Pattern Recognition 45 (2012) 4237–4249

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Removal of noise patterns in handwritten images using expectation maximization and fuzzy inference systems

Mehdi Haji*, Tien D. Bui, Ching Y. Suen

Department of Computer Science and Software Engineering, Concordia University, Montreal, Canada

ARTICLE INFO

Article history:
Received 23 November 2011
Received in revised form
23 April 2012
Accepted 16 May 2012
Available online 31 May 2012

Keywords:
Denoising
Handwritten images
Recognition
Fuzzy inference systems
Expectation maximization
Optimization

ABSTRACT

The removal of noise patterns in handwritten images requires careful processing. A noise pattern belongs to a class that we have either seen or not seen before. In the former case, the difficulty lies in the fact that some types of noise patterns look similar to certain characters or parts of characters. In the latter case, we do not know the class of noise in advance which excludes the possibility of using parametric learning methods. In order to address these difficulties, we formulate the noise removal and recognition as a single optimization problem, which can be solved by expectation maximization given that we have a recognition engine that is trained for clean images. We show that the processing time for a noisy input is higher than that of a clean input by a factor of two times the number of connected components of the input image in each iteration of the optimization process. Therefore, in order to speed up the convergence, we propose to use fuzzy inference systems in the initialization step of the optimization process. Fuzzy inference systems are based on linguistic rules that facilitate the definition of some common classes of noise patterns in handwritten images such as impulsive noise and background lines. We analyze the performance of our approach both in terms of recognition rate and speed. Our experimental results on a database of real-world handwritten images corroborate the effectiveness and feasibility of our approach in removing noise patterns and thus improving the recognition performance for noisy images.

1. Introduction

The ability to handle noise is an indispensable part of any real-world image understanding system. The input data that a system is supposed to process are usually mixed with some unwanted data that deteriorate the performance of the system. The extent to which the performance of a system is affected by noise depends on the underlying models and the type of noise. For example consider an Optical Character Recognition (OCR) application where each line of text is segmented into its constituent characters and then the characters are sent to a character recognition engine. If the character recognition engine is only trained for isolated characters and we send a special symbol or a character from another script that may appear in the document, then the output of the engine could be unpredictable. In the OCR application, we consider as noise any pattern that the recognition engine is not supposed to process. Of course, not every type of input noise will result in unpredictable output behavior. For example, if a character '*l*' is broken into two parts due to noise, then the character recognition engine may recognize the image as '*i*' as its first hypothesis, but '*l*' as its second hypothesis.

In order to reduce the chance of unexpected or degraded behavior, it is desirable to remove or reduce the noise as much as possible. The goal of this research is to improve the performance of the IMDS[1] word spotting system for automatic processing of handwritten mails. Therefore, we propose our methodology for the denoising of handwritten images; however, the underlying idea is general and can be applied to similar types of denoising problems.

There are two types of noise that we have to handle when working with handwritten images: low-level and high-level. Low-level noise is the random variation of intensity in document images that is produced by the hardware equipment during the scanning process. High-level noise refers to parts of the image data that are undesirable for the intended application, and as such they can be inherent parts of the input data or artefacts that are produced by the involved hardware equipment or the processing system. Fig. 1 shows samples of handwritten text with high-level noise. Simply, anything other than text is considered as high-level noise. Besides the dot-shaped (impulsive) and line patterns that contaminate the image data in all of these samples, the interfering character strokes from the upper text lines in Fig. 1(g) and (h) are also undesirable for a recognition application. These unwanted

* Corresponding author.
E-mail address: m_haji@encs.concordia.ca (M. Haji).

[1] http://www.imds-world.com.

0031-3203/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2012.05.013

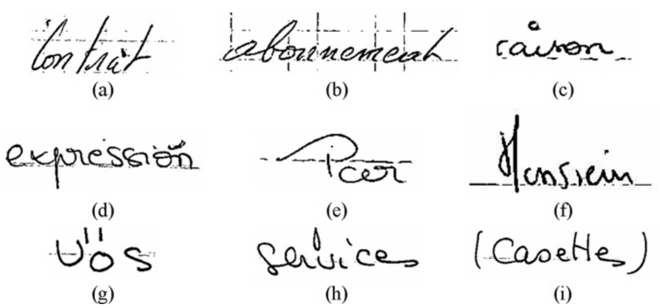

Fig. 1. Samples of handwritten text with high-level noise.

strokes are samples of high-level noise that is introduced to the image data as the result of imperfection in a previous processing step (line/word segmentation). Furthermore, depending on the application, punctuation marks and symbols (Fig. 1(i)) could be considered as high-level noise. They are undesirable parts of the image data in a word spotting application; however, they probably contain useful information in a text-to-speech application.

Low-level noise removal is a well-studied problem in the image processing literature. Recent approaches to low-level noise removal have utilized the state-of-the-art tools in statistics and signal processing [1–4]. These approaches normally address the problem of additive Gaussian noise or impulse noise removal for a general setting where it is often assumed that the image pixels are contaminated by a random process that is independent of the pixel values. However, high-level noise removal depends on the specific application, and obviously the inherent constraints and settings of each problem may call for different treatments. Not surprisingly, the removal of high-level noise in handwritten images has been less studied due to its application dependent nature. For page segmentation applications, a particular type of noise that must be handled is the marginal noise. The marginal noise refers to large black areas around a document image that are normally artefacts produced during the scanning or photocopying process. There are several studies concerning the marginal noise problem [5–7]. However, there has been comparatively less research concerning the detection and removal of other types of noise that appear in document images. In [8], a novel method based on distance transform has been proposed for the detection of removal of clutter in document images, where clutter is defined as unwanted foreground content which is typically *larger* than text. Some common forms of clutter noise in document images are punched holes, ink seeps and ink blots. Another type of noise that especially appears in handwritten images is stroke-like pattern noise, which refers to the background connected components that are similar to character strokes or diacritics. In [9], a classification-based method has been proposed for the detection and removal of stroke-like patterns. The detection of noise patterns is carried out in two phases where the first phase is based on a supervised classification, and the second phase is based on an unsupervised classification technique. The method that we propose in this paper can be considered as an extension of [9] in the sense that our method does not rely on the noise patterns belonging to any particular distribution. Therefore, we formulate the noise removal problem as an unsupervised learning where the optimization criterion is the recognition score for the input image after noise removal. To the best of our knowledge, this work is the first to address the problem of arbitrary noise patterns in handwritten images for recognition applications. We will present an algorithm based on expectation maximization for the unified denoising/recognition optimization problem, and given that prior knowledge about noise is available, we will present a systematic way based on fuzzy logic in order to incorporate that knowledge into the optimization process.

Fuzzy logic is a form of logic derived from fuzzy set theory to deal with variables and reasoning that are approximate. Fuzzy inference systems (FISs) which are rule-based systems based on fuzzy variables have been successfully applied to many fields such as expert systems, data classification, decision making, computer vision and automatic control [10,11]. One main advantage of fuzzy variables and fuzzy rules is that they facilitate the expression of rules and facts that are easily understandable by humans. Furthermore, it is easy to modify a FIS by inserting and deleting rules, meaning that there is no need to create a new system from scratch. In order to train a FIS, it is possible to start with a few rules that are designed by human expert and then fine-tune the parameters of the FIS over a set of training (validation) data.

Recently there has been a great interest in using fuzzy logic for the detection and removal of low-level noise in images [12–16]. In document image processing, fuzzy logic has been applied for the enhancement of low-quality images [17], feature extraction, recognition, etc. [18]. In this paper, we utilize fuzzy logic to incorporate our prior knowledge about some common types of noise patterns into our proposed noise removal algorithm.

2. Problem definition

Let $C_i=\{c^i_1, c^i_2, \ldots, c^i_{ni}\}$ be the set of connected components of a word image W_i. The set of connected components is composed of two disjoint subsets T_i and N_i, where $T_i=\{c^i_j \in Text\colon\ 1\leq j\leq n_i\}$ denotes the subset of connected components that belong to the text, and $N_i=\{c^i_j \notin Text\colon\ 1\leq j\leq n_i\}$ denotes the subset of connected components that do not belong to the text. The text itself is a natural language which is defined over a finite alphabet Σ which is the set of letters, and depending on the application, digits and punctuation marks. A word over the alphabet Σ is defined as a finite sequence of letters. In a natural language not all possible sequences of letters form valid words. Let $V\subset\Sigma^*$ denote the set of valid words, i.e., the vocabulary of the language. The goal is to find the two subsets of text T_i and noise N_i for a word image W_i given the vocabulary V. It should be noted that the vocabulary is application-dependent, and it may be as small as a few 10 of words or it can be as large as tens of thousands of words or even unlimited in which case it must be represented by a set of formation rules or statistical models.

There are two general approaches to find the subsets of text and noise from the image of a word: latent and direct. In the former, we treat the indicator functions associated with the subsets of text and noise as latent variables that have to be inferred from observable variables of the recognition system. In the latter, we either implicitly or explicitly model the likelihood functions of the text and noise based on a priori knowledge. Examples of direct noise removal approaches for handwritten document images are given in the seminal works of Agrawal and Doermann [8,9].

