New approach for segmentation and recognition of handwritten numeral strings

J. Sadri, C. Y. Suen, Tien D. Bui
Center for Pattern Recognition and Machine Intelligence (CENPARMI), Concordia University
1455 de Maisonneuve Blvd. West, Montreal, Quebec, Canada H3G 1M8
Tel: (514)-848-2424 Ext: 7950, Fax: (514)-848-4522,
Email: {j_sadri, suen, bui}@cs.concordia.ca

ABSTRACT

In this paper, we propose a new system for segmentation and recognition of unconstrained handwritten numeral strings. The system uses a combination of foreground and background features for segmentation of touching digits. The method introduces new algorithms for traversing the top/bottom-foreground-skeletons of the touched digits, and for finding feature points on these skeletons, and matching them to build all the segmentation paths. For the first time a genetic representation is used to show all the segmentation hypotheses. Our genetic algorithm tries to search and evolve the population of candidate segmentations and finds the one with the highest confidence for its segmentation and recognition. We have also used a new method for feature extraction which lowers the variations in the shapes of the digits, and then a MLP neural network is utilized to produce the labels and confidence values for those digits. The NIST SD19 and CENPARMI databases are used for evaluating the system. Our system can get a correct segmentation-recognition rate of 96.07% with rejection rate of 2.61% which compares favorably with those that exist in the literature.

Keywords: Handwritten digit recognition, Numeral string segmentation, Numeral string recognition, Genetic algorithms, Neural networks

1. Introduction

Segmentation and recognition of unconstrained handwritten numeral strings is one of the most challenging problems in the area of Optical Character Recognition (OCR). It has been a popular topic of research for years, and it has many potential applications such as postal code reading, bank check processing, and tax form reading [1]. Numeral strings normally include isolated digits (with a lot of variations in their shapes), touched, overlapped, and noisy or broken digits [2]. Some examples of these cases are shown in Figure 1. One of the main challenges in handwritten numeral recognition systems is the segmentation of the connected digits [1].

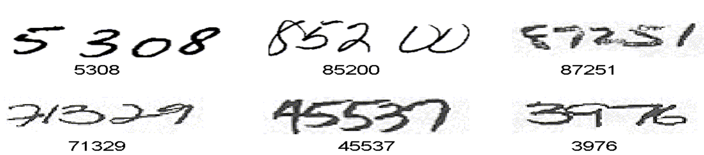


Figure 1: Some examples of numeral strings from NIST SD19 database

Two recognition approaches exist for numeral strings: segmentation-then-recognition, and segmentation-based recognition [3]. In the first approach, the segmentation module provides a single sequence of hypothesis, where each subsequence should contain an isolated character, which is submitted to the recognizer. Since this technique does not use any contextual or recognition feedback for segmentation, it shows its limits rapidly when the correct segmentation does not fit the predefined rules of the segmenter. In the second strategy, first the segmenter provides a list of candidate segments, and then each candidate segment is evaluated by the recognition module. Finally the list is post processed by using contextual information [3].

In this paper, we present a new approach for automatic segmentation and recognition of handwritten numeral strings. We formulate this problem as an optimization problem. The novelty of our method is, first its segmentation module is based on a combination of features from top/bottom foreground-skeletons, top /bottom background-skeletons, and some global information from the string image. Based on a combination of these pieces of information, it tries to find the best cutting points for building segmentation paths. Second, for the first time it uses a genetic algorithm to produce a population of segmentation candidates for the input image string. Each generation of the candidates is evaluated based on its recognition result. Then genetic operations are used to evolve the candidates in the population to improve their segmentation/recognition fitness, and to find solutions for our global optimization criterion. Third, for capturing the variability of the digits in their shapes, a new set of features is used, and a neural network is trained to assign class labels and confidence values to the segmented digits. Our method can prevent under or over segmentation, and also because genetic algorithms are robust in search and global optimization, it can find the best candidates for segmentation and recognition of the input numeral strings.

This paper is organized as follows: Section 2 presents the definition of the problem, Section 3 presents our proposed method, and Section 4 shows the experimental results, and finally Section 5 draws the conclusion.

