Segment Confidence-based Binary Segmentation (SCBS) for Cursive Handwritten Words

Brijesh Verma and Hong Lee
School of Computing Science, CQUniversity
North Rockhampton, Queensland 4702, Australia
b.verma@cqu.edu.au

Abstract – A novel Segment Confidence-based Binary Segmentation (SCBS) for cursive handwritten words is presented in this paper. SCBS is a character segmentation strategy for off-line cursive handwriting recognition. Unlike the approaches in the literature, SCBS is an unordered segmentation approach. SCBS is repetition of binary segmentation and fusion of segment confidence. Each repetition generates only one final segmentation point. The Binary Segmentation module is a contour tracing algorithm to find a segmentation path to divide a segment into two segments. A set of segments before binary segmentation is called pre-segments, and a set of segments after binary segmentation is called post-segments. SCBS uses over-segmentation technique to generate suspicious segmentation points on pre-segments. On each suspicious segmentation point, binary segmentation is performed and the highest fusion value is recorded. If the highest fusion value is greater than the one of pre-segments, the suspicious segmentation point becomes the final segmentation point for the iteration. If not, no more segmentation is required. Segment confidence is obtained by fusing mean character, lexical and shape confidences. The proposed approach has been evaluated on local and benchmark (CEDAR) databases.

1 Introduction

Off-line Cursive Handwriting Recognition (OffCHR) is an automatic process to convert an input handwritten document image into computer-recognizable character representations. *OffCHR* has been active research domain for decades, and industrial beneficiaries have been trying to automate repetitive manpower oriented tasks such as processing postal address, bank checks, form data, historical manuscripts, etc [1]. Despite sleepless research in *OffCHR* for decades, the performance of the state-of-the-art *OffCHR* is below the industrial standard to accommodate the real world problems [2-6]. The researchers in this field agree that the main contributor of the low *OffCHR* performance is the segmentation [7-15].

Segmentation is a process to discriminate each letter from others, prior to recognition into electronic character representations. Typically, *OffCHR* involves a set of processes such as pre-processing, normalization, segmentation, recognition. Pre-processing is a cleanup process to remove unwanted information [16-18]. Noise removal is done in the pre-processing stage. Normalization is to standardize the information, so it can be fitted into a data form that segmentation and recognition need. Normally, thresholding or skeletonization, thinning, slant and slope corrections are performed in normalization process. The normalized handwritten image passes through segmentation process to find letter boundaries. A sub-image bound by two neighboring boundaries is called a segment. The recognition is to classify each segment into a character representation [19-21]. As seen in the typical *OffCHR* framework in Figure 1, the segmentation precedes the recognition. In other words, the recognition process is based on the outcomes of the segmentation process. It implies that better recognition performance can be achieved on better segmentation outcomes. However, the segmentation is a very difficult process because of the nature of cursive handwriting and it has become a major error contributor in *OffCHR* [22-27].

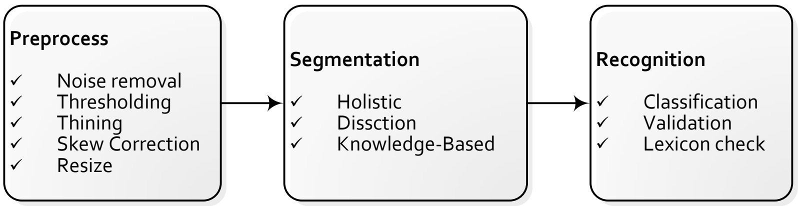


Figure 1. Typical frameworks of Off-line Cursive Handwriting Recognition (*OffCHR*)

Therefore, the aim of the research presented in this paper is to investigate a novel segmentation algorithm which can avoid problems of existing algorithms and improve the segmentation accuracy. The proposed segmentation algorithm is based on novel ideas such as binary segmentation and use of multiple confidence values.

The rest of this paper is organized into 5 sections. Section 2 presents a review of existing literature. The proposed algorithm is described in Section 3. Section 4 presents the experimental results. An analysis of experimental results and a discussion are presented in Section 5. Finally, Section 6 concludes the paper.

2 Review of segmentation in handwriting recognition

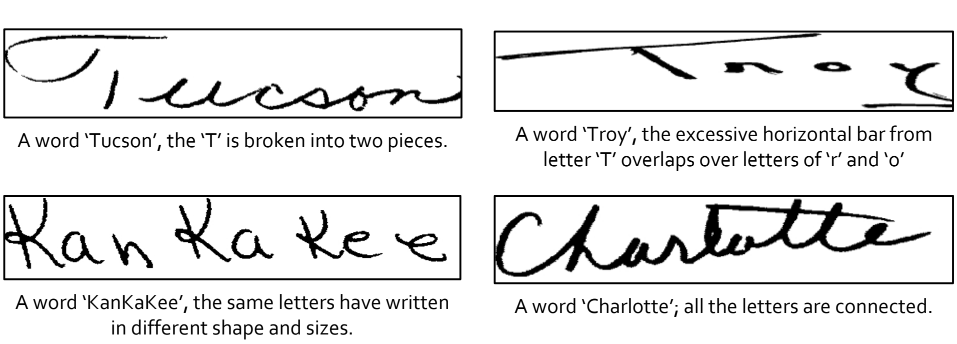
The review of handwriting recognition techniques focused on the segmentation, is presented in this section.

2.1 Representation of segmentation

Traditionally, segmentation was manifested into x-coordinates of input images. Especially in machine printed OCR, segmentation was only a matter of finding zero foreground pixel from vertical histogram of the recognizing image. However, in cursive handwriting recognition, there is no guarantee that neighboring characters will be separated by empty space. Therefore, new representation of segmentation must be introduced. Segmentation path is another technique to represent segmentation between two characters. Segmentation path is a connected list of x-y coordinates to represent a boundary between two neighboring characters. It is virtually impractical to separate two handwritten characters using a vertical line. However, segmentation paths are simply used to define the character boundaries [11,22,25,27].

2.2 The relation between handwriting recognition and segmentation

Researchers often describe relationship between segmentation and recognition processes in *OffCHR*, as chicken and egg relationship. It is arguable which one comes first. Similarly, segmentation cannot be completed until it is correctly recognized. On the other hand, recognition cannot be done without segmenting the whole word image into individual characters [11]. Segmentation is very difficult process in *OffCHR*, and it is one of the main factors for low accuracy. Researchers have found the contributors to make segmentation very hard. The major contributors are shape variability, connectivity, overlapping and brokenness. Some examples of difficult words are shown below in Figure 2.



A word 'Tucson', the 'T' is broken into two pieces.

A word 'Troy', the excessive horizontal bar from letter 'T' overlaps over letters of 'r' and 'o'

A word 'KanKaKee', the same letters have written in different shape and sizes.

