CS7015: DEEP LEARNING
Assignment 2 Report

Team : 7

Authors:
Rahul Chakwate:AE16B005
Soham Dixit:CS16B006
Pranav Gadikar:CS16B115

April 11, 2019

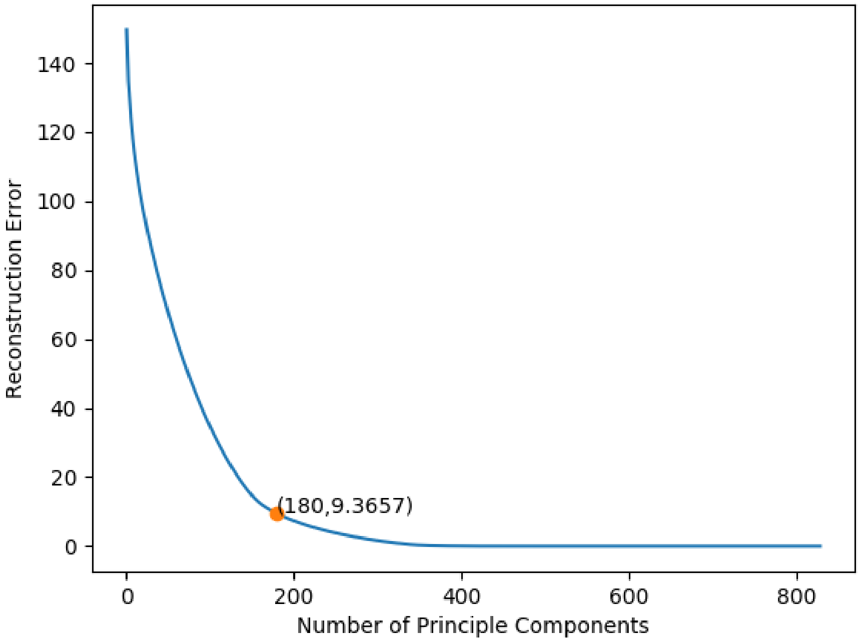
CONTENTS

1 DIMENSION REDUCTION USING PCA AND AANN

1.1 INTRODUCTION:

Experiments are individually performed using PCA and Auto associative Neural Network to find out the reduced dimensions using each of the methods. After performing the dimension reduction the confusion matrix for each of the reduced dimension representation is then compared using their classification on labelled data from Dataset:1

1.2 PLOT FOR RECONSTRUCTION LOSS AND NUMBER OF PRINCIPAL COMPONENTS:



Number of principle components was chosen to be 180 as the error 9.3657 is quite less compared to total reconstruction error and the rate of decrease of error is also very less after 180 as compared to high slope value in the start for lesser values of principle components.

Table 1: Experiments for MLFFN with one hidden layer

Nodes in Hidden-1	Epochs	Loss
100	25	0.9142
80	25	0.9196
70	27	0.9170
60	27	0.9172
50	31	0.9195
40	33	0.9186

Best Model for classification was chosen to be with 70 nodes as it was the model with good training accuracy and least overfitting.

1.3 Experiments for AANN:

Mini batch with batch size=10,Optimizer used is Adam,Learning rate is 0.001,ReLU activation

Table 2: Loss for various number of nodes in 2 hidden layers

Nodes in Hidden-1	Reduced Dimension	Epochs	Loss
200	40	14	0.1396
100	40	18	0.1377
70	40	17	0.1438
200	48	27	0.1304
100	48	25	0.1301
70	48	31	0.1310
200	60	22	0.1308
100	60	21	0.1311

Table 3: Experiments for MLFFN with one hidden layer

Nodes in Hidden-1	Epochs	Loss
60	42	1.1876
50	46	1.1848
40	47	1.1947
70	68	1.1460
80	68	1.1585

Best model for AANN: 100 nodes in hidden layer-1 and 48 as reduced dimension.
Best model for MLFFN: 70 nodes in hidden layer

