Alphabet Recognition System Based on Artificial Neural Network

Myat Thida Tun
Lecturer, Department of Information Technology
Technological University, Republic of the Union of Myanmar
Thanlyin, Banbwegon, Myanmar (Burma)

ABSTRACT
One of the most classical applications of the Artificial Neural Network is the character recognition system. This system is the base for many different types of applications in various fields, many of which are used in daily lives. Cost effective and less time consuming, businesses, post offices, banks, security systems, and even the field of robotics employ this system as the base of their operations. For character recognition, there are many prosperous algorithms for training neural networks. Back propagation (BP) is the most popular algorithm for supervised training multilayer neural networks. In this thesis, Back propagation (BP) algorithm is implemented for the training of multilayer neural networks employing in character recognition system. The neural network architecture used in this implementation is a fully connected three-layer network. The network can train over 16 characters since the 4-element output vector is used as output units. This thesis also evaluates the performance of Back propagation (BP) algorithm with various learning rates and mean square errors. MATLAB Programming language is used for implementation.

Keywords: Artificial Neural Network, Back propagation, Character Recognition

I. INTRODUCTION
Character Recognition is used widely today in the post offices, banks, airports, airline offices, and businesses. Address readers sort incoming and outgoing mail, check readers in banks capture images of checks for processing, airline ticket and passport readers are used for various purposes from accounting for passenger revenues to checking database records. Character Recognition software is also used in scanners and faxes that allow the user to turn graphic images of text into editable documents. Newer applications have even expanded outside the limitations of just characters. Eye, face, and fingerprint scans used in high-security areas employ a newer kind of recognition. More and more assembly lines are becoming equipped with robots scanning the gears that pass underneath for faults, and it has been applied in the field of robotics to allow robots to detect edges, shapes, and colors.

Character recognition is one of the most widely used applications of Neural Networks. However, the problems of slow training process, choosing suitable values for the parameters and escaping from local minima remain a major problem that face the developers when using Neural Network. In recent years, a pen-based computer has become popular as an input device. Since character recognition systems are the preferred input methods, there are many character recognition methods that have been introduced. In this thesis, Back propagation (BP) is developed and trained to recognize characters. Back propagation (BP) applies a non-linear function on the error from each output unit before applying the back propagation phase. The (BP) aims to speed up the training process and escape from local minima.

II. PROPOSED IMPLEMENTATION STEPS OF ALPHABET RECOGNITION
The steps needed for the implementation of alphabet recognition are described in the following articles.
A. Creating the Alphabet Recognition
The Character Recognition System must first be created through a few simple steps in order to prepare it for presentation into MATLAB. The matrixes of each letter of the alphabet must be created along with the network structure. In addition, one must understand how to pull the Binary Input Code from the matrix, and how to interpret the Binary Output Code, which the computer ultimately produces.

B. Alphabet Matrixes
A character matrix is an array of black and white pixels; the vector of 1 represented by black, and 0 by white. They can be created by the use of MATLAB, in size or font imaginable; in addition, multiple fonts of the same alphabet may even be used under separate training sessions. In this thesis, 24 x 21 size of character matrix is used and Tahoma font type is used for testing.

C. Neural Network
The network receives the 504 Boolean values as a 504-element input vector. It is then required to identify the letter by responding with a 4-element output vector. In this thesis, since 16 English characters (A-P) are trained and tested, the system yields 16 4-element output vectors. Each output vector represents a letter. To operate correctly, the network should respond with a value closing to the corresponding targeted output vector of the letter being presented to the network.

D. Architecture
The neural network needs 504 inputs and 4 neurons in its output layer to identify the letters. The sigmoid transfer function was picked because its output range (0 to 1) is perfect for learning to output Boolean values. The hidden layer has 84 neurons.

E. Setting the Weight
There are two sets of weights; input-hidden layer weights and hidden-output layer weights. These weights represent the memory of the neural network, where final training weights can be used when running the network. Initial weights are generated randomly there, after; weights are updated using the error (difference) between the actual output of the network and the desired (target) output. Weight updating occurs each iteration, and the network learns while iterating repeatedly until a net minimum error value is achieved.

F. Training
The network is first trained on ideal vectors until it has a low mean squared error. Then, the network is trained on all sets of vectors. The network is trained on noise-free alphabet to maintain the network's ability to classify ideal input vectors. All training is done using back propagation with adaptive learning rate.

III. IMPLEMENTING THE BACK PROPAGATION LEARNING ALGORITHM FOR A FULLY CONNECTED THREE LAYER NETWORK
Neural networks are developed and trained to recognize characters. Back propagation (BP), which applies a non-linear function on the error from each output unit before applying the back propagation phase. The BP aims to speed up the training process and escape from local minima.

1. Apply the input vector to the input units.
2. Calculate the net-input values to the hidden layer units.
3. Calculate the outputs from the hidden layer.
4. Calculate the net-input values to each unit.
5. Calculate the outputs from each output unit.
6. Calculate the error term for the output units.
7. Calculate the error term for the hidden units.
8. Update weights on the output layer.
9. Update weights on the hidden layer.

IV. EXPERIMENTAL PARAMETERS
Two three-layered neural networks are implemented, each with different learning parameters. Figure shows the training set for each neural network. Although there exist numerous variants of the classical back propagation training algorithm.

