CS6910: Fundamentals of Deep Learning

Programming Assignment-2

AutoEncoders and Convolutional Neural Networks

MLFFNNs

Team Number 19

Aditya Mohan ED19B001 Shreyas Singh ED19B030

Sujay Srivastava ED19B033

Course Instructor:

CHANDRA SEKHAR C

Dept. Of Computer Science

Indian Institute of Technology Madras-600036

Note: For Determining model convergence, **early stopping with patience=3** has been implemented in all experiments

Task 1:

Dimensionality Reduction and using MLFFNN for Classification

a) Dimensional Reduction using PCA

Principal component analysis (**PCA**) is the process of computing the principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest.

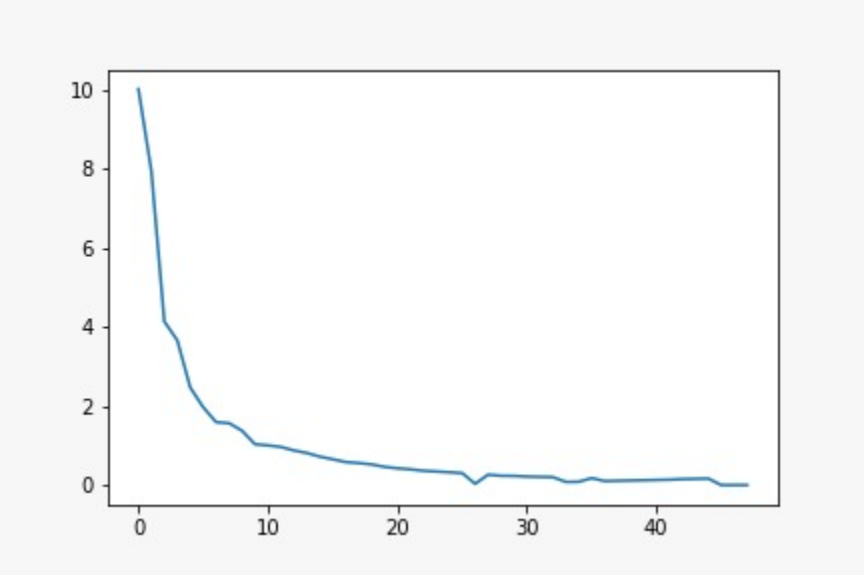

Fig 1.Variance in decreasing order vs Principle Component graph for the 48-feature dataset

