Experimental Analysis of the Modified Direction Feature for Cursive Character Recognition

X. Y. Liu and M. Blumenstein
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

Abstract

This paper describes and analyzes the performance of a structural feature extraction technique for the recognition of segmented/cursive characters that may be used in the context of a segmentation-based, handwritten word recognition system. The Modified Direction Feature (MDF) extraction technique builds upon a previous technique proposed by the authors that extracts direction information from the structure of character contours. This principle 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 MDF technique used in conjunction with neural network classifiers provide recognition rates of up to 90.24%, which are amongst the highest in the literature. This paper also presents a detailed analysis of the characters that were the source of misclassification in the character recognition process. The characters used for experimentation were obtained from the CEDAR benchmark database.

1. Introduction

There are many existing systems, which provide high recognition rates for separated handwritten numerals and characters [1]. However, recognition rates have not been as successful when performing experiments with characters that have been extracted from cursive handwritten words [2]-[5]. Two of the major difficulties in recognizing segmented, handwritten characters are: ambiguity and illegibility of the characters.

In past years, many feature extraction techniques have been proposed for cursive character recognition. Amongst those have been neural network-based techniques using features obtained from the local histogram of the character's chain code [4]. 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 [6]. Gader *et al.* [3] have proposed a feature extraction technique utilizing transition information from background to foreground pixels in the vertical and horizontal directions of a character image. 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 this research, the newly proposed Modified Direction Feature (MDF) extraction technique was extensively investigated. The experimental results using MDF were compared to those obtained using both Direction Feature (DF) and Transition Feature (TF) techniques. A Multi-Layer Perceptron (MLP) and a Radial Based Function (RBF) classifier were used for testing character recognition performance with the features mentioned above.

The remaining sections of this paper consist of five parts. Section 2 explains the cursive character processing procedure used in this research, Section 3 provides details of the MDF technique and the character recognition methodology and Section 4 presents experimental results. A discussion and analysis of results takes place in Section 5 and finally Section 6 draws conclusions and presents future research.

2. Cursive character datasets and processing

Two character sets were used for both the training and testing procedures in this research. The first dataset was obtained from words contained on the CEDAR CD-ROM [6], located in the CITIES/BD training and testing directories. This dataset will be referred to as the CEDAR Automatically Segmented (CAS) dataset. The second dataset was obtained from the BINANUMS/BD & BL directories of the CEDAR CD-ROM. These images were comprised of pre-segmented, Binary Alphanumeric Characters, and will be referred to as the BAC dataset.

2.1 Preprocessing

Before characters could be extracted from the CAS dataset, it was necessary to convert the original word images from the standard CEDAR format to a *.pbm* format to facilitate manipulation and further processing. Each word was then slant corrected [8]. The characters comprising the BAC dataset were likewise converted from their standard CEDAR format.

2.2 Character extraction and processing

To create the CAS dataset, it was necessary to perform automatic character segmentation and extraction. Our technique first proceeded to sequentially locate all non-cursive/printed character components through the use of character component analysis. Next, x-coordinates (vertical segmentations) for each connected character component (defined by our heuristic segmenter [9]) were used to define the vertical boundaries of each character matrix. The horizontal boundaries were determined by conducting a search from both the top and bottom of the word (bounded by the aforementioned vertical boundaries). The first instances of foreground pixels in each direction were deemed as the horizontal boundaries.

Further processing procedures for the CAS and BAC data sets differed slightly. To maintain consistency with processing procedures carried out in previous experiments (for the purpose of comparison), it was necessary to re-scale the binary images of the CAS dataset to a uniform size before further processing. However, images from the BAC dataset were further processed without re-scaling. It was then necessary to perform boundary extraction on images from both datasets in order to derive the required direction information (described below).

3. Feature extraction and character recognition

This section describes the three feature extraction techniques that were investigated in this research. The first was the standard Transition Feature (TF) [3], which was used for detecting transitions between background and foreground pixels in a character image. The second was the standard Direction Feature (DF), previously proposed by the authors [10], which was developed to provide the stroke direction of line segments in handwritten character images. The third is the recently proposed Modified Direction Feature (MDF) technique [11].

3.1 Transition Feature (TF)

The transition feature used for comparison in this research was originally proposed in [3]. In this technique, transition calculations are performed based on traversals in four directions. These are: left to right, right to left, top to bottom and bottom to top. A traversal is defined as the crossing of a particular line in a character image in one of the given directions. During a traversal, when a transition from a background to foreground pixel is encountered, the ratio between the location of the pixel and the distance across the image in that direction is computed and recorded. Each ratio value is referred to as a transition feature. Finally, all transition feature values are processed to form a transition feature vector.

