A fuzzy approach for segmentation of touching characters

Giuseppe Airò Farulla[1], Nadir Murru[2], Rosaria Rossini[3]

giuseppe.airofarulla@polito.it, nadir.murru@unito.it, rossini@ismb.it

[1]Department of Control and Computer Engineering, Politecnico di Torino,
Corso Duca degli Abruzzi 24, 10129, Torino, Italy
[2]Department of Mathematics, University of Turin,
Via Carlo Alberto 10, 10121 Torino, Italy
[3]Istituto Superiore Mario Boella, Center for Applied Research on ICT,
Via Pier Carlo Boggio 61, 10138, Torino, Italy

Abstract

The problem of correctly segmenting touching characters is an hard task to solve and it is of major relevance in pattern recognition. In the recent years, many methods and algorithms have been proposed; still, a definitive solution is far from being found. In this paper, we propose a novel method based on fuzzy logic. The proposed method combines in a novel way three features for segmenting touching characters that have been already proposed in other studies but have been exploited only singularly so far. The proposed strategy is based on a 3–input/1–output fuzzy inference system with fuzzy rules specifically optimized for segmenting touching characters in the case of Latin printed and handwritten characters. The system performances are illustrated and supported by numerical examples showing that our approach can achieve a reasonable good overall accuracy in segmenting characters even on tricky conditions of touching characters. Moreover, numerical results suggest that the method can be applied to many different datasets of characters by means of a convenient tuning of the fuzzy sets and rules.

1 Introduction

Automatic recognition of both printed and handwritten characters remains a challenging problem in pattern recognition. Most existing Optical Character Recognition software (OCR) deal with it by exploiting simultaneously two highly correlated techniques: character segmentation and pattern recognition. As part of the OCR process, character segmentation techniques are applied to patterns representing individual characters to be recognized. The simplest way to perform character segmentation would be to exploit the space between characters. This strategy unfortunately fails when considering mathematical formulae, handwritten and printed words with touching characters, representing well known issues that often occur in degraded (e.g., photocopies) or compressed text images [39]. In these situations, two or more adjacent characters touch together and share common pixels. To identify the touching regions and provide a correct segmentation is crucial to recognition, since incorrectly segmented characters are unlikely to be correctly recognized even from high-performance pattern recognition algorithms [52]. In facts, many researchers state that errors in characters segmentation affect overall pattern recognition performance more than a degradation in the starting image [14]. Segmentation of touching components is crucial to get higher recognition rates by OCR systems [6].

Common techniques for character segmentation exploit several aspects characterizing letters and their shapes, such as vertical projection, pitch estimation or character size, contour analysis, or segmentation–recognition coupled techniques [24], [27]. One of the most difficult problem an image segmentation algorithm has to address is the segmentation of touching characters [38], [41]. Very often, adjacent characters are touching, and may overlap, making hard the task of segmenting a given expression or word correctly into its character components [4, 22]. Given the relevance of such challenging task, several methods have been developed in last years for performing optimal segmentation of touching characters. Kurniawan et al. [18] identify touching positions in Latin handwritten characters by means of self organizing feature maps and a region–based approach. In [20] and [21], the authors also deal with segmentation of Latin handwritten texts. Different approaches involving thinning algorithms can be found in [26] and [36]. Among the earlier pieces of work on touching character segmentation, some approaches rely on contour analysis of the connected components for segmentation [17, 7]. In [45], Sharma et al. study the problem of detecting arbitrarily–oriented text from video frames. Cut positions in touching characters are evaluated using the top distance profile. Roy et al. [40] addressed the problem of segmenting touching characters with different orientations. In [50] and [42], authors developed segmenting approaches leveraging on genetic algorithms. Rehman et al. [37] identify character boundaries by using a set of heuristic rules. Louloudis et al. [23] performed text line and word segmentation of handwritten documents by applying the Hough transform. In [29], the authors addressed the problem of automatically segmenting words from historical handwritten documents. The water reservoir algorithm has been exploited in some researches, e.g., [35] and [19]. In [2] and [47], the authors presented methods derived by combining different segmentation techniques. Further methods can be found in [1], [5], [8], [11] and [51]. It is worth mentioning that works [33] and [9] also deal with segmenting characters and symbols within mathematical expressions.

Generally, these strategies need a preprocessing step where the input RGB image is converted to grayscale by eliminating the hue and saturation values while retaining the illumination, then converting the grayscale image to binary, obtaining a matrix whose entries are 0 for foreground pixels (black) and 1 for background pixels (white) for all other pixels. Since different thresholds are used for detection it is possible that some feature of characters is lost in the process. In our experiments we resort on the Otsu thresholding method [34], which has proven to be very robust to noise and to changes in scenes and input images, providing thresholds for image binarization ensuring that information loss is minimized.

A function is used to evaluate each column of the matrix leveraging on features that characterize typical character positions within the text, giving a value to each column. Finally, cut positions (i.e., columns) are chosen depending on these values. Common functions implied are the ratio of the second difference of the vertical projection (e.g., [15]) and the peak–to–valley (e.g., [25]). Other functions are, e.g., based on number of black pixels, number of white pixels counted from the top of the column to the first black pixel, crossing count (i.e., number of black to white transitions), number of identical black (white) pixels with left (right) column, width to height ratio for the remaining left (right) pattern after cutting (e.g., [3]). Another approach can be found in [10], where the authors used the inverse crossing count, measure of blob thickness and degree of ”middleness” for defining a function that identifies when a column, a row, or a diagonal is a cutting position.

However, we argue that all the methods and approaches above mentioned do not provide a comprehensive answer to the problem of segmenting touching characters. Indeed, their performances are not always optimal, radically depending on the specific set of characters involved in the segmentation. At the moment, there does not exist a standard approach for the segmentation of touching characters. Thus, this is currently an active research field.

There has always been a dilemma whether it is more convenient to segment first and then recognize the patterns, or instead classify while segmenting. Authors in [4] review Casey and Lecolinet work [6] stating that that the strategies for segmentation can be classified into three main strategies as follows:

1. the classical approach already described;
2. recognition-based segmentation, in which a search is made for image components that match with the character classes in a valid alphabet;
3. holistic approach that attempts to recognize the word as a whole.

In this classification, our work lies near to the first class, still presenting some important advances: in fact, we aim at combining some of the previously cited features, usually exploited one at a time, by means of an original fuzzy logic approach in order to improve performances in separating touching characters. Indeed, the selection of the features that characterize touching positions is an art rather than a technique. In other words, the selection of the features mainly depends on the experience of the authors. Thus, in this context, fuzzy logic can be very useful, since it is congenial to capture and to code expert–based knowledge in view of performing targeted simulations. Taking this into strong consideration, we also leverage on optimization techniques to increase overall performances of our approach.

Fuzzy logic has been already exploited to perform image segmentation. For instance, Garain and Chaudhuri [10] used fuzzy multifactorial analysis to combine some of the features previously described. In [32], a survey on image segmentation techniques using fuzzy clustering is presented. Fuzzy logic has been also exploited for developing segmentation–recognition coupled techniques [12]. A non–linear fuzzy approach can be found in [44]. In [31], authors used edge corners and fuzzy logic to develop segmentation techniques exploited to break down Captcha. Further approaches can be found in [13], [46], [48].