A direct noise removal approach can be formulated as a binary classification problem with the two classes of noise and text. Consequently, we have to make some assumptions about the nature of the patterns belonging to one class in order to be able to distinguish them from the patterns belonging to the other class. The main difficulty here lies in the fact that there could be significant overlaps between certain classes of noise patterns and characters or parts of characters. In such cases, we have to use the context knowledge (i.e., transcription) in order to resolve the ambiguity. Therefore, in this paper, we formulate the noise removal and recognition as a single optimization problem involving latent variables. This makes our approach non-parametric in the sense that it does not make any specific assumption about the nature of noise. However, the non-parametric assumption comes at a price. As we will show in the following section, in general, the processing time required by the latent approach is higher than the direct approach by a factor that is linear with the size of the input (in terms of the number of the connected components). Therefore, in order to expedite the processing time we will show that we can efficiently incorporate the a priori knowledge into the latent approach using fuzzy rules.

3. Removal of noise patterns using latent variable approach

We model the recognition system using a latent variable approach where we assume the input data is composed of a set of observable variables and a set of latent variables. The observable variables for a word W_i are the set of connected components $C_i=\{c_1^i, c_2^i, \ldots, c_{ni}^i\}$. Associated with each connected component c_j^i we can define two latent variables z_{Nj}^i and z_{Tj}^i that specify whether the connected component is noise (i.e., $c_j^i \in N_i$) or belongs to the text (i.e., $c_j^i \in T_i$), respectively, where z_{Nj}^i, $z_{Tj}^i \in \zeta=\{0, 1\}$ and $z_{Nj}^i+z_{Tj}^i=1$. Let $Z_{N,T}^i=\{(z_{Nj}^i, z_{Tj}^i)\}$ denote the set of latent variables corresponding to C_i that completely specify the two subsets of T_i and N_i.

Let $S_{Wi} \in V$ denote the unknown transcription of the word image W_i. Depending on the type of the underlying recognition system, beside $Z_{N,T}^i$'s, we have other sets of latent variables in this problem. For example, if we use an analytical method that is based on character recognition, then there are other sets of latent variables that specify whether and where a connected component must be segmented in order to form the constituent characters, and whether a set of neighboring connected components must be merged in order to form a single character. Let's denote the set of all latent variables corresponding to a word image W_i by $Z_i=(Z_{N,T}^i, Z_H^i)$. We define the recognition engine as a function F: $\{(C_i, Z_i)\} \rightarrow \Sigma^*$ that maps the domain of observable and latent variables to the set of strings that belong to the language.

Given that the latent variables are known, we can find the transcription of the image. And given that the transcription is known, we can find the latent variables. One classical way of approaching a problem involving unknown parameters and latent variables is the Expectation Maximization (EM) algorithm. The EM algorithm is an iterative method for finding the maximum likelihood estimates of parameters in a statistical model. There are many instances and variants of the EM algorithm that have been applied to well-known problems such as unsupervised data clustering, learning, data reconstruction etc. We outline our EM-based noise removal algorithm for word images as follows:

Step 1: Initialize $Z_i=(Z_{N,T}^i, Z_H^i)$ to some random values, and obtain an initial estimate for the denoised image using $Z_{N,T}^i$.
Step 2: Calculate the transcription $\theta=S_{Wi}$ for the just-denoised image using the recognition function F and the current estimate for Z_H^i.
Step 3 (expectation): Calculate the expected value of the log-likelihood function $L(\theta;\, C_i, Z_i)=Pr(C_i, Z_i|\theta)$ with respect to the conditional distribution of Z_i given C_i, and then update the value of each latent variable by its expected value, i.e.,

$$Z_i \leftarrow E_{Zi|Ci,\theta}[\log L(\theta; C_i, Z_i)].$$

Step 4 (maximization): Using the new values of $Z_{N,T}^i$ denoise the image again. Then, using the new values of $Z_{N,T}^i$ and the recognition function F, calculate a new estimate for the transcription θ.
Step 5: Iterate between Step 3 and Step 4 until the stopping criterion is met.

Note that this formulation allows us to use any recognition function as long as we can compute $L(\theta;\, C_i,\, Z_i)$ efficiently. A common way of modeling the recognition function is based on Path-Discriminant Hidden Markov Models (PD-HMMs), where we model each input symbol by a meta-state. One advantage of using HMMs is the existence of efficient algorithms for the underlying inference problems. Given that we have a sequence of symbols as the input we can use the Viterbi algorithm [19] in order to find the most likely sequence of hidden states that generate the input. And given the parameters of the model, we can use the so-called forward algorithm in order to compute the probability of an input sequence of symbols. Both algorithms make use of the principle of dynamic programming to efficiently solve the inherent optimization problems.

In the PD-HMM recognition model, the most likely sequence of states corresponds to the most likely transcription for the input sequence. Therefore, given the input sequence is noise free, using the Viterbi algorithm we can actually find the segmentation paths between characters (i.e., all of the latent variables denoted by Z_H^i) and the corresponding transcription θ at the same time. Thus, without violating the non-parametric assumption about noise patterns, we can re-write our denoising algorithm based on the PD-HMM recognition model as follows:

Step 1: Initialize z_{Nj}^i's to some random values in {0, 1}.
Step 2: Calculate the expected value for each z_{Nj}^i as follows:

$$E\left[z_{Nj}^i\right] = \frac{Pr(S_i^1)}{Pr(S_i^0)+Pr(S_i^1)}$$

where: $T_i^0=\{c_k^i \in C_i | z_{Nk}^i < 0.5\} \cup \{c_j^i\}$; $T_i^1=\{c_k^i \in C_i | z_{Nk}^i < 0.5\} \backslash \{c_j^i\}$;
and using the Viterbi algorithm:

- $S_i^0=\arg\max_{\theta \in \Sigma*}(Pr(T_i^0|\theta))$
- $S_i^1=\arg\max_{\theta \in \Sigma*}(Pr(T_i^1|\theta))$

Step 3: Update the value of each z_{Nj}^i by its expected value, i.e.,

$$z_{Nj}^i \leftarrow E[z_{Nj}^i]$$

Step 4: Iterate between Step 2 and Step 3 until the stopping criterion is met.

In Step 2 of the algorithm we invoke the Viterbi algorithm one time for each value of each latent variable. The complexity of the Viterbi algorithm is $O(N_1 \times N_2^2)$ where N_1 is the size of the input sequence and N_2 is the number of states in the HMM. Therefore, the complexity of Step 2 of the algorithm becomes $O(|\zeta| \times |C_i|^2 \times |\Sigma|^2)$, where in our case $|\zeta|=2$. Note that the proposed noise removal algorithm performs the recognition as a by product of Step 2. Given that the input image was noise free, the recognition would be done in $O(|C_i| \times |\Sigma|^2)$ by one application of the Viterbi algorithm.

Therefore under the assumption that the distribution of noise is not known a priori, the recognition time for a noisy input sequence C_i is increased by a factor of $2 \times |C_i|$ in each iteration of the EM algorithm. The EM algorithm is a local search approach, and as such its convergence rate much depends on the initial guess. If the algorithm starts with a good initial guess, it can normally find a good solution quickly. Otherwise, it can take a large number of iterations to converge to a solution. So it is desirable to start the search process with a set of initial guesses that are as good as possible. For this purpose, we propose to incorporate a priori knowledge using fuzzy logic to Step 1 of the proposed noise removal algorithm.

4. Brief review of fuzzy logic

For the sake of clarity of the forthcoming material, in this section we present a brief review of the four basic elements of fuzzy logic, namely, fuzzy sets, fuzzy operators, fuzzy rules and fuzzy inference systems. The interested reader is referred to [10] for more in-depth information about these concepts.

4.1. Fuzzy sets

A fuzzy set is a set whose elements have degrees of membership in the real interval [0,1]. In classical set theory, an element either belongs to a set or not. The membership of an element x in a set A, in classical logic, is defined by an indicator function (a.k.a. characteristic function). The value of the indicator function is 1 when $x \in A$, and 0 when $x \notin A$. In fuzzy logic, the degree of membership of an element in a set is indicated by a value in the real interval [0,1]. In this sense, fuzzy logic is an extension of classical (binary) logic that uses a continuous range of truth degrees in the real interval [0,1], rather than the strict values of 0 and 1. This extension allows the gradual assessment of the membership of elements in a set.

An example is shown in Fig. 2 where we define two fuzzy sets HORIZONTAL and VERTICAL on the orientation (in degrees) of a 2D shape. We use triangular/trapezoidal membership functions which are the most commonly used types of membership functions due to their simplicity and ease of computation. According to these membership functions, when the orientation is 0° or 180°, it is fully included in the fuzzy set HORIZONTAL, and it is not included in the set VERTICAL. When the orientation is 90°, it is fully included in the set VERTICAL, and not included in the set HORIZONTAL. For these three values (0°, 90°, 180°), the memberships can be defined by the classical notion of set as well. However, when the orientation is 22.5° for example, then its degree of membership to the set HORIZONTAL is 0.5, which can be interpreted as *somewhat* horizontal in linguistic terms.

4.2. Fuzzy operators

The basic operations defined on crisp sets, namely intersection (AND), union (OR) and complement (NOT), can be generalized to fuzzy sets. The generalization to fuzzy sets can be achieved in more than one possible way. The most widely used fuzzy set operations that we will use in this work are called standard operations. The three standard fuzzy operations are standard fuzzy intersection (i.e., MIN), standard fuzzy union (i.e., MAX), and standard fuzzy complement.

4.3. Fuzzy rules

In fuzzy logic, we represent logic rules by a collection of IF-THEN statements. Each statement has the general form of IF P THEN Q, where the antecedent P and the consequent Q are fuzzy assignment statements.

