# An Investigation of the 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 Phone: +61 7 5552 8271 Fax: +61 7 5552 8066 E-mail: m.blumenstein@griffith.edu.au B. Verma School of Information Technology Central Queensland University Bruce Highway North Rockhampton QLD 4702, Australia E-mail: b.verma@cqu.edu.au

\*Corresponding Author

# An Investigation of the Modified Direction Feature for Cursive Character Recognition

Abstract - This paper describes and analyses the performance of a novel 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 the Direction Feature (DF) technique proposed previously that extracts direction information from the structure of character contours. This principal was extended so that the direction information is integrated with a technique for detecting transitions between background and foreground pixels in the character image.

In order to improve on the DF extraction technique, a number of modifications were undertaken. With a view to describe the character contour more effectively, a re-design of the direction number determination technique was performed. Also, an additional global feature was introduced to improve the recognition accuracy for those characters that were most frequently confused with patterns of similar appearance. MDF was tested using a neural network-based classifier and compared to the DF and Transition Feature (TF) extraction techniques. MDF outperformed both DF and TF techniques using a benchmark dataset and compared favourably with the top results in the literature. A recognition accuracy of above 89% is reported on characters from the CEDAR dataset.

**Key Words:** Handwritten Character Recognition, Pattern Recognition, Image Processing and Computer Vision, Neural Networks.

#### **1. INTRODUCTION**

Handwriting recognition is a technology that allows machines to reliably recognize handwritten material written by humans. The concept can be divided into 2 main areas – on-line and off-line handwriting recognition. An off-line handwriting recognition technique is an approach that interprets characters, words and/or cursive scripts that have been written on a common surface (i.e. paper) [1]. On-line handwriting recognition refers to automatically recognizing handwritten characters/words using real-time information such as pressure and the order of strokes made by a writer usually employing a stylus and pressure sensitive tablet [2].

The main approaches that exist for off-line cursive word recognition may be divided into segmentation-based and holistic ones. In general, the former approach uses a strategy based on the recognition of individual characters or patterns whereas the latter deals with the recognition of the word image as a whole [2]. To this day, research into the recognition of cursive handwriting still continues to be concerted. This sustained motivation may be attributed in part to the challenging nature of the problem as well as the countless number of commercial areas that it may be applied to [3]. Applications that have received particular attention in recent times include postal address recognition [4], [5], bank cheque processing [6], [7], and form processing [8].

In the segmentation-based strategy for handwritten word recognition, the objective is to oversegment the word a sufficient number of times to ensure that all appropriate letter boundaries have been dissected. To determine the best segmentations, a set of hypotheses are tested by merging segments of the image and invoking a classifier to score the combinations. Most techniques employ an optimization algorithm making use of some sort of lexicon-driven, dynamic programming technique and possibly incorporating contextual knowledge. The basic approach described above was proposed simultaneously by a number of researchers [5], [9]-[12].

A crucial component of the segmentation-based strategy is the development of a classification system for scoring individual characters and character combinations. The literature is replete with high accuracy recognition systems for separated handwritten numerals [13]-[15]. However, although recent techniques show promising results [16], the same measure of success has not been attained for segmented or cursive characters [11], [17]-[24]. There are three main problems faced when dealing with segmented, handwritten character recognition: the first relates to the ambiguity of the character without the context of the entire word, for example, an "I" may look similar to an "i". The second problem relates to the illegibility of certain characters due to the nature of cursive writing, i.e. ornamentation, distorted character shape etc. [25]. Finally, the process of segmentation may itself introduce some anomalies depending on the algorithm used. Certain algorithms may not locate the segmentation path or anchorage point accurately and may sometimes dissect adjacent character components [26].

In order to address the problems discussed above, researchers have explored two main approaches: (1) determining features best suited for recognition and (2) investigation of different classification schemes [20]. Yamada and Nakano investigated a standard technique for feature extraction based on direction histograms in character images [17]. They used a multi-template strategy with clustering for the recognition of segmented characters from words in the CEDAR database [27]. Kimura *et a*/. [18] investigated a similar feature extraction technique calculating local histograms based on chain code information in segmented handwritten characters. They used

statistical and neural classifiers for the recognition of segmented CEDAR characters. Gader et al. have proposed a feature extraction technique utilizing transition information for recognizing segmented characters [11]. Their technique was based on the calculation and location of transitions from background to foreground pixels in the vertical and horizontal directions. The authors used neural networks trained with the back-propagation algorithm for recognizing characters obtained from US postal words. Other studies by Camastra and Vinciarelli [21], [23] have proposed feature extraction techniques generating local and global features. The local features obtained from sub-images of the character included foreground pixel density information and direction information. The global features that were used included the fraction of the character appearing below the word baseline and the character's width/height ratio. The authors used learning vector quantization (LVQ) [21] and a combination of neural gas and LVQ classifiers [23] for the recognition of segmented characters from the CEDAR database. Another recent study into cursive character recognition by Blumenstein et al. [25] has proposed a local feature extraction technique, which retrieves direction information from character contours. Their proposed technique replaces the foreground pixels from a character contour with appropriate numerical direction values. Then the image is evenly divided into a number of windows, and features are locally extracted from each window.

