AMSE JOURNALS –2015-Series: Advances B; Vol. 58; N°1 ; pp1-13
Submitted Oct. 2014; Revised Feb.14, 2015; Accepted March 15, 2015

Extraction of English Alphabets from Words and Sentences using Slider Drifting Method (Sdm)

* R. K. Mandal, **N. R. Manna

*Department of Computer Science & Application, University of North Bengal, Siliguri, Distt : Darjeeling, West Bengal-734013, India (rakesh_it2002@yahoo.com)

** Department of Computer Science & Application, University of North Bengal, Siliguri, Distt : Darjeeling, West Bengal-734013, India (nrmanna12@gmail.com)

Abstract: This is the era of technological development throughout the world. Research is going on in almost all the domains to develop current technologies. Numerous researches are also carried out in the field of handwriting recognition using neural networks. To recognize a word or a sentence it is necessary to recognize each alphabet individually. It is a challenging task, because different individuals have different handwriting styles. The approach, in this paper is to extract out each and individual character by forming a rectangular border around the word or the sentence first and then drifting a slider from the beginning of the word or sentence towards the end to find out the alphabet delimiters and enclose the characters within the box.

Keywords: Segmentation, Slider, Drifting, Encaged, Handwriting, Characters

1. Introduction.

Many techniques are developed to recognize handwritten characters with the help of neural networks. But it is a challenging task to extract out individual characters from a string of characters. In order to recognize handwritten characters, it is very important to extract out single characters from the word/sentence. Idea is to form a box around the character with the boundaries touching the extreme pixels of the characters. Research has already been done in this field. One such work is done in recognizing Amazigh writing, where segmentation of characters

from a single line has been done using vertical histogram, [1]. Methods were also used to enclose the image into windows and cells in order to identify the image, [2]. Morphological analysis was also done in some research work in order to segment units of text. Smallest unit of characters having certain linguistic meaning in it is called morpheme, [3]. The approach in this paper is inspired by much other works that has already been done in the field of segmentation and contour tracing algorithms, [4]. A contour is an enclosure around the character. Basu S et al developed a segmentation method to extract out offline handwritten Bengali script. In this paper a hybrid model of image based dissection and recognition based segmentation is proposed, [5]. Hong C et al worked on segmentation and recognition of continuous handwriting Chinese text. This method performs basic segmentation and fine segmentation based on varying spacing thresholds and minimum variance criteria, [6]. Kurniawan F et al proposed a region based touched character segmentation method. In this approach Self Organizing Maps (SOM) are used to identify the touching portions of the cursive words, [7]. Kumar M et al worked on the segmentation of isolated and touching characters in offline handwritten Gurmukhi script recognition. In this method a technique called water reservoir based technique is applied for the identification and segmentation of touching characters, [8]. Nikolaou N et al worked on segmentation of historical machine-printed documents using adaptive run length smoothing and skeleton segmentation paths, [9]. Ramteke A S and Rane M E worked on offline handwritten Devanagari script segmentation. This method used connected component approach and vertical projection profile, which is the histogram of input image, where the zero valley peaks shows the space between the words and characters, [10]. Arica N and Yarman-Vural F T proposed a method for the recognition of cursive handwriting. In this paper an analytic scheme is proposed which uses a sequence of segmentation and recognition algorithms, [11].

Artifical Neural Networks (ANN) is a very good tool to recognize handwritten and printed characters, [12, 13]. The extracted out characters can be presented to ANN models. Al-Shridah et al worked on the recognition of handwritten and typed Arabic letters, [14]. N Liolios et al applied a new shape transformation approach to recognize handwritten characters, [15].

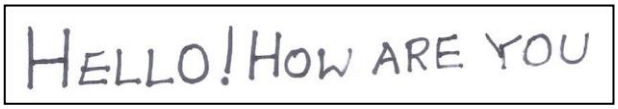

Fig 1. Initially scanned image

The overall program is divided into two parts, forming a boundary tightly enclosing the sentence to be segmented and extracting out individual characters from the sentence.

2. Methodology.

A word or a sentence is first written on a piece of A4 size paper using black marker. The word/sentence is scanned using a high definition scanner, Fig 1.

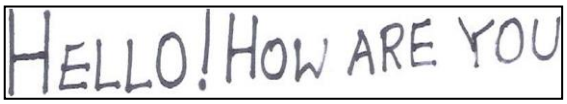

Fig 2. Image obtained after applying Algorithm 1

It can be observed that the initially scanned image matrix is encaged in a rectangular box which consists of unwanted white area other than the text. Algorithm 1 is used to remove the unwanted white area and enclose the text in exactly inside a rectangle the boundary of which touches the largest character reducing the unwanted white area, Fig 2.