A word 'Charlotte'; all the letters are connected.

Figure 2. Handwritten words showing characters for brokenness, overlapping, variability and connectivity.

It is difficult to write characters in exact shapes by the same person. It would be even more difficult to write characters in exact shapes by two or more people. Handwriting projects unique characteristics of the writers. Uniqueness means variability of handwritten character shapes [5]. Connected characters make segmentation the most difficult process in *OffCHR* because it makes difficult to know how many characters to be recognized. OCR in machine printed character recognition has been successful because segmentation in OCR is very easy. It is said easy because segmentation can be done by locating empty spaces between the characters. However, in *OffCHR*, majority of handwritten words are connected and letters can be overlapped. Unlike connected characters, the overlapping factor can be separated by using path finding algorithms. However, unlike vertical line separation, path finding algorithms involve navigating algorithms that come with greater computational costs. Unlike machine printed characters, in *OffCHR* there are often broken characters. The broken characters cause de-segmentation problem. De-segmentation is a process to recognize the broken characters and combine them and recognize them as one character. It is very hard process to find out whether a character is broken or not. There has been some research conducted and published on spotting broken characters [7,8,28].

2.3 Character segmentation methodologies for cursive handwriting recognition

Many researchers have been tackling the segmentation of handwritten image using various approaches. The segmentation techniques used in the literature can be grouped into 3 groups such as holistic, dissection and knowledge-based.

Holistic approach is also called segmentation-free. In holistic approach, global features of handwritten word image are used against the list of words under consideration for recognition. As the name 'holistic' implies, recognizing individual characters is ignored. The recognition accuracy is generally linear to the size of list of words under consideration, which is called, lexicon. So, the holistic approach is appropriate for handwriting recognition problems with small lexicon domain, such as bank checks processing [29-36].

Dissection is a segmentation technique to find boundaries between neighboring characters without involving knowledge about characters. One of the typical dissection approaches is using vertical histogram of foreground pixels of the input image. The vertical histogram based approach was eminent technique in OCR. However, the dissection technique is no longer eligible to handle irregular handwritten images [37].

Neither holistic nor dissection technique uses knowledge about characters during segmentation process. However, researchers have incorporated the idea of using classifiers equipped with character knowledge to cope with the irregularities of handwriting nature. In this type of knowledge based segmentation approach, there are two mainstream techniques. One is to allocate segmentation point where the employed classifier recognizes up to. Typically, a pair of sliding window and classifiers is implemented together. In the technique, a fixed-size sliding window [38-41] scans a handwritten word image from left to right. While scanning, the classifier confirms if the sub-image is recognizable as a legal character or not.

The other method combines the idea of dissection technique and the classifiers in the segmentation process. In this technique, the input image is dissected into many sub-images based on rules and heuristics, namely over-segmentation. The preliminary dissection is to locate all the possible segmentation points. Because of the idea to find all the possible letter boundaries, there might be excessive segmentation points. This technique is called, over-segmentation. The following process is to remove the excessive segmentation points, and the process is called, validation. The primary objective of validation is to remove excessive segmentation points by incorporating classifiers. This technique anticipates the over-segmenter and validator, and that's the reason it is called 'Hybrid' technique. The hybrid tends to find all the letter boundaries. However, it is still unaccomplished problem to remove all the excessive segmentation points and to keep the correct ones [10,24,42-44].

As discussed so far, holistic is plausible for only small lexicon domains such as bank check legal amounts. When it comes to the real word problem with large lexicon, the holistic method shows very little success. It is also hard to draw universal heuristics to find boundaries without knowledge because the nature of informal handwriting. Past research shows that there was little success using a dissection technique. However, knowledge-based segmentation can improve the performance if the accurate classifiers are employed. According to the past results, knowledge-based recognition techniques outperform the dissection technique, and that's why the method is continuously pursued by researchers. There are two types of knowledge-based segmentation techniques. The first one is to put the character boundaries based on recognition using sliding window technique. The second one is hybrid technique based on over-segmentation and validation. Over-segmentation is to put boundaries wherever doubtful, and validation is to remove the excessive segmentation boundaries. Over-segmentation tends to find all letter boundaries. However, there is little success in validating the excessive segmentation points. Therefore, by improving the validation accuracy, the overall segmentation accuracy can be improved.

2.4 Segmentation techniques

Tripathy et al. [45] incorporated water reservoir approach to segment the connected characters in Oriya text recognition. A water reservoir is a region formed by connected components and the region could retain water as if water were poured into. Water reservoir technique is to detect and segment connected regions based on touching position, reservoir base-area points, topological and structural features. Their approach has been experimented on 1840 images of Oriya scripts. The segmentation accuracy was 96.7% on two-character touching images (1458), 95.1% on three-character touching images (311), and 93.3% on four or more character touching images (71). However, the water reservoir approach is for single connection between characters, and they did not address the issues of multi connection between characters. Pal et al. [46] also used water reservoir approach to segment touching numeral digits, and experimented on French bank checks. The segmentation accuracy was 94.8%.

Zhao et al. [47] proposed background thinning segmentation algorithm to segment connected Chinese characters. The background thinning generates feature points such as end points, fork points and corner points. Sub-strokes are the segments between feature points and extracted. The connected points are located by identifying the lengths of sub-strokes and the topological relationship between sub-strokes. Alhajj et al. [48] proposed multi-agents to segment handwritten connected digits. Their strategy is to detect the deepest-top valley and the highest-bottom hill by dedicated agents. Each agent is responsible for nominating segmentation points where connected area, and the final segmentation points are compromised by the degree of the confidence assessed by the agents. This approach was experimented on 4095 images written by 150 writers, and obtained 97.8% segmentation accuracy. This approach is targeting to segment only two touching digits, so it is inappropriate to apply to problems involving multiple characters.

Liang et al. [49] proposed a meta synthetic approach to segment handwritten Chinese character strings. They applied Viterbi algorithm to search linear segmentation paths, and the redundant paths are eliminated by heuristics. Non-linear segmentation paths are obtained by background thinning algorithms. Especially, touching characters are further investigated with foreground and background information. The final segmentation paths are decided by mixture probabilistic density function. Their approach was experimented on 921 Chinese character strings and achieved 87.6% segmentation accuracy. However, their experiment seems biased because their database may contain many linearly separable images shown in the examples.

Dawoud [50] proposed the iterative cross section sequence graph (ICSSG) for handwritten character segmentation. ICSSG is a binarization technique of grey scale image, and the result of the binarization is the segmentation of connected characters. ICSSG is based on the idea that the stroke thickness of the connected points between characters is greater than the average stroke thickness. This method was experimented on 2575 numeral characters from bank checks, and obtained 76.9% recognition accuracy. However, this algorithm would fail where the characters are connected in a line.

