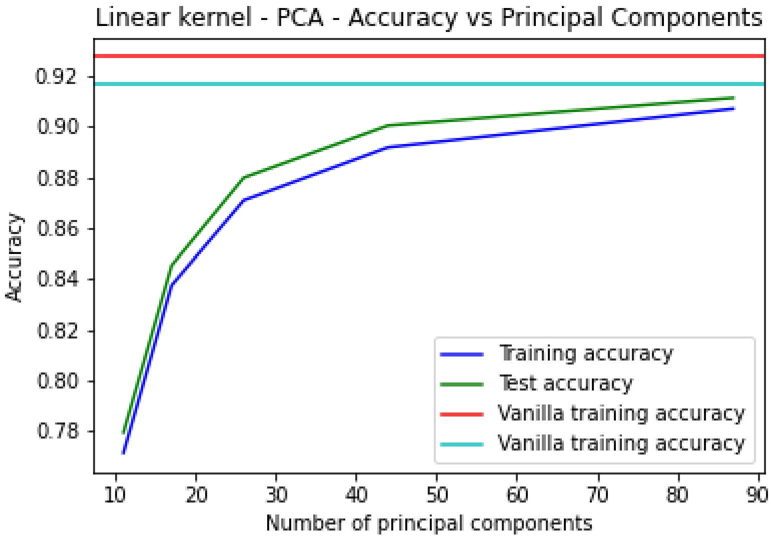
CMSC828C: Project 2

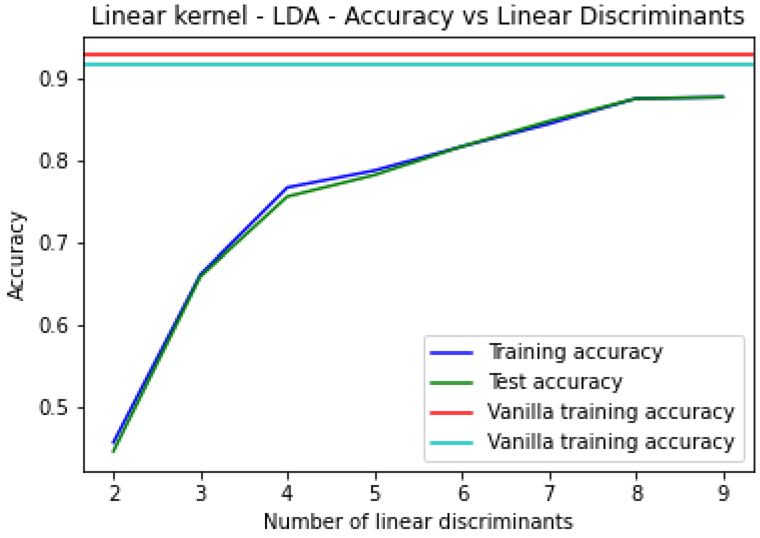
SVM

The number of components for PCA were computed based on the variance. In the below plot, there are five number of components that represents 50%, 60%, 70%, 80%, and 90% of the total variance of the data.

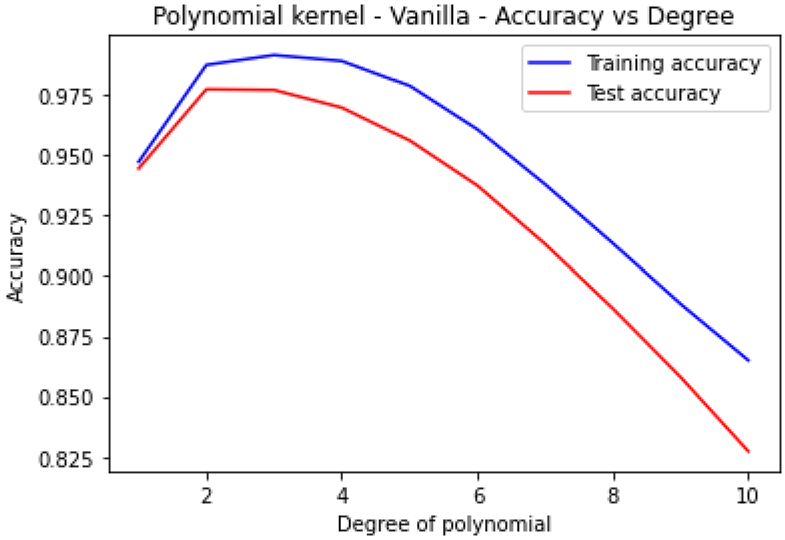
The vanilla training and testing accuracy are also computed. "**Light blue**" line is the vanilla testing accuracy and the plot has a typo.

