A fuzzy approach to segment touching characters

A Fuzzy Approach to Segment Touching Characters

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

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

[1]CINI Assistive Technologies National Lab &
DAUIN, Politecnico di 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

Despite many years of research, correct and reliable segmentation of touching characters is still an hard task to solve. In the recent years, many methods and algorithms have been proposed; nevertheless the problem is still open. In this paper, we propose a novel method based on fuzzy logic that combines three different techniques to segment touching characters. These techniques have already been used in other studies but they have never been used all together. We propose a 3–input/1–output fuzzy inference system with fuzzy rules that are specifically optimized to segment touching Latin characters. The method is applicable to both printed and handwritten characters. We discuss the performances of our method by comparing it with state of the art. Results show that our method provide a better accuracy to segment characters even with noisy touching characters.

Keywords: characters segmentation, fuzzy logic, OCR, touching characters

1 Introduction

Automatic recognition of printed and handwritten characters remains a challenging problem in pattern recognition. The large majority of the existing Optical Character Recognition (OCR) software are based on two techniques: characters segmentation and pattern recognition. Characters segmentation techniques are used, within the OCR process, to recognize atomic blocks (e.g., a word, a character, a mathematical symbol).

When adjacent characters are connected, the segmentation of a text into single characters becomes complex (Roy et al., 2008), (Bansal and Sinha, 2002). A proper identification of connections among characters is crucial for subsequent characters recognition, since a wrong segmentation decreases the accuracy of pattern recognition algorithms (Zhao et al., 2003). In fact, many researches have demonstrated how small errors in characters segmentation affect the overall performance of the OCR system more than the degradation of the starting image (Grafmuller and Beyerer, 2013), (Jung et al., 1999). Thus, segmentation of touching characters is fundamental for OCR systems that aim to achieve good accuracy (Casey and Lecolinet, 1996).

Given the relevance of this task, several methods for performing optimal segmentation of touching characters have been developed in the last years. Common techniques for characters segmentation exploit vertical projections, pitch estimation or character size, contour analysis, or segmentation–recognition coupled techniques (Lu and Shridhar, 1996), (Lu, 1995). However, the state of the art does not provide a comprehensive answer to the problem. As a consequence, at the moment, there is no standard approach for the segmentation of touching characters.

The main contributions of this work can be summarized as follows: we propose a fuzzy approach for fast and reliable segmentation of touching characters (considering also noise); we provide two novel datasets of touching characters (latin printed and handwritten characters) to overcome the lack of good and standardized datasets for fair evaluation of approaches in this research area.

The paper is organized as follows. Section 2 is dedicated to related works with a focus on characters segmentation and fuzzy logic. In Section 3, we present the fuzzy strategy developed for performing segmentation of touching characters. In section 4, we present the numerical results that show the effectiveness of the proposed method. Specifically, in Section 4.1, we describe the datasets used for simulations, while Sections 4.2 and 4.3 are devoted to test the method on datasets of Latin printed and handwritten characters, respectively. Finally, in Section 5 we draw conclusion and present future works.

2 Related work and background

2.1 Characters segmentation

Algorithms for performing characters extraction and segmentation usually need a preprocessing step, called *binarization*, where the input HSV image is firstly converted to grayscale and then binarized through a dedicated thresholding process. The result is a matrix whose entries are conventionally set to 1 for foreground pixels (black) and 0 for background pixels (white). Clearly, the thresholding process has a noticeable impact on the quality of the binary outcome. In our experiments we rely on the classical thresholding method introduced by Otsu in (Otsu, 1979).

Typically, researchers have designed strategies and functions to score each column (or, indifferently, row or diagonal) of the binarized matrix by leveraging on features that characterize the beginning and the ending of characters. Finally, cut positions are chosen on the basis of these scores (usually, the highest ranked columns). Common used functions are the ratio of the second difference of the vertical projection (e.g., (Kahan, 1987)) and the "peak–to–valley" function (e.g., (Lu, 1993)). Other functions are, e.g., based on the number of black pixels, the number of white pixels counted from the top of the column to the first black pixel, the crossing count (i.e., the number of black to white transitions), the number of identical black (white) pixels with left (right) column, the width to height ratio for the remaining left (right) pattern after cutting (e.g., (Bayer and Krebel, 1993)). Yet another different approach can be found in (Garain and Chaudhuri, 2002b), 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 is a good candidate for being cut.

In (Bansal and Sinha, 2002), the authors revised the work presented in (Casey and Lecolinet, 1996), stating that strategies for characters segmentation can be classified into three categories:

1. classical approach based on typical features of characters;
2. recognition-based segmentation;
3. holistic approaches.

In the last years, several researchers have addressed the challenges behind the three above mentioned categories. For instance, Kurniawan et al. (Kurniawan et al., 2011) identify touching positions in Latin handwritten characters by means of self organizing feature maps and a region–based approach. In (Liang et al., 2002), the authors deal also with segmentation of Latin handwritten texts. A different approach involving thinning algorithms can be found in (Lu et al., 1999). Further approaches rely on contour analysis of the connected components (Kim et al., 2000). Lacerda and Mello (Lacerda and Mello, 2013) developed an algorithm for the segmentation of connected handwritten digits by using self–organizing maps. Roy et al. (Roy et al., 2012) addressed the problem of segmenting touching characters with different orientations. In (Saba et al., 2010) and (Wei et al., 2005), authors developed approaches by leveraging on genetic algorithms. Louloudis et al. (Louloudis et al., 2009) performed text lines and words segmentation of handwritten documents by applying the Hough transform. Authors in (Manmatha and Rothfeder, 2005) addressed the problem of automatically segmenting words in historical handwritten documents. The water reservoir algorithm has been exploited in (Kumar et al., 2014) and (Pal et al., 2003). In (Stamatopoulos et al., 2009) and (Bansal and Sinha, 2002), methods derived by combining different segmentation techniques have been presented. Further works in this research field can be found in (He et al., 2015), (Caballero et al., 2012), (Frinken et al., 2011), (Sedighi and Vafadust, 2011), (Camastra et al., 2006) and (Alhajj and Elnagar, 2005). Works, such as (Garain and Chaudhuri, 2005) and (Nomura et al., 2003), also deal with the segmentation of characters and symbols within mathematical expressions. Furthermore, authors in (Olszewska, 2015) present a method based on active contours to automatically extract characters. Specifically, after the binarization of the image, the algorithm delineates the boundaries of the characters by working on active contours. Other researchers have proposed approaches that rely on geometrical features: it is for instance the case of (Olszewska, 2012), (Olszewska and McCulskey, 2011), (Qu and Zheng, 2011), (Song and Wang, 2009), and (Lin and Huang, 2007).

The work presented in this paper contains some important novelties with respect to this prior art: we propose to combine features usually exploited one at a time by means of an original fuzzy logic approach. The reason to investigate fuzzy logic derives from the lack of an objective standard categorization of the features that identify cutting positions within touching characters. Thus, in this context, fuzzy logic can be very useful to capture and to code expert–based knowledge.

2.2 Fuzzy logic

Fuzzy logic was introduced by Zadeh (Zadeh, 1965) and differs from classical logic since a truth degree (usually a real number between 0 and 1) can be assigned to a proposition, in opposition to classical logic where a proposition can only be true or false. Similarly, given a collection of objects X, a fuzzy set F is defined as a set of ordered couples $F = \{(x, \mu(x)) : x \in X\}$, where $\mu : X \to [0, 1]$ is called the membership function of the fuzzy set F. In other words, the membership function assigns a value of belonging to F for each element in X. Thus, an element belongs to a fuzzy set with a certain membership degree, in opposition to classical sets, where elements belongs or not to the set.

Fuzzy logic has already been exploited to perform characters segmentation (Zimmermann, 1996). For instance, Garain and Chaudhuri in (Garain and Chaudhuri, 2002b) used fuzzy multifactorial analysis to combine some of the features described in the previous section. In (Naz et al., 2010), a survey on image segmentation techniques that use fuzzy clustering is presented. A non–linear fuzzy approach can be found in (Sarkar et al., 2008). In (Nachar et al., 2015), authors used edge corners and fuzzy logic to develop segmentation techniques with the aim of recognizing odd-shaped and unconventional characters; this technique has been exploited to break down Captcha codes. In particular, they propose a fuzzy logic–based scheme to match characters against known patterns using their edge corners. Thus, authors constructed a dataset of images used as building blocks to form the warped image. The warped image is fed to the recognition algorithm attempting to correctly recognize the characters. Hence, they mainly use fuzzy logic as a clustering technique where the features for the clustering process are the edge corners. Further approaches in this area can be found in (Tobias and Seara, 2002) and (Fonseca and Jorge, 2000).