#### 1.1. Contributions of the Proposed Research

In this research, a novel feature extraction technique is presented for extracting structural features from cursive handwritten characters. This Modified Direction Feature (MDF) extraction technique combines local feature vector and global structural information and provides integrated features to a neural network for training and testing. The proposed approach first employs an existing character outline tracing technique, which traces the contour of a given character image. Then, the directions of line segments comprising the characters are detected and the foreground pixels are replaced with appropriate direction values. Finally, features of the characters, based on the location of background to foreground pixel transitions, are extracted and neural training and classification is performed.

Following on from earlier work presented in [28], the current paper extends this research through the following novel contributions. A revised and extended version of MDF is proposed including (a) novel enhancements to the direction value detection technique, (b) a new and enhanced version of the algorithm for distinguishing individual line segments and direction normalisation, (c) a detailed algorithm for calculation of location and direction transitions (LTs and DTs respectively), (d) an algorithm for re-sampling the raw LT and DT values for MDF extraction and (e) the proposal of a global feature (ratio) added to the MDF feature vector.

In addition to this, extended experimental results are presented using two individual character data sets, a combined data set and a comparison of MDF and MDF-Ratio. Finally, a new/more detailed analysis of all experimental results is presented including (a) an investigation of incorrectly recognised characters using MDF and MDF-Ratio, (b) an extended comparison of results using MDF and those feature extractors proposed by other researchers in the literature (c) a statistical comparison of results between MDF and MDF-Ratio.

The remainder of this paper is broken down into four sections: Section 2 describes the principle of the proposed feature extraction technique. Section 3 provides experimental results. Analysis and discussions take place in Section 4, and finally Section 5 presents conclusions and future work.

#### 2. Research Methodology

#### 2.1. System Overview

The modified direction feature extraction technique combines the advantages of both the traditional direction feature extraction and transition feature extraction techniques [28]. Part of the direction feature extraction technique was used to provide stroke direction information, and the transition feature extraction technique was also used to provide structural information of a character. The local averaging method was used to avoid local stroke shifts.

There are a number of steps in the whole process of obtaining an MDF vector from character images. The first step is to pre-process the input character image. For cursive handwritten characters, most of them are slanted, skewed and full of noise. Therefore, there is a need to pre-process the images so that more accurate features can be extracted. In general, the image pre-processing stage includes skew, slant detection and correction, noise removal and underline removal.

Then, the boundary of each character needs to be retrieved. This is important to narrow the scope of the information input to the feature extractor and subsequently to the classifier. This step, amongst others, purports to reduce classification time and to facilitate the extraction of significant features. Next, replacement of foreground pixels with direction values needs to take place on the boundary of the image to enable stroke direction determination. This high-level process is illustrated in Figure 1.



Figure 1. System Overview

# 2.2. Modified Direction Feature (MDF) Extraction

The previous section briefly described the methodology of the MDF technique. This section explains the details of each component of the MDF extraction technique. Section a) illustrates the process of obtaining direction values. Section b) explains how MDF features are calculated and finally, the calculation of an additional global feature – "ratio feature" is introduced in Section c).

#### 2.2.1. Obtaining Direction Values

The process of obtaining direction values in the MDF technique is different from that of the Direction Feature (DF) extraction technique [25] in many respects. The following sub-sections depict the detail of detecting direction values from foreground pixels using the MDF technique.

# 2.2.1.1. Assumptions

For the direction value detection algorithm used in this research, there are two assumptions made relating to the image that is to be processed. First, this image is assumed to be binary and second, the image has been pre-processed so that only the boundary of the image remains.

#### 2.2.1.2. Direction Values

The new array of direction values (updated from DF) are i) 2 for vertical direction, ii) 3 for right diagonal direction, iii) 4 for horizontal direction, iv) 5 for left diagonal direction and v) 8 for starting point (see Figure 2). This starting point is the number used in the intermediate process and when processing of the image is completed, all starting point values are normalised; therefore, the direction value 8 is not shown in Figure 2.



Figure 2. Direction Values used for MDF

# 2.2.1.3. Finding Starting Points

The modified algorithm for obtaining direction values searches for a certain black pixel to start the image processing procedure; these pixels are called starting points. The starting point is defined as the first black pixel found at the bottom-most and left-most location in a given character image. In this direction detection algorithm, a new starting point is sought whenever an end of a connected line segment (stroke) is encountered. See Figure 3.