Renaudin et al. [51] proposed over-segmentation and graph construction technique to segment touching digits. Over-segmentation points were located on singular area. Singular areas are where the stroke is disrupted such as intersections, high curvature, thickness variations, etc. Graph was constructed based on over-segmented primitives, and the final segmentation points were found by searching the best path on the graph. Their approach was experimented on touching two-digit images, and produced 68.9% of correct segmentation and recognition. General idea of their method is over-segmentation and best-path searching. Their method was only tested touching two-digit examples. The searching time and complexity will rise when more digits are involved and they are connected. Suwa [52] proposed graph representation technique to segment multiply connected digits. In their approach, the binary patterns are thinned and the edges and vertices are extracted. The patterns are represented as a connected graph. Graph theory and heuristic rules calculate the candidate segmentation path. Also rules are incorporated to eliminate the ligatures and the touching strokes are uniformed by digit boundary detection. The approach was experimented on 2000 pairs of touching digits from NIST-19 database. The segmentation accuracy was reported as 88.4%. They should expand the testing database from two-digits to multiple digits. The real world examples are more likely multiple character strings.

3 Segment Confidence-based Binary Segmentation (SCBS)

The proposed approach, SCBS, is repetitive process of fusion and segmentation of handwritten word images based on a set of suspicious segmentation points (SSPs). The details of SCBS are described in following subsections.

3.1 Overview

Overall system architecture of the proposed approach is presented in Figure 3.

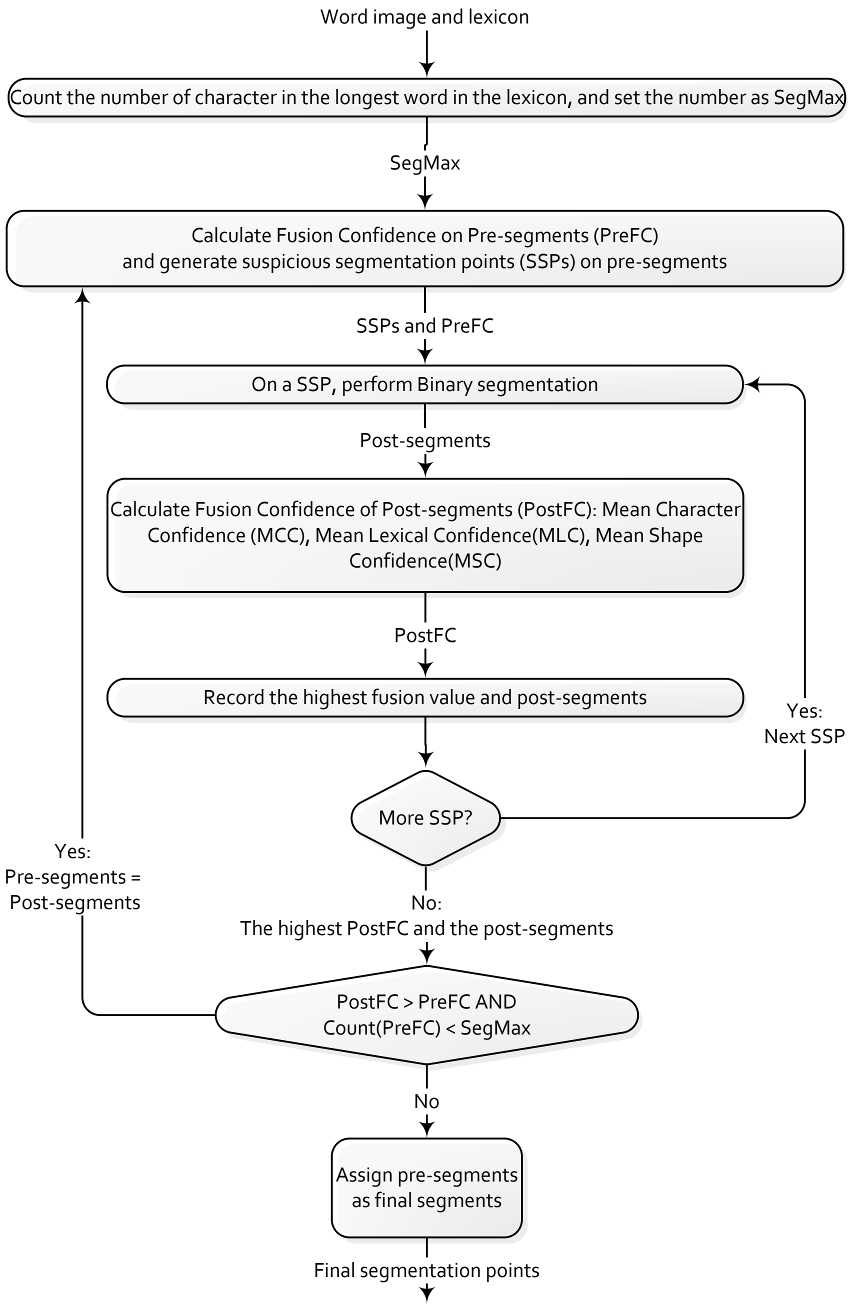

Word image and lexicon
Count the number of character in the longest word in the lexicon, and set the number as SegMax
SegMax
Calculate Fusion Confidence on Pre-segments (PreFC) and generate suspicious segmentation points (SSPs) on pre-segments
SSPs and PreFC
On a SSP, perform Binary segmentation
Post-segments
Calculate Fusion Confidence of Post-segments (PostFC): Mean Character Confidence (MCC), Mean Lexical Confidence(MLC), Mean Shape Confidence(MSC)
PostFC
Record the highest fusion value and post-segments
Yes: Next SSP
More SSP?
Yes: Pre-segments = Post-segments
No: The highest PostFC and the post-segments
PostFC > PreFC AND Count(PreFC) < SegMax
No
Assign pre-segments as final segments
Final segmentation points

Figure 3. Overview of Segment Confidence-based Binary Segmentation (SCBS)