In this paper, we propose a novel fuzzy approach that differs from the state of the art in several aspects. Firstly, we combine by means of a fuzzy strategy some features that have never been exploited together in previous works. Secondly, we develop an original strategy based on an inference system composed by 3–input/1–output with fuzzy rules specifically optimized for the purpose of separating touching characters in the case of Latin printed and handwritten characters. The strength of our fuzzy strategy relies on the possibility to adjust its parameters in such a way that they can fit the characteristics of the data set. In other words, the parameters of the method are extracted a priori considering the different characters in the data set.

The inference engine is based on the Mamdani model with if–then rules, minimax set–operations, sum for composition of activated rules and defuzzification based on the centroid method. We have chosen the Mamdani model since it is congenial to capture and to code expert–based knowledge [28].

The paper is organized as follows. In Section 2, we present the fuzzy strategy conveniently developed for performing segmentation of touching characters. In section 3, we present the numerical results that show the effectiveness of the proposed method. Specifically, in Section 3.1, we describe the datasets used for simulations. Sections 3.2 and 3.3 are devoted to test the method on datasets of Latin printed and handwritten characters, respectively. Finally, in Section 4 we draw some conclusion and present future works.

2 Fuzzy strategy

In the following, we only focus on binarized images, for homogeneity with the solutions already presented. In a binarized image, a pattern can be represented by a matrix whose entries are 0 (black pixels) and 1 (withe pixels). Generally, methods for segmenting touching characters define a function based on some features that characterize cut positions.

Then, such a function is evaluated for each column of the matrix and the cut position is chosen depending on these values. Classical functions of this kind are the peak–to–valley function g and the function h defined as

$$g(i) = \frac{V(l_i) - 2V(i) + V(r_i)}{V(i) + 1}, \quad h(i) = \frac{V(i-1) - 2V(i) + V(i+1)}{V(i)},$$

where $V(i)$ denotes the vertical projection function for the i-th column, l_i and r_i are the peak positions on the left side and right side of i, respectively. The column with the highest value of g (or h) is identified as the cutting column. A further feature, that can suggests if the i–th column can be a cut position, is the distance $f(i)$ between i and the center of the pattern. Indeed, generally, cutting columns are located near to the center of the pattern. Clearly, this feature should be only considered as an indication of the neighborhood where the cutting column is probably located. Indeed, we will exploit such a feature in combination with the previous functions with the aim of use it in order to correct the results provided by the other functions. In this section, we combine functions f, g, and h by means of a fuzzy strategy that conveniently balances these functions.

Let us introduce the notion of a "fuzzy degree" qualifying a column i to be a cut position: in short, $\rho = \rho(i) \in [0, 1]$. In our model, the lower the value of ρ, the more probable is that we have located a good cutting position. The strategy can be detailed by means of the fuzzification of the functions f, g, h.

Given a pattern in a binarized image, let A, m, n, and c be the matrix of pixels of the binarized image, the number of row of A, the number of column of A, and the central column of A, respectively. In the following, when we refer to a column i of A, we refer to the i–th column of A, i.e., we are considering the vector of length m whose elements are the entries of the i–th column or we are only considering its position. This will be clear from the context.

The central column c is evaluated by means of $c = \frac{n+1}{2}$. When n is odd, c is clearly the central column of A; when n is even, we consider as the central column the mean between $\frac{n}{2}$–th column and $\frac{n}{2} + 1$, even if in this case c is not an integer number. In this way, for each column i of A, we define its distance from the center of the pattern as $f(i) = |c - i|$. In our fuzzy strategy, we take into account the normalized distance between each column i and the central column c, i.e., we consider $\bar{f}(i) = \frac{f(i)}{c}$.

Similarly, for each column i of A, instead of directly using the functions g and h, we consider the normalized functions

$$\tilde{g}(i) = \frac{g(i) - \min_{j \in \mathcal{C}} g(j)}{\max_{j \in \mathcal{C}} g(j) - \min_{j \in \mathcal{C}} g(j)}, \quad \tilde{h}(i) = \frac{h(i) - \min_{j \in \mathcal{C}} h(j)}{\max_{j \in \mathcal{C}} h(j) - \min_{j \in \mathcal{C}} h(j)},$$

where $\mathcal{C} = \{1, 2, ..., n\}$ is the set of the columns of A. Note that functions $\tilde{g}$ and $\tilde{h}$ are well–defined since we consider matrices A where at least two columns are different. Finally, in the following we will use the functions

$$\bar{g} = 1 - \tilde{g}, \quad \bar{h} = 1 - \tilde{h}$$

so that low values of $\bar{g}$ and $\bar{h}$ identify cutting columns.

Functions $\bar{f}$, $\bar{g}$, $\bar{h}$ are fuzzified by defining convenient fuzzy sets and related membership functions. The fuzzy degree ρ will be evaluated combining these functions by means of some fuzzy rules. Fuzzy sets, membership functions, fuzzy rules will be specified in sections 3.2 and 3.3.

The inference engine will be the basic Mamdani model [28], with if–then rules, mini-max set–operations, sum for composition of activated rules, and defuzzification based on the centroid method. The Mamdani model is congenial to capture and to code expert–based knowledge in view of performing targeted simulations; accordingly the system's performance is tuned by means of expert–based choices, heuristic criteria and non–linear optimization methods.

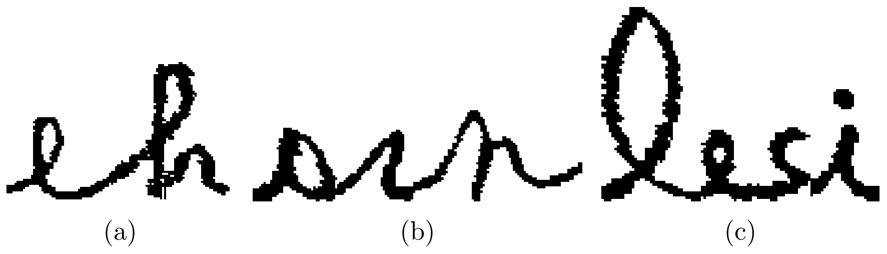
Figure 1: Samples of, respectively, two (a), three (b), and four (c) handwritten cursive touching character patterns.

3 Experimental tests

Although creating a specific dataset of touching character is not our main contribution, in our research we faced the lacking of good and standardized datasets on which testing our algorithms. This reason forced us to build a dataset of touching characters. This need is shared with many other researchers and works, including recent ones as [43] (although this last paper refers to Persian handwriting recognition algorithms rather than to Latin characters-based ones). In particular, we created two different datasets for Latin characters, one containing handwritten cursive characters, the other containing sided machine printed characters. In the following, we discuss our choices and methodology.

Due to the different characteristics of the two datasets, in the remaining parts of the section we discuss the process to select the best parameters for running our method on the two datasets. Our fuzzy strategy has specific parameters meant to capture the differences between one dataset to another. Such parameters are tailored on the specific dataset characteristic by using an heuristic way and non–linear optimization methods. Moreover, we would like to point out that the three features combined in our fuzzy routine seem to be sufficient for obtaining optimal results in the segmentation. In facts, adding further features appear to be useless. For instance, we have verified that the use of the crossing count as a fourth fuzzy input did not lead to improvements. The experimental results are presented in the following.

