Information Technology Journal 5 (3): 476-484, 2006
ISSN 1812-5638

Writer Independent Online Handwritten Character Recognition Using a Simple Approach

Muhammad Faisal Zafar, Dzulkifli Mohamad and Razib M. Othman
Faculty of Computer Science and Information Systems,
UTM, 81310 Skudai, Johor, Malaysia

Abstract: This study describes the simple approach involved in online handwriting recognition. Conventionally, the data obtained needs a lot of preprocessing including filtering, smoothing, slant removing and size normalization before recognition process. Instead of doing such lengthy preprocessing, this study presents a simple approach to extract the useful character information. The whole process requires no preprocessing and size normalization. This research evaluates the use of the Back-propagation Neural Network (BPN) and presents feature extraction mechanism in full detail to work with on-line handwriting recognition. The obtained recognition rates were 51 to 83% using the BPN for different sets of character samples. This study also describes a performance study in which a recognition mechanism with multiple thresholds is evaluated for back-propagation architecture. The results indicate that the application of multiple thresholds has significant effect on recognition mechanism. This is a writer-independent system and the method is applicable for off-line character recognition as well. The technique is tested for upper-case English alphabets for a number of different styles from different subjects.

Key words: Character digitization, back-propagation neural networks, extreme coordinates, pattern recognition

INTRODUCTION

Handwriting recognition has always been a tough problem (Powalka, 1995). Recognition of handwritten characters by computer poses serious problems because of the high variability in the character shapes written by individuals (Sagar and Barman, 2001). As people tend to adjust their handwriting style to personal preferences, the resulting variability of handwriting styles often makes reading difficult even for humans. This problem becomes even more complicated when the writer is unknown (Powalka, 1995). Moreover, pairs of characters can be formed which are ambiguous, both for human and machine recognition, for instance U-V, C-G,Q-G, D-O, F-P.

Two classes of recognition systems are usually distinguished: online systems (Tappert *et al*., 1990; Anoop *et al*., 2004; Liu *et al*., 2004) for which handwriting data are captured during the writing process, which makes available the information on the ordering of the strokes and offline systems (Steinherz *et al*., 1999) for which recognition takes place on a static image captured once the writing process is over. Current focus of the market today is on-line handwriting recognition. With the increase in popularity of portable computing devices such as PDAs and handheld computers (Evan, 2005; PenWindows, 2005), non-keyboard based methods for data entry are receiving more attention in the research communities and commercial sector. Digitizing devices like (Smart Technologies, 2005) and computing platforms such as the IBM Thinkpad TransNote (2005) and Tablet PCs (2005), have a pen-based user interface. Such devices, which generate handwritten documents with online or dynamic (temporal) information, require efficient algorithms for processing and retrieving handwritten data (Anoop *et al*., 2004).

There is extensive work in the field of handwriting recognition and a number of reviews exist. General methodologies in pattern recognition and image analysis are presented in Mantas (Mantas, 1986). Character recognition is reviewed in (Suen *et al*., 1980; Govindan and Shivaprasad, 1990; Alessandro, 2002; Koerich *et al*., 2003; Bortolozzi *et al*., 2005) for off-line recognition and in (Nouboud and Plamondon, 1990; Plamondon and Srihari, 2000) for on-line recognition. Most of the researchers have chosen numeric characters for their experiments (Shridhar and Badreldin, 1986; Gader *et al*., 1991; Ahmed *et al*., 1995; Hebert *et al*., 1998; Iqbal and Zafar, 1998; Alexendre *et al*., 2002; Hiroto *et al*., 2005). So, some maturity can be observed for isolated digit recognition. However, when we talk

Corresponding Author: Muhammad Faisal Zafar, Faculty of Computer Science and Information Systems, UTM, 81310 Skudai, Johor, Malaysia Fax: +60 7 5565044

about the recognition of alphabetic characters, the problem becomes more complicated. The most obvious difference is the number of classes that can be up to 52, depending if uppercase (A-Z) and lowercase (a-z) characters are distinguished from each other. Consequently, there is a larger number of ambiguous alphabetic characters other than numerals. Character recognition is further complicated by other differences such as multiple patterns to represent a single character, cursive representation of letters and the number of disconnected and multi-stroke characters (Koerich, 2002). Few researches have addressed this complicated subject. In fact, it can be said that character recognition is still an open problem (Bortolozzi *et al*., 2005).