2. Problem formulation

Segmentation and recognition of a numeral string can be defined as an optimization problem as follows. There is an input image (I), which contains a string of handwritten digits with unknown length, we are looking for a segmentation of this image denoted by S, such that S has m segments (partition) denoted by $s_j, \quad j = 1, \ldots, m$.

$$I = s_1 \cup s_2 \cup \ldots \cup s_m , \quad s_j \cap s_k = \phi , \quad j \neq k , \quad 0 < m < \infty$$

Also there exists a classification function (f) which assigns to each segment or partition s_j of I, a class label c_j and a value or measure of v_j as confidence (fitness, probability, membership, recognizability, …) as below:

$$(c_j, v_j) = f(s_j) \quad , \quad c_j \in \{0,1,2,\ldots, 9\}, \quad v_j \in (0,1) , \quad j = 1, \ldots, m$$

The value v_i shows the confidence of the classifier for the label c_i. Among the set of all possible segmentations (partitionings) of the image I (see Figure 2), we are looking for the best segmentation S of I that maximizes one of the following global objective functions F as follow:

$$F(S) = Min(v_1, v_2, \ldots, v_m) \quad OR \quad F(S) = \prod_{j=1}^{m} v_j$$

The labels c_i corresponding to this segmentation (S), form the result of the recognition for the input string:

$$c_1 c_2 c_3, \ldots, c_m \quad \text{(Numeral string with m digits)}$$

3. Methodology

Based on the above definition, our method consists of three main parts. The first important part is a segmentation module, which tries to provide the best segmentation cuts for the input string as building blocks of the final solution, and the second part is a genetic algorithm [4] which tries to build and find the optimum solution based on these building blocks, and the last part is the recognition module which provides labels and confidence values for evaluation of each segment to be used by genetic algorithm. The general framework of the proposed system is presented in Figure 3. In the next sub-sections these modules are briefly described.