In this paper, we develop an original fuzzy algorithm that differs from the state of the art in several aspects. Firstly, we combine, by means of a fuzzy strategy, 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

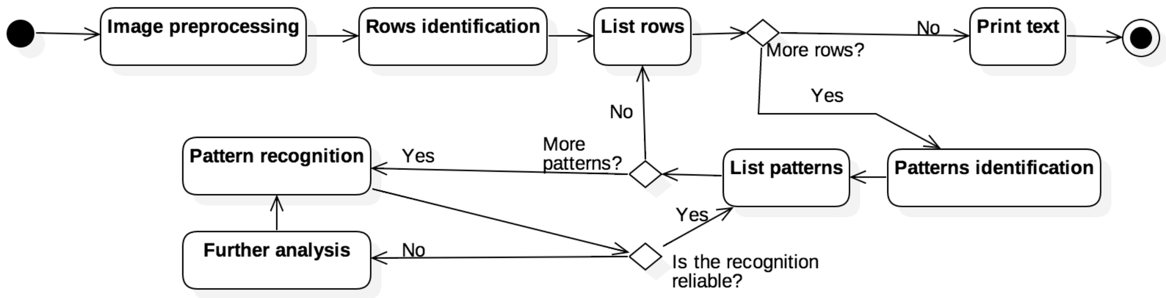

Figure 1: Statechart diagram of the overall proposed approach.

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 features of the dataset. In other words, the parameters of the method are extracted a priori considering the different characters in the dataset. Thus, we do not use fuzzy logic for developing a clustering technique as in many of the works discussed previously.

3 The fuzzy strategy

In this section, without loss of generality, we present a fuzzy strategy meant to be applied column–wise to binary images, similar considerations hold when dealing with row–wise or diagonal–wise image cutting. We aim at identifying the touching position in the input image, i.e., the column of the image matrix where the two characters touch each other, to ensure the best performances for the subsequent recognition phase on the two resulting sub–matrices.

The proposed strategy to segment touching characters defines a function based on features that characterize a touching position. Then, such a function is evaluated for each column of the matrix and the cut position is chosen. The cut position (or cutting column) is the number of the column of the matrix chosen for dividing the pattern into two sub–patterns (i.e., two sub–matrices of the starting matrix representing the whole pattern).

This strategy has been developed as a module enabling the subsequent development of a complete and automatic OCR to be applied within challenging and tricky input documents. Purpose of this OCR will be to assist visually impaired and blind students in dealing with scientific subjects. When dealing with mathematical texts, touching characters represent a considerable challenge for any OCR. Our approach, depicted by the block diagram in Figure 1, aims at minimizing the recognition error in an input image, such as a page of scientific papers or books. After the phase of pre–processing of the input image (needed for the binarization of the image and might comprise de–noising and/or de–warping as well), the image is scanned by rows. For each row, an elementary analysis is conducted for identifying patterns (e.g., segmenting single characters) that will be subsequently recognized by means of machine learning engines (e.g., artificial neural networks). For each pattern, the machine learning engine produces a guess on its content as well as a confidence index. When this index fails below a reliability threshold, the guess is discarded and the pattern is saved for further analysis. Among the possible reasons for the recognition failure, this present work deals with the case when the pattern is not atomic and has to be non–trivially segmented to expose its constituting characters.

Classical functions for similar tasks are the function g (i.e., the peak–to–valley function) and the function h, which are defined as

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

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

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 suggest 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. In the following, we combine functions f, g, and h by means of a fuzzy strategy that balances these functions.

Let us introduce the notion of a "fuzzy degree" by 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 likely 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 P be the matrix of pixels of the binarized image, where m and n are the number of row and the number of column of P, respectively, and where c is the central column of P. In the following, when we refer to a column i of P, we refer to the i–th column of P, 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 coincides with the central column of P; when n is even, we consider as the central column the mean between $\frac{n}{2}$–th column and $\frac{n}{2}+1$. In this way, for each column i of P, 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 P, instead of using the functions g and h, we consider the normalized functions

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

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

where $\mathcal{C} = \{1, 2, ..., n\}$ is the set of the columns of P, and then we 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 touching positions. Note that functions $\tilde{g}$ and $\tilde{h}$ are well–defined since we consider non-trivial matrices P where at least two columns are different.

Functions $\bar{f}$, $\bar{g}$, $\bar{h}$ are fuzzified by defining convenient fuzzy sets and related membership functions describing when the values of $\bar{f}$, $\bar{g}$, $\bar{h}$ can be considered low enough or not. In particular, for each function, three categories (i.e., three fuzzy sets) named "Low", "Medium" and "High" are defined to describe, in a fuzzy way, whether a value of these functions should be considered low, medium or high. The shapes of the membership functions are constrained to be trapezoidal or triangular. These two shapes are the most typical and spread for membership functions as reported in (Da Costa Lobato et al., 2015), (Yasojima et al., 2013), (Noronha et al., 2013) and (D'Errico and Murru, 2012). Indeed, they are identified by a small number of parameters and are easy to modify, which is helpful for a subsequent phase of optimization. Interested readers are invited to read (Wu, 2012) for further details about the choice of the shapes of membership functions. Then, a fuzzy degree ρ is evaluated combining the functions $\bar{f}$, $\bar{g}$, $\bar{h}$ by means of proper fuzzy rules; fuzzy sets, membership functions and fuzzy rules are detailed in Sections 4.2 and 4.3.

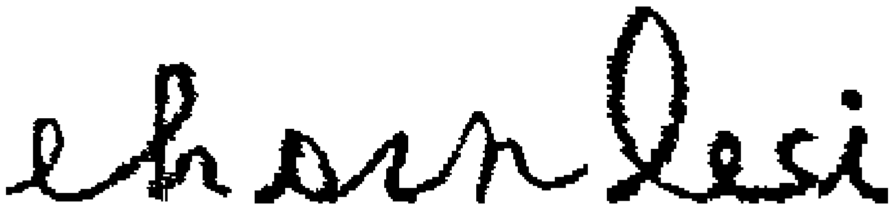
(a) (b) (c)

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

For what concerns the inference engine, we rely to the Mamdani model (Mamdani and Assilian, 1975), with if–then rules, minimax set–operations, sum for composition of activated rules, and defuzzification based on the centroid method. We chose the Mamdani model as it is suitable to capture and code expert–based knowledge; accordingly the system's performance is tuned by means of expert–based choices, heuristic criteria and non–linear optimization methods.

4 Experimental validation of the proposed strategy

We faced the problem of the lack of good and standardized datasets for testing our algorithm: a comprehensive dataset specific on touching characters is still missing. This consideration is shared with many researchers (e.g., (Kurniawan et al., 2011), (Lee and Verma, 2002)). For this reason, the testing, the analysis and the comparison of methods and approaches for character segmentation is troublesome and, in some cases, unfeasible. Actually there exist few datasets, but they are either unfitted for our purposes or not accessible anymore (such as the one discussed in (Roy et al., 2012)). This need forced us to build two novel datasets, which have been designed to be as general as possible to foster further research works; in particular, we created two different datasets of Latin characters, one containing handwritten characters, the other containing machine printed ones. These two datasets are available free of charge on a GitHub repository[1].

Due to the different characteristics of the two datasets, in the remaining parts of the Section we also present the process undertaken to tune and select two different set of best parameters for running our method on the two datasets.

4.1 Synthesis of the datasets

The first dataset, denoted hereby as "dataset A", contains images of handwritten cursive characters that we have built by relying on samples from a standard dataset, as done in (Kurniawan et al., 2011). In particular, we started from CCC dataset (Camastra et al., 2006). CCC dataset contains 57,293 samples of cursive characters that were manually extracted from images coming from different input sources; it includes both upper and lower case letters. Each sample is stored as a binary matrix and each matrix is stored together with the information about the size of the matrix itself and the character that it represents. Starting from the whole dataset, we developed a MATLAB script to randomly extract 1,000 samples by taking care of maintaining an uniform distribution for all the chosen characters. These samples were later combined and merged together to form patterns of two, three, and four touching characters; each pattern is accompanied by a textual descriptor indicating

[1] `https://github.com/guybrush90/OCR`

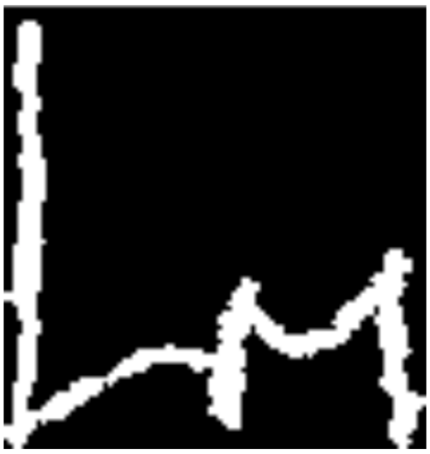

Figure 3: A pattern of two touching characters discarded for the significant difference in heights (shown in negative to remark it has been discarded).