11111111	38444444
1 1	3 8
1 1111 1	3 3444 2
••••••	8 8 8 7
.111.1	
.111.1	
. 1 1 1 1	.2228
1 1 1 1	
$\cdot \pm \cdot \cdot \pm \cdot \cdot \cdot \cdot \cdot \pm \cdot \cdot \pm \cdot \cdot \pm \cdot \cdot \cdot = 1$	2 2 2 2 2
• _ • • _ • • • • • _ • • _ • • _ • • •	
.1111	
.1111	.2822.
. 1 111111 1	. 8 44448 7
1 1	5 2
•••	
$\dots 1111111\dots 111$ .	844448445.
E'	Ctanting Daints
Figure 3. Seeking S	Starting Points

Also, in this algorithm, all starting point pixels are temporarily replaced by the value "8". After the normalisation stage (described below), each starting point is normalised by the predominant direction value in the line segment it belongs to.

# 2.2.1.4. Distinguishing Individual Line Segments

The rules for distinguishing individual line segments used in this algorithm are similar to the rules used in the DF extraction technique. Essentially, the algorithm collects each foreground pixel and based on the previous direction, it converts the foreground pixel to an appropriate direction value whilst at the same time checking for line segment conditions. Various rules serve as line segment conditions. If one of the conditions is met, then a line segment is found. The rules used for distinguishing line segments by the direction value detection algorithm in this research are:

- A corner condition is found OR
- During line segment detection, when a change in direction occurs more than three times OR
- A change in direction AND the previous direction has been continuously the same AND the length of the previous direction is greater than three pixels

In order to save confusion, it is important to clarify what the corner conditions are. During traversal along the boundary of a given character image, if there is a sudden change in direction and such a change forms a particular angle, then a corner is said to be found. As previously described, if a corner condition is met, a line segment (stroke) is found and the search for a new line will begin.

There are 8 corner conditions in total; among these there are 4 vertical corner types. The first type is defined when the previous direction is an upward right direction and the current direction is facing upward left (See Figure 4a). The second type is when the previous direction is upward left and the current is the upward right direction (See Figure 4b). The third type is when the previous

direction is down-right and the current direction is down-left (See Figure 4c). The fourth type is when the previous direction is down-left and the current direction is down-right (See Figure 4d).



There are 4 other horizontal corner types, the first type is when the previous direction is facing the up-right direction and the current direction is facing down-right (See Figure 5a). The second type is when the previous direction is down-right and the current direction is up-right (See Figure 5b). The third type is when the previous direction was going upward left and the current direction is pointing in the downward left direction (See Figure 5c). The last type is when the previous direction was facing down-left and the current direction is pointing up-left (See Figure 5d).



Figure 5. Horizontal Corner Conditions

# 2.2.1.5. Direction Value Normalisation

There are three steps to normalising a line segment. The first step is to find the most frequently occurring direction value in a line segment. The second step is to use the most occurring value to replace the values of the other pixels in the current line segment, except the starting point. The last step is to normalise the starting point pixel. In other words, the value of each starting point will be converted from 8 to the normalised value of the line segment that the starting point belongs to.

Figure 6 illustrates the complete process of direction detection and line segment normalisation. As can be seen, an image of lowercase "a" in its boundary format is shown in Figure 6a. This image is then fed into the new direction detection algorithm. Figure 6b shows the intermediate stage of image processing. One can see from Figure 6b that the location of each starting point and the direction values for all other foreground pixels has been identified. And in Figure 6c, all line segments in the image have been normalised.



Figure 6. Direction Normalisation in a Given Image

# 2.2.2. Obtaining the Modified Direction Feature

The proposed MDF technique builds upon the DF and TF techniques described in previous sections [28]. The main difference is in the way the feature vector is created. For TF, 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 recorded. Therefore, for each transition, a pair of values such as [LT, DT] is stored.

To calculate LT values, it is necessary to scan each row in the image from left-to-right and rightto-left. Likewise, each column in the image must be scanned from top-to-bottom and bottom-totop. As in the standard transition feature extraction technique [11], 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 7a). 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).

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 based on the maximum numerical values used to describe line segments in the DF technique [25] to facilitate the calculation of floating-point values between 0 and 1 (See Figure 7).



Therefore, following the completion of the above, four vectors would be present for each set of feature values (eight vectors in total). For each of the 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 rescaling was set to a value of 5, which is consistent with that used for the TF technique [11]. 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  $5 \times 3$  matrix (as shown in Figure 7, 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 (Figure 8):

Total MDF features = Feature Pair \* Number of Transitions \* Number of Directions \* Re-sampled Matrix size

Figure 8. Calculation of the Total Number of MDF Features

where: Feature Pair [LT,DT] = 2, Number of Transitions = 3 or 4, Number of Directions = 4 and Re-sampled Matrix Size = 5