B4 is the final image matrix encaging the sentence, fit to boundaries, Fig 2. Then, Algorithm 2 is applied to extract out each and individual character, Fig 3.

The column which finds out the alphabet delimiter is treated as a vertical slider that is why this method is named as slider drifting method.

Fig 3. Image obtained after applying Algorithm 2

It can be observed that algorithm 2 extracts out the delimiters of the characters by using a vertical slider, (Fig 3). Algorithm 1 is again applied to the individual characters to remove the white rows created above the small sized characters and encage the characters to fit in the proper boxes, Fig 4.

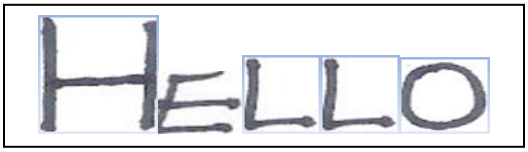

Fig 4. Image of individual characters after applying Algorithm 1

The image matrix formed for different characters are of different sizes. Already existing function in the software can be used to resize the images of different sizes to some standard sized matrix. The standard sized matrix images are used for the training purpose and presented to the net.

3. Result Analysis

SDM is tested for two types of sentences:

Initially, SDM is tested for those sentences where the characters are isolated and have no pixels between two consecutive pixels. SDM displayed 100% accuracy for those sentences.

Number of sentences having isolated characters = 5

Number of alphabets present in each sentence= 100

Total number of alphabets = 500

Number of alphabets encapsulated properly = 500

Accuracy of the method = 100% for the tested sentences

Then SDM is tested for those sentences where some characters are isolated and some are joined. SDM identified those characters which are isolated. So, it can be observed that this method is suited for those handwriting having isolated or disjoint characters. It is very good for the printed text.

Number of sentences having isolated characters = 5

Number of alphabets present in each sentence= 100

Total number of alphabets = 500

Number of alphabets encapsulated properly = 263

Accuracy of the method = 52.6% for the tested sentences

4. Comparisons with other Character Segmentation Methods

SDM is compared with some already developed segmentation methods. These segmentation methods are applied on different scripts. Maximum accuracy achieved is 98%. But the accuracy achieved by SDM is 100% on the tested samples having isolated characters. Only drawback of SDM is that this method is developed to identify only isolated handwritten and printed characters and achieves 100% accuracy for these types of texts only. Table 1 shows the comparison of SDM with different already developed character segmentation methods.

Different types of text paragraphs are presented to the SDM for testing. This method is very good for the printed type of text and can be used in preserving old printed texts in digitized forms. It was found that 100% accuracy is obtained for paragraph having all the isolated characters a vertical column is found between two characters which contain no black pixel.

Table 1. Comparing Slider Drifting Method with some already developed methods in terms of accuracy

S.No.	Title of the research	Script used	Accuracy
1.	Segmentation of Offline Handwritten Bengali Script	Bengali	97.7%
2.	Segmentation and Recognition of Continuous Handwriting Chinese Text	Chinese	85%
3.	Region-Based Touched Character Segmentation in Handwritten Words	English	77%
4.	Segmentation of Isolated and Touching Characters in Offline Handwritten Gurmukhi Script Recognition	Gurmukhi	93.51%
5.	Segmentation of Historical Machine-printed Documents using Adaptive Run Length Smoothing and Skeleton Segmentation Paths	French	70% to 83%
6.	Offline Handwritten Devanagari Script Segmentation, International Journal of Scientific & Technology Research	Devanagri	98%
7.	Optical Character Recognition for Cursive Handwriting	English	95.5%
8.	Slider Drifting Method (Segmentation of isolated handwritten/printed characters in offline English scripts)	English (Text having isolated characters)	100%

5. Discussion

The approach used here is based on the idea of traditional methods of finding gaps between characters and lines called projection profile methods [16]. In projection profile methods, the histogram of zero height is searched for finding out the gaps between character images [17] and therefore is a type of gap finding method. Binarization of characters is not required in these types of methods.

The approach used in this paper uses Binarizaton of characters in order to enhance the efficiency of the system. Algorithms are designed which process the binary values and try to find out the gaps instead of using histograms. Using binary values makes the process very simple.

