Nirbhay Sharma (B19CSE114)

Deep Learning - Assignment - 1

1. **Neural Network from scratch**

   a. the entries are of the form (lr,epochs,activaiton_function)

Parameters	TrainAcc	TestAcc
(0.01, 20, sigmoid)	0.96	0.92
(0.01, 25, sigmoid)	0.98	0.96
(0.01, 30, sigmoid)	0.98	0.94
(0.001, 20, sigmoid)	0.69	0.58
(0.001, 25, sigmoid)	0.72	0.66
(0.001, 30, sigmoid)	0.94	0.88
(0.0001, 20, sigmoid)	0.70	0.58
(0.0001, 25, sigmoid)	0.55	0.60
(0.0001, 30, sigmoid)	0.33	0.34
(0.01, 20, tanh)	0.68	0.66
(0.01, 25, tanh)	0.32	0.24
(0.01, 30, tanh)	0.90	0.66
(0.001, 20, tanh)	0.98	0.94
(0.001, 25, tanh)	0.98	0.86
(0.001, 30, tanh)	0.96	0.92
(0.0001, 20, tanh)	0.99	0.94
(0.0001, 25, tanh)	0.71	0.70
(0.0001, 30, tanh)	0.90	0.88

   b. from the above tables we can infer many things like some of the hyperparameters are such that the model overfits so the training accuracy is too high (~99) but test accuracy is low and some of the model are also under fitting like giving accuracies like 33% etc., one posible reason for that could be weight initialization which are completely random.

