ASSIGNMENT-2.2 REPORT

BY

KURISETI RAVI SRI TEJA
ENTRY NUMBER-2019CS10369

Contents

1 Devanagari-Dataset

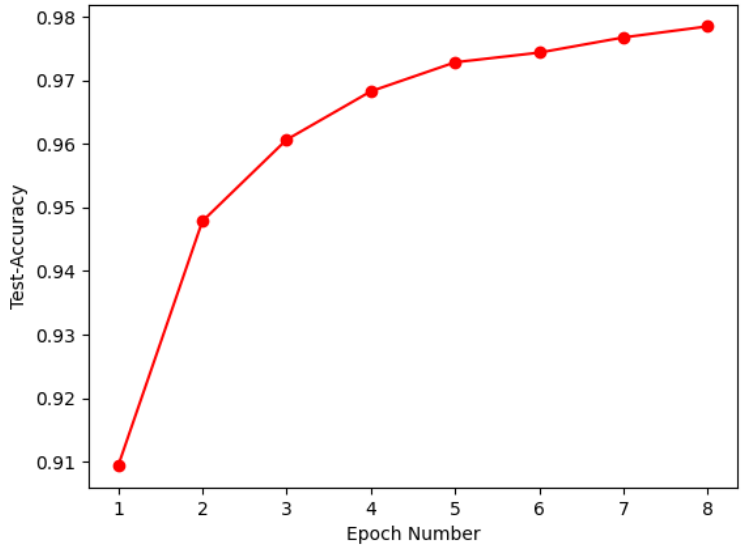
1.1 Results-Obtained

Epoch Number	Training Loss	Training Accuracy
1	1.359	0.913
2	0.331	0.958
3	0.196	0.972
4	0.133	0.981
5	0.099	0.986
6	0.077	0.990
7	0.060	0.992
8	0.047	0.994

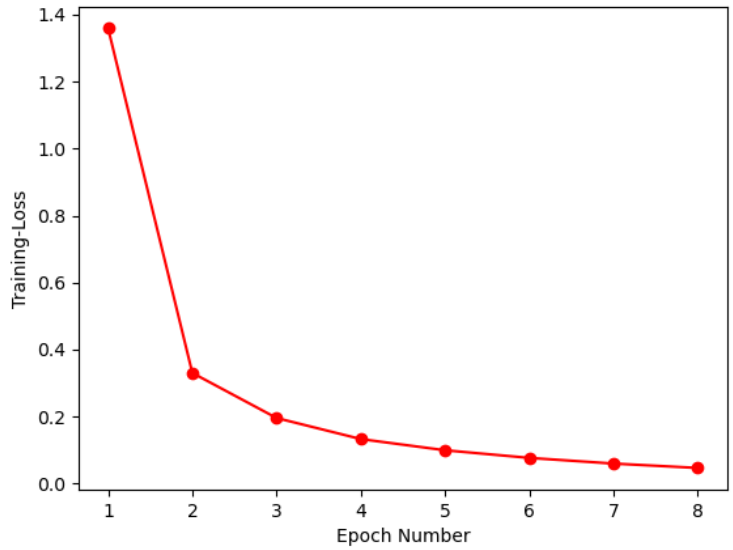
Epoch Number	Public Test-Data Accuracy
1	0.909
2	0.948
3	0.961
4	0.968
5	0.972
6	0.974
7	0.977
8	0.978

