EE5904 Neural Networks

Homework 2

Liu Xingyu

A0116430W

National University of Singapore

Electrical and Computer Engineering

29 Feb 2020

Q1. Rosenbrock’s Valley Problem (10 Marks)

Consider the Rosenbrock's Valley function:

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2$$

Gradient vector G = $\begin{bmatrix} 400x^3 - 400xy + 2x - 2 \\ 200y - 200x^2 \end{bmatrix}$

Hessian matrix H = $\begin{bmatrix} 1200x^2 - 400y + 2 & -400x \\ -400x & 200 \end{bmatrix}$

a). Steepest (Gradient) descent method

w(k+1)=w(k)- ηg(k)

With learning rate η = 0.001, please refer to Figure 1a for the trajectory of y and x and the function value versus iterations. The function value will converge to the global minimum (criterion f < =0.0001). It will take 8507 iterations.

Output of code:

```
The training ends at 8507.000000 iteration and when function value
is less than 0.0001
The final x is 0.990012 and y is 0.980084 and function value is
0.000100
```

With learning rate η = 0.2, please refer to Figure 1b for the trajectory of y and x and the function value versus iterations. We can see that with a larger learning 0.2, the function will never converge to global minimum, it will not converge and will diverge to infinity.

Output of code:

```
The training ends at 8.000000 iteration and when function value will
diverge to infinity when learning rate is 0.2
```