1.4 Comparison for Reduced Dimensions using PCA and AANN:

Table 4: Accuracy: PCA=79% , AANN=80.8%

PCA Confusion					AANN Confusion				
Coast	Forest	Highway	Street	Building	Coast	Forest	Highway	Street	Building
81.5%	4.5%	7.9%	2%	3.86%	83%	2.6%	8%	0%	6.2%
5.8%	90.9%	0%	0.9%	2.27%	1.8%	89.6%	0%	1.8%	6.6%
13%	1.3%	68.1%	7.9%	9.4%	12.8%	0%	76.9%	5.1%	5.1%
0%	2.6%	3.8%	82.9%	10.4%	6.5%	5.4%	3.2%	74.7%	9.8%
12.6%	4.7%	2.4%	12.3%	67.7%	11.9%	5.4%	3.2%	2.1%	77.1%

1.5 Observations and Inferences:PCA:

- Number of PCA were selected using the graph plotted above.
- Initially the slope for error vs number of components was very high later started diminishing.

1.6 Observations and Inferences:AANN:

- Nodes in the hidden layer were chosen experimentally.
- Number of features extracted by AANN was lesser than that extracted by the PCA.
- Also the classification accuracy for AANN was found to be better than PCA.
- Difference was expected to be more,but since the size of data was small,overfitting led to lesser accuracy than it would have actually been for features using AANN.

2 Task 2: Stacked AutoEncoder for Dataset 1

2.1 Introduction to Stacked Autoencoder

Stacked Autoencoder is an effective model to pretrain a neural network architecture. Each of the autoencoder is trained individually and then stacked together to form an architecture which can be used for several tasks like classification, segmentation, regeneration, etc. In stacked autoencoders, the output of the first pretrained autoencoder is fed to the second autoencoder and so on.

2.2 Pretraining the Autoencoders

The experiment is conducted by changing the number of nodes in each autoencoder. The number of layers is kept fixed.

The following hyperparameters are used while pretraining:
Optimizer = Adam
Learning rate = 0.001
convergence = 1e-5
batch size = 4
activation function = ReLU
Note1: Only decoder of the first autoencoder has logistic activation function.
Note2: Only for classification network, number of epochs are fixed instead of convergence criteria since the loss is fluctuating.

Table 5: Convergence Epoch and MSE(of that layer) for different no. of nodes in 1 hidden layer.

No. of nodes	epochs	loss_train	loss_test
80	38	0.7437	0.5561
100	27	0.7474	0.5472

No. of nodes	epochs	loss_train	loss_test
150	31	0.7425	0.5419
200	24	0.7411	0.5278
500	16	0.7407	0.5224

Table 6: Loss and Accuracy of classification network for different no. of nodes in 1 hidden layer.

No. of nodes	epochs	Train Loss	Train Accuracy	Test Accuracy
80	199	0.9048	0.9081	0.9062
100	199	0.9049	0.9019	0.9125
150	199	0.9050	0.9269	0.9250
200	199	0.9049	0.9457	0.9375
500	199	0.9051	0.9457	0.9437

Table 7: Convergence Epoch and MSE(of that layer) for different no. of nodes in 2 hidden layer.

No. of nodes	epochs	loss_train	loss_test
30	27	0.9193	2.0692
60	67	0.3167	1.2957
100	21	0.1592	0.5103
200	21	0.0024	0.0068

Table 8: Loss and Accuracy of classification network for different no. of nodes in 2 hidden layer.

No. of nodes	epochs	Train Loss	Train Accuracy	Test Accuracy
30	199	0.9048	0.9353	0.9344
60	199	0.9049	0.9457	0.9375
100	199	0.9054	0.9395	0.9375
200	199	0.9050	0.9457	0.9437

Table 9: Convergence Epoch and MSE(of that layer) for different no. of nodes in 3 hidden layer.

No. of nodes	epochs	loss_train	loss_test
30	27	0.9132	3.4431
60	82	0.4193	1.6360
100	21	0.0020	0.0100

Table 10: Loss and Accuracy of classification network for different no. of nodes in 3 hidden layer.

No. of nodes	epochs	Train Loss	Train Accuracy	Test Accuracy
30	199	0.9054	0.9395	0.9375
60	199	0.9049	0.9332	0.9219
100	199	0.9051	0.9415	0.9375

2.3 Classification Network

The classification network architecture includes the encoder layers of the three pretrained autoencoders.

The following experiments show how the network behaves when pretrained and not pretrained.

