International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 5, Issue 11, November 2015)

Character Recognition with Neural Networks using Fuzzy Feature Extraction

Dharmendra Hingu[1], Harnisha Gevaria[2]
[1,2]*Veermata Jijabai Technological Institute*

***Abstract* — Calculating systems can be made on the replica of biological neural network was suggested 50 years ago, the progress made in this area lead to improvement of these systems only in last 20-25 years [1]. Numerous advancements have been made in developing intelligent systems. This paper discusses about Character Recognition System using Artificial Neural Network (ANN). Artificial Neural Networks are used for classification, prediction and recognition. Competitive neural networks set the different neurons against each other, hoping that the "winner" will be close to the answer. With the help of Matlab's Neural Network Toolbox, we tried to recognize printed and handwritten characters which are projected on different sized grids. Preprocessing step is acquisition of image which requires the scanning of image followed by filtering of noise, smoothing and normalization and eventually rendering image suitable for segmentation. After that the image is decomposed into numerous sub-images. Extracting features from image improves rate of recognition. Here, the ANN is trained using back proposition algorithm. The characters are represented in binary, input to feature extraction system, output of which then fed to trained neural network. Also, we found, that it is difficult to recognize every writing style using same neural network. Hence, variety of handwritings must be trained first to the neural network. Fuzzy logic can help to tackle this problem.**

***Keywords* – Fuzzy logic, Optical Character Recognition, Neural Network, Multi-layer feed forward neural network, back propagation algorithm, feature extraction**

I. INTRODUCTION

Perceptron by Rosenblatt's, was one of the first neural network able to recognize fixed-font character set. Fuzzy logic theory proposed by Zadeh, changed many of our conceptions about methods to solve complex problems. Neural Networks are a powerful tool to deal with many pattern recognition problems. In many cases supervised learning approach is assumed: A system is trained using a set of labeled examples suggesting for input pattern the intended output. The character recognition is possible using Image Processing, Pattern Recognition and Artificial Intelligence. The goal of optical character recognition is to classify the alphanumeric samples which are saved as digital pictures. The character recognition is multi-step process.

In this paper we focus on recognizing the characters using various classes and different classification strategies. Section II presents the fundamentals of neural networks and fuzzy logic. Section III sheds light on supervised learning and learning procedure.

The proposed system is discussed in Section IV, conclusions are drawn in Section V.

II. METHODOLOGY

The recognition system consists of several steps: Preprocessing, training the neural network and recognition. The following concepts and algorithms are important to understand the proposed character recognition system. These methods define the way in which the neural network is given the input that is processed using different techniques, adapting its parameters to recognize the wanted character-set.

A. Fuzzy Feature Extraction

Let us assume that we have a binary image of a character as an input data. Before applying any other algorithms we have to realize the pre-processing of the image (Fig. 1).

Filtering: Drops the noise making easier extraction of structural features.

Thinning: Also referred as Skeletonizing, A morphological operation that is used to remove selected foreground pixels from binary images.

Searching vertices: At this point we obtain a line junction or end of lines also referred as vertices. Each vertex has a number of branches which is co-located at this point.

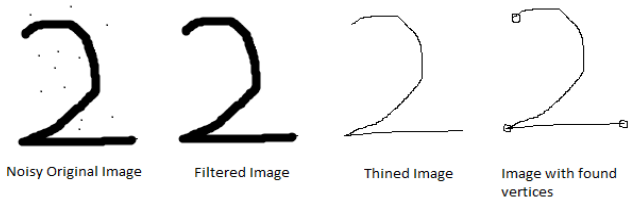

Fig. 1. Pre Processing

The pre-processing of the image allows us to estimate the influence of the obtained results onto quality of the recognition. After that the fuzzy feature extraction algorithm is employed. The task of feature extraction of a character is to separate typical elements and branches or segments.

The algorithm is as follows,

The feature extraction algorithms are,

I. Zoning
II. Projections and Profiles
III. Crossing and Distance

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 5, Issue 11, November 2015)

Input:

I. m x n binary image matrix I,
II. Vertices structure V,
III. Number of vertices V_c.

Fuzzy data structure F of:

I. Symbols of classified segments,
II. Bound boxes of segments,
III. Position of segments,
IV. Co-ordinates of the segment (starting and ending points)
V. Sizes of segments.

These extracted features become the input to our trained neural networks.

