PSA Final Project Report:

Neural Network for Digit Recognition

Professor: Robin Hillyard

Project Team:

Gayathri Thammana (NU ID # 001388264)

Email id : thammana.g@husky.neu.edu

Mahitha Mandalapu (NU ID # 001387807)

Email id : mandalapu.v@husky.neu.edu

Table of Contents

Introduction to Neural Networks

An Artificial Neuron Network (ANN), popularly known as Neural Network is a computational model inspired by biological neural networks. It is like an artificial human nervous system for receiving, processing, and transmitting information in terms of Computer Science.

Basically, there are 3 different layers in a neural network:

1. Input Layer (All the inputs are fed in the model through this layer)
2. Hidden Layers (There can be more than one hidden layers which are used for processing the inputs received from the input layers)
3. Output Layer (The data after processing is made available at the output layer)

A Neural Network has got non-linear activation layers which is what gives the Neural Network a non-linear element. The function for relating the input and the output is decided by the neural network and the amount of training it gets. If you supply two variables having a linear relationship, then your network will learn this if you don't overfit. Similarly, a complex enough neural network can learn any function.

Proof of concept

XOR operation – In order to understand the neural network mechanism, we have built a Neural Network which can predict XOR operation results.

Neural Network Topology:

- Input layer with 2 neurons
- 1 hidden with 4 neurons
- Output with 1 neuron

We have extended this Neural Network to build multiple hidden layers and multiple outputs.

Problem Statement

Hand written Digit recognition - The handwritten digits are not always of the same size and width as they differ from writing of person to person. This makes it difficult for classifying digit images. In order to solve such a problem, we need a neural network

1. to learn digit patterns from humongous datasets,
2. adopt to unknown factors like different styles of handwritings, image sizes and colors
3. predict any unknown digit pattern

Solution:

1. **Design Neural Network**

   a. **Preparing Training Dataset**

   We used open source MNIST data set for Digit Recognition from following website:
   http://yann.lecun.com/exdb/mnist/
   This bundle consists of 60K images, each image consists of 784 bytes(resolution: 28*28 pixels).

   b. **Topology**

   As each input image consists of 784 bytes, we built input layer with 784 neurons. Each input neuron consuming a byte of the input image.

Input Layer	Hidden Layer 1	Hidden Layer 2	Output Layer	Accuracy(%)
784	95	35	10	~96%

   c. **Feed Forward**

   In feed-forward step, the input is given from the input layer and it is passed through the hidden layers until the output layer. The output of each layer is passed as the input to the next layer. The output in each layer and for each neuron is calculated using an activation function.

   The activation function that we have used for our network is the **Sigmoidal function**.

   A sigmoidal function is a mathematical function also known as S-shaped curve is represented as below:

$$y = \frac{1}{1 + e^{-x}}.$$

   d. **Back Propagation**

   We send the result of the output layer back to the first hidden layer recursively to adjust the weights for each neuron according to the feedback from each recursion. So any training step includes calculating the gradient(differentiation in calculus) and then doing backpropagation(integrating the gradient to get back the way the weights should change).

   e. **Checking accuracy with Testing Dataset**

   We executed feedforward and backpropagate steps for **100 epochs**.
   After 100 epochs, we have used testing dataset - “t10k-images-idx3-ubyte” for predicting results.

2. **User Interface to test NN:**
   To test the Neural Network, we developed Java Swing UI which performs following:
   a. **Train and test bundle** – we can initiate neural network to train and test the bundle
   b. **Paint palette** - to draw digit and test. Drawn image is of size 384*325 pixels. We have scaled this image to 28*28 pixels in order to make it work with trained neural network.