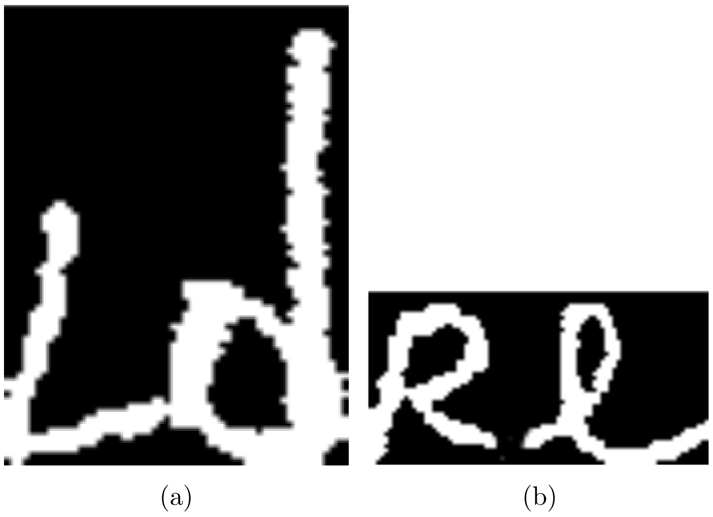

Figure 4: Patterns of two touching characters discarded because found respectively unrealistic (a) or trivial (b) (shown in negative to remark they have been discarded).

the index of the proper cut column (or columns, in the case of patterns with more than two touching characters). One sample of each category of patterns is shown in Figure 2. For instance, the descriptor of the sample represented in Figure 2(a) states that the cut column to properly separate the "e" character from the "h" one is the 52^{nd}.

This merging process was unsupervised and produced also "improper" outcomes, i.e., patterns of touching characters which are not to be expected in a real life scenario. For this reason, prior to the publication of the dataset A, we manually checked all the patterns, discarding some of them as follows.

Firstly, we discarded all the samples presenting a significant difference in the heights of their characters, since we found these patterns unrealistic to happen in real life and inadequate to evaluate the performances of any characters segmentation technique. An example is depicted in Figure 3, where the height of the "L" character is three times the height of the "M" character.

Secondly, we discarded also combinations that seemed impossible to happen in the real world. Words where two contiguous characters do not touch at all (i.e., where characters are well separated) have been discarded as well. Unrealistic patterns were identified taking into account grammatical and writing rules: as a matter of example, the pattern in Figure 4(a) was discarded since a capital consonant "L" is followed by another consonant "d" (this second character being also higher despite being lowercase). Patterns without touching characters were identified simply by means of a visual inspection, as in the case of the

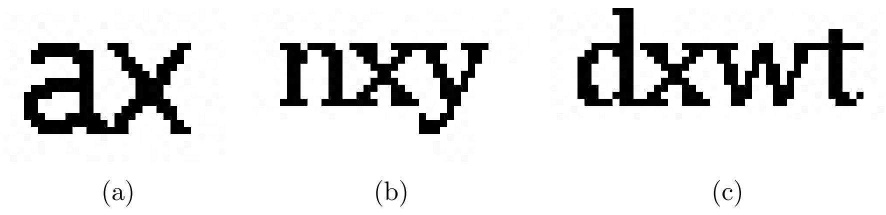

Figure 5: Samples of, respectively, two (a), three (b), and four (c) machine printed touching character patterns.

pattern in Figure 4(b).

At the end, we kept 153 combinations: 139 combinations represent two touching characters and the other are equally divided into three and four touching characters. The disproportion is due to the fact that touching characters occur mostly between two characters (Wei et al., 2005). Moreover, note that the quantity of combinations of touching characters contained in our dataset is compliant to other similar datasets that have been generated by using the CCC dataset (e.g., in (Kurniawan et al., 2011) a dataset of 123 touching characters is used). It is of relevance here to stress the fact that we did not have access to the dataset described in (Kurniawan et al., 2011) while the CCC dataset does not fit completely our purposes because it does not contain touching characters.

The second dataset, denoted hereby as "dataset B", contains images of printed characters and we have built it relying on a second MATLAB script. We have identified a list of six font types (namely Cambria, Candara, Georgia, Lucida Sans Regular, Times New Roman and Verdana Bold) and three sizes (namely 10, 20 and 25). A MATLAB script combines into images the lower characters from the alphabet to form two, three, and four touching characters. Each image is accompanied by a textual descriptor, which indicates the characters represented within it. One sample of each category of patterns is shown in Figure 5. For instance, the descriptor of the sample represented in Figure 5(a) 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 characters. 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.

4.2 Tests on Latin printed characters

In this subsection we discuss the results of our segmentation method focusing on dataset B. In particular, we define the fuzzy sets and membership functions that fuzzify functions $\bar{f}$, $\bar{g}$, $\bar{h}$ and ρ with respect to the dataset.

Given the matrix P defined in Section 3, for each column i of P, the cutting degree $\rho(i)$ is provided by the following inference scheme that takes three inputs $\bar{f}(i)$, $\bar{g}(i)$ and $\bar{h}(i)$ evaluated for each column. The column i with the lowest value of ρ is considered as the cut column. In the following, we define the fuzzy sets and membership functions that fuzzify functions $\bar{f}$, $\bar{g}$, $\bar{h}$ and ρ. As stated previously, we have chosen to define three fuzzy sets named "Low", "Medium", and "High" for each function and trapezoidal and triangular membership functions for these fuzzy sets. The parameters that define the membership functions (and characterize the fuzzy sets) have been initially roughly drafted accordingly to expert–based choices, showing however poor overall outcomes. To boost overall performances of our fuzzy strategy we decided to apply an optimization technique. In particular we developed a MATLAB script that optimizes the construction of the membership functions by means of the Particle Swarm Optimization (PSO) algorithm (Kennedy and Eberhart, 2012). Similarly, the fuzzy rules (described in the remaining of this section) have been tuned using firstly heuristic criteria and secondly the PSO algorithm.

Among several strategies, we chose the PSO algorithm due to its simplicity and adapt-

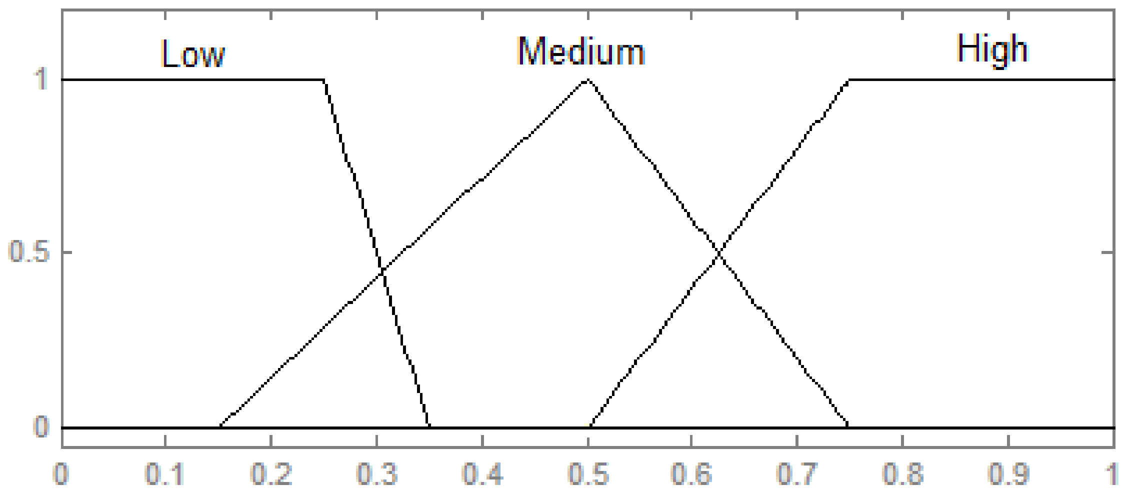

Figure 6: Membership functions of the fuzzy sets related to $\bar{f}$ (for dataset B).

ability in the implementation with respect to other non–linear optimization algorithms (Arora and Gigras, 2013). Moreover, PSO has already been successfully adopted for dealing with mixed variables, both continuous and discrete ones (Hu et al., 2004), as in (Chiaradonna et al., 2015), (Chowdhury et al., 2013) and (del Valle et al., 2008).

The result of the procedure described so far are the fuzzy sets and membership functions used to fuzzify $\bar{f}$, $\bar{g}$ and $\bar{h}$ functions as follow:

- Fuzzification of $\bar{f}$:
  - 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*.
- Fuzzification of $\bar{g}$:
  - 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*.
- Fuzzification of $\bar{h}$:
  - 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*.

Figure 6, 7, and 8 show graphically the membership functions of the fuzzy sets.

