

International Conference on Computer Science and Computational Intelligence (ICCSCI 2015)

Structural off-line handwriting character recognition using approximate subgraph matching and levenshtein distance

Made Edwin Wira Putra*, Iping Supriana Suwardi

**Department of Informatics Engineering, Institut Teknologi Bandung, Jalan Ganesa no. 10, Bandung, Indonesia*

Abstract

This research paper proposed alternatives method for off-line handwriting character recognition in structural approach. The task of character segmentation and normalisation is quite costly when processing large data, especially on off-line handwriting recognition with structural approach. The main idea of this paper is to model a handwritten character into string graph representation. The purpose of those model is to provide ability in improving recognition accuracy without relying in normalisation technique. The graph consists of several edges that indicate the connected vertices. The vertices are the curves that make the character. The curve is extracted by analyzing the character's chain code, and it's string feature is created using certain rules. In this paper, the similarity distance between graph is measured using approximate subgraph matching and string edit distance method. The recognition experiment conducted by comparing both methods on alphabet and number character images taken from ETL-1 AIST Database. We also did recognition accuracy comparison with another related works in number and alphabet handwritten character recognition. The recognition accuracy of levenshtein distance is better than approximate subgraph matching. It also has competitive performance with another method in same class.

© 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

Peer-review under responsibility of organizing committee of the International Conference on Computer Science and Computational Intelligence (ICCSCI 2015)

Keywords: handwriting recognition; structural approach; levenshtein distance; approximate subgraph matching

1. Introduction

* Corresponding author. Tel.: +62 8563 732430;
E-mail address: edwin.wiraputra@gmail.com

Offline text handwriting recognition is a process to recognizing human's handwriting from a scanned paper document and made available in the form of a binary or gray scale image to the recognition algorithm¹⁰. In real application, handwriting recognition used to recognize bank check¹⁶ and recognize address of city's name on German¹. Handwriting recognition also has important role in handwritten text document recognition archived in digital form¹¹. Recognition on unconstrained handwriting today is a difficult challenge because there are many variant of handwriting style¹³. Even after 30 years of research and achievement in handwriting recognition, the development of a general-purpose system which able to recognize unconstrained handwriting text is still an open problem³. Handwriting recognition has been researched on many variety of text. There are research on Arabic handwriting⁶, Chinese handwriting¹⁷, english latin handwriting⁸, and even for mathematic problem solving⁴.

Handwriting recognition is a process to transform the language in its graphical spatial into its symbolic representation. Handwriting data usually obtained by by scanning the text in paper or write using stylus pen on electronic device, which are distinguished as off-line and on-line handwriting respectively. In on-line handwriting, the two dimension coordinates of handwritten text are saved by sequentially ordered. The order of writing available directly. In off-line handwriting, only finished handwritten text that available in the form of image. The image need to be processed to get get the information of handwriting shape in coordinate. Therefore, the handwriting recognition speed of on-line handwriting is faster than off-line handwriting.

The common process in off-line handwriting recognition are classification, distance measure, making prototype, feature extraction, preprocessing, and knowledge acquisition from training data. In Some classification technique that familiar in handwriting recognition are Neural Network⁵, k-Nearest Neighbour¹⁵, and Support Vector Machine^{12,22}. Those classification technique considered as statistical based approach, where the features extracted are labeled by using statistical means.