Neural Nets (NN) and Hidden Markov Models (HMM) are the popular, amongst the techniques which have been investigated for handwriting recognition. It has been observed that NNs in general obtained best results than HMMs, when a similar feature set is applied (Kapp *et al*., 2004). The most widely studied and used neural network is the Multi-Layer Perceptron (MLP) (Bishop, 1995). Such an architecture trained with back-propagation (LeCun *et al*., 1998a) is among the most popular and versatile forms of neural network classifiers and is also among the most frequently used traditional classifiers for handwriting recognition. See (Zhang, 2000) for a review. Other architectures include Convolutional Network (CN) (LeCun *et al*., 1998b), Self-Organized Maps (SOM) (Zhang, 1999), Radial Basis Function (RBF) (Bishop, 1995), Space Displacement Neural Network (SDNN) (Matan, 1992), Time Delay Neural Network (TDNN) (Lethelier, 1995), Quantum Neural Network (QNN) (Zhou, 1999) and Hopfield Neural Network (HNN) (Ling, 1997). The main advantages of neural networks lies in the ability to be trained automatically from examples, good performance with noisy data, possible parallel implementation and efficient tools for learning large databases.

However, the performance of any classifier depends on the way data is presented to it, i.e., the feature selection/extraction plays a vital role. Furthermore, feature extraction is again a crucial step for any recognition system. An excel combination of proper feature set and classification technique might be helpful in developing an efficient recognition system. In this paper, a simple approach has been proposed to extract the useful character information. The main objective of this work is the development of an efficient handwritten character recognition system. In this regard, a combination of uncomplicated proposed features and BPN has been implemented for the recognition of online upper case English alphabets. Although this study deals with a limited number of 26 upper case character classes, there is a space to extend this work for all alphanumeric characters including lower case characters. The performance BPN architecture has also been evaluated.

The recognition system applies a global decision module which decides either to accept the recognition result or reject it. In classification step, a pattern is considered ambiguous if it cannot be reliably assigned to a class, whereas a pattern assigned low confidence for all hypothesized classes can be treated as an outlier. Three different criteria (thresholds) of decision making have been applied.

This paper is organized as follows. Section 2 gives an overview of the proposed system. Section 3 describes the feature extraction steps from the handwritten character. Section 4 introduces BPN architecture. In Section 5, the experimental results are provided with some analyses and discussions. Section 6 presents the concluding remarks and suggestions for future work.

SYSTEM OVERVIEW

This section describes the simple technique involved in the proposed online handwriting recognition. This is a writer-independent system based on the neural net method. Conventionally, the data obtained needs a lot of preprocessing including filtering, smoothing, slant removing and size normalization before recognition process. Instead of doing such lengthy preprocessing, this paper presents a simple approach to extract the useful character information. A block diagram of the proposed online recognition system is shown in Fig. 1. The flow of data during training is shown by the dashed line arrows, while the data flow during recognition is shown by solid line arrows. The input to the system is a sequence of handwritten character patterns. After receiving input from tablet the extreme coordinates i.e., left, right, top and bottom are calculated. Then character is captured in a grid as shown in Fig. 3 and after sensing the character pixels in grid boxes, the character is digitized in a binary string. This binary string is applied at the input of BPN for training and recognition. Grid sizes of 14×8 (i.e., 14 rows and 8 columns) and 15×11 were used in the experiments.

Data acquisition: Tablet SummaSketch III has been used to take the samples from different subjects. Upper case alphabets characters have been used. Each subject was asked to write on tablet board (writing area). No restriction was imposed on the content or style of writing; the only exception was the stipulation on the

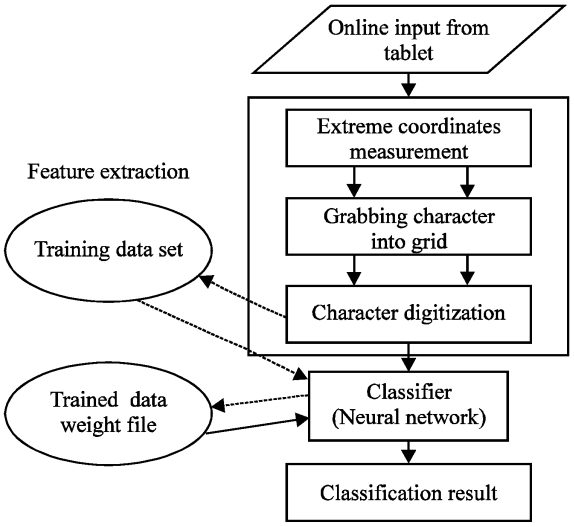

Fig. 1: Block diagram of the system

i.	Find the text boundary of the whole image by scanning from top to bottom for upper border Y1, left to right for left border X1, right to left for right borderX2 and bottom to top for lower borderY2.
ii.	Scan the binary image from Y1 towards Y2,
iii.	If there is a black pixel, then scan from X1 to X2 for that particular row to detect any white pixel.
iv.	If no white pixel found, there is a row gap.
v.	Repeat steps ii - iii for the whole image to find total number of row gaps.

Fig. 2: Sequential algorithm for finding the number of rows

isolation of characters. The writers consisted of university students (from different countries), professors and employees in private companies. The simulation of each written character could be seen on computer screen as white digital ink with black background. Thus one can make use of black and white colours for some useful processing of written characters.