1.2 Corresponding Plots

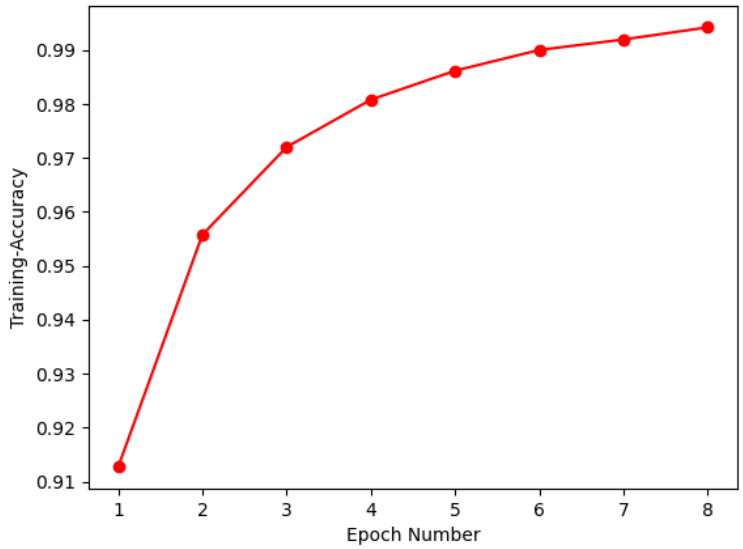
Plot of Test-Accuracy vs Epoch-Number



Plot of Training-Loss vs Epoch-Number



Plot of Training-Accuracy vs Epoch-Number



1.3 Comparision with Simple Neural Network

The CNN takes around 5min 30 sec to run all the 8 epochs on Colab (for me) and produces an accuracy of 97.8% on the public test dataset.The simple neural network with a single fully connected hidden layer took only 3min 55 sec to run all the 8 epochs on Colab (for me) and produces an accuracy of 73.9% on the public test dataset. We see that the CNN produces more accurate results even though it took a little higher time.

Epoch Number	**Public Test-Data Accuracy**
1	0.660
2	0.701
3	0.716
4	0.724
5	0.730
6	0.735
7	0.737
8	0.739

2 CIFAR-10 Dataset with Given Network

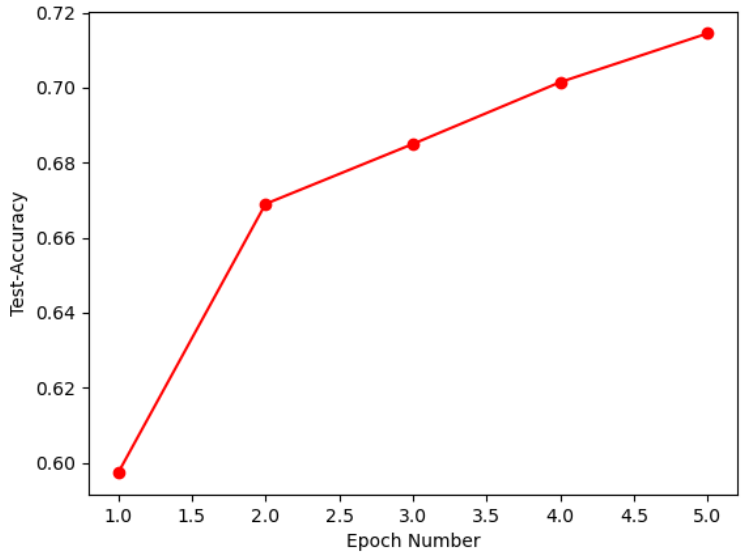
2.1 Results-Obtained

Epoch Number	Training Loss	Training Accuracy
1	1.428	0.613
2	1.052	0.698
3	0.882	0.7505
4	0.751	0.7937
5	0.635	0.8301

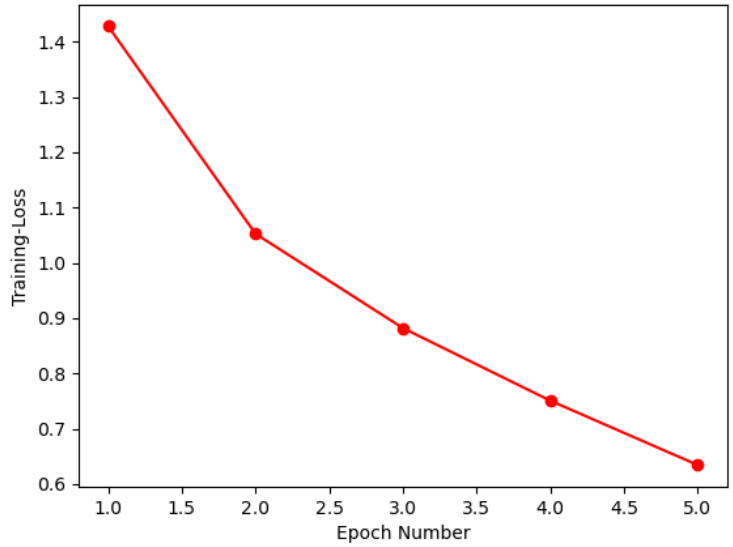
Epoch Number	Public Test-Data Accuracy
1	0.59725
2	0.669
3	0.685
4	0.7015
5	0.7145

