

A Modified Direction Feature for Cursive Character Recognition

M. Blumenstein and X. Y. Liu
School of Information Technology
Griffith University-Gold Coast Campus
PMB 50, Gold Coast Mail Centre,
QLD 9726, Australia
E-mail: m.blumenstein@griffith.edu.au

B. Verma
School of Computing and Information Systems
Central Queensland University
Bruce Highway
North Rockhampton QLD 4702, Australia
b.verma@cqu.edu.au

Abstract- This paper describes a neural network-based technique for cursive character recognition applicable to segmentation-based word recognition systems. The proposed research builds on a novel feature extraction technique that extracts direction information from the structure of character contours. This principal is extended so that the direction information is integrated with a technique for detecting transitions between background and foreground pixels in the character image. The proposed technique is compared with the standard direction feature extraction technique, providing promising results using segmented characters from the CEDAR benchmark database.

I. INTRODUCTION

The literature details many high accuracy recognition systems for separated handwritten numerals and characters [1]. However, research into the recognition of characters extracted from cursive and touching handwriting has not had the same measure of success [2]-[4]. One of the main problems faced when dealing with segmented, handwritten character recognition is the ambiguity and illegibility of the characters.

Traditionally, cursive character recognition techniques have been used in conjunction with dynamic-programming matching-based approaches for handwriting recognition [3]. Yamada and Nakano [2] investigated a standard technique for feature extraction based on direction histograms in character images. They used segmented characters from words in the CEDAR database [5]. Kimura *et al.* [4] investigated a similar feature extraction technique calculating local histograms based on chain code information in segmented handwritten characters (also from the CEDAR dataset). Gader *et al.* [3] have proposed a feature extraction technique utilising transition information from the background to the foreground pixels in the vertical and horizontal directions of a character image. Singh and Hewitt [6] employed the modified Hough Transform on characters from the CEDAR data set. A recent study by Camastra and Vinciarelli [7] has proposed feature extraction techniques generating local and global features. The local features are obtained from sub-images of the character including

foreground pixel density information and directional information. The global features used included the fraction of the character appearing below the word baseline and the character's width/height ratio.

In the proposed research, two feature extraction techniques were investigated for cursive character recognition. The first is the Direction Feature (DF) technique and the second is the proposed Modified Direction Feature (MDF). The success of each feature extraction technique was tested using Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF) classifiers.

The remainder of this paper is broken down into five sections. Section II discusses the cursive character processing techniques used in this research, Section III provides details on the feature extraction techniques and the character recognition methodology, experimental results are presented in Section IV, a discussion of the results takes place in Section V, and finally Section VI presents future research and conclusions.

II. CURSIVE CHARACTER PROCESSING

A. Character Extraction

The character sets used for training and testing were extracted from words in the training and test directories (CITIES/BD) of the CEDAR CD-ROM [5]. This is referred to as the CEDAR Automatically Segmented (CAS) dataset.

To summarise the character extraction process, our technique first proceeded to sequentially locate all non-cursive/printed character components through the use of character component analysis. Finally, x -coordinates (vertical segmentations) for each connected character component (defined by our heuristic segmenter [8]) were used to define the vertical boundaries of each character matrix. To locate the horizontal boundaries (top and bottom of the character matrix), the area bounded vertically (via x -coordinates or the boundaries found as a result of connected component analysis), is examined from the top and bottom. The first instances of foreground pixels located by searching from the top or bottom are deemed as the top-most and

bottom-most y-coordinates for the character matrix respectively.

B. Preprocessing

During initial processing, word images were converted from the standard CEDAR format to a .pbm format. Each word was then thresholded and slant corrected [9]. It was deemed necessary to further preprocess the individual extracted characters. Due to the nature of the classifiers used (neural networks), we were required to produce input vectors of a manageable and uniform size. This was achieved through re-scaling the original image and performing local averaging on the feature vector. To facilitate the detection of direction information for both feature extraction techniques, it was necessary to perform boundary extraction on each character image [10].

III. FEATURE EXTRACTION AND CHARACTER RECOGNITION

The sections below describe two feature extraction techniques that were investigated in this research, the second of which is proposed here for the first time. The first is the standard Direction Feature (DF) [11], which was developed to simply describe the boundary of each character's image. The second feature extraction technique builds on this direction information and integrates it with transition features.

