SCHOOL OF COMPUTING

FACULTY OF ENGINEERING

SECV3213-02 ASAS PEMPROSESAN IMEJ
(FUNDAMENTAL OF IMAGE PROCESSING)

SECTION 02

PROJECT REPORT

HANDWRITTEN DIGIT RECOGNITION USING NEURAL NETWORK

Lecturer	Dr. Md Sah bin Hj Salam
Group Name	ZBL (Group 11)
Group Member	1. Ng Jing Er (A19EC0115) 2. Chiam Wooi Chin (A19EC0034)

TABLE OF CONTENTS

INTRODUCTION

The project proposed by our group for the course Fundamental Of Image Processing is the Handwritten Digit Recognition using Neural Network in Matlab. The chosen object patterns datasets are handwritten digits. The datasets of the handwritten digit patterns are downloaded from the website of kaggle.com. The digit pattern recognition can be defined as the classification of data based on knowledge already gained or on statistical information extracted from patterns. The project scope is to classify the 5 simple digit patterns class chosen, which are the digit 1, 2, 3, 4 and 5.

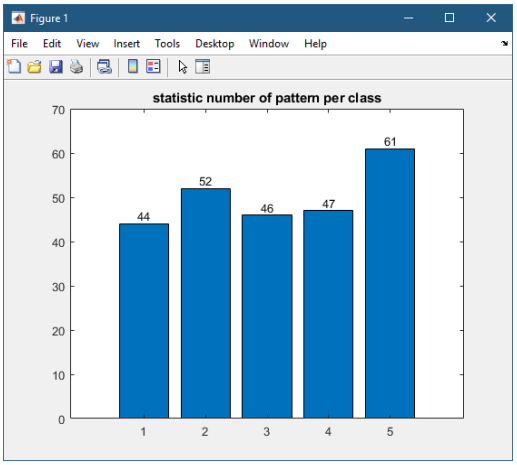
The project process is divided into 3 main phases. The image will undergo 3 phases which are image preprocessing, features extraction and the final step is recognition using Neural Network. Before starting our project, the problems of the image of the datasets selected are determined. After the problems of the images from the datasets chosen are determined, the algorithms that are required to solve image problems or enhance the image are listed. In this project, the images of datasets are implemented by taking the script code provided by the lecturer as our benchmark images to make comparisons with our own improved images. Then we have implemented the image preprocessing on the chosen object pattern that we learnt from this course for features extraction before starting the classification process and make an analysis by comparing the benchmark images with our improved images. The methods that were implemented in our project included sharpening, thresholding and binary, morphological thickening, morphological closing, median filtering and normalisation with resizing and reshape.

Since the Neural Network is applied in our project, the features of the images will be collected and rearranged in a matrix of training patterns to be fed into the Neural Network to undergo the recognition process. The final phase is recognition which includes training and testing with the Neural Network.

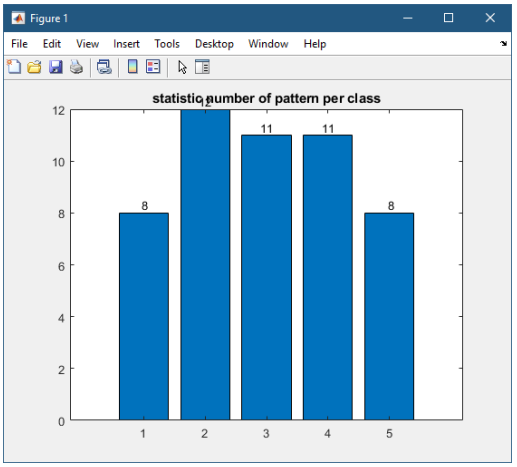
PROJECT DATASETS

In this project, we use 5 classes of simple digit patterns, which are the digit 1, 2, 3, 4 and 5. The project datasets include training patterns data, 2000 patterns which for each class has 400 patterns and we used 250 patterns for training. For test patterns, the project datasets have 240 patterns which for each class has 48 patterns and we take 50 patterns for testing.