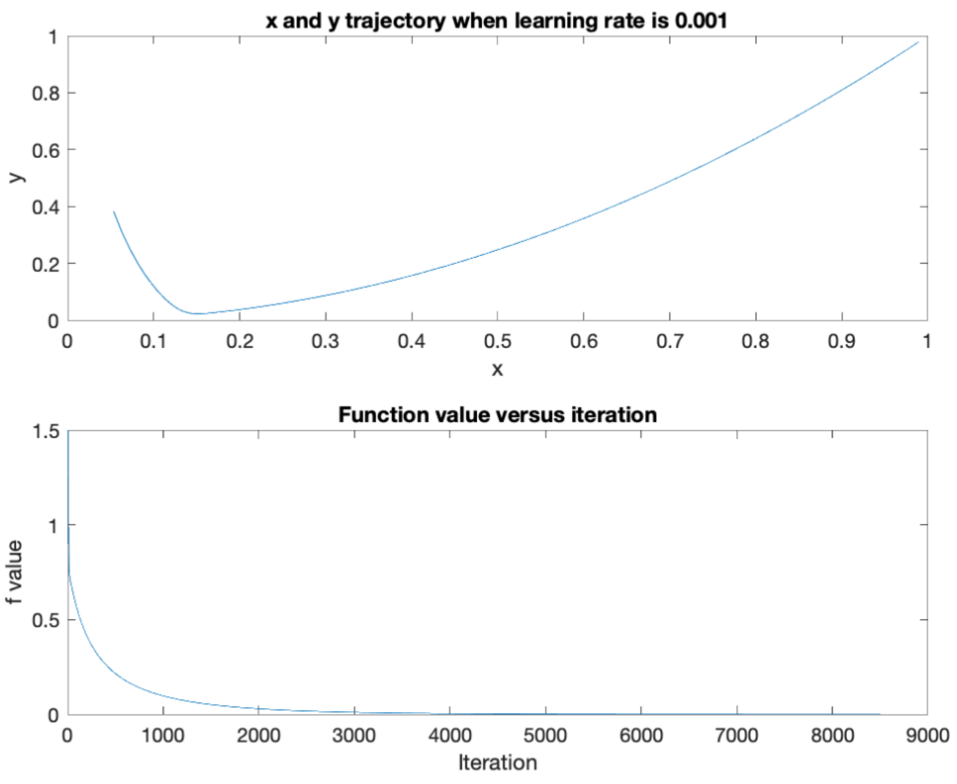


Figure 1a

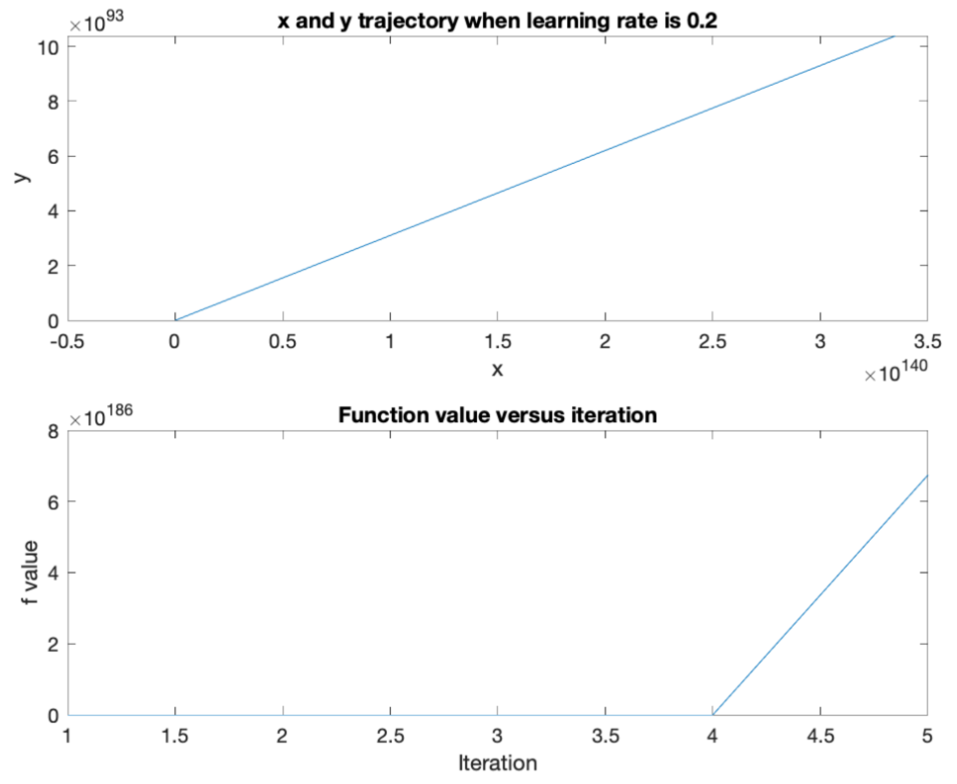


Figure 1b

b). Newton's method

$\Delta w(n) = -H^{-1}(n)g(n)$

Refer to Figure 1c, we can see that the value of x and y will converge to 1 when the function value reaches its global minimum. Not like gradient descent method, the newton's method will only take 6 iterations to reach its global minimum (same criterion f <=0.0001) which is much faster than gradient descent (8507 iterations).

Output of code:

```
The training ends at 6.000000 iteration and when function value is
less than 0.0001
The final x is 0.999950 and y is 0.999900 and function value is
0.000000
```

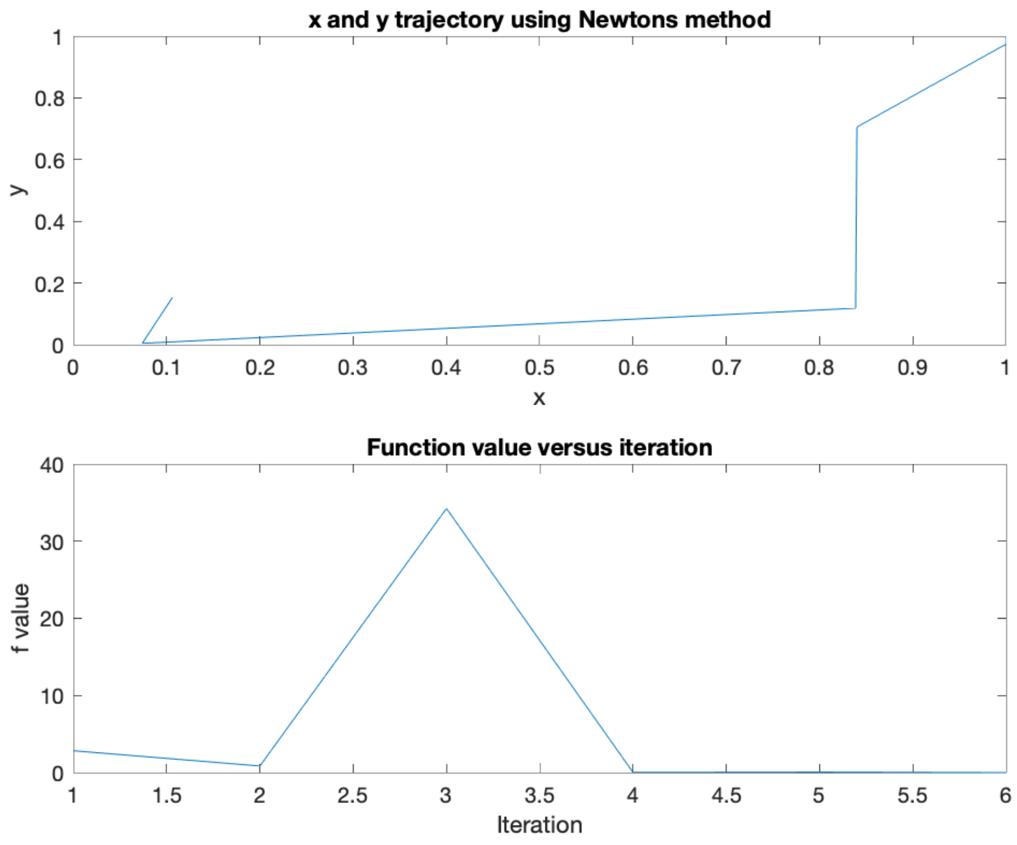
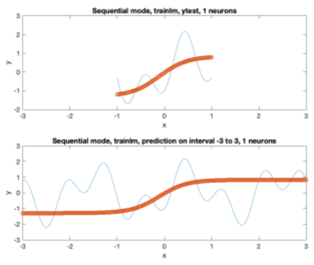