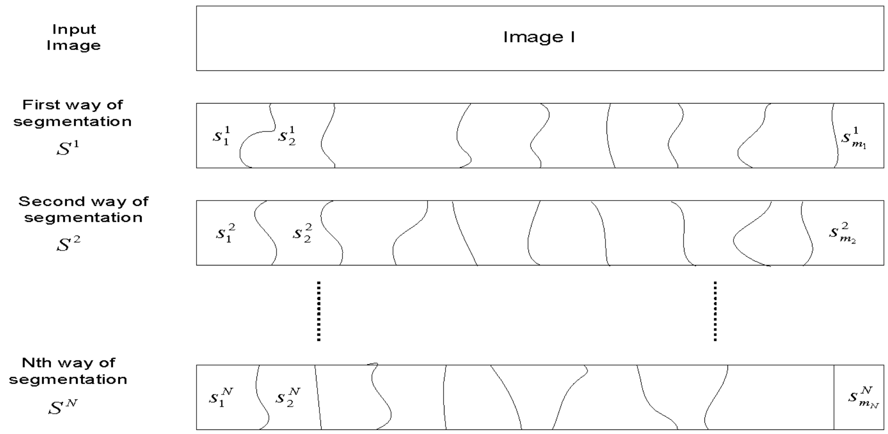


Figure 2: illustrations of possible segmentations for an input string image

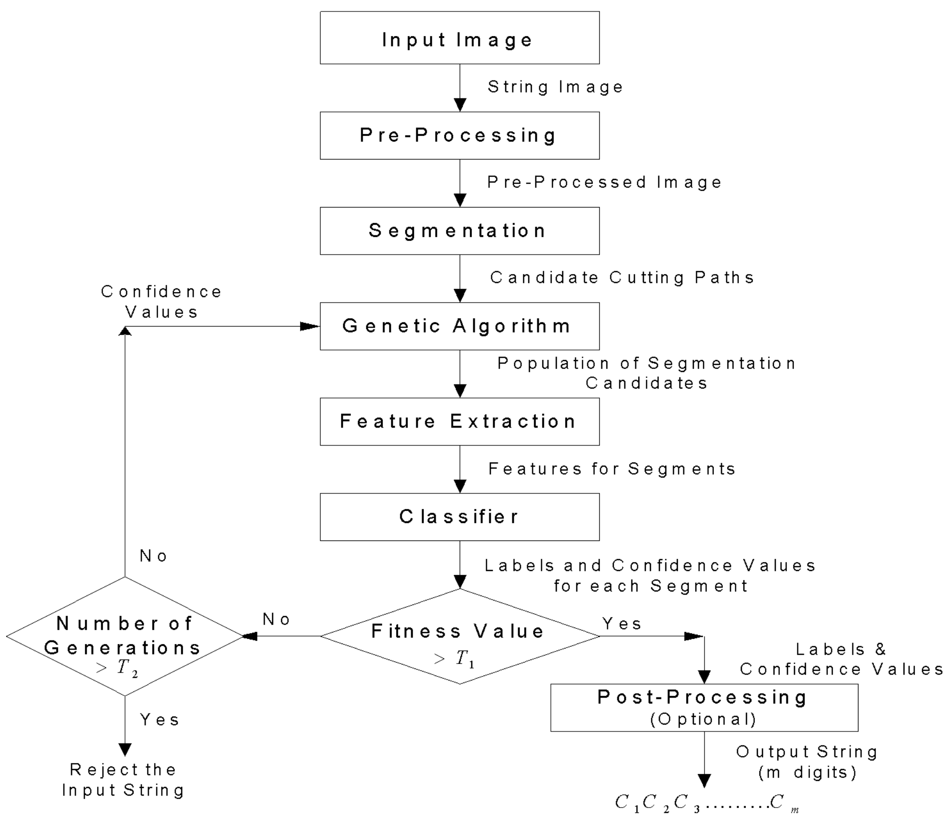
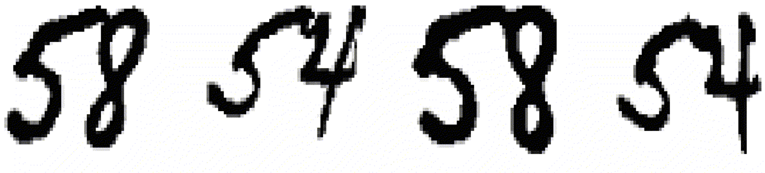


Figure 3: General framework of the system, T_1, and T_2 are two threshold values, which can be adjusted by the user of the system

3.1. Pre-processing

As seen in Figure 1, some images in our database need to be smoothed, by pre-processing steps. The method in [5] is used to remove the small spots of noise and to smooth the edges of the connected components. In the next step, to make the segmentation task easier and the segmentation paths as straight as possible, a method similar to [6] is used to correct the slant of each connected component.



(a) (b) (c) (d)

Figure 4: (a, and b) Original images (c, and d) Smoothed and Slant corrected images respectively.

3.2. Segmentation

The segmentation module consists of two parts, connected component analysis, and splitting of touching digits as illustrated in Figure 5. In the connected component analysis sub module, a string is separated into connected parts. Our observations show, there are three types of possible connected components in a numeral string, parts of a digit, isolated digits, and two or more connected digits, see Figure 6. In touched digits splitting sub module, in order to build segmentation paths, for each component two types of features are extracted: foreground features and background features. In the next two sections we explain how to extract these features.

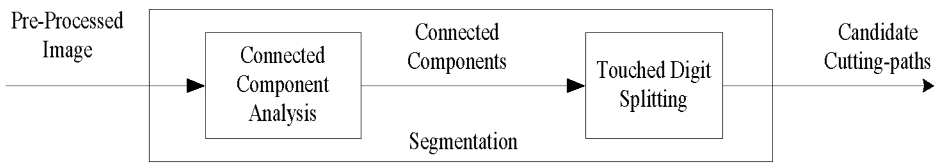


Figure 5: Block diagram of segmentation module

(a) (b) (c)

Figure 6: Three types of connected components found in a numeral string, (a) Parts of a digit, (b) An isolated digit, (c) Two or more connected digits

