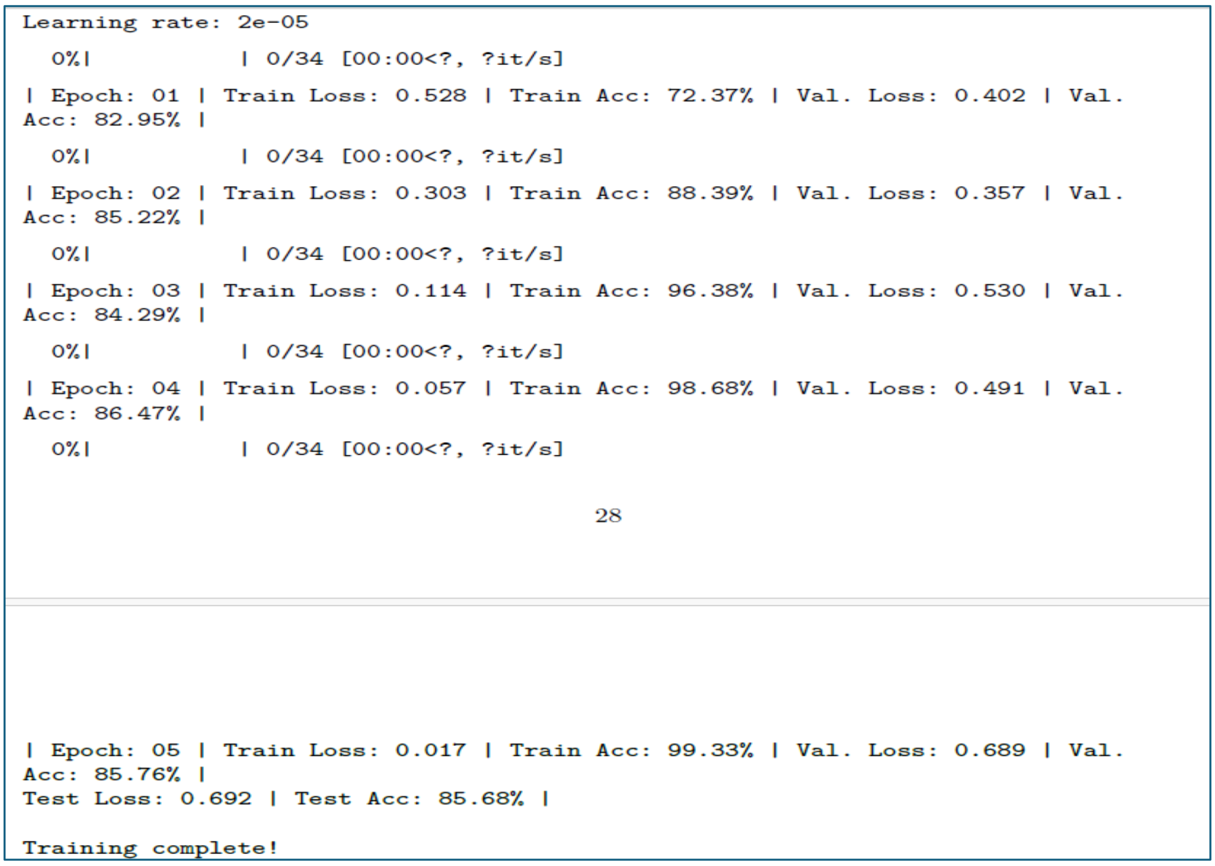


```
Learning rate: 0.001
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 01 | Train Loss: 1.154 | Train Acc: 52.88% | Val. Loss: 0.693 | Val.
Acc: 51.21% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 02 | Train Loss: 0.841 | Train Acc: 49.82% | Val. Loss: 0.698 | Val.
Acc: 48.79% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 03 | Train Loss: 0.821 | Train Acc: 49.08% | Val. Loss: 0.857 | Val.
Acc: 48.79% |
  0%|           | 0/34 [00:00<?, ?it/s]
| Epoch: 04 | Train Loss: 0.826 | Train Acc: 48.07% | Val. Loss: 0.816 | Val.
Acc: 48.79% |
  0%|           | 0/34 [00:00<?, ?it/s]

                                  27

| Epoch: 05 | Train Loss: 0.827 | Train Acc: 48.90% | Val. Loss: 0.712 | Val.
Acc: 48.79% |
Test Loss: 0.708 | Test Acc: 50.00% |

Training complete!
```