Linear kernel:



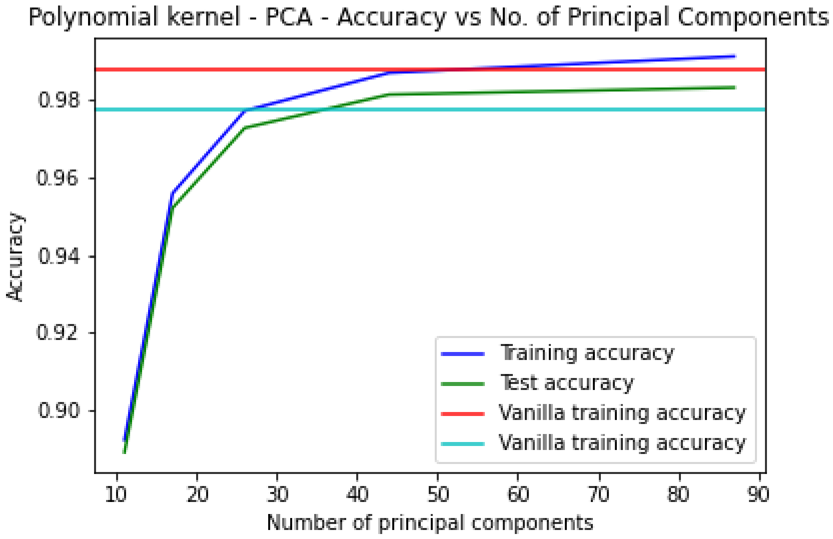


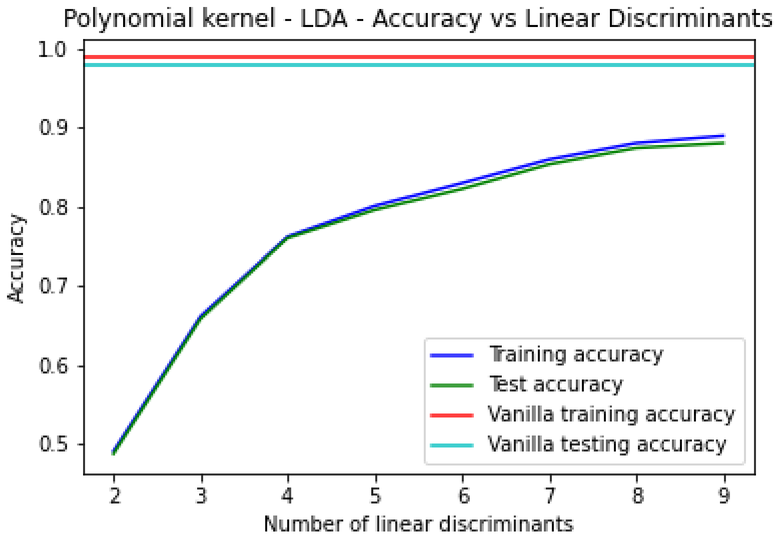
Polynomial kernel:



The accuracy of the polynomial kernel for various degrees were obtained on the plain data as can be seen the above graph. The maximum accuracy of the polynomial kernel was obtained for degree 2.