Binarization also helps to find the difference in gaps between characters of a word and words of a sentence.

SDM is applied for the sentences containing characters having spaces in between. SDM is advantageous for isolated handwritten and printed documents. The method shows excellent performance on printed characters.

The disadvantage of using SDM is that for joined characters efficiency decreases to a great extent. SDM is not at all developed to recognize cursive type of handwritings, where it is almost difficult to find out spaces between the characters. To recognize cursive type of handwriting similar types of methods can be developed which will even be helpful to recognize complicated handwriting types.

6. Conclusion

This method has been developed to extract out characters from a group of characters. A vertical slider is used for the purpose. Emphasis has been given on spaces present between characters which forms a vertical slider used to find gaps between consecutive characters in a sentence. The method works very well for the handwritings with spaces between characters.

References

1. Youssef Es Saady, Ali Rachidi, Mostafa El Yassa, Driss Mammass, Amazigh Handwritten Character Recognition based on Horizontal and Vertical Centerline of Character, International Journal of Advanced Science and Technology Vol. 33, August, 2011, pp 33-50.
   Available: http://www.sersc.org/journals/IJAST/vol33/4.pdf, October, 2012.

2. Amrouch, Y. Es-saady, A. Rachidi, M. El Yassa, D. Mammass, Handwritten Amazigh Character Recognition System Based on Continuous HMMs and Directional Features, International Journal of Modern Engineering Research (IJMER), Vol. 2, Issue 2, Mar-Apr, 2012, ISSN : 2249-6645, October, 2012, pp 436-441.
   Available: http://www.ijmer.com/papers/vol2_issue2/BZ22436441.pdf
3. Shiho Nobesawa, Junya Tsutsumi, Tomoaki Nitta, Kotaro Ono, Sun Da Jiang, MasaKazu Nakanishi, Segmenting a Sentence into Morphemes using Statistic Information between Words, October, 2012.
   Available: http://acl.ldc.upenn.edu/C/C94/C94-1036.pdf,
4. David D. Palmer, The MITRE Corporation, Tokenisation and Sentence Segmentation, October, 2012.
   Available: http://comp.mq.edu.au/units/comp348/ch2.pdf,
5. Basu S, Chaudhuri C, Kundu M, Nasipuri M, Basu D K, Segmentation of Offline Handwritten Bengali Script**,** Proc. of 28th IEEE ACE, pp 171-174, Dec-2002, Science City, Kolkata.
6. Hong C, Loudon G, Wu Y, Zitserman R, Segmentation and Recognition of Continuous Handwriting Chinese Text, International Journal of Pattern Recognition, Artificial Intelligence, 12(2), pp 223-232, 1998.
7. Kurniawan F, Shafry Md, Rahim Md, Daman D, Rehman A, Mohamad D, Shamsuddin S M, Region-Based Touched Character Segmentation in Handwritten Words, International Journal of Innovative Computing, Information and Control, Vol 7, No 6, 2011, pp 1-14.
8. Kumar M, Jindal M K, Sharma R K, Segmentation of Isolated and Touching Characters in Offline Handwritten Gurmukhi Script Recognition, International Journal of Information Technology and Computer Science, 2014, pp 58-63.
9. Nikolaou N, Makridis M, Gatos B, Stamatopoulos N, Papamarkos N, Segmentation of historical machine-printed documents using Adaptive Run Length Smoothing and skeleton segmentation paths, Image and Vision Computing 28, Elsevier, 2010, pp 590–604.

10. Ramteke A S, Rane M E, Offline Handwritten Devanagari Script Segmentation, International Journal of Scientific & Technology Research, Issue 4, MAY 2012, Vol 1, pp 142-145.
11. A rica N, Yarman-Vural F T, Optical Character Recognition for Cursive Handwriting, IEEE Transactions on Pattern Recognition and Machine Intelligence, Vol 24, No 6, June, 2002, pp 801-813.
12. G.N. Swamy, G. Vijay Kumar, "Neural Networks", Scitech, India, 2007.
13. L. Fausett, "Fundamentals of Neural Networks, Architectures, Algorithms and Applications", Pearson Education, India, 2009.
14. Al-Shridah, Nuhammad, Sharieh, Ahmad, "Recognition process of handwritten and typed Arabic letters", AMSE Journal, France, Advances in Modelling, Signal Processing and Pattern Recognition, Volume 45, Issue 1, 2002, pp 1.
15. N Liolios, K Anastasiou, B Kostos, "A new shape transformation approach to handwritten character recognition", AMSE Journal, France, Advances in Modelling, Signal Processing and Pattern Recognition, Volume 46, Issue 6, 2003, pp 55.
16. L. Likforman-Sulem, A. Zahour, B. Taconet, "Text line segmentation of historical documents: a survey", International journal of Document Analysis and Recognition,Volume 9, 2007, pp 123 – 138.
17. R Sanjeev Kunte and R D Sudhaker Samuel, "A Simple and efficient optical character recognition system for basic symbols in printed kannada text", Sadhana, Vol 32, Part 5, October 2007, pp 521 – 533.