- No. of features v/s Accuracy of the Model

Sr. No	No. of features	Accuracy
1	2	42.46%
2	4	42.85%
3	8	44.94%
4	16	47.81%

5	32	48.46%

- Loss v/s Epoch for Classifier using PCA

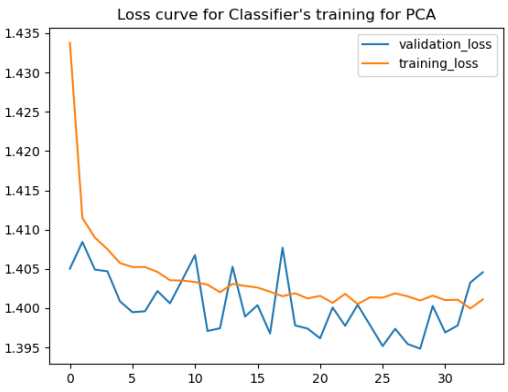

Fig 1.1 Loss vs Epoch for PCA with 2 features

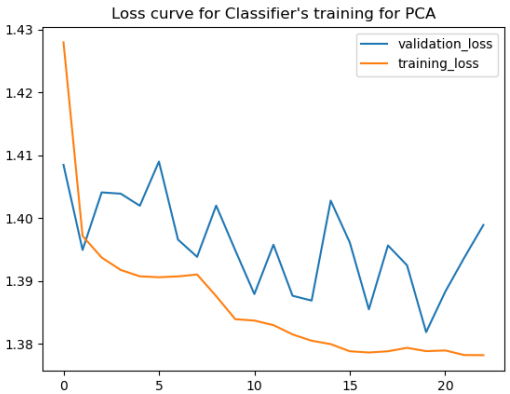

Fig 1.2 Loss vs Epoch for PCA with 4 features

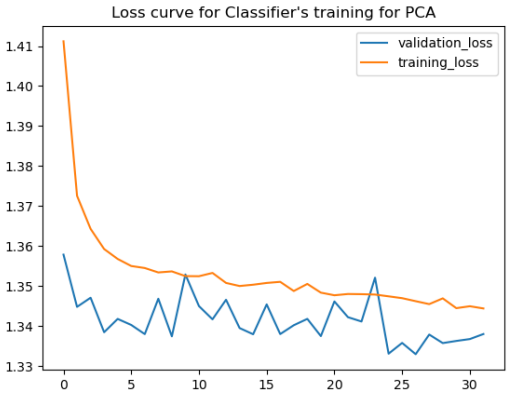

Fig 1.3 Loss vs Epoch for PCA with 8 features

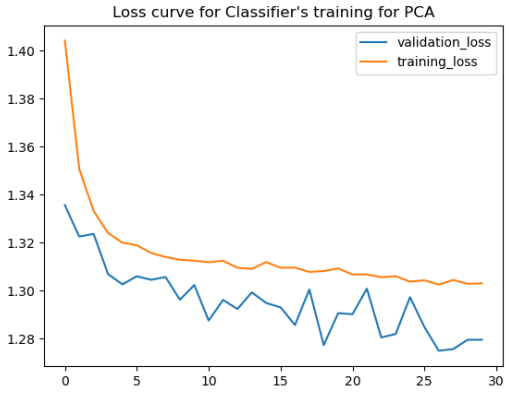

Fig 1.4 Loss vs Epoch for PCA with 16 features

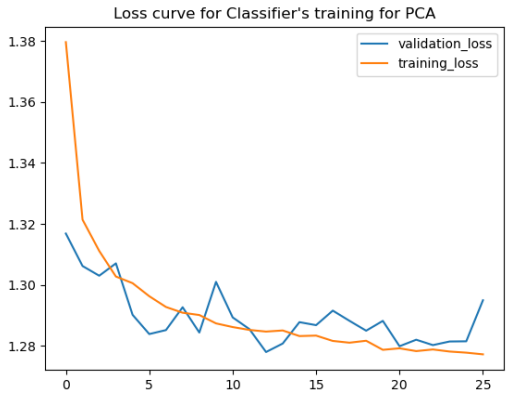

Fig 1.5 Loss vs Epoch for PCA with 32 features

b) Dimensional Reduction using AANN

Sr. No	AANN Architecture	Accuracy
1	[48 ,16 ,8 , 2 , 8, 16, 48]	39.72%
2	[48 ,24 ,12 , 4 , 12, 24, 48]	43.18%
3	[48 ,32 ,16 ,8 , 16, 32, 48]	45.59%
4	[48 ,36 ,24 ,16 , 24, 36, 48]	47.16%
5	[48 ,128, 64, 32 , 64, 128, 48]	47.75%

Table 2 Accuracy of the model for different AANN architecture

- Loss vs Epoch for Classification with AANN

1) AANN architecture [48 ,16 ,8 , 2 , 8, 16, 48]

