International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:3, 2008

Handwritten Character Recognition Using Multiscale Neural Network Training Technique

Velappa Ganapathy, and Kok Leong Liew

***Abstract*—**Advancement in Artificial Intelligence has lead to the developments of various "smart" devices. Character recognition device is one of such smart devices that acquire partial human intelligence with the ability to capture and recognize various characters in different languages. Firstly multiscale neural training with modifications in the input training vectors is adopted in this paper to acquire its advantage in training higher resolution character images. Secondly selective thresholding using minimum distance technique is proposed to be used to increase the level of accuracy of character recognition. A simulator program (a GUI) is designed in such a way that the characters can be located on any spot on the blank paper in which the characters are written. The results show that such methods with moderate level of training epochs can produce accuracies of at least 85% and more for handwritten upper case English characters and numerals.

***Keywords*—**Character recognition, multiscale, backpropagation, neural network, minimum distance technique.

I. Introduction

CHARACTER recognition is a form of pattern recognition process. In reality, it is very difficult to achieve 100% accuracy. Even humans too will make mistakes when come to pattern recognition. Pattern distortion, presence of unwanted objects or disoriented patterns will affect the percentage accuracy. The most basic way of recognizing patterns is through using the probabilistic methods. This has been done by using the Bayesian decision theory as mentioned by Po, H.W, [1] and Liou C.Y. & Yang, H.C, [2].

Another alternative method in pattern recognition is the k-nearest neighbor algorithm used in Dynamic Classifier Selection by Didaci, L. & G, Giacinto [3]. In k-nearest neighbor (k-NN) algorithm, the pattern class (say x) is obtained by looking into k number of nearest pattern sets that have the least Euclidean distance with that pattern itself (pattern x).

Common ways used for character recognition would be the use of artificial neural networks and feature extraction methods as in Brown, E.W, [4]. Neural network such as feedforward backpropagation neural network requires long training time just to "memorize" all possible input vectors that are fed into the network. However, there is always a possibility that the network will give false results due to poor generalization capability. Generalization problems can be overcome by using the Multiscale Training Technique (MST) [5], which is the concept that will be emphasized in this paper with modifications in the input training vectors.

The recognition accuracy of the handwritten characters depends a lot on the exemplars that are used for recognition. In general, the overall recognition process can be divided into 3 main sections, namely segmentation, preprocessing, and classification [6]. Segmentation requires isolating the characters individually before they are fed to the preprocessing unit where the important features of characters (feature extraction) are identified. Finally, classification process is done by determining the category or the group of each character used during the recognition process.

II. Overview of the Proposed Character Recognition Simulator Program and Methodology

The characters were prepared on a piece of blank paper. Black Artline70 marker pen was used to write down all of the characters. Samples of 10 for each character (each having different style) were used, hence a total of 360 characters were used for recognition (26 upper case letters + 10 numerical digits). These characters were captured using a digital camera, Canon IXUS 40. The captured images were then used to generate input vectors for the backpropagation neural network for training. In this case, multiscale technique with selective thresholding is proposed and the results using this method will be compared with other methods. For simulation, separate character sets were also prepared using the same method as mentioned previously. The characters were captured automatically using the GUI created and the captured images were stored into a temporary folder for quick reference. Input vectors are generated and fed into the trained neural network for simulation.

This simulator program is designed in such a way that the characters can be written on any spot on the blank paper. The program has the ability to capture the characters through careful filtering. 8-connectivity is used to allow connected pixels in any direction to be taken as the same object.

Manuscript received :31st March 2008. This work was supported by Monash University, Sunway Campus, Malaysia.

Velappa Ganapathy is with the Monash University, Sunway Campus, Malaysia (phone: +603-55146250; fax: 603-55146207; e-mail: Velappa.ganapathy@ eng.monash.edu.my).

Kok Leong Liew is with Monash University Sunway Campus Malaysia (kllie3@eng.monash.edu.my).

III. Exemplars Preparation

The characters prepared as explained in Section 2, are scanned using a scanner or captured using a digital camera and these characters will be segregated according to their own character group. One example is shown below in Fig. 1.

Fig. 1 Sample of character A

Note that the scanned or captured images are in RGB scale. These images have to be converted into grayscale format before further processing can be done. Using appropriate grayscale thresholding, binary images are to be created.