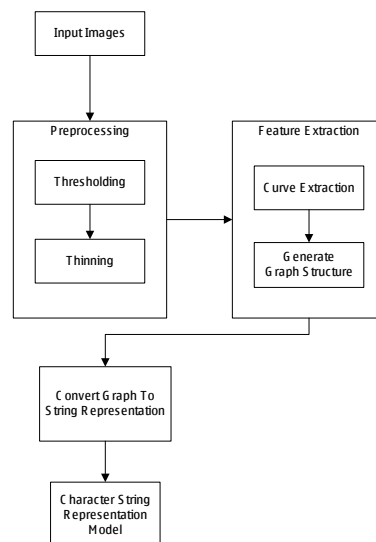


Fig. 1. Feature Extraction Process of the Proposed Method

Another approach is structural based, where the feature extracted is labeled by analyzing it's structure. One of the is ressearch of Hidden Markov Model based for word recognition². Keypoints extracted as feature from character's skeleton and adding connection points to create the graph representation. To measure similarity between graph, the graph converted into feature vector and the similarity calculated using graph edit distance. Hidden Markov Model for number recognition also researched²¹. The research also performed on Hangul character recognition⁷, where the feature extracted based on the primitive stroke model and hierarchical random graph is used to represent structure of stroke connection. The matching score between graphs is not acquired by similarity function, but by probabilistic measures.

To recognize handwritten text, segmentation stage comes before the feature extraction stage. The segmentation procedures are usually consist of text line segmentation, word segmentation, and character segmentation. Logically we can see that the handwritten text are collections of words and the words are collections of characters. If we see the text as graph, then the words will be their vertices. In smaller scope, the words also can be seen as smaller graph consists of character as it's vertices. If we see into smaller detail, the characters are seen as the subgraph of words graph, which consists of connected curves (directly or indirectly). Those curves can be considered as the smallest elements of text that connecting to form graph of character, word, and the text itself. To improve recognition accuracy, the process relied on normalisation technique i.e. slant correction and skew correction. Those technique are required to reduce the error on recognition caused by diversity in handwriting style.

This paper proposed two methods of structural based offline handwriting character recognition that using graph to represent the character structure, and curves as the graph elements. The curves will be modeled as a sequence of string, obtained using certain rule that considering the curves shape directionality. We experiment using two method to measure the similarity distance between character's graph feature. The first method is approximate subgraph matching⁹ to generate any possible candidate matching between two graphs. The second method is string edit distance, where the character's graph feature converted into string representation, and the edit distance between strings measured. The string edit distance algorithm using Levenshtein distance method¹⁹ and related work using levenshtein distance for on-line handwriting recognition is also mentioned²⁰.

The proposed method scheme for feature extraction can be seen in Fig. 1 and the scheme for recognizing using both method can be seen in Fig. 2. The experiment conducted to evaluate the proposed method accuracy and compares it with the accuracy of another method mentioned in^{5,22,15,21}. This research uses handwritten character images of number and alphabet dataset taken from ETL-I Database of Electrotechnical Laboratory National Institute of Advanced Industrial Science and Technology (AIST).

This paper is organized as follows: Sections 2, 3 and 4 describe the preprocessing, curve feature extraction and graph based representation, respectively. Section 5 present the experimental analysis and results. Conclusions are presented in section 6.

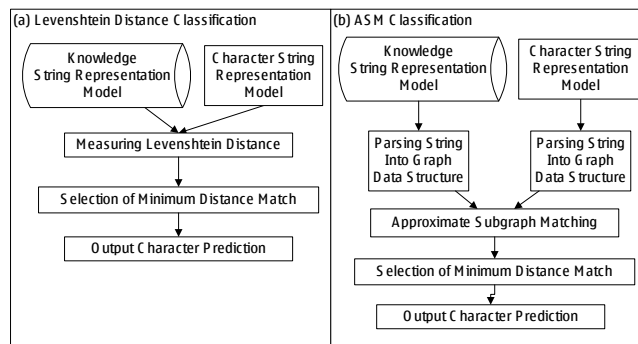


Fig. 2. (a) Classification Process using Levenshtein Distance; (b) Classification Process using Approximate Subgraph Matching

2. Preprocessing

2.1. Thresholding

Thresholding process is necessary to convert the image into binary image. Thresholding has two approach, global thresholding and local thresholding. Global thresholding will calculate a value as threshold based on whole pixel of image. The value will applied to each pixel to decide whether the pixel is foreground or background. Local thresholding calculate different value of thresholding for each pixel of image, based on the neighbouring pixel.