3.2.1. Generating foreground features

To find feature points on the foreground (black pixels) of each connected component, a new algorithm based on skeleton tracing, is introduced. First by Zhang-Suen thinning algorithm [7], the skeleton of each connected component is extracted, (see Figure 7-b). Then on this skeleton, two points called starting and ending points (denoted by S, and E respectively) are found, as below. In each connected component, starting point, and ending points are the points with the smallest and largest value of the x coordinates respectively. Then from the starting point (S), foreground skeleton is traversed in two different directions: first clockwise, and then anti-clockwise until both traverses reach the ending point (E). In the figure 7-b, we define the traverse in the clockwise as top-skeleton, and the traverse in the anti-clockwise as bottom-skeleton. Actually top / bottom skeletons are subsets of pixels of the original (foreground) skeleton, which are visited during the traversals in clockwise / anti-clockwise directions. When the algorithm traverses the top / bottom skeletons, it looks for intersection points (IPs) which are visited on the skeleton. IPs are points which have more than two connected branches. Corresponding to each visit of any intersection point in the skeletons, there is an angle where its bisector can be found. The intersections of these bisectors with the contour of the connected component are obtained, they are denoted by □ in Figure 7-c. These points form our foreground feature points.

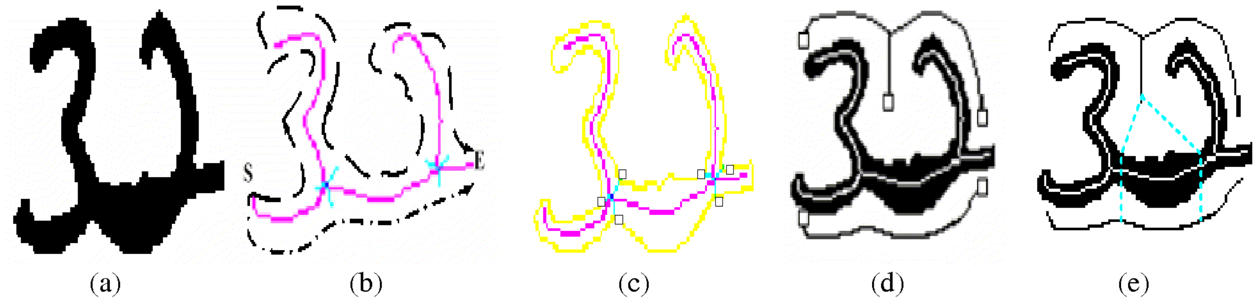


Figure 7: (a) Original image (b) From the starting point (S), skeleton is traversed in two different directions (clockwise: dashed arrows, and anti-clockwise: dotted arrows) to the end point (E), these are called top/bottom foreground-skeletons respectively, (c) Mapping of intersection points on the outer contour by bisectors, (d) Background feature points, (e) Feature points in the foreground and background are assigned to each other to construct the segmentation paths.

3.2.2. Generating background features

Considering the background (white pixels) of each connected component, its skeleton is found (see Figure 7-d). Then on the background skeleton, all the end points are extracted (points which have one black neighbor and they are denoted by □ in Figure 7-d).

3.2.3. Constructing segmentation paths

After finding all the feature points in foreground (denoted by □ in Figure 7-c), and background features (denoted by □ in Figure 7-d), these feature points from top to bottom, or from bottom to top, are assigned together alternatively to construct all possible segmentation paths (denoted by dashed lines in Figure 7_e). The details of our segmentation algorithm can be found in [8]. After finding all possible segmentation paths for the input numeral string they are passed to the genetic algorithm.

3.3. Genetic algorithm

Genetic Algorithms (GA's) can provide robust search in complex spaces, for solving optimization problems [4, and 9]. Although GA's are computationally simple, they are powerful, and they can be utilized for searching spaces which are not continuous, or their objective functions are not uni-modal (see Figure 8). By powerful genetic operations such as reproduction, crossover, and mutation, initial population of candidate solutions can be evolved to better population of solutions in terms of the average of their fitness. Finally a set of candidates whose fitness are greater than a desired threshold, can be selected.

.