Finally, for the fuzzy output ρ we define the following fuzzy sets, whose membership functions are depicted in Figure 9:

- 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 combines 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 P relying on the following rules:

1. if $\bar{f}(i)$ is Low and $\bar{h}(i)$ is Low, then $\rho(i)$ is Low;

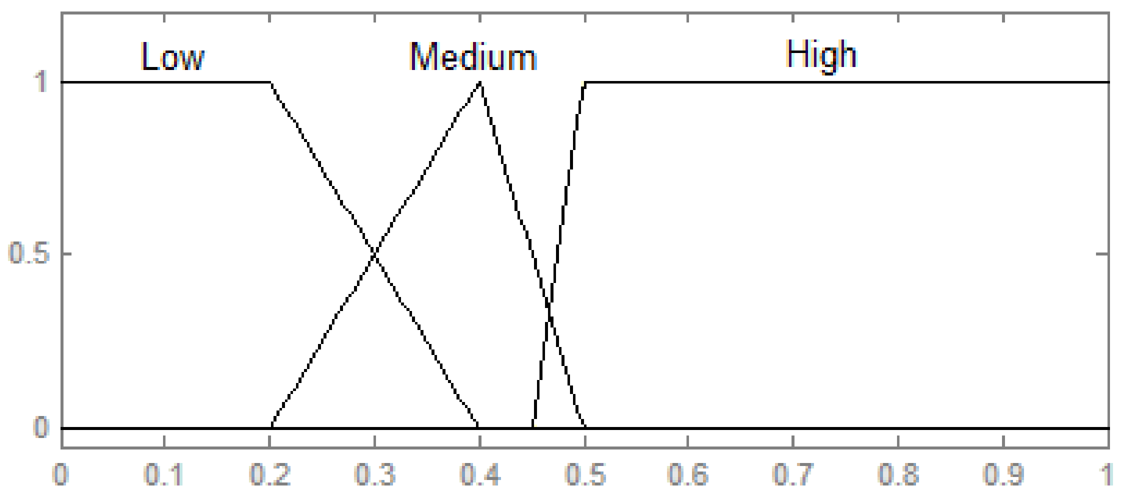

Figure 7: Membership functions of the fuzzy sets related to $\bar{g}$ (for dataset B).

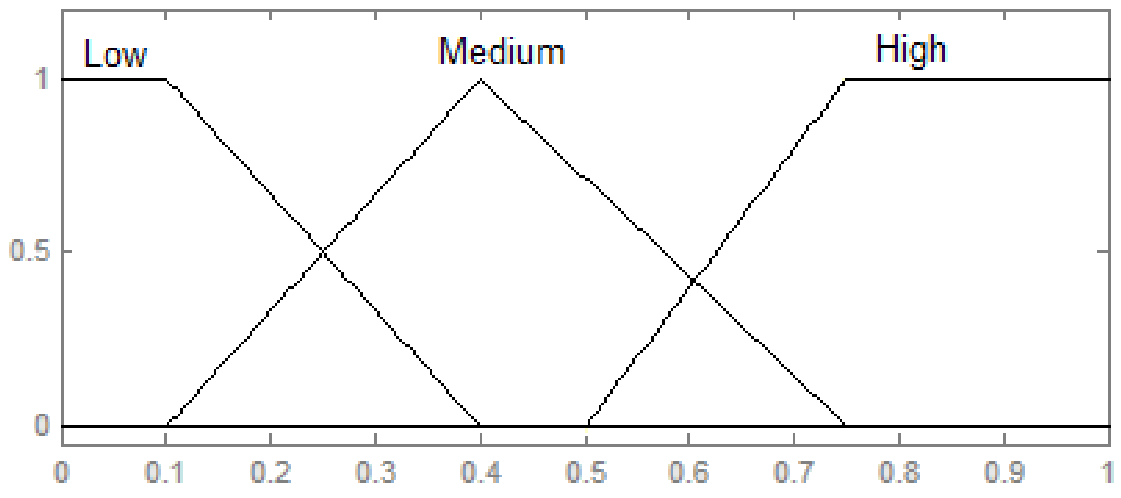

Figure 8: Membership functions of the fuzzy sets related to $\bar{h}$ (for dataset B).

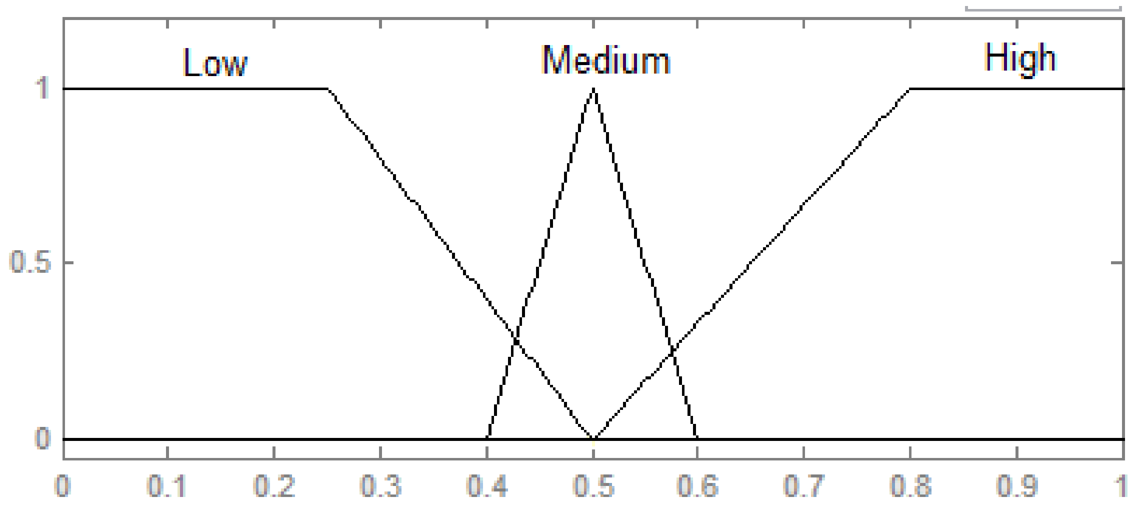

Figure 9: Membership functions of the fuzzy sets related to ρ (for dataset B).

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;
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, as done by other works reviewed in (Zhou et al., 2002).

We consider two touching characters as correctly segmented by our approach only when the pattern recognition algorithm correctly recognizes both of them, considering only true positives. 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 support this statement against, for instance, the usage of only the functions g and h, a numerical example is reported below. Let us consider the touching characters "vu" (Times New Roman font) depicted in Figure 10. 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 11, with application to the column 12. On the other hand, both g and h separately identify the column 16 as the cutting column.

Furthermore, we can observe that the use of the three features implemented by f, g and h appears to be sufficient for obtaining optimal results in the segmentation. To further support this observation, we add a fourth feature in our approach. For instance, we can consider the crossing count as a fourth input in our routine. The crossing count is the number of transitions from white to black pixels, and vice-versa, and it is a common feature used to identify touching position, since, in general, touching position encounters a single black run (Garain and Chaudhuri, 2002a). Here, we consider the fuzzy sets and membership functions for the crossing count function c depicted in Figure 12; they have been constructed using both the heuristic criteria and optimization methods presented previously. The addition of the crossing count as fourth input does not affect the performances of our fuzzy routine: still, the precision of the approach against dataset B does not overpass the 96.1%.

As explanatory example, we report in Table 1 the values of $\bar{f}, \bar{g}, \bar{h}, c$ for each column of the pattern in Figure 10 (excluding the first and the last column, which cannot be identified as cutting columns). In the last two columns, we report the values of the fuzzy output: column ρ_1 shows the result when we consider as inputs only $\bar{f}, \bar{g}, \bar{h}$ and column ρ_2 shows the result when we add as fourth input the crossing count c. The reader can observe that the changes produced by adding the crossing count are not significant in the identification of the cutting column.

4.3 Tests on Latin handwritten characters

In the following, we perform segmentation of touching characters using dataset A. Similarly to what described in the previous section, fuzzy sets, membership functions, and fuzzy rules have been defined accordingly to expert–based choices and further optimized leveraging the

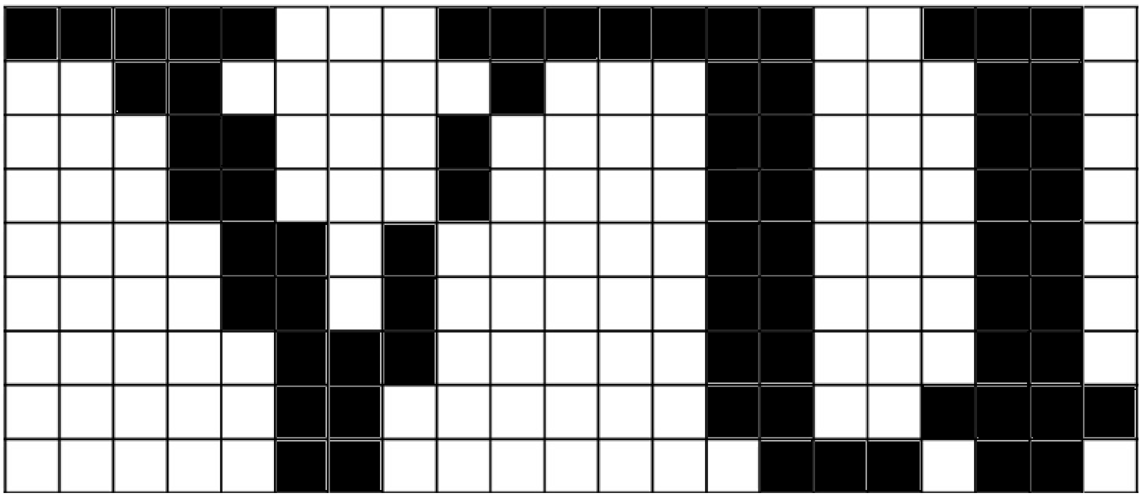

Figure 10: Touching characters “vu” for font Times New Roman and font size of 20.