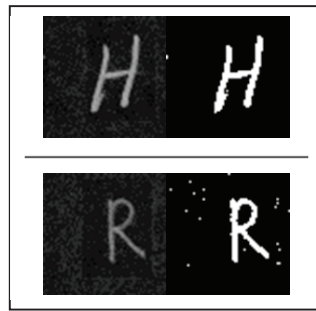


Fig. 3. Example of thresholding results for two images of character

In our experiment, we use Otsu's global thresholding¹⁴ to convert the input handwritten character image into binary image. Otsu's thresholding method involves iterating through all the possible threshold values and calculating a measure of spread for the pixel levels each side of the threshold, i.e. the pixels that either fall in foreground or background. The aim is to find the threshold value where the sum of foreground and background spreads is at its minimum. Fig. 3 is the example of thresholding result. Images in the left side are the original image, while images in the right side are the thresholding result.

2.2. Thinning

Thinning process applied to the binarized image in order to obtain the skeleton of character image. The thinning algorithm used in this paper is Zhang & Suen's¹⁸. The Zhang Suen algorithm consists of successive of two basic steps applied to the contour points of the given region, where a contour is any pixel with value 1 and having at least one 8-neighbour valued 0. Fig. 4 is the example of the thinning result. The images in left side are images from thresholding result that the color is inverted, while the images in right side are the thinning result.

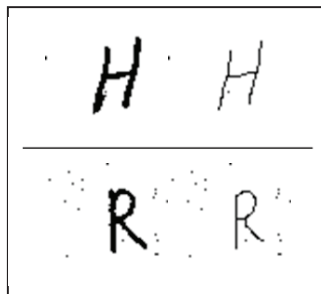


Fig. 4. Example of thinning result after thresholding

3. Feature Extraction

The feature extraction processes consists of two procedure: curve extraction and string feature representation.

3.1. Curve Extraction

Thresholding process is necessary to convert the image into binary image. Thresholding has two approach, global thresholding and local thresholding. Global thresholding will calculate a value as threshold based on whole pixel of image. The value will applied to each pixel to decide whether the pixel is foreground or background. Local thresholding calculate different value of thresholding for each pixel of image, based on the neighbouring pixel. The curve extracted from the skeleton image of character by using connected component analysis method. For each component identified, the skeleton is iterated from left to right to extract it's chain code feature. While processing

the chain code feature extraction, sets of point will be declared as a curve using following rules

- Chain code segmentation. The curve can only consists of maximum 3 kinds of code. Those 3 kinds code group are “1,2,3”, “5,6,7”, “3,4,5”, and “7,8,1”. While iterating the skeleton pixel, if the direction of next pixel is cannot included in current group direction, then current curve will be saved and the next pixel will begin as new curve. Fig. 5 is the 8-neighbour direction code that we use in this paper. The middle are represent current pixel, and the next pixel’s direction represented as number ranged from 1 to 8 as seen in the picture.

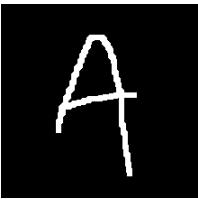

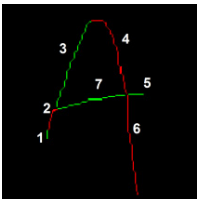
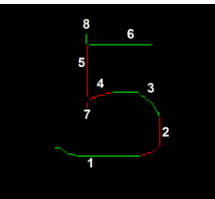
4	3	2
5		1
6	7	8

Fig. 5. The 8-neighbour direction code

- Branch taking. If a curve meet branch while extracting chain code, then the current curve will be saved, and new curve will be extracted.
- End of component. When there is no pixel to visit, then current curve will be saved.

Every point’s coordinate on curve will be transformed with the first point of curve serve as origin point. The examples of extracted curve can be seen in Table 1.

Table 1. Curve Extraction Examples.






Original Image		
Extracted Curve		

3.2. String Feature Representation

The curves extracted will be represented in string contains information about the curve . The pattern of a curve is created using following rules :

- Representation length adjustment. If the curve’s size is not equal or longer than the length requested, then the length will be adjusted until the condition accepted.

