CS7015: Assignment 2 Report

Team 5
Abhik Debnath – MA17M001
Sushant Kumar Seet – MA17M012

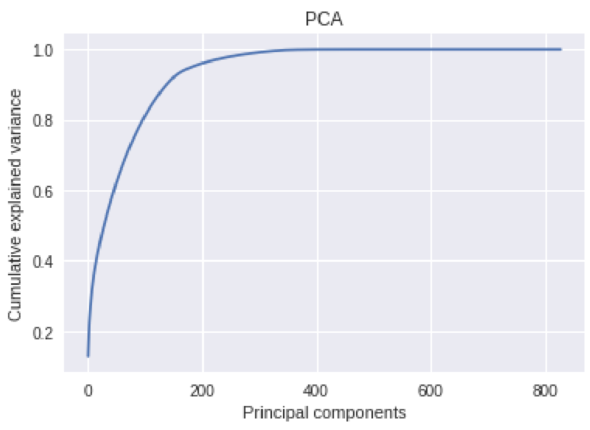
Task 1: Dimension reduction on Dataset 1 using PCA and AANN

MLFFNN Configuration:

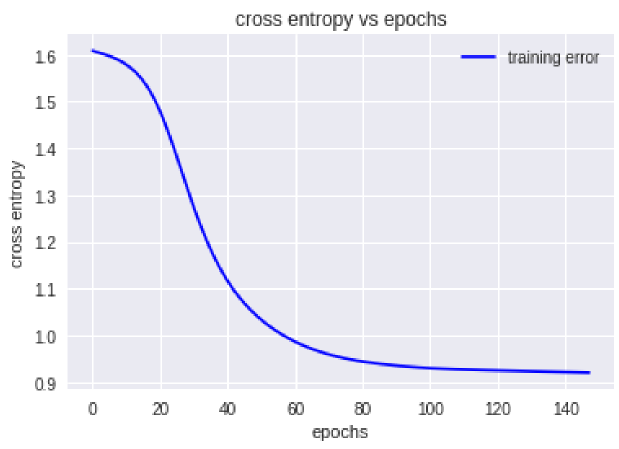
Hidden Layers: 2
Hidden Neurons: 50 in each layer
Activation function: ReLU
Optimizer: Adam

1.1 PCA

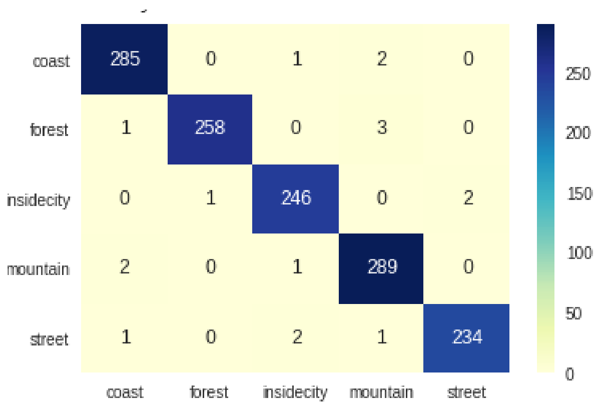
NPCA	Train Error	Train Accuracy	Test Error	Test Accuracy	Epochs
500	0.91	99.676	1.192	70.84	140
450	0.91	99.742	1.194	70.75	139
400	0.911	99.629	1.189	71.56	138
350	0.911	99.556	1.187	71.74	144
300	0.917	99.218	1.188	71.74	145
250	0.917	99.089	1.193	71.2	152
200	0.92	98.939	1.189	71.38	157
150	0.928	97.982	1.184	72.04	172
100	0.944	96.381	1.188	71.35	201
50	0.978	93.061	1.194	70.54	255
25	1.011	89.67	1.213	68.8	318

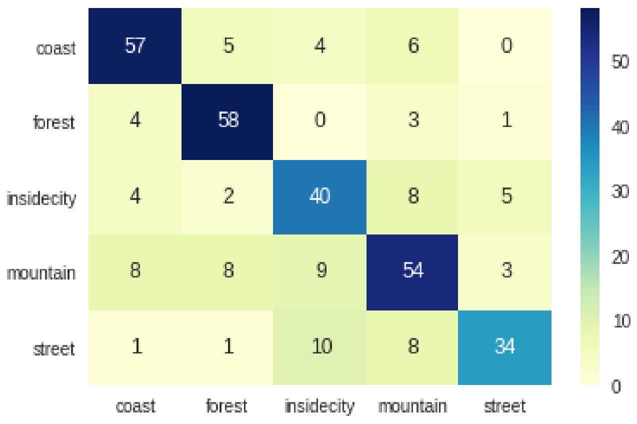


From the table, we observe that even with 300 principal components, the accuracy and error is almost same as 500 principal components.
Also, the number of Principal components from the curve as given by the cumulative explained variance ratio <=0.99 is 292.
Hence, we shall choose 300 as starting point for the bottleneck size of AANN.