---

Fig.1 Training Set for Each Neural Network

The experiments were made on recognizing characters from A to P. Characters are represented as N-vector which is given to the input layer of each neural network. As shown in Fig.1 the number of pixels (number of units in input layer) of the first neural network is 504 (24x21 matrix). Black pixels are represented by 1’s and white pixel by 0’s (i.e. Binary input values were used).
After main program has opened, the cover window appears as shown in Figure 2. In this screen, there are two submenus under Main menu. If the user wants to train the data set, the submenu “Train” can be chosen. Otherwise, the submenu “Test” can be chosen if the user want to test. If the user chooses the “Train” menu, the command window appears as in Fig. 5.4, requesting the user to give desired learning rate and mean square error. Then the training program perform its task (training data set) and outputs the results of total iteration number, total time taken (in seconds), mean squared error and actual output vector as shown in figure.

*************** RESULTS  ***************
Total_Number_of_Iterations_Taken = 291
Total_Time_Taken_in_Seconds = 74.8111
Mean_Square_Error = 9.9382e-006
Output Vector* 
0 0 0 0 0
0 0 0 1 1
0 0 1 0 0
0 0 1 1 1
0 1 0 0 0
0 1 0 1 1
0 1 1 1 0
0 1 1 1 1
1 0 0 0 0
1 0 0 1 1
1 0 1 0 0
1 0 1 1 1
1 1 0 0 0
1 1 0 1 1
1 1 1 1 1

Fig.3 The Output Results of Training Program for MSE = 0.00001

<table>
<thead>
<tr>
<th>Learning Rate</th>
<th>Iterations</th>
<th>Time Taken(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>899</td>
<td>196.3616</td>
</tr>
<tr>
<td>0.2</td>
<td>437</td>
<td>95.9667</td>
</tr>
<tr>
<td>0.3</td>
<td>291</td>
<td>74.8111</td>
</tr>
<tr>
<td>0.4</td>
<td>215</td>
<td>49.2186</td>
</tr>
<tr>
<td>0.5</td>
<td>156</td>
<td>35.8624</td>
</tr>
</tbody>
</table>

Table.1 The results of training process for MSE = 0.00001
Table shows the results of training process with MSE = 0.00001 for five different learning rates. From this table, it is found that the more learning rate, the less iteration number and the faster the trained network reaches to desired goal (mean square error).

<table>
<thead>
<tr>
<th>MSE</th>
<th>Iterations</th>
<th>Time Taken (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0001</td>
<td>40</td>
<td>2.8467</td>
</tr>
<tr>
<td>0.00001</td>
<td>288</td>
<td>64.0111</td>
</tr>
<tr>
<td>0.000001</td>
<td>2390</td>
<td>528.8830</td>
</tr>
<tr>
<td>0.0000001</td>
<td>21888</td>
<td>4.6590e+003</td>
</tr>
</tbody>
</table>

Table 2 shows the results of training process with learning rate = 0.3 for five various mean square error parameters. Firstly, the user uses the learning rate parameter = 0.3 for mean squared error = 0.0001. Then, the result output can be seen as the total iteration numbers are 40 and the time taken is 2.8467 seconds.

<table>
<thead>
<tr>
<th>MSE</th>
<th>Iterations</th>
<th>Time Taken (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0001</td>
<td>40</td>
<td>2.8467</td>
</tr>
<tr>
<td>0.00001</td>
<td>288</td>
<td>64.0111</td>
</tr>
<tr>
<td>0.000001</td>
<td>2390</td>
<td>528.8830</td>
</tr>
<tr>
<td>0.0000001</td>
<td>21888</td>
<td>4.6590e+003</td>
</tr>
</tbody>
</table>

Table 3 shows the results of training process with learning rate = 0.5 for five various mean square error parameters. Firstly, the user uses the learning rate parameter = 0.5 for mean squared error = 0.0001. Then, the result output can be seen as the total iteration numbers are 157 and the time taken is 38.9225 seconds.

Table 3 shows the results of training process with learning rate = 0.5 for five various mean square error parameters. Firstly, the user uses the learning rate parameter = 0.5 for mean squared error = 0.0001. Then, the result output can be seen as the total iteration numbers are 157 and the time taken is 38.9225 seconds.

This paper explains the processing steps of Character Recognition System. The user can be easily seen the processing and running steps of Neural Network programs. This thesis aims to apply the classical BP algorithm on one of the Back propagation neural network applications, which is character recognition. In this thesis, to obtain fair and independent results, network architecture was trained with various MSE values for the different learning rate at each time. The experimental results show that using the BP can speed up convergence of the training process and recognize trained characters (A to P) from the Image Database.

**VI. CONCLUSIONS**

This paper aims to apply the BP algorithm on one of the Back propagation neural network application, which is character recognition. The architecture of 504×84×4 was used with various values for the learning rate. The experimental results show that using the BP can speed up convergence of the training process. Although small values for the learning rate were used, the BP was able to train the neural network. The performance of recognition can be increased when the input training data files are increased. The size of database set is presented as an important role of the training section. The Back propagation Neural Network is discussed in the Chapter four of this thesis. It can also be used for almost any image recognition applications. The network is a 3 layer configuration, with the required input and output layers, as well as a single hidden layer that choosing the number of hidden layers is a difficult task with no hard rules or guidelines. However, the size of a hidden layer is related to the features or distinguishing characteristics that are to be discerned from the data. Here, the network applies one hidden layer, which has 84 neurons. The output
layer is where the output vector can be retrieved. Each neuron in this layer outputs a value between 0 and 1 by applying the logical sigmoid function \( \text{sig} = 1 / (1 + \exp(-1*\text{s})) \). In addition, to obtain the accurate output values (0 or 1), the threshold value of 0.5 is used when calculating each neuron’s output.

REFERENCES