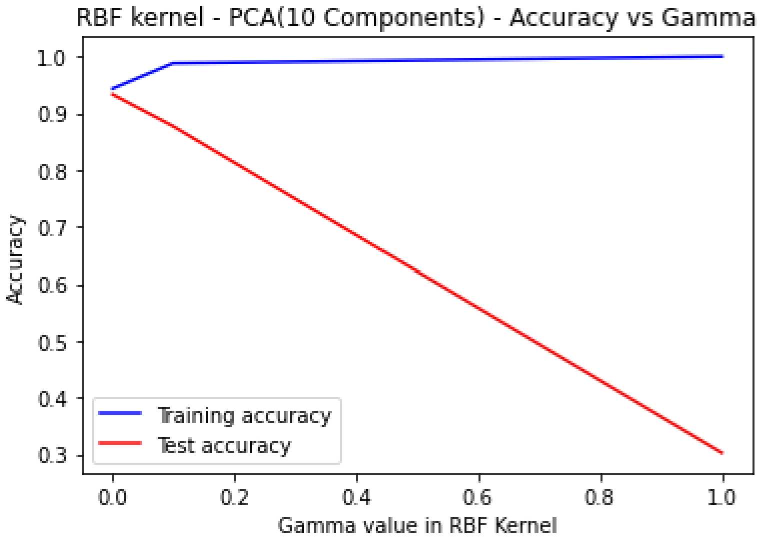
For the below, the degree of the polynomial is taken as 2.





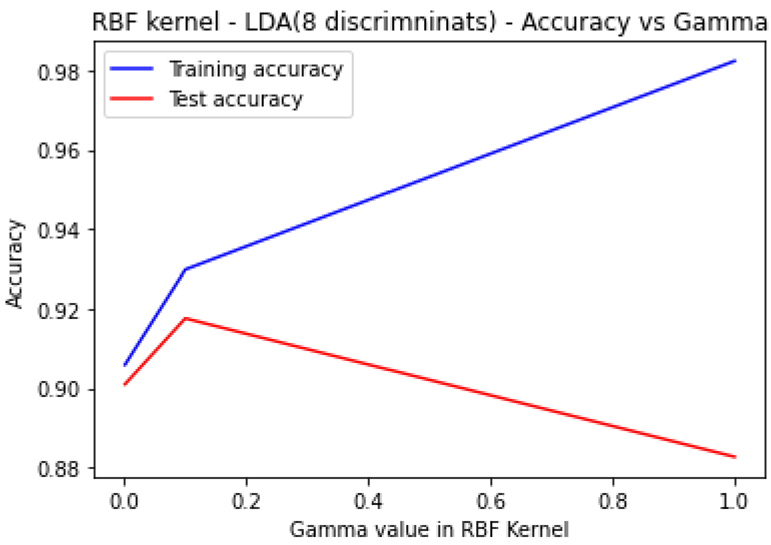
RBF Kernel:

Experiments were not performed on various number of principal components and number of linear discriminants due to high computation time. The number of principal components and linear discriminants were fixed and taken as 10 and 8 respectively.



.

Best value of gamma for the PCA with 10 components is 0.01



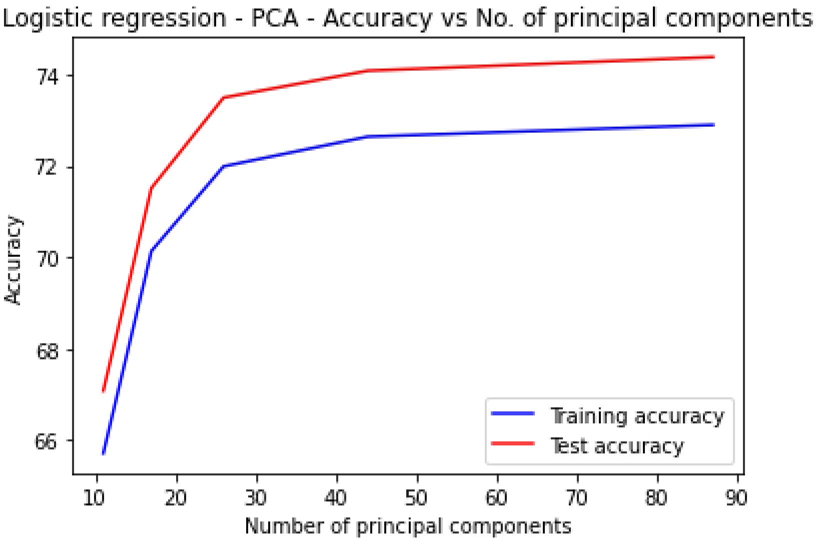
Best value of gamma for LDA with 8 components is 0.01