Character detection: As stated earlier that character is written with white digital ink, so the algorithm for character detection is quite simple. It searches from left to right for white pixels starting from left-top corner of the area specified for writing. A trace of a white pixel is the indication of the presence of a character.

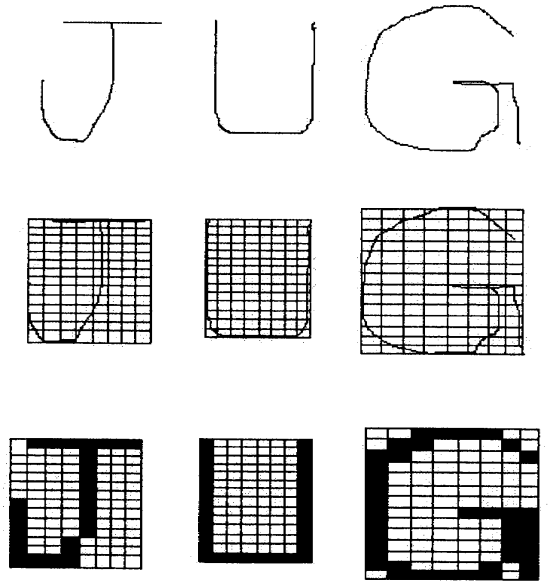

Fig. 3: Steps in feature extraction

Calculating the number of rows: Going from top to bottom, the algorithm searches for the presence and absence of white pixels. The continuous absence of white pixels (or presence of black pixels) could be a gap between two rows. To make sure whether it is a gap, algorithm searches from left to right against every black pixel, if there is no trace of white pixel for the entire row, the gap is confirmed. In this way, all the horizontal gaps, in the image, are traced out and from this, number of rows are calculated. Figure 2 shows a sequential algorithm for finding the number of rows in the document.

Character boundary calculation: After detecting the character, the next step is to calculate its boundary. The algorithm checks from left to right and top to bottom for left, top, right and bottom boundaries of the character. While going from left to right, the first white pixel is the left boundary and last white pixel is the right boundary of the character. Similarly from top to bottom, first white pixel is the top boundary and last white pixel is the bottom boundary of the character. If there is a vertical gap between two portions of the same character, e.g., 'H' then the algorithm also checks from top to bottom for that particular area. The presence of white pixel will eliminate the doubt of a true gap. Similar checks are employed for horizontal gaps between two portions of the same character like 'S' and 'F'. In this way boundaries of the characters in a row are calculated and stored in a data base. After calculating the total number of characters in a row, the individual width and height of each character is measured.

FEATURE EXTRACTION

In the proposed online handwriting recognition system, feature extraction consists of three steps: extreme coordinates measurement, grabbing character into grid and character digitization. The handwritten character is captured by its extreme coordinates from left /right and top/bottom and is subdivided into a rectangular grid of specific rows and columns. The algorithm automatically adjusts the size of grid and its constituents according to the dimensions of the character. Then it searches the presence of character pixels in every box of the grid. The boxes found with character pixels are considered "on" and the rest are marked "off". A binary string of each character is formed locating the "on" and "off" boxes (named as character digitization) and presented to the neural network input for training and recognition purposes. The total number of grid boxes represented the number of binary inputs. A 14×8 grid thus resulted in 112 inputs to the recognition model. An equivalent statement would be that a 14×8 grid provided a 112 dimensional input feature vector. The developed software contains a display of this phenomenon by filling up the intersected squares. The effect has been produced in Fig. 3.

BACK-PROPAGATION NEURAL NETWORK

Extensive material has appeared on BPN and its applications (Kotanzad and Lu, 1988; Ahmed *et al*., 1995; Ho and Yang, 1994; Zafar and Dzulkifli, 2005; Ferand, 2001) since the publication of the work done by Rumelhart *et al.* (1986). Neural network classifiers exhibit powerful discriminative properties and they have been used in handwriting recognition particularly with digits, isolated characters and words in small vocabularies (Alessandro, 2002). Fig. 4 shows a BPN network with an input layer, one hidden layer and an output layer. In such an architecture, information is processed as follows: the outputs from the Processing Element (PEs) of the input layer, after multiplying with the corresponding interconnecting weights, serve as inputs to the PEs of the hidden layer. The output from the PEs of the hidden layer, after multiplication with corresponding interconnecting weights, serve as input to the PEs of the output layer. A bias processing element supplies a constant output of +1 to all the PEs of the hidden and the output layer. BPN models with more than one hidden layer process information on the same principle.