(Benchmark) Class Distribution :

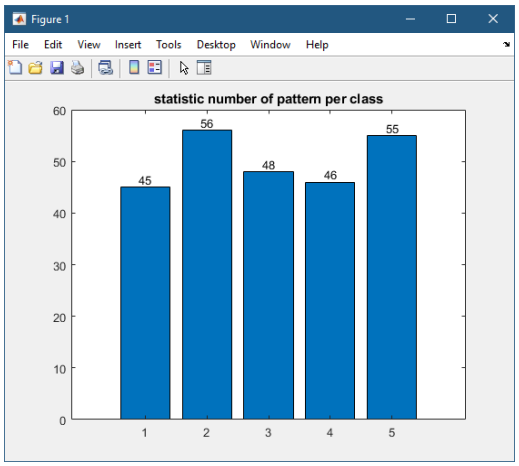


Bar Chart of Training Class Pattern

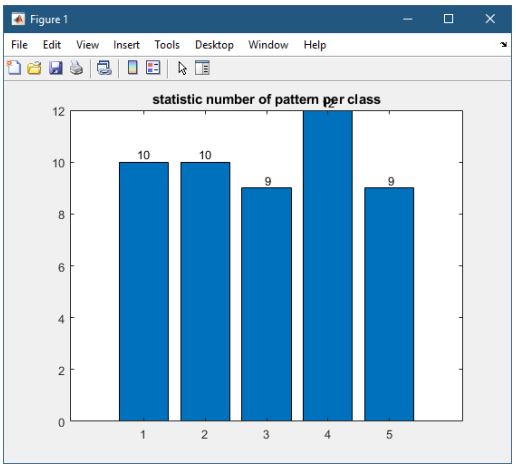


Bar Chart of Test Class Pattern

(Improved) Class Distribution :



Bar Chart of Training Class Pattern



Bar Chart of Test Class Pattern

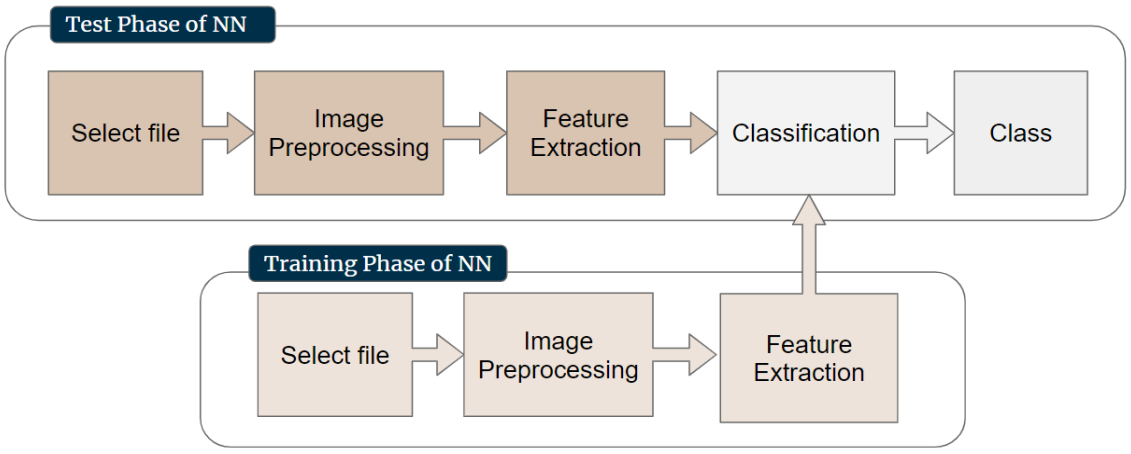
IMAGE PROBLEMS

The image problems of the image of the datasets chosen for enhancement are identified and the suitable image processing algorithms are selected to resolve the problems.

1. Datasets Images show that the digit thickness is not consistent where some of the digits in the images are too thin.
2. Datasets images have the small gaps or holes from some of the digit objects.
3. Datasets Images are not clear.