4.4. Fuzzy inference system

Fuzzy inference is the process of the mapping from a given set of inputs to a set of outputs using fuzzy logic. A set of fuzzy rules combined with a method of fuzzy inference is called Fuzzy Inference System (FIS). In this work, we use the so-called MIN–MAX (a.k.a. Mamdani's) inference method that provides a simple and efficient way of computing the output based on standard (i.e., MIN and MAX) fuzzy operations.

4.4.1. Defuzzification

In some applications such as function approximation or decision problems, the output of the fuzzy system typically has to be expressed by a single value at the end. For example, in our noise removal algorithm, we want to determine whether or not a connected component should be initially considered as noise ($z_N = 1$) or not ($z_N = 0$). Defuzzification is the process of transforming a fuzzy set into a single crisp value. There are different methods to defuzzification [10]. In this work, we use the COG method because the choice of triangular/trapezoidal membership functions along with the MIN–MAX inference allows us to compute the center of gravity at a very low computational cost [20].

5. Incorporation of high-level knowledge into the algorithm using FIS

Let $Pr(Text|c_j^i \in W_i)$ denote the posterior probability of the connected component c_j^i in word W_i being part of the text, and $Pr(Noise|c_j^i \in W_i) = 1.0 - Pr(Text|c_j^i \in W_i)$ denote the posterior probability of the connected component being noise. According to Bayes' theorem, we can compute these posterior probabilities as follows:

$$Pr(Text|c_j^i \in W_i)\frac{Pr(c_j^i \in W_i|Text) \times Pr(Text)}{Pr(c_j^i \in W_i)} \tag{1}$$

$$Pr(Noise|c_j^i \in W_i) = \frac{Pr(c_j^i \in W_i|Noise) \times Pr(Noise)}{Pr(c_j^i \in W_i)} \tag{2}$$

where $Pr(Text)$ and $Pr(Noise)$ are the prior probabilities of the text and noise, respectively, and $Pr(c_j^i \in W_i)$ is the prior probability of the connected component $c_j^i \in W_i$ which acts as a normalizing constant for both equations.

In the absence of any further information, we assume $Pr(Text) = Pr(Noise) = 0.5$. Therefore, the likelihood of a connected component being text and noise is defined as follows:

$$L(Text|c_j^i \in W_i) = Pr(c_j^i \in W_i|Text) = Pr_{Text}(c_j^i \in W_i) \tag{3}$$

$$L(Noise|c_j^i \in W_i) = Pr(c_j^i \in W_i|Noise) = Pr_{Noise}(c_j^i \in W_i) \tag{4}$$

In the following, we will show how to use fuzzy inference systems in order to estimate the density functions in Eqs. (3) and (4) for two classes of noise patterns that frequently appear in handwritten images, namely, impulsive noise and background lines. The high-level knowledge about these types of noise can easily be incorporated into the noise removal algorithm using fuzzy logic as both classes can easily be described by linguistic rules.

We start the process of building the FISs by the extraction/normalization of features, then we will talk about the specification of the fuzzy sets and the definition of the rule bases for the

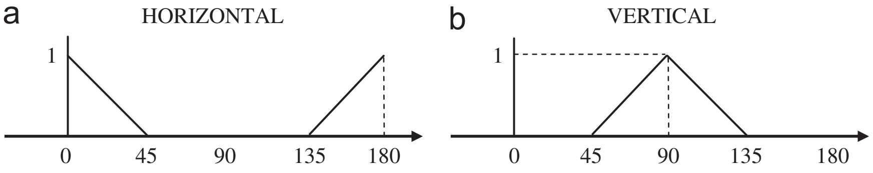

Fig. 2. Examples of membership functions defined on variable Orientation.

estimation of density functions, and finally, we will present the initialization of latent variables based on the estimation of density functions.

5.1. Feature extraction

In order to estimate whether a connected component is noise or belongs to the text, in absence of any further information, we rely on the geometrical properties (features) of the connected component.

The features that we extract from a connected component in order to estimate whether it is a dot or small noise could be as simple as: height, width, aspect ratio (defined as the ratio of height to width) and y-coordinate of the center of gravity (which can measure how close the connected component is to the upper baseline). However, for the detection of background lines from more complex character shapes, we add three more features: orientation, eccentricity, and compactness.

Eccentricity is an indication of how much a shape is extended in spatial length which is defined to be 0 for a circle and 1 for a line segment [21]. Compactness is an indication of solidness which is defined as follows.

5.1.1. Compactness

Let B be a binary shape, for an arbitrary axis L, the compactness of B is defined as the average of density of shape pixels over all lines along the axis. The density of a shape for a given line is defined as the number of shape pixels lying on the line over the distance between the two farthest boundary-points (i.e., intersections of the line and the shape). We define the compactness of a shape as the average of compactness for horizontal and vertical axes.

5.2. Feature normalization

In order to facilitate the definition of the fuzzy sets, we want the values of the features to be independent from the size and coordinate system of the image. Therefore, we normalize the height, width and y-coordinate of the center of gravity by the height of the image (i.e., number of rows when the image is represented by a raster data structure).

Table 1
Fuzzy sets defined on shape features.

Feature	Fuzzy sets
Normalized Y-coordinate of Center of gravity	TOP, BOTTOM
Aspect ratio	AROUND_1
Normalized height	SMALL_COMPARED_TO_NASW, EQUAL_TO_NASW, LARGE_COMPARED_TO_NASW, SMALL, MEDIUM, HIGH
Normalized width	SMALL_COMPARED_TO_NASW, EQUAL_TO_NASW, LARGE_COMPARED_TO_NASW, SMALL, MEDIUM, HIGH
Orientation	HORIZONTAL, VERTICAL, DIAGONAL_LEFT, DIAGONAL_RIGHT
Eccentricity	AROUND_0
Compactness	SMALL, MEDIUM, HIGH

5.3. Specification of fuzzy sets

The number of fuzzy sets that we define on an input variable depends on the level of the expert knowledge that is expressed by the corresponding linguistic rules. We typically use between 1 and 4 terms to quantify a variable in a linguistic system. For example, in order to determine whether a small dot belongs to a character, a human expert uses a linguistic rule such as: "if the dot is near the top of the image then it most likely belongs to a character". Therefore, in this case, only one or two fuzzy sets will be enough: TOP≡near the top of the image, and BOTTOM≡near the bottom of the image.

The complete list of fuzzy sets that we define on each shape feature is given in Table 1.

Fig. 3(a) shows the fuzzy sets TOP and BOTTOM that we define on the feature y-coordinate of the center of gravity (Y_{COG}). On the feature Aspect Ratio (AR), we only define one fuzzy set: ARONUD_1, which measures how close the aspect ratio is to unity. Let x denote the value of the input feature AR. The membership function of AROUND_1 is defined as a triangular with the value of 1 at $x=1$ which linearly goes to 0 at $x=0.5$ and $x=2$ as shown in Fig. 3(b), which means that the aspect ratio is not around 1 when the height is two or more times larger than the width, or the width is two or more times larger than the height. Similarly, on the input variable Eccentricity, we define only one fuzzy set: AROUND_0, which specifies how close the eccentricity is to that of a circle. Let x denote the value of the input feature Eccentricity. The membership function of AROUND_0 is defined as a triangular with the value of 1 at $x=0$ which linearly goes to 0 at $x=1$.

Fig. 4(a) shows the four fuzzy sets of HORIZONTAL, VERTICAL, DIAGONAL LEFT and DIAGONAL RIGHT that we define on the input variable Orientation. Fig. 4(b) shows the three fuzzy sets of SMALL (or LOW), MEDIM and LARGE that we define on the input variable Compactness. In most applications of fuzzy logic, these are the typical fuzzy sets that we define on a real variable in the interval [0,1]. We define the same fuzzy sets (SMALL, MEDIUM and LARGE) on the input variables Normalized Height and Normalized Width, as they quantify the size (width and height) of a shape in terms of the dimensions of the image. However, in our application it is also useful to quantify the size of a shape in terms of the Normalized Average Stroke Width (NASW).

The NASW is a useful property of the text that could help us keep the overlap between the distributions of noise and text small. Given that we have an estimate of the NASW, we know that a small dot-shaped connected component that is closer to the NASW (in height and width) is more likely to be a character dot and not an impulsive noise. Therefore, we define the three fuzzy sets of SMALL COMPARED TO NASW, EQUAL TO NASW and LARGE COMPARED TO NASW as shown in Fig. 5. These fuzzy sets

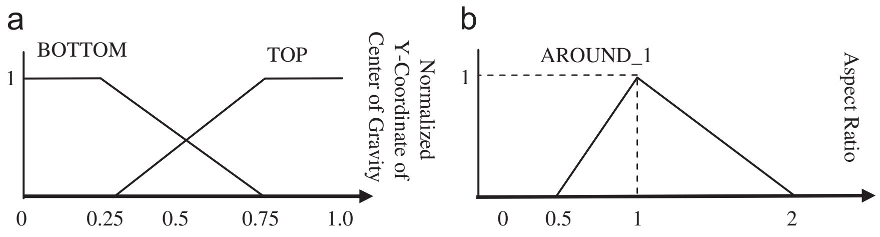

Fig. 3. Fuzzy sets defined on variables Normalized Y_{COG} and Aspect Ratio. (a) Fuzzy sets defined on Normalized Y_{COG}; (b) fuzzy set defined on Aspect Ratio.