Partha Sai Madallapalli(pua528)



```
Learning rate: 1e-05
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 01 | Train Loss: 0.611 | Train Acc: 65.75% | Val. Loss: 0.396 | Val.
Acc: 83.35% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 02 | Train Loss: 0.402 | Train Acc: 83.06% | Val. Loss: 0.366 | Val.
Acc: 84.29% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 03 | Train Loss: 0.243 | Train Acc: 91.61% | Val. Loss: 0.388 | Val.
Acc: 83.88% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 04 | Train Loss: 0.158 | Train Acc: 95.22% | Val. Loss: 0.635 | Val.
Acc: 79.64% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 05 | Train Loss: 0.127 | Train Acc: 96.57% | Val. Loss: 0.389 | Val.
Acc: 86.16% |
Test Loss: 0.407 | Test Acc: 87.50% |

Training complete!
```

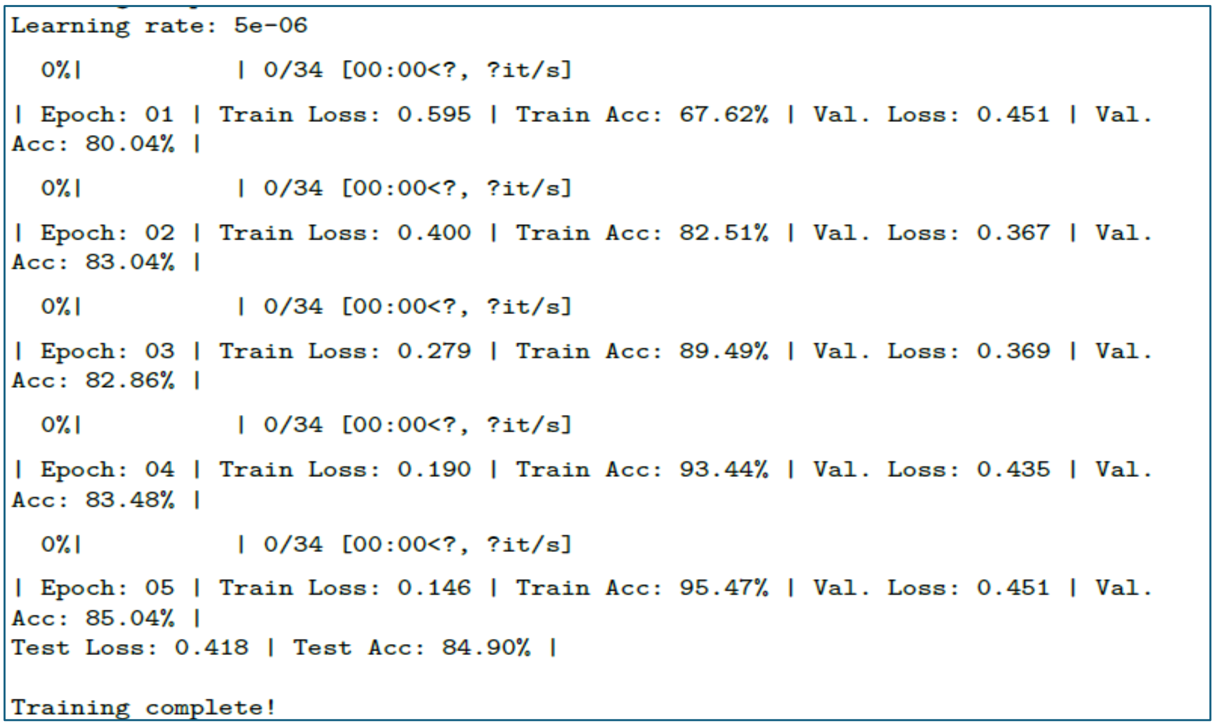


```
Learning rate: 2e-05
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 01 | Train Loss: 0.528 | Train Acc: 72.37% | Val. Loss: 0.402 | Val.
Acc: 82.95% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 02 | Train Loss: 0.303 | Train Acc: 88.39% | Val. Loss: 0.357 | Val.
Acc: 85.22% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 03 | Train Loss: 0.114 | Train Acc: 96.38% | Val. Loss: 0.530 | Val.
Acc: 84.29% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 04 | Train Loss: 0.057 | Train Acc: 98.68% | Val. Loss: 0.491 | Val.
Acc: 86.47% |
  0%|          | 0/34 [00:00<?, ?it/s]
```