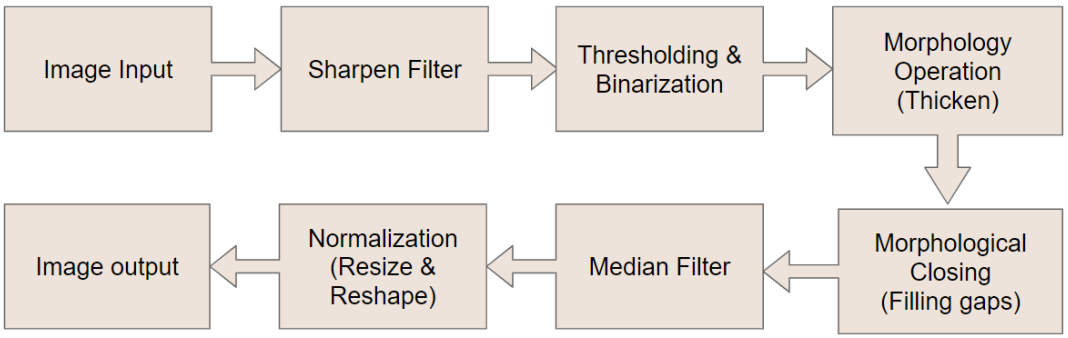
PIPELINE PROCESS

There are two types of image enhancement used to involve in our project. The Finger Recognition NN script provided by the lecturer is used as the benchmark for the project comparison where the datasets are replaced with our project handwritten digit datasets. The datasets are trained and tested for both the benchmark image enhancement and improved version. The pipeline of the image enhancement of the benchmark and our improved version of image enhancement are shown below.



Pipeline of Benchmark Image Enhancement:

Pipeline of Improved Image Enhancement:



The benchmark image enhancement is only using binarization, median filter and normalization. While the improved version has changed the sequence of the image enhancement steps. For the improved version, the dataset image will undergo a sharpened filter by using the laplacian filter before binarization. Besides, there are two morphological operations used in the improved version of image enhancement, the morphological thicken and the morphological closing. The next step is continued with the median filter that is used in the benchmark image enhancement. And finally the image normalization using resize and reshape function.

QUALITATIVE RESULTS & DESCRIPTION

A. Image Pre-processing

The pre-processing phase is an important step for our digit pattern recognition as it will help to remove the not necessary data information of the digital image and improve the recognition accuracy. In our pre-processing phase of this project, we have implemented some of the methods such as sharpening, thresholding and binary, morphological thickening, morphological closing, median filtering and normalisation with resizing and reshape.

1. Sharpen Filter

```
%do sharpening filter
Id = im2double(A);
f = fspecial('laplacian',0);
I_filt = imfilter(Id,f);
S = imsubtract(Id,I_filt);

subplot(241);imshow(A);title('original image');
subplot(242);imshow(S);title('sharpen image');
```

Image after sharpen filter	
Before	After
	
Analysis	
Initially, the image is implemented using a sharpening *filter* using a *laplacian filter*. A laplacian filter is an edge detector used to compute the second derivatives of an image. It focuses on sharpening pictures. We create a laplacian filter using the *fspecial()* function, apply the filter value using *imfilter()* function and assign it to a variable of *I_filt*. To make the image datasets sharpened, the *imsubtract()* function is used to to subtract I_filt from the original image that converted into double precision.	

2. Thresholding and Binarization

```
% do thresholding and convert to binary
level = graythresh(S);
BW = im2bw(S,level);
subplot(243);imshow(BW);title('binary image');
```

Image after thresholding and binarization	
Before	After
	
Analysis	
Thresholding is a type of image segmentation. After reading the image file, the thresholding algorithm is used to change the pixels of the digit image to make the image analysation easier. The threshold value is determined using *graythresh()* function Then the image is then converted into binary. The *binarization* is the stepwise process of converting the original gray-scale image into a binary image which is shown as 0 and 1 by setting the threshold value using *level* via *im2bw()* function.	

