Lab 5: Artificial Neural Networks

Spring 2021

BRAE 428

Dr. Bo Liu

Emma Gray, Chandler Jones, Gracely Speth

Abstract:

Artificial Neural Networks (ANNs) are designed to operate similarly to biological neural networks. They are inspired by and constitute similarities to animal brains. ANNs consist of image recognition, such as identifying if an image of an animal is a “cat” or “dog”. The network takes in the image, and through a trained program can output the image into a category. In this lab we will be doing something similar but categorizing numbers. The computer will input an image of a handwritten number and be able to identify it as 0-9.

Introduction:

This lab uses the Neural Network Toolbox in Mathworks MATLAB. This program uses the example numbers to infer rules in recognizing specific numbers in a process called network training. When the number of examples in increased the program can further improve its recognition skills as it picks up more detailed rules. This improves the accuracy of the program. Overtraining is a possibility and a potential problem. If a program is overtrained, it will narrow down the rules such that a number written in different handwriting will not be assigned correctly if at all.

Data is given to the students before the lab, and it consists of hand-drawn digits from 0-9. The given data is in grayscale and each image is 28 by 28 pixels. This comes out to 784 pixels total per image. The first column of the data is called “label” and is the output (ie. 0,1,2,3, …, 9). The rest of the columns contain the pixel values of the image.

Materials and Methods

Materials:

Computer

Printer paper

Black pen

MATLAB

MATLAB Applications

Methods:

1. Download the handwritten data set from canvas and load the data into MATLAB.
2. Plot some examples to see what the images look like (Figure 1).

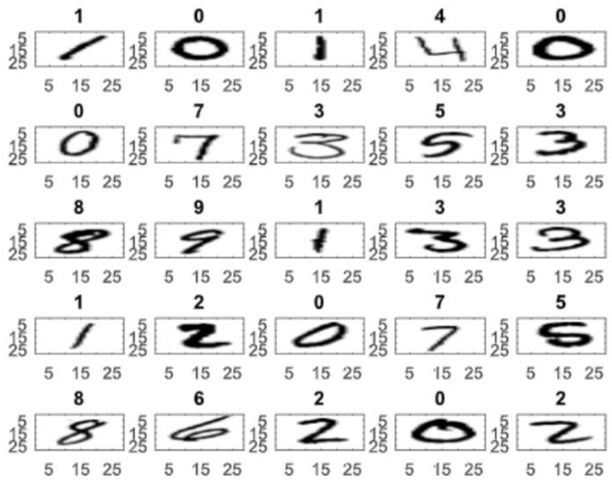


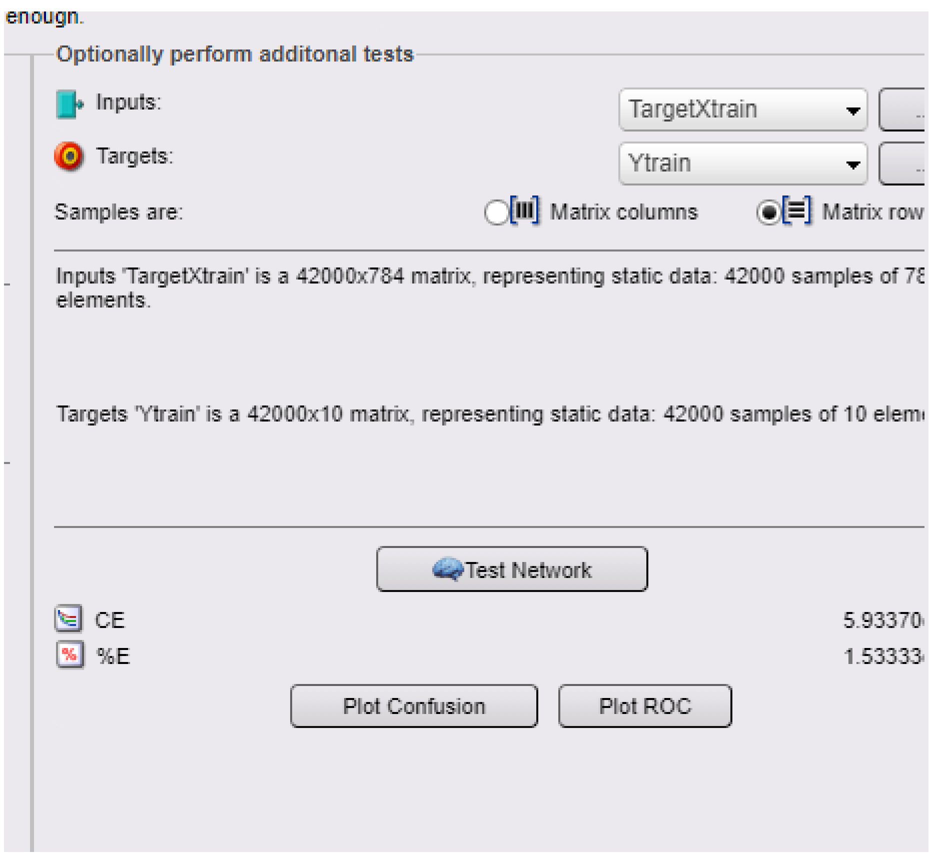
Figure 1. Sample images of the imported data.

3. The data is given with the first column storing the correct results (number labels) and the remaining columns storing the pixel values of the image. To prepare this data to be trained, we need to relabel it into (labels, Ytrain, and Xtrain). The labels contain the first column of number labels. This is then placed under the variable Ytrain as a dummy variable. Xtrain is the remaining data of the pixel information for each image.
4. Open the pattern recognition toolbox from the Apps and open the Neural Network Program Recognition Tool. We used the command *nprtool* in the command window to open this toolbox.
5. Click next on the welcome screen.
6. Select Xtrain as inputs, and Ytrain as targets. Remember that all the data was stored in rows per image, so select matrix rows for the sample layout.