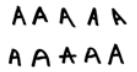

Fig. 2 Binary image of sample character A

Fig. 2 shows the generated binary image using image processing tool box in MATLAB and 8-connectivity analysis.

The next step is to obtain the bounding box of the characters (Fig. 3). Bounding box is referring to the minimum rectangular box that is able to encapsulate the whole character. The size of this bounding box is important as only a certain width-to-height (WH) ratio of the bounding box will be considered in capturing. Those objects of which bounding boxes are not within a specific range will not be captured, hence will be treated as unwanted objects. The next criteria to be used for selection of objects are relative-height (RH) ratio and also relative-width (RW) ratio. RH ratio is defined as the ratio of the bounding box height of one object to the maximum bounding box height among all the objects of that image. Likewise, RW refers to the ratio between the widths of one bounding box to the maximum width among all bounding boxes widths in that image.

Fig. 3 The outer rectangle that encapsulates the letter A is the bounding box

$$WH = \frac{w}{h} \tag{1}$$

Width-to-height ratio is defined as ratio of bounding box width, w, to bounding box height, h, of that object.

$$RW = \frac{w}{w_{\max}} \tag{2}$$

Relative-width ratio is defined as object's bounding box width, w, over maximum bounding box width among all objects, w_{max}.

$$RH = \frac{h}{h_{\max}} \tag{3}$$

Relative-height ratio is defined as object's bounding box height, h, over maximum bounding box height among all objects, h_{max}.

For example, if the RW of one object exceeds RW_{min}, (where RW_{min} is the threshold value for comparison), that object will be captured. Similar analogy goes for RH and threshold value RH_{min} which applies to bounding box height of objects. Typical values chosen for RW_{min} and RH_{min} are 0.1 and 0.3 respectively.

The captured objects should now all consist of valid characters (and not unwanted objects) to be used for neural network training. Each of the captured character images have different dimensions measured in pixels because each of them have different bounding box sizes. Hence, each of these images needs to be resized to form standard image dimensions. However, in order to perform multiscale training technique [5], different image resolutions are required. For this purpose, the images are resized into dimensions of 20 by 28 pixels, 10 by 14 pixels, and 5 by 7 pixels. Note that these objects are resized by using the averaging procedure. 4 pixels will be "averaged" and mapped into one pixel (Fig. 4).

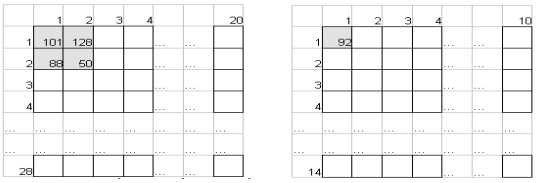

Fig. 4 The pixels shaded in gray on the left are "averaged" (computing mean of 101, 128, 88, and 50) and the pixel shaded in gray on the right is the "averaged" pixel intensity value. This example shows averaging procedure from 20 by 28 pixel image into 10 by 14 pixel image

IV. Artificial Neural Network Training

Backpropagation neural network is used to perform the character recognition. The network used consists of 3 layers and they include input layer, hidden layer, and output layer. The number of input neurons depends on the image resolution. For example, if the images that are used for training have a resolution of 5 by 7 pixels, then there should be 35 input neurons and so on. On the other hand, the number of output neurons is fixed to 36 (26 upper case letters + 10 numerical digits). The first output neuron corresponds to letter A, second corresponds to letter B, and so on. The sequence is A, B, C… X, Y, Z, 0, 1, 2 … 7, 8, 9. The number of neurons

in the hidden layer (layer 2) is taken arbitrarily by trial and error to be 1500 [7].