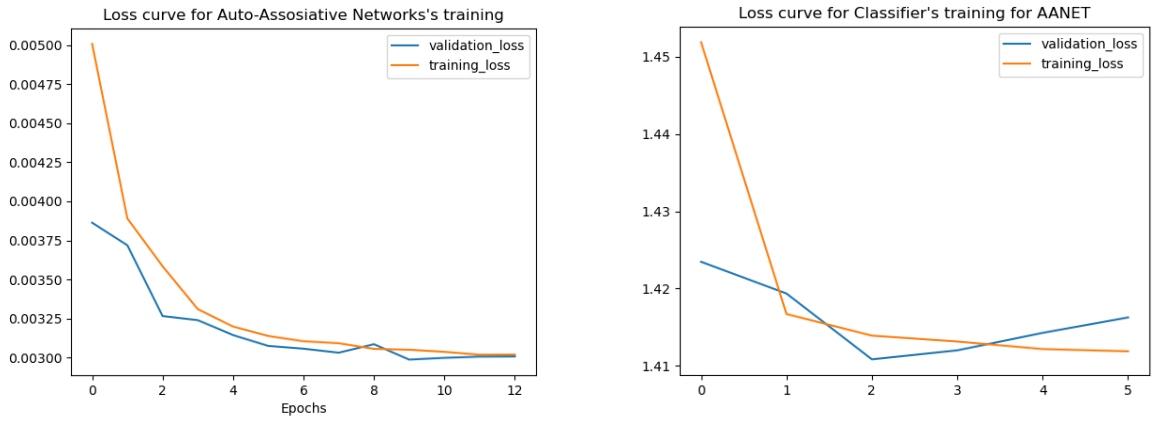

Fig 1.6 Loss vs epoch for training the AANN (left) and Loss vs epochs for classification after using AANN for dimensional reduction

2) AANN architecture- [48 ,24 ,12 , 4 , 12, 24, 48]

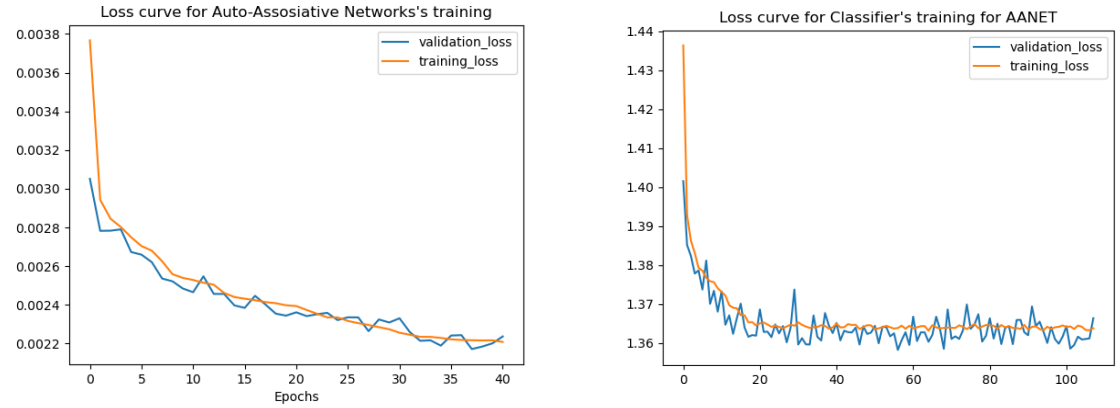

Fig 1.7 Loss vs epoch for training the AANN (left) and Loss vs epochs for classification after using AANN for dimensional reduction

3) AANN architecture- [48 ,32 ,16 ,8 , 16, 32, 48]

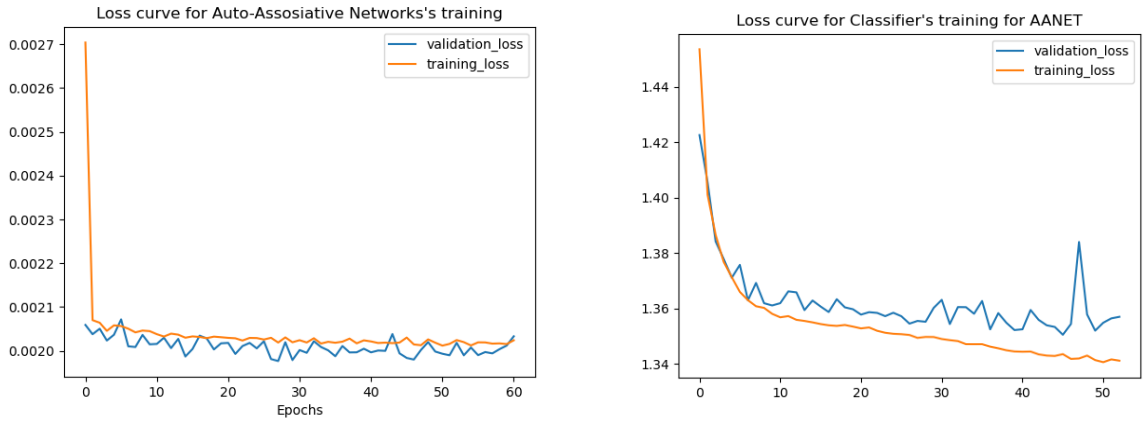

Fig 1.8 Loss vs epoch for training the AANN (left) and Loss vs epochs for classification after using AANN for dimensional reduction