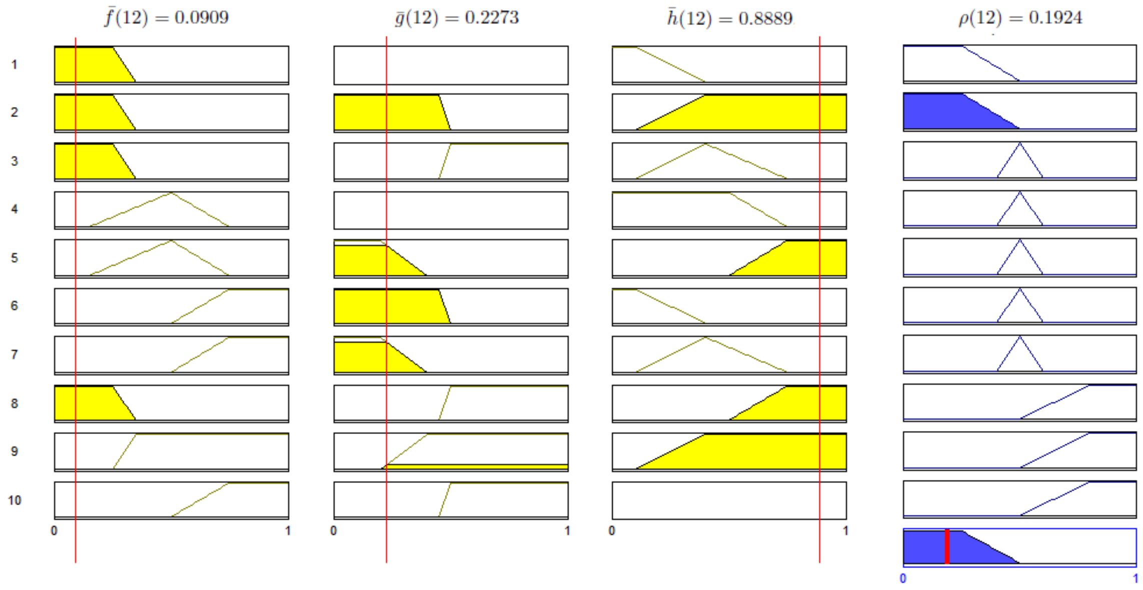

Figure 11: Application of the fuzzy inference system to the column 12 of the pattern “vu” shown in Figure 10.

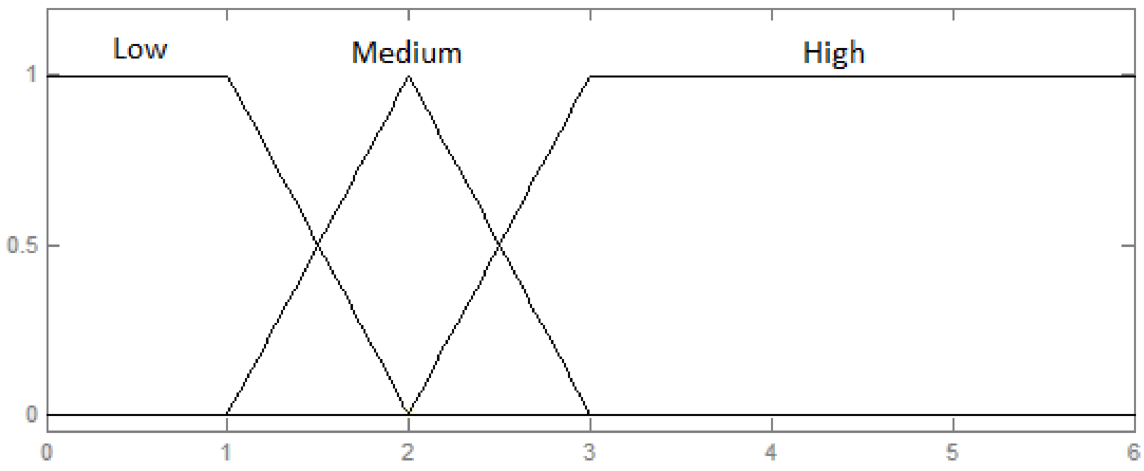

Figure 12: Membership functions of the fuzzy sets related to crossing count c (for dataset B)

Table 1: Values of $\bar{f}$, $\bar{g}$, $\bar{h}$, c for the columns of the touching characters "vu" shown in Figure 10(the first and the last columns are excluded). Values ρ_1 and ρ_2 are the fuzzy outputs when we consider as inputs only $\bar{f}, \bar{g}, \bar{h}$ and when we add as fourth input the crossing count c, respectively.

Column	$\bar{f}$	$\bar{g}$	$\bar{h}$	c	ρ_1	ρ_2
2	0.8182	0.4545	0.7778	1	0.7849	0.7694
3	0.7273	0.6818	0.8333	1	0.8123	0.6820
4	0.6364	0.8409	0.9167	1	0.7908	0.7105
5	0.5455	0.8523	0.9111	3	0.8073	0.8073
6	0.4545	0.8333	0.9333	1	0.8102	0.8102
7	0.3636	0.6818	0.8148	1	0.7949	0.7949
8	0.2727	0.6818	0.8889	2	0.8047	0.8047
9	0.1818	0.6818	0.9259	2	0.8169	0.8169
10	0.0909	0.5303	0.8889	1	0.8169	0.8169
11	0	0.2273	0.7778	1	0.1984	0.1984
12	0.0909	0.2273	0.8889	1	0.1924	0.1924
13	0.1818	0.2273	0.1111	1	0.2078	0.2078
14	0.2727	0.9343	0.9722	1	0.8047	0.8047
15	0.3636	0.9205	1	0	0.7949	0.7949
16	0.4545	0	0	1	0.5000	0.5000
17	0.5455	0	0.7778	1	0.6305	0.6305
18	0.6364	0.3788	0.9556	3	0.7302	0.7749
19	0.7273	0.9091	0.9753	0	0.8123	0.6820
20	0.8182	1	0.9877	0	0.8169	0.7132

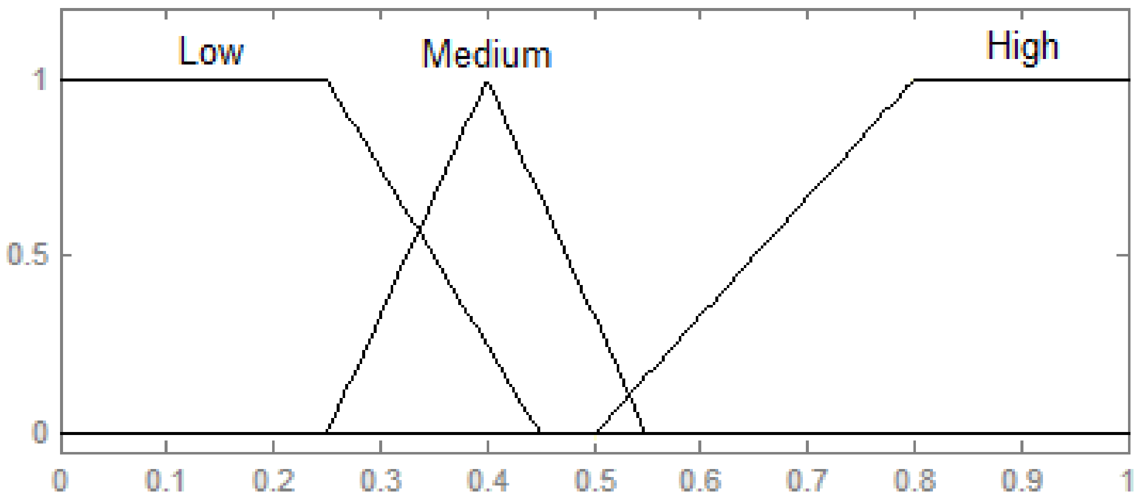

Figure 13: Membership functions of the fuzzy sets related to $\bar{f}$ (for dataset A).