NPCA = 300



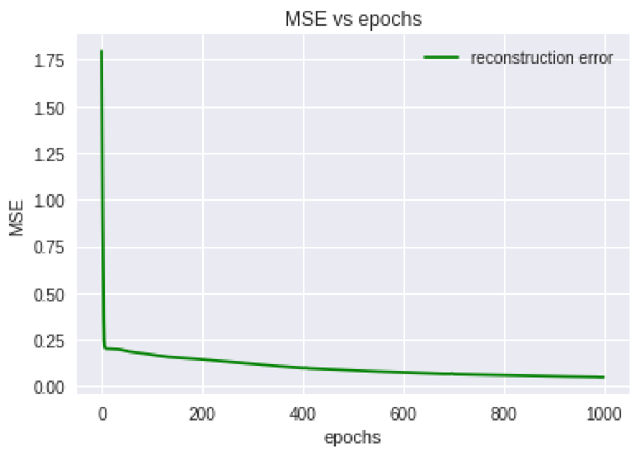


Confusion matrix for Train data

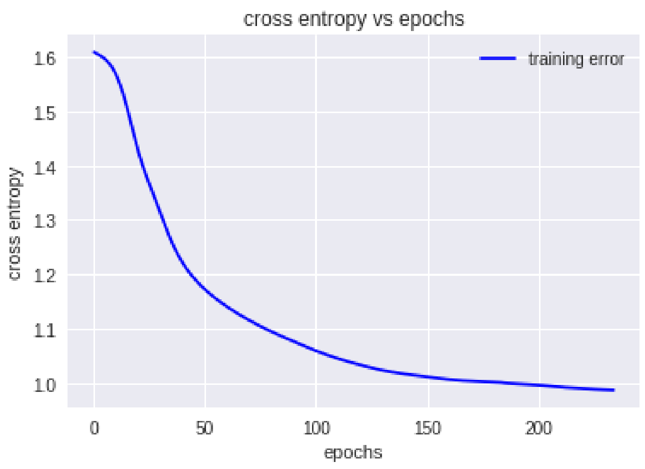
Confusion matrix for Test data

1.2 AANN

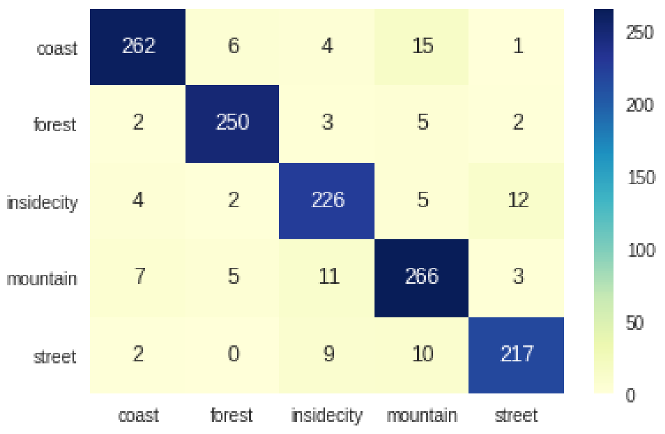
BottleNeck Layer Size	Reconstruction Error	Train Error	Train Accuracy	Test Error	Test Accuracy	Epochs
300	0.0339	1.012	89.45	1.228	66.84	171
275	0.0337	1.001	90.50	1.246	65.19	178
250	0.0343	1.002	90.21	1.216	68.53	177
225	0.0356	1.012	89.45	1.244	65.73	182
200	0.0375	1.008	89.89	1.245	65.16	184
175	0.0393	1.011	89.85	1.227	67.21	191
150	0.0409	1.009	89.91	1.218	68.08	193
125	0.0435	0.993	91.25	1.226	67.24	213
100	0.0481	0.993	91.27	1.218	68.04	218
75	0.0564	0.983	92.28	1.232	66.60	232
50	0.0755	0.994	91.21	1.213	68.37	270
25	0.1013	1.033	87.44	1.217	67.96	313
10	0.1249	1.069	84.02	1.262	63.66	484
5	0.1481	1.159	75.44	1.276	62.25	672