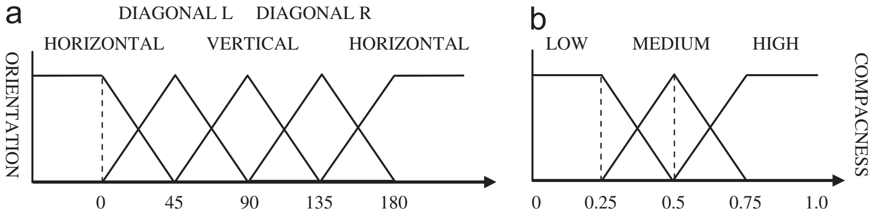

Fig. 4. Fuzzy sets defined on variables Orientation and Compactness. (a) Fuzzy sets defined on Orientation; (b) fuzzy sets defined on Compactness.

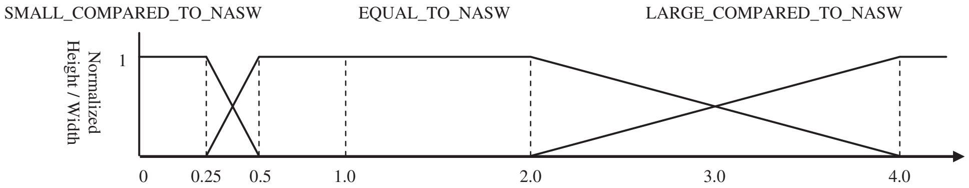

Fig. 5. Fuzzy sets defined on variables Normalized Height and Normalized Width.

quantify how small, equal or large the width and height of a shape are compared to the NASW.

5.3.1. Estimation of normalized average stroke width

Let B be a binary image where the foreground is represented by black pixels and the background is represented by while pixels. We estimate the Average Stroke Width (ASW) as the median of run-lengths of black pixels in all rows and all columns of the image:

$$\mathrm{ASW}_B = \mathrm{median}(\mathrm{length}(R_H) \cup \mathrm{length}(R_V)) \tag{5}$$

where R_H={black runs in all rows of B} and R_V={black runs in all columns of B}.

In order to obtain the NASW, we simply normalize the ASW by the height of the image:

$$\mathrm{NASW}_B = \mathrm{ASW}_\mathrm{B}/(\text{number of rows of } B) \tag{6}$$

5.4. Estimation of density function for impulsive noise in handwriting

Impulsive noise refers to small dot-shaped connected components that appear at random locations in document images. Samples of handwritten text with pronounced impulsive noise are given in Fig. 1(a) and (b).

5.4.1. Definition of rule base for impulsive noise

We define the rule base for the estimation of density function for impulsive noise to be composed of rules of the following form:

IF (Normalized Height is ...) AND (Normalized Width is ...) AND
(Normalized Y_{COG} is ...) AND (Aspect Ratio is ...) AND
(Eccentricity is ...) AND (Compactness is ...) AND
(Orientation is ...) THEN
(Dot is ...) AND (Impulsive Noise is ...);

Of course, the antecedent of a rule of this form does not need to contain all parts of the conjunction. Since the density functions for impulsive noise and character dots inevitably overlap, it is not always possible to distinguish a small character dot from an impulsive noise without the recognition information. The idea is to keep the overlap between the density functions small. Therefore, we define the rule base to cover the two basic cases where (1) impulsive noise is likely and character dots are unlikely; and (2) character dots are likely and impulsive noise is unlikely. The fuzzy rules corresponding to these two basic cases are as follows:

Rule 1:= IF (Normalized Height is SMALL_COMPARED_TO_NASW) AND (Normalized Width is SMALL_COMPARED_TO_NASW) THEN (Dot is LOW) AND (Impulsive Noise is HIGH);

Rule 2:= IF (Normalized Height is EQUAL_TO_NASW) AND (Normalized Width is EQUAL_COMPARED_TO_NASW) THEN (Dot is HIGH) AND (Impulsive Noise is LOW);

Now, we can refine these rules by adding more knowledge about the location of the connected component. We know that if a small connected component appears near the bottom of the image, it is less likely to be a character dot, compared to when it appears near the top of the image. Therefore, based on the location of the connected component, we can decompose Rule 1 into two rules and modify Rule 2 as follows:

Rule 1-1:= IF (Normalized Height is SMALL_COMPARED_TO_NASW) AND (Normalized Width is SMALL_COMPARED_TO_NASW) AND (Normalized Y_{COG} is BOTTOM) THEN (Dot is very LOW) AND (Impulsive Noise is very HIGH);

Rule 1-2:= IF (Normalized Height is SMALL_COMPARED_TO_NASW) AND (Normalized Width is SMALL_COMPARED_TO_NASW) AND (Normalized Y_{COG} is not BOTTOM) THEN (Dot is somewhat LOW) AND (Impulsive Noise is somewhat HIGH);

Rule 2:= IF (Normalized Height is EQUAL_TO_NASW) AND (Normalized Width is EQUAL_COMPARED_TO_NASW) AND (Normalized Y_{COG} is not BOTTOM) THEN (Dot is very HIGH) AND (Impulsive Noise is very LOW);

Where we have used the fuzzy hedges "very"/"somewhat" to increase/decrease the emphasis on their corresponding fuzzy sets [10]. We can further refine these rules using more features such as aspect ratio and compactness. However, in the current implementation, we only use the three rules listed above. As we will illustrate later in the experimental results, the addition of more rules does not necessarily improve the convergence speed of the algorithm.

5.5. *Estimation of density function for background line noise in handwriting*

Background lines are typically used as guidelines to help the user keep their writing consistent. Samples of handwritten text with pronounced background line noise are given in Fig. 1(b)–(f). The guidelines are usually printed in light colors, i.e., lighter than the ink that is used in pens. Therefore, in most cases we are able to remove the guidelines with proper binarization. However, in some situations the binarization algorithm may not be able to remove the guidelines, for example when we apply a global binarization operator to the whole document. Background lines are undesirable as they may adversely affect the processes of word segmentation and recognition.

Based on the features that we extract from a connected component, we can define a background line a as an elongated shape that is horizontal, whose height is small, whose width is medium or large, and appears near the bottom of the image. Depending on the application, we may want to distinguish dashes (or accents) from background line noises. Therefore, we also add the knowledge about separator dashes to the FIS. We define a dash as an elongated shape that is almost horizontal, whose height is small, whose width is medium (compared to average width of characters), and appears near the baseline of the text. The process of defining the rule base for background line noise is similar to that of impulsive noise. However, in order to accommodate the definition of linguistic rules for background line noise, we slightly modify some of the fuzzy sets as explained in the following.

5.5.1. *Modification of fuzzy sets*

Let Y_{baseline} be the normalized estimated baseline; that is the estimated row of the baseline divided by the number of rows of the image. We obtain Y_{baseline} using the robust projection profile-based technique described in [22]. Now, in order to measure whether or not a connected component is close to the baseline, we define a new fuzzy set on the feature Y_{COG}. We call this new fuzzy set CENTER which is a triangular function with a maximum value of 1.0 at $y_{\text{cog}} = Y_{\text{baseline}}$, that linearly goes to 0.0 at $y_{\text{cog}} = 0.0$ and $y_{\text{cog}} = 1.0$.

Furthermore, we need to change the unit to which we compare the widths of shapes. For impulsive noises, we compared the widths of shapes to the average stroke width. For background lines, we must compare the widths of shapes with the Average Character Width (ACW).

5.5.1.1. Estimation of average character width. Let B be a binary image corresponding to one or more text lines. Let $C = \{c_1, c_2, \ldots, c_N\}$ be the set of connected components of B. Let C_H be the subset of the connected components of C whose heights are not smaller than k_1 times the average stroke width:

$$C_H = c_i \in C \,\big|\, \text{height}(ci) \geq k_1 \times \text{ASW} \tag{7}$$

where in our experiments we set $k_1 = 2$.

We estimate the Average Character Height (ACH) as the average height of the connected components in C_H:

$$\text{ACH} = \text{sum}(\text{height}(c_i))/\left|C_H\right| : c_i \in C_H \tag{8}$$

In order to estimate the ACW we have to note that a connected component in a handwritten image may correspond to more than one character where the text is written cursively. Using the estimate of ACH, we exclude the connected components that may correspond to more than one character from the computation of ACW.

Let C_W be the subset of the connected components of C_H whose widths are not larger than k_2 times the ACH:

$$C_W = c_i \in C_H \,\big|\, \text{width}(c_i) \leq k_2 \times \text{ACH} \tag{9}$$

where in our experiments we set $k_2 = 1$.

We estimate the ACW as the average width of the connected components in C_w:

$$\text{ACW} = \text{sum}(\text{width}(c_i))/\left|C_W\right| : c_w \in C_W \tag{10}$$

Now, we add three fuzzy sets to specify whether the normalized width of a shape is small, equal or large compared to the Normalized ACW (NACW). These fuzzy sets are called SMALL COMPARED TO NACW, EQUAL TO NACW and LARGE COMPARED TO NACW, and they have the same definition as their corresponding fuzzy sets in Fig. 5.

