Deep Learning
Minor-1 Report
(M22MA003)

Roll Number - ABC	003
DD/MM/YYYY	21/06/1997
Dataset	CIFAR10
Weight Initialisation	Xavier Normal
Data Augmentation	Rotate by 10 degrees and Gaussian Noise
Pooling Operation	Average Pool
Target Classes	0,2,4,6,8

Question 1 : Train a CNN.

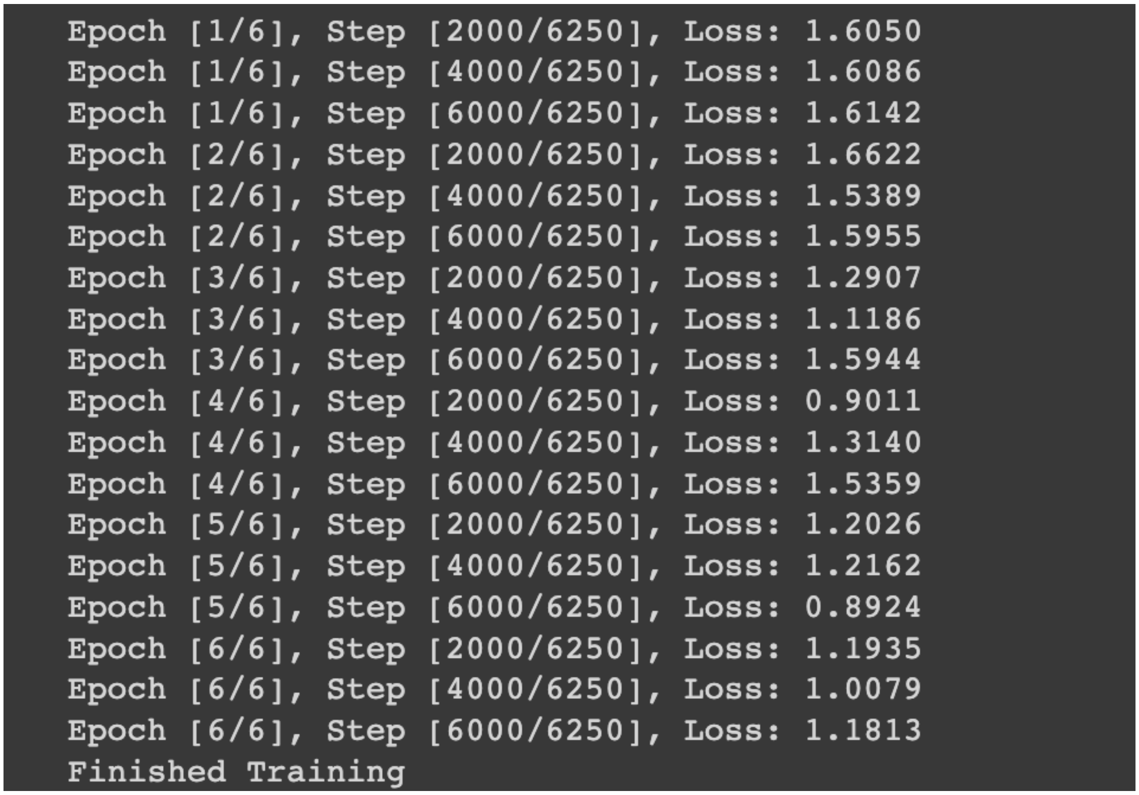
Layers	6 Conv Layers + 1 Pool Layer + 1 FC (512 Nodes)
Loss Function Used	CrossEntropy

Steps Followed:

1. Imported the required libraries.
2. Defined Parameters - number of epochs, batch size, learning rate.
3. Using pytorch datasets to import CIFAR10 data.
4. Transformed the data according to requirements : Rotate by 10 degrees, Add Gaussian Noise.

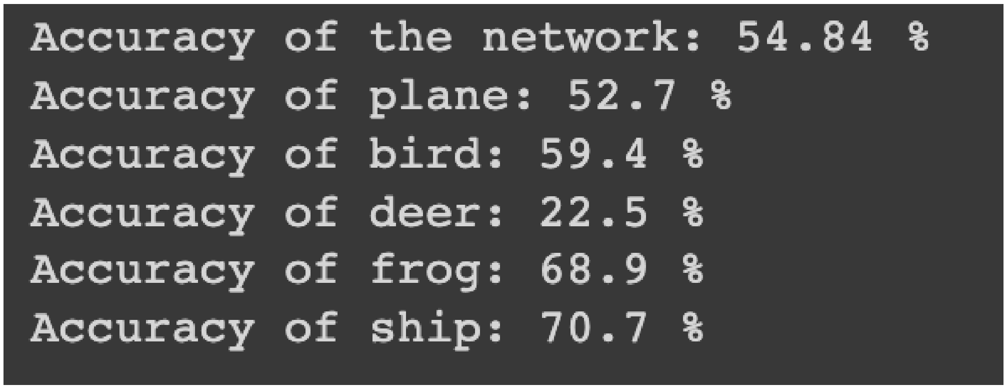
5. Selected the data for required target classes based on actual labels.