Reconstruction error for Bottleneck size=100

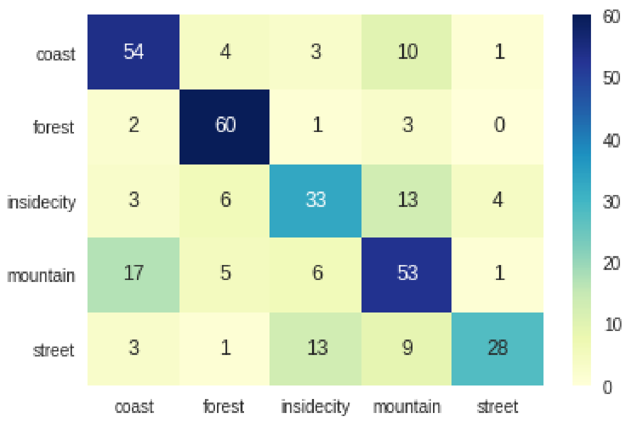


Training Error



	coast	forest	insidecity	mountain	street
coast	262	6	4	15	1
forest	2	250	3	5	2
insidecity	4	2	226	5	12
mountain	7	5	11	266	3
street	2	0	9	10	217

Confusion matrix for Train data



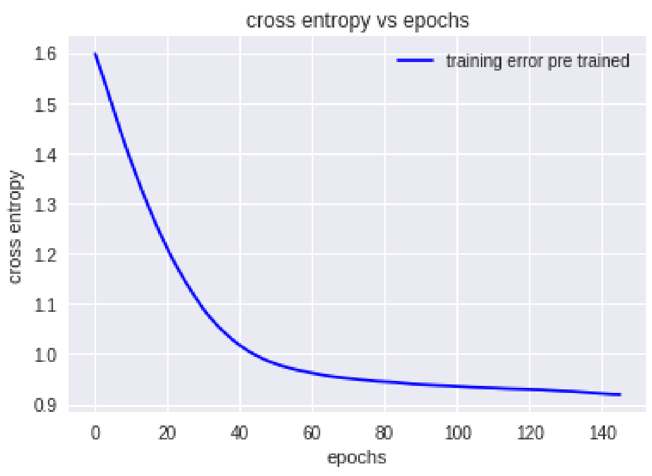
	coast	forest	insidecity	mountain	street
coast	54	4	3	10	1
forest	2	60	1	3	0
insidecity	3	6	33	13	4
mountain	17	5	6	53	1
street	3	1	13	9	28

Confusion matrix for Test data

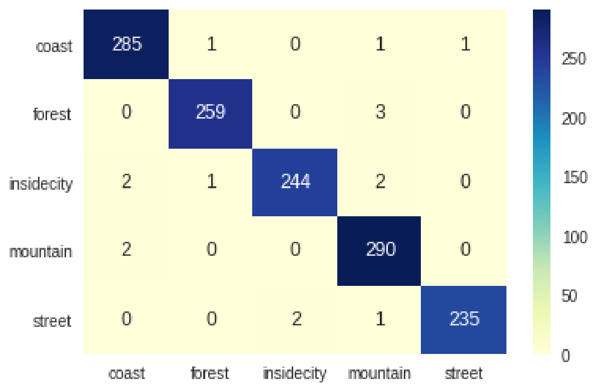
Task 2: Stacked Autoencoder based pre-training of DNN classifier for Dataset 1

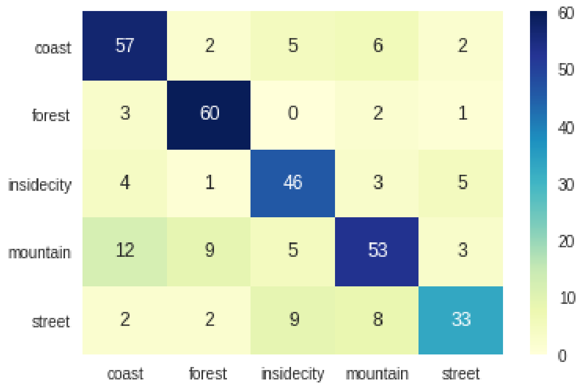
*Merged the weight matrices before and after non-linear layers of stacked autoencoders.

Bottleneck size 1	**Bottleneck size 2**	**Bottleneck size 3**		**Train Error**	**Train Accuracy**	**Test Error**	**Test Accuracy**	**Epochs**
150	73	35	**Pre-training**	0.9440	96.03	1.1829	70.59	97
			No Pre-training	1.0623	84.33	1.3215	59.35	131
100	49	24	**Pre-training**	0.9263	97.81	1.1482	76.62	135
			No Pre-training	1.0314	87.77	1.3432	56.91	209
50	24	11	**Pre-training**	0.9175	99.69	1.1655	74.47	233
			No Pre-training	1.1703	72.23	1.3353	55.85	250
25	12	5	**Pre-training**	0.9255	99.39	1.1753	73.87	455
			No Pre-training	1.4561	40.18	1.5182	33.33	198



Training Error for Bottleneck size 1 = 100 (Pre-trained)