A. Direction Feature

The first technique (DF) sought to simplify each character's boundary through identification of individual stroke or line segments in the image. Next, in order to provide a normalized input vector to the neural network classification schemes, the new character representation was broken down into a number of windows of equal size (zoning) whereby the number, length and types of lines present in each window was determined.

1) *Determining Directions*: The line segments that would be determined in each character image were categorised into four types: 1) Vertical lines, 2) Horizontal lines, 3) Right diagonal and 4) Left diagonal. Aside from these four line representations, the technique also located intersection points between each type of line.

To facilitate the extraction of direction features, the following steps were required to prepare the character pattern [11]:

1. Starting point and intersection point location
2. Distinguish individual line segments
3. Labelling line segment information
4. Line type normalization

Following the steps described above, individual strokes in the character images are characterised by one of four numerical direction values (2, 3, 4 or 5). This process is illustrated in Figure 1 below.

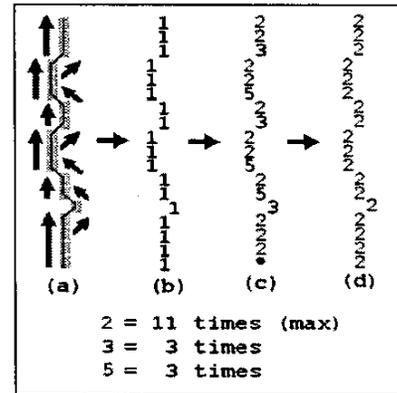


Figure 1: (a) Original line, (b) Line in binary file, (c) After distinguishing directions, (d) After direction normalization

2) *Formation of Feature Vectors*: Once line segments were determined, a methodology was developed for creating appropriate feature vectors. In the first step, the character pattern marked with direction information was zoned into windows of equal size (the window sizes were varied during experimentation). In the next step, direction information was extracted from each individual window. Specific information such as the line segment direction, length, intersection points, etc. were expressed as floating point values between -1 and 1 [11]. Figure 2 illustrates the process of input vector creation.

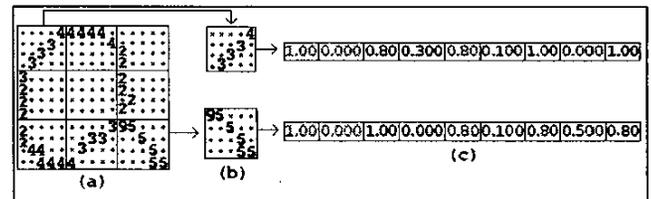


Figure 2: (a) Processed image, (b) Zoned windows, (c) Input vector components

B. Modified Direction Feature

The proposed MDF technique builds upon the DF technique described in Section A. The main difference is in the way the feature vector is created. For MDF, feature vector creation is based on the calculation of transition features from background to foreground pixels in the vertical and horizontal directions. A number of researchers have proposed feature extraction techniques based on transition information, an example may be found here [3]. In MDF, aside from calculating Location Transitions (LTs), the

direction value at that location is also stored (Direction Transitions - DTs).

1) *Determining LT Values:* To calculate LT values, it is necessary to scan each row in the image from left-to-right and right-to-left. Likewise, each column in the image must be scanned from top-to-bottom and bottom-to-top. The LT values in each direction are computed as a fraction of the distance traversed across the image [3]. Therefore, as an example, if the transitions were being computed from left-to-right, a transition found close to the left would be assigned a high value compared to a transition computed further to the right (See Figure 3). A maximum value (MAX) was defined to be the largest number of transitions that may be recorded in each direction. Conversely, if there were less than MAX transitions recorded (n for example), then the remaining $MAX - n$ transitions would be assigned values of 0 (to aid in the formation of uniform vectors).

2) *Determining DT Values:* Once a transition in a particular direction is found, along with storing an LT value, the direction value at that position is also stored (DT). The DT value is calculated by dividing the direction value by a predetermined number, in this case: 10. The value 10 was selected to facilitate the calculation of a decimal value between 0 and 1 (See Figure 3).