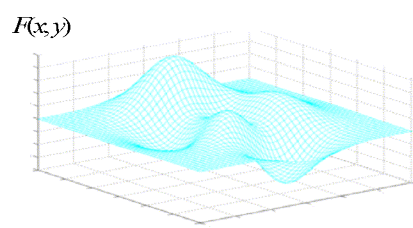


Figure 8: Genetic algorithm can be used for global optimization of multi-modal objective functions in very complex search spaces

After finding all segmentation paths for the input string, they are put together and ordered from left to right, based on the position of their center of gravity (see Figure 9). Each cut path is considered as a gene for building a solution, and denoted (encoded) as a bit in a binary chromosome. For example, if segmentation algorithm could find totally n=5 cutting paths, then 32 possible cases for segmentation of the input string exist. Therefore as Figure 9 shows, each possible segmentation of the input image can be encoded as a bit string with length n=5. By this method all hypotheses for segmentation of a numeral string are represented as chromosomes with the same length. If a bit is zero, the corresponding cutting path (dotted path) is inactive, and it is not considered in the segmentation. If a bit is one, then corresponding cutting path is active, and it is considered in the segmentation. In general for n cutting paths (n bits), there would be totally 2^n possible segmentation hypotheses. We are looking for a solution (segmentation hypothesis) that can give us the maximum confidence in recognition of the input string. (Refer to the definition in section 2). The space of possible segmentations for a long string of numerals can be quite large with 2^n points, where n is the number of cutting paths found by the segmentation module. It is usually much greater than the number of digits in the input string. Since here the search space is not continuous, and we need a feedback of recognition for evaluation of different hypotheses of segmentation, Genetic Algorithm can be a good candidate for searching this space. Also by the above method of representation of the segmentation hypotheses we are able to use all genetic operations such as reproduction, cross over, and mutation to produce new generations of segmentation candidates. In this paper we use the simplest versions of these operations, such as random single-point crossover/mutation, and selection based on roulette wheel [4, and 9]

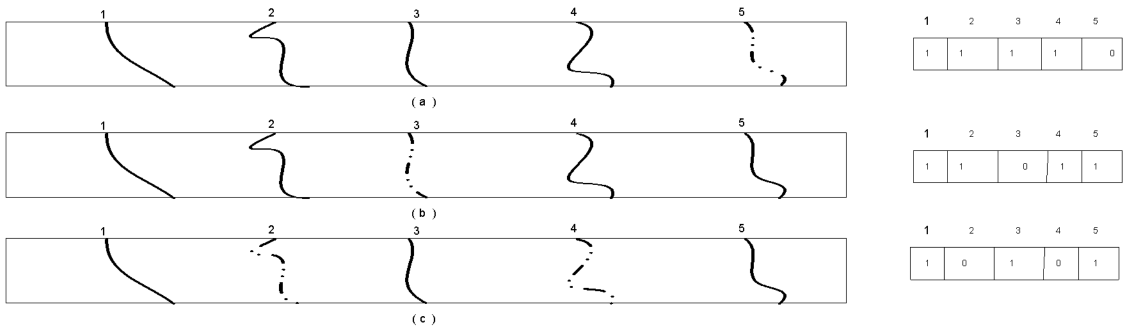


Figure 9: Five cutting paths can make 32 possible ways of segmentation for the input string, each hypothesis is coded as bit string with length 5, dotted paths are not considered, and are coded as zero in the bit string

3.4. Recognition and evaluation

Figure 10 shows all segmentation hypotheses, which are produced by our segmentation algorithm for a sample string. Here each segment is considered as a candidate for an isolated digit, so it can be evaluated based on its recognition score. The general structure of our recognition module is depicted in Figure 11. The details of preprocessing and feature extraction are shown in Figure 12. As Figure 12-b shows, slant of each segment is corrected [6], and its size is normalized into a matrix of size 45 by 45 [10]. For feature extraction, the skeleton of the normalized image is taken (Figure 12-c), and it is divided into 15 by 15 zones such that each zone contains a window of 3 by 3 pixels. If there is at least one black pixel in a zone, its center pixel is set to black; otherwise its center pixel will remain white. Then all the pixels in a zone except center pixel will be removed. Two examples of this transformation are shown in Figures 12-e, and 12-f. Since different arrangements of the black pixels inside the zones (3 by 3 windows) are always replaced by just a single black pixel in the center, it can greatly reduce the variations in the skeletons of the handwritten digits, and extract their basic shapes. The result of this process is shown in Figure 12-d. This image has 15 by 15 pixels which is considered as feature vector, and feed into a MLP neural network for classification. All neurons in our MLP classifier are sigmoid neurons, and the network has 3 layers (225 neurons in input layer, and 85 neurons in hidden layer, and 10 neurons in output layer). By using back propagation learning algorithm [11], this network is trained to produce labels and confidence values for our segmented digits. The network has ten outputs which produce values between 0, and 1, and those values are considered as confidence values for the segmented digits.