For a PE, the output is typically a function of the sum of input into it. For BPN models, PEs with linear,

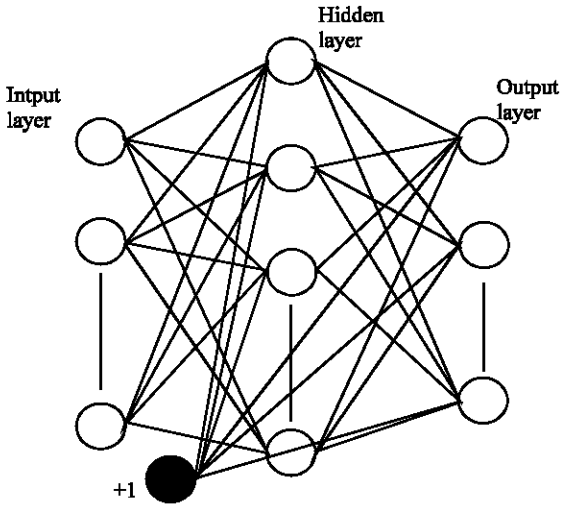

Fig. 4: A typical BPN with one hidden layer

sigmoid and hyperbolic tangent transfer function are typically used depending upon the position of the PE in the network architecture and the nature of the problem under consideration. With sigmoid PEs in the hidden and the output layer, the output of a PE in the network's output layers becomes highly non-linear function of the input to the network. For only one PE in the output layer, the network can be thought of as representing a non-linear function of the inputs. For n PEs in the input layer and m in the output layer, the network represents m non-linear functions of n variables.

One has to digress here and discuss the classical regression techniques for fitting functions on input-output data (in the standard statistics terminology, input and output data would correspond to the independent or 'controlled' variable data and dependent variable data, respectively). Least square regression techniques involve development of model on the basis of available data by first appropriately choosing a function type and then evaluating function parameters which would minimize the Sum of the Squared Error (SSE) for the available data. Thus the mean square error function is defined as:

$$E = \frac{1}{2}\sum_{k}\left(t_{kj} - o_{kj}\right)^2 \quad (1)$$

where

t_{kj} is the target output and o_{kj} is the network output

For a fix data set, the SSE becomes a function of parameters; for k parameters, one gets a k+1 dimensional SSE hyper surface. If the chosen function is linear in parameters, then the convexity of the SSE surface reduces

the problem of parameter evolution to solving a system of linear equation. For functions which are non-linear in the parameter, the search for the SSE minimum becomes iterative. Gradient descent method is one such iterative technique; at a particular iteration, parameters are updated in the direction of steepest descent on the SSE surface.

The BPN architectures actually represent specialized non-linear functions used as regression models (the exact function may vary with variation in the network architecture and the types of the non-linear PEs used). The interconnecting weights correspond to parameters in the regression models. The weights are adjusted during training using the following formula :

$$w_{kj}(t+1) = w_{kj}(t) + \frac{\alpha_k \partial E(w)}{\partial w_{kj}} + \beta \Delta w_{kj}(t) \qquad (2)$$

where

$w_{kj}(t + 1)$ is the updating weight from output PEk to hidden PE j at time t,

$w_{kj}(t)$ is the weight from output PE k to hidden PEj at time t,

$\partial E(w)/\partial w_{kj} = \sum \delta_k o_k$,

$\delta_k = f'(net_k) (t_k - o_k)$ is the error signal for the output PE,

$o_k = f'(net_k) \sum w_{kj} \delta_{kj}$ is the output for the output layer,

$\Delta w_{kj}(t) = w_{kj}(t) - w_{kj}(t - 1)$, change of weights at time t,

η is the momentum factor,

α_k, (k = 1, 2, ..., n) is the learning rate and depend on

$$\left| \frac{\partial E(w)}{\partial w_{kj}} \right|$$

Training in the BPN models is actually the iterative search for the global minimum. A strict adherence to the principle of weight changes in the direction of steepest descent translates into one instance of network weight updates after presenting the entire input-output pairs to the network one after other (i.e, making calculations for every such presentation and updating the network rates after all such presentation have been made for the entire data set). To speed up conversion, the Generalized Delta Rule (GDR) (Freeman and Skapura, 1991) is typically used which, as an approximation to the strict steepest descent algorithm, suggest weight updates at every presentation of an input-output data pair during training.

RESULTS

Data set and model parameters: The data used in this work was collected using tablet SummaSketch III. It has an electric pen with sensing writing board. An interface was developed to get the data from tablet. Anoop *et al*, (2004) pointed out that the actual device for data collection is not important as long as it can generate a temporal sequence of x and y positions of the pen tip. However, the writing styles of people may vary considerably on different writing surfaces and the script classifier may require training on different surfaces.