3. Thicken

```
%thicken
B= imcomplement(BW);
T = bwmorph(B,'thick');
BT= imcomplement(T);
subplot(244);imshow(BT);title('thicken image');
```

<table>
<tr><td colspan="2">Image after morphological thicken</td></tr>
<tr><td>Before</td><td>After</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td colspan="2">Analysis</td></tr>
<tr><td colspan="2">After the process of thresholding and binarization, we are able to apply a thickening effect on the digit's pixel of the image using the bwmorph() function. This function involves the morphology operation of 'thick'. The image is converted into its complement before thickening. After the bwmorph() function operates the 'thick' process by adding pixels to the exterior of objects , the image is complemented again. The output showed that the pixel of the digit is thickened.</td></tr>
</table>

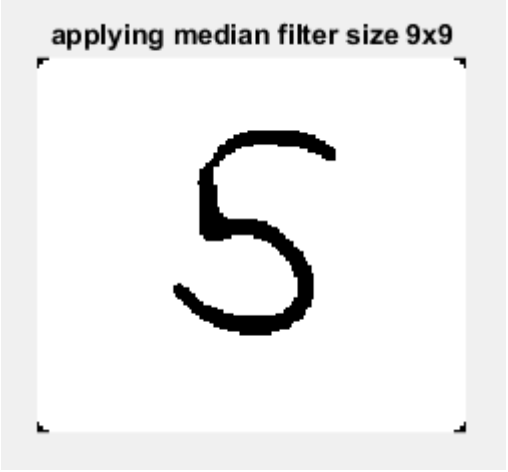
4. Morphological closing

```
%Morphological Closing to Fill Gaps in an Image
SE = strel('square',5);
C= imcomplement(BT);
BW2 = imclose(C,SE);
CT= imcomplement(BW2);
subplot(245),imshow(CT), title('Fill gap in image')
```

Image after morphological closing	
Before	After
	
Analysis	
After applying a *thickening* effect on the image, we perform *morphological closing* on the image, using the structuring element *SE*. The morphological close operation is a dilation followed by an erosion, using the same structuring element for both operations. We use a *square* structuring element to preserve the nature of the object and specify a radius of *5* pixels so that the gap gets filled. The image is converted into its *complement* before the closing process. After the *imclose()* function operates the 'close' process by filling gaps on the objects, the image is complemented again. The output showed that the image is filled with the gaps by morphological closing.	

5. Median filter

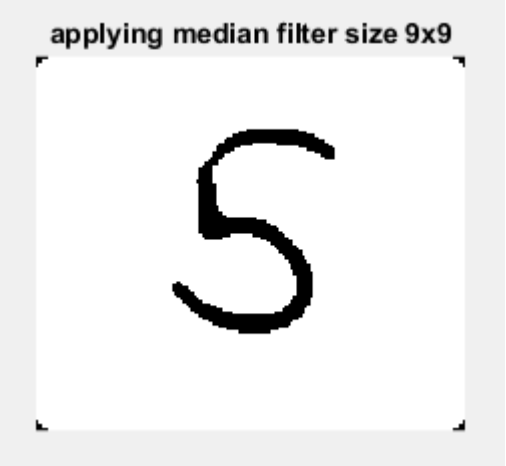
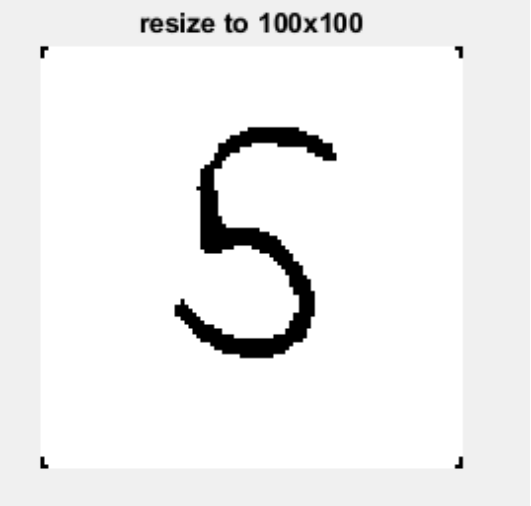
```
%apply median filter
J = medfilt2(CT, [9 9]);
subplot(246);imshow(J);title('applying median filter size 9x9');
```

<table>
<tr><td colspan="2">Image after median filter</td></tr>
<tr><td>Before</td><td>After</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td colspan="2">Analysis</td></tr>
<tr><td colspan="2">A median filter is implemented for the image enhancement process by using the medfilt2() function which contains the median value in the 9-by-9 neighbourhood around the corresponding pixel in the input image.
Median filter is able to reduce the sharp features and preserve the edges in the output image whilst filtering the noise after binarization and thickening. After applying the median filter on the dataset image, the digit shown in the image apparently smoothen with much detailed edges.</td></tr>
</table>