3.2 SegMax variable

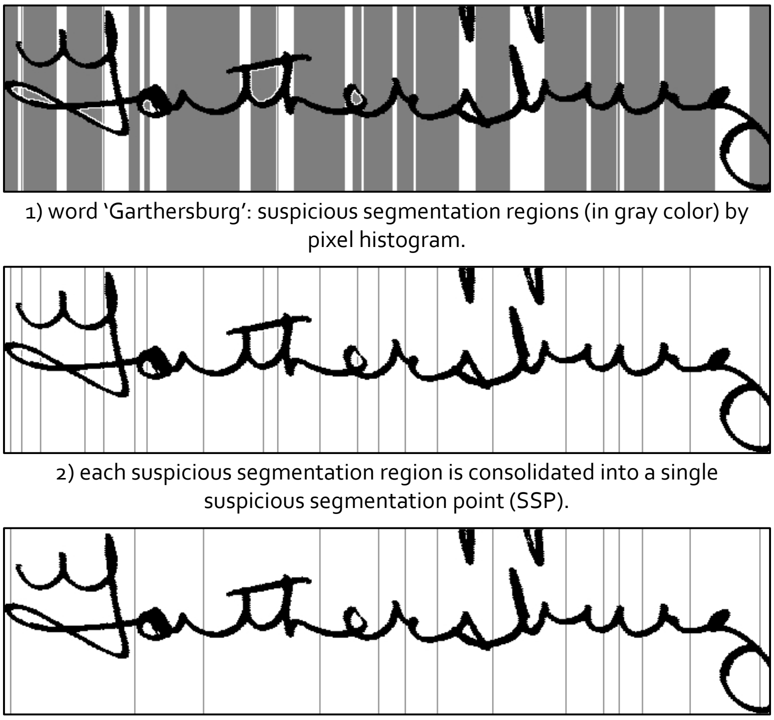
SegMax is a variable to limit the maximum number of segments. The ultimate goals of OffCHR are to recognize the input word image, not the individual characters. Character recognition is essential for word recognition. For the word recognition, most of OffCHR uses lexicon. Lexicon plays a role as a dictionary for the recognition domain. It provides an important clue that the maximum sub-images (segments) for a word image should be no more than the number of characters of the longest word in the lexicon. Since each segment represents a character, the maximum number of segments should be equal to the number of characters in the longest word from considering lexicon. In our proposal, the variable SegMax has been set to the number of characters of the longest word in lexicon.

3.3 Generating Suspicious Segmentation Points (SSPs)

The core idea of introducing over-segmentation into OffCHR is not to miss any letter boundaries. So, successful over-segmentation generates a segmentation set containing all letter boundaries regardless of existence of excessive segmentation points, which are called over-segmentation points. The best way to increase the chances to have successful over-segmentation is to locate as many segmentation points as possible. Often many rules and heuristics are applied to achieve successful over-segmentation. Every segmentation point from over-segmentation points can be a correct segmentation point, so it is called Suspicious Segmentation Point (SSP).

In the proposed approach, the SSPs are generated by using vertical foreground pixel density and stroke thickness variable. The stroke thickness is the most occurring continuous foreground pixel count. It is measured by scanning the segmenting word vertically and horizontally. While scanning, the occurrences are recorded and the most occurring continuous foreground pixel count becomes the stroke thickness of the segmenting word. The details of the stroke thickness measurement are described in [44]. Once the stroke thickness is estimated, the SSPs are located where the vertical foreground pixel density is less than the stroke thickness. However, to increase the chance of locating the correct boundaries, the SSPs are located where the vertical foreground pixel density is less than three times of the stroke thickness. The continuous SSPs are consolidated as a single SSP by finding the one in the middle.

The suspicious segmentation points are screened by hole detection module to remove the ones crossing hole regions. The reason to incorporate this screening process is two-fold. Firstly, reducing the number of SSP cuts down the computational costs significantly. The computational cost of validation for a SSP is much cheaper than validation by classifier in the later stage. The second reason is to reduce the number of segments. A segment is a sub-image defined by two neighboring segmentation points. The lesser segments, the lesser spatial segment combinations for classifier have to validate. An example of SSP generating process is described in Figure 4.



1) word 'Garthersburg': suspicious segmentation regions (in gray color) by pixel histogram.

2) each suspicious segmentation region is consolidated into a single suspicious segmentation point (SSP).

3) Hole detection removes SSPs crossing hole region, and remaining SSPs become the final set of SSPs.

Figure 4. An example of generating a set of Suspicious Segmentation Points (SSP) from the word, 'Garthersburg'.

3.4 Calculating Fusion of Segment Confidence (FSC)

The result of binary segmentation is a set of segments. A set of segments before the binary segmentation is defined as pre-segments. A set of segments after the binary segmentation is defined as post-segments. Therefore, the number of post-segments is always one bigger than the one of pre-segments. To calculate FSC, the three types of confidence values are estimated individually to a set of segments and they are fused together by applying pre-set weight factors. The three types of confidences are Mean Character Confidence (MCC), Mean Lexical Confidence (MLC) and Mean Shape Confidence (MSC). Let W_L, W_C, W_S be the weight factors for lexical, character and shape confidences accordingly. In the proposed approach, the weight factors were set as W_L = 0.4, W_C = 0.35 and W_S = 0.25. The final FSC was calculated in the following equation. How to estimate MLC, MCC and MSC is described in the following subsections.

$$FSC = MLC * W_L + MCC * W_C + MSC * W_S$$

3.4.1 Mean Character Confidence (MCC)

A character confidence is measured by using the output from the neural network based classifier, which is pre-trained on correctly segmented characters. The classifier produces 52 confidence values (26 lowercases and 26 uppercases of English alphabets) for each segment. The decision of the classification is made by finding the highest confidence value out of 52. Therefore, the mean character confidence for a set of segments is found by dividing the sum of individual mean character confidence (MCC) with the number of segments.

Let *S* be a set of segments, n be the number of segments in S and *Top(S$_i$)* be the highest confidence value of the i_{th} segment in *S*:

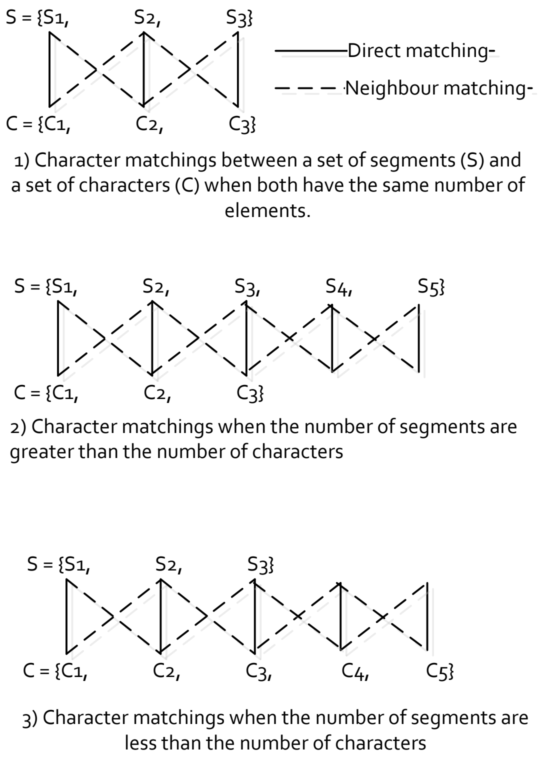
$$MCC = \frac{\sum_i^n Top(S_i)}{n}$$

3.4.2 Mean Lexical Confidence (MLC)

A character matching confidence score between a segment and a character is defined as finding a corresponding confidence value from the segment's neural outputs for the character. For example, the character matching confidence score for character 'a' is the first neural confidence value for a segment, and the 27th neural confidence value for character 'A'. In the following equation, the neural output O(S) for segment S is described by subscripting the characters for their corresponding values. In the equation, the character matching confidence scores between a segment and a character are as follows: a=O_a, b=O_b, $_c$=O_c … A=O_A, B=O_B, C=O_C … Z=O_Z.

$$O(S) = \{O_a, O_b, O_c \dots\ O_z, O_A, O_B, O_C, \dots O_Z\}$$

A word matching confidence score between segments and characters (a lexical word) are calculated by dividing the total character matching score with the number of character matching comparison. In the proposed approach, there are two types of character matching algorithms such as direct matching and neighbor matching. As shown in Figure 5, the direct matching is a character matching where the segment index in the segments and the character index in a word are the same. The neighbor matching is a character matching between S_i in a set of segments *S* and $C_{(i-1)}$ in a word *C*, or S_i and $C_{(i+1)}$.