Figure 1c

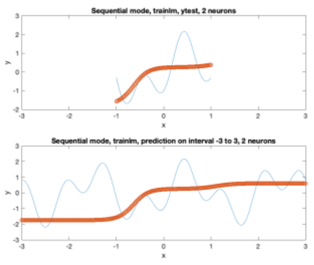
Q2. Function Approximation (20 Marks)

y=1.2sin(πx)–cos(2.4πx), for $x \in [-1,1]$.

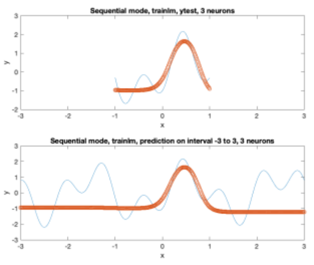
a). Using the sequential mode with BP algorithm and experiment with MLP: 1-n-1 (where n = 1, 2,,…, 10, 20, 50). Please refer to Figure 2a. For each Sequential Mode 1-n-1 graph, the upper subplot is the test values versus the desired function value in range -1 to 1, the lower subplot is the predicted values versus the desired function in range -3 to 3. We can see that for n = 1 to 5, the results are under-fitting in range -1 to 1. For n = 6 to 20 in range -1 to 1, the results are proper-fitting. For n = 50, the results are over-fitting in range -1 to 1.
However, we can see that in range -3 to 3, the MLP is not able to predict the values outside the training range -1 to 1.