6. Normalisation (resize & reshape)

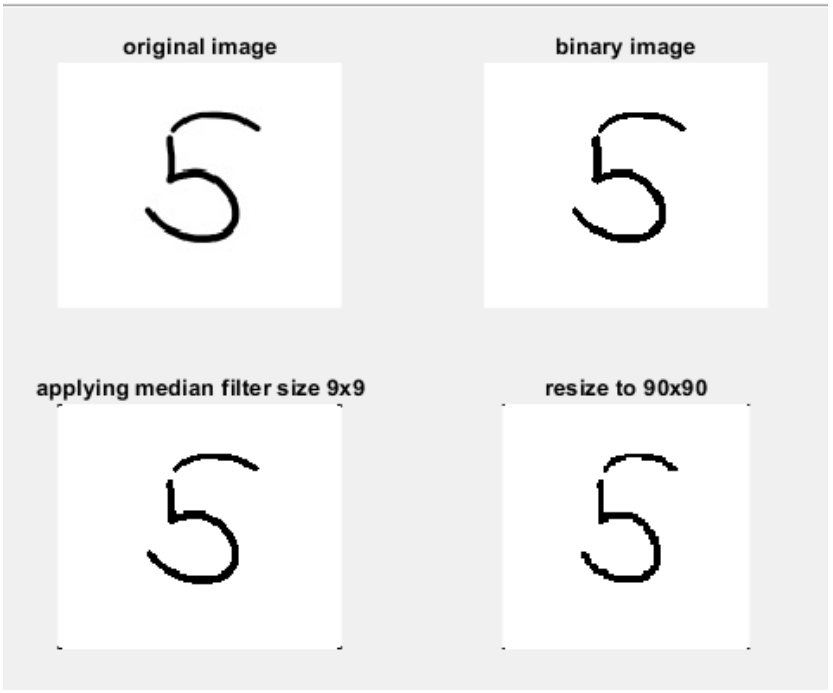
```
% resized
k=imresize(J,[100 100]);
subplot(247);imshow(k); title('resize to 100x100');

X = reshape(k,[],100*100);
```

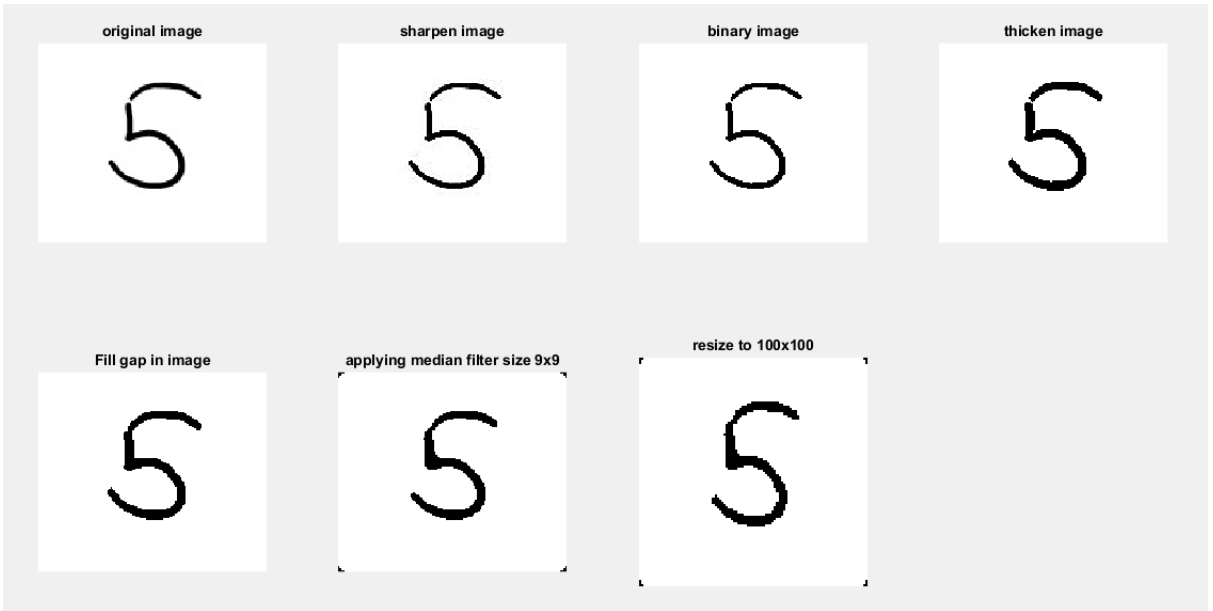
Image after resize & reshape	
Before	After
	