1) Character matchings between a set of segments (S) and a set of characters (C) when both have the same number of elements.

2) Character matchings when the number of segments are greater than the number of characters

3) Character matchings when the number of segments are less than the number of characters

Figure 5. Character matching: 1) performs total of 7 character matching (3 directs + 4 neighbors), 2) and 3) total of 13 matching (5 directs + 5 neighbors)

Let S be a set of segments, C be a set of characters in a word, M(S$_i$, C$_i$) be the matching score between i_{th} segment and i_{th} character. The total number of matching performed would be calculated by the number of elements in S as 'Q' multiplied by 3, minus 2 because the first and the last elements performed one less neighbor matching.

$$MLC = \frac{\sum_i^Q \llbracket (M(S_i, C_i) + M(S_i, C_{i-1}) + M(S_i, C_{i-1}) \rrbracket)}{Q * \mathbf{3} - \mathbf{2}}$$

3.4.3 Mean Shape Confidence (MSC)

Shape Confidence (SC) is to measure how well each segment fits to the ideal character shape. The ideal shape is universally defined as related to the height and the width of a segment. The ideal character shape satisfies the fact that the difference between the height and the width of a segment is very close to zero.

Let *h* and *w* be the height and width of a segment, and the SC in the proposed methodologies are calculated by the following equation:

$$SC = 1 - \left(\frac{w - h}{w + h}\right)^2$$

Therefore, the Mean Shape Confidence (MSC) is the sum of SC for all segments, and divided by the number of segments. Let *S* be a set of segments, n be the number of segments in *S*, and *SC(S$_i$)* be the function to measure the shape confidence of *i*$_{\text{th}}$ element in *S*. MSC is estimated by the following equations:

$$MSC = \frac{\sum_i^n SC(S_i)}{n}$$

Implementing MSC has two advantages. The first is to give higher segmentation priority to the wider segment, which is likely to contain more characters. The second is that MSC becomes the driving force for segmentation to be performed when the Mean Character Confidence and Lexical Confidence are lower than threshold.

3.4.4 Binary Segmentation Algorithm (BSA)

The core idea of Binary Segmentation Algorithm (BSA) is to split an image/sub-image into two sub-images on a given Suspicious Segmentation Point. BSA is applied to the connected components or characters. Contour tracing algorithm is already introduced to segment non-connected components. However, the contour tracing will not work on connected components because there is no path through from lower bound to upper bound. BSA is devised to work similar way as the contour tracing, but it can find a path through foreground pixels to make a path, where SSP lies.

As shown in Figure 6, the tracing starts from a random tracing start point on the lower bound. A random tracing start point is a randomly picked background pixel on lower bound. The tracing continues recursively through all the neighboring background pixels until there are no more neighboring background pixels to be navigated. While navigating, the encountered coordinates of the foreground pixels lying on SSP are recorded. In the recorded coordinates, the tracing algorithm takes one with smallest y-coordinate value, and continues navigating towards upper bound through neighboring foreground pixels on SSP until it reaches an untraced background pixel. The tracing ends when a pixel on upper bound is reached. Until then, tracing through background pixels and tunneling through foreground pixels on SSP are repeated. In Figure 6, the characters of ‘i’ and ‘g’ are connected. Since they are connected, BSA must be used to dissect them. There are two foreground crossings in SSP. However, the foreground crossing 2 should not be crossed since the tracer can make to upper bound without crossing 2.

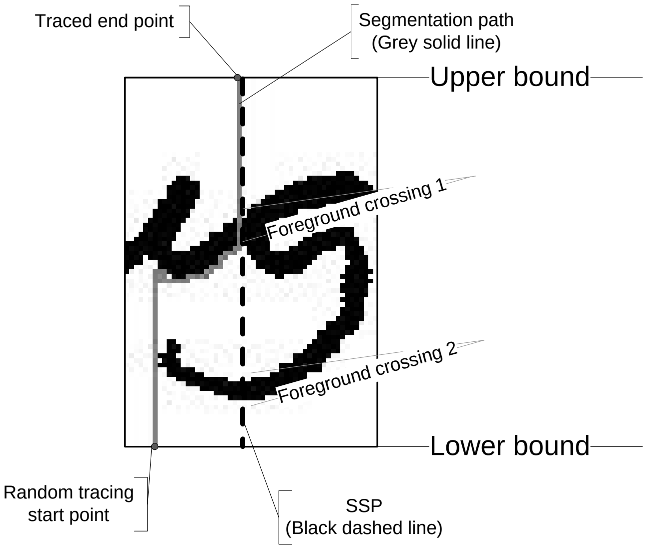


Figure 6. An example of Binary Segmentation Algorithm (BSA)

3.4.5 Termination of SCBS

As mentioned earlier, the SCBS is an iterative algorithm, which repeats cycles of Binary Segmentation and evaluation of Fusion of Segment Confidence. In the proposed approach, there are two terminating conditions. The first condition concerns the SegMax variable. SegMax variable defines the number of segments. The total number of segments should not exceed the SegMax.

The other condition is the improvement factor. The pre-segments are the current set of segments before the Binary Segmentation is applied. The post-segments are defined as a result set of Binary Segmentation on a set of pre-segments. The fusion of segment confidence is estimated on pre-segments ad post-segments. If fusion of segment confidence on post-segments is greater than the one on pre-segments, then improvement has been made. Otherwise, no improvement can be made on any SSP. Therefore SCBS terminates.

3.4.6 SCBS in Steps

In previous sections, sub-processes for SCBS are discussed. In this section, stepwise algorithm for SCBS is presented.

Step 1. Input cursive handwritten word image.

Step 2. Estimate parameter.

Step 3. Generate Suspicious Segmentation Points (SSPs).

Step 4. Use binary segmentation for each SSP, calculate FSC and record the highest confidence value.