3.1 Construction of the datasets

The ultimate purpose of getting good segmentation results, especially when touching character are considered, is to boost the the performances for what concerns overall recognition accuracy. Despite the fact thus that, to be fair, different approaches should be evaluated on the basis of their recognition performances, there are not many authors publishing their results on a benchmark database [49]. Also, as stated in many state-of-the-art work, e.g., [18], unfortunately at the present a comprehensive dataset specific on touching characters is still missing. Given that, for researches it is difficult to conduct experiments and to analyze any proposed method; in addition, it is hard to fair compare performances and results obtained from different authors. We try to overtake such an obstacle by proposing two datasets.

The first one, called dataset A, contains images of handwritten cursive characters we have built relying on samples from a standard dataset, in fashion of [18]. In particular we started from the CCC database [5]. The CCC database contains 57'293 samples of cursive characters that were manually extracted from images coming from different input sources, mainly related to American Post Services. They include both upper and lower case

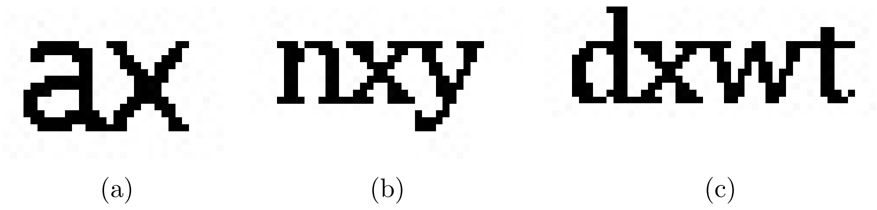

Figure 2: Samples of, respectively, two (a), three (b), and four (c) machine printed touching character patterns.

letters. Each sample is stored as a binary matrix within the database, and accompanied with information about the size of the matrix itself and the character that is represented. Starting from the whole database, we developed a MATLAB script to randomly extract 1'000 of its samples, taking care of maintaining a uniform distribution for all the characters chosen, both in their upper and lower version. These samples were later combined and merged together to form two, three, and four touching character patterns, each of whom is accompanied by a textual descriptor indicating the index of the proper cut column (or columns, in the case of three and four multiple touching character patterns). One sample of each category of patterns is shown in Figure 1. For instance, the descriptor of the sample represented in Figure 1a states that the proper column to cut to properly separate the “e” and the “h” characters is the 52^{th}.

This merging is however unsupervised, and so improper combinations happen during the process. So, we had to filter out the most unrealistic ones. Firstly, we discarded all the samples with significant difference in their heights. Secondly, we manually removed combinations without touching patterns (i.e., the characters were well separated) or with touching patterns that seemed impossible to happen in real world. At the end, we kept 153 combinations, of which 139 represent two touching character patterns, and the other are equally divided into three and four touching character patterns. The disproportion because the common touching characters consist of two characters, while three or more touching characters are rare [50]. Moreover, note that the quantity of combinations of touching characters contained in our dataset is in compliance with other similar datasets constructed using the CCC database (e.g., in [18] a dataset of 123 touching characters is used).

The second one, called dataset B, contains images of sided machine printed characters we have built resorting on a second MATLAB script. We have identified a list of font types (namely Cambria, Candara, Georgia, Lucida Sans Regular, Times New Roman and Verdana Bold) and sizes (namely 10, 20 and 25); for each type and size a MATLAB script combines into images the lower characters from the alphabet to form two, three, and four touching character patterns. Each image is accompanied by a textual descriptor, which in this case indicates directly the characters represented. One sample of each category of patterns is shown in Figure 2. For instance, the descriptor of the sample represented in Figure 2a states that the images represent the string “ax”. Also in this case, we preferred to revise manually the dataset to remove missing, or unrealistic, touching patterns. At the end, we kept the most promising 189 combinations (where 168 are composed by two touching characters), in order to define a challenging dataset to test our approach.

3.2 Tests on Latin printed characters

In the following, we discuss the results of segmentation of touching characters from the dataset B described in the previous section, accordingly to the fuzzy strategy described in section 2. Fuzzy sets and membership functions related to $\bar{f}$, $\bar{g}$ and $\bar{h}$ are defined accordingly to expert based choices. Moreover, their construction has been optimized by using the Particle Swarm Optimization (PSO) algorithm [16] in order to improve overall performances of our fuzzy strategy. Similarly, the fuzzy rules (described in the following) have been tuned using both heuristic criteria and the PSO algorithm.

Given a matrix A as defined in section 2, for each column i of A, $\bar{f}(i)$, $\bar{g}(i)$ and $\bar{h}(i)$ are evaluated and the degree $\rho(i)$ is provided by the following inference scheme that includes three inputs (fuzzification of $\bar{f}$, $\bar{g}$, $\bar{h}$) and one output (cutting degree ρ). The column i with the lowest value of ρ is considered as the cut column.

The function $\bar{f}$ is fuzzified by defining the following fuzzy sets:

- if $\bar{f}(i) \leq 0.35$, then distance from the center of the pattern is *Low*;
- if $\bar{0}.15 \leq f(i) \leq 0.75$, then distance from the center of the pattern is *Medium*;
- if $\bar{f}(i) \geq 0.5$, then distance from the center of the pattern is *High*.

For the function $\bar{g}$, we define the following fuzzy sets:

- if $\bar{g}(i) \leq 0.4$, then $\bar{g}(i)$ is *Low*;
- if $0.2 \leq \bar{g}(i) \leq 0.5$, then $\bar{g}(i)$ is *Medium*;
- if $\bar{g}(i) \geq 0.45$, then $\bar{g}(i)$ is *High*.

The function $\bar{h}$ is fuzzified by means of the following fuzzy sets:

- if $\bar{h}(i) \leq 0.4$, then $\bar{h}(i)$ is *Low*;
- if $\bar{0}.1 \leq h(i) \leq 0.75$, then $\bar{h}(i)$ is *Medium*;
- if $\bar{h}(i) \geq 0.5$, then $\bar{h}(i)$ is *High*;

Figures 3, 4, and 5 show the membership functions of the previous fuzzy sets.

Finally, for the fuzzy output ρ we define the following fuzzy sets, whose membership functions are depicted in Figure 6:

- if $\rho(i) \leq 0.5$, then $\rho(i)$ is *Low*;
- if $0.4 \leq \rho(i) \leq 0.6$, then $\rho(i)$ is *Medium*;
- if $\rho(i) \geq 0.5$, then $\rho(i)$ is *High*;

The inference system is based on the following rules, that combine the three inputs $\bar{f}(i), \bar{g}(i), \bar{h}(i)$ in order to produce the fuzzy output $\rho(i)$, for each column i of A:

1. if $\bar{f}(i)$ is Low and $\bar{h}(i)$ is Low, then $\rho(i)$ is Low;
2. if $\bar{f}(i)$ is Low and $\bar{g}(i)$ is not High and $\bar{h}(i)$ is not Low, then $\rho(i)$ is Low;
3. if $\bar{f}(i)$ is Low and $\bar{g}(i)$ is High and $\bar{h}(i)$ is Medium, then $\rho(i)$ is Medium;
4. if $\bar{f}(i)$ is Medium and $\bar{h}(i)$ is not High, then $\rho(i)$ is Medium;
5. if $\bar{f}(i)$ is Medium and $\bar{g}(i)$ is Low and $\bar{h}(i)$ is High, then $\rho(i)$ is Medium;
6. if $\bar{f}(i)$ is High and $\bar{g}(i)$ is not High and $\bar{h}(i)$ is Low, then $\rho(i)$ is Medium;
7. if $\bar{f}(i)$ is High and $\bar{g}(i)$ is Low and $\bar{h}(i)$ is Medium, then $\rho(i)$ is Medium;
8. if $\bar{f}(i)$ is Low and $\bar{g}(i)$ is High and $\bar{h}(i)$ is High, then $\rho(i)$ is High;
9. if $\bar{f}(i)$ and $\bar{g}(i)$ and $\bar{h}(i)$ are not Low, then $\rho(i)$ is High;