B. Digitization; Matrix Creation from Character Image; Edge Detection and Normalization

If the given character image is colored, first we need to digitized it into matrix i.e. converting it to binary. Colored image is first converted to the gray scale image as follows [3]:

GrayScale = (int) (0.33 * RedPix + 0.56 *GreenPix +0.11 *BluePix)

Where, RedPix is the red component of a pixel, GreenPix is the green component of a pixel, BluePix is the blue component of a pixel and GrayScale is the gray scale level value of a pixel. The gray scale image is then converted to binary by replacing all the gray values of the image into binary value 0 (for values 129 to 255) interpreted as pixel present or 1 (for values 0 to 128) interpreted as pixel present.

After obtaining binary matrix from the input image, we need to employ edge detection algorithm for correclty recognizing the character. The steps to detect an edge is given by, [4]

I. *Top Edge Detection*: Scan the character matrix starting at the top-left corner and delete all rows from top having only 0's. If there is at least two consecutive 1's in two consecutive rows, then the first row of the two consecutive rows from top becomes top edge.
II. *Bottom Edge Detection*: Scan the character matrix starting at the bottom-left corner and delete all rows from bottom having only 0's. If there is at least two consecutive 1's in two consecutive rows, then the first row of the two consecutive rows from bottom becomes bottom edge.
III. *Left Edge Detection*: Scan the character matrix starting at the top-left corner and delete all columns from left having only 0's. If there is at least two consecutive 1's in two consecutive columns, then the first column of the two consecutive columns from left becomes left edge.
IV. *Right Edge Detection*: Scan the character matrix starts at the top-right corner and delete all columns from right having only 0's. If there is at least two consecutive 1's in two consecutive columns, then the first column of the two consecutive columns from right becomes right edge.

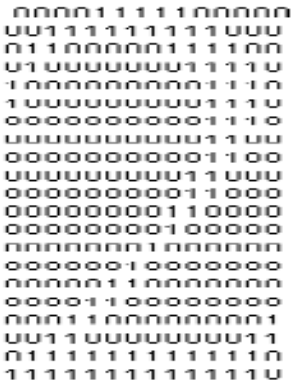

Fig. 2. Edge Detection of binary image

After Edge detection, we need to normalize the resultant image. Normalization is the process of making the size of all extracted character bitmaps same. For size independent character recognition, we have converted the edge detected input character image matrix into 12×8 normalized image matrix. The steps to normalize are listed below.

I. Pick top row and left column with an assumption that they contain salient features
II. Then take alternate row and column until desired 12×8 character matrix is found. For example, the size of input character (after boundary detection) is 15×11.Then during normalization,
    a. Pick first row
    b. Remove all even row (i.e. 2,4,6)
    c. Consider all the remaining rows
    d. Now Converted the matrix to 12×11
III. Same process for no. of column. Finally we have the matrix is converted to 12×8

Now apply the fuzzy extraction algorithm.

III. Supervised Learning & Learning Procedure

The features retrieved in using section II methods are then fed to Neural Network (Fig. 3). We use back propagation (BP) algorithm for better recognition of the characters. The backpropagation algorithm's most specific feature is the error that the neural network gets on its output. This is error is equal with the difference between the real and required output values. When using backpropagation, we look at the weight/change, which mostly minimizes the error in the output [2].

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 5, Issue 11, November 2015)

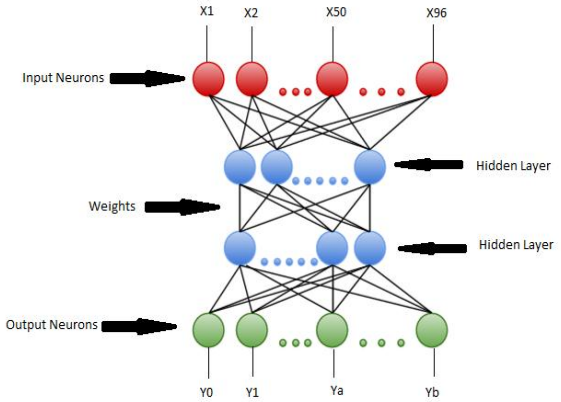

Fig. 3. Multi Layered Neural Network

The algorithm is a method that depends on the gradient value of the moment. The learning starts when all of the training data was showed to the network at least once. The learning method, as for every network learning algorithm, consists of the modification of the weights.

The BP algorithm works with little iterative steps: after showing all samples to the network, it will generate an output value with the pre-initialized weights. This output value will be compared to the required output value which we gave, and the mean square error is calculated. After this, our network will go backwards through the weights. It will use the gradient of the criteria-field to determine the best weight/modification to minimize the mean square error. This method will go through with all samples, minimizing the error until a level that we specified. After this, we can say, that our network learned to solve the problem good enough. The network will never learn how to solve the problem perfectly, but it will close on it asymptotically.