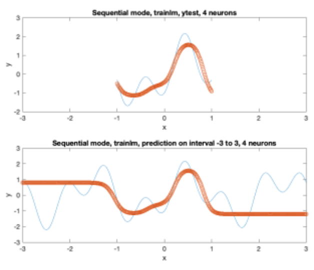
Sequential Mode 1-1-1



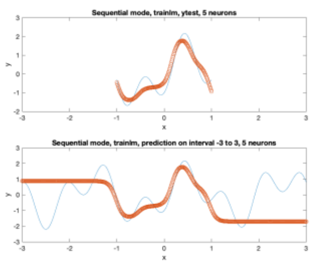
Sequential Mode 1-2-1



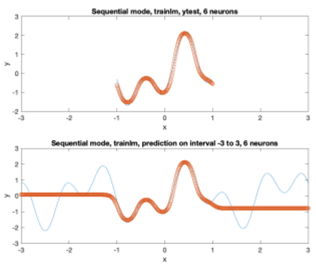
Sequential Mode 1-3-1



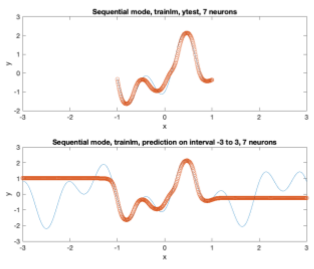
Sequential Mode 1-4-1



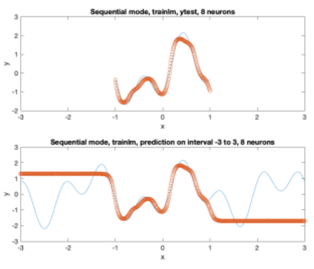
Sequential Mode 1-5-1



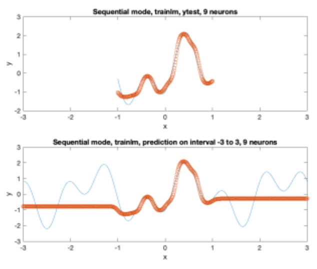
Sequential Mode 1-6-1



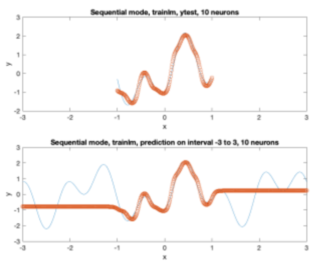
Sequential Mode 1-7-1



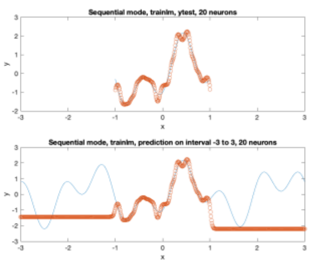
Sequential Mode 1-8-1



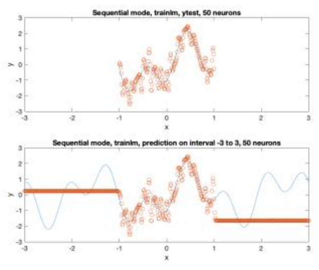
Sequential Mode 1-9-1



Sequential Mode 1-10-1



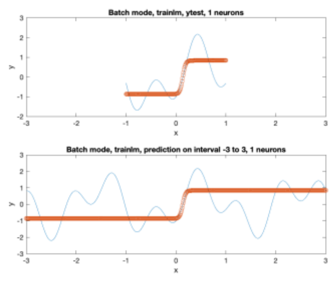
Sequential Mode 1-20-1



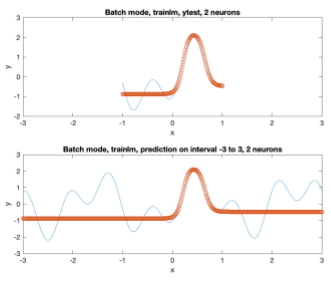
Sequential Mode 1-50-1

Figure 2a

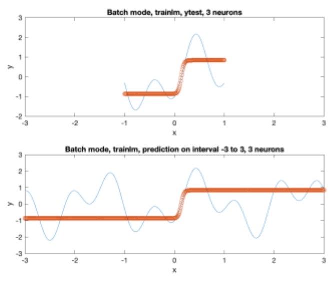
b). Using the batch mode with trainlm algorithm and experiment with MLP: 1-n-1 (where n = 1, 2,,…, 10, 20, 50). Please refer to Figure 2b. For each Sequential Mode 1-n-1 graph, the upper subplot is the test values versus the desired function value in range -1 to 1, the lower subplot is the predicted values versus the desired function in range -3 to 3. We can see that for n = 1 to 3, the results are under-fitting in range -1 to 1. For n = 4 to 20 in range -1 to 1, the results are proper-fitting. For n = 50, the results are over-fitting in range -1 to 1.
However, we can see that in range -3 to 3, the MLP is not able to predict the values outside the training range -1 to 1.