Upper case English alphabets were considered as case study. Selecting only a few characters from the entire character set to analyze the behaviour of different recognition models appeared appropriate (Ahmed *et al*., 1995). In the data set, the total number of handwritten characters is about 2000 characters, collected from 40 subjects. Experiments were examined with grid size of 14×8 and 15×11. Every developed model was tested on characters drawn by individuals who did not participate in the sample collection for data set. Each subject was asked to write on tablet board (writing area). No restriction was imposed on the content or style of writing; the only exception was the stipulation on the shape of 'I'. The grid based character digitization proved improper for characters with negligible width. The shape for handwritten 'I' was thus standardized with horizontal lines at the top and the base. The writers consisted of university students (from different countries), professors and employees in private companies.

Learning/Training: In the BPN, sigmoid PEs were used in the hidden and the output layer. Twenty-six output layer processing elements (PEs) corresponded to Twenty-six English alphabets to be recognized. For example, a 'high' output value on the second PE in the output layer and 'low' on the others would mean that the network classifies the input as a 'B'. As another example, a network output vector of [0.00 0.01 0.16 0.02 0.09 0.96 0.13 0.00-0.15 0.04] would be translated as the model classifying the input character as 'F'.

The GDR was preferred over the exact steepest descent algorithm because of the large difference in convergence time between the two. The authors experimented with the number of hidden layer PEs in search of better convergence behaviour. The selection of *learning rate* and *momentum coefficient* (proportionality constant for sliding in the direction of negative gradient) was also more of an art than a science; their values were kept between 0 and 1 with typically a gradual decrease in magnitude as training progressed.

Experiments were started with a strict convergence criterion: training was stopped only when the network classified all the training samples correctly. While

Table 1: Details of Different parameter values used for BPN during training phase

	165-dimentional input			112-dimentional input		
Sample/Char.	5	11	22	5	11	22
Iterations	13520	8120840	2882600	8680	15316	51000
SSE	0.171610	1.5	12.5	0.145210	0.166071	2.515676
Learning rate	0.999	0.99	1.09	0.9	0.9	2.0
Momentum Parameter	0.5	0.5	0.5	0.5	0.5	0.3
Hidden Elements	35	35	40	30	30	15
untrained Characters	0	3	25	0	1	5

Remaining experiments were carried out using 14×8 grid size. Table 2 presents the detail of different parametric values for BPN during training phase

Table 2: Details of different parameter values used For BPN during training phase

Samples / Character	Total no. of characters	Iterations	Sum of Squared Error (SSE)	Learning rate	Momentum parameter	Hidden elements	# of untrained characters
5 Each	130	8680	0.145210	0.9	0.5	30	0
11 Each	286	15316	0.166071	0.9	0.5	30	1 (0.3%)
22 Each	572	51000	2.515676	2.0	0.3	15	5 (0.8%)
33 Each	858	7129980	58.005	0.999	0.5	25	116 (13%)
44 Each	1144	880880	147.54	1.5	0.9	30	295 (26%)
55 Each	1430	1801800	214.54	2.0	0.5	35	429 (30%)
66 Each	1716	3517800	172.54	0.9999	0.5555	40	345 (20%)

Table 3: Performance Of BPN models with three different criteria of classification

	'Threshold': NONE			'Threshold': 0.5			'Threshold': 0.9		
Samples/Character	CRs (%)	FRs (%)	RFs (%)	CRs (%)	FRs (%)	RFs (%)	CRs (%)	FRs (%)	RFs (%)
5 Each	70	30	0	65	10	25	60	10	30
11 Each	73	27	0	71	5	24	65	5	30
22 Each	75	25	0	81	4	15	83	6	11
33 Each	77	23	0	71	7	22	67	2	31
44 Each	65	35	0	60	4	36	51	6	43
55 Each	71	29	0	62	4	34	53	4	43
66 Each	83	17	0	79	6	15	79	2	19

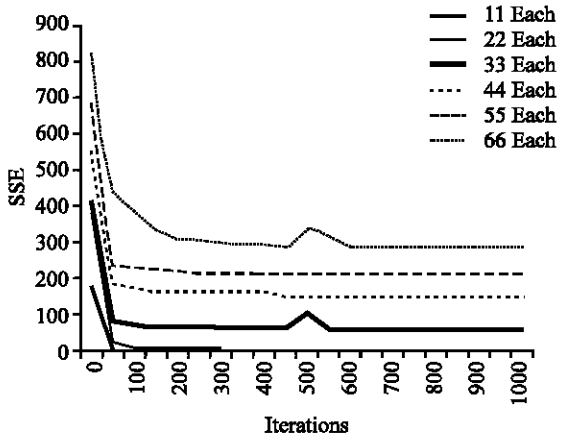

Fig. 5: Convergence trends of different BPN models

checking the network's performance during training, an output layer PE's output value of ≥0.9 was translated as 'high'. Thus, for this criterion, an output vector [0.00 0.03 0.06 0.12 0.09 0.16 0.03 0.91 0.17- 0.15 0.14] for character 'H' in the training sample set would be termed as proper classification; a 0.85 instead of 0.91 in the output vector would render the training input as not properly recognizable till that stage in the training process.