4) AANN architecture- [48 ,36 ,24 ,16 , 24, 36, 48]

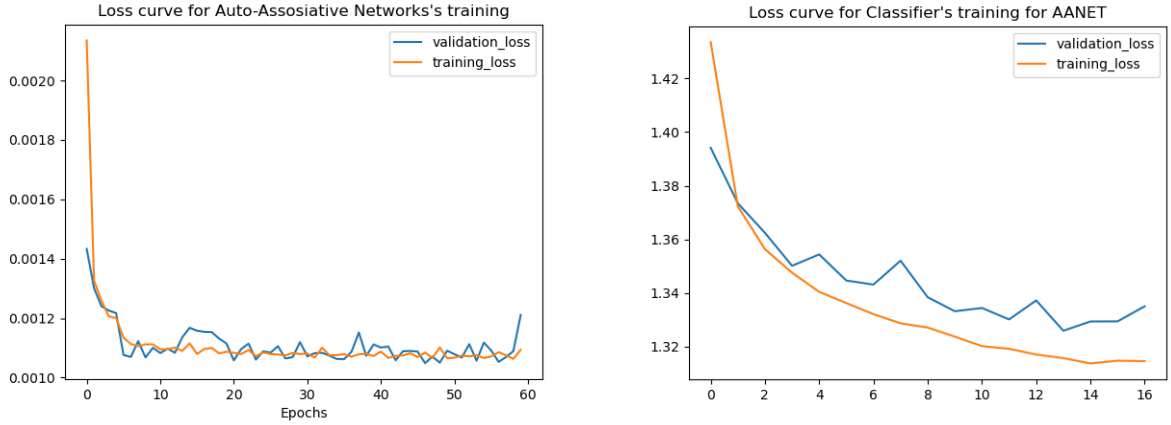

Fig 1.9 Loss vs epoch for training the AANN (left) and Loss vs epochs for classification after using AANN for dimensional reduction

5) AANN architecture- [48 ,128, 64, 32 , 64, 128, 48]

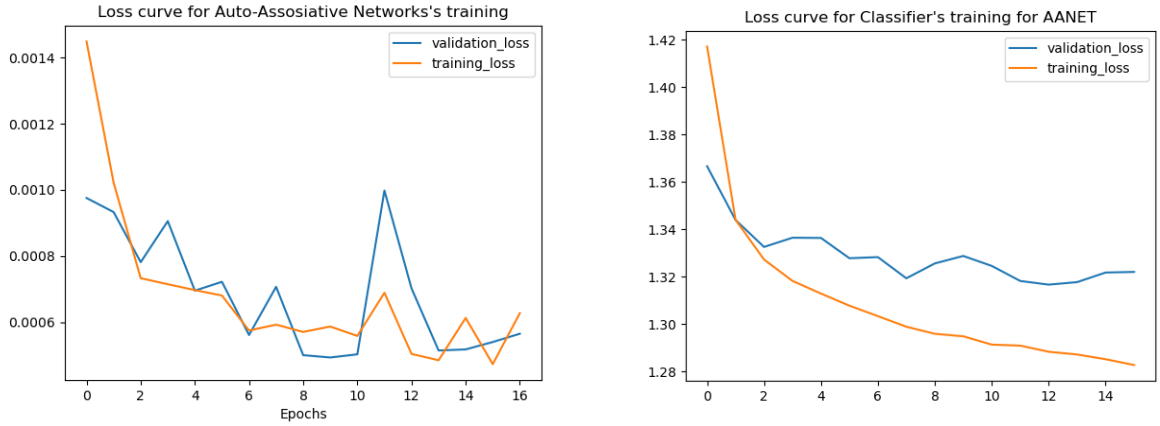

Fig 1.10 Loss vs epoch for training the AANN (left) and Loss vs epochs for classification after using AANN for dimensional reduction