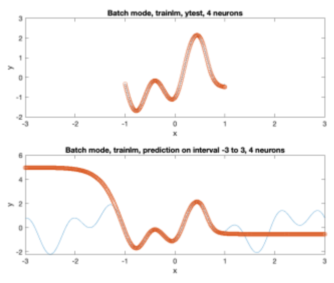
Batch Mode 1-1-1



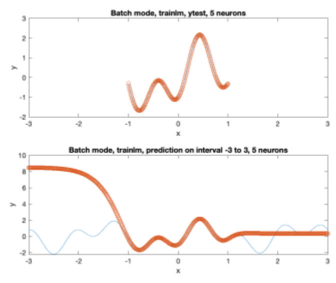
Batch Mode 1-2-1



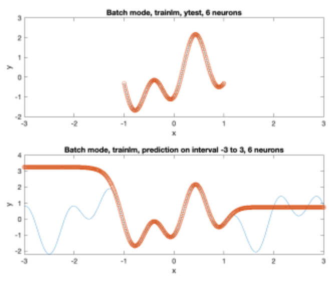
Batch Mode 1-3-1



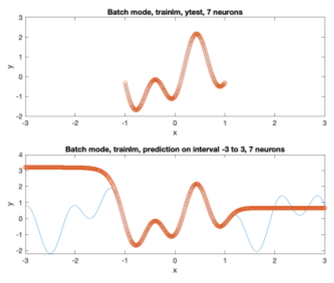
Batch Mode 1-4-1



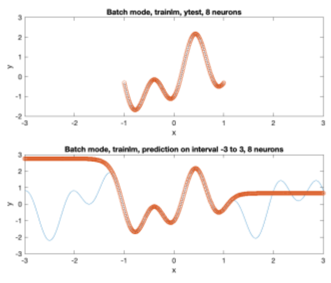
Batch Mode 1-5-1



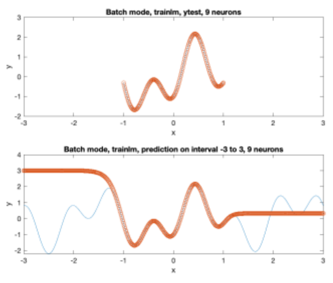
Batch Mode 1-6-1



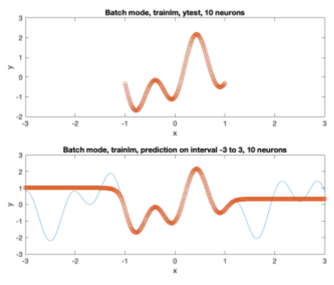
Batch Mode 1-7-1



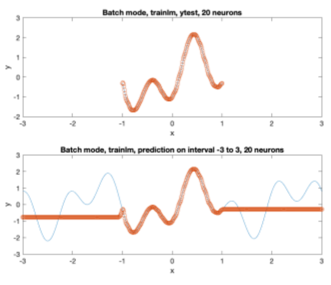
Batch Mode 1-8-1



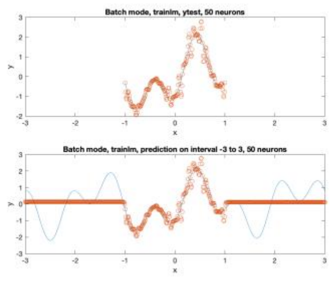
Batch Mode 1-9-1



Batch Mode 1-10-1



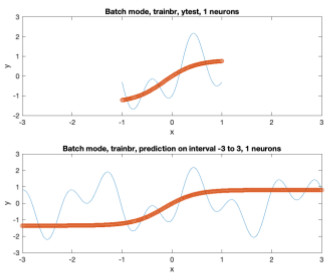
Batch Mode 1-20-1



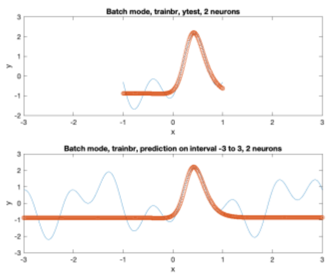
Batch Mode 1-50-1

Figure 2b

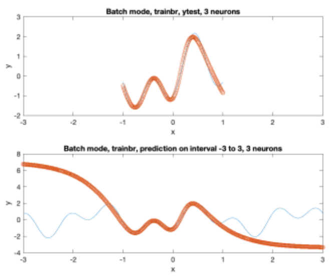
c). Using the batch mode with trainbr algorithm and experiment with MLP: 1-n-1 (where n = 1, 2,,…, 10, 20, 50). Please refer to Figure 2c. For each Sequential Mode 1-n-1 graph, the upper subplot is the test values versus the desired function value in range -1 to 1, the lower subplot is the predicted values versus the desired function in range -3 to 3. We can see that for n = 1 to 2, the results are under-fitting in range -1 to 1. For n = 3 to 50 in range -1 to 1, the results are proper-fitting. There is no over-fitting results for n = 1 to 50.
However, we can see that in range -3 to 3, the MLP is not able to predict the values outside the training range -1 to 1.