Initially experiments were started with relatively small data sets i.e., 5 samples/character, 11 samples/character and 22 samples/character. Table 1 shows the summary of different parameter's values used for BPN during training. Training was stopped for the 15x11 grid, with 3 samples (out of 286) and 25 samples (out of 572) remained unclassified after 8120840 and 2882600 training presentations for 11 samples/character model and 22 samples/character model, respectively (Table 1). Similarly, one sample and 5 samples remained unclassified after 15316 and 51000 training presentations for 11 samples/character model and 22 samples/character model, respectively for the grid 14x8 (Table 2).

Remaining experiments were carried out using 14×8 grid size. Table 2 presents the detial of different parametric values for BPN during training phasee

The performance trends for all data sets, presented in Table 2, have been shown graphically in Fig. 5.

Recognition Performance: Seven different data sets: 5 samples/character, 11 samples/character, 22 samples/character 33 samples/character, 44 samples/character, 55 samples/character and 66 samples/character were being experimented to evaluate the performance of both models with gradually

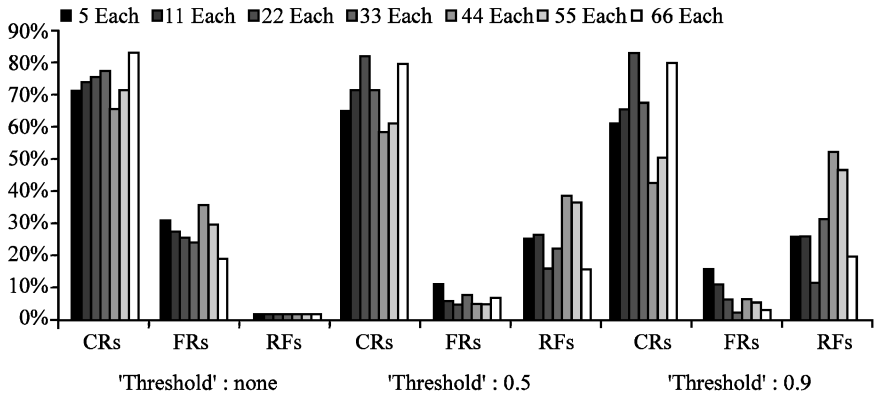

Fig. 6: Graphical presentation of BPN performances with three different criteria of Recognition

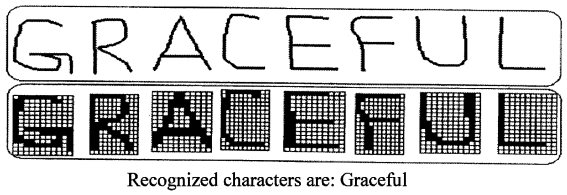

Fig. 7: Steps in character recognition

increasing the number of samples/character. As mentioned earlier, models were evaluated on samples taken from individuals who did not participate in the initial process of setting up the training data set. This was done keeping in view the eventual aim of using the models in practical online recognition systems.

For the developed BPN model, the debate on a valid high PE output in the output layer was resolved by evaluating the performance for different decision making criteria. Model was tested with high thresholds of 0.9 and 0.5, using the PE with the highest value above the threshold for input classification. Another criterion used in translating the BPN model's outputs was to eliminate the concept of threshold and simply use the highest value. Note that the first criteria will always have the possibility of a recognition failure: a network decision of not attributing any character to the input data. The last criteria will eliminate this somewhat desirable feature in the decision making process. Table 3 presents the statistics. CRs, FRs and RFs are abbreviation for Correct Recognitions, False Recognitions and Recognition Failures, respectively.

Performance analysis: Initial experiments showed that 112-dimentional input vector proved much better than 165-dimentional input vector. This shows that after certain input size limit, no improvement can be achieved in recognition. Also larger size input took lot of time for training and some samples remained yet to train (Table 1). On the other hand, in case of 112-dimentional input vector the training time was much less and untrained samples were few. This factor convinced the authors to carry on further experiments with grid size of 14×8.

For developed BPN models, it was observed that learning became more difficult and, even after long time of training, models were unable to fully learn the training sets. The gradual increase in the percentage of untrained samples can be seen in Table 2. An increase of hidden PEs was also helpful towards the convergence when sample/character were increased. Sometimes, initially a big value of learning rate showed a rapid learning even with less number of hidden PEs but most of the time a value <1 appeared suitable for learning rate. Generally, the recognition performance of BPN models improved with increase in samples/character.