28

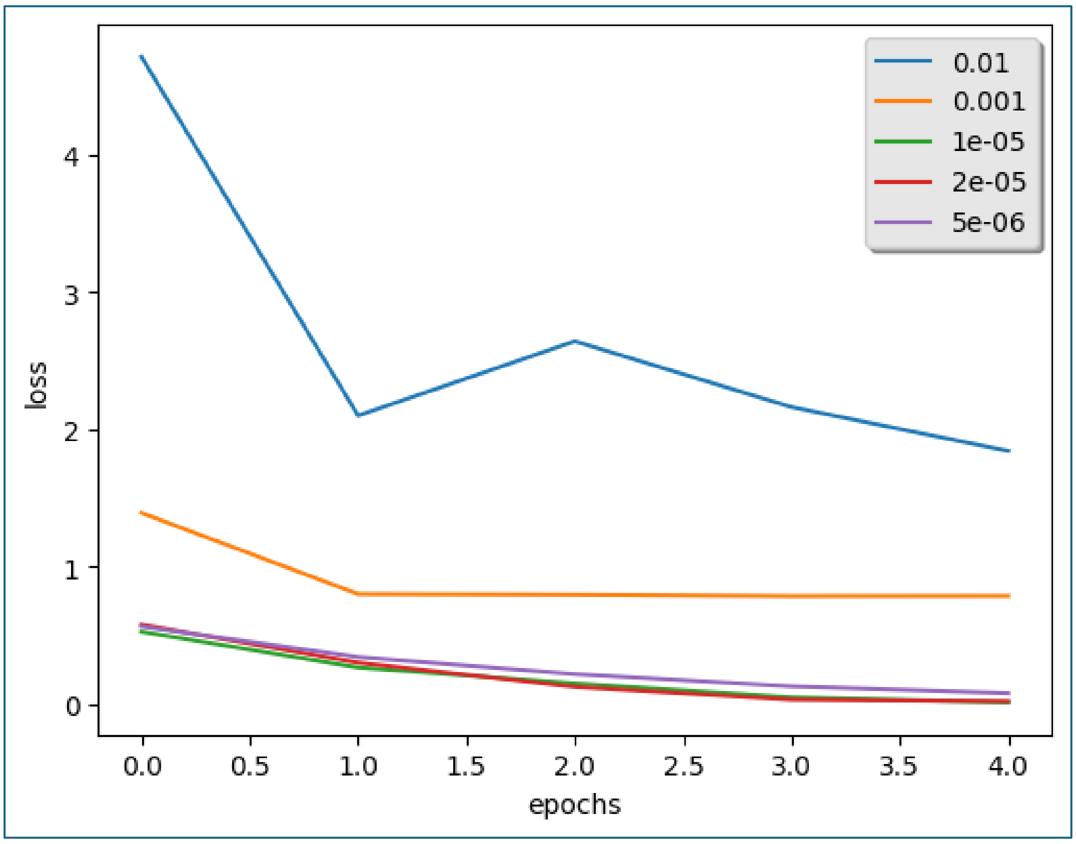
```
| Epoch: 05 | Train Loss: 0.017 | Train Acc: 99.33% | Val. Loss: 0.689 | Val.
Acc: 85.76% |
Test Loss: 0.692 | Test Acc: 85.68% |

Training complete!
```

