DEPARTMENT OF COMPUTER
ENGINEERING AND IT

AMIRKABIR UNIVERSITY
OF TECHNOLOGY

Homework 3 for Artificial Neural Network

Prof. Safabakhsh

Milad Bohlouli

97131069

1397-1398

Part 1:

a. In this part we will focus on the functionality of the SOM[1] networks, thus SOM will be applied to a dataset to realize how well clustering could be accomplished using SOMs in different policies.

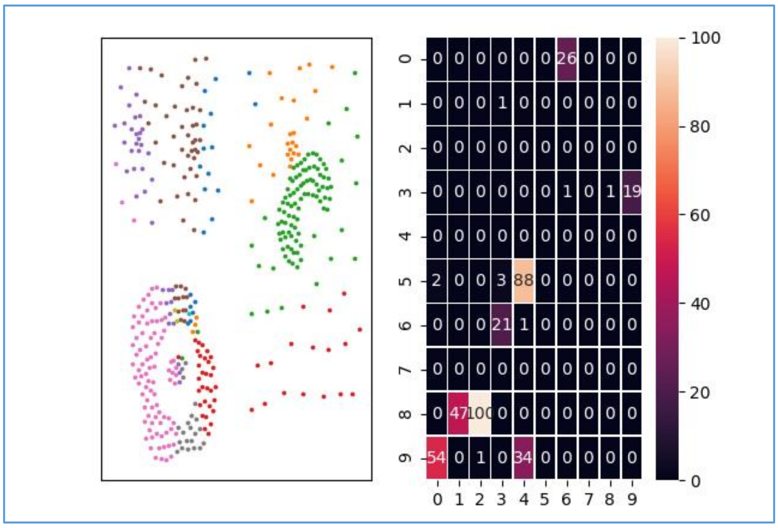
■ Clustering using SOMs but different neighborhood shape policies. Tested shapes:

{ circular, rectangular, hexagonal }

In all of the following tests, all the parameters except the shape of the neighborhood are as below:

Table 1 Parameter definition of the test

Size	Initial radius	Learning rate	Learning rate update policy	Neighbor Weight policy	Radius updating policy
[10, 10]	9	0.1	Exponential	Exponential	Exponential



	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	26	0	0	0
1	0	0	0	1	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	1	19
4	0	0	0	0	0	0	0	0	0	0
5	2	0	0	3	88	0	0	0	0	0
6	0	0	0	21	1	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	47	100	0	0	0	0	0	0	0
9	54	0	1	0	34	0	0	0	0	0