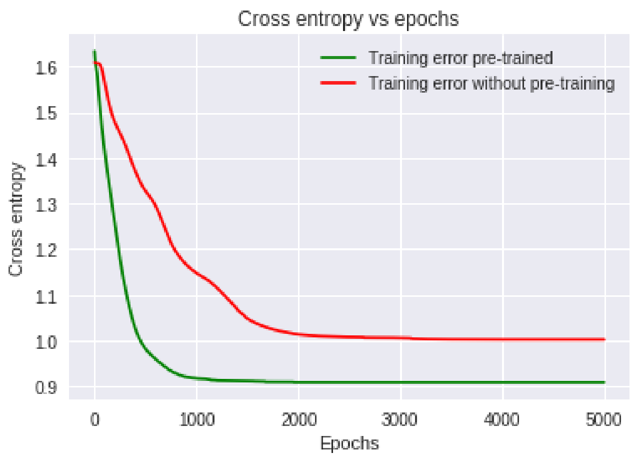


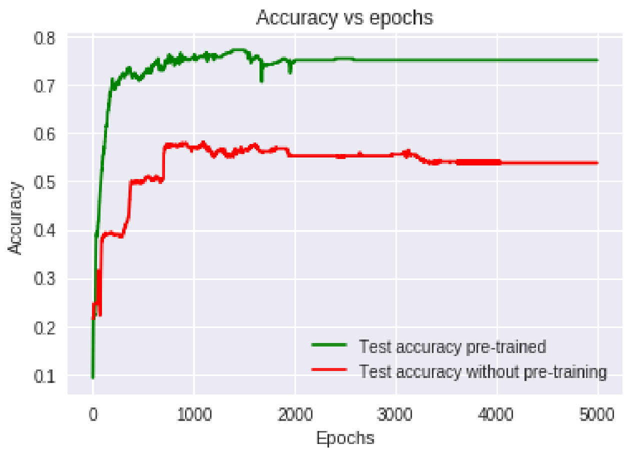


Confusion matrix for Train data

Confusion matrix for Test data

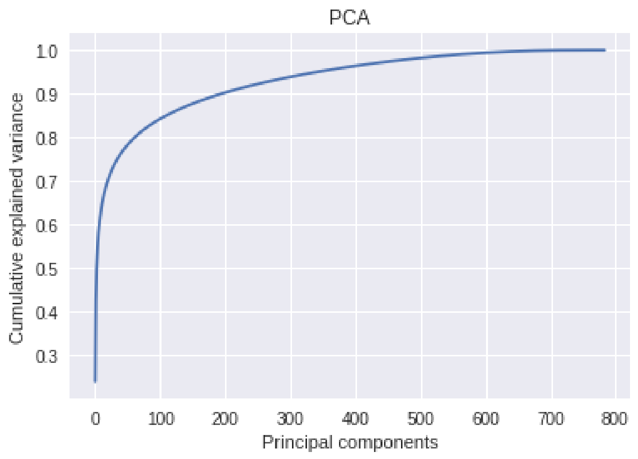
<u>Observation</u>:





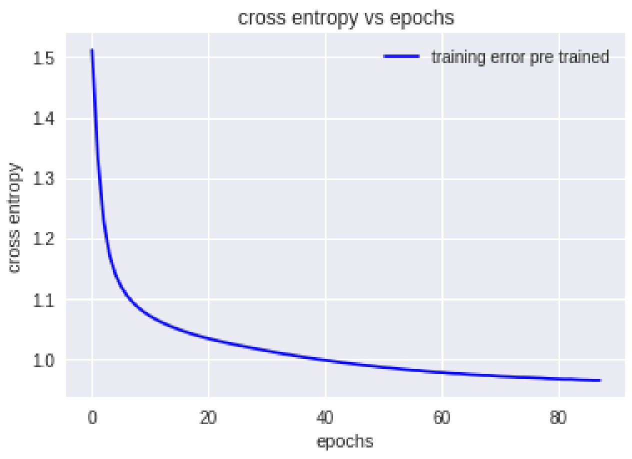
From the above two plots, we observe that Auto-encoder based pre-training model converges faster than models without pre-training.

Task 3: Stacked Autoencoder based pre-training of DNN classifier for Dataset 2

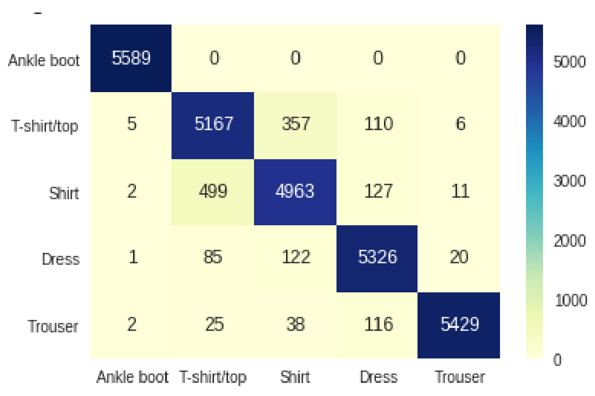


The number of Principal components from the curve as given by the cumulative explained variance ratio <=0.99 is 559.
Hence, we shall choose 500 as starting point for the bottleneck size of Stacked Autoencoder.
*Merged the weight matrices before and after non-linear layers of stacked autoencoders.