Analysis	
After the median filter with size 9*9, we want to resize the image into *100*100* size using the *imresize()* function so that the image is not too big or too small. We set the output image with desired size with passing imresize a vector that contains the number of rows (100) and columns (100) in the output image. Then, we pass the resize output into the *reshape* function with specify [] for the first dimension to let reshape automatically calculate the appropriate number of rows and *100*100* to reshape with the 10000 number of columns in the matrix of the final output image of preprocessing.	

B. Benchmark and Improved Image Enhancement Output

Benchmark Image Enhancement:



Improved Image Enhancement:



C. Feature Extraction

The Feature Extraction is used to choose important features and to remove inconsistencies and redundancies from the pre-processed image datasets.

Features:

- The features of the resize image of datasets
- The features based on the grid binary from the resized image

The features will be collected and rearranged in a matrix of training patterns to be fed into the Neural Network. The steps are as follow:

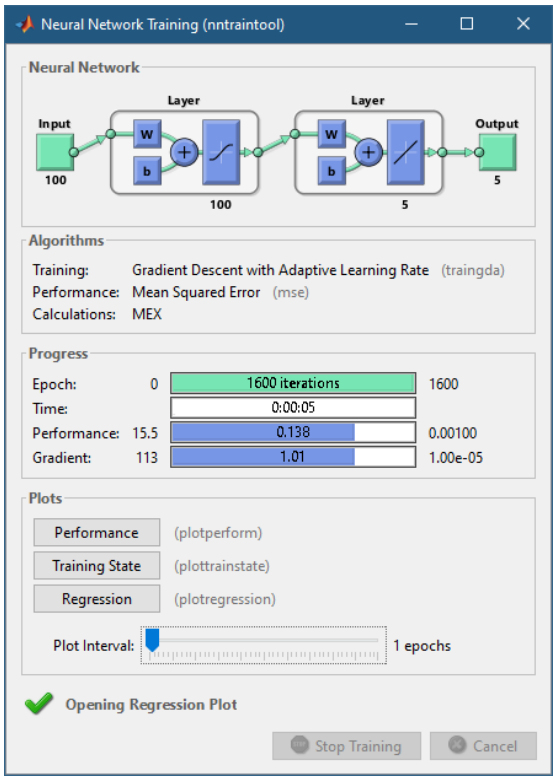
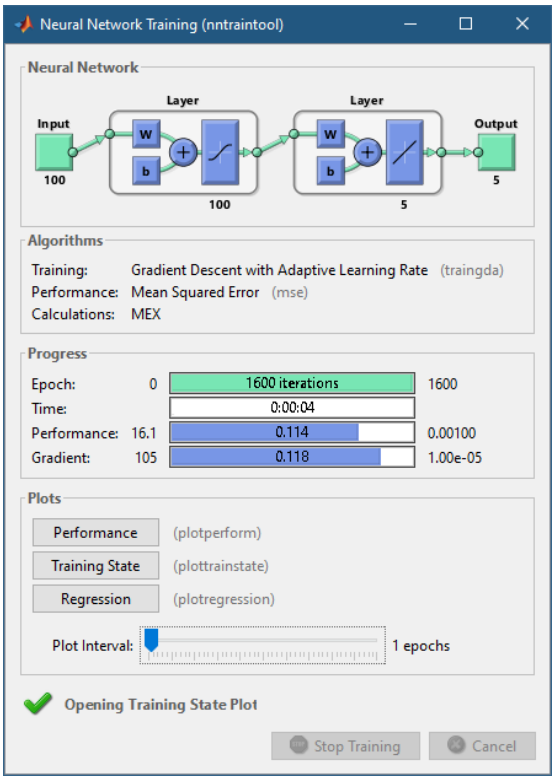
1. The binary features from the whole image from the pre-processing phase are extracted.
    - The features are 10000 after resized to [100x100] which is smaller than the original size of the image .
2. The grid features are extracted from the resized image.
    - Grid (10x10) is used to make the features of only 100 nodes.
3. The pattern is rearranged into a training / testing matrix.
4. The target class is extracted based on the filename for each digit pattern (1,2,3,4,5).