Step 5. Check if terminating conditions are met

Step 6. If not terminating, the post-segments become the pre-segments. Go to step 3. If terminating the post –segments become the final segmentation points.

4 Experimental results

4.1 Implementation

The proposed approach has been implemented in Java programming language and many experiments were conducted.

4.2 Database preparation

Two sets of experiments were conducted on a local database and CEDAR benchmark database to check the effectiveness of the proposed approach. The local database was created by our group, which has been obtained from multiple writers. The CEDAR benchmark database was taken from CEDAR\TEST\CITIES\BD directory.

4.3 Neural networks training

A MLP neural network with a single hidden layer was trained on pre-segmented characters with back-propagation learning algorithm. It takes 100 inputs, and produces 52 outputs. The 52 outputs represent 52 alphabets (upper and lower cases). The number of hidden units and the number of iteration were varied during the training. The number of hidden units with the best training result was used in the experiment.

4.4 Segmentation performance criteria

As described in [27], the numbers of over-segmentation, under-segmentation, and bad-segmentation points are counted by manual inspection. The over-segmentation is defined as a character segmented into more than three segments. Under-segmentation points are the missing segmentation points between two neighbouring characters. Finally, the bad-segmentation is the rest of inappropriate cuts that don't belong to under-segmentation and over-segmentation and don't separate two characters correctly. The final segmentation results are calculated by dividing each categorical result with total number of characters used in the experiment.

TABLE 1. SEGMENTATION PERFORMANCE RESULTS

Database	Size		Segmentation rate (%)			
	word	Character	Under	Over	Bad	Average
Local	293	1215	7.82	0.74	3.13	3.90
CEDAR	161	973	10.79	0.31	2.88	4.66

5 Analysis and discussion

As shown in Table 1, there are two experimental results for local and CEDAR databases. As mentioned in previous section, the segmentation results are analyzed by the number of segmentation errors in each category. The segmentation error categories are over, under and bad segmentation errors. For the experiment results from the local database, the highest segmentation error was generated by the under segmentation, which recorded 7.82%. Over and bad segmentation errors were 0.74% and 3.13% accordingly. The average segmentation error for the experiment on the local database was 3.90%. The result from the experiment on CEDAR benchmark database shows that the highest error was generated from under segmentation error similar to the local database results. Over and bad segmentation errors were 0.31% and 2.88% accordingly. The average segmentation error for the CEDAR experiment was 4.66%.

The overall segmentation error was higher in CEDAR experiments than in local. However, the similar segmentation error pattern has been shown that the under segmentation errors were the highest in both experiments. Also, the bad segmentation errors were the next highest errors in both experiments. Finally, the over segmentation error was the lowest error category in both experiments.

Table 2 shows the segmentation error characters and their contribution percentage in each segmentation error category for both experiments. Overall, the under segmentation errors were the highest in each category. For the local database, the most under segmented character pairs were 8.79 for ht, 5.02 for an, 4.60 for fo, etc. The most over segmented characters were w(33.33) c(22.22) f(11.11) g(11.11) d(11.11) and o(11.11) in order. The worst segmented characters were n(26.53), h(22.45), w(20.41), m(6.12), s(6.12), f(4.08), e(2.04), c(2.04), a(2.04), i(2.04), u(2.04), r(2.04), and y(2.04) in order.

For CEDAR database, the most under segmented character pairs were ch(5.71) al(5.71) el(4.76) no(3.81) il(3.81) er(3.81), etc. The over segmented characters were w(33.33) e(33.33) and s(33.33). The bad segmented characters were m(25.00) e(14.29) r(14.29) u(10.71) n(10.71) a(7.14) w(3.57) d(3.57) b(3.57) c(3.57) and i(3.57).

TABLE 2. CHARACTERS OR CHARACTER PAIRS CAUSED SEGMENTATION ERRORS IN EACH SEGMENTATION ERROR CATEGORY FROM LOCAL AND CEDAR DATABASES

Category	Character/Character pair (Frequency in %) from local
Under	Ht(8.33) fo(7.29) ot(6.25) ir(6.25) it(6.25) in(6.25) ar(5.21) an(4.17) ll(3.12) ap(3.12) no(2.08) or(2.08) rs(2.08) et(2.08) er(2.08) ae(2.08) nu(1.04) ls(1.04) hi(1.04) ho(1.04) di(1.04) su(1.04) dn(1.04) sy(1.04) be(1.04) ew(1.04) is(1.04) ev(1.04) mo(1.04) iv(1.04) at(1.04) aw(1.04) io(1.04) eh(1.04) al(1.04) ai(1.04) el(1.04) tu(1.04) ek(1.04) tt(1.04) ci(1.04) ep(1.04) en(1.04) ch(1.04) es(1.04) ry(1.04)
Over	w(33.33) c(22.22) f(11.11) g(11.11) d(11.11) o(11.11)
Bad	n(28.95) w(26.32) h(23.68) m(5.26) e(2.63) s(2.63) r(2.63) c(2.63) a(2.63) y(2.63)
Category	Character/Character pair (Frequency in %) from CEDAR
Under	Ch(5.71) al(5.71) el(4.76) no(3.81) il(3.81) er(3.81) ot(2.86) or(2.86) ls(2.86) ir(2.86) ai(2.86) tt(2.86) cl(1.90) ce(1.90) lu(1.90) lt(1.90) di(1.90) dn(1.90) as(1.90) im(1.90) ek(1.90) ac(1.90) et(1.90) nt(0.95) ny(0.95) bo(0.95) st(0.95) ou(0.95) op(0.95) oo(0.95) ko(0.95) ow(0.95) oz(0.95) gn(0.95) co(0.95) ci(0.95) gs(0.95) go(0.95) lo(0.95) hs(0.95) lp(0.95) hi(0.95) dl(0.95) iv(0.95) ez(0.95) in(0.95) io(0.95) au(0.95) eg(0.95) tu(0.95) ap(0.95) ep(0.95) iy(0.95) ad(0.95) en(0.95) ag(0.95) ah(0.95) es(0.95) ae(0.95)
Over	w(33.33) e(33.33) s(33.33)
Bad	m(25.00) e(14.29) r(14.29) u(10.71) n(10.71) a(7.14) w(3.57) d(3.57) b(3.57) c(3.57) i(3.57)

Overall segmentation error contribution by character is shown in Table 3. The values in the table were generated by counting the occurrences in CEDAR result from Table 2. It describes that the letter ‘l’ contributed the most segmentation error of 10.83%. On the other hand, the least segmentation error was caused by the letter ‘x’ for 0.09%.