PSO algorithm. Patterns in dataset A are greatly different from 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 dataset A. Taking this into account, the optimization conducted by using the PSO algorithm has been strategic because 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 as follows:

- 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*;

Figure 13, 14, and 15 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 16:

- 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*;

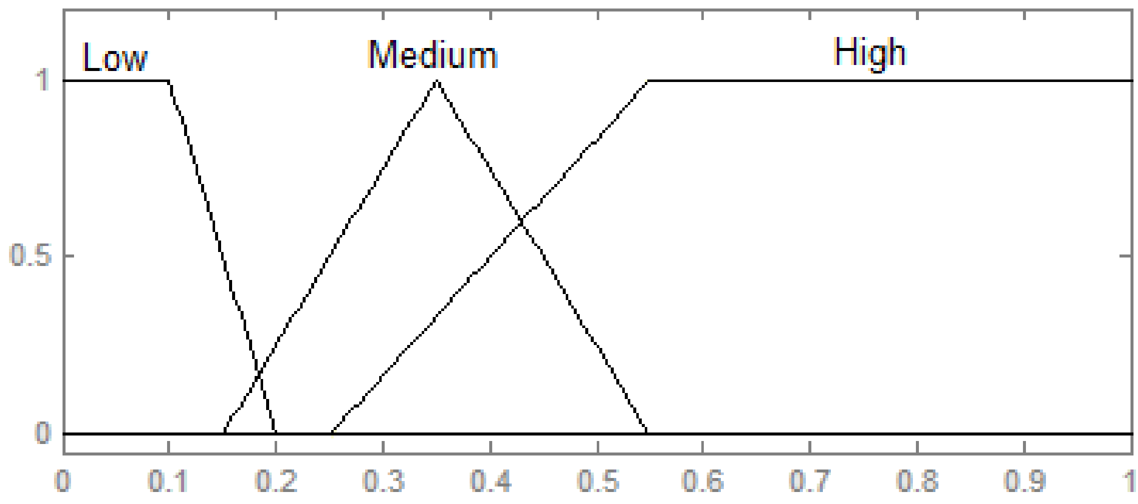

Figure 14: Membership functions of the fuzzy sets related to $\bar{g}$ (for dataset A).

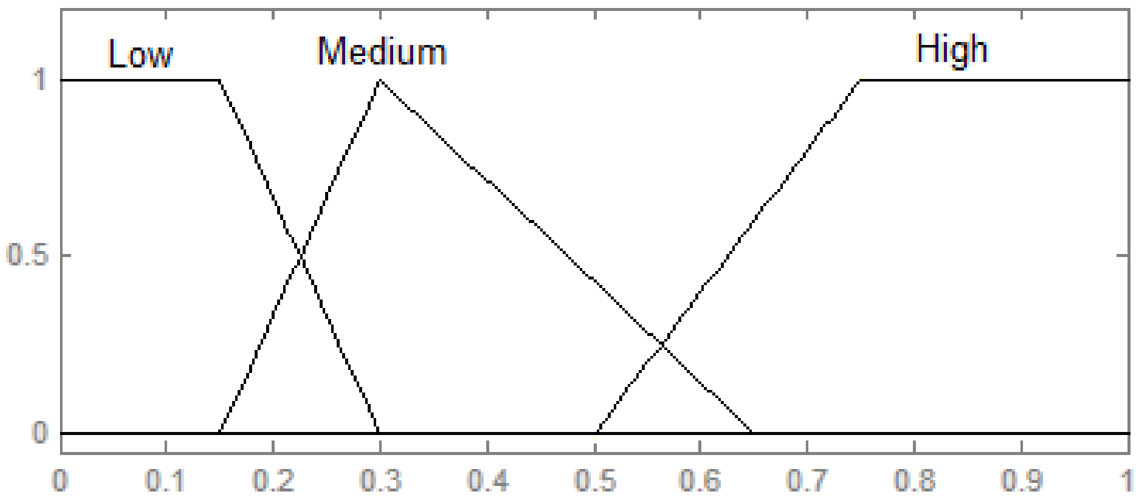

Figure 15: Membership functions of the fuzzy sets related to $\bar{h}$ (for dataset A).

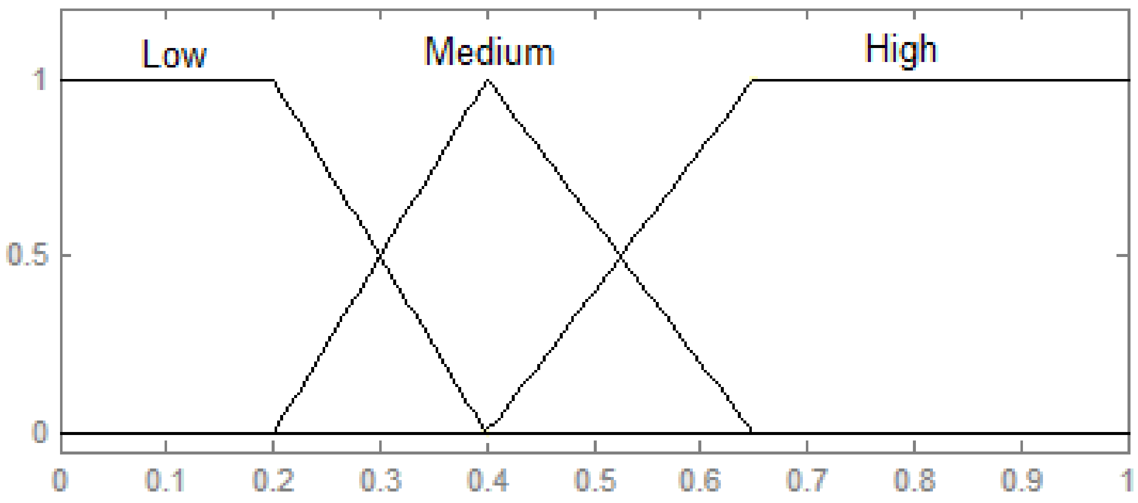

Figure 16: Membership functions of the fuzzy sets related to ρ (for dataset A).

The inference system relies 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 the dataset A are correctly segmented in the 81.1% of the cases. Since in this dataset we have stored the textual descriptor indicating the index of the proper cut column, we consider a segmentation as correct only when the routine locates exactly such a column. The CCC dataset provides a challenging set of characters for segmentation purposes. The only reference where segmentation of touching characters obtained from CCC dataset is performed similarly to this section is in (Kurniawan et al., 2011), where the 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 the dataset used in (Kurniawan et al., 2011) are different, even if they both derive from the same CCC dataset.

Then, we compare more in-depth our work with (Kurniawan et al., 2011) replicating a second experiment reporter in their paper and taking into account also inaccurate segmentation. The authors in (Kurniawan et al., 2011) consider as correct the segmentation when a cutting column is found in the surroundings the correct position (without giving further details); they report a success percentage of 91.9%. We replicated such an experiment by considering proper, even if inaccurate, segmentation when the cut position identified is in a range of five columns from the exact position. In this case, our success percentage is 88.9%. As stated in (Lee and Verma, 2002), although research in handwriting recognition has been an active research area for more than an half-century, the maturity of the segmentation techniques is still very low. Our proposed approach focused on improving the segmentation accuracy, achieving comparable results to other works.

Other results that show the behavior of our method are reported below. These results refer to the segmentation performed on touching characters "eh", "'rt", "xm", and "ao" depicted in Figure 2(a), 17(a), 17(b), and 17(c), 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.

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

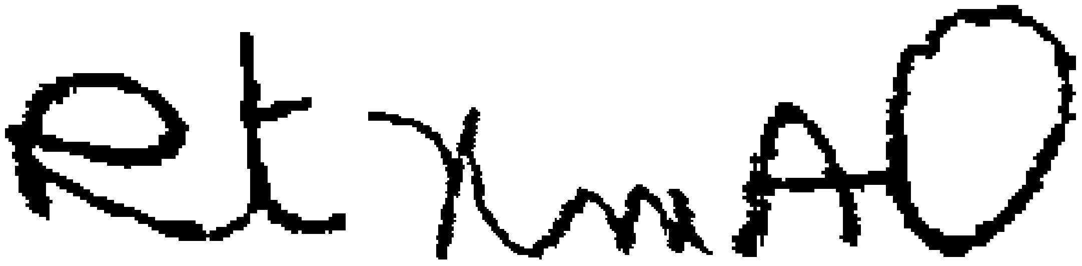
(a) (b) (c)

Figure 17: Touching characters “rt”, “xm”, and “ao” extracted from dataset A.

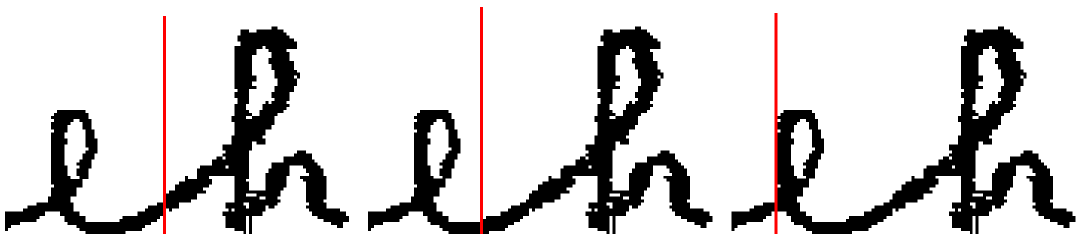
(a) (b) (c)

Figure 18: Cutting positions located by ρ (a), $\bar{g}$ (b), and $\bar{h}$ (c), better seen in colors.