V. Multiscale Training Technique

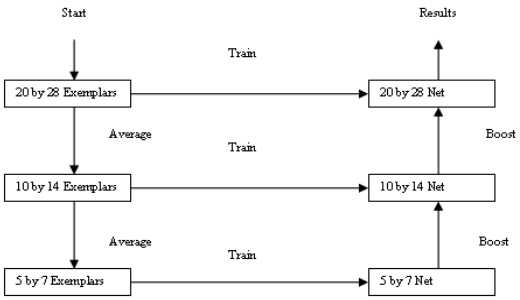

Fig. 5 Conceptual diagram of multiscale training technique

The training begins with 5 by 7 pixels exemplars (stage 1). These input vectors are fed into the neural network for training. After being trained for a few epochs, the neural network is "boosted" by manipulating the weights between the first and the second layer. This resultant neural network is trained for another few epochs by feeding in 10 by 14 pixels exemplars (stage 2). Again, the trained neural network is boosted for the next training session. In a similar fashion, the boosted neural network is fed in with 20 by 28 pixels exemplars for another few epochs until satisfactory convergence is achieved (stage 3). The conceptual diagram of multiscale neural network is shown in Fig. 5.

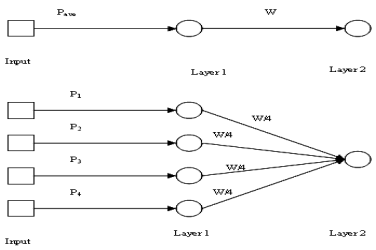

Fig. 6 Original neural network (top) and the "boosted" neural network (bottom)

Referring to Fig. 6, P_1, P_2, P_3, and P_4 are pixel intensity values and P_{ave} is the averaged pixel intensity value of these pixels. After the boosting process, the original weight value W, is split into 4, each connected to one pixel location.

VI. Image Capturing for Simulation

Exemplars for network simulation also have to go through geometrical based filtering that require parameters such as WH, RW, $RW_{min,}$ RH, and RH_{min} as shown in Fig. 7. Similarly, the captured and cropped images will be resized to standard resolutions: 20 by 28 pixels, 10 by 14 pixels, and 5 by 7 pixels.

For network simulation, it is important to ensure that the output printed characters are arranged in appropriate order. Hence, identified characters need to be reassembled after neural network simulation.

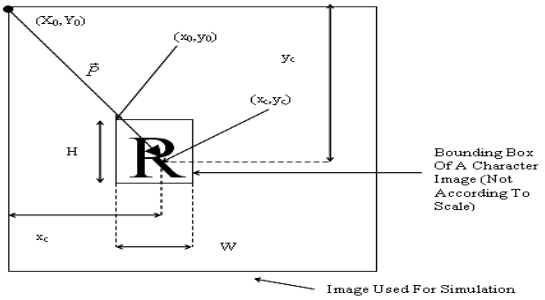

Fig. 7 The parameters used for character reassembly. Bounding box is defined with the width, w, height, h, horizontal centroid distance, xc, vertical centroid distance, yc, and vector that locates the centroid of a character from origin (X0,Y0), $\vec{P}$

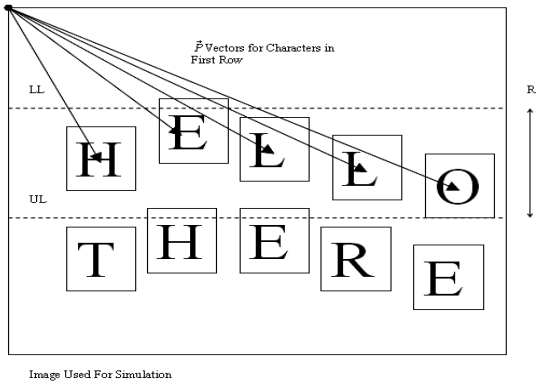

Fig. 8 "HELLO THERE" example

In order to determine which character is to be printed first, the vectors that locate the centroids of each character are obtained. The magnitudes of the vectors are to be computed simply by using $|\vec{P}| = \sqrt{x_c^2 + y_c^2}$. Next, the character with the smallest $|\vec{P}|$ is considered, which is the first character of the first row. Based on this first character, the range of search, R, is determined where R = UL – LL. The upper and lower limits, UL and LL are computed as follows:

$$UL = yc\ first + 0.7\ Hmax \quad (4)$$

$$LL = yc\ first - 0.7\ Hmax \quad (5)$$

Note that yc first is the vertical centroid distance of the first character to the Y0. Hmax refers to the largest bounding box height among all characters in the sample image. The constant, 0.7, is taken to be arbitrary. Within this search range R, the centroid positions of the characters that are located within UL and LL will be stored temporarily as these characters are actually located in the first row. Referring to Fig. 8, "HELLO" is located in the first row, but not "THERE" because the centroid positions of letters T, H, E, R, and E are not within UL and LL, hence they are neglected. Next, the characters within the first row will be arranged by