   c. from the above experiments the best hyperparameter is (0.01, 25, sigmoid) with train and test accuracies are 0.98,0.96 respectively and another best hyper parameter is (0.0001, 20, tanh) with train and test accuracies as 0.99, 0.94 respectively
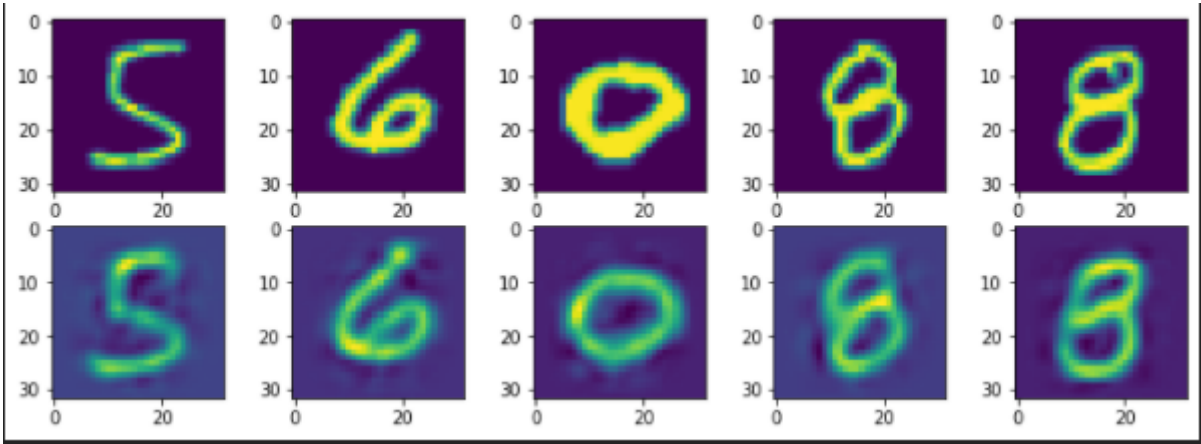
d. below are loss and accuracy curve for the best hyperparameters (rest other are included in .ipynb file)
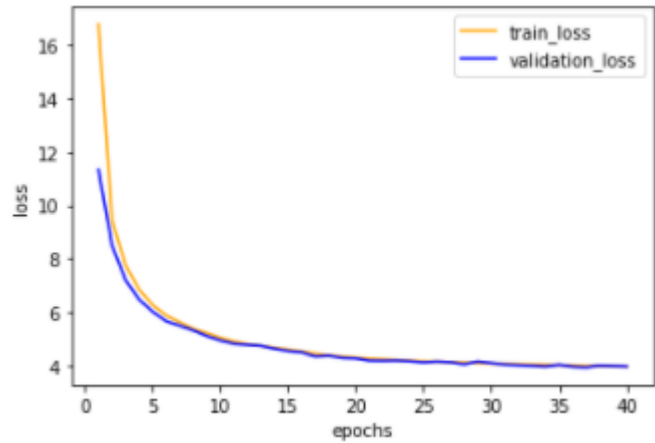
Hyperparameter	loss_curve	accuracy_curve
(0.01, 25, sigmoid)		
(0.0001, 20, tanh)		
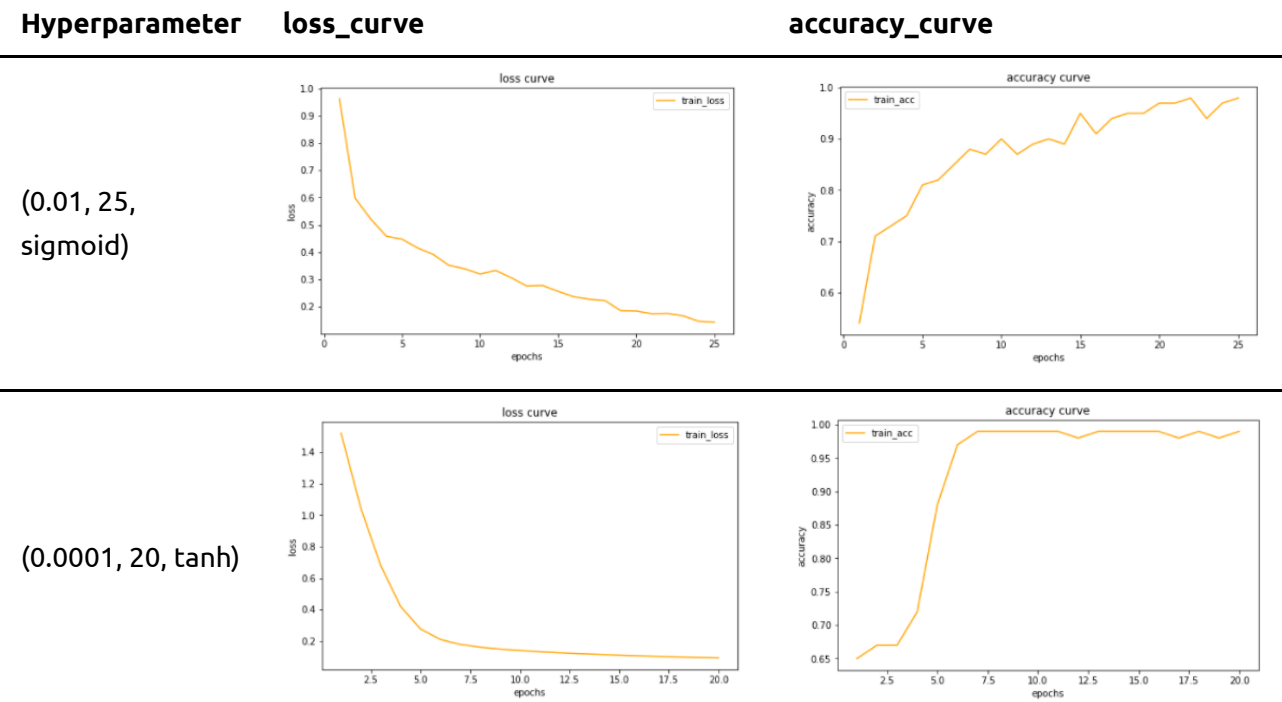
2. **Denoising autoencoder**

Results for Denoising autoencoder (using fully connected network)