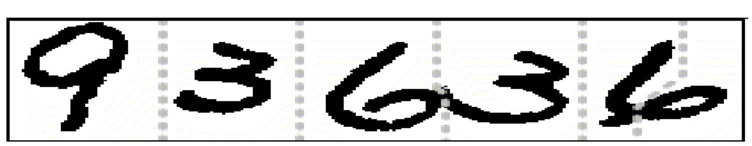


Figure10: Dashed lines are cutting paths produced by our segmentation algorithm. Each segment (the region between two consecutive paths) is considered as a candidate for an isolated digit, and it is evaluated based on its recognition score.

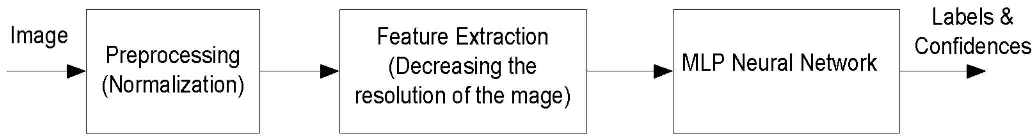
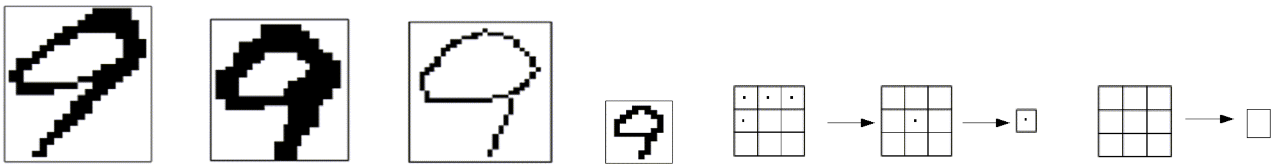


Figure11: Recognition module: After preprocessing and feature extraction, a MLP neural network assigns class labels and confidence values (recognition scores) to each segment



(a) (b) (c) (d) (e) (f)

Figure 12: An example of feature extraction, (a) Original image, (b) Pre-processed, slant corrected, and normalized image (45 by 45 pixels), (c) Skeleton of part b, (d) Reducing the resolution of the skeleton in horizontal and vertical directions by 1/3; the resulting image is considered as a feature vector, (e, and f) Two examples of the window which is used to reduce the variability of the pixels on the skeleton. All the black pixels inside the window are represented by a single pixel in the center of the window; white pixels around the center are removed.

4. EXPERIMENTAL RESULTS

For our experiments we took randomly 300 strings of 4 different lengths (2, 3, 4, and 5 digits per string) totally 1200 images from NIST SD19 (hsf_7) database as a test set. After a visual analysis, we found that in 96.87% of the time, the best segmentation candidates are among the paths that produced by our segmentation algorithm. Our MLP neural network is trained on CENPARMI isolated digit database. This database has 4000 training samples and 2000 testing samples, and compare to similar databases its samples have more variations in their shapes. To improve the outlier rejection [12] of the neural network, 800 outlier samples that produced during the segmentation process are added to the training samples of the CENPARMI digit database. After training the neural network, the recognition result on the test set of the CENPARMI database was 98.2% (although it is not the highest in the literature for this database, it shows good outlier rejection). After some experiments on our genetic algorithm, we selected its parameters as Table 1. In this table parameters P_r, P_c, P_m, M, and G are probability of reproduction, probability of crossover, probability of mutation, size of the initial population, and maximum number of generations respectively. Here n is the number of segmentation hypotheses generated by our segmentation algorithm. As Figure 3 shows, the algorithm terminates in two cases, first if a solution found with desired fitness (a segmented string with enough confidence for its segmentation and recognition), second if it reaches to maximum number of generations, and yet no good solution found. For the first case the algorithm accepts the string, and it outputs the labels, and confidence values for its segments. For the latter case the algorithm rejects the input string. In the algorithm in Figure 3, we take $T_1 = 0.96$ (threshold for acceptance), T_2 = G (maximum number of generations). Table 2 summarizes the segmentation/recognition rates of our system on different string lengths. In average our system can correctly segment, and recognize 96.07% of the test images from NIST SD19(hsf_7), with rejection rate of 2.61%. This result compares favorably to those exist on the literature [3, and 13].