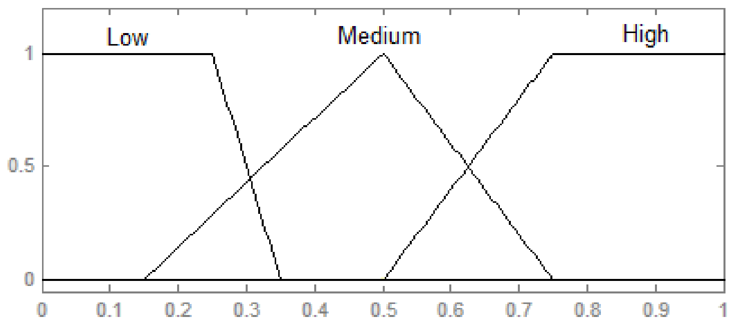

Figure 3: Membership functions of the fuzzy sets related to $\bar{f}$ (for dataset B)

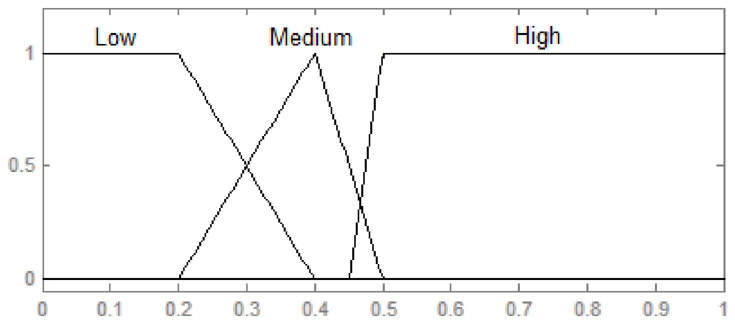

Figure 4: Membership functions of the fuzzy sets related to $\bar{g}$ (for dataset B)

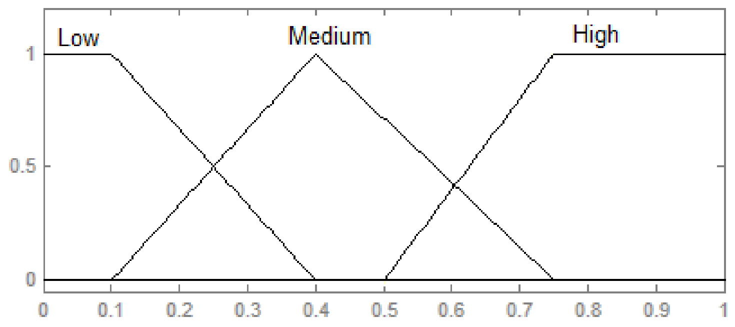

Figure 5: Membership functions of the fuzzy sets related to $\bar{h}$ (for dataset B)

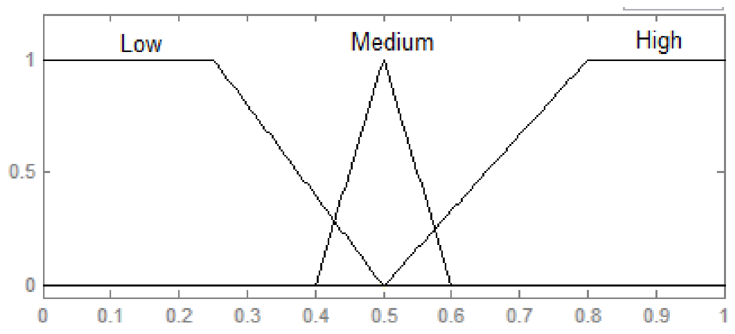

Figure 6: Membership functions of the fuzzy sets related to ρ (for dataset B)

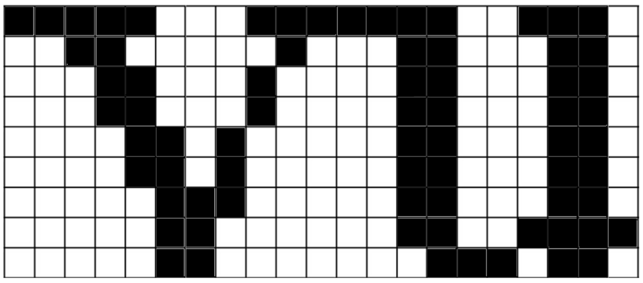

Figure 7: Touching characters "vu" for font Times New Roman and font size of 20

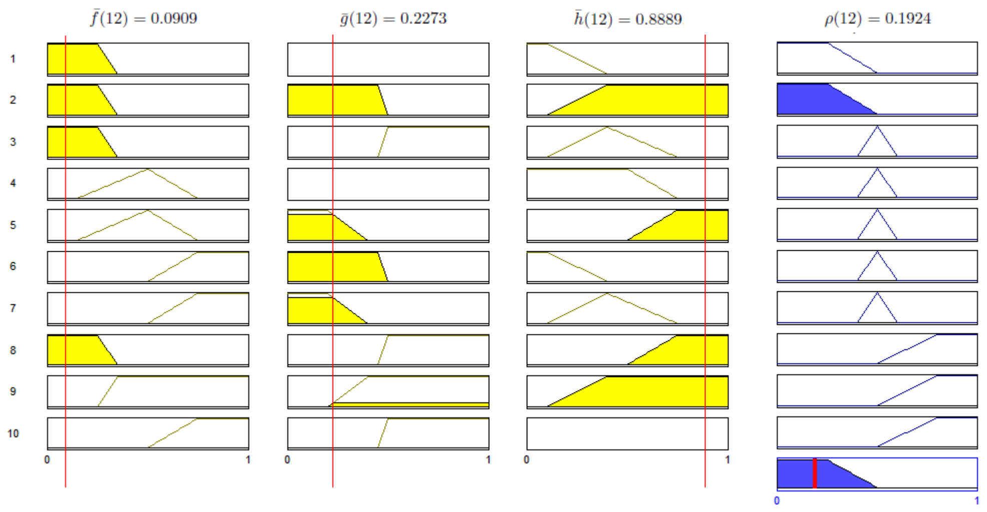

Figure 8: Application of the fuzzy inference system to the column 12 of the pattern "vu"

10. if $\bar{f}(i)$ and $\bar{g}(i)$ are High, then $\rho(i)$ is High.

The touching characters in dataset B are correctly segmented in the 96.1% of the cases. For evaluating the correctness of the segmentation, we used a pattern recognition algorithm constructed by a neural network trained on the characters that compose the dataset B, in fashion of what done by other works reviewed in [53], and obtaining comparable results. We consider that touching character are correctly segmented when the pattern recognition algorithm correctly recognizes the characters after the segmentation.