Therefore, following the completion of the above, four vectors would be present for each set of feature values (eight vectors in total). For both LT and DT values, two vectors would have dimensions $MAX \times NC$ (where NC represents the Number of Columns/width of the character) and the remaining two would be $MAX \times NR$ (where NR represents the Number of Rows/height of the character).

A further re-sampling of the above vectors was necessary to ensure that the NC/NR dimensions were normalised in size. This was achieved through local averaging. The target size upon re-scaling was set to a value of 5. Therefore, for a particular LT or DT value vector, windows of appropriate dimensions were calculated by determining an appropriate divisor of NC/NR, and the average of the LT/DT values contained in each window were stored in a re-sampled 5×5 matrix (as shown in Figure 4 for vectors obtained from a left-to-right direction traversal). This was repeated for each of the remaining transition value vectors so that a final 120 or 160 element feature vector could be formed using the following formula:

$$nrFeatures \times nrTransitions \times nrVectors \times resampledMatrixHeight(Width)$$

(1)

where:

- $nrFeatures = 2,$
- $nrTransitions = 3$ or $4,$
- $nrVectors = 4$ and
- $resampledMatrixHeight(width) = 5$

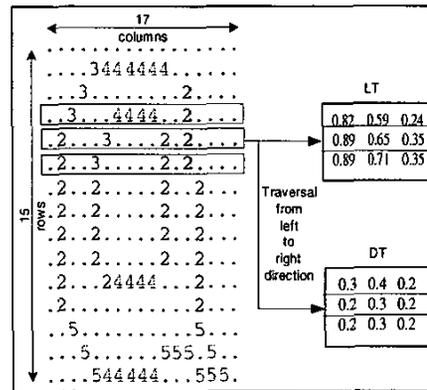
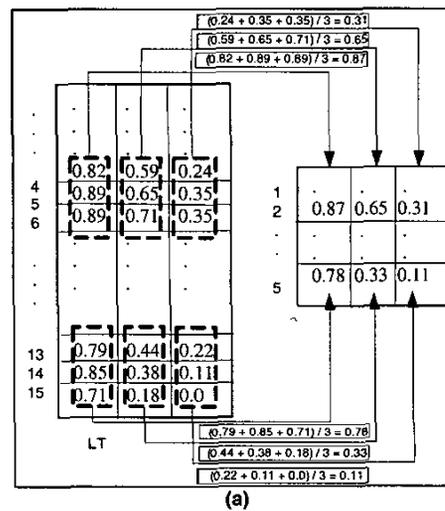
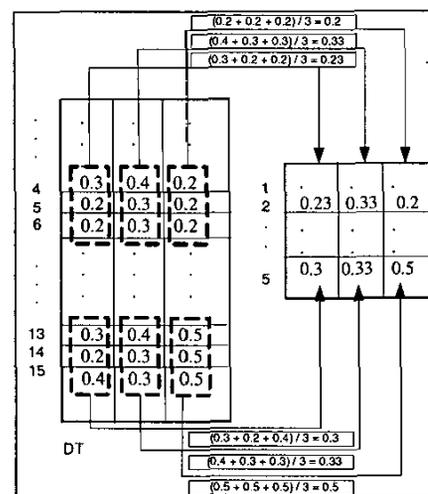


Figure 3: Processing of DT and LT values in the left-to-right direction



(a)



(b)

Figure 4: (a) Calculation and creation of a resampled LT value vector (left-to-right direction), (b) Calculation and creation of a resampled DT value vector (left-to-right direction)

C. Configuration of the Neural Classifiers

In the sections above, two feature extraction techniques were detailed for the purpose of providing meaningful structural character features that would be useful as inputs to a classifier. The classifiers chosen for this task was a feed-forward MLP trained with the backpropagation algorithm and an RBF network. For experimentation purposes, the architectures were modified varying the number of inputs, outputs, hidden units (or centres) and hidden layers.

The number of inputs to each network was associated with the size of the feature vector for each image. Various vector dimensions were investigated for experimentation. The most successful vector configurations were of size 81 for the DF and 120/160 for the MDF.

For each classifier type, two neural networks were trained with 27 outputs each. Therefore, one neural network was trained with upper case characters (A-Z) and the other with lower case characters (a-z). The 27th output in each network was a "reject" neuron to deal with sub-characters and multiple-character components.