Figure 1 SOM using circular neighborhood shape

[1] Self Organizing Maps

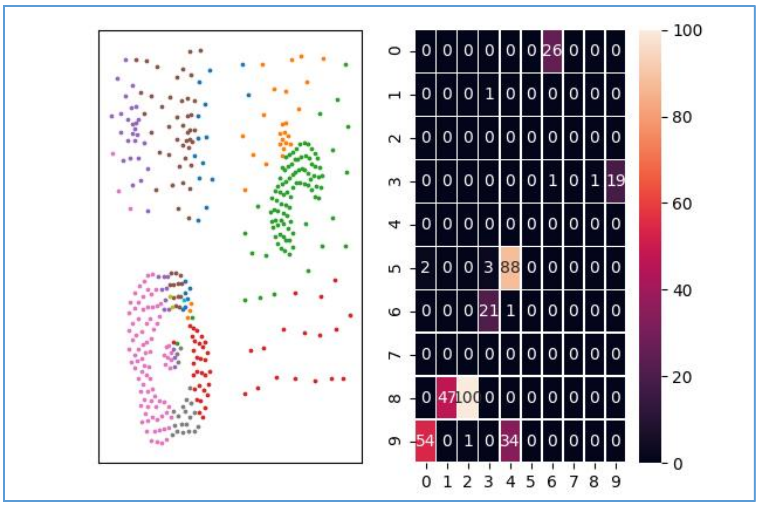


Figure 2 SOM using rectangular neighborhood shape

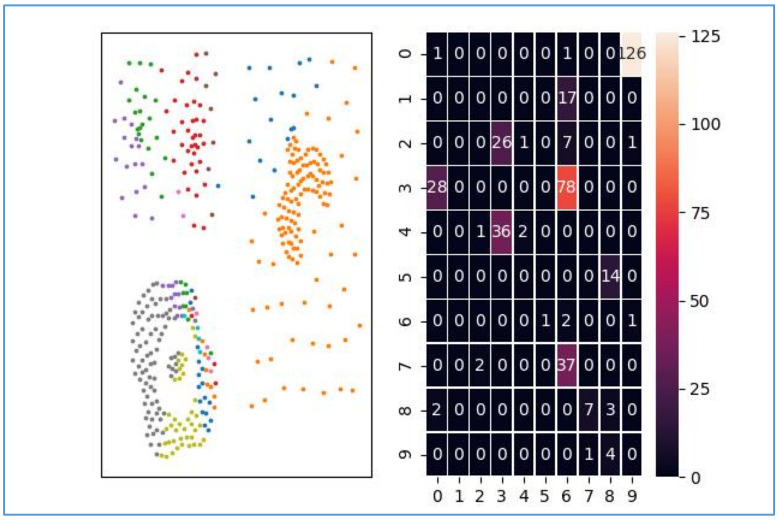


Figure 3 SOM using hexagonal neighborhood shape

The final comparison among the different policies for neighborhood shape is depicted in Image 4.

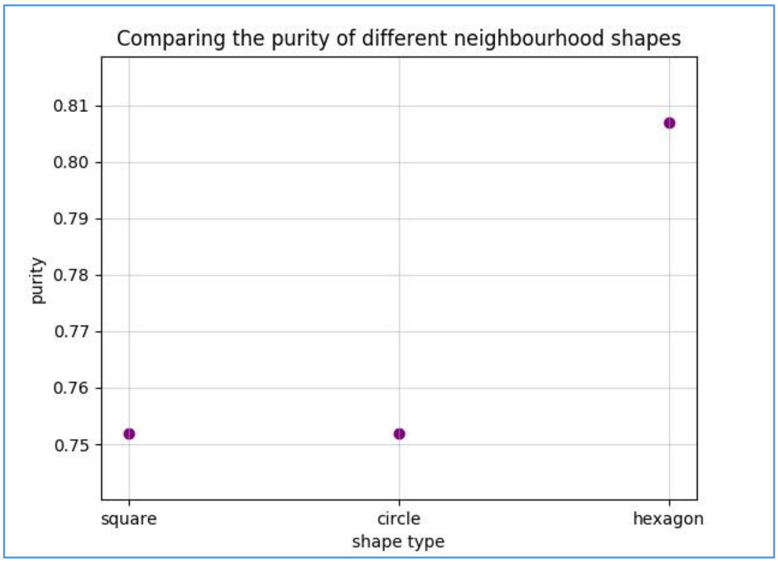


Figure 4 Final comparison among different shapes

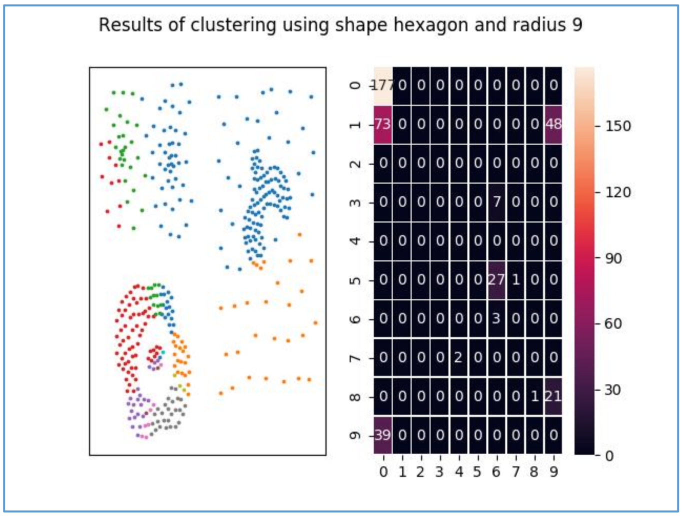
- Clustering using SOMs but different neighborhood radius policies. Tested policies:

{ undetectable radius, detectable radius{2, 4, 6} }

The radius update policy will be set to constant while we are trying to check the effect of neighborhood width on the results of clustering. All of the parameters that are set to this model is shown in table 2. As it can be seen from the this table, the neighborhood shape is set to hexagonal, because it has surpassed other shapes according to the results of the previous section shown in image 4.

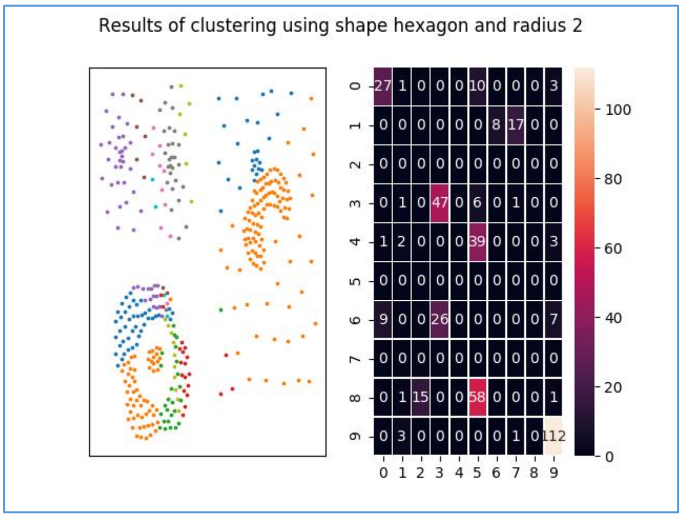
Table 2 Parameter definition of the test

Size	Learning rate	Learning rate update policy	Neighbor Weight policy	Radius updating policy	Neighborhood shape
[10, 10]	0.1	Exponential	Exponential	constant	hexagonal


Results of clustering using shape hexagon and radius 9

	0	1	2	3	4	5	6	7	8	9
0	177	0	0	0	0	0	0	0	0	0
1	73	0	0	0	0	0	0	0	0	48
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	7	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	27	1	0	0
6	0	0	0	0	0	0	3	0	0	0
7	0	0	0	0	2	0	0	0	0	0
8	0	0	0	0	0	0	0	0	1	21
9	39	0	0	0	0	0	0	0	0	0

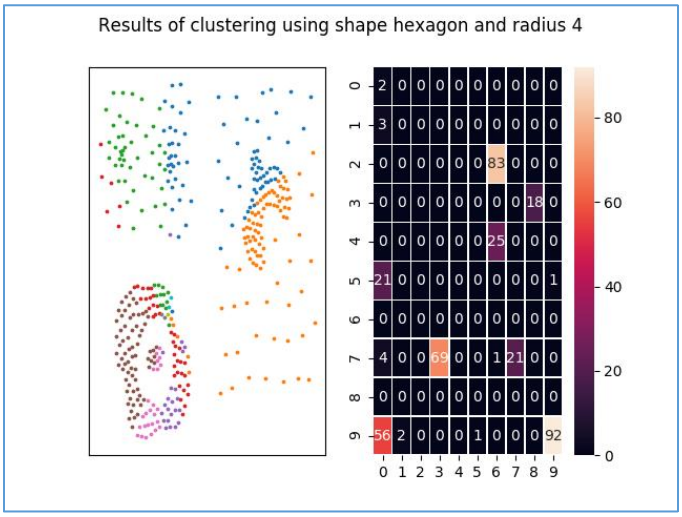
Figure 5 unbounded radius while training