Simulations show that the fuzzy combination of the functions f, g, h improves the correct identification of the cutting column with respect to their separated use.

To assess the performances of the fuzzy strategy compared to the usage on only the functions g and h, a numerical example is reported below. Let us consider the touching characters "vu" (font Times New Roman) depicted in Figure 7. Our fuzzy routine correctly identifies the cutting column as the column 12 which is assigned the minimum value of ρ among all the columns of the pattern. Specifically, we obtain $\rho(12) = 0.1924$. The fuzzy procedure performance is shown in Figure 8, with application to the column 12. On the other hand, both g and h separately locate the column 16 as the cutting column. Indeed, we can observe that, e.g., $h(16) = 8$ that is greater than $h(i)$, for $i = 1, ..., 21$, $i \neq 16$, for example $h(12) = 0$. In Table 1, we report the values of $\bar{f}, \bar{g}, \bar{h}, \rho$ for each column of the previous pattern (except for the first and the last column that are not surely cutting columns).

3.3 Tests on Latin handwritten characters

In the following, we perform segmentation of touching characters from the dataset. Similarly to what described in the previous section, fuzzy sets, membership functions, and

Table 1: Values of $\bar{f}$, $\bar{g}$, $\bar{h}$, ρ for the columns of the touching characters “vu” (excluded first and last column)

Column	$\bar{f}$	$\bar{g}$	$\bar{h}$	ρ
2	0.8182	0.4545	0.7778	0.7849
3	0.7273	0.6818	0.8333	0.8123
4	0.6364	0.8409	0.9167	0.7908
5	0.5455	0.8523	0.9111	0.8073
6	0.4545	0.8333	0.9333	0.8102
7	0.3636	0.6818	0.8148	0.7949
8	0.2727	0.6818	0.8889	0.8047
9	0.1818	0.6818	0.9259	0.8169
10	0.0909	0.5303	0.8889	0.8169
11	0	0.2273	0.7778	0.1984
12	0.0909	0.2273	0.8889	0.1924
13	0.1818	0.2273	0.1111	0.2078
14	0.2727	0.9343	0.9722	0.8047
15	0.3636	0.9205	1	0.7949
16	0.4545	0	0	0.5000
17	0.5455	0	0.7778	0.6305
18	0.6364	0.3788	0.9556	0.7302
19	0.7273	0.9091	0.9753	0.8123
20	0.8182	1	0.9877	0.8169

fuzzy rules have been defined accordingly to expert based choices and further optimized leveraging the PSO algorithm. Patterns in the dataset A are greatly different from the ones in dataset B. For instance, touching positions in dataset B are often near to the center of the pattern. In the case of dataset A, cutting columns may occur more frequently at high distance from the center. On the other hand, the peak to valley function seems to have better performances in the case of the dataset A. Taking this into account, the optimization conducted by using the PSO algorithm has been strategic as it allowed us to highlight properties and connections among functions f, g, h which are not noticeable at a glance. All these features are reflected in the following definition of fuzzy sets, membership functions, and fuzzy rules.

The fuzzy sets related to $\bar{f}$ are defined by

- if $\bar{f}(i) \leq 0.45$, then distance from the center of the pattern is *Low*;
- if $\bar{0}.25 \leq f(i) \leq 0.55$, then distance from the center of the pattern is *Medium*;
- if $\bar{f}(i) \geq 0.5$, then distance from the center of the pattern is *High*.

For the function $\bar{g}$, we define the following fuzzy sets:

- if $\bar{g}(i) \leq 0.2$, then $\bar{g}(i)$ is *Low*;
- if $0.15 \leq \bar{g}(i) \leq 0.55$, then $\bar{g}(i)$ is *Medium*;
- if $\bar{g}(i) \geq 0.25$, then $\bar{g}(i)$ is *High*.

The fuzzy sets related to $\bar{h}$ are defined by

- if $\bar{h}(i) \leq 0.3$, then $\bar{h}(i)$ is *Low*;
- if $\bar{0}.15 \leq h(i) \leq 0.65$, then $\bar{h}(i)$ is *Medium*;
- if $\bar{h}(i) \geq 0.5$, then $\bar{h}(i)$ is *High*;

Figures 9, 10, and 11 show the membership functions of the previous fuzzy sets.

Finally, for the fuzzy output ρ we define the following fuzzy sets, whose membership functions are depicted in Figure 12:

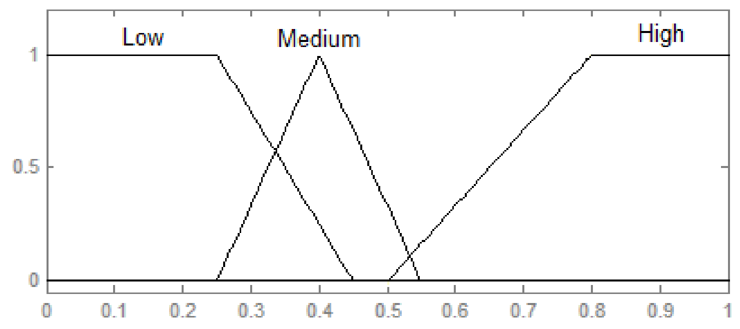

Figure 9: Membership functions of the fuzzy sets related to $\bar{f}$ (for dataset A)

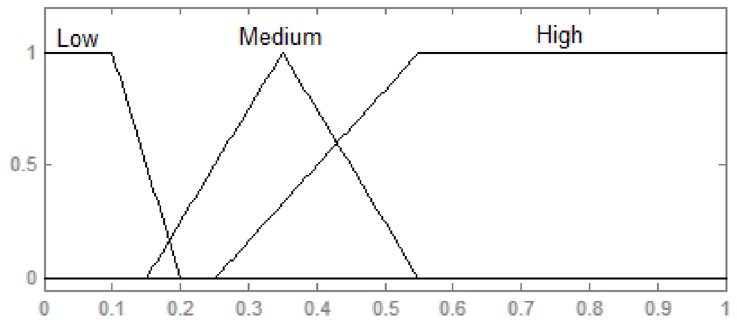

Figure 10: Membership functions of the fuzzy sets related to $\bar{g}$ (for dataset A)

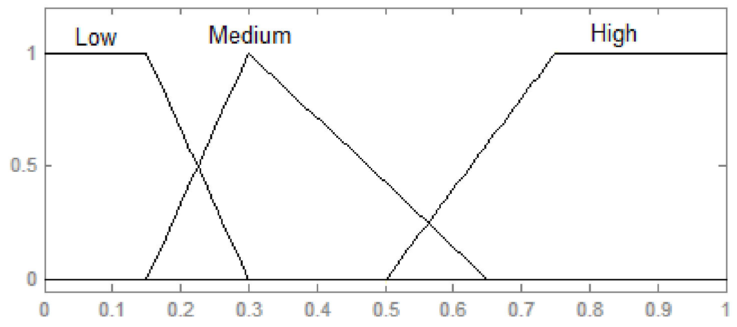

Figure 11: Membership functions of the fuzzy sets related to $\bar{h}$ (for dataset A)