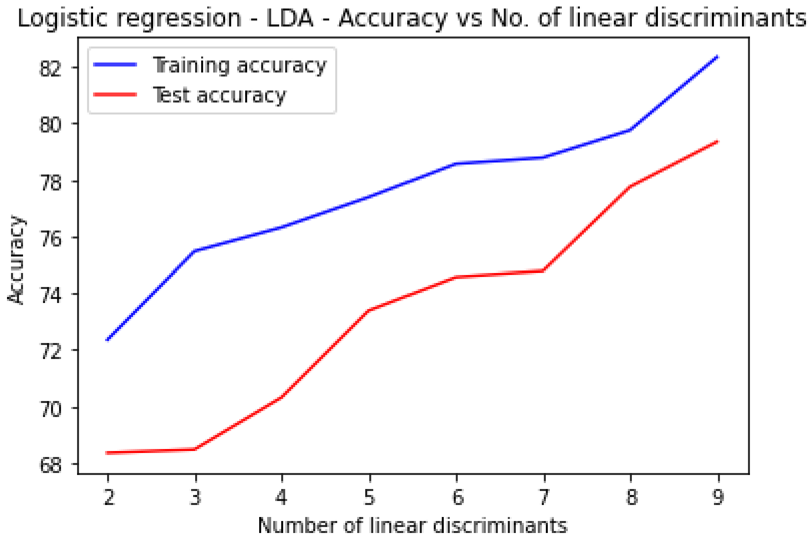
Results:

RBF Kernel achieves better performance followed by the polynomial kernel and the linear kernel.

CMSC828C: Project 2

Logistic regression:





Results:

The performance of the logistic regression is poor than the SVM kernels.

CMSC828C: Project 2

Deep Learning:

Framework: PyTorch

Neural Network architecture: LeNet5

Optimizer: Adam

Learning rate: 0.001

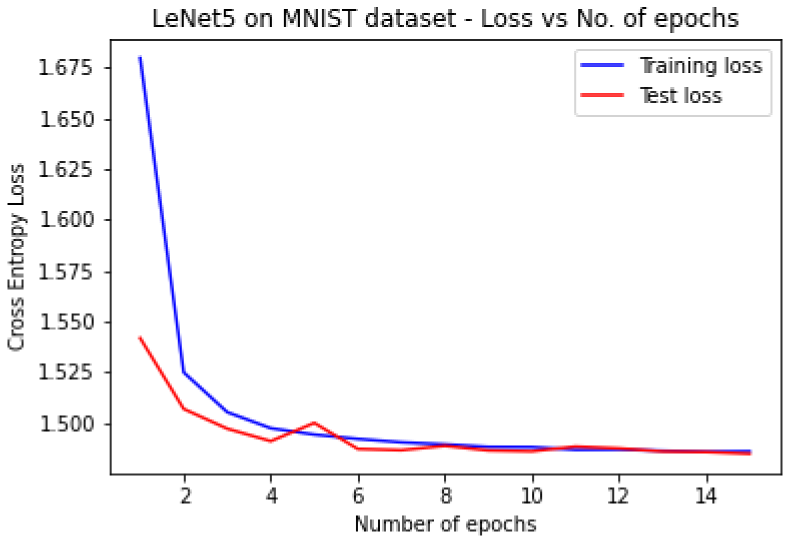
Weight decay rate: 0.001

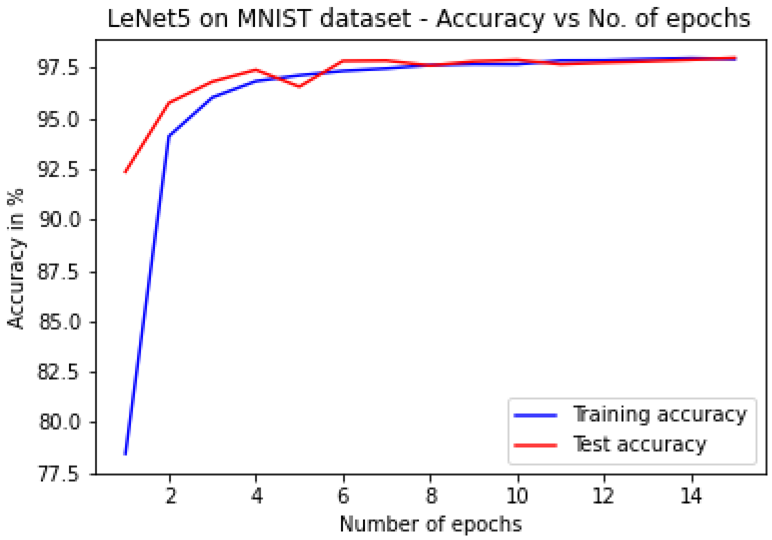
Loss: Cross Entropy loss

Number of epochs = 15

Batch size = 64

Images were rescaled to size 32 x 32





CMSC828C: Project 2

The accuracy of the model on the training and the test set increases with the number of epochs. The maximum training and test accuracy obtained is about 97.5%.

Similarly, the cross-entropy loss for the training and the test set decreases with the number of epochs.

CMSC828C: Project 2

Part 2:

Transfer Learning:

Simple CNN

Framework: PyTorch

Architecture: 5 convolutional layers followed by 2 fully connected layers

Batch normalization, ReLu, maxpooling, and dropout with a probability of 0.5 was performed at the output of every layer.