5.5.2. *Rule base for background line noise*

The process of the definition of the rule base for background line noise is similar to that of the impulsive noise. We start with two basic rules that correspond to the two cases where the overlap between the density functions for noise and text is small:

Rule 1:= IF (Normalized Height is EQUAL_TO_NASW) AND (Normalized Width is EQUAL_TO_NACW) AND (Orientation is HORIZONTAL) THEN (Dash is HIGH) AND (Background Line Noise is LOW);

Rule 2:= IF (Normalized Height is not EQUAL_TO_NASW) AND (Normalized Width is LARGE_COMPARED_TO_NACW) AND (Orientation is HORIZONTAL) THEN (Dash is LOW) AND (Background Line Noise is HIGH);

Now, we can refine these rules by taking the location of the connected component into account:

Rule 1:= IF (Normalized Height is EQUAL_TO_NASW) AND (Normalized Width is EQUAL_TO_NACW) AND (Normalized YCOG is CENTER) AND (Orientation is HORIZONTAL) THEN (Dash is very HIGH) AND (Background Line Noise is LOW);

Rule 2-1:= IF (Normalized Height is not EQUAL_TO_NASW) AND (Normalized Width is LARGE_COMPARED_TO_NACW) AND (Normalized Y_{COG} is not CENTER) AND (Orientation is HORIZONTAL) THEN (Dash is very LOW) AND (Background Line Noise is very HIGH);

Rule 2-2:= IF (Normalized Height is not EQUAL_TO_NASW) AND (Normalized Width is LARGE_COMPARED_TO_NACW) AND (Normalized YCOG is CENTER) AND (Orientation is HORIZONTAL) THEN (Dash is somewhat LOW) AND (Background Line Noise is somewhat HIGH);

Similar to the discussion of the rule base for impulsive noise (Section 5.4.1), we can further refine these rules using more features such as eccentricity and compactness. However, in the current implementation, we only use the three rules listed above.

5.6. *Initialization of latent variables based on estimation of density functions*

Having obtained the estimation of density functions for common types of noise patterns, we need a decision rule in order to initialize the corresponding latent variables. A decision rule is a function that maps from an observation to an appropriate action.

Let:

$$q = (q_1 = L(\mathit{Text}\,|\,c_j^i \in W_i), \quad q_2 = L(\mathit{Noise}\,|\,c_j^i \in W_i)) \tag{11}$$

be the observable random vector associated with a connected component $c_j^i \in W_i$. We obtain q_1 and q_2 by aggregating the output of the fuzzy inference systems that we defined for the known classes of noise patterns. In general, let $F=\{F_1, F_2, \dots, F_n\}$ be the set of FISs that we have defined for n classes of noise patterns, where each F_k provides an estimate for a class of noise denoted by $Noise^k$ and the corresponding class of text denoted by $Text^k$. Then, we obtain q_1 and q_2 as follows:

$$q_1 = \max(L(Text^1 | c_j^i \in W_i), \quad L(Text^2 | c_j^i \in W_i) \ , \dots, \ L(Text^n | c_j^i \in W_i))$$

$$q_2 = \max(L(Noise^1 | c_j^i \in W_i), \quad L(Noise^2 | c_j^i \in W_i) \ , \dots, \ L(Noise^n | c_j^i \in W_i)) \tag{12}$$

Now we have to make a decision whether or not we want to initialize the latent variables based on the estimated likelihood values for those known classes of noise patterns. Therefore, we define the set of possible actions as follows:

$$A = a_1 = \text{'initialize based on estimated likelihood values'},$$

$$a_2 = \text{'initialize randomly'} \tag{13}$$

The reason we need a decision function is, first, we do not know the probability distributions of all possible classes of noise patterns, and second, some classes of noise patterns may overlap with some classes of text patterns. The idea is to start the optimization process with an initial solution that is as close to the optimal solution as possible. Therefore, we initialize the latent variable z_{Nj}^i to 1 (or 0) only when we are sure that the corresponding connected component is (or is not) noise. Otherwise, we randomly initialize z_{Nj}^i to 0 or 1.

Formally, we define the decision rule E_α: $Q \rightarrow A$ as follows:

$$E_\alpha = \begin{cases} z_{Nj}^i \leftarrow 1 & q_2 - q_1 > \alpha_{\text{diff\_min}} \text{ and } q_2 > \alpha_{\text{val\_min}} \\ z_{Nj}^i \leftarrow 0 & q_1 - q_2 > \alpha_{\text{diff\_min}} \text{ and } q_1 > \alpha_{\text{val\_min}} \\ z_{Nj}^i \leftarrow 0 \text{ or } 1 \text{ randomly} & \text{otherwise} \end{cases} \tag{14}$$

where $Q = \{q = (q_1, \ q_2)\}$ is the domain of observable random vectors, and $\alpha = (\alpha_{\text{val\_min}}, \ \alpha_{\text{diff\_min}}) > 0$ is the set of parameters of the decision rule. We will explain how the choice of the parameters would affect the convergence speed of the algorithm in the next section.

6. Experimental results

In the following, we present an experimental analysis of the proposed algorithm based on a database of real-world handwritten images. In order to show the effectiveness and feasibility of our approach, we evaluate the performance in terms of both the recognition rate and speed.

6.1. Analysis of recognition rate

We start by showing examples of density estimation using FISs. In Fig. 6, we have calculated the likelihood estimate of the impulsive noise versus character dots using the FIS-based method presented in Section 5.4. As can be seen, for the dot that belongs to the character '*i*', denoted by c_2, the estimated likelihood of being text is higher than noise, and for all impulsive noises the estimated likelihood of noise is higher than text in this image. However, the difference between the likelihood values of text and noise is different for each shape. In general, the difference is small if the shape can resemble text and noise (or neither one), and large otherwise. Using the decision rule defined in Eq. (14) with $\alpha_{\text{val\_min}} = 0.6$ and $\alpha_{\text{diff\_min}} = 0.3$, all latent variables z_{Nj}'s can be initialized to their correct (i.e., optimal) values for the character dot and all impulsive noises; except for the leftmost impulsive noise, denoted by c_1, for which the difference between the likelihood values of text and noise is not high ($0.67 - 0.48 = 0.19 < \alpha_{\text{diff\_min}}$). Therefore, for c_1 the corresponding latent variable z_{N1} is set randomly to 0 or 1. Let's say $z_{N1} \leftarrow 0$ which means that we start the optimization process assuming that c_1 is a part of the text.

Fig. 7 shows how the value of z_{N1} is updated in Step 2 of the proposed noise removal algorithm. We perform the recognition on the image two times, corresponding to the two hypotheses of "c_1 is noise" and "c_1 is not noise". As can be seen, the value of z_{N1} is updated to 0.67 after the first iteration, which means that the recognition engine favours the hypothesis of "c_1 is noise" given that all other latent variables are fixed. Fig. 8 shows an example of the likelihood estimation of the background line noise patterns versus separator dashes using the FIS-based method presented in Section 5.5. Again using the decision rule defined in Eq. (14) with the default values of parameters, most latent variables z_{Nj}'s can be initialized to their correct values.

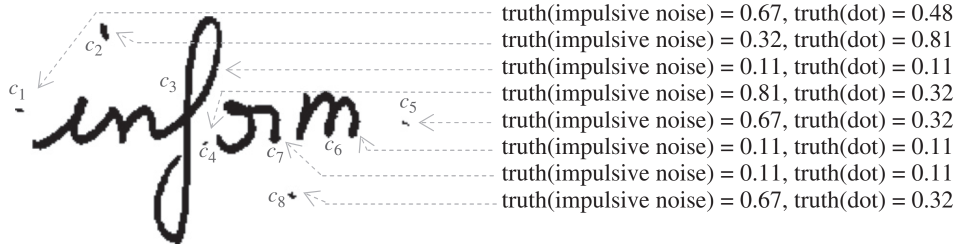
c_1
c_2
c_3
c_4
c_5
c_6
c_7
c_8
truth(impulsive noise) = 0.67, truth(dot) = 0.48
truth(impulsive noise) = 0.32, truth(dot) = 0.81
truth(impulsive noise) = 0.11, truth(dot) = 0.11
truth(impulsive noise) = 0.81, truth(dot) = 0.32
truth(impulsive noise) = 0.67, truth(dot) = 0.32
truth(impulsive noise) = 0.11, truth(dot) = 0.11
truth(impulsive noise) = 0.11, truth(dot) = 0.11
truth(impulsive noise) = 0.67, truth(dot) = 0.32

Fig. 6. Example of estimation of density function for impulsive noise patterns versus character dots using the corresponding FIS.

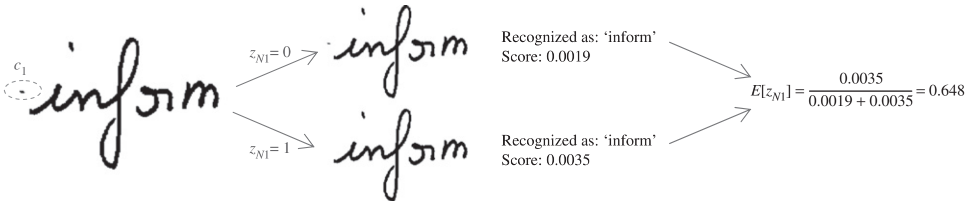

Fig. 7. Example of how a latent variable is updated using the recognition engine.