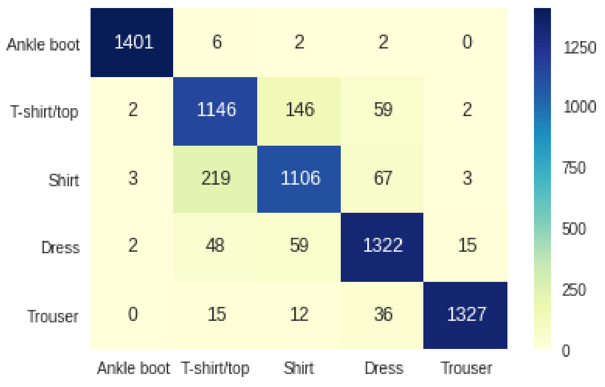
Bottleneck size 1	**Bottleneck size 2**	**Bottleneck size 3**		**Train Error**	**Train Accuracy**	**Test Error**	**Test Accuracy**	**Epochs**
150	73	35	**Pre-training**	0.9826	93.05	1.0864	89.67	61
			No Pre-training	0.9834	92.52	1.0139	89.14	73
100	49	24	**Pre-training**	0.9616	94.59	1.0038	90.12	89
			No Pre-training	0.9949	91.88	1.0242	88.68	68
50	24	11	**Pre-training**	0.9569	95.15	1.0083	90.12	139
			No Pre-training	1.0252	90.45	1.0486	88.19	118
25	12	5	**Pre-training**	0.9856	95.46	1.0385	89.45	163
			No Pre-training	1.1343	78.24	1.1553	76.13	150



Training Error for Bottleneck size 1 = 100 (Pre-trained)



	Ankle boot	T-shirt/top	Shirt	Dress	Trouser
Ankle boot	5589	0	0	0	0
T-shirt/top	5	5167	357	110	6
Shirt	2	499	4963	127	11
Dress	1	85	122	5326	20
Trouser	2	25	38	116	5429

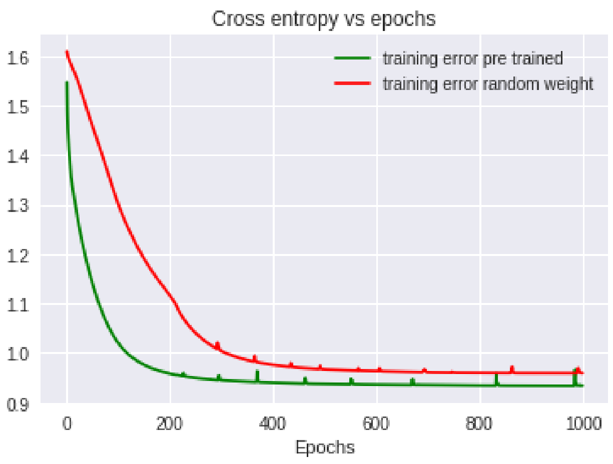


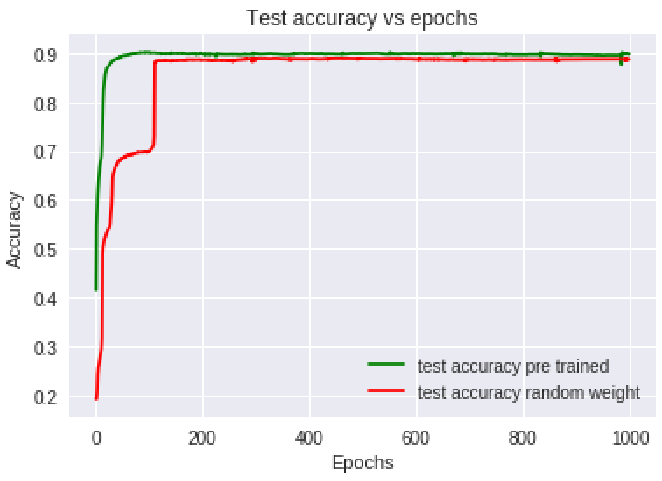
	Ankle boot	T-shirt/top	Shirt	Dress	Trouser
Ankle boot	1401	6	2	2	0
T-shirt/top	2	1146	146	59	2
Shirt	3	219	1106	67	3
Dress	2	48	59	1322	15
Trouser	0	15	12	36	1327

Confusion matrix for Train data

Confusion matrix for Test data

<u>Observation</u>:

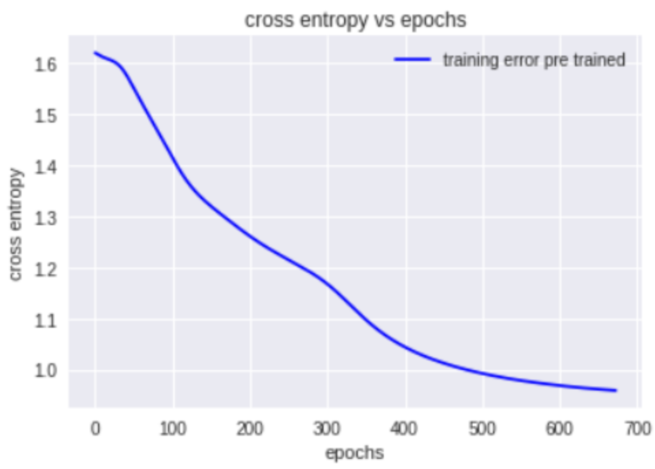




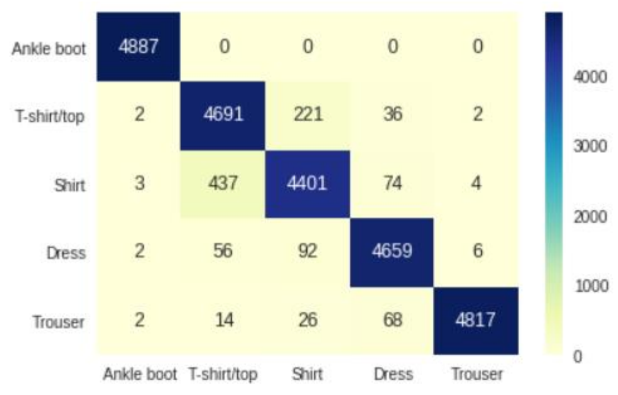
From the above two plots, we observe that Auto-encoder based pre-training model converges faster than models without pre-training.

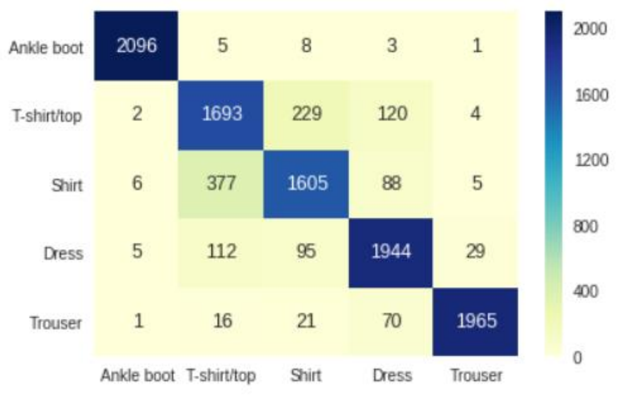
Task 4: Stacked RBM based pre-training of DNN classifier for Dataset 2 using Binary-Binary RBMs

Hidden Layer-1 size	Hidden Layer-2 size	Hidden Layer-3 size		Train Error	Train Accuracy	Test Error	Test Accuracy	Epochs
500	250	125	**Pre-training**	0.9735	93.50	1.0178	88.68	227
			No Pre-training	0.9765	93.18	1.0184	88.60	232
400	200	100	**Pre-training**	0.9735	93.53	1.0179	88.71	263
			No Pre-training	0.9764	93.32	1.0198	88.55	239
300	150	75	**Pre-training**	0.9660	94.30	1.0173	88.74	316
			No Pre-training	0.9781	93.22	1.0240	88.22	281
200	100	50	**Pre-training**	0.9589	95.04	1.0179	88.78	386
			No Pre-training	0.9659	94.36	1.0202	88.52	384
100	50	25	**Pre-training**	0.9550	95.83	1.0236	88.53	559
			No Pre-training	0.9586	95.67	1.0276	88.17	629
75	37	18	**Pre-training**	0.9588	95.73	1.0248	88.60	673
			No Pre-training	0.9616	96.49	1.0467	87.12	877



Training Error for Hidden layer size 1 = 75 (Pre-trained)