before training

after training
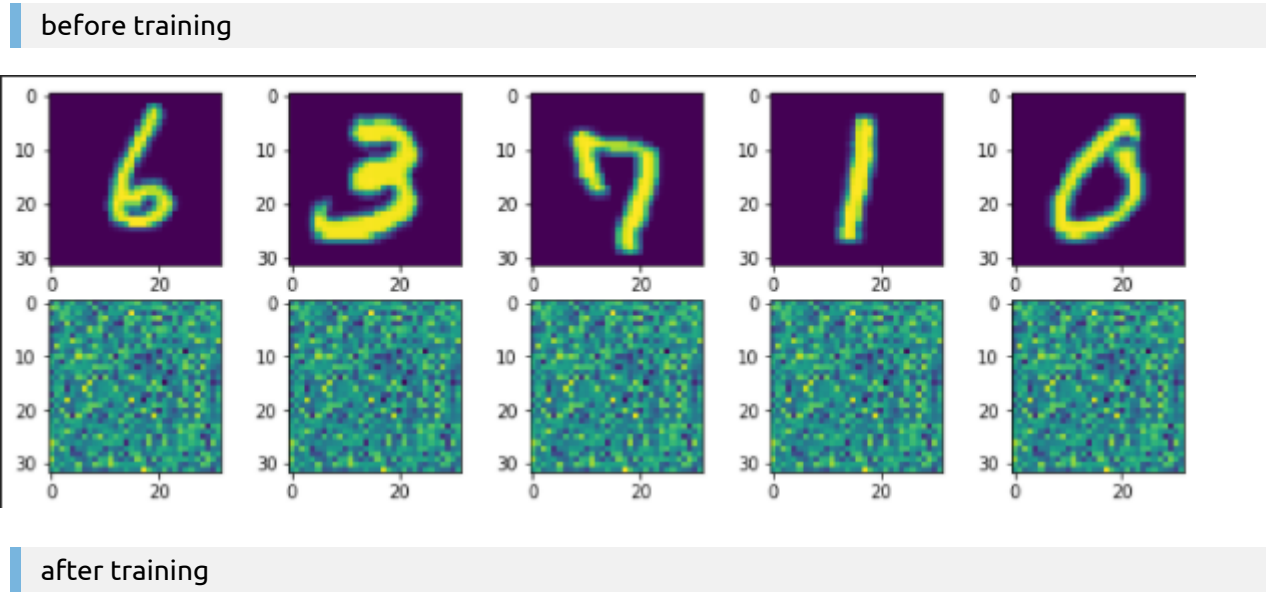
> loss_function

Results of using 1 fc layer and 2 fc layer

Accuracy	fc = 1	fc = 2
Train	0.979	0.988
Validation	0.969	0.965
Test	0.973	0.973

conclusion - we can see that the model with (fc = 2) is trying to overfit as compared to (fc = 1) but by applying both the layers the accuracy on training and testing set is almost similar. The results for loss, accuracy, roc_curves are shown in below tables
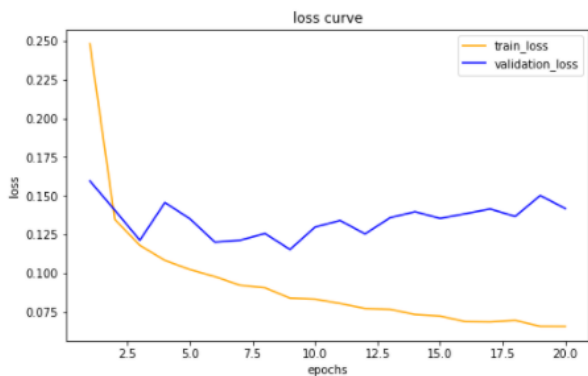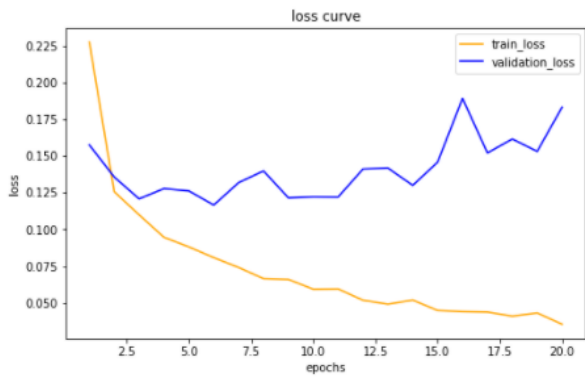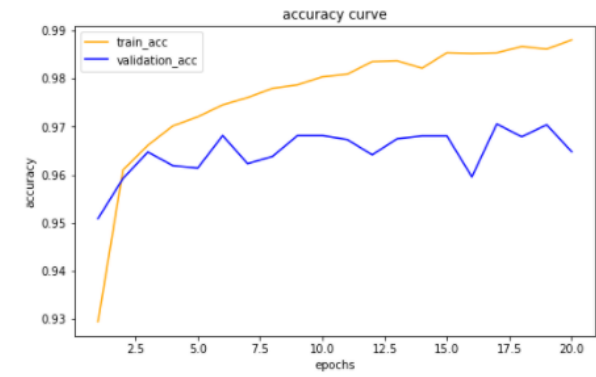
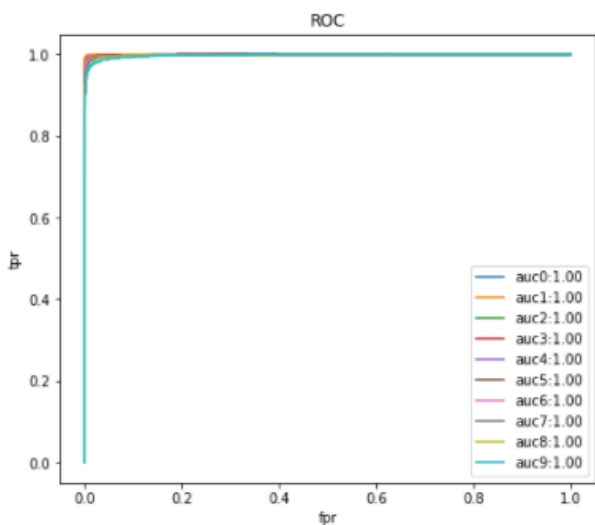**loss (fc = 1)**   **accuracy (fc = 1)**
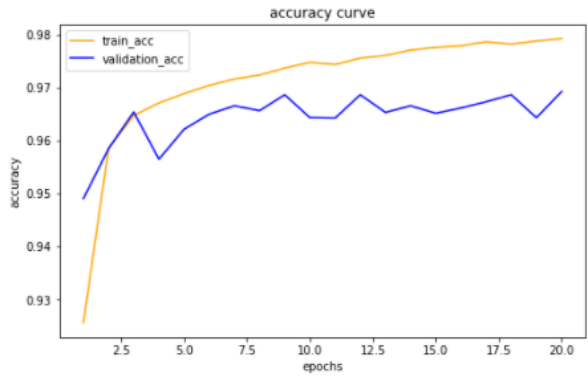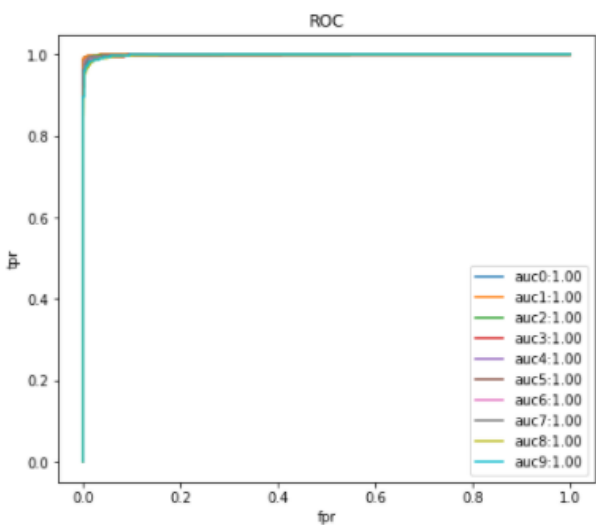
loss (fc = 2)

accuracy (fc = 2)

roc_curve (fc = 1)

roc_curve (fc = 2)

3. **Conv layers**

   a. below table is showing us the analysis for including and excluding various layers

Accuracy	network-1	network-2	network-3
Train	0.732	0.704	0.898
Validation	0.672	0.664	0.677
Test	0.67	0.66	0.68

from the above table we can infer that while we have removed the fc layers from network-1 and network-2 then the accuracies were not much but when we include fc layers then the clear overfitting is visible since fc layers will try to learn more detailed features and hence they end up in overfitting but we can also see that it also gives the best test accuracy among all the three networks
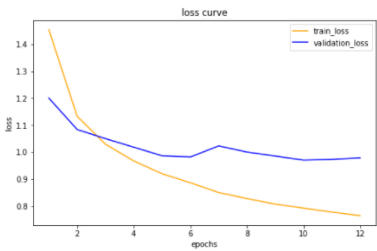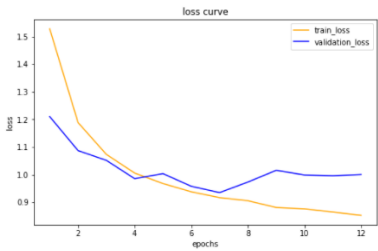
> Loss curves