3.2 Direction Feature (DF)

The Direction Feature has been extensively detailed elsewhere [10] and will only briefly be described in this section.

3.2.1 Direction determination: The goal of the DF was to simplify each character's boundary through identification of individual stroke or line segments in the image. To achieve this, character boundaries were traced from a given starting point to known intersection points. Once at an intersection, a clockwise investigation was conducted to determine the beginning and end of prospective line segments based on a number of rules [10].

Next all foreground (black) pixels (comprising individual line segments) were replaced with a set of "direction values". Direction values may be categorized into four types based on the calculated direction of individual pixels: 1) vertical, 2) horizontal, 3) right diagonal and 4) left diagonal. After replacing foreground pixels with appropriate direction values, a normalization procedure was required to generalize the direction of individual line segments. This was achieved by calculating the most frequently occurring direction value in a given line segment [10].

3.2.2 Formation of feature vectors: Once the general direction of line segments was 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 [10].

3.3 Modified Direction feature (MDF)

The MDF technique builds upon the TF and DF techniques described in Sections 3.1 and 3.2. 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. However, in MDF, aside from calculating the Location of Transitions (LTs), the Direction Transition (DT) values at a particular location are also stored. Therefore, for each transition, a pair of values such as [LT, DT] is recorded.

3.3.1 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. As in the standard transition feature extraction technique [3], the LT values in each direction are computed as a fraction of the distance traversed across the image. 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 1). 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).

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

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 normalized 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 2 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$\times$rsMatrixHeight(Width) (1)

where:
nrFeatures = 2,
nrTransitions = 3 or 4,
nrVectors = 4 and
rsMatrixHeight(Width) = 5

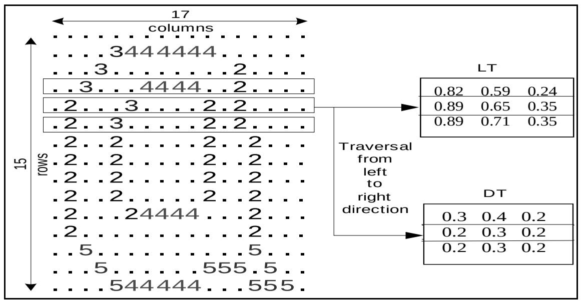
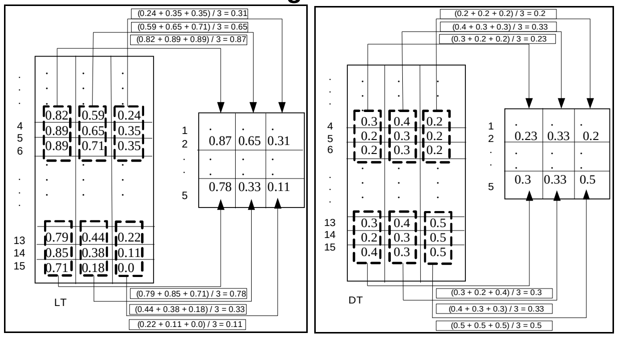


Figure 1: Processing of DT and LT values in the left-to-right direction



(a) **(b)**

Figure 2: Calculation and creation of a resampled (a) LT value vector (left-to-right direction), (b) DT value vector (left-to-right direction)

3.4 Configuration of the neural classifiers

The classifiers chosen for the task of handwritten character recognition 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 centers) 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 through experimentation. The most successful vector configurations were of size 81 for the DF and 120/160 for the MDF.

The configurations of the neural classifiers were different for each data set. For each classifier type (when using the CAS dataset), two neural networks were trained with 27 outputs each. Each network consisted of 26 desired output neurons (a-z or A-Z) and 1 reject neuron. For the BAC data set, desired outputs were combined into 36 categories and there was no reject neuron (similar to the settings described in [5]).

4. Experimental results

As mentioned previously, the training and testing data used were from the CEDAR benchmark database [6]. In this section, we present the experimental results of this research, comparing them with the performance of other techniques.

Results are displayed in tabular form for each set of experiments. Table 1 and Table 2 display the experimental results using the CAS dataset. 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. Table 3 and Table 4 present experimental results using the BAC dataset, which consisted of 19145 characters for training and 2183 characters for testing. Table 1 and Table 3 present top results for the TF, DF and MDF extraction techniques using MLPs whilst Table 2 and Table 4 present top results employing RBF networks. For comparison purposes, all feature extraction techniques were tested on boundary representations of resized characters from the CAS dataset and boundary representations of non-resized characters from the BAC dataset.