It is important to note that the sigmoid functions in the output layer behave as 'smoothed' bipolar switches; the inputs to these bipolar switches are values of the decision functions (Dzulkifli and Zafar, 2004). These decision functions or decision surfaces have positive value for a PE's output greater than 0.5 and negative values for PE outputs of less than 0.5. The evaluation of weights during training can be thought of as development of such decision functions. Poor performance of a trained neural network may imply improper decision functions which are good enough for the training samples but not appropriate for other inputs. Figure 6 presents a graphical overview of BPN performances with three different decision criteria of Recognition. The recognition rate without any threshold (NONE) was highest (up to 83%) but at the cost of more false recognition. This recognition rate gradually decreases by applying tough thresholds (0.5 and 0.9) but this makes the system more reliable by tempting less false recognitions. However, overall the

false recognitions were much less than Recognition Failure (RFs), after applying thresholds, which is a plus point. More RFs are due to a large number of untrained samples. This number can be reduced by experimenting more suitable combinations of hidden PEs and learning rates. It will ultimately improve the recognition rate. Figure 6 shows a graphical overview of BPN performances presented in Table 3.

In general, the performance of all developed BPN models was observed up to the mark. Figure 7 shows an example of recognized handwritten text.

CONCLUSIONS

An elementary online handwriting recognition prototype for isolated upper case English characters has been developed using a very simple approach without an application of preprocessing process. The system is writer-independent based on neural network approach. For training and recognition purposes BPN has been used. The preliminary results are quite encouraging. The recognition rates achieved are still worth considering and highly desirable in pattern recognition. The experiments provided the authors an opportunity to explore the pattern recognition methodology; the exercise provided a theoretical base for further investigations and impetus for development work in this discipline. The obtained results motivate the continuity of the system development considering a preprocessing mechanism including normalization and slant removal. Other future work might involve some new feature extraction approaches.

REFRENCES

Alessandro, V., 2002. A survey on off-line cursive word recognition. Pattern Recognition, 35: 1433.

Ahmed, S.M., M. Ahmad, M. Asif, A. Majid and M.Z. Faisal, 1995. Experiments in character recognition to develop tools for an optical character recognition system. IEEE Inc. 1st National Multi Topic Conf. proc. NUST, Rawalpindi, Pakistan, pp: 61-67.

Lemieux, A., G. Christian and P. Marc, 2002. Genetical Engineering of Handwriting Representations. Proc. of the International Workshop on Frontiers in Handwriting Recognition (IWFHR), Niagara-On-The-Lake, August 6-8.

Anoop, M., A. Namboodiri and K. Jain, 2004. Online handwritten script recognition. IEEE Trans. PAMI. 26: 124-130.

Bishop, M., 1995. Neural Networks for Pattern Recognition. Oxford Univ. Press, Oxford-UK.

Bortolozzi, F., A. Britto Jr, L.S. Oliveira and M. Morita, 2005. Recent Advances in Handwriting Recognition. In: Umapada, P. *et al.* (Eds.). Document Analysis, pp: 1-31.

Dzulkifli, M. and M.F. Zafar, 2004a. Recognition of complex patterns using a novel feature vector by backpropagation neural-network. Pattern Recognition and Image Analysis, 14: 479-487.

Koblentz, E., 2005. The Evolution of the PDA: 1975-1995. Computer Collector Newsletter, Version 0.993.

Ferand, R., O.J. Bernier, J.E. Viallet and M. Collobert, 2001. A fast and accurate face detection based on neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23: 1.

Freeman, J.A. and D.M. Skapura, 1991. Neural Networks: Algrithms, Applications and Programming Tecniques, Addison-Wesley Publishing Company.

Gader P.D., B. Forester, M. Ganzberger, A. Billies, B. Mitchell, M. Whalen and T. Youcum, 1991. Recognition of handwritten digits using template and model matching. Pattern Recognition, 5: 421-431.

Govindan, V.K. and A.P. Shivaprasad, 1990. Character recognition: A review. Pattern Recognition, 23: 671-683.

Jean-Francois, H., P. Marc and N. Ghazali, 1998. A new fuzzy geometric representation for on-line isolated character recognition. Proc. of the 14th International Conference on Pattern Recognition, Brisbane (Australia), pp: 1121-1123.

Mitoma, H., S. Uchida and H. Sakoe, 2005. Online character recognition based on elastic matching and quadratic discrimination. Proceedings of 8th International Conference on Document Analysis and Recognition (ICDAR 2005, Seoul, Korea), 1: 36-40.

Kim, H.J. and H.S. Yang, 1994. A Neural network capable of learning and inference for visual pattern recognition. Pattern Recognition, 27: 1291-1302.

IBM ThinkPad TransNote, 2005. http://www-132.ibm.com/content/search/transnote.html .

Iqbal, A., Zafar and M. Faisal, 1998. PC based optical objective test marking system using neural networks. International Workshop on Recent Advances in Computer Vision Proc. SZABIST, Karachi, Pakistan,: 41-50.