Results of clustering using shape hexagon and radius 2

	0	1	2	3	4	5	6	7	8	9
0	27	1	0	0	0	10	0	0	0	3
1	0	0	0	0	0	0	8	17	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	1	0	47	0	6	0	1	0	0
4	1	2	0	0	0	39	0	0	0	3
5	0	0	0	0	0	0	0	0	0	0
6	9	0	0	26	0	0	0	0	0	7
7	0	0	0	0	0	0	0	0	0	0
8	0	1	15	0	0	58	0	0	0	1
9	0	3	0	0	0	0	0	1	0	112

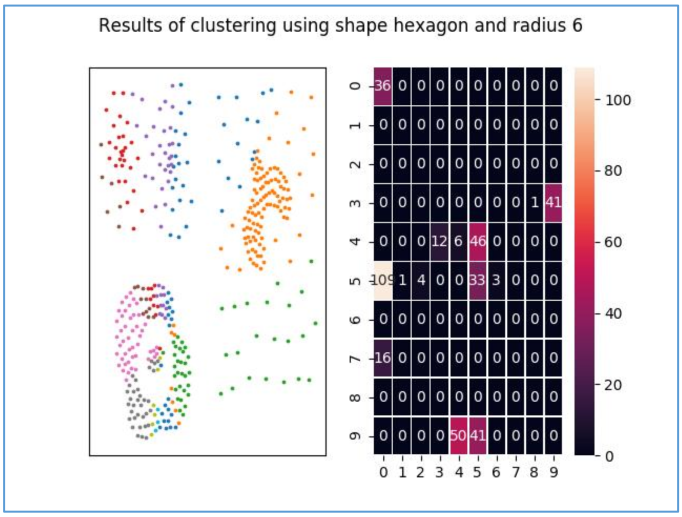
Figure 6 SOM with radius = 2



Results of clustering using shape hexagon and radius 4

	0	1	2	3	4	5	6	7	8	9
0	2	0	0	0	0	0	0	0	0	0
1	3	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	83	0	0	0
3	0	0	0	0	0	0	0	0	18	0
4	0	0	0	0	0	0	25	0	0	0
5	21	0	0	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0	0	0	0
7	4	0	0	69	0	0	1	21	0	0
8	0	0	0	0	0	0	0	0	0	0
9	56	2	0	0	0	1	0	0	0	92

Figure 7 SOM with radius = 4



Results of clustering using shape hexagon and radius 6

	0	1	2	3	4	5	6	7	8	9
0	36	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	41
4	0	0	0	12	6	46	0	0	0	0
5	109	1	4	0	0	33	3	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	16	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	50	41	0	0	0	0