reconsidering the vector $\vec{P}$ of all these characters. The character with the next smallest $|\vec{P}|$ will be the second character (first row, second column) and so on until all of the characters in the first row have been considered. Once this is done, the vector magnitude, $|\vec{P}|$ of the remaining characters (T, H, E, R, and E) will be computed to determine the first character of the second row. Again, the one with the smallest $|\vec{P}|$ will be the first character of second row. Similar procedure is repeated to determine the remaining characters in the second row.

VII. Neural Network Simulation Using Selective Thresholding Minimum Distance Technique (MDT)

Prior to network simulation, the captured character images will need to be converted into input vectors and this step is described in the previous section, namely Artificial Neural Network Training. The input vectors will be cascaded and fed into the neural network. Unlike neural network training, the targeted output matrix is not required; instead, the network will produce an output matrix similar to the one in Fig. 9.

$$\begin{bmatrix} 0.2 & 0.8 & 0.7 & 0.2 & 0.02 \\ 0.9 & 0.2 & 0.11 & 0.01 & 0.1 \\ 0.15 & 0.3 & 0.12 & 0.88 & 0.6 \end{bmatrix}$$

Fig. 9 An example of output matrix after neural network simulation

Again the assumption here is 3 possible outputs with 5 character samples. If the first row, second row, and third row correspond to characters A, B, and C respectively, then the first character (first column) is giving an output of character B because the highest output value is 0.9 which is located at the second row. Similar way goes for the rest. The output characters for this example would be BAACC.

However, there is a possibility that the neural network might produce wrong output characters due to character clashing and to reduce such possibility, selective thresholding minimum distance technique is used. Selective thresholding requires the use of certain heuristic to be used in determining whether minimum distance technique is applicable in that situation. For instance, if the generated output vector for one particular sample is $[0.33\ 0.81\ 0.72]^T$, the output character would be letter B (using the same assumption used previously for 3 possible outputs). However, it can be observed that letter C is quite likely to be the correct character output as well due to close output values for second and third row (0.81 and 0.72). To resolve this problem, differential ratio, dfr, is computed and will be compared with a certain threshold value to determine if minimum distance technique should be applied to resolve such clashing.

$$\mathrm{dfr}_{1,2} = \frac{O_1 - O_2}{O_1} \quad (6)$$

Differential ratio is calculated by computing the difference between highest output value O_1 and second highest output value O_2 of that sample as a ratio to O_1.

The calculated $\mathrm{dfr}_{1,2}$ will be compared against the threshold $\mathrm{th}_{1,2}$ where the subscripts 1 and 2 refer to the output values used (1 means highest output, 2 means second highest output). Note that $\mathrm{th}_{1,2}$ is not fixed (different pair of characters has different $\mathrm{th}_{1,2}$) and is determined based on certain algorithm (selective thresholding). If $\mathrm{th}_{1,2} \geq \mathrm{dfr}_{1,2}$, then minimum distance would be applicable. Minimum distance, MD, simply means the sum of the squared differences of the corresponding pixel intensity values between a pair of image set (template image and the input sample image).

$$MD = \sum_{i=1}^{m} \sum_{j=1}^{n} (a_{ij} - b_{ij})^2$$

Equation 7: Given template image and input image samples with m number of rows and n number of columns, the MD is computed as shown, where ai,j and bi,j are pixel intensity values for template image and input image respectively located at i-th row and j-th colum*n*

Referring back to output vector example $[0.33\ 0.81\ 0.72]^T$, the character output can be either B or C. Thus, 2 sets of MD is required where the first set, MD_1 is between template image character B with the input image sample, whilst the second set, MD_2 is between template image character C with input image sample. If $MD_1 > MD_2$, then character C is the most possible output due to smallest difference in terms of pixel intensity value between input sample and template character C.

VIII. Results and Discussion

Comparisons were made between the neural networks that were trained using the brute force method and the ones using multiscale training method. The results are shown below. In addition, comparisons were also made between ordinary network simulation and the ones using selective Minimum Distance Technique.