The complete algorithm for the calculation of location transitions and direction transitions can be viewed in the format of pseudo-code and is shown below:

```
For each direction of traversal
```

```
For i = 0 to number of lines
           For j = 0 to number of transitions
                   IF traversal is from left to right THEN
                           LT = 1 - v / character width
                   ELSE IF traversal is from top to bottom THEN
                           LT = 1 - v / character height
                   ELSE IF traversal is from right to left THEN
                          LT = v / character width
                   ELSE IF traversal is from bottom to top THEN
                          LT = v / character height
                   END IF
                   DT_{(v)} = d_v / 10
                   Record [LT, DT_{(v)}] as a feature pair in feature vectors
           END For
    END For
END For
```

In the algorithm above, a line of traversal refers to a row or a column, let *LT* be the value of the transition value and  $DT_{(v)}$  be the value of the direction feature at the position where a transition occurs. When a background to foreground pixel transition occurs, the exact location of the transition is represented by v, and the direction number at the position of transition is denoted by  $d_v$ .

As mentioned earlier, after obtaining the raw feature vectors in 4 directions, a further resampling (local averaging) of the above vectors was necessary so that the NC/NR dimensions were normalized in size. The complete algorithm of performing such a local averaging task is shown below in the format of pseudo-code:

```
For each dimension of the feature matrix
       IF it is an NC dimension THEN
              L = number of columns
       ELSE
              L = number of rows
       END IF
       G = L / re-sampling size
       For i = 0 to L
              For j = 0 to number of transitions
                      Let m = i
                      Let n = i + G
                      aveLT = (LT_m + LT_{m+1} + ... + LT_{n-1}) / G
                      aveDT = (DT_m + DT_{m+1} + ... + DT_{n-1}) / G
                      record [aveLT, aveDT] as a feature pair in the re-sampled feature matrix
              END For
              Set i = n
       END For
 END For
```

In the algorithm above, L is used to denote the length of either a row or a column. The re-sampled matrix size in this research is set to 5. G represents the number of elements (intervals) needed to perform local averaging. Variables m and n represent the starting and ending positions for local averaging. Finally, *aveLT* and *aveDT* represent the averaged LT and DT values.

# 2.2.3. Adding Ratio Feature

Based on the analysis undertaken in the preliminary investigation of the MDF technique, an extra feature was deemed to be necessary. As may be seen in section IV, confusion occurs very frequently when encountering alphabet letters '1' and 'i'. It has been observed that in general, lower case 'i' has a shorter and wider shape compared to the lower case '1', conversely, the letter '1' has a taller and narrower shape compared to 'i'. Therefore, the authors were interested in seeing the significance of introducing a width to height ratio to the list of features. In the following sub-section, the feature itself and the description of the calculation procedure are discussed.

#### 2.2.3.1. The Ratio Feature

The extra feature that was deemed necessary and was added to the feature vector was the character width to height ratio. Since this extra feature of width to height ratio was added, an extra assumption was required. This assumption stated that character images could not be resized. The reason for this was because if all training character images were resized to a uniform size in the pre-processing stage, then the ratio value would be the same for all characters and therefore would not influence the recognition rate.

# 2.2.3.2. The Problem of Obtaining Width to Height Ratio

The easiest approach for acquiring the ratio feature is by performing a division of the character width and height directly. However, there is a problem with this approach. The problem comes from both the image size and the neural network classifier.

On one hand, the Multi-layer Perceptron (MLP) classifier used in this research accepts predominantly feature values ranging between 0 and 1, if the feature values are outside of this boundary, undesirable results may occur. On the other hand, it was assumed previously, that the image must not be resized. So the value of width to height ratio can be much greater than 1.0. For example, if the character image is very long in terms of width and short in terms of height, i.e. 150 pixels wide and 15 pixels high, the ratio value will be 10 and this is not acceptable since it is greater than 1; therefore, this way of getting the ratio value is not feasible and other methods of calculating the ratio value have been investigated.

# 2.2.3.3. The Solution to the Problem

The solution to this problem is to use the angle ratio to represent the width to height ratio. The idea is to view a character image as a rectangle. Then, the rectangle is divided diagonally, resulting in two symmetrical right triangles (See Figure 9).



Figure 9. Dividing a Character Image Symmetrically

In fact, only one triangle is needed, and in this research, the top triangle was chosen to calculate the ratio. As illustrated in Figure 10, firstly, the angle  $\alpha$  is calculated by using the arc tan function. Once the character width and height are input to the formula, the angle  $\alpha$  is obtained in radians. The ratio is then calculated by dividing the  $\alpha$  value by half of pi.



Figure 10. Calculation of the Ratio Value

The advantage of retrieving the ratio value in this way is that no matter how wide and short or tall and thin the images are, ratio values ranging between 0 - 1 are always ensured. The wider and shorter the image is, the closer the ratio is to 1. Also, the taller and thinner the image is, the closer the ratio value is to 0. But 0 and 1 will never be achieved since there is always a proportion between width and height. Although this value is not directly calculated by dividing the width and height values, it does however use the ratio between the width and height. So it is a true alternate representation of a character width and height ratio. This version of the MDF technique, incorporating the ratio feature, is called MDF-Ratio (MDF-R).