Figure 8 SOM with radius = 6

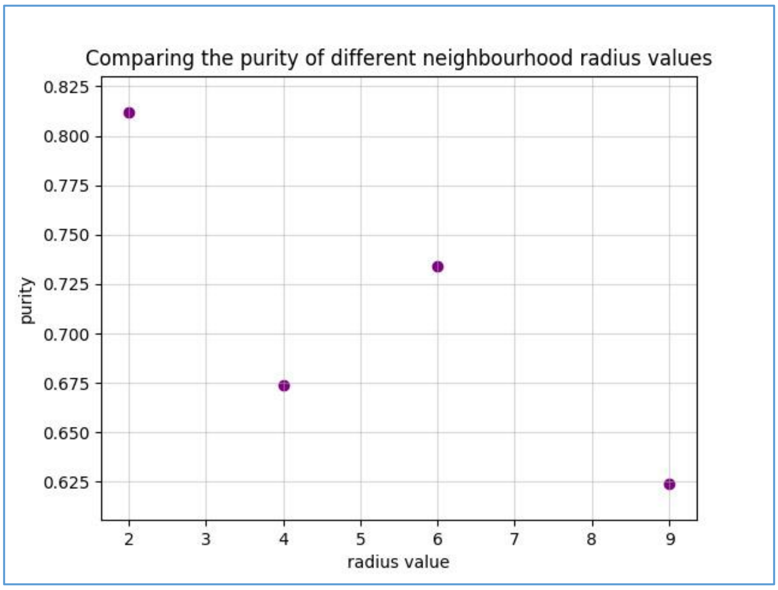


Figure 9 Different radius values comparison

b. Using GSOM[2] for clustering the 2D dataset. The growing algorithm and the initial conditions will be explained. SOMs are a decent solution for clustering but they use too much of neurons while there are just a few clusters. One way to tackle this problem is to benefit Growing networks. An initial number of neurons will be defined and will then be expanded in a way to reduce the error rate. These are called GSOMs.

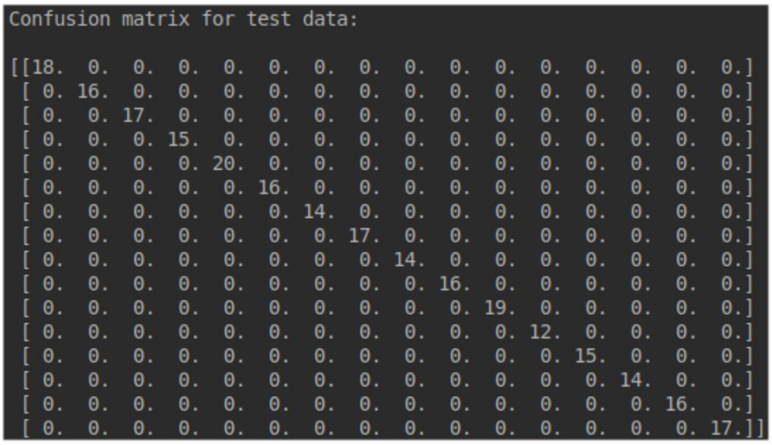
In this section a GSOM is implemented and tested on 2D dataset. Main goal of this sections is to compare the efficiency and the performance of SOMs with GSOMs.

[2] Growing SOM

Part 2:

The main focus of this section is on using SOMs for dimension reduction for datasets with too much dimensions. One other approach that could be used is PCA[3]. To compare the efficiency of both these methods. In one part SOM and in another part PCA will be used to reduce the dimensions of a dataset with 1024 dimensions in 16 classes.

- At first a FFNN[4] is used to classify the classes of the dataset. The Accuracy and confusion matrix of the are depicted in the next Figures. The training of the model took 50.4 seconds.



```
Confusion matrix for test data:

[[18.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0. 16.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0. 17.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0. 15.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0. 20.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0. 16.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0. 14.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0. 17.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0. 14.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0. 16.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 19.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 12.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 15.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 14.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 16.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 17.]]
```