Partha Sai Madallapalli(pua528)

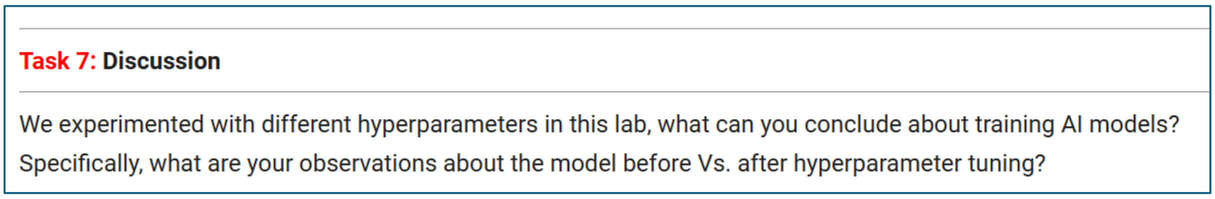


```
Learning rate: 5e-06
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 01 | Train Loss: 0.595 | Train Acc: 67.62% | Val. Loss: 0.451 | Val.
Acc: 80.04% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 02 | Train Loss: 0.400 | Train Acc: 82.51% | Val. Loss: 0.367 | Val.
Acc: 83.04% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 03 | Train Loss: 0.279 | Train Acc: 89.49% | Val. Loss: 0.369 | Val.
Acc: 82.86% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 04 | Train Loss: 0.190 | Train Acc: 93.44% | Val. Loss: 0.435 | Val.
Acc: 83.48% |
  0%|          | 0/34 [00:00<?, ?it/s]
| Epoch: 05 | Train Loss: 0.146 | Train Acc: 95.47% | Val. Loss: 0.451 | Val.
Acc: 85.04% |
Test Loss: 0.418 | Test Acc: 84.90% |

Training complete!
```

Plotting Curves for different learning rate vs training loss:



Interpretation: From the above graph it is understandable that the training loss decrease as the learning rate decreases at 1e-05, also the model is converged as the Training and Test Accuracy are at around 96% and 87% respectively. There is no sign of model overfitting here.



Task 7: Discussion

We experimented with different hyperparameters in this lab, what can you conclude about training AI models? Specifically, what are your observations about the model before Vs. after hyperparameter tuning?

Interpretation:

As experimented with different hyperparameters, I understood that hyperparameter tuning have yielded significant improvements in the model performance as follows:

- **Test Accuracy Improvement: -**

  Before Tuning : Test Accuracy was 85.16%.

  After Tuning : Test Accuracy increased to 87.50% as reflection of substantial enhancements in the model accuracy through fine tuning method.

- **Training Stability and Convergence Speed: -**

  Before Tuning: The model exhibited fluctuations in performance and potentially slower convergence.

  After Tuning: The model performance appeared more stable, with consistent improvements in accuracy over epochs, for instance with the learning rate at 1e-05 the test accuracy increased steadily from 83.35% to 87.50% over 5 epochs, indicating improved stability and convergence.

- **Effectiveness: -** Through hyperparameter tuning there is a significant amount of model performance improvement with respect to tuning on number of epochs and learning rate to substantially enhance the accuracy of the model and get converged.

Hence from the above metrics and findings it is understood that model convergence speed, accuracy, stability all is dependent on the systematic hyperparameter optimization in the training the AI models.