Table 1. Character recognition rates with an MLP trained using boundary information from resized characters (CAS dataset)

	Test Set Recognition Rate [%]			
	TF	DF	MDF (120)	MDF (160)
Lowercase	67.81	69.73	**70.22**	70.17
Uppercase	79.23	77.32	**80.83**	80.40

Table 2. Character recognition rates with an RBF network trained using boundary information from resized characters (CAS dataset)

	Test Set Recognition Rate [%]			
	TF	DF	MDF (120)	MDF (160)
Lowercase	70.31	70.63	71.33	**71.52**
Uppercase	79.13	75.93	**81.58**	79.98

Table 3. Character recognition rates with an MLP trained using boundary information from non-resized characters (BAC dataset)

	Test Set Recognition Rate [%]			
	TF	DF	MDF (120)	MDF (160)
36 outputs	82.82	83.65	**89.46**	88.82

Table 4. Character recognition rates with an RBF network trained using boundary information from non-resized characters (BAC dataset)

	Test Set Recognition Rate [%]			
	TF	DF	MDF (120)	MDF (160)
36 outputs	85.48	80.99	**90.24**	88.73

5. Discussion and analysis of results

5.1 General observations

Two general observations may be made from the experimental results presented above. For the MDF, two sets of inputs were created, giving feature vectors of size 120 and 160. One general prediction was that by providing more detailed input data to the neural network (in terms of the number of transitions extracted), it was possible to obtain a better recognition rate. However, as can be seen from the tables above, the increase in detail produced a higher accuracy in only one out of six cases.

It was also observed that in general, the RBF network provided more superior results when compared with the MLP. This may be due to the Gaussian function in the hidden layer of the RBF network, which functions more effectively at distinguishing between characters patterns than the MLP network.

5.2 Comparison of feature extraction techniques

As shown in all tables above, the networks trained with the MDF provided a higher recognition rate than the DF and TF techniques in each case. In particular, the MLP network trained with the MDF (120 inputs) for the BAC dataset, demonstrated an increase of approximately 6% and 7% over the DF and TF techniques respectively. This increase may be attributed to the enhanced feature information obtained from both the LT and DT values.

5.3 Investigation of incorrectly recognized characters

In this research, aside from testing the performance of the MDF technique compared with two other techniques, it was also important to perform an analysis of the characters that were incorrectly recognized to determine the distribution of errors in the networks.

Of great interest were the classification errors generated by the MDF technique using the BAC data set, as it provided some of the top results in this research. For the network configuration obtaining one of the top recognition rates (MLP), out of a total of 2183 characters in the testing set, there were 250 characters that were not successfully recognized. Firstly, an investigation was performed recording which individual classes of characters provided the most errors for the network. It was found that characters ‘i’, ‘l’, ‘v’, and character ‘R’ provided the most misclassifications out of the 36 possible outputs. Figure 3 illustrates the distribution.

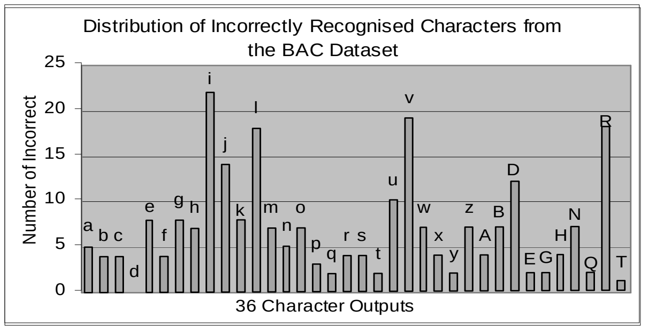
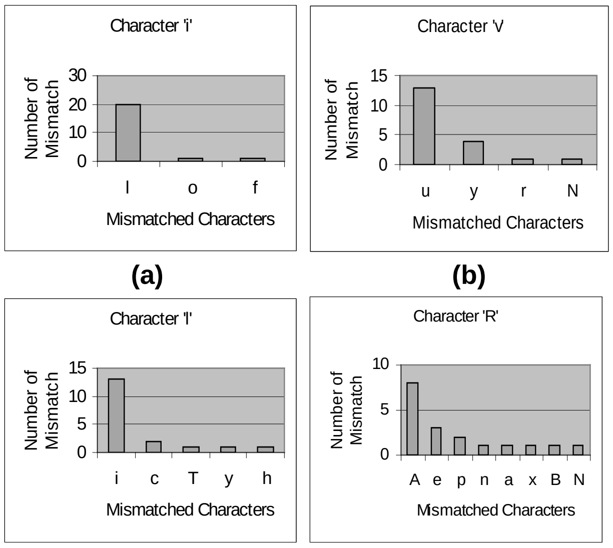