Figure 10 confusion matrix for test data without dimension reduction

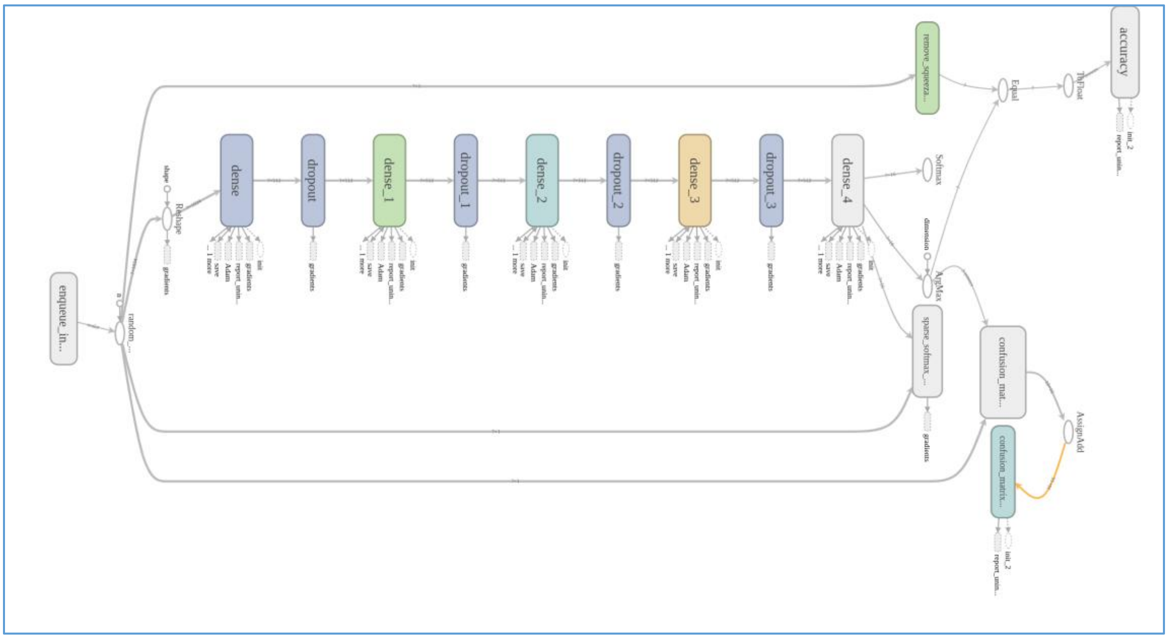


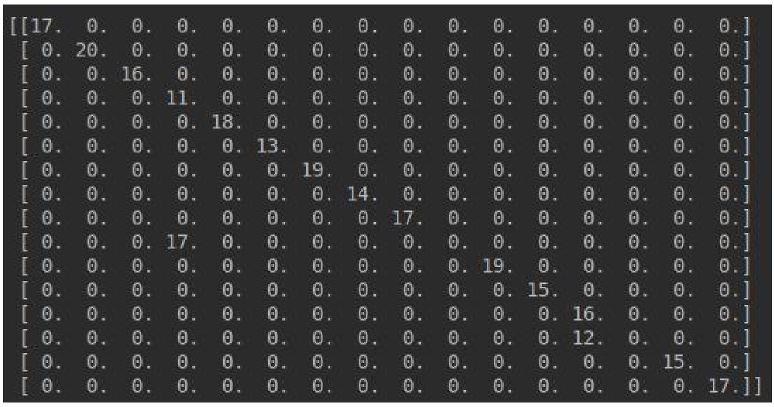
Figure 11 The FFNN model which has been used for classification

As it can be inferred from figure 10 the test accuracy is 100% for the test data.

[3] Principle Component Analysis

[4] FeedForward Neural Network

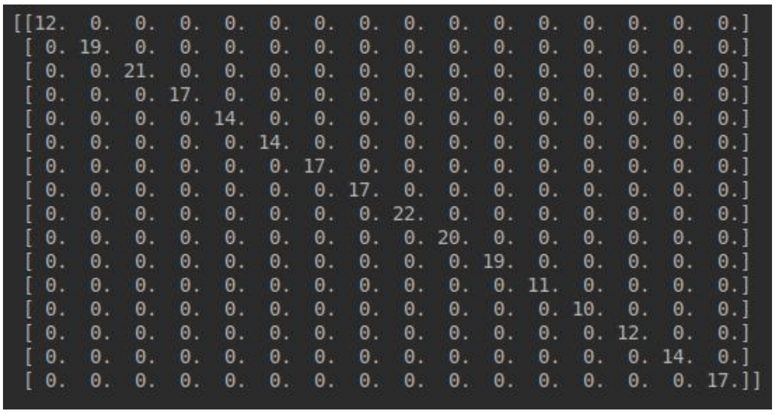
- In this section we will use PCA to reduce the dimensions of the dataset into 3 different dimensions(2, 4, 6) and classify the new dataset using FFNN. In figure 12, 13 and 14 the dimensions of the dataset has been reduces into 2, 3 and 4 respectively. The confusion matrix for each case has been depicted and also the accuracy of the predictions for the test data and the training time of each case are gathered in table 3.



```
[[17.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0. 20.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0. 16.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0. 11.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0. 18.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0. 13.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0. 19.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0. 14.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0. 17.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0. 17.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 19.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 15.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 16.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 12.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 15.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 17.]]
```