TABLE 3. OVERALL SEGMENTATION ERROR CONTRIBUTION RATIO (%) BY CHARACTER FROM CEDAR RESULT

A	**b**	**C**	**d**	**e**	**f**	**G**	**h**	**i**	**j**	**K**	**l**	**M**
8.89	1.3	4.44	2.96	8.43	1.85	1.94	3.61	8.52	0.37	1.11	10.83	1.76
N	**o**	**P**	**q**	**r**	**s**	**T**	**u**	**v**	**w**	**X**	**y**	**z**
5.46	6.9	2.04	0.09	6.57	5.56	9.44	3.98	1.3	0.74	0.09	1.67	0.19

The results obtained using the proposed approach are compared with the published results. However, it should be noted that it is very difficult to compare the results as many researchers use their own databases and many do not report the segmentation results. As shown in Table 4, the proposed approach using CEDAR database generated higher under segmentation errors. However, the results of over segmentation error were much better than [10,42] and very similar to [27]. Bad segmentation error also exhibits the similar comparison as the over segmentation error. The overall average segmentation performance has been improved over [10,42] but shows lower performance than [27]. It should be noted that the local database was used in [27]. The proposed approach heavily depends on the neural confidence value to calculate the mean lexical confidence and the mean character confidence. Those confidence values have higher weight factors to decide if a suspicious segmentation point is good or bad. In the proposed approach, the neural network classifier was trained and it has 61% accuracy. The segmentation can be improved by testing with more accurate neural classifier.

TABLE 4. SEGMENTATION PERFORMANCE COMPARISON

	Database		Segmentation rate (%)			
			Over	Under	Bad	Average
[10]	CEDAR	317	7.4	2.0	11.6	7.0
[27]	Local	750	0.1	0.8	2.3	1.06
[42]	CEDAR	317	10.0	0.2	8.7	6.3
SCBS	Local	293	0.74	7.82	3.13	3.90
	CEDAR	161	0.31	10.79	2.88	4.66

6 Conclusions and future research

In this paper, a novel Segment Confidence-based Binary Segmentation is proposed as a segmentation strategy for off-line cursive handwriting recognition. The proposed approach has been tested on local and CEDAR benchmark databases. The proposed approach uses over-segmentation technique to generate Suspicious Segmentation Points (SSP). Based on SSPs, binary segmentation was applied on each SSP and the segment confidence was evaluated to see if any improvement was made. The promising results were obtained on both databases. The results showed that the proposed approach produced higher under segmentation error than the results from the literature. Over and bad segmentation was moderate comparing to the literature, and the overall segmentation performance was also moderate. The proposed segmentation approach heavily depends on the output from the neural classifier. Therefore, the segmentation results can be improved by using more accurate neural classifier on CEDAR benchmark database. In our future research, we will focus on improving neural confidence and optimizing the weighting for fusion of multiple confidences.

References

[1] H. Fujisawa, "Forty years of research in character and document recognition--an industrial perspective," *Pattern Recognition,* vol. 41, Aug. 2008, pp. 2435-2446.

[2] L. Zhang, A.M. Yip, M.S. Brown, and C. Lim Tan, "A unified framework for document restoration using inpainting and shape-from-shading," *Pattern Recognition,* vol. In Press, Accepted Manuscript.

[3] R.M. Suresh and S. Arumugam, "Fuzzy technique based recognition of handwritten characters," *Image and Vision Computing,* vol. 25, 2007, pp. 230-239.

[4] S. Lu and C. Tan, "Retrieval of machine-printed Latin documents through Word Shape Coding," *Pattern Recognition,* vol. 41, 2008, pp. 1799-1809.

[5] M. Djioua and R. Plamondon, "Studying the variability of handwriting patterns using the Kinematic Theory," *Human Movement Science,* vol. In Press, Corrected Proof.

[6] M.T. Das and L.C. Dulger, "Signature verification (SV) toolbox: Application of PSO-NN," *Engineering Applications of Artificial Intelligence,* vol. In Press, Corrected Proof.

[7] N. Arica and F.T. Yarman-Vural, "An overview of character recognition focused on off-line handwriting," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on,* vol. 31, 2001, pp. 216-233.

[8] N. Arica and F.T. Yarman-Vural, “Optical Character Recognition for Cursive Handwriting,” *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 24, 2002, pp. 801-813.

[9] I. Bar-Yosef, A. Mokeichev, K. Kedem, I. Dinstein, and U. Ehrlich, “Adaptive shape prior for recognition and variational segmentation of degraded historical characters,” *Pattern Recognition*.

[10] M. Blumenstein and B. Verma, “Analysis of Segmentation Performance on the CEDAR Benchmark Database,” *Proceedings of the Sixth International Conference on Document Analysis and Recognition*, IEEE Computer Society, 2001, pp. 1142-1142.

[11] R.G. Casey and E. Lecolinet, “A Survey of Methods and Strategies in Character Segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 18, 1996, pp. 690-706.

[12] E. Vellasques, L.S. Oliveira, A.S. Britto Jr, A.L. Koerich, and R. Sabourin, “Filtering segmentation cuts for digit string recognition,” *Pattern Recognition,* vol. 41, 2008, pp. 3044-3053.

[13] E. Vellasques, L. Oliveira, A. Britto Jr., A. Koerich, and R. Sabourin, “Filtering segmentation cuts for digit string recognition,” *Pattern Recognition,* vol. 41, Oct. 2008, pp. 3044-3053.

[14] B. Verma, P. Gader, and W. Chen, “Fusion of multiple handwritten word recognition techniques,” *Pattern Recognition Letters,* vol. 22, 2001, pp. 991-998.

[15] S. Zhao, Z. Chi, P. Shi, and H. Yan, “Two-stage segmentation of unconstrained handwritten Chinese characters,” *Pattern Recognition,* vol. 36, 2003, pp. 145-156.

[16] Z. Lu, Z. Chi, W. Siu, and P. Shi, “A background-thinning-based approach for separating and recognizing connected handwritten digit strings,” *Pattern Recognition,* vol. 32, 1999, pp. 921-933.

[17] S. Nomura, K. Yamanaka, T. Shiose, H. Kawakami, and O. Katai, “Morphological Preprocessing Method to Thresholding Degraded Word Images,” *Pattern Recognition Letters,* vol. In Press, Accepted Manuscript.

[18] Y. Sun, T. Butler, A. Shafarenko, R. Adams, M. Loomes, and N. Davey, “Word segmentation of handwritten text using supervised classification techniques,” *Applied Soft Computing,* vol. 7, Jan. 2007, pp. 71-88.

[19] L.M. Lorigo and V. Govindaraju, “Offline Arabic handwriting recognition: a survey,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 28, 2006, pp. 712-724.

[20] R. Plamondon and S.N. Srihari, “Online and off-line handwriting recognition: a comprehensive survey,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 22, 2000, pp. 63-84.