D. Preparation of Training Data for the Neural Classifiers

For neural network training it was necessary to include samples for each type of character (a-z, A-Z). The training/test files needed to be manually prepared, however character matrix boundaries were determined based on the output of our heuristic segmenter. Each extracted character was viewed by a human operator and was labelled manually as belonging to a particular character class. For the reject classes, the human operator was instructed to subjectively decide what constituted a 1/4 character, half a character and a multiple character component.

IV. EXPERIMENTAL RESULTS

For experimentation of the feature extraction techniques detailed in Section III, we used handwritten words from the CEDAR benchmark database [5]. In particular we used word samples contained in the "BD/cities" directory of the CD-ROM. Characters were obtained as per the extraction technique described in Section II.

The results in this research are displayed in tabular form for each set of experiments. Table 1 presents top results for the DF and MDF extraction techniques using the MLP whilst Table 2 presents results employing the RBF network. For the comparison, both feature extraction techniques were tested on boundary representations of resized characters. Separate experiments were conducted for lower case and upper case character patterns. A total of 18655 lower case and 7175 upper case character patterns were generated for training. A

further 2240 lower case and 939 upper case patterns were used for testing.

In the case of the MLP networks, many experiments were performed varying settings such as the number of iterations, the number of hidden units, learning rate (η) and momentum (α). For the RBF network, the number of centres was adjusted to provide the best results. The tables below show the top results in each case.

TABLE 1. CHARACTER RECOGNITION RATES WITH AN MLP NETWORK TRAINED USING BOUNDARY INFORMATION FROM RESIZED CHARACTERS

	Test Set Recognition Rate [%]		
	DF	MDF (120)	MDF (160)
Lowercase	69.73	70.22	70.17
Uppercase	77.32	80.83	80.40

TABLE 2. CHARACTER RECOGNITION RATES WITH AN RBF NETWORK TRAINED USING BOUNDARY INFORMATION FROM RESIZED CHARACTERS

	Test Set Recognition Rate [%]		
	DF	MDF (120)	MDF (160)
Lowercase	70.63	71.33	71.52
Uppercase	75.93	81.58	79.98

V. DISCUSSION OF RESULTS

A. The Effect of Decreased Vector Size

An observation that was made whilst conducting our investigation was the influence of feature vector size on the overall recognition rate. In an attempt to boost the MDF recognition rate, the number of transitions included for vector creation was increased from three to four. This resulted in feature vectors of size 120 and 160 respectively. As may be seen from Tables 1 and 2, the increase in information to the network produced a higher recognition accuracy for only one of the experiments: lowercase characters using MDF with an RBF network. In general, however, the 120 input vector provided sufficient information to produce top results.

B. MLP and RBF Networks

As may be seen from Tables 1 and 2, the results for character recognition using MDF and an RBF network were superior to those using an MLP network. This may be attributed to the fact that the Gaussian function in the hidden layer of the RBF network was more successful at distinguishing between character and non-character (reject) patterns than was the MLP network.

C. DF vs MDF

As may be seen in Tables 1 and 2 above, in each case, the network trained with the MDF provides a higher

recognition rate than that trained with the standard DF. In particular, the RBF network trained with the MDF (120 inputs) using uppercase characters, demonstrated an increase of over 5% over the standard DF. This increase may be attributed to the enhanced feature information obtained from both the LT and DT values.

D. Comparison of Character Recognition Results with other Researchers in the Literature

It is always a difficult task to compare results for handwritten character recognition with other researchers in the literature. The main problems that arise are differences in experimental methodology, different experimental settings and difference in the handwriting database used. The comparisons presented below have been chosen for two main reasons. The handwriting database used by the researchers is similar to the one used in this research and/or the results are some of the most recent in the literature.

Yamada and Nakano [2] presented a handwritten word recognition system that included a character recogniser. Their classifier was trained on segmented characters from the CEDAR benchmark database. The classifier was trained to output one of 52 classes (a-z, A-Z). They recorded recognition rates of 67.8% and 75.7% for the recognition of characters where upper case letters and lower case letters were distinguished (case sensitive) and not distinguished (non-case sensitive) respectively. Therefore, if the top lower case (71.52%) and upper case (81.58%) character recognition scores in our research are averaged, a recognition accuracy of 76.55% is obtained. This recognition rate compares well with their results.