Observations

- Accuracy of the classification model increases with both dimension reduction methods(PCA and AANN) with the increase in number of nodes in the bottleneck layer and the number of features. It can be attributed to higher retention of information.
- We see the accuracy increases only slightly when the number of features are increased from 16 to 32. We can see from the Principle component analysis that these extra features do not hold significant information and using only 16 features does not affect the accuracy of the model significantly.Hence, it is a good trade-off.
- Increasing the no. of nodes in bottleneck layer reduces the net loss of Auto-Associative Neural Network training

Task 2:

Stacked Autoencoder based pre-training for Classification

Architecture of Autoencoders used:-

Architecture 1:

Encoder-Decoder 1: [48, 256,128, 32, 128, 256, 48]

Encoder-Decoder 2: [32 , 64 , 24 , 64 , 32]

Encoder-Decoder 3: [24 , 32 , 16 , 32 , 24]

Classificier: [48, 32, 16, 8, 5]

Accuracy achieved: 54.65%

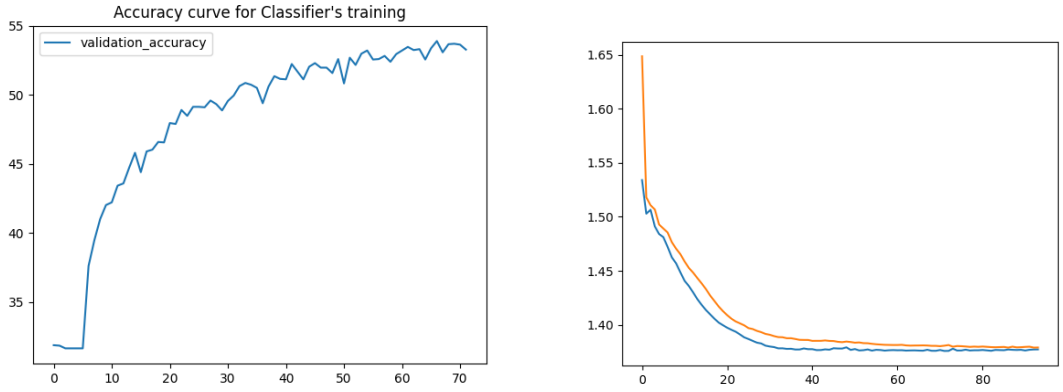

Fig 2.1 Accuracy vs epoch for the Architecture 1(left) and Loss vs epoch for the classification on the features extracted using stacked autoencoder(right)

Architecture 2:

Encoder-Decoder 1: [48 , 32 , 48]

Encoder-Decoder 2: [32 , 24 , 32]

Encoder-Decoder 3: [24 , 16 , 24]

Classifier: [48, 32, 16, 8, 5]

Accuracy achieved: 42.25%

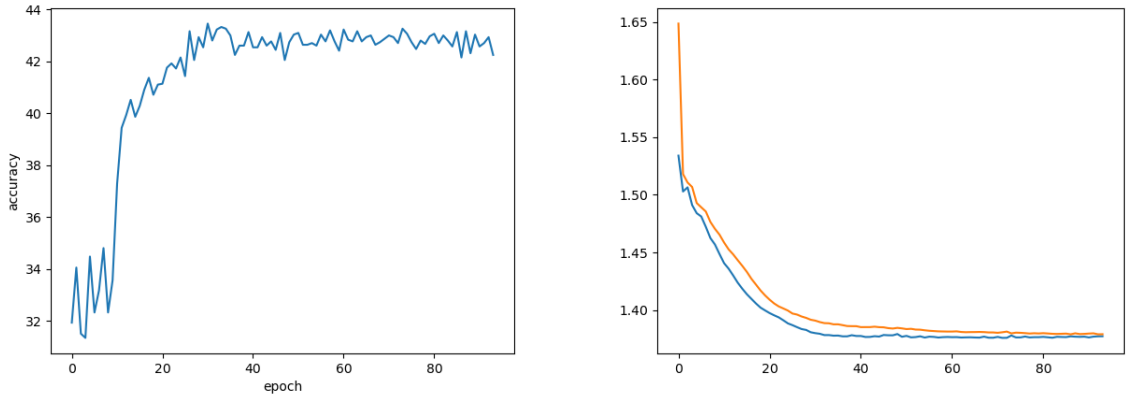

Fig 2.2 Accuracy vs epoch for the Architecture 2(left) and Loss vs epoch for the classification on the features extracted using stacked autoencoder(right)