Analysis:

1. XOR Operation:

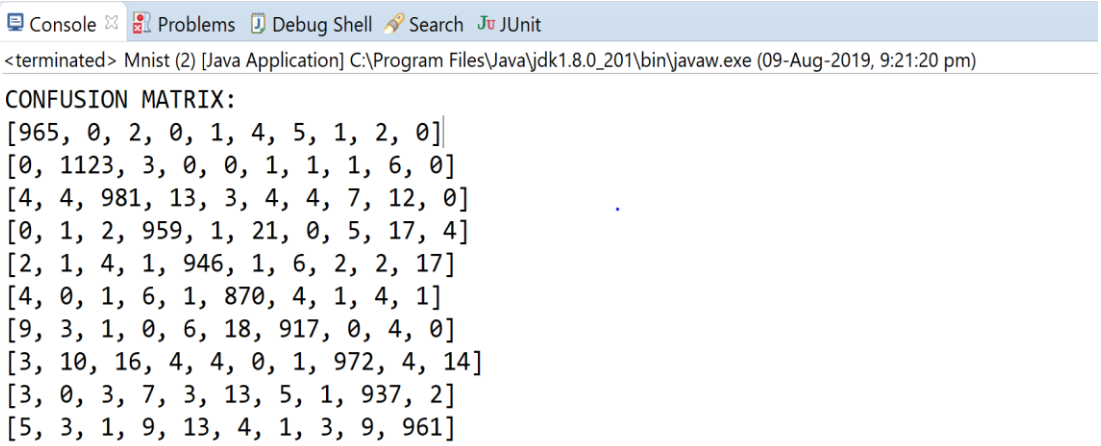
For topology {2,2,3,1} the XOR output from Neural Network is as follows:

```
Input 1: {0,0}
Output: [0.005714965437963332]
Input 2: {0,1}
Output: [0.9942119369137427]
Input 3: {1,0}
Output: [0.9943037163100399]
Input 4: {1,1}
Output: [0.0053899842969071295]
```

2. Digit Recognition:

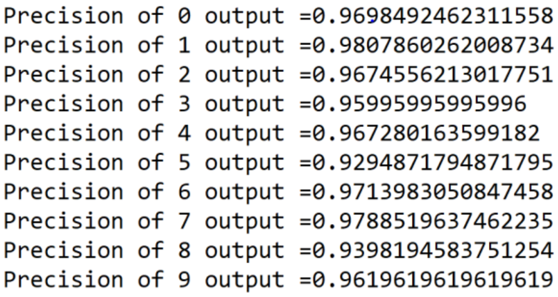
For topology {784, 95, 35, 10}, the output with confusion matrix, recall, precision and accuracy is as follows:

Confusion Matrix:



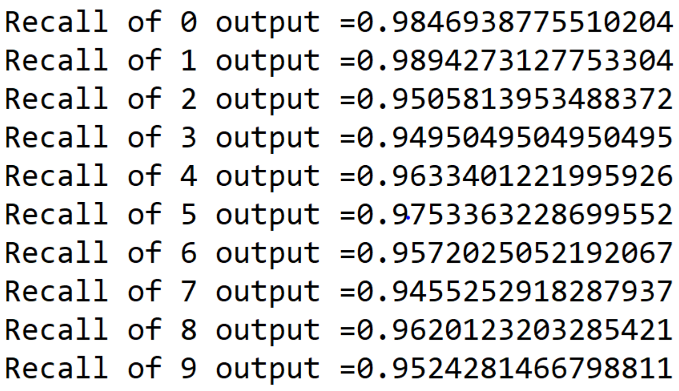
```
<terminated> Mnist (2) [Java Application] C:\Program Files\Java\jdk1.8.0_201\bin\javaw.exe (09-Aug-2019, 9:21:20 pm)
CONFUSION MATRIX:
[965, 0, 2, 0, 1, 4, 5, 1, 2, 0]
[0, 1123, 3, 0, 0, 1, 1, 1, 6, 0]
[4, 4, 981, 13, 3, 4, 4, 7, 12, 0]
[0, 1, 2, 959, 1, 21, 0, 5, 17, 4]
[2, 1, 4, 1, 946, 1, 6, 2, 2, 17]
[4, 0, 1, 6, 1, 870, 4, 1, 4, 1]
[9, 3, 1, 0, 6, 18, 917, 0, 4, 0]
[3, 10, 16, 4, 4, 0, 1, 972, 4, 14]
[3, 0, 3, 7, 3, 13, 5, 1, 937, 2]
[5, 3, 1, 9, 13, 4, 1, 3, 9, 961]
```