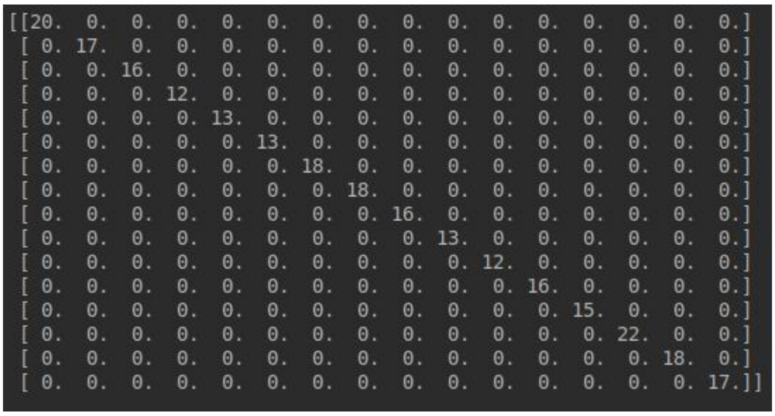
,

Figure 12 Confusion matrix for test data after dimensional reduction using PCA into 2 dimensions



```
[[12.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0. 19.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0. 21.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0. 17.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0. 14.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0. 14.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0. 17.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0. 17.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0. 22.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0. 20.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 19.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 11.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 10.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 12.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 14.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 17.]]
```

Figure 13 Confusion matrix for test data after dimensional reduction using PCA into 3 dimensions



```
[[20.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0. 17.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0. 16.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0. 12.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0. 13.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0. 13.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0. 18.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0. 18.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0. 16.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0. 13.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 12.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 16.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 15.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 22.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 18.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 17.]]
```

Figure 14 Confusion matrix for test data after dimensional reduction using PCA into 4 dimensions

As it can be interpreted from table 3, as the number of the dimensions increase, the accuracy of the predictions for the test data increases as well as the training time that also increases.

Table 3 Comparison between different dimensions after reductions using PCA

Dimensions	1024	2	3	4
Training time	50.24	25.8	26.1	28.0
Test accuracy (percent)	100	88.85	100	100

- In this sections the previous dataset with 1024 dimensions will be compressed into 2 dimensions. The results are depicted in a 2D scatter plot in figure 15.

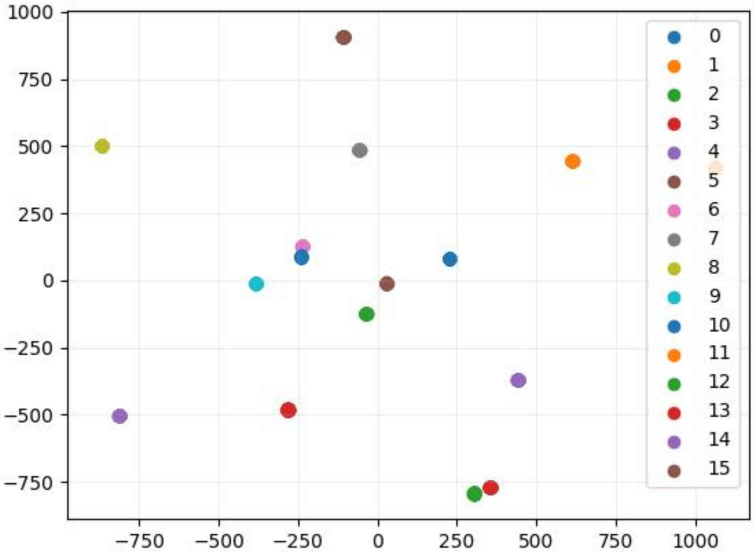
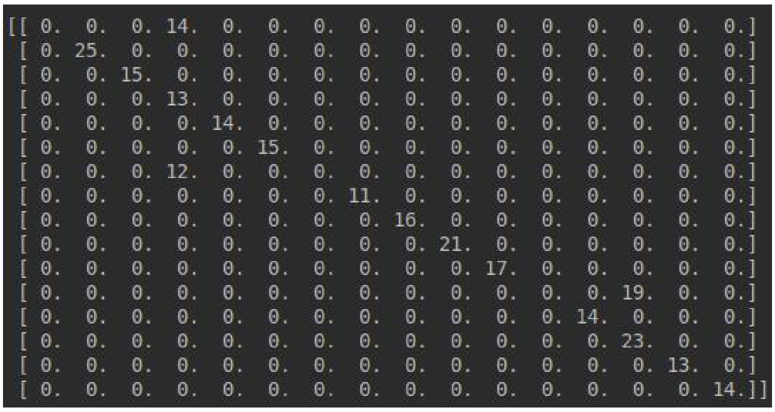