5. CONCLUSIONS AND FUTURE WORKS

Segmentation and recognition of unconstrained handwritten numeral strings is one of the most difficult problems in the area of Optical Character Recognition. The proposed method considers the segmentation and recognition of numeral strings as an optimization problem, and tries to solve the problem by introducing a new method for segmentation. Also for the first time, it uses a genetic algorithm to find the best segmentation paths according to the recognition results of the proposed segments. For recognition of isolated segments, it extracts new features, and utilizes a MLP neural network. In future, we would like to improve the performance and efficiency of our segmentation and recognition algorithms by better exploiting all sources of knowledge, and using advanced machine learning techniques.

Table 1: Different parameter values for our genetic algorithm

P_r	P_c	P_m	M	G
0.4	0.55	0.05	$2^{\lfloor n/2 \rfloor}$	$2^{\lfloor n/2 \rfloor}$

Table 2: Recognition rates on the test samples

String length	Recognition Rate (%)	Rejection Rate (%)
2	97.56	2.23
3	96.34	2.31
4	95.27	2.79
5	95.12	3.10
Average rates	96.07	2.61

REFERENCES

[1] U. Pal, A. Belaid, and Ch. Choisy, "Touching Numeral Segmentation Using Water Reservoir Concept," Pattern Recognition Letters, vol. 24, pp. 261-272, 2003.

[2] K.K. Kim, J.H. Kim, and C.Y. Suen, "Segmentation-Based Recognition of Handwritten Touching Pairs of Digits Using Structural Features," Pattern Recognition Letters, Vol. 23, pp. 13-24, 2002.

[3] L.S. Oliveira, R. Sabourin, F. Bortolozzi, and C.Y. Suen, "Automatic Segmentation of Handwritten Numerical Strings: A Recognition and Verification Strategy," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 11, pp. 1438-1454, Nov. 2002.

[4] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison–Wesley Longman Inc., 1989.

[5] N.W. Strathy, C.Y. Suen, and A. Krzyzak, "Segmentation of Handwritten Digits Using Contour Features," Proc. Int. Conf. Document Analysis and Recognition, pp. 577-580, 1993.

[6] T.Yamaguchi, Y.Nakano, M.Maruyama, H.,Miyao, and T.Hananoi, "Digit Classification on Signboards for Telephone Number Recognition", Proc. of Seventh Int. Conf. on Document Analysis and Recognition, pp. 359-363, 2003.

[7] T.Y. Zhang, and C.Y. Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns", Communication ACM, vol. 27, no. 3, pp. 236-239, 1984.

[8] J. Sadri, C. Y. Suen, T. D. Bui, "Automatic Segmentation of Unconstrained Handwritten Numeral Strings", to appear in the Proceeding of 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR'04), Tokyo, Japan, Oct., 2004.

[9] J.R. Koza, *Genetic Programming - On The Programming of Computers by Means of Natural Selection*, The MIT Press, 1992.

[10] C.L. Liu, K. Nakashima, H. Sako, H. Fujisawa, "Handwritten Digit Recognition: Investigation of Normalization and Feature Extraction Techniques," Pattern Recognition, Vol., 37, pp. 265-279, 2004.

[11] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 2003.

[12] C.L. Liu, H. Sako, H. Fujisawa, "Integrated Segmentation and Recognition of Handwritten Numerals: Comparison of Classification Algorithms", Proceeding of 8th International Workshop on Frontiers in Handwriting Recognition (IWFHR'02), Ontario, Canada, pp. 303-308, 2002.

[13] Y.K, Chen, and J.F. Wang, "Segmentation of Single or Multiple Touching Handwritten Numeral String Using Background and Foreground Analysis," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, pp. 1304-1317, 2000.