# **3. EXPERIMENTAL RESULTS**

The word images used for experiments were obtained from the CEDAR benchmark dataset [27]. The first character data set used is called the CEDAR Automatically Segmented (CAS) data set. The second data set was comprised of pre-segmented Binary Alphanumeric Characters (BAC) and therefore called the BAC data set. More details on the creation of CAS and the nature of BAC may be found in [1]. The classifier used in this research was a feed forward, Multi-layer Perceptron (MLP) trained with the Error Backpropagation (EBP) algorithm. The dataset and experimental environment was identical throughout all phases of experimentation.

#### 3.1. Individual Data Set Experiments

The first sets of results are displayed in tabular form for each sub-set of experiments. Table 1 displays 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 1 presents the top result for the TF, DF and MDF extraction techniques using the MLP. For comparison purposes, all feature extraction techniques were tested on boundary representations of resized characters from the CAS dataset.

	Test Set Recognition Rate [%]												
	TF DF MDF												
Lowercase	67.81	69.73	70.26										
Uppercase	79.23	77.32	80.72										

 Table 1. Character recognition rates with an MLP network trained using boundary information from resized characters on the CAS dataset

The second set of results is displayed in tabular form for each sub-set of remaining individual data set experiments. Table 2 displays the experimental results using the BAC dataset. Experiments were conducted for a combination of lower case and upper case character patterns. It was noticed by Singh [19] that for some characters, their upper case shapes are similar to their lower case equivalents. Therefore, by combining the upper and lower case characters, some characters may be merged into one category or class. For example, the shape of the upper case letter 'O' and the lower case letter 'o' are extremely similar especially in the context of cursive handwriting. The order of the 36 outputs used in this research is shown in Figure 11 below.

а	b	С	đ	e	f	g	h	i	j	k	1	m	n	0	p	q	ľ	S	t	u	V	W	X	y	Ζ	A	В	D	E	G	Н	Ν	Q	R	Τ
1	2	3	4	5	б	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36

Figure 11. 36 Categories of Outputs

A total of 19145 lower case and upper case character patterns were prepared for training. A further 2183 lower case and upper case patterns were used for testing. Table 2 presents the top results for the TF, DF and MDF extraction techniques using the MLP trained with EBP. For comparison purposes, all feature extraction techniques were tested on boundary representations of non-resized characters from the BAC dataset.

 

 Table 2. Character recognition rates with an MLP network trained using boundary information from nonresized characters on the BAC dataset

	Test Set Recognition Rate [%]												
	TF	DF	<b>MDF(120)</b>										
Combined 36 outputs	82.82	83.65	89.01										

#### 3.2. Combined Data Set Experiments

A combined dataset were created for further experiments. This dataset consisted of training and testing characters from both CAS and BAC datasets mentioned above. There are three reasons why the BAC and CAS datasets were combined. The first reason was that by combining these two datasets, a much larger training and testing database would result and theoretically more training examples could be provided to the neural network for superior learning. The second reason was that the BAC dataset was known as an easy or 'clean' dataset, and the CAS dataset was known as a difficult dataset even for humans to recognise. Again in theory, the presentation of character patterns of mixed difficulty was hypothesised to provide the neural network with a more effective data set for training. The final reason was that by combining the BAC and CAS datasets, it was easier and more meaningful to compare the results of this research with the recognition rates of other researchers in the literature that used larger datasets.

After combining the datasets, a total of 34243 training characters and 3372 testing characters were generated. For the CAS dataset, there were many 'garbage' or non-character images, which were used for training the classifiers as reject patterns. In order to create a dataset with 36 outputs, the reject characters needed to be removed from both the CAS training dataset and the CAS testing dataset.

Based on the fact that MDF provided a higher recognition rate than the TF and DF extraction techniques in Section 3.1, further experiments were conducted using a combined data set to ascertain the significance of adding a ratio feature (MDF-R) to MDF. Experiments were conducted with identical neural classifier settings in order to compare the performance between MDF and

# MDF-R in terms of recognition rate. A random sub-set of the experimental results conducted,

using the combined data set, is shown in Table 3.

	<b>Recognition Rate [%]</b>				
Num. Hidden Units	Num. Iterations	Num. Features	Num. Outputs	MDF	MDF-R
50	3500	120	36	80.34	80.6
65	4500	120	36	81.97	82.3
80	6000	120	36	83.16	82.74
110	9000	120	36	83.16	83.07
120	10000	120	36	83.45	84.16
135	11500	120	36	83.78	84.25
138	9000	120	36	83.87	84.19
140	10000	120	36	83.78	84.13
150	11000	120	36	83.69	84.16
160	11500	120	36	83.9	84.46
170	14500	120	36	83.51	84.34
180	14000	120	36	84.37	84.49
180	14500	120	36	84.57	84.58
180	15000	120	36	84.4	84.46
190	16000	120	36	83.78	84.05

Table 3. Character recognition rates with an MLP trained using boundary information from non-resized characters on the combined dataset employing MDF and MDF-R techniques

# 4. ANALYSIS AND DISCUSSION OF RESULTS

An analysis was conducted after the completion of each set of experiments. The analysis following the individual data set experiments (Section 3.1) was used as a general direction for subsequent experiments in this research. In this section, the analysis of each experiment set is summarised in the following sub-sections.