2.2 Corresponding Plots

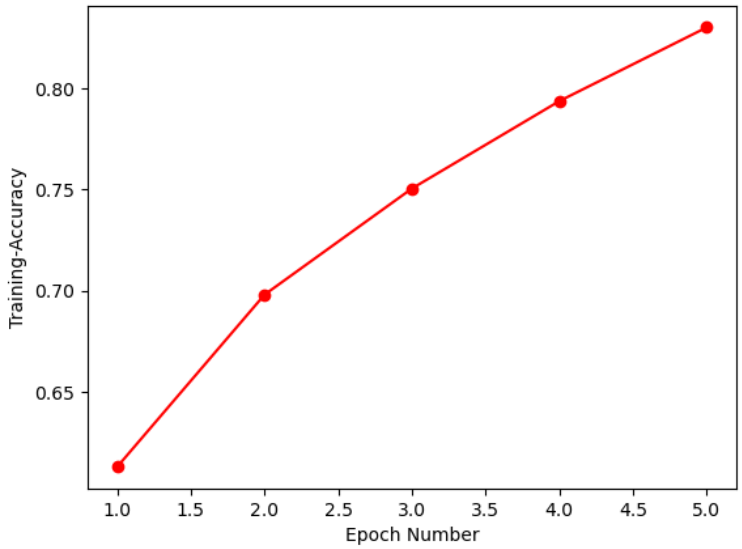
Plot of Test-Accuracy vs Epoch-Number



Plot of Training-Loss vs Epoch-Number



Plot of Training-Accuracy vs Epoch-Number



2.3 Comparision with Simple Neural Network

The CNN takes around 6min 30 sec to run all the 5 epochs on Colab (for me) and produces an accuracy of 71.45% on the public test dataset.The simple neural network with a single fully connected hidden layer takes around 3min 20 sec to run all the 5 epochs on Colab (for me) and produces a meagre accuracy of 40.2% on the public test dataset. We see that the CNN produces more accurate results even though it took a little higher time.

Epoch Number	**Public Test-Data Accuracy**
1	0.366
2	0.389
3	0.394
4	0.398
5	0.402