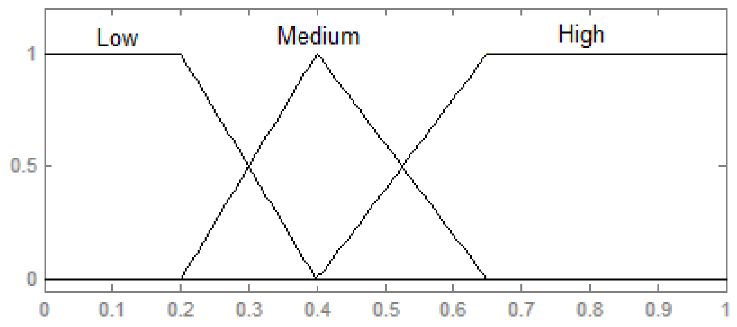

Figure 12: Membership functions of the fuzzy sets related to ρ (for dataset A)

- if $\rho(i) \leq 0.4$, then $\rho(i)$ is *Low*;
- if $0.2 \leq \rho(i) \leq 0.65$, then $\rho(i)$ is *Medium*;
- if $\rho(i) \geq 0.4$, then $\rho(i)$ is *High*;

The inference system is based on the following rules, that combine the three inputs $\bar{f}(i), \bar{g}(i), \bar{h}(i)$ in order to produce the fuzzy output $\rho(i)$, for each column i of the matrix of pixels:

1. if $\bar{f}(i)$ is not High and $\bar{g}(i)$ is not High and $\bar{h}(i)$ is Low, then $\rho(i)$ is Low;
2. if $\bar{f}(i)$ is Low and $\bar{g}(i)$ is Low and $\bar{h}(i)$ is Medium, then $\rho(i)$ is Low;
3. if $\bar{f}(i)$ is Low and $\bar{g}(i)$ is High, then $\rho(i)$ is Medium;
4. if $\bar{g}(i)$ is Medium and $\bar{h}(i)$ is Medium, then $\rho(i)$ is Medium;
5. if $\bar{f}(i)$ is High and $\bar{g}(i)$ is Low, then $\rho(i)$ is Medium;
6. if $\bar{f}(i)$ is Medium and $\bar{g}(i)$ is Low and $\bar{h}(i)$ is Medium, then $\rho(i)$ is Medium;
7. if $\bar{f}(i)$ is High and $\bar{g}(i)$ is Medium and $\bar{h}(i)$ is Low, then $\rho(i)$ is Medium;
8. if $\bar{f}(i)$ is Medium and $\bar{g}(i)$ is High, then $\rho(i)$ is High;
9. if $\bar{f}(i)$ is High and $\bar{g}(i)$ is High, then $\rho(i)$ is High;
10. if $\bar{f}(i)$ is High and $\bar{g}(i)$ is Medium and $\bar{h}(i)$ is High, then $\rho(i)$ is High.

The touching characters in dataset A are correctly segmented in the 81.1% of the cases. Let us remember that in dataset A we have stored the textual descriptor indicating the index of the proper cut column. We consider a correct segmentation when the routine locates such a column. Let us note that CCC database provides a challenging set of characters for segmentation purposes. The only reference where segmentation of touching characters obtained from CCC database is performed similarly to this section is [18] whose authors obtained correct segmentation in the 76.2% of the cases. Let us observe that these results are not directly comparable and are reported just to give a reference of the goodness of our approach, since our dataset A and dataset used in [18] are different, even if they are obtained starting from the same CCC database. Also, authors in [18] took into account percentage of inaccurate segmentation, evaluating when a cutting column is found around the correct position; in this case they report a success percentage of 91.9%, without giving further details. We replicated such an experiment by considering proper, even if inaccurate, segmentation when the cut position identified is distant at most 5 columns from the exact position. In this case, our success percentage is 88.9%. As stated in [49], even though research in handwriting recognition has been an active research area for more than a half century, the maturity of the segmentation techniques is still very low. Our proposed approach focused to improve the segmentation accuracy, achieving comparable, when not better, results.

Some numerical results showing the behavior of our method are reported below, where segmentation is performed on touching characters "eh", "'rt", "'xm", and "ao" depicted in Figures 1a, 13a, 13b, and 13c, respectively.

For the touching characters "eh", our fuzzy routine identifies the correct cutting column as the column 52, whereas function $\bar{f}$ assumes the lowest value in correspondence of the column 56, function $\bar{g}$ in correspondence of columns 34, 35, 36, and 37, and function $\bar{h}$ in correspondence of the column 15.

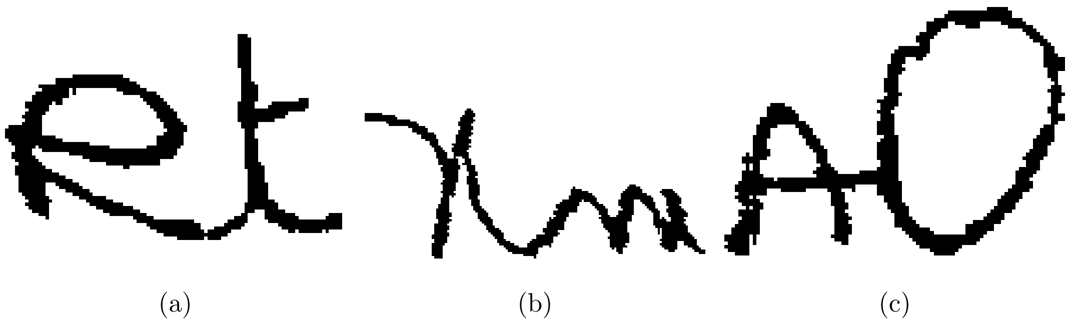

Figure 13: Touching characters “rt”, “xm”, and “ao” extracted from dataset A.

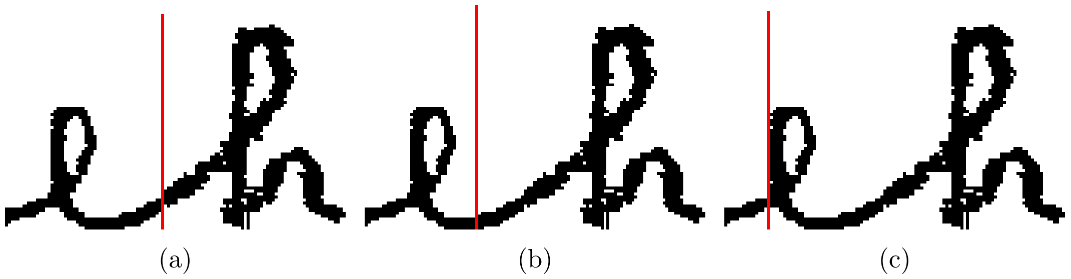

Figure 14: Cutting positions located by ρ (a), $\bar{g}$ (b), and $\bar{h}$ (c).

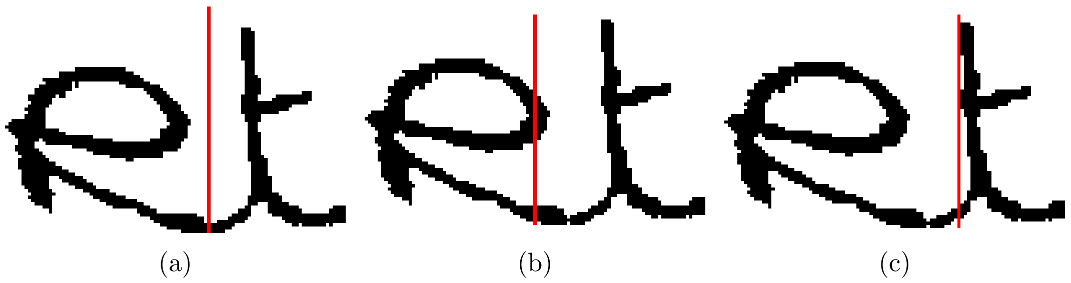

Figure 15: Cutting positions located by ρ (a), $\bar{f}$ (b), and $\bar{h}$ (c).