#### 4.1 Analysis of Experiments

The analysis of individual data set experiments (Section 3.1) was undertaken in three main ways, and each is described in a separate sub-section. Sub-section 4.1.1 analyses the performance of each feature extraction technique. Then, an investigation of incorrectly recognized characters is

conducted in sub-section 4.1.2. Finally, sub-section 4.1.3 provides a comparison of the character recognition results with other researchers in the literature.

#### 4.1.1 Comparison Between Feature Extraction Techniques

As shown in Table 1 and Table 2, 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 5% and 6% 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.

# 4.1.2 Investigation of Incorrectly Recognized Characters Employing MDF

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 provided by the MLP classifiers.

Of great interest were the classification errors generated by the MDF technique using the combined data set, as it provided some of the top results in this research. For the network configuration obtaining one of the top recognition rates employing MDF, out of a total of 3372 characters in the testing set, there were 520 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', 'e', and the character 'a' provided the most misclassifications out of the 36 possible outputs. Figure 12 illustrates this distribution.



Figure 12. 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 'e' was mainly confused with the character 'i', and 'a' was mostly confused with 'o'. The following figures present the error distributions of each of the above characters (Figure 13).



Figure 13. Error Distribution of Mismatched Characters for (a) character 'l' (b) character 'e' (c) character 'i' and (d) character 'a'.

During the investigation, it was noted that due to the ambiguities of some character images, in certain instances, the neural network was unable to give a reliable confidence to determine a correct output at all. 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 combined dataset (mentioned above), there were 77 out of 520 such characters that were incorrectly recognized in this manner.

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 520 incorrectly recognized characters to 300. Hence, when the top two confidences output by the MLP are taken into account, the character recognition rate reaches approximately 91%.

# 4.1.3 Investigation of Incorrectly Recognized Characters Employing MDF-R

A similar analysis on the classification errors generated by the MDF-R technique using the combined data set was also conducted. Although the overall distribution of the classification error did not change dramatically and the top 4 mis-classified characters remained the same as those obtained using MDF, there are some noticeable differences. The most noticeable improvement is that by adding the ratio feature value to the MDF vector, the mis-classification of the lower case letter '1' decreased by approximately 21.5%. As can be seen from Figure 13(a), '1' was most confused with lower case character 'i'. However, after the ratio feature was employed, the mis-classification from '1' to 'i' decreased by almost 31%. In addition, the result of the analysis showed a similar figure on the number of mis-classifications between 'i' and '1'. In other words,

the ratio feature seemed to assist in differentiating between the characters 'l' from 'i' whilst preventing large increases in mis-classifications from 'i' to 'l'.

Another noticeable improvement was the decrease of recognition error when the top two character confidences were used to determine the recognition rate. Previously, when employing MDF, the recognition error was decreased from 520 to 300 when the top two character confidences were taken into account. After employing the additional ratio feature, this error was further reduced to 285. In other words, the recognition accuracy increased from 91.1% to 91.55% by just adding this global feature.

In summary, the addition of the ratio feature lowered the confusion in distinguishing between the most difficult character classes, at the expense of increasing the error slightly for some of the remaining characters. Overall, this is an acceptable outcome in our investigation.

# 4.1.4 Comparison Between 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 (i.e. cursive/segmented characters from the CEDAR database) and/or the results are some of the most recent in the literature.

Yamada and Nakano [17] 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, the top recognition rate from the combined dataset in this research (84.58%) should be used for comparison purposes. This recognition rate compares well with their result. The top recognition accuracy using the BAC data set in this research (89.01%) exceeds their top result by more than 13%.

Another example where a 52-output classifier was used for segmented character recognition was in research presented by Kimura *et al.* [18]. 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 again was comparable to the combined dataset result of 84.58% presented in this research. The top recognition accuracy in this research using the BAC dataset exceeded theirs by more than 16%.

Singh and Hewitt [19] obtained a recognition rate of 67.3% using a Linear Discriminant Analysisbased classifier. The result obtained from the combined dataset (84.58%) compares well to their recognition rate. Furthermore, the best result using the BAC dataset exceeds their top recognition rate by nearly 22%. From previous experimentation [25], it was found that the standard transition feature, as proposed by Gader *et al.* [11], produced results of 70.31% and 79.23% for lowercase and uppercase characters respectively. The most recent results on the CAS data set are higher than those described above. Moreover, the top result obtained from the combined dataset, which also contains upper and lower case characters, exceeds their top result by more than 5%.