Figure 15 Representation of the data after dimension reduction using PCA

- In this section the dimension reduction will be accomplished using SOM networks. Then the new data will be feed to a FFNN for classification to be compared with PCA. The final results of classification including the confusion matrix and the accuracy of the predictions on test data are depicted as below.



```
[[ 0.  0.  0. 14.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0. 25.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0. 15.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0. 13.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0. 14.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0. 15.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0. 12.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0. 11.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0. 16.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0. 21.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 17.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 19.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 14.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 23.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 13.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. 14.]]
```

Figure 16 Confusion matrix for the test data classification after dimension reduction using SOM

Table 4 Comparison between dimension reduction using PCA or SOM

Dimension reduction to 2D	PCA	SOM
Test accuracy (percent)	88.85	82.69

As it can be inferred from table 4 SOM has worked better in dimension reduction. Another problem is the many parameters that has to be set for the SOM to work. The reason that PCA has surpassed SOM could be because of bad initialization of the hyper-parameters of the SOM.

- As the final section the compressed data will be represented according to different classes.

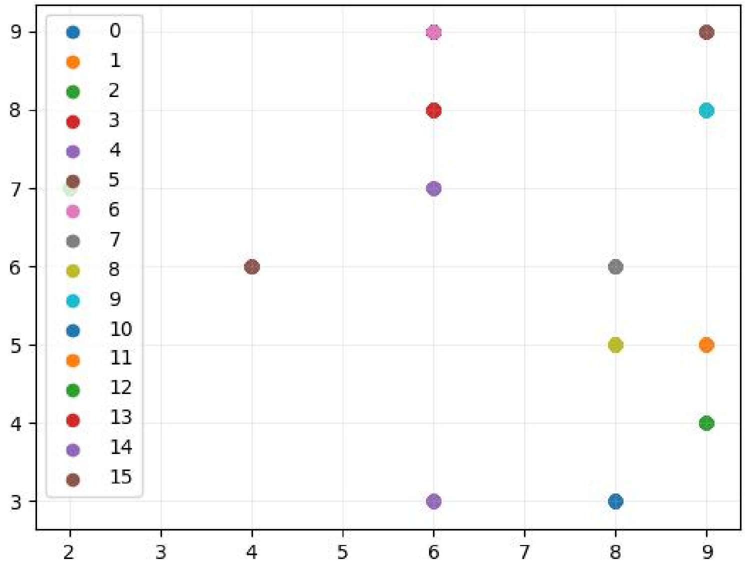


Figure 17 Representation of data after dimension reduction using SOM

If we compare the representation of the data in figure 15 and figure 17, it can be seen that different classes has been separated in a better manner using SOM rather than PCA. But despite this fact, the results of classification after reduction using PCA is better compared to SOMs as shown in Table4.