Results

In Fig. 10 curves, 1, 2 and 3 indicate MST stage 1, stage-2 and stage-3 respectively and curve 4 indicates the neural network trained using brute force. Note that MST x-y-z means that the neural network is trained using multiscale training technique [5] for x time units in stage 1, y time units in stage 2, and z time units in stage 3.

Training Results on Time Axis

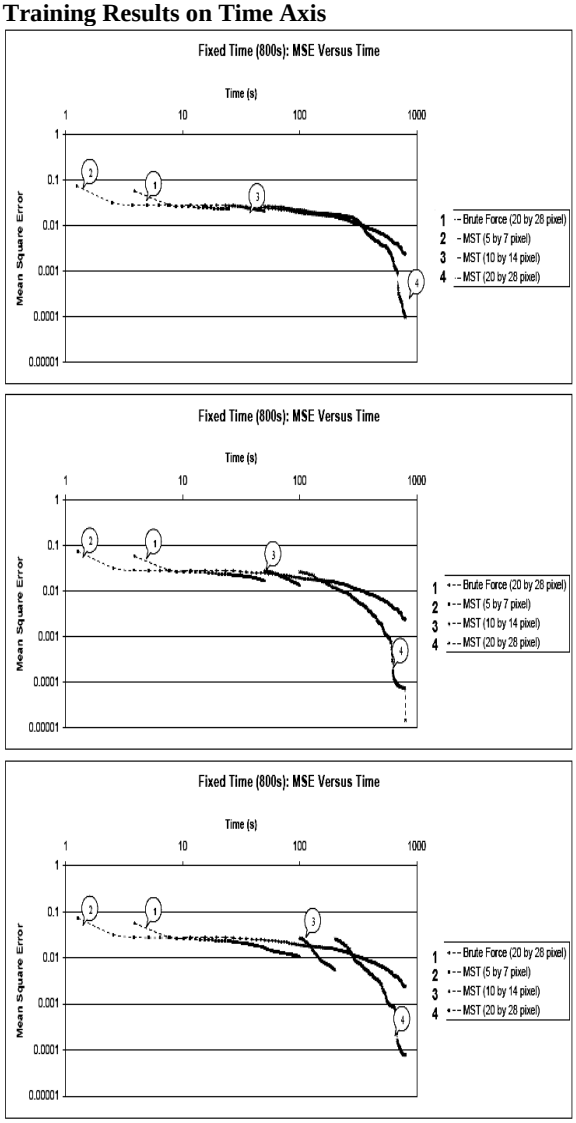

Fig. 10 Graphs of Mean Square Error (MSE) versus Time units. Comparison made between brute force and MST 25-25-150 (left), MST50-50-100 (middle), and MST 75-75-50 (right)

Neural Network Parameters used in the Experiment

Number of hidden neurons = 1500
Number of epoch = 200
Training algorithm = 'trainscg'
Transfer function used in hidden layer = 'tansig'
Transfer function used in output layer = 'logsig'

Simulation Results

Stage	MST 25-25-150	MST 50-50-100	MST 75-75-50
1	41.67	53.61	57.22
2	53.33	69.17	74.72
3	82.78	84.17	76.67

Fig. 11 Simulation results showing the percentage of correctly identified characters. For brute force method (not shown in table), percentage accuracy was only measured at the end of stage 3. The accuracy measured was 73.61%

Neural Network	Accuracy (%) (A-J)	Accuracy (%) (K-T)	Accuracy (%) (U-3)	Accuracy (%) (4-9)	Average Accuracy (%)	Total Time (s)
20 by 28	86	88	82	85	85.27778	28.0922
10 by 14	90	86	75	88.3333	84.44444	28.8411
5 by 7	91	73	61	85	76.66667	29.0648

Neural Network	Accuracy (%) (A-J)	Accuracy (%) (K-T)	Accuracy (%) (U-3)	Accuracy (%) (4-9)	Average Accuracy (%)	Total Time (s)
20 by 28	86	89	88	85	87.22222	30.3464
10 by 14	92	89	77	88.3333	86.38888	31.8727
5 by 7	93	72	62	86.6667	77.50001	33.6127