Table 2. String Feature Representation Examples

Curve Images					
String Feature Length 4	-Z-Z-Z-Z	XXXX	ZZZZ	-Z-Z-Z-Z	-Z-Z-Z-Z
String Feature Length 8	-Z-Z-Z-Z-Z-Z-Z-Z	XXXXXXXX	ZZZZZZZZ	-Z-Z-Z-Z-Z-Z-Z-Z	-Z-Z-Z-Z-Z-Z-Z-Z

- Points selection. The points taken is based on the length of representation after adjusted. Those points split the curve into approximately same size e.g. if the curve's size is 16 and the representation length requested is 4, then the points taken to represent the curve will be point number 4, 8, 12 and 16. The string will be decided by conditions : (1) if the point is closer to x-axis line, then it is "x" if x-axis value is positive and "-x" if it is negative, (2) if the point is closer to y-axis line, then it is "y" if y-axis value is positive and "-y" if it is negative, and (3) if the point has same distance to both line, then it is "z" if both axis value are positive, "-z" if only x-axis value is negative, "+z" if only y-axis value is negative, and "-+z" if both axis value are negative. Fig. 6 is the picture of area to map the curve's points into string representation on coordinate cartesius. Examples of string feature representation from a curve can be seen in Table 2.

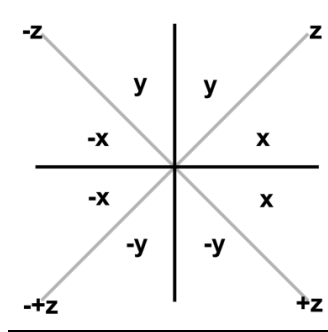


Fig. 6. The area of points coordinate for string representation

3.3. Graph String Representation

The curve feature extracted in the previous process are can be described as connected at some point, whether it directly connected or indirectly connected. The nodes of the graph is curve's string representation, and edges connecting the vertices is labeled by "t" and "f" which represent directly connected and indirectly connected respectively. The string representation of character's structure graph created using following rules :

- The graph is represented by describing all edges that make the graph's structure.
- The string representation of a curve is written as the curve's vertex number first, followed by it's string feature representation separated by slash symbol.
- Edge indicates the two connected curve and label of the edge indicates the connecting state. The edge's string representation consists of both connected curves string representation separated by comma and enclosed by parenthesis. The edge label written before the parenthesis. The first curve of the graph is always stated to connecting with a null curve. The null curve's string representation is "-1/null".

All edges that make the graph's structure are written in single line of string, separated by semicollon. The examples of string graph representation from a character can be seen in Table 3.

Table 3. String Graph Representation Examples

Original Image	String Graph Representation
	t(0/zzzz,-1/null); t(1/-z-z-z-z,0/zzzz); t(2/-x-x-x-x,1/-z-z-z-z); t(3/zzzz,2/-x-x-x-x); t(4/xxxx,3/zzzz); t(5/-+z-z-z-z,+z,4/xxxx); t(6/+z+z+z+z,5/-+z-z-z-z);
	t(0/xxxx,-1/null); t(1/-x-x-z-z,0/xxxx); t(2/-+z-z-x-x,1/-x-x-z-z); t(3/xxxx,2/-+z-z-x-x); t(4/zzzz,3/xxxx); t(5/zzzz,4/zzzz); t(6/xxxx,5/zzzz);

4. Classification

In this paper we experiment using two classification method, string edit distance and approximate subgraph matching.

4.1. Approximate Subgraph Matching

The string representation of character's graph is splitted using regex and mapped into vertex and edge. The vertex of the graph is matched using string edit distance. We set the distance threshold value for node matching as 1, so every matched node with the distance more than 1 will not be accepted. Those value taken in assumption that there is only slight different in curve length expected, since the string pattern of curve supposed to covers various invariant features of curve's structure.