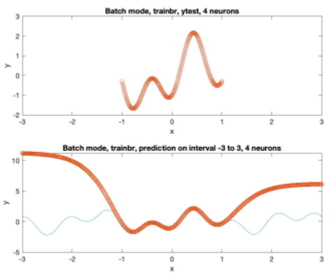
Batch Mode 1-1-1



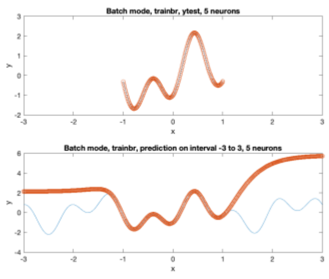
Batch Mode 1-2-1



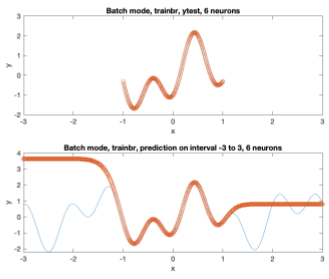
Batch Mode 1-3-1



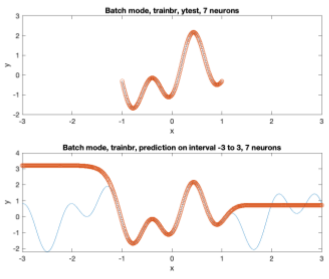
Batch Mode 1-4-1



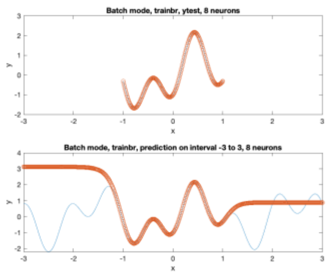
Batch Mode 1-5-1



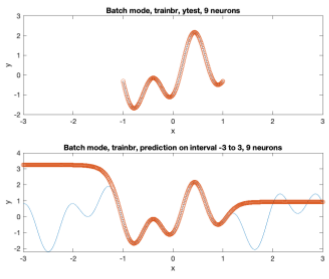
Batch Mode 1-6-1



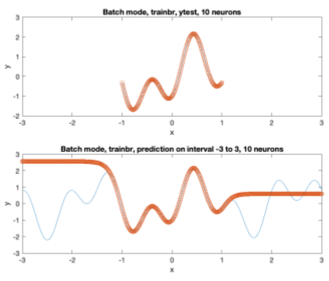
Batch Mode 1-7-1



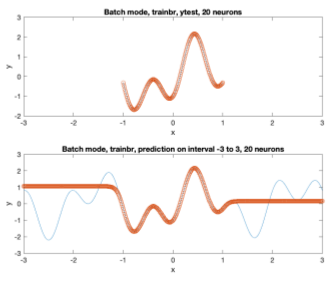
Batch Mode 1-8-1



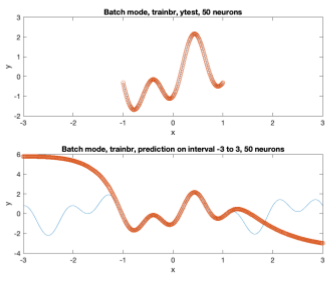
Batch Mode 1-9-1



Batch Mode 1-10-1



Batch Mode 1-20-1



Batch Mode 1-50-1

Figure 2c

Q3. Scene Classification (40 Marks)

a). Rosenblatt's perceptron (single layer perceptron)

please refer to the output of code, we can see that for single layer perceptron, the training accuracy can achieve 100%, but the validation accuracy can only achieve 68.8623% which is not good enough for the image classification.

Output of code:

```
Accuracy of single perceptron for training data is 1.000000.
Accuracy of single perceptron for validation data is 0.688623.
```

b). Apply PCA and resize the image to 128x128, 64x64 and 32x32

Accuracy table:

	128x128		64x64		32x32	
	Training	Validation	Training	Validation	Training	Validation
With PCA	100%	55.6886%	100%	56.8862%	100%	48.5030%
Without PCA	100%	68.2635%	100%	67.0659%	100%	65.8683%

We can see from the accuracy table that training accuracy is always 100% for any size reduction and no matter with or without PCA. By comparing accuracy of with PCA and without PCA, we can conclude that the accuracy will be reduced once apply PCA. Also, we can see that size reduction will reduce the validation accuracy as well.

Output of code:

```
Accuracy of single perceptron for training data with pca of resize
128x128 is 1.000000.
Accuracy of single perceptron for validation data with pca of resize
128x128 is 0.556886.
Accuracy of single perceptron for training data with pca of resize
64x64 is 1.000000.
Accuracy of single perceptron for validation data with pca of resize
64x64 is 0.568862.
Accuracy of single perceptron for training data with pca of resize
32x32 is 1.000000.
Accuracy of single perceptron for validation data with pca of resize
32x32 is 0.485030.
Accuracy of single perceptron for training data without pca of
resize 128x128 is 1.000000.
Accuracy of single perceptron for validation data without pca of
resize 128x128 is 0.682635.
Accuracy of single perceptron for training data without pca of
resize 64x64 is 1.000000.
Accuracy of single perceptron for validation data without pca of
resize 64x64 is 0.670659.
Accuracy of single perceptron for training data without pca of
resize 32x32 is 1.000000.
Accuracy of single perceptron for validation data without pca of
resize 32x32 is 0.658683.
```

c). MLP

Set all the parameters as default. Get the accuracy of training and validation using different number of hidden neurons.
We can see that the best accuracy for training and validation happens when number of hidden neurons is 20 which is 86.4271% and 74.8503% respectively.