For the touching characters “rt”, our fuzzy routine identifies the correct cutting column as the column 61, whereas function $\bar{f}$, $\bar{g}$, and $\bar{h}$ identify the cutting position in correspondence of the columns 51, 61, and 70, respectively.

For the touching characters “xm”, our fuzzy routine identifies the correct cutting column as the column 75, whereas function $\bar{f}$, $\bar{g}$, and $\bar{h}$ identify the cutting position in correspondence of the columns 79, 72, and 137, respectively.

Finally, for the touching characters “ao”, our fuzzy routine identifies the correct cutting column as the column 43, whereas function $\bar{f}$ assumes the lowest value in correspondence of columns 48 and 49, function $\bar{g}$ in correspondence of the column 38, and function $\bar{h}$ in correspondence of the column 43.

In Figures 14 and 15, we show the cutting columns found by ρ, $\bar{g}$, and $\bar{h}$ related to touching characters “eh” and cutting columns found by ρ, $\bar{f}$, $\bar{h}$ related to touching characters “rt”, respectively.

For the sake of readability, we do not report values of ρ, $\bar{f}$, $\bar{g}$, $\bar{h}$ for each column since these patterns have usually more than 100 columns.

4 Conclusion

A fuzzy approach for segmentation of touching characters has been presented. The proposed method combines three classical features of touching characters usually exploited one at a time. Experiments have been conducted on two very different datasets composed by Latin printed and handwritten characters, respectively. Fuzzy sets, membership functions, and fuzzy rules, which characterize the fuzzy inference scheme, have been properly constructed by means of expert–based choices, heuristic criteria, and the PSO algorithm for each dataset. Numerical results are encouraging and show that the proposed method has an optimal capability of correctly separating touching characters and may be adjusted for very different varieties of characters (not only for the types considered in our experiments).

Our research activity have shown that even by using alone the functions f, g, h it is sometime possible to correctly segmenting touching characters. In fact, some experiments (not presented here) have shown that adding other inputs to the fuzzy routine does not provide significant improvements. This is surely a perspective that should be further investigated and motivated. Indeed, looking at perspective advancements, the following issues could be addressed in future works:

- study of characterization of performance improvements when adding further features as inputs in the fuzzy inference system;
- experiments on further datasets not involving Latin characters;
- experiments on segmentation of formulae, taking also into account possibility of segmenting touching characters vertically, horizontally and diagonally;
- use of fuzzy models different form the Mamdani one (as, e.g., the Sugeno model).

Acknowledgments

This work has been developed in the framework of an agreement between IRIFOR/UICI (Institute for Research, Education and Rehabilitation/Italian Union for the Blind and Partially Sighted) and Turin University.

This research has been partly supported from the National Thematic Laboratory "AsTech" of CINI.

Special thanks go to Dr. Tiziana Armano and Prof. Anna Capietto for their support to this work.

References

[1] V. Alexandrov, *Using critical points in contours for segmentation of touching characters*, Proc. of the 5th Int. Conference on Computer Systems and Technologies, 1–5, New York, 2014.

[2] V. Bansal, R. M. K. Sinha, *Segmentation of touching and fused Devanagari characters*, Pattern Recognition, Vol. **35**, 875–893, 2002.

[3] T. A. Bayer, U. H. G. Krebel, *Cut classification for segmentation*, IEEE Proc. of 2th International Conference on Document Analysis and Recognition (ICDAR), 565–568, 1993.

[4] V. Bansal, R. M. K. Sinha, *Segmentation of touching and fused devanagari characters*, Pattern recognition, Vol. **35**, No. **4**, 875–893, 2002.

[5] F. Camastra, M. Spinetti, A. Vinciarelli, *Offline cursive character challange: a new benchmark for machine learning and pattern recognition algorithms*, Proc. of the 18th Int. Conference on Pattern Recognition, Vol. **2**, 913–916, 2006.

[6] R. G. Casey, E. Lecolinet, *A survey of methods and strategies in character segmentation*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. **18**, No. **7**, 690–706, 1996.

[7] L. A. Fletcher, R. Kasturi, *A robust algorithm for text string separation from mixed text-graphics images*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. **20**, No. **6**, 910–918, 2002.

[8] V. Frinken, A. Fischer, R. Mammatha, H. Brunke, *A novel word spotting method based on recurrent neural networks*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. **34**, No. **2**, 211–224, 2011.

[9] U. Garain, B. B. Chaudhuri, *Segmentation of touching symbols for OCR of printed mathematical expressions: an approach based on multifactorial analysis*, IEEE Proc. of 8th International Conference on Document Analysis and Recognition (ICDAR), Vol. **1**, 177–181, 2005.

[10] U. Garain, B. B. Chaudhuri, *Segmentation of touching characters in printed Devnagari and Bangla scripts using fuzzy multifactorial analysis*, IEEE Trans. on Systems, Man and Cybernetics, Vol. **32**, No. **4**, 449–459, 2002.

[11] S. He, M. Wiering, L. Schomaker, *Junction detection in handwritten documents and its application to writer identification*, Pattern Recognition, Vol. **48**, 4036–4048, 2015.

[12] J. F. Hebert, M. Parizeau, N. Ghazzali, *Learning to segment cursive words using isolated characters*, Proc. of Conference on Vision Interface, 33–40, 1999.

[13] M. K. Jasim, A. H. Al–Saleh, A- Aijanaby, *A fuzzy based feature extraction approach for handwritten characters*, International Journal of Computer Science, Vol. **10**, No. **4**, 208–215, 2013

[14] M. C. Jung, Y. C. Shin, S. N. Srihari, *Machine printed character segmentation method using side profiles*, Proceedings of IEEE International Conference onSystems, Man and Cybernetics, New York, 863–867, 1999.

[15] S. Kahan, *On the recognition of printed characters of any font and size*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. **9**, No. **2**, 274–288, 1987.

[16] J. Kennedy, R. Eberhart, *Particle swarm optimization*, IEEE Int. Conference on Neural Networks, Perth, Australia, Vol. IV, 1942–1948, 2012.

[17] K. K. Kim, J. H. Kim, C. Y. Suen, *Recognition of unconstrained handwritten numeral strings by composite segmentation method* In Pattern Recognition, 2000. Proceedings. 15th International Conference on, Vol. **2**, 594–597. IEEE, 2000.

[18] F. Kurniawan, M. S. M. Rahim, D. Daman, A. Rehman, D. Mohamad, S. M. Shamsuddin, *Region–based touched character segmentation in handwritten words*, International Journal of Innovative Computing, Information and Control, Vol. **7**, No. **6**, 3107–3120, 2011.

[19] M. Kumar, M. K. Jindal, R. K. Sharma, *Segmentation of isolatedtouching characters in offline handwritten Grumukhi script recognition*, Int. J. of Information Technology and Computer Science, Vol. **2**, 58–63, 2014.