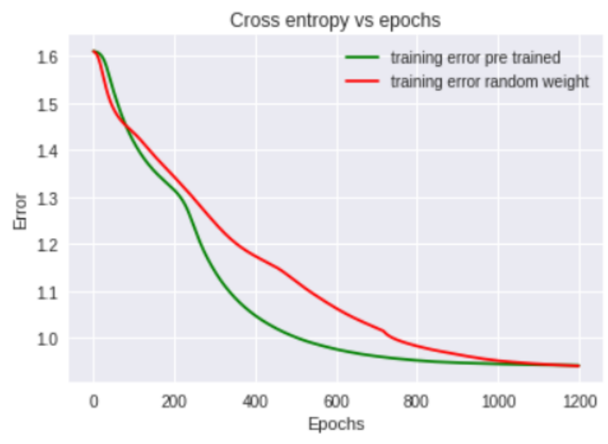


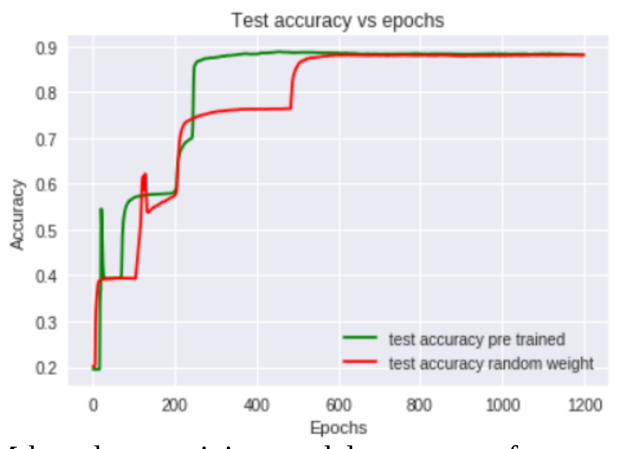


Confusion matrix for Train data

Confusion matrix for Test data

Observation:





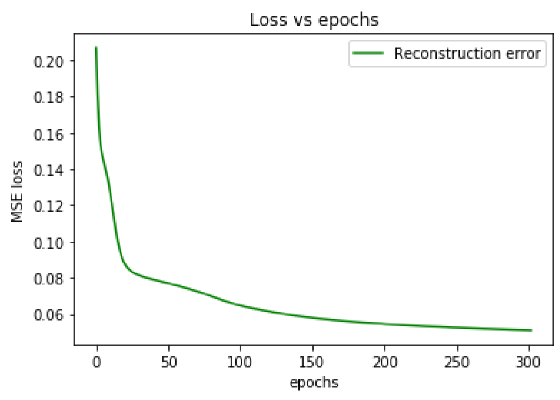
From the above two plots, we observe that RBM based pre-training model converges faster than models without pre-training.

Task 5: Stacked Denoising Autoencoder for noisy data generated using images in Dataset 2

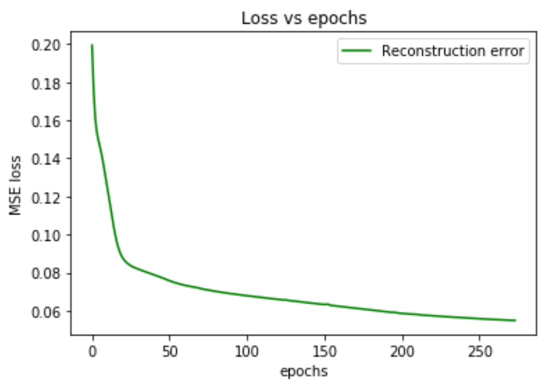
*Used Stacked autoencoder image denoising instead of single autoencoder
*Used Sum squared error instead of BCE

Bottleneck layer 1 size = 100, Bottleneck layer 1 size = 49, Bottleneck layer 1 size = 24

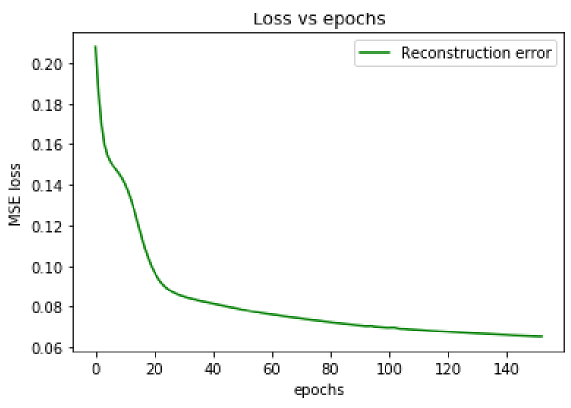
Noise Percentage (%)	Training Reconstruction Error	Test Reconstruction Error	Epochs
20	0.05099	0.05411	303
30	0.05482	0.05929	274
40	0.06525	0.06974	153
50	0.07981	0.08608	98
60	0.09800	0.12924	120