Observations

- We can see that a deeper architecture (1) outperforms the shallower architecture (2). Feature representation learnt using deeper encoder-decoder model during unsupervised learning phase is more robust as compared to shallower model, due to deeper models' better representation capacity
- Stacked autoencoder outperforms Auto Associative Neural Network with similar number of parameters, showing that stacking autoencoders is a more efficient method of training AANNs.

Task 3:

MLFFNN for classification with Deep CNN to extract features from image

a) VGGNet as Deep CNN

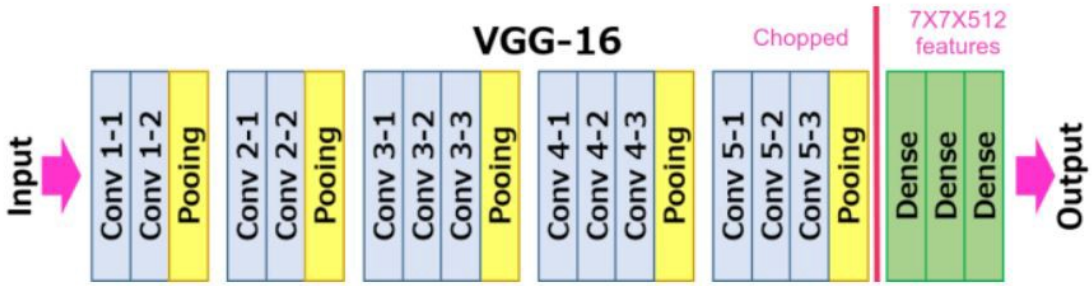

Fig 3.1 VGG-16 Architecture

```
Sequential(
  (0): Linear(in_features=25088, out_features=4096, bias=True)
  (1): Tanh()
  (2): Linear(in_features=4096, out_features=512, bias=True)
  (3): Tanh()
  (4): Linear(in_features=512, out_features=5, bias=True)
)
```

Fig 3.2 Classification head on VGG-16 Architecture

VGGNet gives 25088 dimensional feature vectors after flattening

We then added a MLFFNN classification head of 4096 nodes, then next layer of 512 nodes and at last a 5 node softmax layer for classification.

Total number of parameters to be trained in the classification head

25088* 4096 + 4096*512+512*5+ 4096+512+5=

104,864,773 parameters

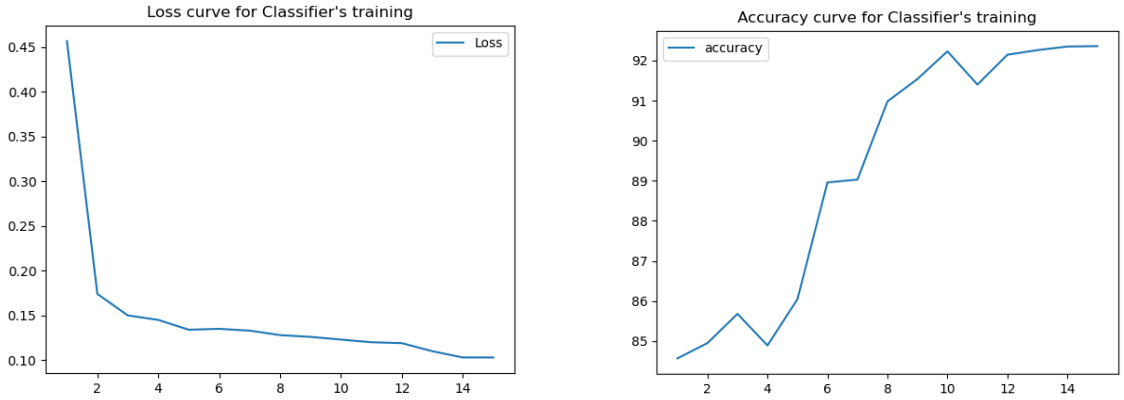

Figure 3.3 Loss vs epoch for VGGNet classifier(left) and Accuracy vs epoch for VGGNet Classifier(right)