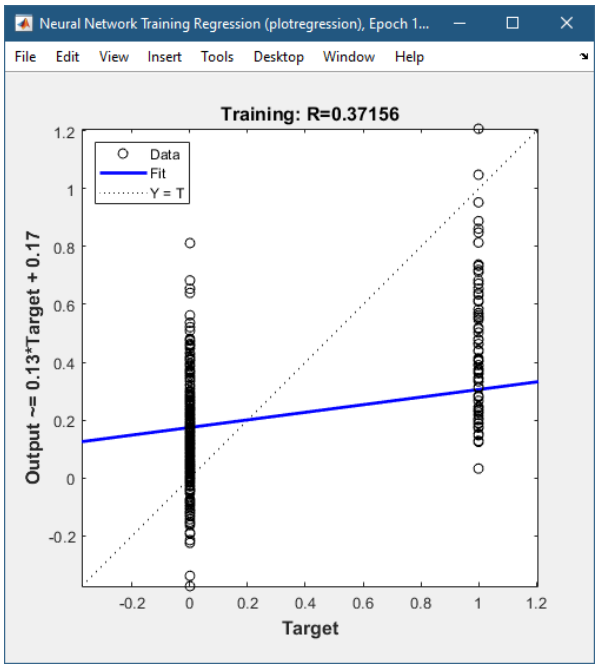
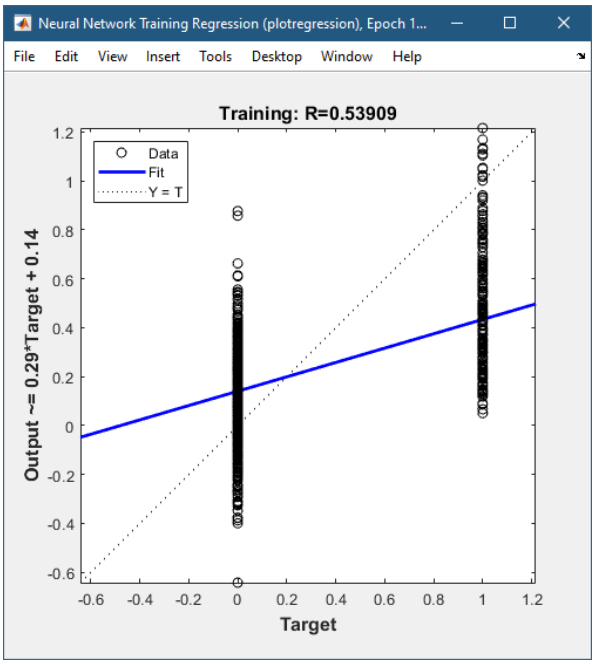
QUANTITATIVE RESULT AND DISCUSSION