[21] A. Vinciarelli, “A survey on off-line Cursive Word Recognition,” *Pattern Recognition,* vol. 35, 2002, pp. 1433-1446.

[22] A. Elnagar and R. Alhajj, “Segmentation of connected handwritten numeral strings,” *Pattern Recognition,* vol. 36, 2003, pp. 625-634.

[23] K. Hussein, A. Agarwal, A. Gupta, and P. Wang, “A knowledge-based segmentation algorithm for enhanced recognition of handwritten courtesy amounts,” *Pattern Recognition,* vol. 32, 1999, pp. 305-316.

[24] J. Sadri, C.Y. Suen, and T.D. Bui, “A genetic framework using contextual knowledge for segmentation and recognition of handwritten numeral strings,” *Pattern Recogn.,* vol. 40, 2007, pp. 898-919.

[25] X. Xiao and G. Leedham, “Knowledge-based English cursive script segmentation,” *Pattern Recognition Letters,* vol. 21, 2000, pp. 945-954.

[26] Q. Xu, L. Lam, and C. Suen, “Automatic Segmentation and Recognition System for Handwritten Dates on Canadian Bank Cheques,” *Proceedings of the Seventh International Conference on Document Analysis and Recognition,* 2003, pp. 704-708.

[27] B. Yanikoglu and P.A. Sandon, “Segmentation of off-line cursive handwriting using linear programming,” *Pattern Recognition,* vol. 31, 1998, pp. 1825-1833.

[28] C. Liu, K. Nakashima, H. Sako, and H. Fujisawa, “Handwritten digit recognition: benchmarking of state-of-the-art techniques,” *Pattern Recognition,* vol. 36, 2003, pp. 2271-2285.

[29] Z.A. Aghbari and S. Brook, “HAH manuscripts: A holistic paradigm for classifying and retrieving historical Arabic handwritten documents,” *Expert Systems with Applications*.

[30] A. Benouareth, A. Ennaji, and M. Sellami, “Semi-continuous HMMs with explicit state duration for unconstrained Arabic word modeling and recognition,” *Pattern Recognition Letters,* vol. 29, Sep. 2008, pp. 1742-1752.

[31] S. Gunter and H. Bunke, “HMM-based handwritten word recognition: on the optimization of the number of states, training iterations and Gaussian components,” *Pattern Recognition,* vol. 37, Oct. 2004, pp. 2069-2079.

[32] S. Lee and J. Kim, “Complementary combination of holistic and component analysis for recognition of low-resolution video character images,” *Pattern Recognition Letters,* vol. 29, Mar. 2008, pp. 383-391.

[33] J. Ruiz-Pinales, R. Jaime-Rivas, and M. Castro-Bleda, “Holistic cursive word recognition based on perceptual features,” *Pattern Recognition Letters*, vol. 28, 2007, pp. 1600-1609.

[34] S. Srihari, J. Collins, R. Srihari, H. Srinivasan, S. Shetty, and J. Brutt-Griffler, “Automatic scoring of short handwritten essays in reading comprehension tests,” *Artificial Intelligence*, vol. 172, Feb. 2008, pp. 300-324.

[35] T. Su, T. Zhang, D. Guan, and H. Huang, “Off-line recognition of realistic Chinese handwriting using segmentation-free strategy,” *Pattern Recognition*, vol. 42, Jan. 2009, pp. 167-182.

[36] X. Wang, V. Govindaraju, and S. Srihari, “Holistic recognition of handwritten character pairs,” *Pattern Recognition*, vol. 33, Dec. 2000, pp. 1967-1973.

[37] B. Verma and H. Lee, “A segmentation based adaptive approach for cursive handwritten text recognition,” *IEEE International Joint Conference on Neural Networks*, Orlando, Florida, USA: IEEE IJCNN'07, 2007, pp. 2212-2216.

[38] H. Al-Muhtaseb, S. Mahmoud, and R. Qahwaji, “Recognition of off-line printed Arabic text using Hidden Markov Models,” *Signal Processing*, vol. 88, Dec. 2008, pp. 2902-2912.

[39] S.M. Awaidah and S.A. Mahmoud, “A multiple feature/resolution scheme to Arabic (Indian) numerals recognition using hidden Markov models,” *Signal Processing*, vol. 89, Jun. 2009, pp. 1176-1184.

[40] A. Benouareth, A. Ennaji, and M. Sellami, “Semi-continuous HMMs with explicit state duration for unconstrained Arabic word modeling and recognition,” *Pattern Recognition Letters*, vol. 29, Sep. 2008, pp. 1742-1752.

[41] T. Su, T. Zhang, D. Guan, and H. Huang, “Off-line recognition of realistic Chinese handwriting using segmentation-free strategy,” *Pattern Recognition*, vol. 42, Jan. 2009, pp. 167-182.

[42] B. Verma, “A contour code feature based segmentation for handwriting recognition,” *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*, 2003, pp. 1203-1207.

[43] J. Chiang, “A hybrid neural network model in handwritten word recognition,” *Neural Networks*, vol. 11, Mar. 1998, pp. 337-346.

[44] H. Lee and B. Verma, “A novel multiple experts and fusion based segmentation algorithm for cursive handwriting recognition,” *Neural Networks, 2008. IJCNN 2008.*

(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, 2008, pp. 2994-2999.

[45] N. Tripathy and U. Pal, “Handwriting segmentation of unconstrained Oriya text,” *Sadhana*, vol. 31, Dec. 2006, pp. 755-769.

[46] U. Pal, A. BelaI[combining diaeresis above]d, and C. Choisy, “Touching numeral segmentation using water reservoir concept,” *Pattern Recognition Letters*, vol. 24, Jan. 2003, pp. 261-272.

[47] S. Zhao and P. Shi, “Segmentation of Connected Handwritten Chinese Characters Based on Stroke Analysis and Background Thinning,” *PRICAI 2000 Topics in Artificial Intelligence*, 2000, pp. 608-616.

[48] R. Alhajj and A. Elnagar, “Multiagents to Separating Handwritten Connected Digits,” *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 35, 2005, pp. 593-602.

[49] Z. Liang and P. Shi, “A metasynthetic approach for segmenting handwritten Chinese character strings,” *Pattern Recognition Letters*, vol. 26, Jul. 2005, pp. 1498-1511.

[50] A. Dawoud, “Iterative Cross Section Sequence Graph for Handwritten Character Segmentation,” *Image Processing, IEEE Transactions on*, vol. 16, 2007, pp. 2150-2154.

[51] C. Renaudin, Y. Ricquebourg, and J. Camillerapp, “A General Method of Segmentation-Recognition Collaboration Applied to Pairs of Touching and Overlapping Symbols,” *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, 2007, pp. 659-663.

[52] M. Suwa, “Segmentation of connected handwritten numerals by graph representation,” *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, 2005, vol. 2, pp. 750-754.