Another example where a 52-output classifier is used for segmented character recognition is in research presented by Kimura *et al.* [4]. They used neural and statistical classifiers to recognise segmented CEDAR characters. For case sensitive experiments, their neural classifier produced an accuracy of 73.25%, which was comparable to our lower case and upper case average of 76.55%.

Singh and Hewitt [6] obtained a recognition rate of 67.3% using a Linear Discriminant Analysis-based classifier. Our best results compare favourably with their top recognition rate.

Through our own experimentation [11], we found that the standard transition feature, as proposed by Gader *et al.* [3], produced results of 70.31% and 79.23% for lowercase and uppercase characters respectively. Our most recent results are higher than those described the above.

Finally, the results presented in this research (specifically those for upper case characters – 81.58%) are comparable to those presented by Camastra and Vinciarelli [7] who obtained a recognition rate of 84.52%. It must be noted that our technique requires less processing for the classification and feature extraction stages than those described in [7].

Also, a precise comparison is difficult as their classifier configuration and data set were significantly different to those described in our research.

VI. CONCLUSIONS AND FUTURE RESEARCH

This paper presented an investigation of feature extraction techniques that may be applied to the classification of cursive characters for handwritten word recognition. An MDF extraction technique was presented and was found to outperform the DF extraction technique in terms of recognition accuracy.

In future research, experiments will be undertaken using non-resized character images along with resized ones. Also, further experiments will be undertaken using thinned character image representations along with character boundaries. Finally, more experiments will be conducted with additional benchmark datasets.

REFERENCES

- [1] S-B. Cho, "Neural-Network Classifiers for Recognizing Totally Unconstrained Handwritten Numerals", *IEEE Trans. on Neural Networks*, vol. 8, pp. 43-53, 1997.
- [2] H. Yamada and Y. Nakano, "Cursive Handwritten Word Recognition Using Multiple Segmentation Determined by Contour Analysis", *IEICE Transactions on Information and Systems*, vol. E79-D, pp. 464-470, 1996.
- [3] P. D. Gader, M. Mohamed and J-H. Chiang, "Handwritten Word Recognition with Character and Inter-Character Neural Networks", *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, vol. 27, 1997, pp. 158-164.
- [4] F. Kimura, N. Kayahara, Y. Miyake and M. Shridhar, "Machine and Human Recognition of Segmented Characters from Handwritten Words", *4th International Conference on Document Analysis and Recognition (ICDAR '97)*, Ulm, Germany, 1997, pp. 866-869.
- [5] J. J. Hull, "A Database for Handwritten Text Recognition", *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 16, pp. 550-554, 1994.
- [6] S. Singh and M. Hewitt, "Cursive Digit and Character Recognition on Cedar Database", *International Conference on Pattern Recognition, (ICPR 2000)*, Barcelona, Spain, 2000, pp. 569-572.
- [7] F. Camastra and A. Vinciarelli, "Combining Neural Gas and Learning Vector Quantization for Cursive Character Recognition", *Neurocomputing*, vol. 51, 2003, pp. 147-159.
- [8] M. Blumenstein and B. Verma, "A New Segmentation Algorithm for Handwritten Word Recognition", *Proceedings of the International Joint Conference on Neural Networks, (IJCNN '99)*, Washington D.C., 1999, pp. 2893-2898.
- [9] M. Blumenstein and B. Verma, "Neural Solutions for the Segmentation and Recognition of Difficult Words from a Benchmark Database", *Proceedings of the Fifth International Conference on Document Analysis and Recognition, (ICDAR '99)*, Bangalore, India, 1999, pp. 281-284.
- [10] Parker, J. R., *Practical Computer Vision using C*, John Wiley and Sons, New York, NY, 1994.
- [11] M. Blumenstein, B. K. Verma and H. Basli, "A Novel Feature Extraction Technique for the Recognition of Segmented Handwritten Characters", *7th International Conference on Document Analysis and Recognition (ICDAR '03)*, Edinburgh, Scotland, 2003, pp. 137-141.