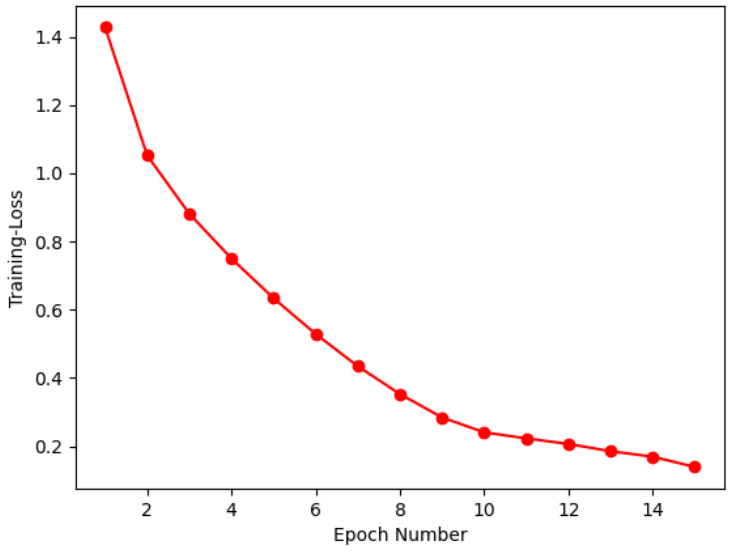
3 CIFAR-10 Dataset Experimentation

3.1 Number of Epochs on given Network

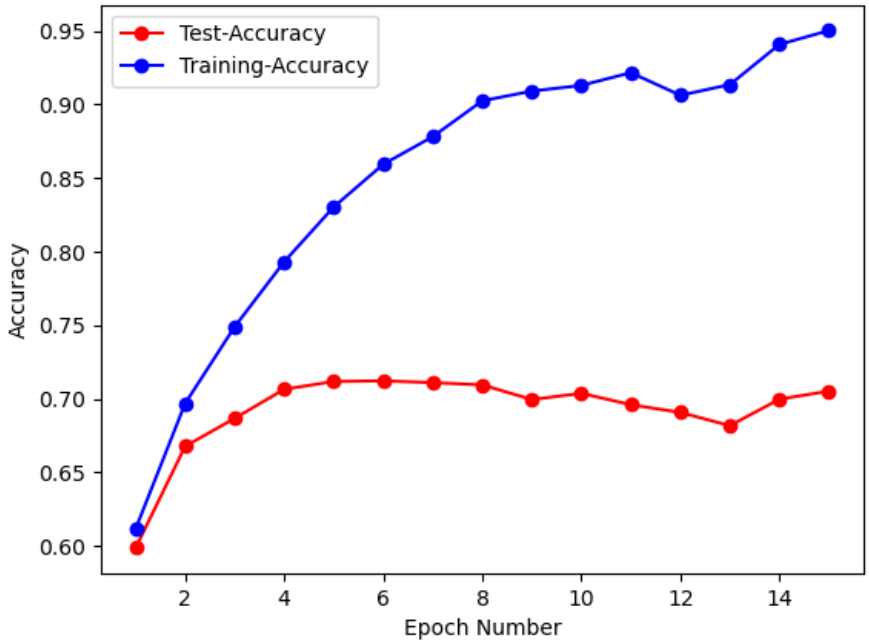
The first thing I had done was to run the given Neural Network upto 15 epochs.Upto 5 epochs,the training accuracy and public test accuracy started to increase. But from then test-accuracy started to decrease even though training-loss was decreasing and training accuracy was increasing i.e., overfitting had started for this network.

3.1.1 Corresponding Plots

Plot of Training-Loss vs Epoch-Number



Plot of Accuracy vs Epoch-Number



3.2 Making the Network Deeper

The second thing I tried is to increase the number of convolution layers in the network.For this I used padding so that the H×W dimensions of image can remain same after convolution and added two extra dropout layers to make sure that over-fitting does not happen.On making the network deeper,the test accuracy got better as the number of epochs increased.In 30 epochs of iteration using Adam Optimizer and deeper network, I could get an accuracy of 83% on the public-test data and this grows if the number of epochs are made higher.

4 Final Architecture Used

4.1 Details Of Final Architecture

I used a 24-layer deep Convolutional Neural Network and a fully connected linear layer at the end.The layers used are:

1. Conv2d(3, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
2. ReLU(inplace=True)
3. BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
4. Conv2d(32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
5. ReLU(inplace=True)
6. BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
7. MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
8. Dropout(p=0.2, inplace=False)
9. Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
10. ReLU(inplace=True)
11. BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)

12. Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
13. ReLU(inplace=True)
14. BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
15. MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
16. Dropout(p=0.3, inplace=False)
17. Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
18. ReLU(inplace=True)
19. BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
20. Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
21. ReLU(inplace=True)
22. BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
23. MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
24. Dropout(p=0.4, inplace=False)
25. Linear(in_features=2048, out_features=10, bias=True)

Number of Features=308394(0.3M)