Approximate subgraph matching will generated any possible combination of matched node between graphs, and measured the distance by taking account the graph structure and edge label. To find possible combination, algorithm will find the matched start node, then calculating the structure distance and label distance of the graph. The structure distance is measured by calculating the pairwise shortest distance. The label distance is measured by calculating the difference size on list of label between graphs. The subgraph distance value is measured by calculating the addition of structure distance and label distance.

4.2. Levenshtein Distance

Levenshtein distance algorithm is a string edit distance algorithm which utilize dynamic programming for its operation. Levenshtein distance is the minimum distance required to change one string into another. The change operations are insertion, substitution and deletion, applied on each string's elements sequentially. Fig. 7 showed two examples of calculating distance between strings using Levenshtein distance. The edit distance $D_{s,s'}$ between two strings s and s' is described by the following recursive relation.

$$D_{s,s'}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0 \\ \min \begin{cases} D_{s,s'}(i-1,j) + W_a \\ D_{s,s'}(i,j-1) + W_b \\ D_{s,s'}(i-1,j-1) + W_c \end{cases} & \text{otherwise} \end{cases} \quad (1)$$

The length of strings s and s' are i and j respectively, and the weight of each operations are W_a , W_b and W_c . The $D_{s,s'}$ increases with the decreasing of similarity between the two strings. In our experiment, the operation of insertion, substitution and deletion are scored equally, thus we set 1 to each value of W_a , W_b and W_c .

		k	i	t	t	e	n			S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6		0	1	2	3	4	5	6	7	8
s	1	<u>1</u>	2	3	4	5	6	S	1	<u>0</u>	<u>1</u>	<u>2</u>	3	4	5	6	7
i	2	2	<u>1</u>	2	3	4	5	u	2	1	1	2	<u>2</u>	3	4	5	6
t	3	3	2	<u>1</u>	2	3	4	n	3	2	2	2	3	<u>3</u>	4	5	6
t	4	4	3	2	<u>1</u>	2	3	d	4	3	3	3	3	4	<u>3</u>	4	5
i	5	5	4	3	2	<u>2</u>	3	a	5	4	3	4	4	4	4	<u>3</u>	4
n	6	6	5	4	3	3	2	y	6	5	4	4	5	5	5	4	<u>3</u>
g	7	7	6	5	4	4	3										

Fig. 7. Examples of Levenshtein Distance calculation between strings

5. Experimental Analysis

Experiment has been conducted to evaluate the accuracy of proposed method. The handwritten character database image used in the experiment taken from ETL-1 Database of Electrotechnical Laboratory National Institute of Advanced Industrial Science and Technology (AIST). These dataset contains various images of number, alphabet, special character, and japanese katakana character, acquired from 1445 writers of 7 kinds of job. In our experiment, we use only number and alphabet characters dataset. A few samples from this database are shown in Fig. 8.

5.1. Data

On approximate subgraph matching method, test performed on total 1000 and 2600 sets data handwritten character images of number and alphabet respectively. The 3600 images consists of 100 images for each label on number character (0-9) and alphabet character (A-Z). The training operation used 10% of each label's image as training data. The procedure is to extract the string graph representation of each images and labeled it. The string graph representation are stored as the knowledge for the classification procedure.

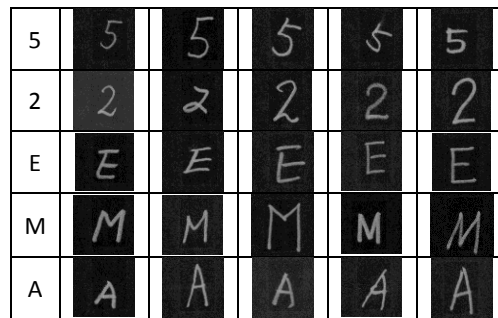


Fig. 8. Examples of Levenshtein Distance calculation between strings

On string edit distance method, test performed on total 8826 and 22647 sets data handwritten character images of number and alphabet respectively. Each data sets contains different number of images for each label. We did training operation two times. The first training use 10% of images as training data, and the second one use 5% of images as training data.