The following hyperparameters are used in the classification network:
Optimizer = SGD
Learning rate = 0.001
convergence = NA
number of epochs = 200
batch size = 4
activation function = ReLU
Note: Last layer has softmax activation function.

The following are the tables comparing the confusion matrix for pretrained and not pretrained model for classification.

Table 11: Confusion Matrix for Pretrained Network

_	coast	forest	highway	street	tallbuilding
coast	103	1	5	1	2
forest	2	86	1	0	1
highway	2	0	84	2	1
street	1	0	1	77	0
tallbuilding	1	3	0	5	100

Table 12: Confusion Matrix for Not Pretrained Network

_	Ankle boot	T-shirt/top	Bag	Coat	Trouser
Ankle boot	2100	13	13	0	3
T-shirt/top	0	1775	16	15	21
Bag	14	153	1993	70	22
Coat	0	84	22	2039	39
Trouser	0	20	3	5	2080

Table comparing final accuracies obtained from pretrained and not pretrained networks are given below.

2.4 Observation and Conclusion

Observations in pretraining the network and conclusions on classification network.

- Dataset 1 has less number of examples which are further divided into labeled and unlabeled data points.
- This results in more parameters than the data points which increases the risk of overfitting.
- As the number of nodes is increased, the train and validation loss of the corresponding autoencoder decreases. This is because the model overfits the data as no. of nodes is increased.
- The final classification accuracy also shows increase with the no. of nodes. This is also a result of overfitting. Since train and test data is sampled from the same dataset, the test accuracy also increases almost equally as train accuracy.

3 Task 3: Stacked AutoEncoder for Dataset2

3.1 Pretraining the Autoencoders

The experiment is conducted by changing the number of nodes in each autoencoder. The number of layers is kept fixed.

The following hyperparameters are used while pretraining:
Optimizer = Adam
Learning rate = 0.001
convergence = 1e-5
batch size = 128
activation function = ReLU
Note: Only decoder of the first autoencoder has logistic activation function.

Table 13: Convergence Epoch and MSE for different no. of nodes in 1 hidden layer.

No of Nodes	epochs	MSE loss train	MSE loss validation
64	51	0.0294	0.0446
128	58	0.0145	0.0344
256	41	0.0077	0.0271
512	34	0.0060	0.0227

No of Nodes	epochs	MSE loss train	MSE loss validation

Table 14: Convergence Epoch and MSE for different no. of nodes in 2 hidden layer.

No of Nodes	epochs	MSE loss train	MSE loss validation
64	35	0.2327	0.1726
128	39	0.1436	0.1244
256	28	0.0332	0.0457
384	32	0.0087	0.0157

Table 15: Convergence Epoch and MSE for different no. of nodes in 3 hidden layer.

No of Nodes	epochs	MSE loss train	MSE loss validation
32	22	0.5035	0.3793
64	21	0.4051	0.2674
128	37	0.2742	0.1948
256	31	0.0832	0.0963

3.2 Classification Network

The classification network architecture includes the encoder layers of the three pretrained autoencoders.

The following experiments show how the network behaves when pretrained and not pretrained.

The following hyperparameters are used in the classification network:
Optimizer = SGD
Learning rate = 0.01
convergence = 1e-7
batch size = 128
activation function = ReLU
Note: Last layer has softmax activation function.

The following are the tables comparing the confusion matrix for pretrained and not pretrained model for classification.

Table 16: Confusion Matrix for Pretrained Network

_	Ankle boot	T-shirt/top	Bag	Coat	Trouser
Ankle boot	2106	18	9	1	6
T-shirt/top	0	1677	12	8	9
Bag	8	92	1994	24	12
Coat	0	207	28	2090	33
Trouser	0	51	4	6	2105

Table 17: Confusion Matrix for Not Pretrained Network

_	Ankle boot	T-shirt/top	Bag	Coat	Trouser
Ankle boot	2100	13	13	0	3
T-shirt/top	0	1775	16	15	21
Bag	14	153	1993	70	22
Coat	0	84	22	2039	39
Trouser	0	20	3	5	2080

The following are the observed trends in accuracies.

