# Evaluation of SOM, SVM and MLP on Assignment's given data

Wang Yushuo · School of EEE · Nanyang Technological University, Singapore
Email: WANG1533@e.ntu.edu.sg

**Abstract:**

A brief report of the experiment on Self-organizing map(SOM), Support vector machine(SVM) and a two-fully connected layer network(MLP) with back-propagation.
**code is available at: https://github.com/LeslieWongCV/EE7207*

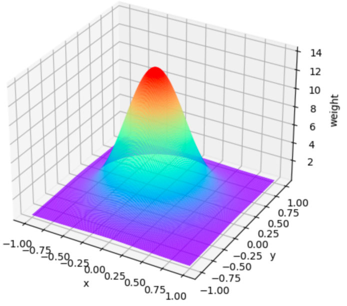
## I. Data

The dataset you gave us contains 330 training samples, 21 testing samples, each sample is a vector with length of 33. There are two classes. Label of the data is [-1 , 1].

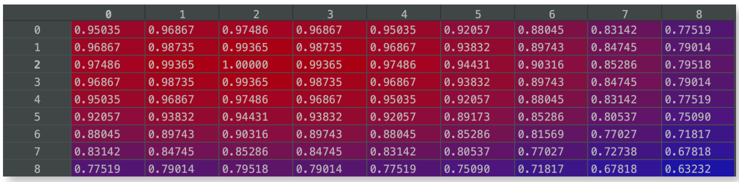
## II. Self-organizing map

### A. *Structure*

a. num_neural: $\text{int}(5\sqrt{N}) = 81$ , $N = 330$
b. num_iteration: 500 * num_neural = 40,500
c. sigma: 3, (at the iteration t we have sigma(t) = sigma / (1 + t/T), where T = num_iteration/2
d. learning_rate: 0.5, (at the iteration t we have learning_rate(t) = learning_rate / (1 + t/T) )
e. Neighborhood Function: Gaussian distribution with sigma=1 and sigma=5

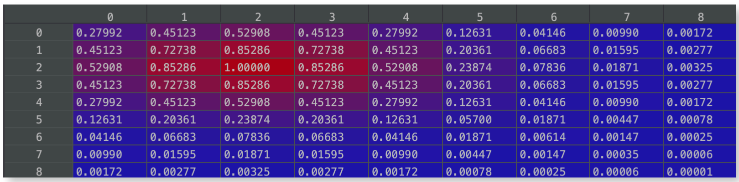


Neighborhood Function



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.95035 | 0.96867 | 0.97486 | 0.96867 | 0.95035 | 0.92057 | 0.88045 | 0.83142 | 0.77519 |
| 1 | 0.96867 | 0.98735 | 0.99365 | 0.98735 | 0.96867 | 0.93832 | 0.89743 | 0.84745 | 0.79014 |
| 2 | 0.97486 | 0.99365 | 1.00000 | 0.99365 | 0.97486 | 0.94431 | 0.90316 | 0.85286 | 0.79518 |
| 3 | 0.96867 | 0.98735 | 0.99365 | 0.98735 | 0.96867 | 0.93832 | 0.89743 | 0.84745 | 0.79014 |
| 4 | 0.95035 | 0.96867 | 0.97486 | 0.96867 | 0.95035 | 0.92057 | 0.88045 | 0.83142 | 0.77519 |
| 5 | 0.92057 | 0.93832 | 0.94431 | 0.93832 | 0.92057 | 0.89173 | 0.85286 | 0.80537 | 0.75090 |
| 6 | 0.88045 | 0.89743 | 0.90316 | 0.89743 | 0.88045 | 0.85286 | 0.81569 | 0.77027 | 0.71817 |
| 7 | 0.83142 | 0.84745 | 0.85286 | 0.84745 | 0.83142 | 0.80537 | 0.77027 | 0.72738 | 0.67818 |
| 8 | 0.77519 | 0.79014 | 0.79518 | 0.79014 | 0.77519 | 0.75090 | 0.71817 | 0.67818 | 0.63232 |