Kapp, M.N., C. Freitas and R. Sabourin, 2004. Handwritten Brazilian month recognition: An analysis of two NN architectures and a rejection Mechanism. 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR-9), Tokyo, Japan,

Koerich, L., 2002. Large vocabulary off-line handwritten word recognition. Ph. D thesis, école de technologie supérieure, montreal-canada.

Koerich, A.L., R. Sabourin and C.Y. Suen, 2003. Large vocabulary off-line handwriting recognition: A survey. Pattern Anal. Applic., 6: 97-121.

Kotanzad, A. and J.H. Lu, 1988. Distortion Invariant Character Recognition by a Multi-layer Perception and Back-propagation learning. In Proc. IEEE 2nd Intl. Conf. Neural Networks, 1: 625-631.

LeCun Y., L. Bottou, G.B. Orr and K.R. Muller, 1998a. Eficient Backprop. In Orr, G. and K. Miller (Eds.). Neural Networks: Tricks of the Trade. Springer.

LeCun Y., L. Bottou, Y. Bengio and P. Haffner, 1998b. Gradient-based learning applied to document Recognition. Proc. IEEE, 86: 2278-2324.

Lethelier, L.M. and M. Gilloux, 1995. An automatic reading system for handwritten numeral amounts on french checks. In Proc. 3rd International Conference on Document Analysis and Recognition, Montreal-Canada, August, pp: 92-97.

Ling, M. N.G. Lizaraga and A. Koerich, 1997. A Prototype for Brazilian Bankcheck Recognition. in Impedovo, S. *et al.* (Eds.). Intl. J. Pattern Recognition and Artificial Intelligence, World Scientific, pp: 549-569.

Liu Cheng-Lin, J. Stefan and M. Nakagawa, 2004. Online recognition of chinese characters: The state-of-the-art. IEEE Trans. on Pattern Analysis and Machine Intelligence, 26: 198-203.

Matan, O., J.C. Burges, Y. LeCun, J.S. Denker, 1992. Multi-digit Recognition Using a Space Displacement Neural Network. In Moody, J. E., S.J. Hanson and R.L. Lippmann (Eds.). Advances in Neural Information Processing Systems, volume 4, Morgan Kaufmann, pp: 488-495.

Nouboud, F. and Plamondon, 1990. On-line recognition of handprinted characters: Survey and beta tests, Pattern Recognition, 23: 1031-1044.

Pen Computing Magazine: PenWindows, 2005. http://www.pencomputing.com/PenWindows/index.html,

Rejean, P. and S.N. Srihari, 2000. On-line and off-line handwriting recognition: A comprehensive survey. 1EEE Transactions on PAMI, 22: 63-84.

Powalka, R.K., 1995 "n algorithm toolbox for on-line cursive script recognition. Ph. D Thesis The Nottingham Trent University.

Rumelhart, D.E., G.E. Hinton and R.J. Williams, 1986. Learning Internal Representations by Error Propagation. In: Rumelhart, D.E. and J.J. McClelland (Eds.). Parallel Distributed Processing: Explorations in the Microstructure of Cognition, MIT Press, Cambrige, 1: 318-362

Shridhar, M. and A. Badreldin, 1986. Recognition of isolated and simply connected handwritten numerals. Pattern Recognition, 19: 1-12.

Sagar, G. and S. Barman, 2001. Isolated handwritten character recognition. Proc. of Intelligent Processing and Manufacturing of Materials IPMM-2001, 3rd International Conference July 29 August 3, 2001Vancouver, British Columbia, Canada.

Smart Technologies Inc. Homepage, 2005. http://www.smarttech.com/

Suen, C.Y., M. Berthod and S. Mori, 1980, Automatic recognition of hand printed character-the state of the art. Proceedings of IEEE, 68: 469-487.

Steinherz, T., E. Rivlin and N. Intrator, 1999. Offline cursive script word recognition-a survey. Intl. J. Document Analysis and Recognition, 2: 90-110.

Tappert, C.C., C.Y. Suen and T. Wakahara, 1990. The state of the art in on-line handwriting recognition. IEEE Trans. on Pattern Analysis and Machine Intelligence, 12: 787-808.

Windows XP Tablet PC Edition Homepage, 2005. http://www.microsoft.com/windowsxp/tabletpc/default.asp

Zafar, M.F. and M. Dzulkifli, 2005. Comparison of two different proposed feature vectors for classification of complex image. J. Teknologi, Universiti Teknologi Malaysia, 42: 65-82.

Zhang, G.P., 2000. Neural networks for classification: A survey. IEEE Transactions on Systems, Man and Cybernetics-Part C: Applic. Rev., 30: 451-462.

Zhang, F.M., H. Yan and M.A. Fabri, 1999. Handwritten digit recognition by adaptativesubspace self organizing map. IEEE Trans. on Neural Networks, 10: 939-945.

Zhou, J., 1999. Recognition and verification of unconstrained handwritten numeral. Ph. D Thesis, Concordia University, Montreal-Canada.