Optimizer: Adam

Learning rate: 0.001

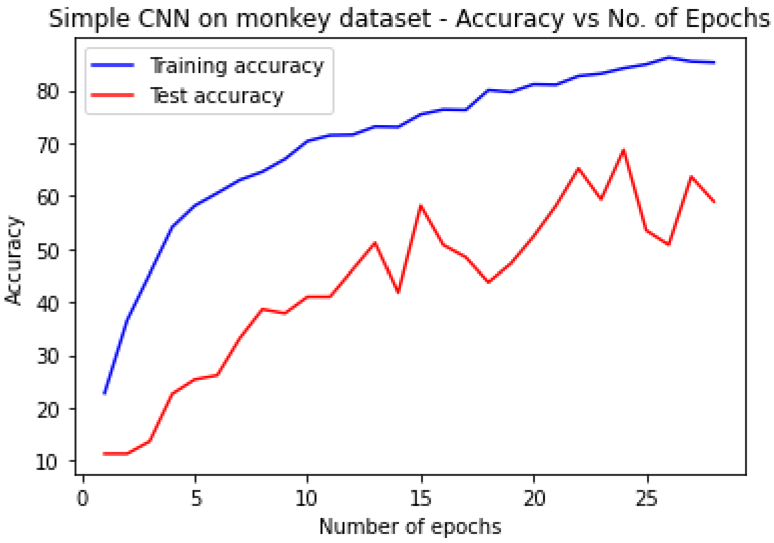
Weight decay rate: 0.001

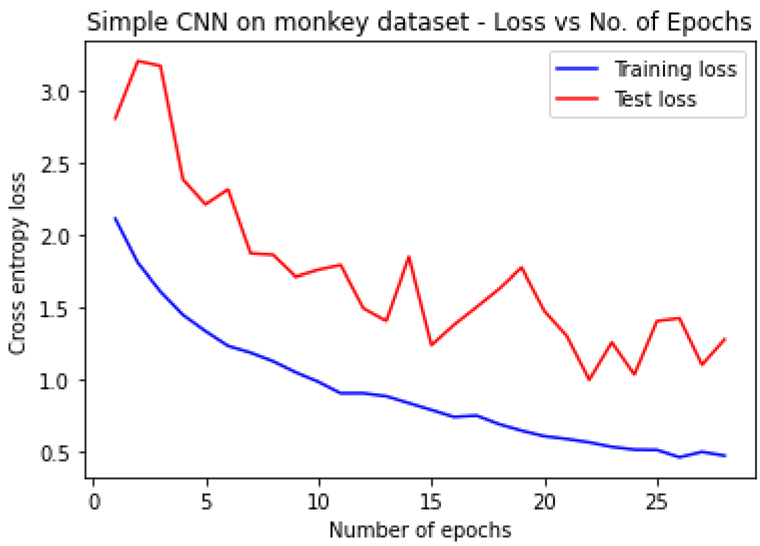
Loss: Cross Entropy loss

Number of epochs = 28

Batch size = 64

Data augmentation: Image were randomly horizontally flipped and center cropped.





The maximum accuracy obtained on the training set is about 84% and on that on the test set is about 70%.

VGG16:

For transfer learning, VGG16 model was used. The weights in the feature extractor or the convolutional layers were kept fixed and only the weights involved in the fully connected layers were optimized using mini-batch gradient descent.