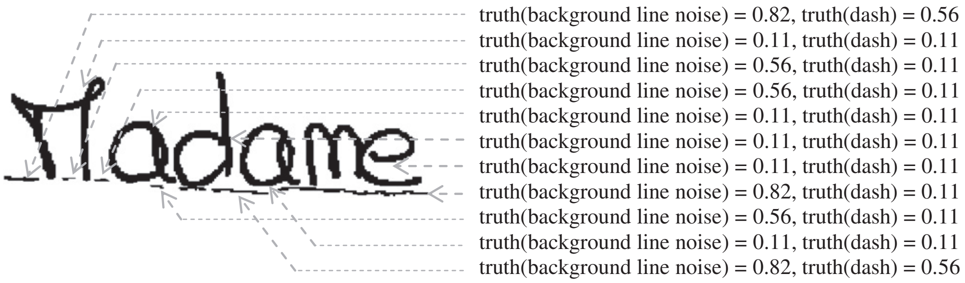
truth(background line noise) = 0.82, truth(dash) = 0.56
truth(background line noise) = 0.11, truth(dash) = 0.11
truth(background line noise) = 0.56, truth(dash) = 0.11
truth(background line noise) = 0.56, truth(dash) = 0.11
truth(background line noise) = 0.11, truth(dash) = 0.11
truth(background line noise) = 0.11, truth(dash) = 0.11
truth(background line noise) = 0.11, truth(dash) = 0.11
truth(background line noise) = 0.82, truth(dash) = 0.11
truth(background line noise) = 0.56, truth(dash) = 0.11
truth(background line noise) = 0.11, truth(dash) = 0.11
truth(background line noise) = 0.82, truth(dash) = 0.56

Fig. 8. Example of estimation of density function for background line noise patterns versus separator dashes using the corresponding FIS.

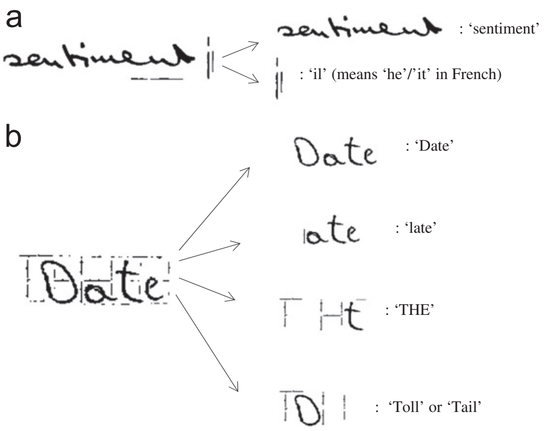

Fig. 9. Examples showing that noise removal problem may have more than one solution when no distribution is assumed for noise.

We have to note that in order for the algorithm to be able to separate the noise from the text, not both the distribution of noise and the lexicon of words can be unknown. Otherwise, the noise removal problem may not have a unique solution. Fig. 9 shows examples of handwritten words defiled by border/background line noise. As can be seen, it is possible that some noise components with or without some parts of data can form patterns that resemble valid entries in the lexicon. In such cases, the optimization process converges to one of the solutions, which is normally the one that is closer to the initial guess. These examples suggest that in order to increase the chance of finding the correct answer where the distribution of the text and the noise are close and the lexicon is large, we have to redo the optimization process several times with different initial guesses (for those z^i_{Nj}'s in Eq. (14) that are initialized randomly). Furthermore, we have to adjust the confidence scores that come from the recognition engine based on a measure of uniformity between the constituent parts of the input image. As can be seen in Fig. 9(b), all hypotheses are acceptable if we use a general recognition engine, however the correct answer is the most uniform in terms of the stroke width.

In order to adjust the recognition scores based on the uniformity of the input image, we compiled a database of training images from our collection of documents. The database contains two classes named 'word' and 'non-word', referring to whether a sample represents a clean, real word image or not. We extracted 500 samples for each class. The class 'word' contains images that represent both machine-printed and handwritten words with different writing styles. The class 'non-word' contains images that are either noise or mixture of noise and text. We already saw a few examples of non-words in Fig. 9. Fig. 10 shows some more examples from our database.

In order to discriminate the two classes of 'word' and 'non-word', we represented each image by a set of Gabor features [23].

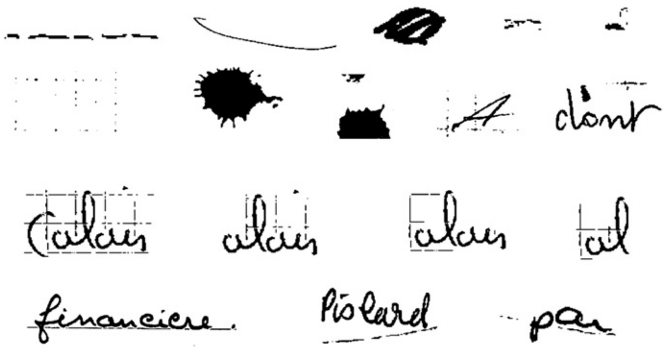

Fig. 10. Samples of non-words and poorly written words from our database for adjusting recognition scores.

We used eight Gabor filters corresponding to four orientations $\theta=0, \pi/4, \pi/2, 3\pi/4$ and two wavelengths $\lambda=0.05, 0.1$. We divided each image into 4×4 cells, and then we computed the percentage of pixels within each cell whose values are higher than the average value of the cell for each filtered image. Therefore, we extracted 128 features from each image. Then, we trained a binary Support Vector Machine (SVM) classifier with the Radial Basis Function (RBF) kernel in the feature space. We used a randomly selected 60% of the database for training and the remaining 40% for testing. The binary SVM achieved a performance of 94.1% on the database over a 10-fold cross validation. In order to enhance this performance, we probably need a larger training database, more elaborate features or better optimized classifiers. However, we should note that the performance of the denoising algorithm is not limited by the performance of the 'word'/'non-word' classification step which is only used to adjust the recognition scores for tricky images where the lexicon is large. As the score adjustment mechanism, we used a simple penalty function that decreases the normalized recognition score by a fixed amount of $p_{nw}=0.3$ for an input image that is classified as 'non-word'.

In order to assess the performance of the proposed denoising algorithm when used inside the recognition system, we compiled a test database of handwritten words from our collection of document images which are real-world scanned letters submitted to the customer service of a company by its clients. Fig. 11 shows sample documents from this collection that we used in our testbed. The lexicon of the test database contains 65 keywords that the company is interested to spot in these document images. We collected 10 samples per keywords. Then, we calculated the frequency distribution of noise patterns with respect to the database. Table 2 shows the percentage of the word images that are contaminated by different types of noise patterns. As we mentioned earlier, impulsive noise and background lines are predominant types of noise in these documents.

In order to estimate how the level of noise in an image affects the recognition performance, we defined a Signal-to-Noise Ratio

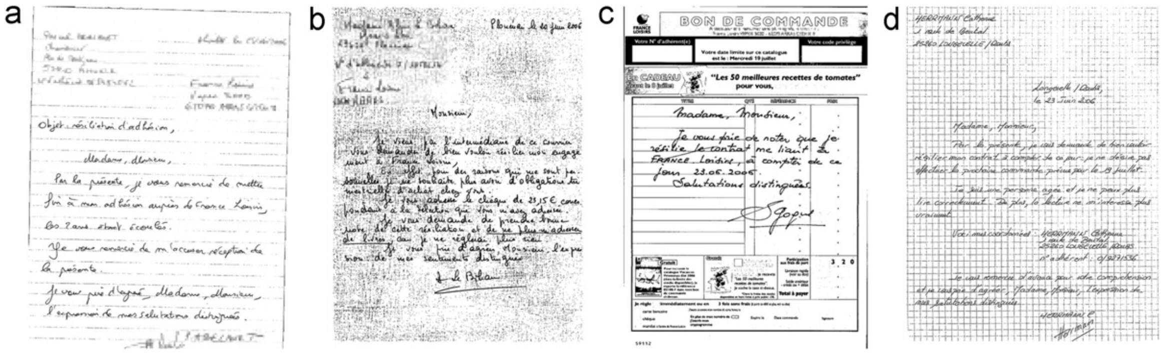

Fig. 11. Samples of real-world documents used in our testbed.

Table 2
Percentage of word images contaminated by different types of noise.

Impulsive noise	60.3%
Background line noise	44.7%
Interfering parts from upper/lower lines	5.4%
Other types of noise patterns and artefacts	18.3%

(SNR) measure for a word image W as follows:

$$\mathrm{SNR}(W) = \frac{\text{number of characters in } W}{\text{number of noise patterns in } W} \tag{15}$$

The reason we prefer the number of characters/patterns to the number of pixels is that the recognition may be affected by noise patterns that are smaller than the text (in terms of the number of constituent pixels). As an example of how we compute the SNR measure, the word image of Fig. 9(a) is composed of 9 letters and 5 connected components that are noise, and thereby it has a SNR of $9/5 = 1.8$. The word image of Fig. 9(b), wherein noise patterns are more abundant, has a SNR of $4/45 \approx 0.09$.

Fig. 12 shows how the top-1 and top-2 recognition rates are affected as the level of noise increases. As can be seen, when no denoising is performed, the recognition rate gradually decreases until the SNR ratio goes below 1 where a sharp decline in the recognition performance is observed. However, when the recognition is combined with denoising, the recognition performance remains almost unaffected. The difference between the highest recognition rate (at $\mathrm{SNR} \geq 12$) and the lowest recognition rate (at $\mathrm{SNR} \leq 1$) is 3% with denoising, versus 42–45% without denoising. Provided that we had an "ideal" denoising module, the recognition rate would have to be constant irrespective of the level of the noise. In order to find out how well our proposed denoising algorithm performs compared with an ideal denoising algorithm, we manually denoised our test database. The top-1 and top-2 recognition rates over the clean database were 91.2% and 94.5%, respectively. The top-1 and top-2 recognition rates over the original database were 74.6% and 81.3%, respectively, which were increased to 88.9% and 92.7% using our proposed recognition/denoising approach. The difference between the recognition performance obtained with the proposed denoising and the ideal denoising diminishes as we increase the number of top-n hypotheses. The difference became 0.3 at top-3, and 0.0 at top-4. These results corroborate that our proposed approach is effectual in removing noise patterns and thus improving the recognition rate for noisy images.