Precision:



```
Precision of 0 output =0.9698492462311558
Precision of 1 output =0.9807860262008734
Precision of 2 output =0.9674556213017751
Precision of 3 output =0.95995995995996
Precision of 4 output =0.967280163599182
Precision of 5 output =0.9294871794871795
Precision of 6 output =0.9713983050847458
Precision of 7 output =0.9788519637462235
Precision of 8 output =0.9398194583751254
Precision of 9 output =0.9619619619619619
```

Recall:

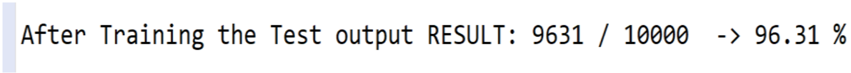


```
Recall of 0 output =0.9846938775510204
Recall of 1 output =0.9894273127753304
Recall of 2 output =0.9505813953488372
Recall of 3 output =0.9495049504950495
Recall of 4 output =0.9633401221995926
Recall of 5 output =0.9753363228699552
Recall of 6 output =0.9572025052192067
Recall of 7 output =0.9455252918287937
Recall of 8 output =0.9620123203285421
Recall of 9 output =0.9524281466798811
```

Accuracy:

- We tested Neural Network with different topologies as follows:
- **The best accuracy is 96.31% for topology {784, 95, 35, 10}**

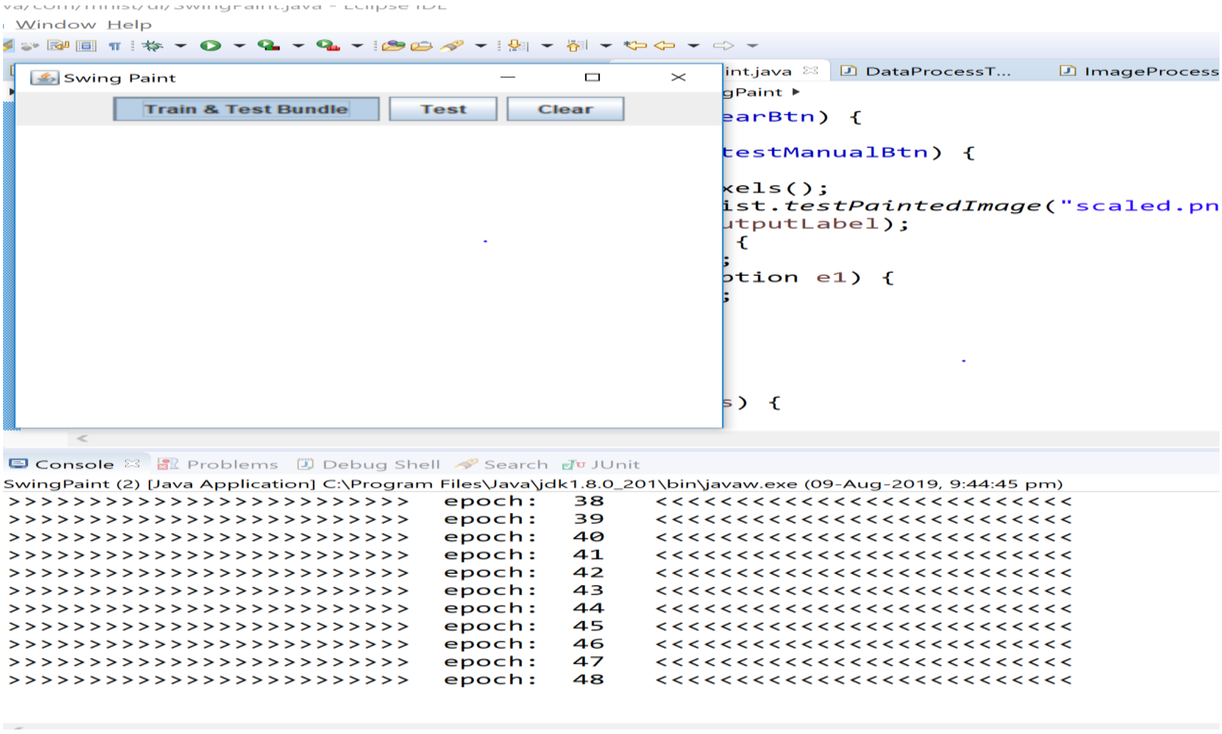
Input Layer	Hidden Layer 1	Hidden Layer 2	Output Layer	Accuracy(%)
784	95	35	10	~96%
784	65	25	10	~94%
784	200	100	10	~92%



```
After Training the Test output RESULT: 9631 / 10000  -> 96.31 %
```