Fig. 12 Comparison between ordinary network simulation (top) and the one with selective thresholding minimum distance technique (bottom)

Discussion

It is shown that MST training allows faster convergence. MST training enables large resolution images (20 by 28 pixels) to be used for training at much less time compared to brute force that uses large resolution images throughout the entire training process. MST makes use of smaller resolution images for speed and larger resolution images for accuracy. Multiple stages of training allow the network to make better use of training time compared to ordinary brute force method. In terms of percentage accuracy, MST networks generally produce greater number of correctly identified characters as percentage of total number of characters in simulation samples compared to brute force network (with only 73.61% in the example considered). This proves that MST networks have greater generalization ability. Fig. 10 gives the comparison made between brute force and MST 25-25-150 (top), MST 50-50-100 (middle), and MST 75-75-50 (bottom).

However, different MST configurations will produce different degrees of accuracies as shown in Fig. 11. Results also show that networks that use selective thresholding minimum distance technique generally produce higher percentage accuracy compared to the networks that do not use it. This is shown in Fig. 12.

IX. Conclusion

When the resolution of the character images grows larger, neural network training tends to be slow due to more processing for larger input matrix. If the character images have lower resolution, the training process is much faster. However, some important details might be lost. Hence, it is a tradeoff between image resolution and training speed to recognize hand written characters. To optimize between these two parameters, it has been shown that one can adopt the multiscale training technique with modifications in input vectors as it provides faster training speed for different image resolutions. On the other hand, it is shown that selective thresholding MDT can also be used to increase the percentage accuracy of the identified characters at the cost of simulation time. Results also show that networks that use selective thresholding minimum distance technique generally produce higher percentage accuracies compared to the networks that

do not use it. Efficient algorithm is still to be explored to determine the appropriate threshold level to allow MDT to be used effectively.

REFERENCES

[1] Wu, P.H. (2003), Handwritten Character Recognition, B.Eng (Hons) Thesis, the School of Information Technology and Electrical Engineering, the University of Queensland.

[2] Liou, C.Y. & Yang, H.C. (1996), "Hand printed Character Recognition Based on Spatial Topology Distance Measurement", IEEE Transactions On Pattern Analysis and Machine Intelligence, Vol. 18. No. 9, pp 941-945.

[3] Didaci, L. & Giacinto, G. (2004), *Dynamic Classifier Selection by Adaptive k-Nearest-Neighbourhood Rule*, Available: http://ce.diee.unica.it/en/publications/papers-prag/MCS-Conference-19.pdf (Accessed: 2007, October 11th).

[4] Brown, E.W. (1993), *Applying Neural Networks to Character Recognition*, Available: http://www.ccs.neu.edu/home/feneric/charrecnn.html (Accessed: 2007, October 11th).

[5] Robinson, G. (1995), *The Multiscale Technique*, Available: http://www.netlib.org/utk/lsi/pcwLSI/text/node123.html (Accessed: 2007, October 11th).

[6] *Handwritten Character Recognition*, Available: http://tcts.fpms.ac.be/rdf/hcrinuk.htm (Accessed: 2007, October 11th).

[7] Rivals I. & Personnaz L. A statistical procedure for determining the optimal number of hidden neurons of a neural model. Second International Symposium on Neural Computation (NC.2000), Berlin, May 23-26 2000.

Velappa Ganapathy was born on 1st May 1941 at Singalandapuram, Salem, India. He had obtained his Bachelor of Engineering in Electrical & Electronics Engineering and Master of Science in Electrical Engineering both from the University of Madras, India. He did his PhD in Electrical Engineering from the Indian Institute of Technology, Madras, India. He had worked in various capacities as Associate Lecturer, Lecturer, Assistant Professor, Associate Professor and Professor at the Government College of Technology, Coimbatore, Anna University Chennai, Multimedia University, Malaysia and Monash University Malaysia.
His research interests are Digital Signal Processing, Robotics, Artificial Intelligence and Image Processing. Currently he is with the Monash University Malaysia.

S. B. Kok Leong Liew is with Monash University Sunway Campus Malaysia. He had just completed his Bachelor of Engineering Degree in Mechatronics Engineering and joined a private company in Malaysia as a Technical Trainee.