6.1.1. *Comparison with other denoising methods*

To put our results into perspective, we carried out some experiments with two remarkable denoising approaches: a high-level noise removal method for stroke-like patterns and a standard low-level denoising software for document images.

In [9], a recent denoising method for the detection and removal of stroke-like patterns in document images is proposed. This method is composed of two processing phases: first, prominent text components are detected using a supervised classification, and second, noise patterns are separated using k-means clustering. We used a subset of 250 word images containing over 1700 connected components for the training of the supervised classifier. The top-1 and top-2 recognition rates over the whole test database were 80.1% and 85.4%, respectively, compared to 88.9% and 92.7% using our approach. However, we should note that [9] is designed only for the removal of stroke-like noise patterns, and the images in our test database contain other types of noises as well (Table 2). Therefore, we manually selected a subset of the test images that only contain stroke-like pattern noises. The top-1 and top-2 recognition rates using [9] over this subset were 88.9% and 92.6%, respectively, compared to 89.3% and 92.8% using our approach. This means that although the authors in [9] did not consider the recognition rate as a design criterion, their denoising method can be used in recognition applications where we know the image is mainly contaminated by stroke-like noise patterns. In practice, [9] can be combined with other denoising methods as the authors have mentioned. Our results are slightly better for stroke-like noise patterns because, first, our method is based on the optimization of the recognition performance, and second, the recognition performance is not sensitive to single misclassifications in the initialization step.

In order to see how a general-purpose denoising method would perform in the context of handwritten noise patterns, we experimented with ScanFix[2] which is a state-of-the-art application development kit that is used in successful commercial OCR software packages such as PrimeOCR[3] .

Fig. 13(a) shows a word image with impulsive and background line noise patterns. Knowing that the image is contaminated by these two types of noise, we defined the two corresponding denoising filters in ScanFix: despeckle and line removal. Then, we fine-tuned the parameters of each filter so that we obtained the denoised image of Fig. 13(b). The selected values of parameters for this image were as follows: speck width = 10; speck height = 13 (for the despeckle filter); and maximum character repair size = 25; maximum gap = 10; maximum thickness = 10; minimum aspect ratio = 10; minimum length = 10 (for the line removal filter). Fig. 13(f) and (j) show the denoising results for the images of Fig. 13(e) and (i) with the same filters that were optimized for Fig. 13(a). As can be seen in these examples, there

[2] http://www.accusoft.com/scanfix.htm.

[3] http://primerecognition.com/augprime/prime_ocr.htm.

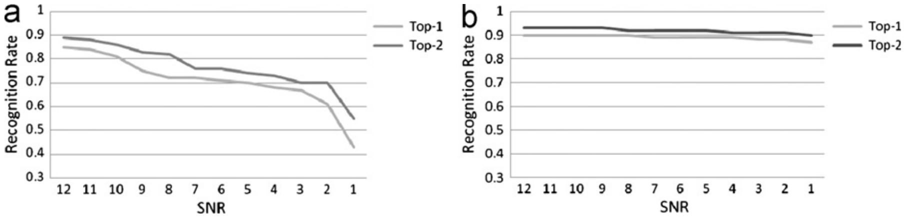

Fig. 12. Recognition rate versus SNR with and without denoising. (a) without denoising; (b) with denoising.

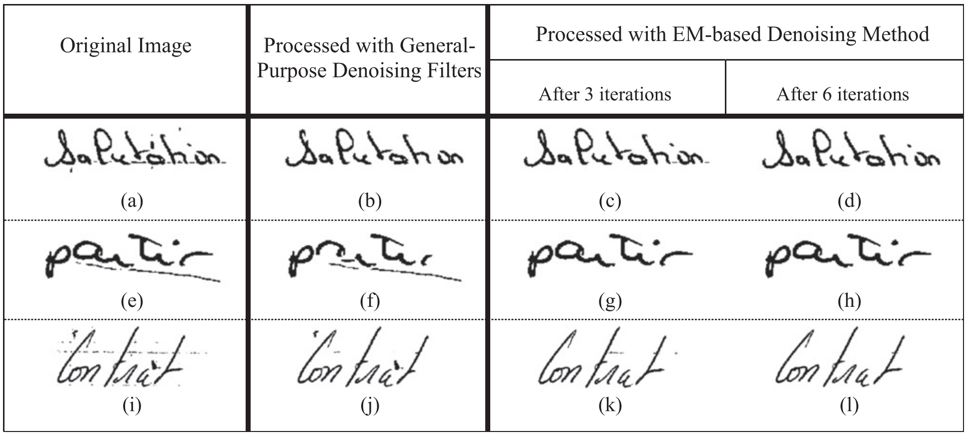
Original Image
Processed with General-Purpose Denoising Filters
Processed with EM-based Denoising Method
After 3 iterations
After 6 iterations
(a)
(b)
(c)
(d)
(e)
(f)
(g)
(h)
(i)
(j)
(k)
(l)

Fig. 13. Results of the processing of some noisy handwritten word images using a general-purpose vs. proposed denoising method.

are three problems concerning the use of a general-purpose filtering method for the removal of noise patterns in handwritten images. First, we have to know the types of noise that exist in the input image so that we are able to specify the appropriate set of filters. Second, a set of filters that are optimized for one image does not necessarily result in the best output for another image. Third, there is no guarantee that a general-purpose filter always keeps all parts of the data, as can be seen the character dots are removed in Fig. 13(b) and (f). The results of the processing of the original word images using our proposed denoising method are shown in the third and forth columns of Fig. 13. As can be seen, in all cases, the algorithm is able to completely separate the noise from the text after 6 iterations of the EM. More discussion about the convergence speed is given in the next section.

We repeated the same recognition experiments on the subset of the test images that only contained stroke-like pattern noises. The top-1 and top-2 recognition rates using the despeckle+line removal filters over this subset were 83.8% and 87.9%, respectively, which is unsurprisingly lower than the results using the high-level denoising methods reported above.

6.2. *Analysis of speed*

To ensure the practicality of the approach, we must show that the improved recognition performance is not at the cost of sacrificing too much speed.

For an input image I, the runtime of the algorithm is $2N \times O_R(I) \times T$, where N is the number of connected components, $O_R(I)$ is the recognition time, and T is the number of iterations required by the optimization process. Therefore, in order to assess the run-time performance of the algorithm, we carried out some experiments to analyze T as a function of the quality of the initial guess in terms of how close/far it is to/from the final solution. The quality of an initial guess depends on the distributions of noise and text and the decision function, which determine the number of randomly initialized latent variables $N_{Z|R}$ and the number of incorrectly initialized latent variables $N_{Z|I}$. In general, the higher the number of incorrectly initialized random variables, the more the number of iterations required for the convergence. Assuming that the chance of a randomly initialized latent variable to be correctly initialized is 1/2 on average, we can define $m_Q = 1 - N_{Z|R,I}/N$ as a measure of quality of an initial guess, where $N_{Z|R,I} = N_{Z|R}/2 + N_{Z|I}$. In the absence of FIS systems, $N_{Z|R} = N$, $N_{Z|I} = 0$ and $m_Q = 0.5$. Therefore, the condition for the FIS systems to improve the convergence speed is that $m_Q > 0.5$.

Fig. 14 shows the average m_Q over the test database using the decision function defined in Eq. (14) as a function of the parameters $\alpha = (\alpha_{\text{val\_min}}, \alpha_{\text{diff\_min}})$. The definition of the decision function implies that we avoid random initialization when the following two conditions are met: 1) the estimation of the density function for one class is high (larger than $\alpha_{\text{val\_min}}$); and 2) the estimation of the density function for one class is higher than that of the other class by a certain amount ($\alpha_{\text{diff\_min}}$). The lower $\alpha_{\text{val\_min}}$ and $\alpha_{\text{diff\_min}}$, the higher the chance of incorrect initialization. As can be seen in Fig. 14, $m_Q > 0.5$ is met everywhere except for the blue area where the parameters are both low ($\alpha_{\text{val\_min}} < 0.4$ and $\alpha_{\text{diff\_min}} < 0.4$). On the other hand, the higher $\alpha_{\text{val\_min}}$ and $\alpha_{\text{diff\_min}}$, the lower the chance of incorrect initialization, and the higher the chance of random initialization. The right compromise between the correct initialization rate and the random initialization rate is made when $\alpha_{\text{val\_min}}$ and $\alpha_{\text{diff\_min}}$ are neither too low nor too high. In our experiments, the average m_Q reached its maximum of 0.85 at $0.6 \le \alpha_{\text{val\_min}} \le 0.7$ and $0.3 \le \alpha_{\text{diff\_min}} \le 0.4$.

Fig. 15 shows the average number of iterations required by the optimization process T as a function of the average quality of the initial guess m_Q over the test database. At $m_Q = 0.5$, which corresponds to initialization without FIS systems, the average number of iterations is 8 (between 7 to 9 for different sizes of inputs) which is reduced to an average of 2 to 3 iterations at the highest m_Q which corresponds to the initialization using FIS systems with the optimized decision function.