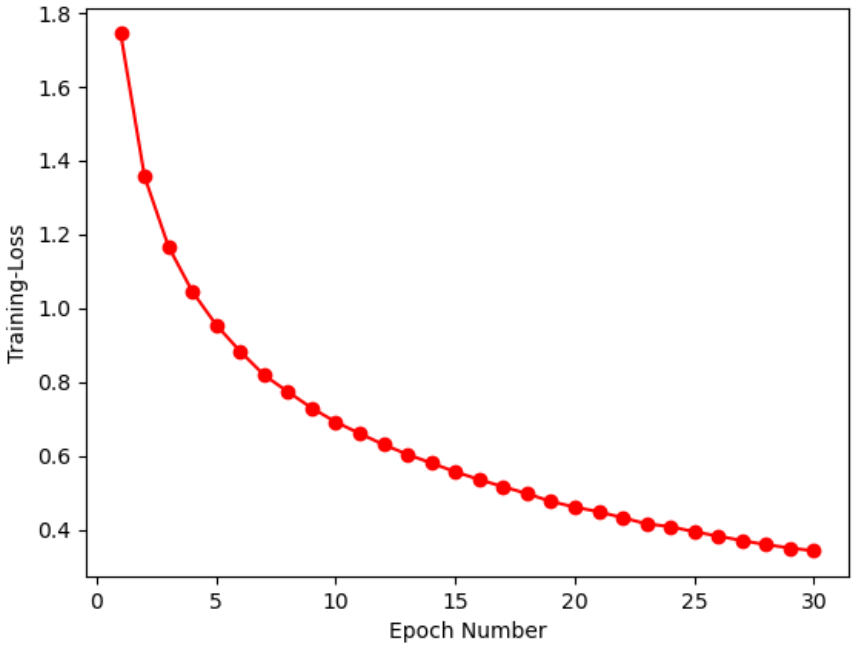
4.2 Results Obtained

Epoch Number	Public Test-Data Accuracy
1	0.485
5	0.652
9	0.745
13	0.7835
17	0.7998
21	0.811
25	0.8232
29	0.8285
30	0.8325

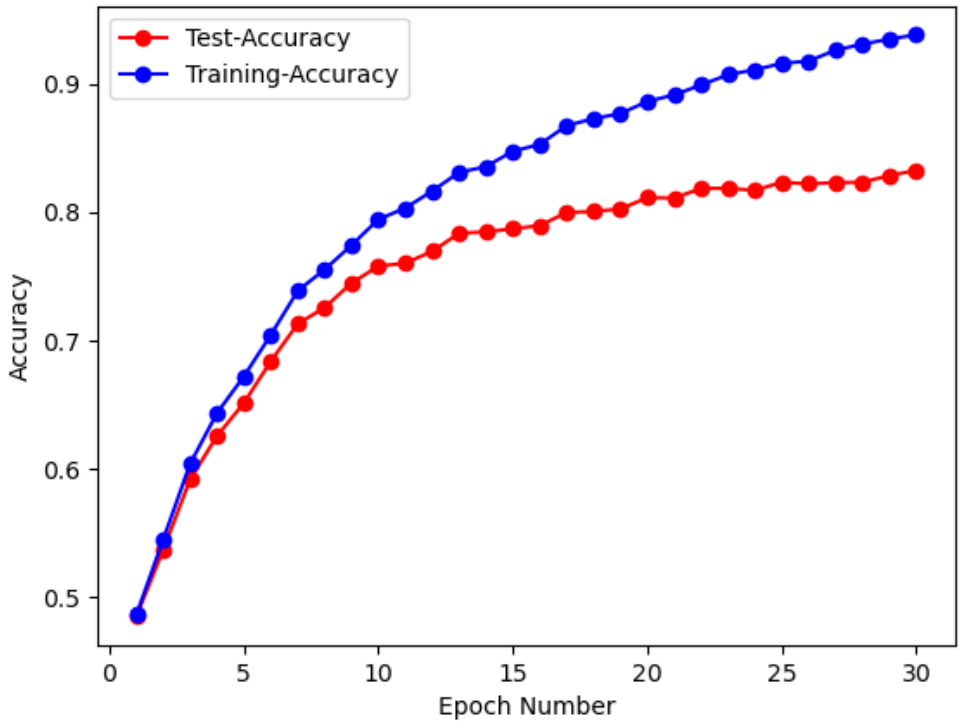
Epoch Number	Training Loss	Training Accuracy
1	1.745	0.490
5	0.953	0.6722
9	0.729	0.7743
13	0.603	0.8310
17	0.516	0.8674
21	0.448	0.8913
25	0.394	0.9155
29	0.350	0.9347
30	0.342	0.9380

4.3 Corresponding Plots

Plot of Training-Loss vs Epoch-Number



Plot of Accuracy vs Epoch-Number



5 Conclusion

The accuracy of model will increase if number of epochs are raised from 30 to 40,50 and even 100.But it takes a lot of time i.e., 1 epoch is taking about 50 seconds on Google Colab.So I couldn't increase the number of epochs past 30.But I would do it personally(independent of assignment to see if better results are possible).