Table 18: Learning of Pretrained Network

epoch 1	loss:0.9057	train_accuracy:0.9348
epoch 2	loss:0.9051	train_accuracy:0.9447
epoch 3	loss:0.9050	train_accuracy:0.9495
epoch 4	loss:0.9050	train_accuracy:0.9516
epoch 5	loss:0.9049	train_accuracy:0.9540
epoch 6	loss:0.9048	train_accuracy:0.9587
epoch 7	loss:0.9048	train_accuracy:0.9602
epoch 8	loss:0.9048	train_accuracy:0.9577
epoch 9	loss:0.9048	train_accuracy:0.9530
epoch 10	loss:0.9048	train_accuracy:0.9446

Table 19: Learning of Not Pretrained Network

epoch 1	loss:1.5800	train_accuracy:0.6248
epoch 2	loss:1.1302	train_accuracy:0.5699
epoch 3	loss:0.9135	train_accuracy:0.8950
epoch 4	loss:0.9055	train_accuracy:0.9174
epoch 5	loss:0.9052	train_accuracy:0.9253
epoch 6	loss:0.9051	train_accuracy:0.9366
epoch 7	loss:0.9050	train_accuracy:0.9405
epoch 8	loss:0.9050	train_accuracy:0.9430
epoch 9	loss:0.9050	train_accuracy:0.9429
epoch 10	loss:0.9049	train_accuracy:0.9437
epoch 11	loss:0.9049	train_accuracy:0.9432
epoch 12	loss:0.9049	train_accuracy:0.9446
epoch 13	loss:0.9049	train_accuracy:0.9459
epoch 14	loss:0.9049	train_accuracy:0.9458
epoch 15	loss:0.9049	train_accuracy:0.9448
epoch 16	loss:0.9049	train_accuracy:0.9453
epoch 17	loss:0.9048	train_accuracy:0.9474

3.3 Observation and Conclusion

Observations and conclusions derived from the experiments carried above:

- Dataset 2 has sufficient data points for the model to pretrain from the unlabeled data.
- The pretrained model is seen to be initialized at a better point than the not pretrained model. First epoch if pretrained model begins at around 93 percent accuracy as compared to 60 percent accuracy of the not pretrained model.

4 Denoising Autoencoder:

4.1 Introduction:

Experiments are carried out using 3 stacked encoders and decoders. Number of nodes are chosen greedily deepending upon the loss obtained on each of the configuration of nodes. Adam Optimizer with learing rate=0.001 and ReLU activation function, mini batch mode with batch size=128 has been used for all experiments.

4.2 Experiments for Denoising Autoencoder:

Table 20: Experiments for different number of nodes in 1st hidden layer of Stacked Autoencoder:

Nodes in Stack-1	Epochs	Loss
80	50	0.0536
70	34	0.0548
60	42	0.0548

80 is chosen to be the best number of nodes as the loss is not decreasing fast after increasing number of nodes from 80,while it increases fast with decrease in number of nodes.

Table 21: Experiments for different number of nodes in 2nd hidden layers of Stacked Autoencoder:

Nodes in Stack-2	Epochs	Loss
70	45	0.0519
65	42	0.0520
60	44	0.0508

70 is chosen to be the best number of nodes as the loss is not decreasing fast after increasing number of nodes from 70,while it increases fast with decrease in number of nodes.

Table 22: Experiments for different number of nodes in 3rd hidden layers of Stacked Autoencoder:

Nodes in Stack-3	Epochs	Loss
60	42	1.1876
50	46	1.1848
40	47	1.1947
70	68	1.1460

65 is chosen to be the best number of nodes as the loss is not decreasing fast after increasing number of nodes from 65,while it increases fast with decrease in number of nodes.