Sigma=5



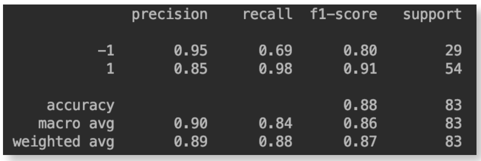
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.27992 | 0.45123 | 0.52908 | 0.45123 | 0.27992 | 0.12631 | 0.04146 | 0.00990 | 0.00172 |
| 1 | 0.45123 | 0.72738 | 0.85286 | 0.72738 | 0.45123 | 0.20361 | 0.06683 | 0.01595 | 0.00277 |
| 2 | 0.52908 | 0.85286 | 1.00000 | 0.85286 | 0.52908 | 0.23874 | 0.07836 | 0.01871 | 0.00325 |
| 3 | 0.45123 | 0.72738 | 0.85286 | 0.72738 | 0.45123 | 0.20361 | 0.06683 | 0.01595 | 0.00277 |
| 4 | 0.27992 | 0.45123 | 0.52908 | 0.45123 | 0.27992 | 0.12631 | 0.04146 | 0.00990 | 0.00172 |
| 5 | 0.12631 | 0.20361 | 0.23874 | 0.20361 | 0.12631 | 0.05700 | 0.01871 | 0.00447 | 0.00078 |
| 6 | 0.04146 | 0.06683 | 0.07836 | 0.06683 | 0.04146 | 0.01871 | 0.00614 | 0.00147 | 0.00025 |
| 7 | 0.00990 | 0.01595 | 0.01871 | 0.01595 | 0.00990 | 0.00447 | 0.00147 | 0.00035 | 0.00006 |
| 8 | 0.00172 | 0.00277 | 0.00325 | 0.00277 | 0.00172 | 0.00078 | 0.00025 | 0.00006 | 0.00001 |

Sigma=1

*Figure.1 Neighborhood Function Visualization*

f. Weight initializing: Picking random samples from data.

### *B. Results*



```
              precision    recall  f1-score   support

          -1       0.95      0.69      0.80        29
           1       0.85      0.98      0.91        54

    accuracy                           0.88        83
   macro avg       0.90      0.84      0.86        83
weighted avg       0.89      0.88      0.87        83
```

Accuracy of testing set = 88%

### *C. Visualizing Data*

To visualize the result of the training I plot the distance map using a pseudo color where the neurons of the maps are displayed as an array of cells and the color represents the (weights) distance from the neighbor neurons. On top of the pseudo color I add markers that represent the samples mapped in the specific cells:

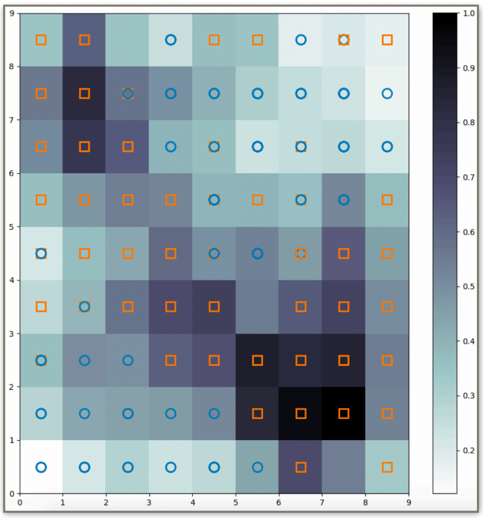


*Figure.2 Distance map and samples mapping*

To have an overview of how the samples are distributed across the map a scatter chart can be used where each dot represents the coordinates of the winning neuron. A random offset is added to avoid overlaps between points within the same cell.

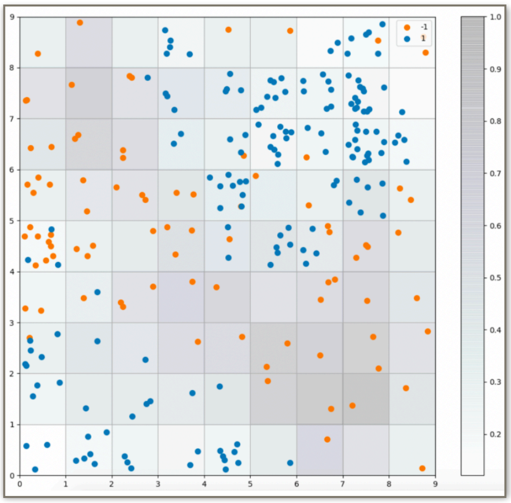


*Figure.3 Distance map and samples distribution*

Because this is a supervised problem, we can visualize the proportion of samples per class falling in a specific neuron using a pie chart per neuron:

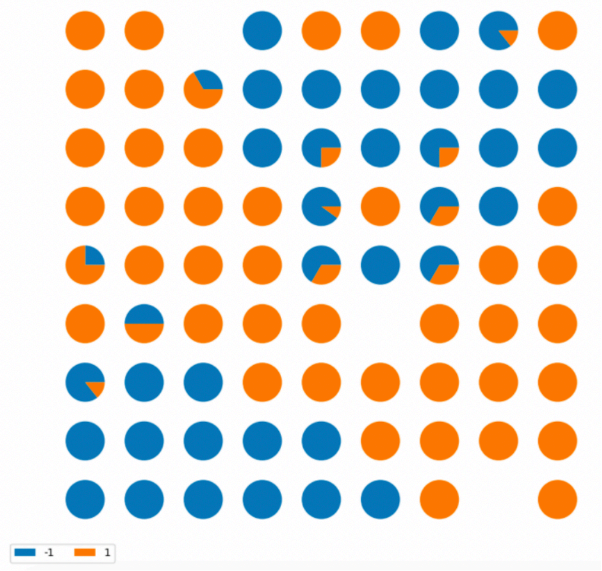


*Figure.3 Proportion of samples of each neuron*

## *III. Support vector machine*

### *A. Structure*

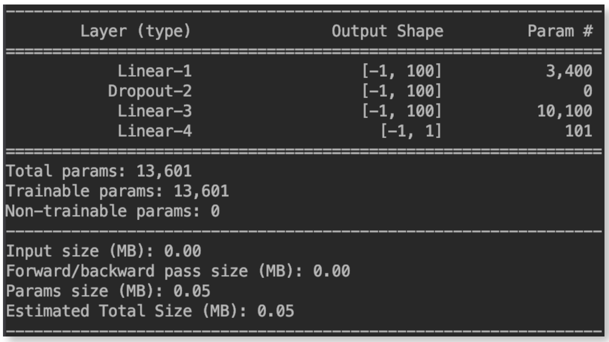
a. C = 1.0
b. Kernel: Radial basis function kernel
c. Gamma = 0.1

### *B. Results*

Acc of the training set = 96.26%.
Acc of the testing set = 93.96%.

## IV. Multilayer Perceptron

### *A. Structure*



```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Linear-1                 [-1, 100]           3,400
           Dropout-2                 [-1, 100]               0
            Linear-3                 [-1, 100]          10,100
            Linear-4                   [-1, 1]             101
================================================================
Total params: 13,601
Trainable params: 13,601
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.00
Params size (MB): 0.05
Estimated Total Size (MB): 0.05
----------------------------------------------------------------
```

*Figure.4 Network Structure*

b. Loss: MSE
c. Optimizer: SGD
d. Dropout rate: 0.5

### *B. Results*

I ran for 30,000 epochs in total, and this is the result:

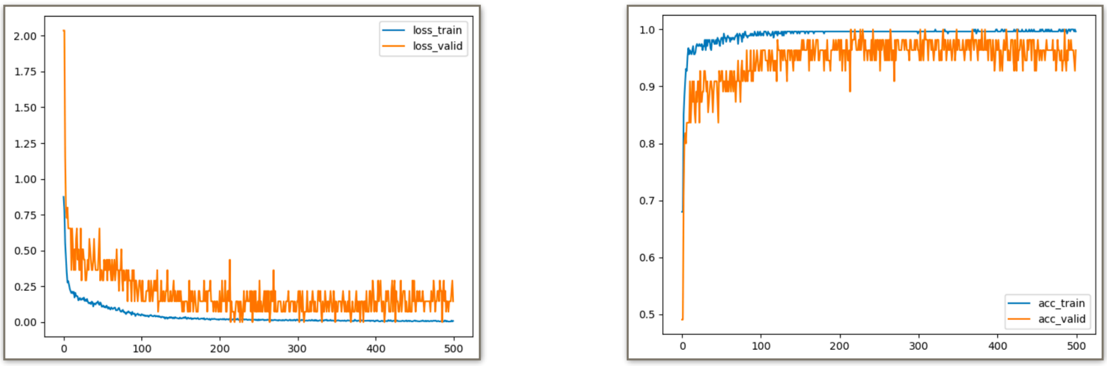


*Figure.5 Result of the MLP*

Acc of the training set = 99.64%.
Acc of the testing set = 96.36%.

## V. Conclusion

| | SOM | SVM | MLP |
|---|---|---|---|
| Acc | 88.0% | 94.0% | 96.4% |

*Figure.6 results comparison*

a. SOM can be used for classification by doing: for each neural we store the label of the classes that appeared most frequently in the training process.

b. MLP has the best performance while it needs more time for training.

* This is not an assignment.