APPENDIX A: <u>Algorithm 1</u>. Encaging the sentence in a rectangle with the image fit to boundaries

STEP 1: Read image matrix A.

STEP 2: Initialize the variable j to 1, j is the location of the first column from the left touching the sentence boundary.

STEP 3: Initialize the counter to n (n is number of rows in the matrix)

STEP 4: Repeat while counter is equal to n

Initialize counter to 0

Initialize i to 1

Repeat while i less than equal to n

If A(i, j) equal to 0, where 0 indicates white pixel

Increment counter by 1

End of while

End of If

If counter equals n

Increment j to 1

End of If

End of Step 4 loop

STEP 5: Set matrix B1=A(1 to n, j to m), where m is number of columns in the image matrix.

Step 6. Initialize k to m, k is the location of the last column from the right touching the sentence boundary.

STEP 7: Initialize counter to n

STEP 8: Repeat while counter is equal to n

Initialize counter to 0

Initialize i to 1

Repeat while i is less than equal to n

If A(i,k) equal to 0, where 0 indicates white pixel

Increment counter by 1

End of If

End of while

If counter equals n

Decrement k by 1

End of If

End of Step 8 loop

STEP 9: Set matrix B2=A(l to n, j to k)

STEP 10: Initialize l to 1, l is the location of the first row from the top touching the sentence boundary.

STEP 11: Initialize counter to m

STEP 12: Repeat while counter is equal to m

Initialize counter to 0

Initialize i to 1

Repeat while i is less than equal to m

If A(l,i) equal to 0, where 0 indicates white pixel

Increment counter by 1

End of If

End of while

If counter equals m

Increment l by 1

End of If

End of Step 12 loop

STEP 13: Set matrix B3=A(1 to n, j to k)

STEP 14: Initialize p to n, p is the location of the last row from the bottom touching the sentence boundary

STEP 15: Initialize counter to m

APPENDIX B: <u>Algorithm 2</u>. Extract out alphabets from the sentences.

STEP 1: Initialize k to 1, s_k to j, where s_k locates the column representing the alphabet delimiter and j is the starting location from where the column starts drifting in order to find out the alphabet delimiter

STEP 2: Initialize counter to 0

STEP 3: Repeat while counter is not equal to n, n is the number of rows in the image matrix encapsulating the sentence

STEP 4: Initialize counter to 0

STEP 5: Initialize i to 1

STEP 6: Repeat while i is not equal to n

STEP 7: If A(i, s_k) equals 0

Counter = counter + 1

End of If

End of Step 6 loop

If counter is not equal to n

$s_k = s_k + 1$

End of If

End of Step 3 loop

STEP 8: Set s_k=A(1 to m, j to s_k-1), where matrix s_k encages the first alphabet of the sentence matrix

STEP 9: Assign s_gap to sk, where s_gap locates the gap between two alphabets in the sentence

STEP 10: Initialize counter to n

STEP 11: Repeat while counter is equal to n

STEP 12: Initialize counter to 0

STEP 13: Initialize i to 1

STEP 14: Repeat while i is not equal to n

STEP 15: If A(i,s_gap) equals 1

counter =counter + 1

End of If

End of Step 14 loop

STEP 16: If counter equals n

s_gap = s_gap + 1

End of If

End of Step 11 loop

STEP 17: Assign s_k+1 to s_gap, where s_k+1 is the starting position of the next alphabet in the sentence

STEP 18: [Test stopping condition, where stopping condition is the length if the sentence]

If stopping condition is false go to Step 1

STEP 16: Repeat while counter is equal to m

Initialize counter to 0

Initialize i to 1

Repeat while i is less than equal to m

If A(p,i) equal to 0, where 0 indicates white pixel

Increment counter by 1

End of If

End of while

If counter equals m

Decrement p by 1

End of If

End of Step 16 loop

STEP 17: Set matrix B4=A(1 to p, j to k)