Architecture: VGG16

CMSC828C: Project 2

Optimizer: Adam

Learning rate: 0.001

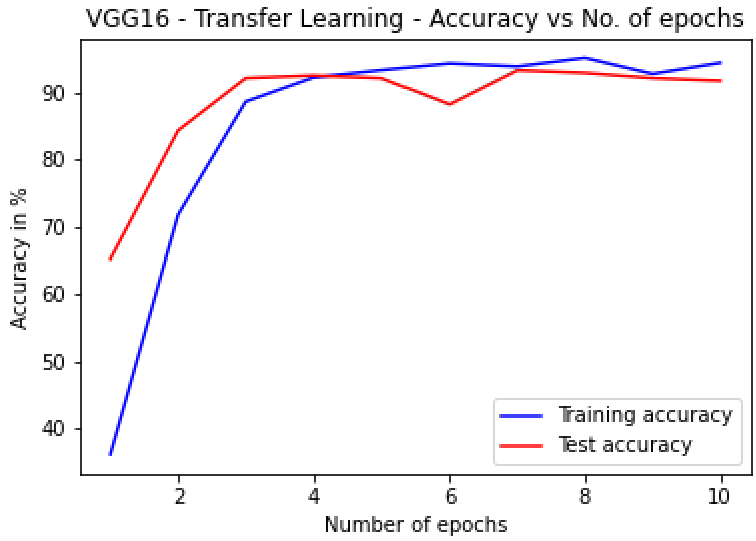
Weight decay rate: 0.001

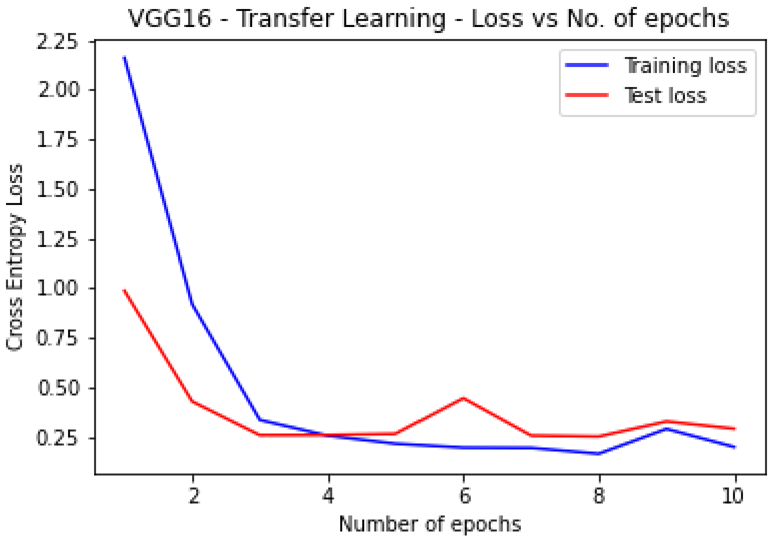
Loss: Cross Entropy loss

Number of epochs = 10

Batch size = 64

Data augmentation: Image were randomly horizontally flipped and center cropped.

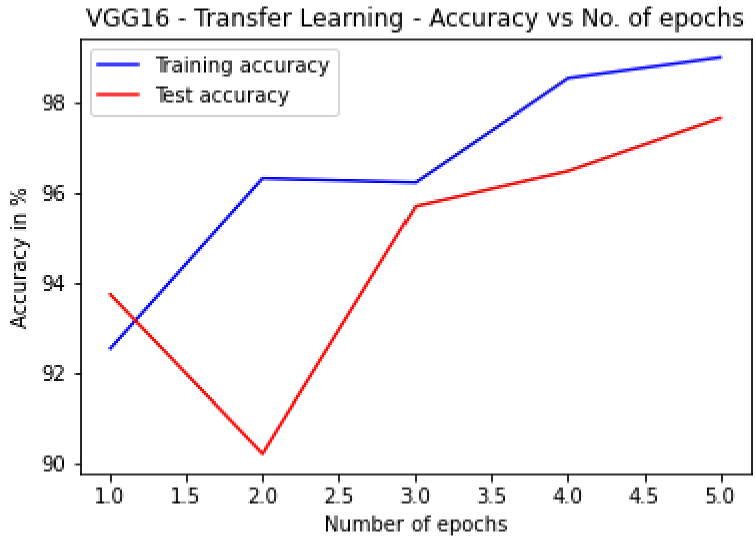


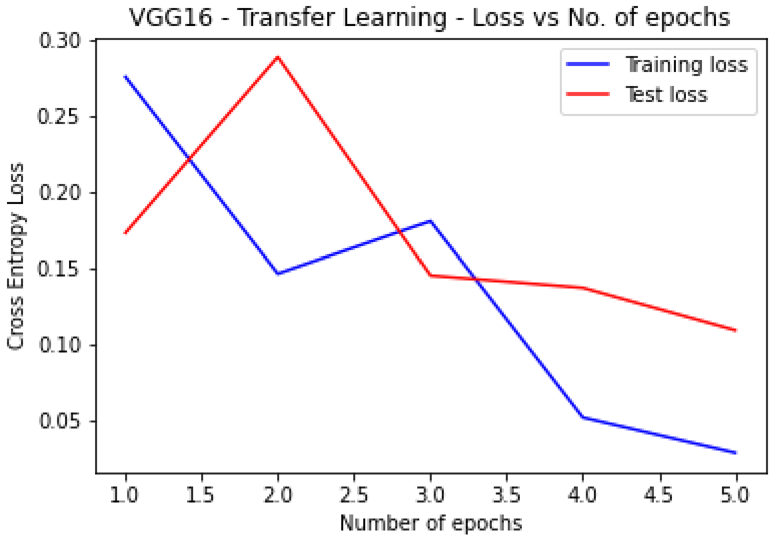


The VGG16 model achieved better performance on the training and test set compared to the simple neural network.

The feature extractor layers of VGG16 were unfreezed such that their weights can be optimized using the batch gradient descent.

Learning rate: 0.0001





The performance of the model improved considerably and it achieved an accuracy of about 99% and 98% on the training and test set respectively.