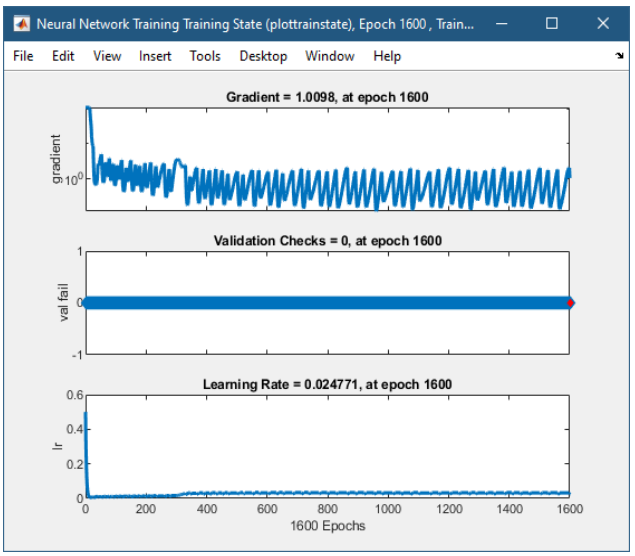
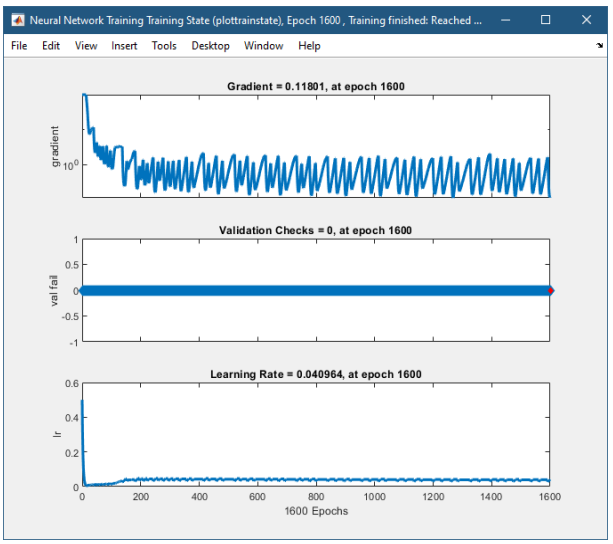
The benchmark result of the recognition rate for our project is 24%. After applying the image preprocessing algorithms proposed, we manage to improve the result of the recognition rate from **24%** to **52%**.

Training with Neural Network *(NNTrain.m)*

1. Input of NN: the number of features Hidden nodes need to be chosen (100)
2. Output of NN: the number of class (5)
3. Learning rate : 0.5
4. Momentum constant : 9

<table>
<tr><td colspan="2">Neural Network Training Tool</td></tr>
<tr><td>Benchmark</td><td>Improved</td></tr>
<tr><td> 
</td><td> 
</td></tr>
</table>

Neural Network Training Regression	
Benchmark	Improved
	

Neural Network Training State	
Benchmark	Improved
	

Recognition rate	
Benchmark	Improved
D = 0 0 0 0 8 0 0 0 0 12 0 1 0 1 9 0 0 0 5 6 0 1 0 0 7 ans = 5 5 recognition rate is : 24	D = 10 0 0 0 0 4 4 0 2 0 3 2 1 2 1 1 1 2 7 1 4 1 0 0 4 ans = 5 5 recognition rate is : 52

Based on the Neural Network Training Regression result, the benchmark regression result obtained is R=0.37156; while the regression result of the improved version is R=0.53909. The regression result R is improved by 0.16753. According to these results, the higher regression value indicates a more perfect fit to the work, and is thus a more reliable model for future forecasts.

From the Neural Network Training State Plot, the validation check is 0 for both benchmark and improved results. The learning rate of benchmark result is 0.024771, at epoch 1600 while the learning rate of improved result is 0.040964, at epoch 1600. This means that the learning rate has improved by 0.016963. The learning rate of the improved result showed that the neural network model of the improved result is learning the image problem faster than the benchmark result and this achieves better performance on analysis of our image problem.

In our proposed image enhancement, the final result of the recognition rate is able to increase compared with the benchmark result. The benchmark result of recognition rate is 24%, while the recognition rate of the improved version of proposed solution is 52%. The recognition rate is increased by the average of 28% of all classe

CONCLUSION

In this FIP Project, we are able to apply the knowledge of image processing learnt in order to complete the task of Pattern Classification. Compared with the benchmark results, the improved image enhancement is able to increase the rate of recognition from 24% to 52%. The proposed approaches involve various image processing techniques such as logic operation for brightening image, morphological image processing for thicken operation, neighbourhood operation of median filter; which help in Neural Network for recognizing the patterns. By using the proposed image enhancement, the project is able to display a clearer results of image output compared with the benchmark results.