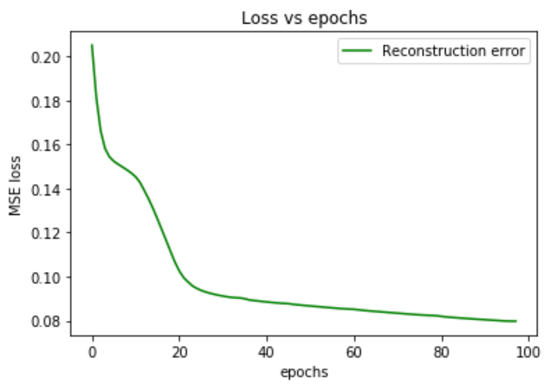
Reconstruction error for 20% noisy data



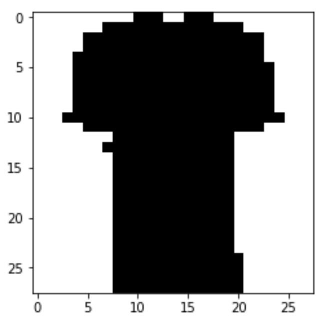
Reconstruction error for 30% noisy data

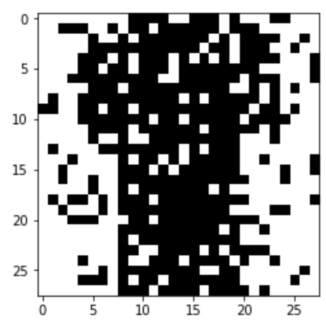


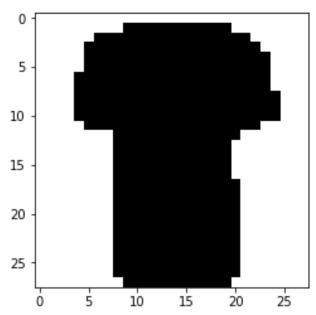
Reconstruction error for 40% noisy data

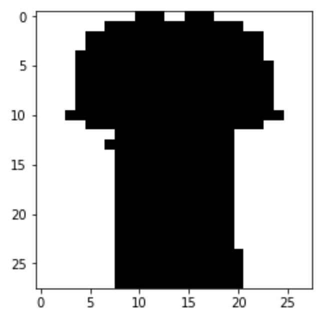


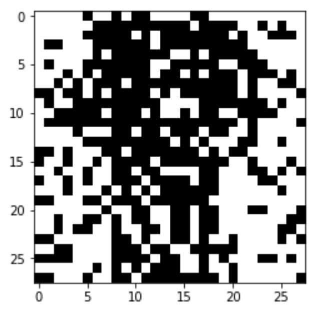
Reconstruction error for 50% noisy data

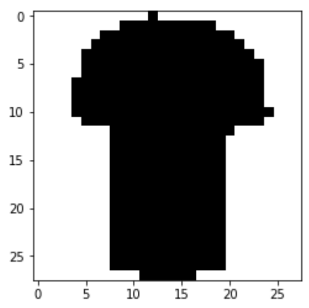

Original Image

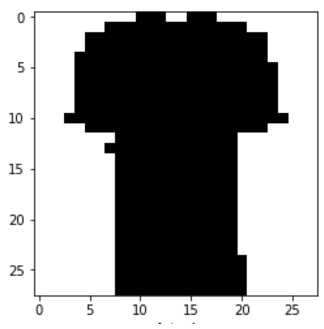

Noisy image (20%)

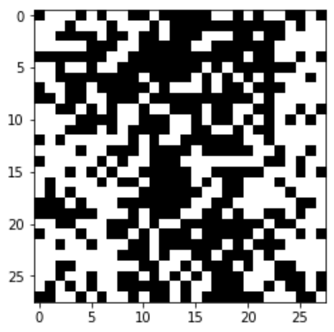

De-noised image

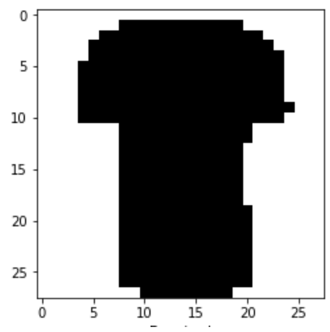

Original Image

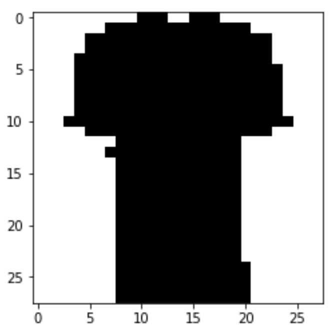

Noisy image (30%)

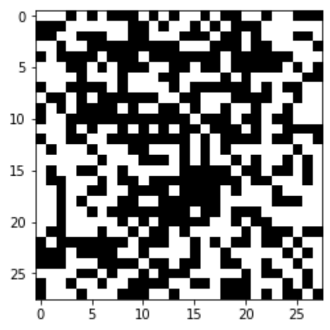

De-noised image

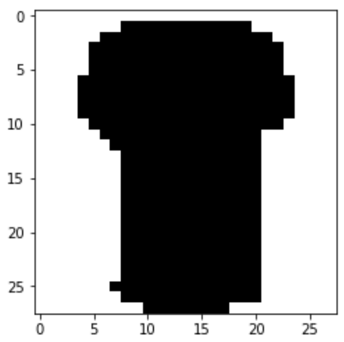

Original Image


Noisy image (40%)


De-noised image


Original Image


Noisy image (50%)


De-noised image