4.3 Reconstructed images for increasing hidden layers vs Epochs:

1 Autoencoder

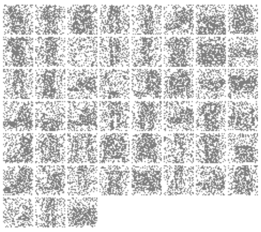


Figure 1: Orignal

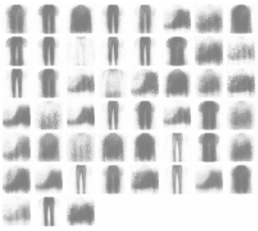


Figure 2: Epoch 7

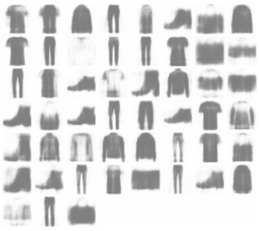


Figure 3: Epoch 14

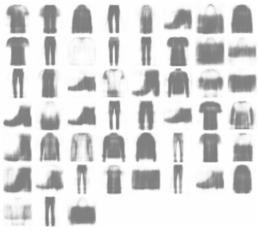


Figure 4: Final

2 Stacked Encoders

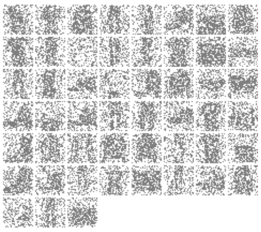


Figure 5: Orignal

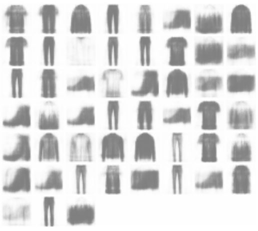


Figure 6: Epoch 1

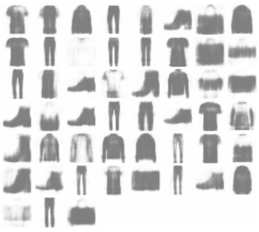


Figure 7: Epoch 14

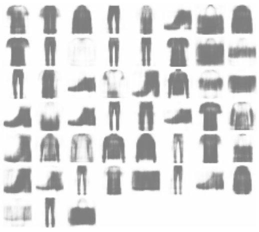


Figure 8: Final

3 Stacked Encoders

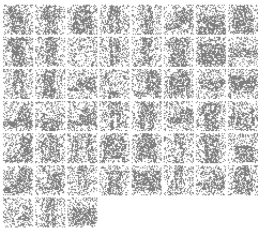


Figure 9: Orignal

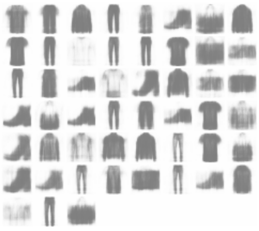


Figure 10: Epoch 7

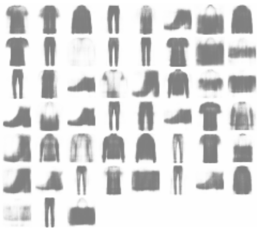


Figure 11: Epoch 14

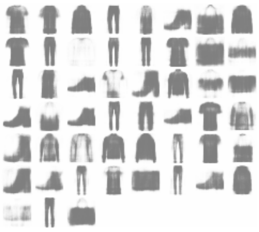


Figure 12: Final

4.4 EXPERIMENTS FOR VARIOUS ERROR PERCENTAGES:

Table 23: Experiments for different number of nodes in 1st hidden layer of Stacked Autoencoder:

Percentage Error	Reconstruction Error
30	7.58
20	6.21
10	5.6

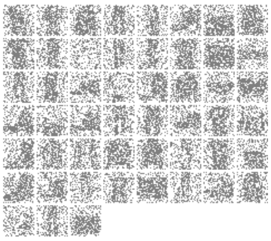


Figure 13: Image with 30% Error

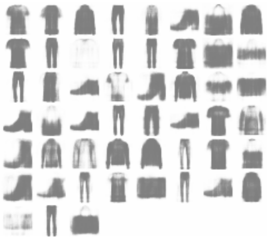


Figure 14: Reconstructed

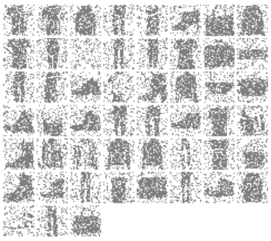


Figure 15: Image with 20% Error

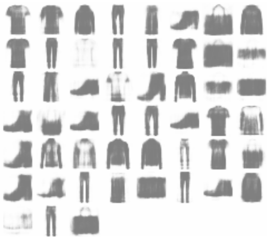


Figure 16: Reconstructed

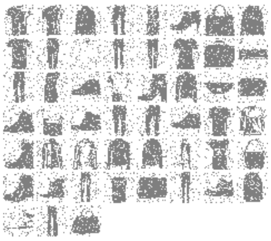


Figure 17: Image with 10% Error

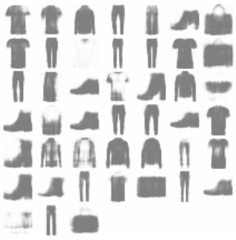


Figure 18: Reconstructed

4.5 Observations and Inferences:

- Loss associated with stacked encoder decreased with increase in number of stacked encoders.
- Increasing the number of stacking layers improved the reconstruction of the image.
- More compplicated features got extracted leading to better image reconstruction.
- Further increase in accuracy is expected with increase in size of especially first hidden layer whose size needs to be restricted due to size of the dataset being small and input dimension being large.

5 Stacked RBM based pre-training of a DNN based Classifier using Binary-Binary RBMs

5.1 Introduction:

The number of hidden layers in the DNN is fixed to 3.The pre-training is carried out by training RBM greedily. The optimal number of nodes in a hidden layer is decided based on the accuracy shown on the training data. First, the number of nodes in first hidden layer is fixed, followed by the second layer and third layer.

5.2 Experiments on Size of Hidden Layers:

First the size of the first hidden layer is varied.

Table 24

Nodes in Hidden-1	Epochs	Training Accuracy
500	85	94.55%
400	71	94.35%
300	46	94.27%

Based on training accuracy, the first hidden layer size is fixed to 500 nodes. With the size of hidden layer-1=500, the second layer is varied.

Table 25

Nodes in Hidden-2	Epochs	Training Accuracy
200	85	94.55%
150	73	94.34%
100	87	94.22%

The size of second layer is fixed to 200.

Table 26

Nodes in Hidden-3	Epochs	Training Accuracy
50	85	94.55%
25	97	94.44%
10	71	94.31%

The final size of the hidden layers is 500,200,50 respectively.

5.3 Comparison of DNN performance with and without pre-training:

While training the DNN, each hidden layer uses ReLU acitvation, convergence criteria is 0.0001 and Adam update rule is used.

With Pre-Training:

Table 27: Training Accuracy: 94.55%, Test Accuracy:94.57%, Epochs=85

Training data Confusion					Test data Confusion				
Ankle boot	T-shirt	Bag	Coat	Trouser	Ankle boot	T-shirt	Bag	Coat	Trouser
98.84%	0.18%	0.85%	0.06%	0.06%	98.56%	0.28%	0.93%	0.14%	0.07%
0.06%	93.87%	2.8%	2.2%	0.84%	0.29%	94.2%	2.8%	2%	0.65%
1.1%	4%	92.43%	2.2%	0.22%	1.1%	3.2%	93.08%	2.1%	0.35%
0.05%	6%	1.2%	92.4%	0.29%	0.36%	5.7%	1.6%	91.8%	0.5%
0%	3.5%	0.24%	0.67%	95.5%	0%	3.6%	0.14%	1%	95.2%

Without Pre-Training:

Table 28: Training Accuracy: 93.8%, Test Accuracy:93.9%, Epochs=106

Training data Confusion					Test data Confusion				
Ankle boot	T-shirt	Bag	Coat	Trouser	Ankle boot	T-shirt	Bag	Coat	Trouser
98.78%	0.3%	0.95%	0.06%	0%	98.78%	0.35%	0.7%	0.07%	0.07%
0.3%	93.57%	2.8%	2.6%	0.66%	0.29%	93.53%	2.4%	2.9%	0.87%
1.8%	4.7%	91.4%	1.8%	0.17%	1.3%	4.5%	92.2%	1.7%	0.14%
0.05%	5.4%	1.5%	92.4%	0.52%	0.35%	4.9%	1.9%	91.66%	1.1%
0%	3.8%	0.18%	0.86%	95.12%	0%	3.4%	0.07%	1.1%	95.3%

5.4 Observations:

Pre-training reduces the number of epochs required for convergence by a considerable number. The accuracy also increased slightly by pre-training. Also, while experimenting on the number of nodes in hidden layers, training accuracy tended to decrease with number of nodes.