[20] H. Lee, B. Verma, *Binary segmentation algorithm for English cursive handwriting recognition*, Pattern Recognition, Vol. **45**, 1306–1317, 2012.

[21] J. Liang, I. T. Phillips, R. M. Haralick, *An optimization methodology for document structure extraction on Latin character documents*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. **23**, No. **7**, 719–734, 2002.

[22] S. Liang, M. Shridhar, M. Ahmadi, *Segmentation of touching characters in printed document recognition*, Pattern Recognition, Vol. **27**, No. **6**, 825–840, 1994.

[23] G. Louloudis, B. Gatos, I. Pratikakis, C. Halatsis, *Text line and word segmentation of handwritten documents*, Pattern Recognition, Vol. **42**, 3169–3183, 2009.

[24] Y. Lu, *Machine printed character segmentation – an overview*, Pattern Recognition, Vol. **28**, No. **1**, 67–80, 1995.

[25] Y. Lu, *On the segmentation of touching characters*, IEEE Proc. of 2th International Conference on Document Analysis and Recognition (ICDAR), 440–443, 1993.

[26] Z. Lu, Z. Chi, W. C. Siu, P. Shi, *A background–thinning–based approach for separating and recognizing connected handwritten digit strings*, Pattern Recognition, Vol. **32**, 921–933, 1999.

[27] Y. Lu, M. Shridhar, *Character segmentation in handwritten words – an overview*, Pattern Recognition, Vol. **29**, No. **1**, 77–96, 1996.

[28] E. H. Mamdani, S. Assilian, *An experiment in linguistic synthesis with a fuzzy logic controller*, International Journal of Man–Machine Studies, Vol. **7**, No. **1**, 1–13, 1975.

[29] R. Manmatha, J. L. Rothfeder, *A scale space approach for automatically segmenting words from historical handwritten documents* IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. **27**, No. **8**, 1212–1225, 2005.

[30] T. Mondal, N. Ragot, J. Y. Ramel, U. Pal, *Flexible sequence matching technique: an effective learning–free approach for word spotting*, Pattern Recognition, Vol. **60**, 596–612, 2016.

[31] R. Nachar, E. Inaty, P. J. Bonnin, Y. Alayli, *Breaking down Captcha using edge corners and fuzzy logic segmentation/recognition technique*, Security and Communication Networks, Vol. **8**, 3995–4012, 2015.

[32] S. Naz, H. Majeed, H. Irshad, *Image segmentation using fuzzy clustering: a survey*, Proc. of 6th International Conference on Emerging Technologies (ICET), 181–186, 2010.

[33] A. Nomura, K. Michishita, S. Uchida, M. Suzuki, *Detection and segmentation of touching characters in mathematical expressions*, IEEE Proc. of 7th International Conference on Document Analysis and Recognition (ICDAR), Vol. **1**, 126–130, 2003.

[34] N. Otsu, *A treshold selection method from gray–level histograms*, IEEE Trans. Sys., Man., Cyber., Vol. **9**, No. **1**, 62–66, 1979.

[35] U. Pal, A. Belad, C. Choisy, *Touching numeral segmentation using water reservoir concept*, Pattern Recognition Letters, Vol. **24**, 261–272, 2003.

[36] S. Pravesjit, A. Thammano, *Touching character segmentation method of archaic Lanna script*, Chapter E–business and Telecommunication, Vol. **314**, Series Communications in Computer and Information Science, 400–408, 2012.

[37] A. Rehman, F. Kurniawan, D. Mohamad, *Off–line cursive handwriting segmentation: a heuristic rule–based approach*, Journal of Institute of Mathematics and Computer Science, Vol. **19**, No. **2**, 135–140, 2008.

[38] P. P. Roy, U. Pal, J. Llados, *Proccedings of the IEEE SiRecognition of multi–oriented touching characters in graphical documents*, Proceedings of the IEEE Sixth Indian Conference on Computer Vision, Graphics and Image Processing, 297–304, 2008.

[39] P. P. Roy, U. Pal, J. Lladós, M. Delalandre, *Multi-oriented touching text character segmentation in graphical documents using dynamic programming*, Pattern Recognition, Vol. **45**, No. **5**, 1972–1983, 2012.

[40] P. P. Roy, U. Pal, J. Llados, M. Delandre, *Multi–oriented touching character segmentation in graphical documents using dynamic programming*, Pattern Recognition, Vol. **45**, 1972–1983, 2012.

[41] T. Saba, G. Sulong, A. Rehman, *A survey on methods and strategies on touched characters segmentation*, International Journal of Research and Reviews in Computer Science, Vol. **2**, No. **1**, 103–114, 2010.

[42] T. Saba, G. Sulong, A. Rehman, *Non–linear segmentation of touched Roman characters based on genetic algorithm*, Int. J. on Computer Science and Engineering, Vol. **2**, No. **6**, 2167–2172, 2010.

[43] J. Sadri, M. R. Yeganehzad, J. Saghi, *A novel comprehensive database for offline persian handwriting recognition*, Pattern Recognition, Vol. **60**, 378–393, 2016.

[44] R. Sarkar, B. Sen, N. Das, S. Basu, *Handwritten Devanagari script segmentation: a non–linear fuzzy approach*, Proc. of IEEE Conference on AI Tools and Engineering (ICAITE), 2008.

[45] N. Sharma, P. Shvakumara, U. Pal, M. Blumenstein, C. L. Tan, *A new method for character segmentation from multi–oriented video words*, IEEE Proc. of 12th International Conference on Document Analysis and Recognition (ICDAR), Vol. **1**, 413–417, 2013.

[46] Z. Shi, V. Govindaraju, *Line separation for complex document images using fuzzy runlength*, IEEE Proc. of the 1st International Workshop on Document Image Analysis for Libraries, 306–312, 2004.

[47] N. Stamatopoulos, B. Gatos, S. J. Perantonis, *A method for combining complementary techniques for document image segmentation*, Pattern Recognition, Vol. **42**, 3158–3168, 2009.

[48] O. J. Tobias, R. Seara, *Image segmentation by histogram thresholding using fuzzy sets*, IEEE Trans. on Image Processing, Vol. **11**, No. **12**, 2002.

[49] H. Lee, B. Verma, *Binary segmentation algorithm for english cursive handwriting recognition*, Pattern Recognition, Vol. **45**, No. **4**, 1306–1317, 2012.

[50] X. Wei, S. Ma, Y. Jin, *Segmentation of connected Chinese characters based on genetic algorithm*, IEEE Proc. of 8th International Conference on Document Analysis and Recognition (ICDAR), Vol. **2**, 645–649, 2005.

[51] J. J. Weinman, E. L. Miller, A. R. Hanson, *Text recognition using similarity and lexicon with sparse belief propagation*, IEEE Pattern Analysis and Machine Intelligence, Vol. **31**, 1733–1746, 2009.

[52] S. Zhao, Z. Chi, P. Shi, H. Yan, *Two-stage segmentation of unconstrained handwritten chinese characters*, Pattern Recognition, Vol. **36**, No. **1**, 145–156, 2003.

[53] J. Zhou, A. Krzyzak, C. Y. Suen, *Verification–a method of enhancing the recognizers of isolated and touching handwritten numerals*, Pattern Recognition, Vol. **35**, No. **5**, 1179–1189, 2002.