Finally, the results presented in our research (specifically those for the BAC dataset – 89.01% and the combined dataset – 84.58%) compare favourably to those presented by Camastra and Vinciarelli [23] 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 this research.

#### 4.2 Comparison Between MDF and MDF-R

Although the top results obtained by the MDF-R technique in Table 3 (84.58%) show a marginally better performance than the standard MDF technique (84.57%) whilst employing the same network configuration, this type of comparison is not sufficient on the whole to indicate whether MDF-R outperforms MDF. Hence, in order to evaluate overall whether the MDF-R technique is a better feature extraction technique than MDF for off-line character recognition, it was decided to perform a paired t-test to evaluate on average which feature extraction technique performed better, based on all the results obtained in Table 3. This was carried out in preference to an independent t-test, to determine if the results obtained were attributed to the feature extraction technique employed and not as a result of the treatment utilised (the neural network classifier and configuration used). It was hypothesized that if there was no significant difference between the

recognition rates obtained by the two techniques, then p>0.05. However, if there was a significant difference in the recognition rates obtained by the two techniques, then p<0.05.



Figure 14. Modified box plot illustrating the recognition rate performance based on the MDF and MDF-R feature extraction techniques.

The results obtained from the paired t-test found that there was a significant difference in the recognition rates obtained by the two techniques, as p<0.05, being 0.004 (n=15), irrespective of the treatment employed. A box plot in Figure 14, was used to show which technique performed better. As can be seen from Figure 14, the MDF-R technique had a higher average recognition rate of 83.7% for all 15 experiments, than did the MDF technique, which had a mean recognition rate of 83.4%. There is a large amount of variation in the recognition rates obtained using MDF-R between the lower quartile and minimum value, however this is only as a result of 3 values. In addition, there exists a potential outlier ( $*^1$  representing 80.6%) as illustrated in the Figure. The

majority (11) of the recognition rates obtained using MDF-R for the 15 experiments, lay above the mean value of 83.73%, whereas for the MDF technique, only 8 recognition rates lay above the mean value of 83.45%. Also, the modified box plot identifies two of the results obtained by MDF as potential outliers ( $o^{17}$  representing 81.97% and  $*^{16}$  representing 80.34%).

Overall, it was found that the recognitions rates obtained using MDF-R were significantly higher than the recognitions rates obtained by the MDF technique and hence it may be concluded that it performed better as a feature extraction technique.

#### **5. CONCLUSIONS AND FUTURE RESEARCH**

In conclusion, a novel feature extraction technique for cursive handwritten characters is proposed. Experiments have been conducted exhaustively to test the significance of the proposed technique. From the results, the original MDF outperformed the traditional TF and DF in terms of recognition accuracy. This research also described a further improvement with regards to recognition rate, by adding an additional feature and modifying the extraction process. A comparison with top results in the literature indicates that the proposed feature extraction technique provides comparable and in most cases higher recognition accuracy for cursive characters.

Besides performing experiments, an analysis and discussion of the results, including the strengths and weaknesses of the technique, were also investigated. Upon performing the experiments and conducting an analysis of the results, it was concluded that the MDF extraction method which combines direction values, transition features and width to height ratio (MDF-R) provided the most descriptive information for MLP classification and consequently provided the best overall recognition rate.

In future research, a number of considerations will be addressed to further investigate and enhance the technique. These include conducting further experiments with thinned characters and comparing the results obtained between the boundary and thinned versions of character images. Also curve information will be incorporated in the process of determining direction values. Finally, further investigations into mis-classifications will be conducted.

#### REFERENCES

- [1] B. Verma, M. Blumenstein, M. Ghosh, "A novel approach for structural feature extraction: contour vs. direction", Pattern Recognition Letters, vol. 25, pp. 975 988, 2004.
- [2] R. Plamondon, S. N. Srihari, "On-line and off-line handwriting recognition: A comprehensive survey", IEEE Tran. Pattern Anal. Mach. Intell., vol. 22, pp. 63 84, 2000.
- [3] C. Y. Suen, R. Legault, C. Nadal, M. Cheriet, L. Lam, "Building a new generation of handwriting recognition system", Pattern Recognition Letters, vol. 14, pp. 305-315, 1993.
- [4] C.J.C. Burges, J. I. Be, C. R. Nohl, "Recognition of handwritten cursive postal words using neural networks", Proc. of the 5<sup>th</sup> USPS Advanced Tech. Conf., pp. 117 124, 1992.
- [5] F. Kimura, S. Tsuruoka, M. Shridhar, Z. Chen, "Context-directed handwritten word recognition for postal service applications", Proc. of the 5<sup>th</sup> USPS Advanced Tech. Conf., pp. 199 213, 1992.
- [6] T. Paquet, Y. Lecourtier, "Handwritten recognition: Application on bank cheques", Proc. of the first Int. Conf. on Doc. Anal. and Rec., pp. 749 757, 1991.
- [7] D. Guillevic, C. Y. Suen, "HMM-KNN word recognition engine for bank cheque processing", Proc. of the 14<sup>th</sup> Int. Conf. on Pattern Rec., pp. 1526 1529, 1998.
- [8] T. Bruel, "Design and implementation of a system for recognition of handwritten responses on US census form", Proc. of the IAPR Workshop on Doc. Anal. Sys., pp. 237 264, 1994.
- [9] F. Kimura, S. Tsuruoka, Y. Miyake, M. Srihari, "A lexicon directed algorithm for recognition of unconstrained handwritten words", IEICE Trans. Information Sys., E77-D, pp. 785 793, 1994.
- [10] P. D. Gader, M. Magdi, J. H. Chiang, "Segmentation based handwritten word recognition", Proc. of the 5<sup>th</sup> USPS Advanced Tech. Conf., pp. 215 – 226, 1992.