Output of code:
```
Accuracy of 1 hidden neutrons for training data is 0.746507.
Accuracy of 1 hidden neutrons for validation data is 0.742515.
Accuracy of 2 hidden neutrons for training data is 0.740519.
Accuracy of 2 hidden neutrons for validation data is 0.664671.
Accuracy of 3 hidden neutrons for training data is 0.680639.
Accuracy of 3 hidden neutrons for validation data is 0.634731.
Accuracy of 4 hidden neutrons for training data is 0.776447.
Accuracy of 4 hidden neutrons for validation data is 0.688623.
Accuracy of 5 hidden neutrons for training data is 0.842315.
Accuracy of 5 hidden neutrons for validation data is 0.712575.
Accuracy of 6 hidden neutrons for training data is 0.754491.
Accuracy of 6 hidden neutrons for validation data is 0.664671.
Accuracy of 7 hidden neutrons for training data is 0.798403.
Accuracy of 7 hidden neutrons for validation data is 0.664671.
Accuracy of 8 hidden neutrons for training data is 0.734531.
Accuracy of 8 hidden neutrons for validation data is 0.688623.
Accuracy of 9 hidden neutrons for training data is 0.730539.
Accuracy of 9 hidden neutrons for validation data is 0.694611.
Accuracy of 10 hidden neutrons for training data is 0.818363.
Accuracy of 10 hidden neutrons for validation data is 0.724551.
Accuracy of 20 hidden neutrons for training data is 0.864271.
Accuracy of 20 hidden neutrons for validation data is 0.748503.
Accuracy of 50 hidden neutrons for training data is 0.770459.
Accuracy of 50 hidden neutrons for validation data is 0.724551.
```

d). MLP with weights regularization

I will use 20 hidden neurons in this experiment since it has been proved that the accuracy is the best when hidden neutrons are in section c). I will test on the 20 hidden neurons with and without regularization factor and compare the accuracy.
Firstly, test on 20 hidden neurons without regularization with all default parameters. Please refer to Figure 3d for the training process. The blue Train, green Validation and red Test data are all from the training data folder. The net will divide the training data into 3 category. By default, trainRatio = 0.7, valRatio = 0.15 and testRatio = 0.15. we can see that the 1-20-1 MLP reached the best accuracy in epoch 11 and started to over-fit in epoch 12.

Output of code (20 hidden neurons without regularization):
```
Accuracy of 20 hidden neutrons for training data without
regularization is 0.862275.
Accuracy of 20 hidden neutrons for validation data without
regularization is 0.724551.
```

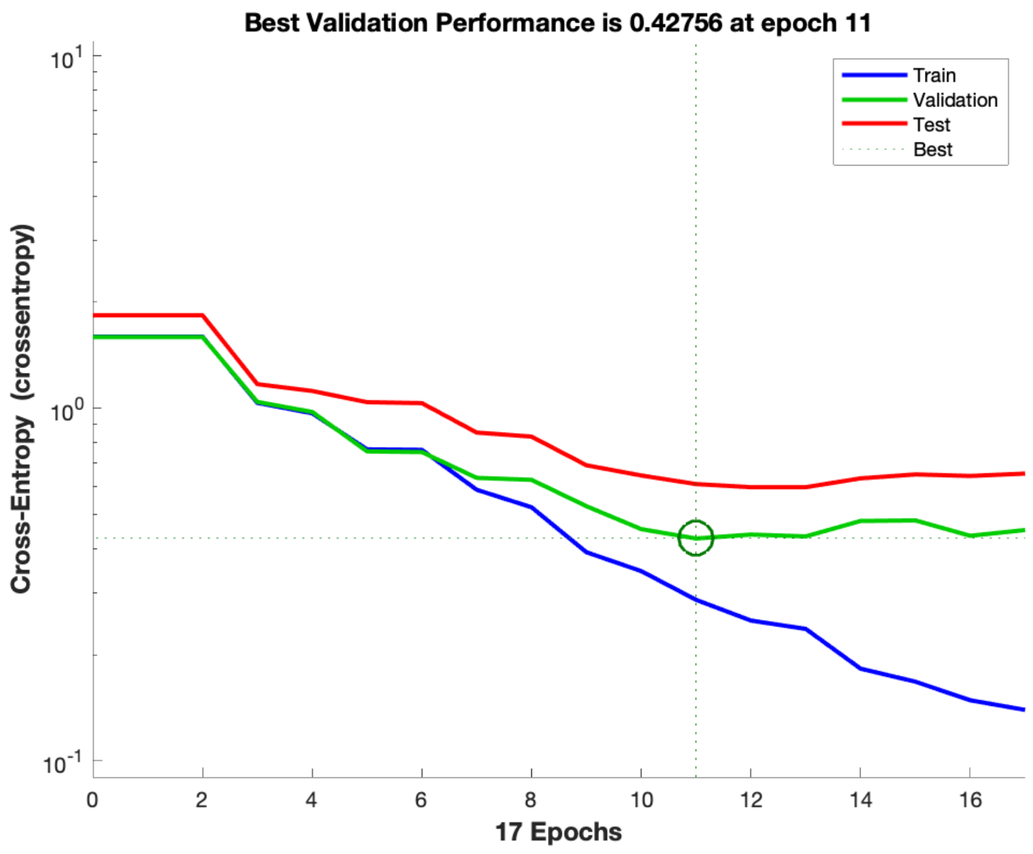


Figure 3d

Set all parameters as default except for the regularization factor. Choose the regularization factor from 0 to 1 with stepsize 0.1 and test on 20 hidden neurons network. Below is the result. We can see that regularization will not have significant improvement on the accuracy of the training and validation data.