3. Digit Recognition User Interface:

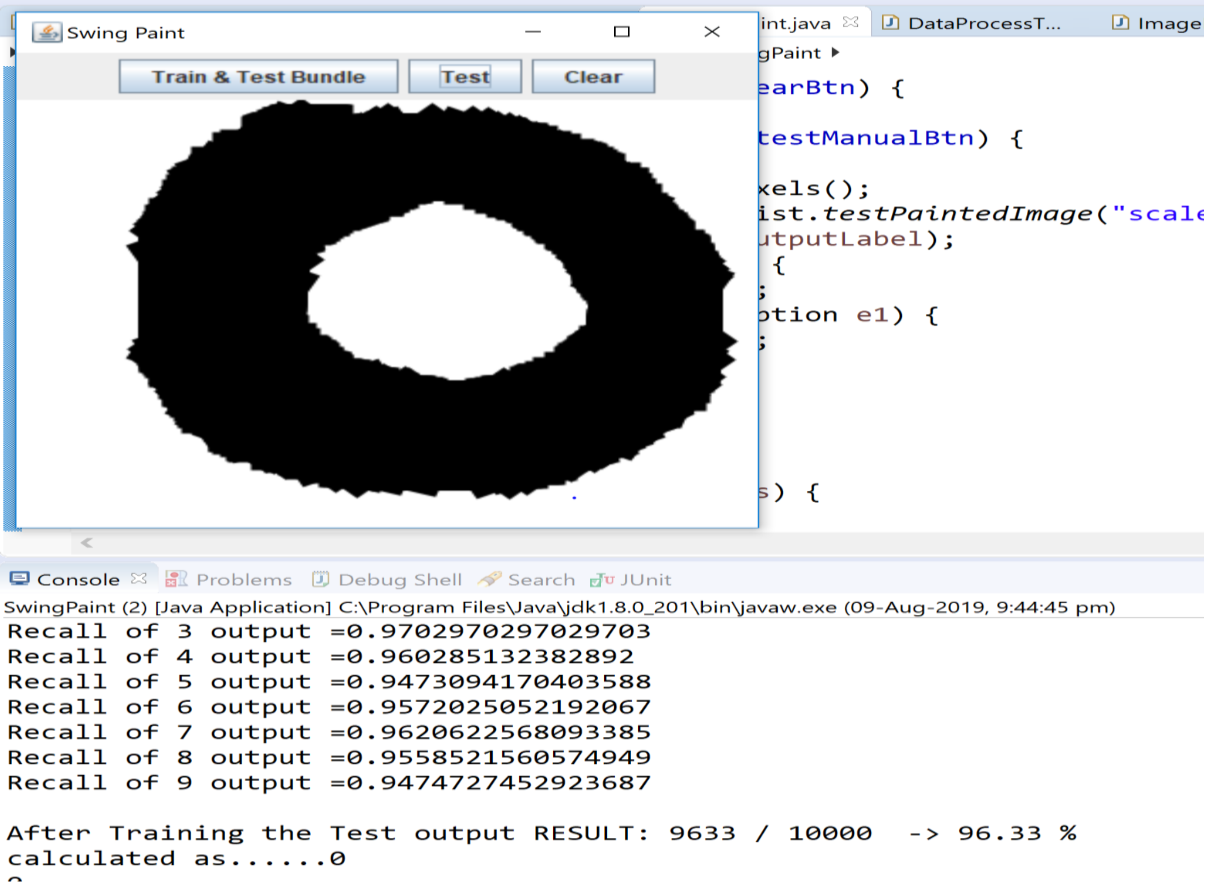
a. The User Interface and output for each action as follows:



```
gPaint ▸
earBtn) {

testManualBtn) {

xels();
ist.testPaintedImage("scaled.pn
utputLabel);
 {
;
ption e1) {
;

s) {
```

```
SwingPaint (2) [Java Application] C:\Program Files\Java\jdk1.8.0_201\bin\javaw.exe (09-Aug-2019, 9:44:45 pm)
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  38   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  39   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  40   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  41   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  42   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  43   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  44   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  45   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  46   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  47   <<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>  epoch:  48   <<<<<<<<<<<<<<<<<<<<<<<<<
```