6. Defined a CNN with 12 filters in the first layer.
7. Added a total of six Feature Extraction Layers and one pooling layer.
8. Initialized the respective weights of the six layers using Xavier_normal initialisation.
9. Defined two linear layers out of which preceding is Fully Connected Layer and later is output layer.
10. Performed training on the CNN model with 5 epochs. Below are the results.



```
Epoch [1/6], Step [2000/6250], Loss: 1.6050
Epoch [1/6], Step [4000/6250], Loss: 1.6086
Epoch [1/6], Step [6000/6250], Loss: 1.6142
Epoch [2/6], Step [2000/6250], Loss: 1.6622
Epoch [2/6], Step [4000/6250], Loss: 1.5389
Epoch [2/6], Step [6000/6250], Loss: 1.5955
Epoch [3/6], Step [2000/6250], Loss: 1.2907
Epoch [3/6], Step [4000/6250], Loss: 1.1186
Epoch [3/6], Step [6000/6250], Loss: 1.5944
Epoch [4/6], Step [2000/6250], Loss: 0.9011
Epoch [4/6], Step [4000/6250], Loss: 1.3140
Epoch [4/6], Step [6000/6250], Loss: 1.5359
Epoch [5/6], Step [2000/6250], Loss: 1.2026
Epoch [5/6], Step [4000/6250], Loss: 1.2162
Epoch [5/6], Step [6000/6250], Loss: 0.8924
Epoch [6/6], Step [2000/6250], Loss: 1.1935
Epoch [6/6], Step [4000/6250], Loss: 1.0079
Epoch [6/6], Step [6000/6250], Loss: 1.1813
Finished Training
```

11. Performed training on the CNN model. Below are the results.



```
Accuracy of the network: 54.84 %
Accuracy of plane: 52.7 %
Accuracy of bird: 59.4 %
Accuracy of deer: 22.5 %
Accuracy of frog: 68.9 %
Accuracy of ship: 70.7 %
```

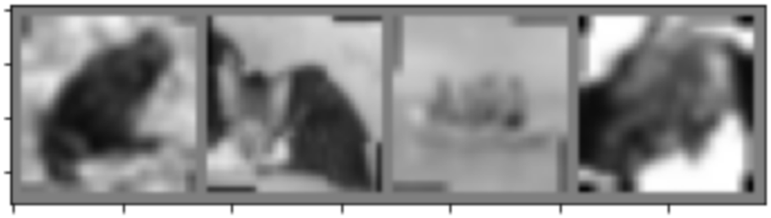
Colab Link Question 1

Question 2 : Train an AutoEncoder.

Layers	3 AE Layers + 1 FC (512 Nodes)
Loss Functions Used	CrossEntropy(Classification) + MSE(Decoder)

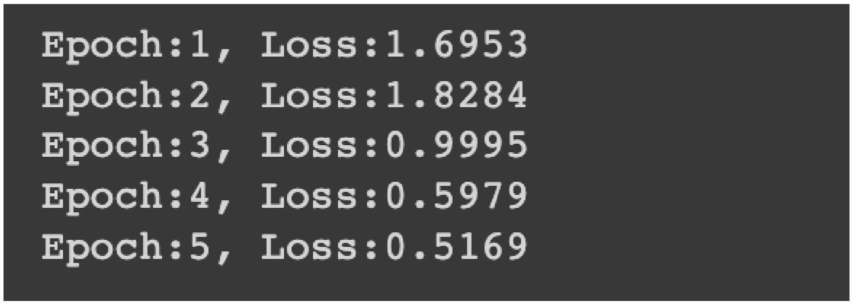
Steps Followed:

1. Imported the required libraries.
2. Defined Parameters - number of epochs, batch size, learning rate.
3. Using pytorch datasets to import CIFAR10 data.
4. Transformed the data according to requirements : Rotate by 10 degrees, Add Gaussian Noise.