Accuracy achieved: 92.73%

b) GoogLeNet as Deep CNN

Fig 3.4 GoogLeNet Architecture

```
Linear(in_features=1024, out_features=5, bias=True)
```

Fig 3.5 Classification head on GoogLeNet Architecture

- GoogleNet gives 1024 dimensional feature vectors after flattening
- We added a softmax layer of 5 nodes on feature vectors from GoogLeNet
- Total number of parameters to be trained in the classification head 1024*5+ 5 = 5125 parameters

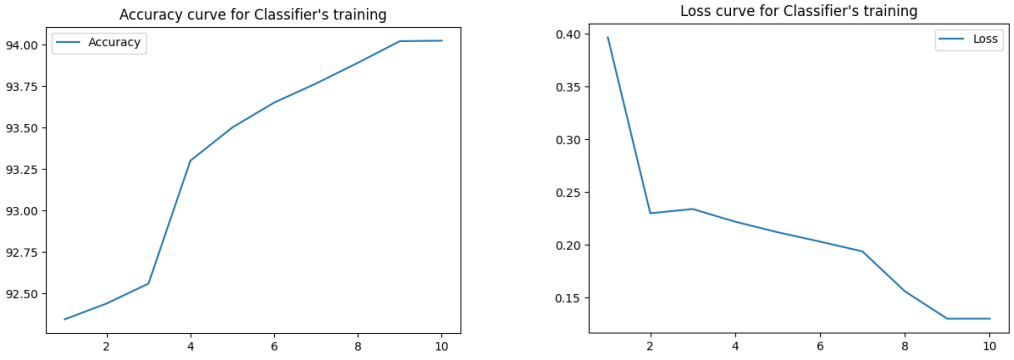

Figure 3.7 Loss vs epoch for GoogLeNet classifier(left) and Accuracy vs epoch for GoogLeNet Classifier(right)

Accuracy achieved: 94. 52%

Observations

- Despite having significantly lesser number of parameters , final accuracy of GoogLeNet is higher than of VGGNet-16 , showing that its a better Deep CNN architecture
- Fine- tuning GoogLeNet takes lesser number of epochs than VGG-16 for convergence, due to GoogLeNet having ~7 million parameters only and VGG-16 having 138 million parameters

Task 4:

Using CNN for Image Classification

```
ConvNet(
  (conv1): Conv2d(3, 4, kernel_size=(3, 3), stride=(1, 1))
  (avgpooling1): AvgPool2d(kernel_size=(2, 2), stride=(2, 2), padding=0)
  (conv2): Conv2d(4, 32, kernel_size=(3, 3), stride=(1, 1))
  (avgpooling2): AvgPool2d(kernel_size=(2, 2), stride=(2, 2), padding=0)
  (flatten): Flatten(start_dim=1, end_dim=-1)
  (fc1): Linear(in_features=93312, out_features=1024, bias=True)
  (dropout): Dropout(p=0.5, inplace=False)
  (fc2): Linear(in_features=1024, out_features=5, bias=True)
  (relu): ReLU()
)
```

Fig 4.1 CNN Architecture used for classification

Sr. No	No. of feature maps in CL2	Accuracy
1.	16	62.31%
2.	32	65.28%
3.	64	69,65%

Table 4.1 No. of feature maps vs accuracy table

Observations

- With the increase of number of feature maps in convolutional layer 2, the representation capacity of the model increases, hence the increase in accuracy
- Increase in number of feature maps leads to increase in number of parameters, hence the number of epochs required for convergence increases
- The accuracy achieved using this classifier is significantly less than the accuracy obtained via fine-tuning of VGG-16 and Google-LeNet due to the absence of any pre-training and lesser representation capacity of the model