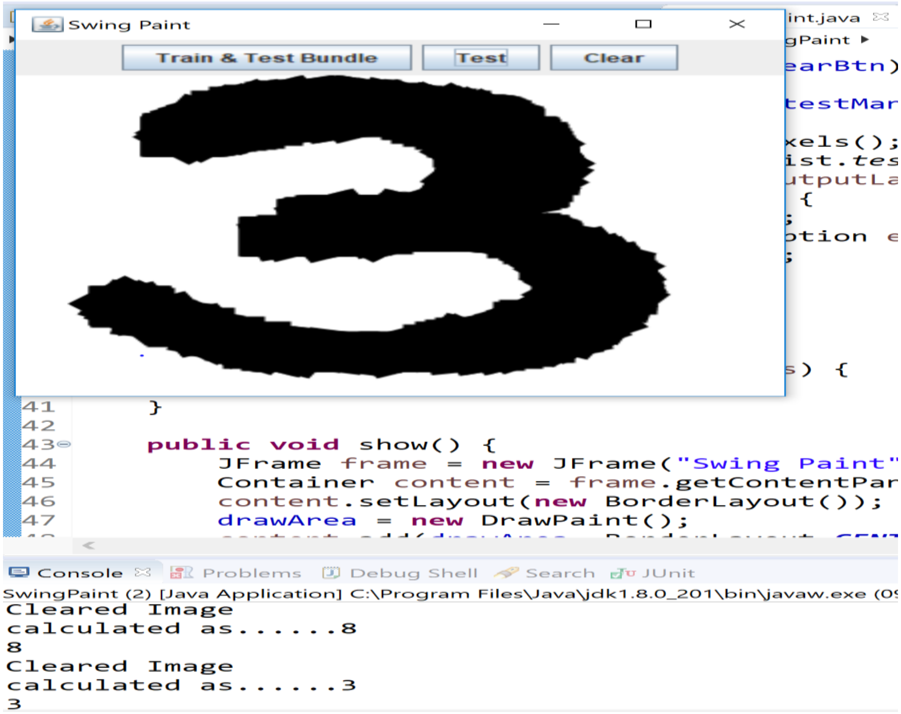
b. When zero is drawn it predicted calculated output as 0



```
gPaint ▸
earBtn) {

testManualBtn) {

xels();
ist.testPaintedImage("scale
utputLabel);
 {
;
ption e1) {
;

s) {
```

```
SwingPaint (2) [Java Application] C:\Program Files\Java\jdk1.8.0_201\bin\javaw.exe (09-Aug-2019, 9:44:45 pm)
Recall of 3 output =0.9702970297029703
Recall of 4 output =0.960285132382892
Recall of 5 output =0.9473094170403588
Recall of 6 output =0.9572025052192067
Recall of 7 output =0.9620622568093385
Recall of 8 output =0.9558521560574949
Recall of 9 output =0.9474727452923687

After Training the Test output RESULT: 9633 / 10000  -> 96.33 %
calculated as......0
```