7. Click next to go to Validation and Test Data, here we will accept the default settings and click next again. These default settings separate the training process into a 70-15-15 ratio of training, validation, and sets.
8. In the Network Architecture we will set the value of hidden neurons to 100. Then click next again.
9. The train network will pop up. Select "Train" button to begin training the program. This may take a few minutes. When complete select next, and next again skipping Evaluate Network.
10. In "Deploy Solution" click NATLAB Matrix-Only Function" and save the generated code.
11. You can save the script and the model you just created in the "save Results" window.
12. Lastly, we will test the trained program. To do this we will take the black pen and paper and draw a number. Then import it into MATLAB and convert to gray scale with 28 by 28 pixels in one row so it matches the sample data. Then call the trained function forward and see if identifies the correct output for the image of the written number input.

Results:

In developing our artificial Neural Network, we first trained our model using the Neural Network Program Recognition Tool. Inside the application, we specified our inputs, outputs, and the number of hidden neurons. This can be seen in the figure below.



enough.

Optionally perform additonal tests

Inputs: TargetXtrain

Targets: Ytrain

Samples are: ○ Matrix columns ● Matrix row

Inputs 'TargetXtrain' is a 42000x784 matrix, representing static data: 42000 samples of 78 elements.

Targets 'Ytrain' is a 42000x10 matrix, representing static data: 42000 samples of 10 elem

CE	5.93370
%E	1.53333

Figure 2. Neural Network Pattern Recognition Tool

After training, we then used microsoft's paint application to draw our own numbers, modified to be the correct size in pixels. Some examples are shown below in figure.

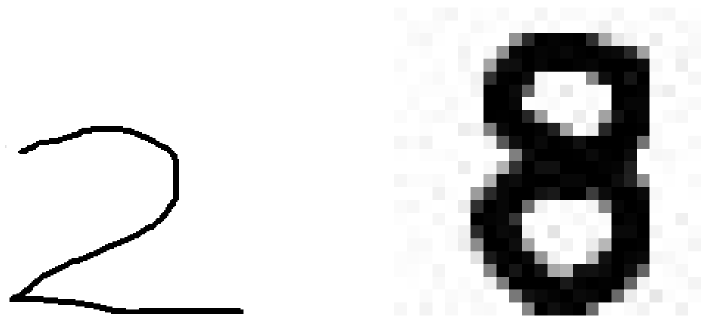
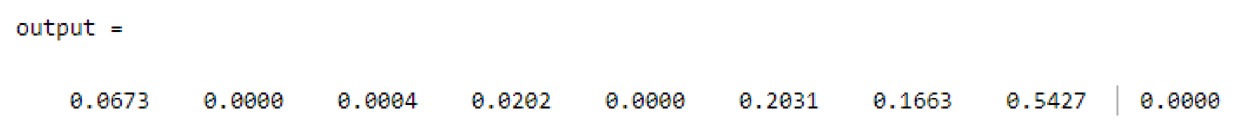


Figure 3. Sample images created in Paint

With our model trained, we sent our new images through the model, to attempt to pinpoint the correct number provided. As seen below, our model was able to correctly identify the sent image, 8.



```
output =

    0.0673    0.0000    0.0004    0.0202    0.0000    0.2031    0.1663    0.5427    0.0000
```

Figure 4. Our Neural Network's Output

In future versions, this output code could be modified to guess the correct number and output that instead based on a threshold model.

Discussion and Conclusion

Overall, the team was successful in achieving a fully working ANN. Our initial efforts trained a Neural Network which regardless of the input number the program would put out the number 8. After retraining the network, the program would only put out the number 9. After retraining the network for a third time the team was able to achieve 50 percent accuracy on the outputs. Given that this classification accuracy is much better than guessing, we count our neural network as successful. Troubleshooting this program was very tricky and the team struggled to determine the cause of the wrong outputs. After doing some post lab research ANN networks can be very sensitive during the training process because they are based on non-mathematical properties therefore there is room for interpretation. It could be possible that we did not have a large enough sample size with enough diversity for the network to form sufficient identification pathway.

Attached Code

```
%% Lab for for BRAE 428
% By: Chandler Jones
% Grace Speth
% Emma Gray

%% Part 1
data = csvread('lab5Data.csv',1,0);
% Show figure

figure                                              % plot images
image = zeros(28,28);
colormap(gray)
% set to grayscale
for i = 1:25                                        % preview first 25 samples
    subplot(5,5,i)                                  % plot them in 5 x 5 grid
    for k=1:28
        image(k,:) =data(i,28*(k-1)+2:28*k+1);
    end
    imagesc(image)
    title(num2str(data(i, 1)))                      % show the label
end

%% Section 4

%% Data Preparation
labels  =data(:,1);% 1st column is |label|
labels(labels == 0) = 10;          % use '10' to present '0'
% The labels range from 0 to 9, but we will use '10' to represent '0'
% because MATLAB is indexing is 1-based.
Ytrain = dummyvar(labels);        % convert label into a dummy variable.
TargetXtrain = data(:,2:end);           % Input pixels values.

%% Test the model

pic = imread('n4.png');
figure
imshow(pic)
pic1 = rgb2gray(pic);


I = [];
for k =1:28
    I = [I pic1(k,:)];
end

output = NN2(double(I))
```