5.2. Evaluation

The first experiment conducted is comparing performance between approximate subgraph matching and string edit distance method. String edit distance provided superior performance in computational cost and accuracy than the approximate subgraph matching method. The result of the test showed on Table 4.

The second experiment is comparing performance for change applied on data training number and string curve feature representation length. The dataset of number characters used on this experiment.

Table 4. Test Result of Different Method

Data set Category	Data Training Size	String Curve Feature Length	Data Test Size	Method	Recognition Rate (%)
Alphabet	10%	4	2600	Approximate Subgraph Matching	52.58%
Number	10%	4	1000	Approximate Subgraph Matching	75.00%
Alphabet	10%	4	22647	Levenshtein Distance	63.24%
Number	10%	4	8826	Levenshtein Distance	82.46%

The reduction number of data used for training from 10% into 5% is affect to decreasing accuracy. The test result on data set of number showed that the accuracy down from 82.4% into 77.7%. But the string curve feature representation length is proved to has significant effect on classification. Accuracy is increased from 77.7% into 83.6% when using 5% data for training and string curve feature length set as 8. It gives superior performance than when using 10% data for trainig and string curve feature length set as 4. The result is showed on Table 5.

Table 5. Test Result of Different String Curve Length

Data set Category	Data Training Size	String Curve Feature Length	Data Test Size	Method	Recognition Rate (%)
Number	5%	4	8826	Levenshtein Distance	77.70%
Number	5%	8	8826	Levenshtein Distance	83.59%
Alphabet	5%	8	22647	Levenshtein Distance	66.38

The comparative result of proposed method performance against another related research are seen in Table 6. From these table, it can be seen that the recognition rate of Levenshtein Distance has achieved competitive performance against Hidden Markov Model²¹ in same class of structural approach. The statistical approach^{15,22} still achieved superior performance in recognition rate of number recognition, but our method has competitive performance with Neural Network method⁵ in alphabet recognition category.

Table 6. Test Result of Different String Curve Length

Data set Category	Data Training Size	Data Test Size	Method	Recognition Rate (%)
Number	441	8826	Levenshtein Distance	83.59%
Alphabet	1132	22647	Levenshtein Distance	66.38%
Number	100	1000	Approximate Subgraph Matching	75.00%
Alphabet	260	2600	Approximate Subgraph Matching	55.28%
Number	400	400	Hidden Markov Model ²¹	84.50%
Number	60000	10000	k-NN ¹⁵	98.73%
Number	60000	10000	Hybrid CNN-SVM ²²	99.81%
Alphabet	520	3410	Neural Network ⁵	62.93%

6. Conclusion

We have introduced the alternatives method of off-line structural handwriting character recognition. To provide ability in improving recognition accuracy without relied on normalisation technique, we use string sequence model to represent the curves, and graph structure for analyzing the character's structure which is also represented in string model. In this paper, we use approximate subgraph matching and levenshtein distance as the classification method. We obtained evaluation result on number and alphabet dataset. The dataset acquired from ETL-1 AIST databases that contains various character of number, alphabet, special character, and japanese katakana, acquired from 1445 writers of 7 kinds of job. The comparison result between both method shown that levenshtein distance has achieved better performance in recognition accuracy for our model. It also has competitive performance with another method in the same class.

We think that our model can be improved and optimized in further research. We can consider on using curve's position relative to the character's size in string feature model. We need to study the proposed scheme for word level recognition of off-line handwriting. We believe, by improving the model in this paper we can utilize the graph characteristic to recognize word without using character segmentation process.