c. When digit 3 is drawn it predicted calculated output as 3

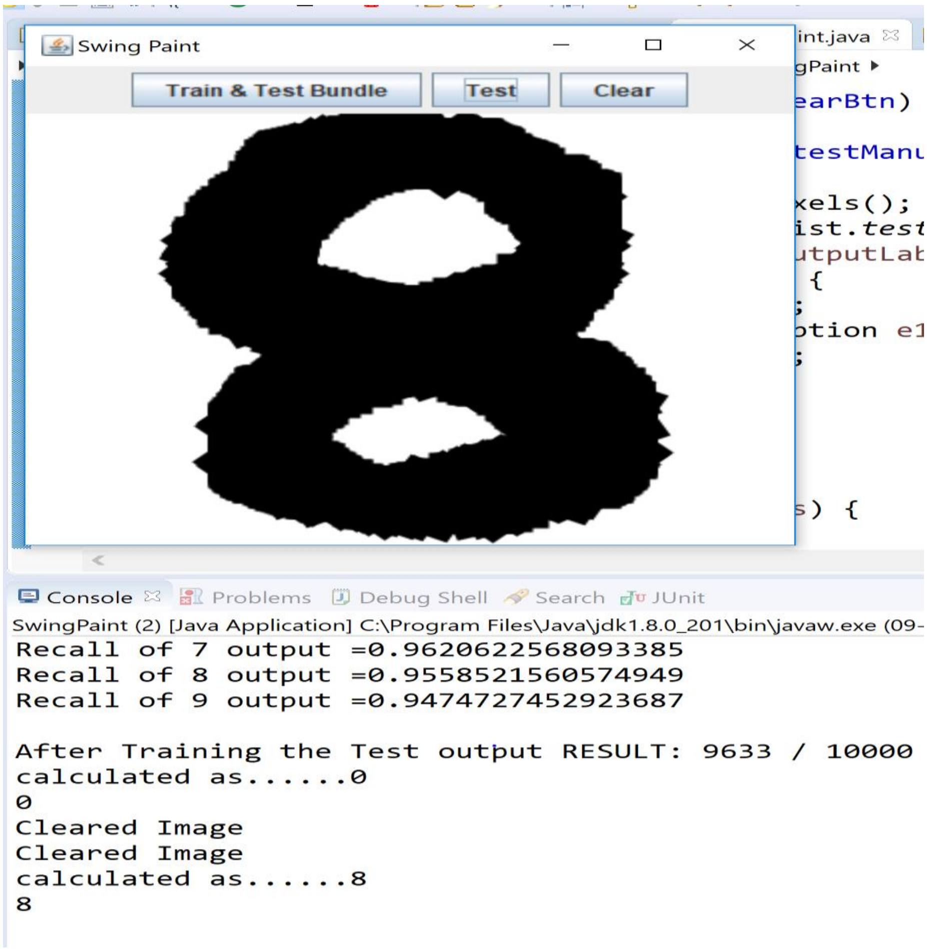


```
    }

    public void show() {
        JFrame frame = new JFrame("Swing Paint"
        Container content = frame.getContentPar
        content.setLayout(new BorderLayout());
        drawArea = new DrawPaint();
```

SwingPaint (2) [Java Application] C:\Program Files\Java\jdk1.8.0_201\bin\javaw.exe (0

```
Cleared Image
calculated as......8
8
Cleared Image
calculated as......3
3
```

d. When digit 8 is drawn it predicted calculated output as 8

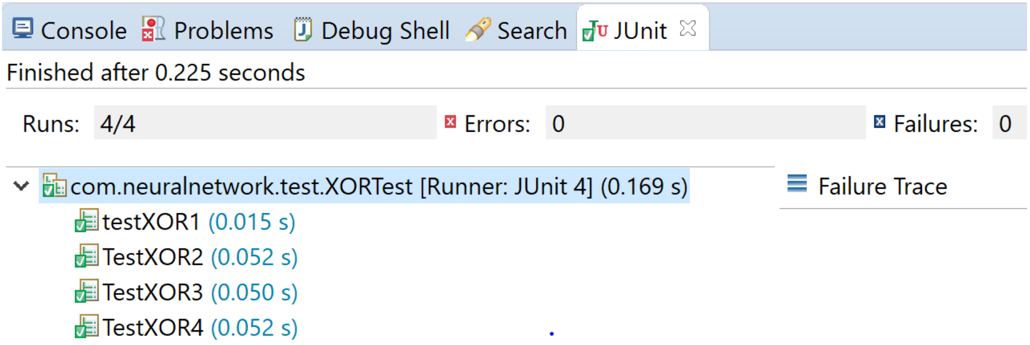


SwingPaint (2) [Java Application] C:\Program Files\Java\jdk1.8.0_201\bin\javaw.exe (09-

```
Recall of 7 output =0.9620622568093385
Recall of 8 output =0.9558521560574949
Recall of 9 output =0.9474727452923687

After Training the Test output RESULT: 9633 / 10000
calculated as......0
0
Cleared Image
Cleared Image
calculated as......8
8
```