Output of code (20 hidden neurons with regularization):

```
Accuracy of 20 hidden neutrons with regularization 0.100000 for
training data is 0.874251.
Accuracy of 20 hidden neutrons with regularization 0.100000 for
validation data is 0.742515.
Accuracy of 20 hidden neutrons with regularization 0.200000 for
training data is 0.904192.
Accuracy of 20 hidden neutrons with regularization 0.200000 for
validation data is 0.718563.
Accuracy of 20 hidden neutrons with regularization 0.300000 for
training data is 0.860279.
Accuracy of 20 hidden neutrons with regularization 0.300000 for
validation data is 0.718563.
Accuracy of 20 hidden neutrons with regularization 0.400000 for
training data is 0.866267.
Accuracy of 20 hidden neutrons with regularization 0.400000 for
validation data is 0.724551.
Accuracy of 20 hidden neutrons with regularization 0.500000 for
training data is 0.742515.
Accuracy of 20 hidden neutrons with regularization 0.500000 for
validation data is 0.700599.
```

```
Accuracy of 20 hidden neutrons with regularization 0.600000 for
training data is 0.934132.
Accuracy of 20 hidden neutrons with regularization 0.600000 for
validation data is 0.736527.
Accuracy of 20 hidden neutrons with regularization 0.700000 for
training data is 0.924152.
Accuracy of 20 hidden neutrons with regularization 0.700000 for
validation data is 0.724551.
Accuracy of 20 hidden neutrons with regularization 0.800000 for
training data is 0.742515.
Accuracy of 20 hidden neutrons with regularization 0.800000 for
validation data is 0.712575.
Accuracy of 20 hidden neutrons with regularization 0.900000 for
training data is 0.592814.
Accuracy of 20 hidden neutrons with regularization 0.900000 for
validation data is 0.628743.
Accuracy of 20 hidden neutrons with regularization 1.000000 for
training data is 0.477046.
Accuracy of 20 hidden neutrons with regularization 1.000000 for
validation data is 0.508982.
```

e). MLP with sequential mode training

MLP with sequential mode training is extremely slow. It took about one hour to train all the MLP using 1:10, 20 neurons. And there is no significant improvement on the accuracy comparing to batch mode training. So I will recommend batch mode training rather than sequential training.

Output of code:

```
Accuracy of 1 hidden neutrons for training data is 0.538922.
Accuracy of 1 hidden neutrons for validation data is 0.538922.
Accuracy of 2 hidden neutrons for training data is 0.614770.
Accuracy of 2 hidden neutrons for validation data is 0.580838.
Accuracy of 3 hidden neutrons for training data is 0.714571.
Accuracy of 3 hidden neutrons for validation data is 0.670659.
Accuracy of 4 hidden neutrons for training data is 0.586826.
Accuracy of 4 hidden neutrons for validation data is 0.491018.
Accuracy of 5 hidden neutrons for training data is 0.786427.
Accuracy of 5 hidden neutrons for validation data is 0.754491.
Accuracy of 6 hidden neutrons for training data is 0.758483.
Accuracy of 6 hidden neutrons for validation data is 0.646707.
Accuracy of 7 hidden neutrons for training data is 0.716567.
Accuracy of 7 hidden neutrons for validation data is 0.670659.
Accuracy of 8 hidden neutrons for training data is 0.788423.
Accuracy of 8 hidden neutrons for validation data is 0.682635.
Accuracy of 9 hidden neutrons for training data is 0.724551.
Accuracy of 9 hidden neutrons for validation data is 0.694611.
Accuracy of 10 hidden neutrons for training data is 0.776447.
Accuracy of 10 hidden neutrons for validation data is 0.694611.
Accuracy of 20 hidden neutrons for training data is 0.828343.
Accuracy of 20 hidden neutrons for validation data is 0.754491.
Accuracy of 50 hidden neutrons for training data is 0.884232.
Accuracy of 50 hidden neutrons for validation data is 0.724551.
```

f). Scheme of improvement

“Batch learning with second order algorithm (such as trainlm in MATLAB) is usually faster, but prone to local minima. Sequential learning can produce better solution, in particular for large database with lots of redundant samples. But it is slow.”
So we can improve the training by combining batch mode training and sequential mode training. Use sequential mode training to avoid the local minima trap and then use batch mode training to speed up the training process.