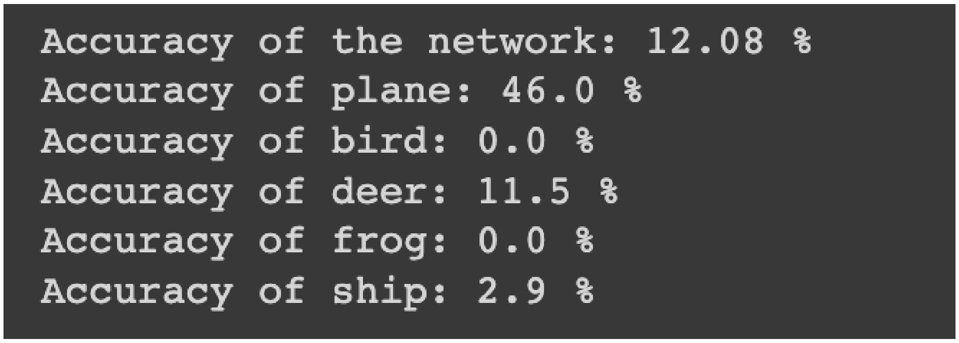
5. Selected the data for required target classes based on actual labels.
6. Defined an AutoEncoder with 3 layers in Encoder and 3 layers in Decoder.
7. Added a fully connected layer with 512 nodes.
8. Forward function includes backward propagation of loss containing two quantities. **One loss is recorded from the Classification layer and a second loss is recorded from the Decoder layer. The summation of the two is considered while performing backpropagation.**

9. Performed training on the AE model. Below are the results.



```
Epoch:1, Loss:1.6953
Epoch:2, Loss:1.8284
Epoch:3, Loss:0.9995
Epoch:4, Loss:0.5979
Epoch:5, Loss:0.5169
```

10. Performed testing on the AE mode. Below are the results.



```
Accuracy of the network: 12.08 %
Accuracy of plane: 46.0 %
Accuracy of bird: 0.0 %
Accuracy of deer: 11.5 %
Accuracy of frog: 0.0 %
Accuracy of ship: 2.9 %
```

Comparison between CNN and AutoEncoder :-

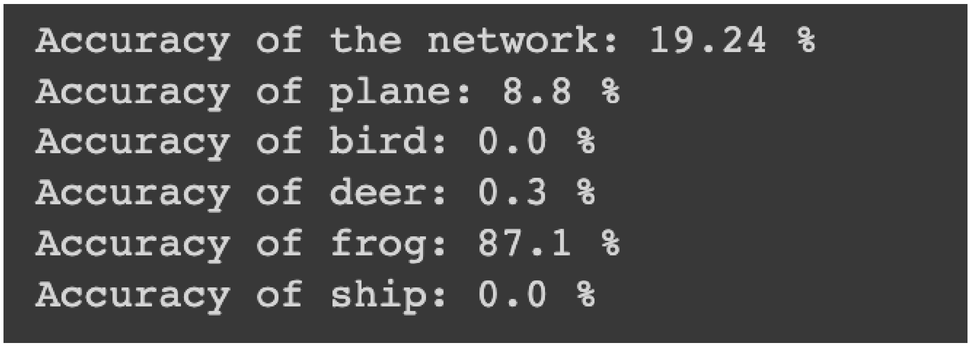
Loss Comparison with five epochs:

A. For the first two epochs, the loss is nearly equal.
B. For the later epochs, it was observed that the loss came down for both CNN and Autoencoder but with a relatively higher good rate for Autoencoder.

Accuracy Comparison for Testing Data:

A. Accuracy of AutoEncoder turns out to be not good.
B. Accuracy of CNN was 54.8% whereas the accuracy of AutoEncoder is 12.08%.

Dropout is used as Regularization to work on the accuracy of AutoEncoder. After introducing dropout in the model, the test accuracy rises to 19%.
Below are the results:-



```
Accuracy of the network: 19.24 %
Accuracy of plane: 8.8 %
Accuracy of bird: 0.0 %
Accuracy of deer: 0.3 %
Accuracy of frog: 87.1 %
Accuracy of ship: 0.0 %
```

This would need further tuning to obtain the accuracy of Autoencoder at an acceptable level.

Colab Link Question 2

References:

https://pytorch.org/docs/stable/nn.init.html
https://www.youtube.com/watch?v=pDdP0TFzsoQ
https://www.youtube.com/watch?v=zp8clK9yCro
https://wandb.ai/authors/ayusht/reports/Implementing-Dropout-in-PyTorch-With-Example--VmlldzoxNTgwOTE