IV. The Proposed System

1. Training

a. Preprocessing – processing the data in the form needed.
b. Features extraction – we minimize the needed data with saving only the needed information. This will give us a vector with scalar values.
c. Estimation of the model – With a number of vectors we have to estimate the model for every class of training data.

2. Testing

a. Preprocessing
b. Features extraction
c. Classification – we compare the vector to the model with the extracted features to find the best suiting value. There are more methods to do this.

The algorithm:

i. Define a training-sample for the network.
ii. Compare the gotten output value with the required ones, and calculate the error for every output neuron.
iii. We must calculate the required output for every single neuron. We must also count the incremental factor, which shows us how much every neuron weight has to be changed, so that they will be perfect in values. This shows the local error.
iv. Modify the weight in from of every neuron in the way to minimize the local error.
v. Give a level of blame to every neuron, this way giving higher responsibility for those neurons with greater weights before them.
vi. Repeat the method from step 3 for the neurons of the previous layer, using the "blame" as factor.

The equation looks as:

$$x_{k+1} = x_k - a_k g_k$$

Where $\mathbf{x_k}$ – vector representing the momentary weights and biases

$\mathbf{g_k}$ – momentary gradient

$\mathbf{a_k}$ – learning rate

Gradient-descent

During the training session we search the optimization of the criteria-field, more exactly that "w" weight value, where the criteria-function will minimize the error level. Visually, the criteria-function is a surface that depends from a parameter. This is called the criteria-field. The finding of the values that define the maximum or the minimum of this field is our goal. So these learning functions are optimization functions. We have a lot of these methods in the neural terminology, also used in other branches of science. In case of neural networks, for us, the gradient-descent method is the most important.

This searches the minimum of the field.

$\nabla[C(w)] = \frac{\partial C(w)}{\partial w} = 0$ where ∂ is the generalized error

This will provide us with an analytic result, which can cause us problems in harder cases. In a lot of cases, we can't give an analytic result. In these cases we change the parameters of the system until we reach the minimum of the criteria-field.

The most important method when speaking about character recognition is the steepest descent or gradient descent method. This method iterates in the negative direction of the gradient. The iterative equation of it [2]:

$$w(k+1) = w(k) + \mu(-\Delta C(k))$$

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 5, Issue 11, November 2015)

V. Results And Discussions

Based on our proposed system in the preceding sections, our method is able to recognize all 10 digits (0 - 9) and 26 capital alphabets (A - Z). We have used 1500 character samples to test our system. The proposed system can recognize only numeric digits (0 - 9) with 97.89%, only alphabets (A - Z) with 96% accuracy and alphanumeric characters (0 - 9, A - Z) with more than 91% accuracy on average. Table shows Empirical Results.

Character	No. of Input Sample	No. of Correct Identification	No. of Wrong Identification	Correctness (%)
2	20	16	4	80
Z	20	18	2	90
0	20	20	0	100
O	20	19	1	95
8	20	19	1	95
B	20	18	1	90

VI. Conclusions

Fuzzy logic and NN can be applied at many levels in complex handwriting recognition process. Fuzzy logic systems can represent high-level knowledge.

In this paper, we have proposed an artificial neural network-based simple character recognition system, including colored image to recognize alpha-numeric characters. Our proposed system gives excellent result when they trained and tested individually but produces only satisfactory result when processed together. Also the proposed system is easy to implement.

REFERENCES

[1] Hertz J., Krogh A., Palmer R. G. – "Introduction to the Theory of Neural Computation" (1991)

[2] Altrichter Marta, Horvath Gabor, Pataki Bela, Strausz Gyorgy, Takacs Gabor, Valyon Jazsef – "Neuralis Halozatok" (2006)

[3] R. C. Gonzalez and R. E. Woods, "Digital Image Processing", 2nd Ed.

[4] M.F. Kader,M.K.Hossen, Asaduzzaman and A.S.M Kayes - "An Offline Handwritten Signature Verification System as a Knowledge Base" (2006)

[5] Alexander J. Faaborg (Cornell University, Ithaca NY) – "Using Neural Networks to Create an Adaptive Character Recognition System" (2002)

[6] Md Fazlul Kader & Kaushik Deb – "Neural Network-Based English Alphanumeric Character Recognition" (2012)

[7] Rókus Arnold & Póth Miklós – "Character Recognition Using Neural Networks" (2010)

[8] S.Yanushkevich, Phil Phillips, J. Gilewski and D.Popel – "Handwriting Recognition Neural Networks Fuzzy Logic" (2010)