References

1. Brakensiek A, Rigoll G. Handwritten Address Recognition Using Hidden Markov Models. Reading and Learning, Vol. 2956 of Lecture Notes in Computer Science. Springer Berlin/Heidelberg; 2004.p.103–122.
2. Fischer A, Riesen K, Bunke H. Graph Similarity Features for HMM-Based Handwriting Recognition in Historical Documents. 2010 12th Int. Conf. Front. Handwrit. Recognit; 2010.p.253–258.
3. Graves A, Liwicki M, Fernández S, Bertolami R, Bunke H, Schmidhuber J. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans Pattern Anal Mach Intell.* 2009 May;31(5):855–68.
4. Vuong BQ, He Y, Hui SC. Towards a web-based progressive handwriting recognition environment for mathematical problem solving. *Expert Syst. Appl.*, vol. 37, no. 1; 2010.p.886–893.
5. Gupta A, Srivastava M, Mahanta C. Offline handwritten character recognition using neural network. 2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE) [Internet]. IEEE; 2011. p. 102–7.
6. Hamad HaA, Zitar RA. Development of an efficient neural-based segmentation technique for Arabic handwriting recognition. *Pattern Recognit.*, vol. 43, no. 8; 2010. p. 2773–2798.
7. Kim H-Y, Kim JH. Hierarchical random graph representation of handwritten characters and its application to Hangul recognition. *Pattern Recognit.* 2001 Feb; 34(2):187–201.
8. Lee H, Verma B. Binary segmentation algorithm for English cursive handwriting recognition. *Pattern Recognit.*, vol. 45, no. 4; Apr. 2012. p. 1306–1317.
9. Liu H, Hunter L, Kešelj V, Verspoor K. Approximate subgraph matching-based literature mining for biomedical events and relations. *PLoS One.* 2013 Jan; 8(4):e60954.
10. Sas J, Markowska-Kaczmar U. Similarity-based training set acquisition for continuous handwriting recognition. *Inf. Sci. (Ny).*, vol. 191; May 2012. p. 226–244.
11. Likforman-Sulem L, Zahour A, Taconet B. Text line segmentation of historical documents: a survey. *International Journal of Document Analysis and Recognition* 9; 2007.p.123–138.
12. Adankon MM, Cheriet M. Model selection for the LS-SVM, Application to handwriting recognition. *Pattern Recognit.*, vol. 42, no. 12; Dec. 2009. p. 3264–3270.
13. Tanvir Parvez M, Mahmoud S a. Arabic handwriting recognition using structural and syntactic pattern attributes. *Pattern Recognit. Elsevier;* 2013 Jan; 46(1):141–54.
14. Otsu N. A threshold selection method from gray-level histogram. *IEEE Trans. Syst. Man Cybern. SMC-9(1)*; 1979.p.62–66.
15. Impedovo S, Mangini FM, Barbuzzi D. A novel prototype generation technique for handwriting digit recognition. *Pattern Recognit.*, vol. 47, no. 3; Mar. 2014. p. 1002–1010.
16. Impedovo S, peiWang PS, Bunke H(Eds.). *Automatic Bankcheck Processing.* World Scientific;1997.
17. Su TH, Zhang TW, Guan DJ, Huang HJ. Off-line recognition of realistic Chinese handwriting using segmentation-free strategy. *Pattern Recognit.*, vol. 42, no. 1; Jan. 2009. p. 167–182.
18. Zhang TY, Suen CY. A fast parallel algorithm for thinning digital patterns. *Commun ACM.* 1984 Mar;27(3):236–9
19. Levenshtein V. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*10 (8); Feb 1966.p. 707–710.
20. Chowdhury SD, Bhattacharya U, Parui SK. Online Handwriting Recognition Using Levenshtein Distance Metric. *Document Analysis and Recognition (ICDAR)*, 2013 12th International Conference; Aug. 2013.p.79,83, 25–28.
21. Saritha BS Hemanth S. An Efficient Hidden Markov Model for Offline Handwritten Numeral Recognition. *InterJRI Comput. Sci. Netw.*, vol. 1; 2010. p. 7–12.
22. Niu XX, Suen CY. A novel hybrid CNN–SVM classifier for recognizing handwritten digits. *Pattern Recognition*, Volume 45, Issue 4; April 2012.p. 1318–1325.