JUNIT Test Cases

1. Test cases for multiple inputs of XOR :

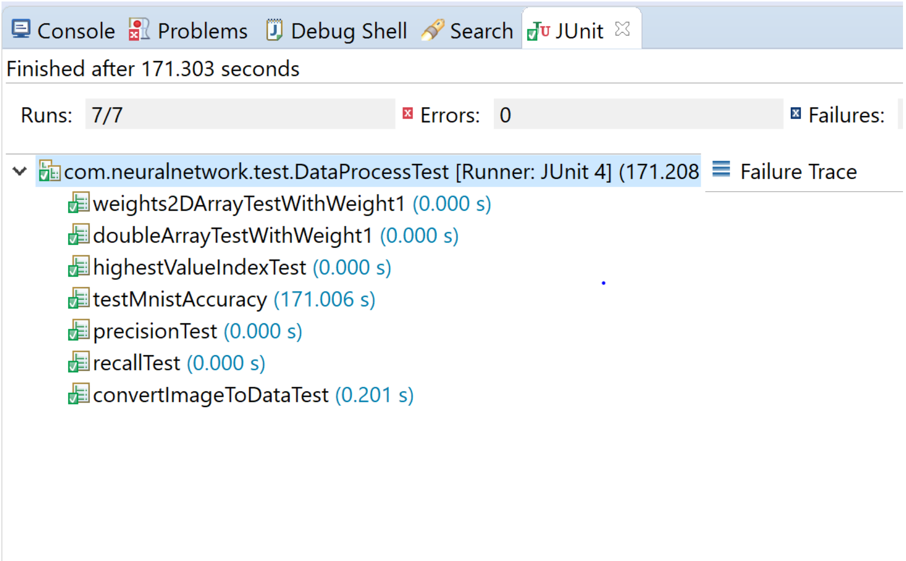


Finished after 0.225 seconds

Runs: 4/4 ☒ Errors: 0 ☒ Failures: 0

```
com.neuralnetwork.test.XORTest [Runner: JUnit 4] (0.169 s)
    testXOR1 (0.015 s)
    TestXOR2 (0.052 s)
    TestXOR3 (0.050 s)
    TestXOR4 (0.052 s)
```

2. Functional Test cases for Neural Network:

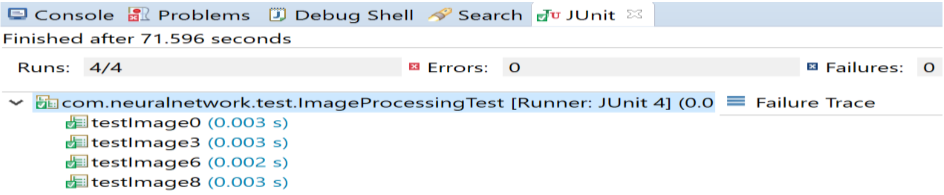


Finished after 171.303 seconds

Runs: 7/7 Errors: 0 Failures:

```
com.neuralnetwork.test.DataProcessTest [Runner: JUnit 4] (171.208
    weights2DArrayTestWithWeight1 (0.000 s)
    doubleArrayTestWithWeight1 (0.000 s)
    highestValueIndexTest (0.000 s)
    testMnistAccuracy (171.006 s)
    precisionTest (0.000 s)
    recallTest (0.000 s)
    convertImageToDataTest (0.201 s)
```

3. Test cases for Individual images:



```
Finished after 71.596 seconds
Runs:  4/4          Errors:  0          Failures:  0
com.neuralnetwork.test.ImageProcessingTest [Runner: JUnit 4] (0.0
    testImage0 (0.003 s)
    testImage3 (0.003 s)
    testImage6 (0.002 s)
    testImage8 (0.003 s)
```

Conclusion:

1. When we increase number of epochs, the accuracy of NN is observed to improve.
2. The number of neurons in hidden layers should not be very less or very high. The ideal topology can be identified after multiple trials.