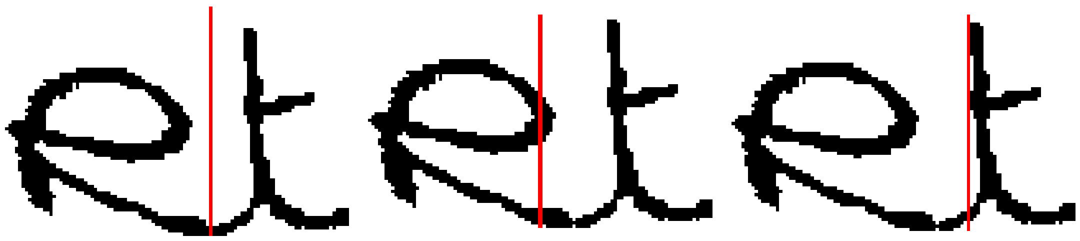
(a) (b) (c)

Figure 19: Cutting positions located by ρ (a), $\bar{f}$ (b), and $\bar{h}$ (c), better seen in colors.

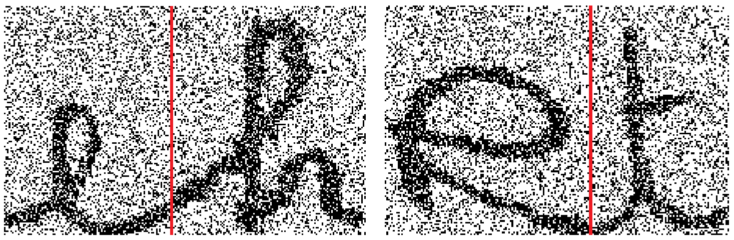
(a) (b)

Figure 20: Cutting positions located by ρ for images shown in Figure 18(a) and in Figure 19(a) after injection of salt and pepper noise with a density of 0.5 (i.e., 50% ca. of the pixels are affected), better seen in colors.

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 Figure 18 and in Figure 19, 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 the patterns have usually more than 100 columns.

Table 2: Overall recognition accuracy performances measured on a noisy version of the dataset B.

Noise	Overall recognition accuracy
No noise	96.1%
Salt and pepper, $d = 0.01$	86.1%
Salt and pepper, $d = 0.05$	72.0%
Gaussian, $M = 0, V = 0.01$	96.1%
Gaussian, $M = 0, V = 0.05$	79.4%

4.4 Tests on noisy data

To test the performance of our fuzzy routine against more realistic data, we have conducted experiments considering images affected by noise. Relying on the `imnoise` function of MATLAB, we were able to emulate errors and artificial patterns of low-quality copy processes. In particular, we conducted the same experiments presented in Sections 4.2 and 4.3, leaving the fuzzy sets unchanged for all the functions, while we injected noise of both Gaussian and "salt and pepper" (switching selected pixels from black to white or viceversa) type in the two datasets. For what concerns Gaussian noise, we considered zero-mean white noise ($M = 0$) with a variance of 0.01 ($V = 0.01$) or 0.05 ($V = 0.05$); for what concerns salt and pepper noise we considered a density of 0.01 ($d = 0.01$) or 0.05 ($d = 0.05$).

The results of the noisy version of the dataset B are reported in Table 2. The hypothesis of noise that decreases performances would represent a reasonable guess; in fact experimental data seem to confirm this assumption (although we note that light Gaussian noise does not affect overall performances at all). However, it is interesting to compare these outcomes to the ones resulting from the experiments on a noisy version of the dataset A; these second outcomes are reported in Table 3. In this case, we find that noise does not affect the performances of our system at all. This demonstrates that our approach is very resistant to noise and image degradation that may affect real-world images, while the overall performance degradation shown in Table 2 depends on a decreased confidence on the pattern recognition system (and on its neural network) due to the noise. Noise resistance of our approach exploits the quality of the fuzzy logic implemented: its parameters are so adequate and precisely tuned to the task that they focus only on real features of interest for determining cutting positions, not being influenced in this from aleatory processes such as noise.

For a purely demonstrative purpose, we replicated the experiment on the dataset A considering a salt and pepper noise with a density of 0.5, meaning that roughly 50% of pixels in every image in the dataset is flipped (from 0 to 1 and vice-versa) as an effect of the noise. Although this level of noise is way over the one to be expected in a real scenario, still, the cutting performance remained unchanged. The reader is invited to compare Figure 18(a) and Figure 19(a) with respectively Figure 20(a) and Figure 20(b) to get an insight of the amount of noise that our strategy is capable of tolerate without incurring in performance degradation. Due to the particular combination we follow with the features of interest, our method is capable of highlighting the strength points of each feature especially when the others might be disguised and influenced by noise. As an example, in the specific case of Figure 20(a) and Figure 20(b), the function $\bar{f}(i)$ compensates the degradation effect that the noise has on the other features and functions (since the distance of the i^{th} column from the center of the image is robust to noise).

Table 3: Cutting performances measured on a noisy version of the dataset A (a cut is considered as correct when it is distant at most by five columns from the exact position).

Noise	Cutting accuracy
No noise	88.9%
Salt and pepper, $d = 0.01$	88.9%
Salt and pepper, $d = 0.05$	88.9%
Gaussian, $M = 0, V = 0.01$	88.9%
Gaussian, $M = 0, V = 0.05$	88.9%

5 Conclusion

This paper presents a novel fuzzy strategy to segment touching characters. The proposed method combines three state-of-the-art features of touching characters that have been usually exploited one at a time. Experiments have been conducted on two different datasets composed by Latin printed and handwritten characters, respectively, and considering also noise injection. Our fuzzy strategy has specific parameters meant to capture the differences between our two datasets. Fuzzy sets, membership functions, and fuzzy rules, which characterize the fuzzy inference scheme, have been properly constructed by means of expert–based choices and heuristic criteria, and then optimized for each dataset by means of the PSO algorithm. Numerical results are encouraging and show that the proposed method has an optimal capability of correctly separating touching characters also in a noise scenario; in addition, it can be adjusted for cover very different varieties of characters (not only for the classes considered in our experiments).

The three features considered are the distance from the center of the pattern (function f), the peak-to-valley function (function g) and the ratio of the second difference of the vertical projection (function h). We would like to point out that these features combined in our fuzzy routine seem to be sufficient for obtaining optimal results in the segmentation. In fact, adding further features appear to be useless. For instance, we have verified that the use of the crossing count as a fourth fuzzy input does not lead to improvements. This is surely a perspective that should be further investigated and motivated.

Dealing with perspective advances, future works will deal with segmentation of mathematical symbols and non–Latin characters, taking also into account the possibility of segmenting touching characters diagonally, and with the characterization of fuzzy models different form the Mamdani one (as, e.g., the Sugeno model).

Acknowledgment

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 the University of Turin.

This research has been partially supported by the "Smart Cities and Social Innovation Under 30" program of the Italian Ministry of Research and University through the PARLOMA Project (SIN_00132).

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

We would like to thank the anonymous referees whose suggestions helped to improve the paper.

References

Alhajj, R. and Elnagar, E. (2005). Multiagents to separating handwritten connected digits. *IEEE Transactions on Systems, Man, and Cybernetics*, 35(5):593–602.

Arora, T. and Gigras, Y. (2013). A survey of comparison between various meta–heuristic techniques for path planning problem. *International Journal of Computer Engineering and Science*, 3(2):62–66.

Bansal, V. and Sinha, R. M. K. (2002). Segmentation of touching and fused devanagari characters. *Pattern Recognition*, 35:875–893.

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

Caballero, A. F., Lopez, M. T., and Castillo, J. C. (2012). Display text segmentation after learing best–fitted ocr binarization parameters. *Expert Systems with Applications*, 39(4):4032–4043.

Camastra, F., Spinetti, M., and Vinciarelli, A. (2006). Offline cursive character challange: a new benchmark for machine learning and pattern recognition algorithms. In *Proceedings of 18th International Conference on Pattern Recognition*, pages 913–916.

Casey, R. G. and Lecolinet, E. (1996). A survey of methods and strategies in character segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(7):690–706.

Chiaradonna, S., Di Giandomenico, F., and Murru, N. (2015). On enhancing efficiency and accuracy of particle swarm optimization algorithms. *International Journal of Innovative Computing, Information and Control*, 11(4):1165–1189.

Chowdhury, S., Weiyang, T., A., M., and Zhang, J. (2013). A mixed-discrete particle swarm optimization algorithm with explicit diversity–preservation. *Struct. Multidisc. Optim.*, 47:367–388.