- [11] P. D. Gader, M. Mohamed, J. H. Chiang, "Handwritten word recognition with character and intercharacter neural networks", IEEE Trans. on Sys. Man and Cybernetics – part B: Cybernetics, vol.27, pp. 158 – 164, 1997.
- [12] G. Kim, V. Govindaraju, "A lexicon driven approach to handwritten word recognition for real time applications", IEEE Trans. Pattern Anal. Mach. Intell., vol. 19, pp. 366 – 379, 1997.
- [13] J. Cai, Z. Q. Liu, "Integration of structural and statistical information for unconstrained handwritten numeral recognition", IEEE Trans. Pattern Anal. Mach. Intell., vol. 21, pp. 263 270, 1999.
- [14] S. W. Lee, "Off-line recognition of totally unconstrained handwritten numerals using multilayer cluster neural network", IEEE Trans. Pattern Anal. Mach. Intell., vol. 18, pp. 648 652, 1996.
- [15] S. B. Cho, "Neural network classifiers for recognizing totally unconstrained handwritten numerals", IEEE Trans. of Neural Networks, vol.8, pp. 43 – 53, 1997.
- [16] X. Wang, X. Ding and C. Liu, "Gabor filters-based feature extraction for character recognition", Pattern Recognition, vol. 38, pp. 369-379, 2005.
- [17] H. Yamada, Y. Nakano, "Cursive handwritten word recognition using multiple segmentation determined by contour analysis", IEICE Trans. Information Sys., E79-D, pp. 464 470, 1996.
- [18] F. Kimura, N. Kayahara, Y. Miyake, M. Shridhar, "Machine and human recognition of segmented characters from handwritten words", Proc. of 4<sup>th</sup> Int. Conf. on Doc. Anal. and Rec., pp. 866 – 869, 1997.
- [19] S. Singh, M. Hewitt, "cursive digit and character recognition on cedar database", Int. Conf. on Pattern Rec., pp. 569 – 572, 2000.
- [20] B. Lazzerini, F. Marcelloni, "A linguistic fuzzy recognizer of off-line handwritten characters", Pattern Recognition Letters, vol.21, pp. 319 327, 2000.
- [21] F. Camastra, A. Vinciarelli, "Cursive character recognition by learning vector quantization", Pattern Recognition Letters, vol.22, pp. 625 629, 2001.
- [22] N. Arica and F.T. Yarman-Vural, "Optical Character Recognition for Cursive Handwriting", IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, pp. 801-813, 2002.
- [23] F. Camastra, A. Vinciarelli, "Combining neural gas and learning vector quantization for cursive character recognition", Neurocomputing, vol.51, pp. 147-159, 2003.
- [24] M. Hanmandlu, K. R. M. Murali, S. Chakraborty, S. Goyal, D. R. Choudhury, "Unconstrained handwritten character recognition based one fuzzy logic", Pattern Recognition, vol.36, pp. 603 – 623, 2003.
- [25] M. Blumenstein, B. K. Verma, H. Basli, "A Novel Feature Extraction Technique for the Recognition of Segmented Handwritten Characters", 7<sup>th</sup> Int. Conf. on Doc. Anal. and Rec., pp. 137-141, 2003.
- [26] M. Blumenstein, B. K. Verma, "Analysis of segmentation performance on the CEDAR bench mark database", Proc. of 6<sup>th</sup> Int. Conf. on Doc. Anal. and Rec., pp. 1142 – 1146, 2001.

- [27] J.J. Hull, "A database for handwritten text recognition", IEEE Trans. Pattern Anal. Mach. Intell., vol.16, pp. 550 554, 1994.
- [28] M. Blumenstein, X. Y. Liu and B. Verma, "A Modified Direction Feature for Cursive Character Recognition", Int. Joint Conf. on Neural Networks (IJCNN '04), Budapest, Hungary, pp. 2983-2987, 2004.