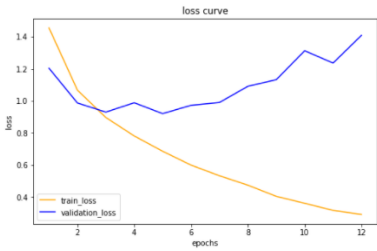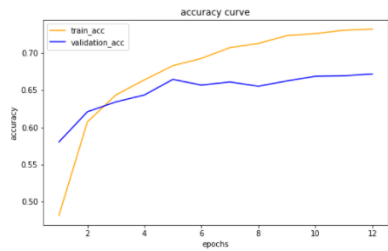**network-1**   **network-2**   **network-3**
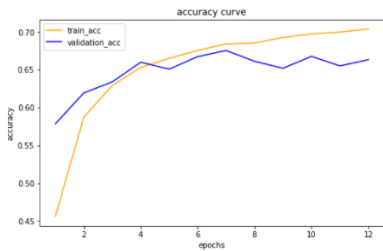
network-1	network-2	network-3

Accuracy curves

network-1	network-2	network-3
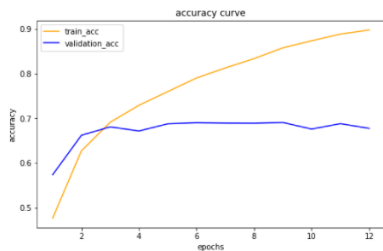
Roc curves
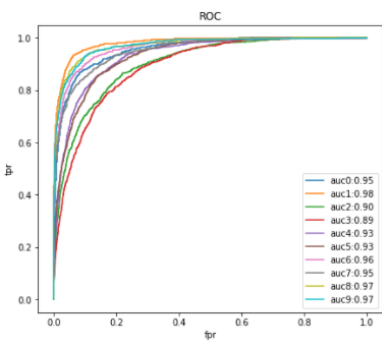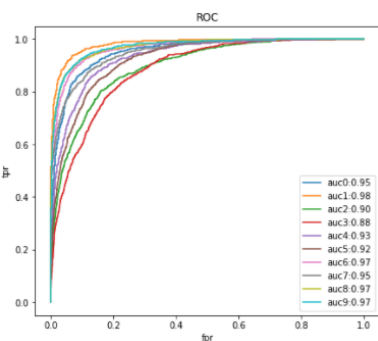
network-1	network-2	network-3
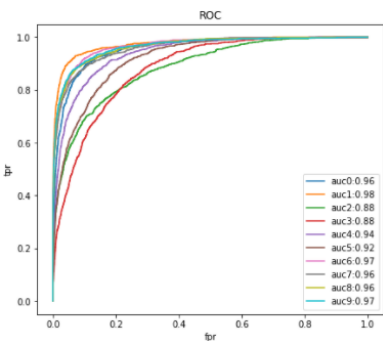
b. Pooling vs accuracy for all the three networks

Accuracy	pooling=2	pooling=3	pooling=4
NETWORK-1			
Train	0.732	0.652	0.631
Validation	0.672	0.618	0.612
Test	0.67	0.62	0.61
NETWORK-2			
Train	0.704	0.657	0.517
Validation	0.664	0.647	0.525
Test	0.66	0.64	0.52

Accuracy	pooling=2	pooling=3	pooling=4
NETWORK-3			
Train	0.898	0.709	0.642
Validation	0.677	0.632	0.612
Test	0.68	0.62	0.61

c. Theoritically effect of presence of more than one fully connected layers depends totally on the dataset and the weights initalized at the time of training but according to the above results we can observe that fc layers should not be added blindly as they may start memorizing the inputs and leads to overfitting but at the same time in cifar10 dataset they are giving the best test accuracy.

Accuracy	pooling=2	pooling=3	pooling=4
NETWORK-3			