Da Costa Lobato, T., Hauser-Davis, R., De Oliveira, T., Maciel, M., Tavares, M., and Da Silveira, A. (2015). Categorization of the trophic status of a hydroelectric power plant reservoir in the brazilian amazon by statistical analyses and fuzzy approaches. *Science of the Total Environment*, 506–507:613–620.

del Valle, Y., Venayagamoorthy, G., Mohagheghi, S., Hernandeza, J., and Harley, R. (2008). Particle swarm optimization: basic concepts, variants and applications in power systems. *IEEE Transactions on Evolutionary Computation*, 12(2):171–195.

D'Errico, G. and Murru, N. (2012). Fuzzy treatment of candidate outliers in measurements. *Advances in Fuzzy Systems*, 2012:Article ID 783843.

Fonseca, M. and Jorge, J. (2000). Using fuzzy logic to recognize geometric shapes interactively. In *The Ninth IEEE International Conference on Fuzzy Systems*, pages 291–296.

Frinken, V., Fischer, A., Mammatha, R., and Brunke, H. (2011). A novel word spotting method based on recurrent neural networks. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(2):211–224.

Garain, U. and Chaudhuri, B. B. (2002a). On ocr of degraded documents using fuzzy multifactorial analysis. In *AFSS International Conference on Fuzzy Systems*, pages 388–394.

Garain, U. and Chaudhuri, B. B. (2002b). Segmentation of touching characters in printed devnagari and bangla scripts using fuzzy multifactorial analysis. *IEEE Trans. on Systems, Man and Cybernetics*, 32(4):449–459.

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

Grafmuller, M. and Beyerer, J. (2013). Performance improvement of character segmentation in industrial applications using prior knowledge for more reliable segmentation. *Expert Systems with Applications*, 40(17):6955–6963.

He, S., Wiering, M., and Schomaker, L. (2015). Junction detection in handwritten documents and its application to writer identification. *Pattern Recognition*, 48:4036–4048.

Hu, X., Shi, Y., and Eberhart, R. (2004). Recent advances in particle swarm. In *IEEE Congr. Evol. Comput.*, pages 90–97.

Jung, M. C., Shin, Y. C., and Srihari, S. N. (1999). Machine printed character segmentation method using side profiles. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, pages 863–867.

Kahan, S. (1987). On the recognition of printed characters of any font and size. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(2):274–288.

Kennedy, J. and Eberhart, R. (2012). Particle swarm optimization. In *Proceedings of IEEE International IEEE Conference on Neural Networks*, pages 1942–1948.

Kim, K. K., Kim, J. H., and Suen, C. Y. (2000). Recognition of unconstrained handwritten numeral strings by composite segmentation method. In *Proceedings of IEEE 15th International Conference on Pattern Recognition*, pages 594–597.

Kumar, M., Jindal, M. K., and Sharma, R. K. (2014). Segmentation of isolated touching characters in offline handwritten grumukhi script recognition. *International Journal of Information Technology and Computer Science*, 2:58–63.

Kurniawan, F., Rahim, M. S. M., Daman, D., Rehman, A., Mohamad, D., and Shamsuddin, S. M. (2011). Region–based touched character segmentation in handwritten words. *International Journal of Innovative Computing, Information and Control*, 7(6):3107–3120.

Lacerda, E. B. and Mello, C. A. B. (2013). Segmentation of connected handwriting digits using self–organizing maps. *Expert Systems with Applications*, 40(15):5867–5877.

Lee, H. and Verma, B. (2002). Binary segmentation algorithm for english cursive handwriting recognition. *Pattern Recognition*, 45(4):1306–1317.

Liang, J., Phillips, I. T., and Haralick, R. M. (2002). An optimization methodology for document structure extraction on latin character documents. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(7):719–734.

Lin, C. and Huang, W. (2007). Locationg license plate based on edge features of intensity and saturation. In *IEEE International Conference on Innovative Computing, Information and Control*, pages 227–230.

Louloudis, G., Gatos, B., Pratikakis, I., and Halatsis, C. (2009). Text line and word segmentation of handwritten documents. *Pattern Recognition*, 42:3169–3183.

Lu, Y. (1993). On the segmentation of touching characters. In *Proceedings of IEEE 2nd International Conference on Document Analysis and Recognition (ICDAR)*, pages 440–443.

Lu, Y. (1995). Machine printed character segmentation – an overview. *Pattern Recognition*, 28(1):67–80.

Lu, Y., Chi, Z., Siu, W. C., and Shi, P. (1999). A background thinning based approach for separating and recognizing connected handwritten digit strings. *Pattern Recognition*, 32:921–933.

Lu, Y. and Shridhar, M. (1996). Character segmentation in handwritten words – an overview. *Pattern Recognition*, 29(1):77–96.

Mamdani, E. H. and Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man–Machine Studies*, 7(1):1–13.

Manmatha, R. and Rothfeder, J. L. (2005). A scale space approach for automatically segmenting words from historical handwritten documents. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(8):1212–1225.

Nachar, R., Inaty, E., Bonnin, P. J., and Alayli, Y. (2015). Breaking down captcha using edge corners and fuzzy logic segmentation/recognition technique. *Security and Communication Networks*, 8:3995–4012.

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

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

Noronha, T., Oliveira, T., Silveira, A., da Silva, R., and Saraiva, A. (2013). Knowledge acquisition of vibrations in high-power transformers using statistical analyses and fuzzy approaches a case study. *Electric Power Systems Research*, 104:110–115.

Olszewska, J. (2012). Multi–target parametric active contours to support ontological domain representation. In *RFIA Conference*, pages 779–784.

Olszewska, J. (2015). Active contour based optical character recognition for automated scene understanding. *Neurocomputing*, 161:65–71.

Olszewska, J. and McCulskey, T. (2011). Ontology–coupled active contours for dynamic video scene understanding. In *IEEE International Conference on Intelligent Engineering Systems*, pages 369–374.

Otsu, N. (1979). A treshold selection method from gray–level histograms. *IEEE Trans. Sys., Man., Cyber.*, 9(1):62–66.

Pal, U., Belaid, A., and Choisy, C. (2003). Touching numeral segmentation using water reservoir concept. *Pattern Recognition Letters*, 24:261–272.

Qu, P. and Zheng, W. (2011). Sport video intelligence analysis using mid–level and low–level vision information. In *International Conference on Future Computer Science and Education*, pages 165–168.

Roy, P. P., Pal, U., and Llados, J. (2008). Recognition of multi–oriented touching characters in graphical documents. In *Proceedings of IEEE 6th Indian Conference on Computer Vision, Graphics and Image Processing*, pages 297–304.

Roy, P. P., Pal, U., Llados, J., and Delalandre, M. (2012). Touching numeral segmentation using water reservoir concept. *Pattern Recognition*, 45(5):1972–1983.

Saba, T., Sulong, G., and Rehman, A. (2010). Non–linear segmentation of touched roman characters based on genetic algorithm. *International Journal on Computer Science and Engineering*, 2(6):2167–2172.

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

Sedighi, A. and Vafadust, M. (2011). A new and robust method for character segmentation and recognition in license plate images. *Expert Systems with Applications*, 38(11):13497–13504.

Song, Y. and Wang, W. (2009). Text localization and detection for news video. In *IEEE International Conference on Information and Computing Science*, pages 98–101.

Stamatopoulos, N., Gatos, B., and Perantonis, S. J. (2009). A method for combining complementary techniques for document image segmentation. *Pattern Recognition*, 42:3158–3168.

Tobias, O. J. and Seara, R. (2002). Image segmentation by histogram thresholding using fuzzy sets. *Pattern Recognition*, 11(12).

Wei, X., Ma, S., and Jin, Y. (2005). Segmentation of connected chinese characters based on genetic algorithm. In *Proceedings of IEEE 8th International Conference on Document Analysis and Recognition (ICDAR)*, pages 645–649.

Wu, D. (2012). Twelve considerations in choosing between gaussian and trapezoidal membership functions in interval type-2 fuzzy logic controllers. In *IEEE International Conference on Fuzzy Systems.*

Yasojima, C., Furmigare, M., De Souza Brazil, F., De Oliveira, T., and Da Silveira, A. (2013). Partial discharge analysis and inspection alert generation in high power transformers: A case study of an autotransformer bank at eletrobrs-eletronorte vila do conde station. *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pages 367–378.

Zadeh, L. (1965). Fuzzy sets. *Information and Control*, 8(3):338–353.

Zhao, S., Chi, Z., Shi, P., and Yan, H. (2003). Binary segmentation algorithm for english cursive handwriting recognition. *Pattern Recognition*, 36(1):145–156.

Zhou, J., Krzyzak, A., and Suen, C. Y. (2002). Verification–a method of enhancing the recognizers of isolated and touching handwritten numerals. *Pattern Recognition*, 35(5):1179–1189.

Zimmermann, H. (1996). *Fuzzy set theory and its applications.* Publishers Ltd and Kluwer Academic Publishers, 2 edition.