Figure 3. Error distribution for incorrectly recognized characters

The investigation continued to determine which of the remainder of the characters were mostly being confused with the top four incorrectly recognized classes. It was found that the character ‘i’ was mostly incorrectly recognized as the character ‘l’ (and vice-versa). The character ‘v’ was mainly confused with the character ‘u’, and ‘R’ was mostly confused with ‘A’. The following figures present the error distributions of each of the above characters.



(c) (d)

Figure 4. Error distribution of mismatched characters for (a) character 'i' (b) character 'v' (c) character 'l' and (d) character 'R'

During the investigation, it was noticed that in certain instances, due to the ambiguities of some character images, the neural network was not at all able to give a reliable confidence to determine a correct output. In other words, when the MLP encountered 'ambiguous' character features, it ranked many of the possible outputs at confidence values below 0.3. For the MLP test case using the BAC dataset (mentioned above), there were 37 out of 250 such characters that were incorrectly recognized.

Another interesting result of the investigation was that if the top two character confidences output by the MLP were used to determine the character recognition rate, the recognition error decreased from 250 incorrectly recognized characters to 101. Hence, when the top two confidences output by the MLP are taken into account, the character recognition rate reaches approximately 95%.

5.4 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 the difference in the handwriting database used. The comparisons presented below have been chosen for two main reasons. The handwriting databases used by the researchers were similar to those 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 recognizer. 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 rates from our research are averaged (using the CAS dataset), a 76.55% recognition accuracy is obtained. This recognition rate compares well with their results. Our top recognition accuracy using the BAC data set (90.24%) exceeds their top result by nearly 14%.

Another example where a 52-output classifier was used for segmented character recognition was in research presented by Kimura *et al.* [4]. They used neural and statistical classifiers to recognize 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%. Our top recognition accuracy using the BAC dataset exceeded theirs by almost 17%.

Singh and Hewitt [5] obtained a recognition rate of 67.3% using a Linear Discriminant Analysis-based classifier. Our best result using the BAC dataset exceeds their top recognition rate by nearly 23%.

Through our own experimentation [10], 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 on the CAS data set are higher than those described above.

Finally, the results presented in our research (specifically those for the BAC dataset - 90.24%) compare favorably to those presented by Camastra and Vinciarelli [7] who obtained a recognition rate of 84.52%. As in most of the results above, a precise comparison is difficult, as Camastra and Vinciarelli's classifier configuration and dataset composition were different to those described in our research.

6. Conclusions and future research

This paper presents a detailed investigation and comparison of feature extraction techniques for cursive character recognition. The MDF technique provided the top recognition rate for cursive handwritten characters, outperforming the best TF result by nearly 5% and the best DF result by nearly 6%. The MDF also compared well with some of the top results in the literature, outperforming other techniques by 6-23%.

In addition, a detailed analysis of the incorrectly classified characters provided by one of the MDF neural network configurations revealed four main character classes consistently giving misclassifications. These four character classes were then further scrutinized to determine the source of confusion. The results of this analysis provide future directions for enhancing the current classification configuration to take into account those characters that provide the most errors.

In future research, a number of additional considerations will be addressed to further investigate and enhance the recognition process. These include conducting further experiments with additional benchmark datasets, improving the preprocessing methodology and enhancing segmentation techniques. Further combinations of classifiers and local and global features will also be explored.

7. References

[1] S-B. Cho, “Neural-Network Classifiers for Recognizing Totally Unconstrained Handwritten Numerals”, *IEEE Trans. on Neural Networks*, vol. 8, 1997, pp. 43-53.

[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, 1996, pp. 464-470.

[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] 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.

[6] J. J. Hull, “A Database for Handwritten Text Recognition”, *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 16, 1994, pp. 550-554.

[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, "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,.

[9] 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.

[10] 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.

[11] M. Blumenstein, X. Y. Liu and B. Verma, "A Modified Direction Feature for Cursive Character Recognition", *International Joint Conference on Neural Networks (IJCNN ’04)*, Budapest, Hungary, 2004, (accepted).