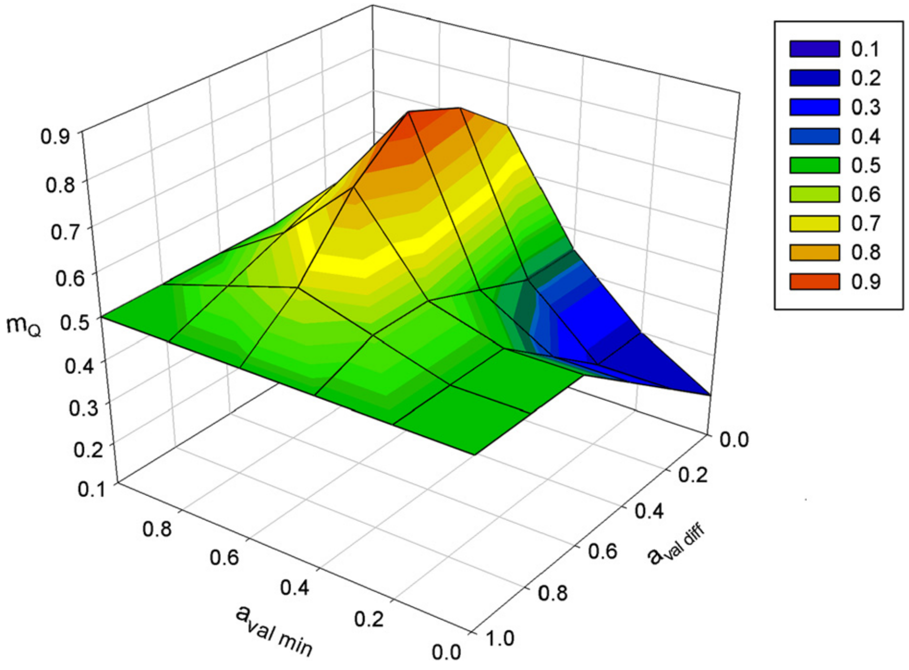

Fig. 14. Average m_Q as a function of the parameters of the decision function $\alpha_{\text{val\_min}}$ and $\alpha_{\text{diff\_min}}$.

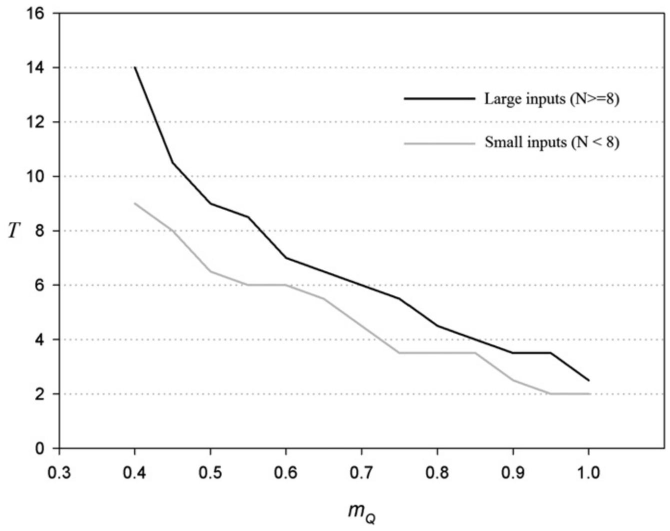

Fig. 15. Average number of iterations T as a function of m_Q for different input sizes.

7. Conclusion

We presented a novel approach to the removal of noise patterns from handwritten images for recognition applications. The difficulty of the problem lies in the fact that the family of noise patterns that appear in handwritten images could be large (or virtually unlimited) and some classes of noise patterns look similar to certain characters or parts of characters. Therefore, we proposed an unsupervised learning approach that does not rely on the noise patterns belonging to any particular distribution. We formulated the noise removal and recognition as a single optimization problem involving latent variables. Thus, we used the EM algorithm in order to find the values of the latent variables (and therefore the noise patterns) based on an optimization criterion which is defined to be the recognition score for the input image after noise removal. In this sense, the main novelty of our work is to propose a noise removal algorithm for improving the recognition performance of document processing systems without making any particular assumption about the distribution of noise patterns.

However, the benefit of our approach comes at the cost of higher computational complexity. We showed that under the non-parametric assumption about noise patterns, the denoising/recognition time for a noisy input is higher than the recognition time for a noise-free input by a factor of two times the number of connected components of the input in each iteration of the optimization process. Therefore, in order to speed up the convergence, we presented a method based on fuzzy logic to incorporate prior knowledge into the optimization process. We showed that for some common classes of noise patterns, we can utilize FISs to improve the initial guesses for latent variables. Our runtime analysis and experimental results confirmed that the improved choice of initial guesses is an important factor in reducing the convergence time of the algorithm.

We developed and evaluated our method for the processing of French documents, but it should be mentioned that it can be applied to other Indo-European languages such as Spanish, English, Arabic, etc., with no or little modifications in the fuzzy inference systems. The scope of applicability of our method is not limited to the denoising of word images. Text detection in natural scene images, for example, can be formulated in a similar way as a binary classification problem where the distribution of only one of the classes (i.e., text) is known. Therefore, it would be interesting to study the extent to which one-class classification approaches can be used in such denoising/detection/recognition applications as well.

Acknowledgments

The authors would like to thank Dr. Dominique Ponson, the Vice President, Research and Development, of IMDS Software for providing the necessary research facilities as well as bringing valuable ideas and insights throughout the word spotting project. The authors would also like to thank the MITACS and NSERC of

Canada for financial support of this research through the MITACS Accelerate Award and the CRD grant.

References

[1] D. Cho, T.D. Bui, Multivariate statistical modeling for image denoising using wavelet transforms, Signal Processing: Image Communication 20 (1) (2005) 77–89.

[2] X. Zhang, X. Jing, Image denoising in contourlet domain based on a normal inverse Gaussian prior, Digital Signal Processing, 20, Academic Press, Inc., 2010 1439-1446.

[3] D. Zhang, S. Mabu, K. Hirasawa, Image denoising using pulse coupled neural network with an adaptive Pareto genetic algorithm, IEEJ Transactions on Electrical and Electronic Engineering, Wiley Subscription Services, Inc., A Wiley Company, 2011.

[4] A. Buades, B. Coll, J.-M. Morel, A Non-local algorithm for image denoising, Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) – Volume 2 – Volume 02, IEEE Computer Society, 2005, 60–65.

[5] K.-C. Fan, Y.-K. Wang, T.-R. Lay, Marginal noise removal of document images, Pattern Recognition *35* (2002) 2593–2611.

[6] F. Shafait, J. van Beusekom, D. Keysers, T. Breuel, Document Cleanup Using Page Frame Detection, 11, International Journal on Document Analysis and Recognition, Springer, Berlin/Heidelberg, 2008 81–96.

[7] M.M. Haji, T.D. Bui, C.Y. Suen, Simultaneous document margin removal and skew correction based on corner detection in projection profiles, ICIAP '09: Proceedings of the 15th International Conference on Image Analysis and Processing, Springer-Verlag, 2009, 1025–1034.

[8] M. Agrawal, D. Doermann, Clutter Noise Removal in Binary Document Images Proceedings of the 2009 10th International Conference on Document Analysis and Recognition, IEEE Computer Society, 2009, 556–560.

[9] M. Agrawal, D. Doermann, Stroke-like pattern noise removal in binary document images, International Conference on Document Analysis and Recongition, 2011.

[10] J.-S.R. Jang, C.-T. Sun, Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, Prentice Hall, 1997.

[11] Cordon, Oscar, Herrera, Francisco, F.H.L.M., genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases (advances in fuzzy systems—Applications & Theory), World Scientific Publishing Company, 2002.

[12] H.-C. Chen, W.-J. Wang, Efficient impulse noise reduction via local directional gradients and fuzzy logic, Fuzzy Sets and Systems *160* (2009) 1841–1857.

[13] J.-G. Camarena, V. Gregori, S. Morillas, A. Sapena, Two-step fuzzy logic-based method for impulse noise detection in colour images, Pattern Recognition Letters 31 (2010) 1842–1849.

[14] T.-C. Lin, Decision-based fuzzy image restoration for noise reduction based on evidence theory, Expert Systems with Applications *38* (2011) 8303–8310.

[15] T. Mélange, M. Nachtegael, S. Schulte, E.E. Kerre, A fuzzy filter for the removal of random impulse noise in image sequences, Image and Vision Computing *29* (2011) 407–419.

[16] M.S. Nair, G. Raju, Additive noise removal using a novel fuzzy-based filter computers & electrical engineering, 2011.

[17] F. Sattar, D. Tay, Enhancement of document images using multiresolution and fuzzy logic techniques, IEEE Signal Processing Letters 6 (10) (1999) 249–252.

[18] R. Ranawana, V. Palade, G.E.M.D.C. Bandara, Automatic fuzzy rule base generation for on-line handwritten alphanumeric character recognition, International Journal of Knowlege-Based Intelligence Engineering Systems, 9, IOS Press, 2005 327-339.

[19] M. Zimmermann, H. Bunke, Automatic segmentation of the IAM off-line database for handwritten english text, Pattern Recognition, International Conference on, IEEE Computer Society, 2002, 4, 35–39.

[20] E.V. Broekhoven, B.D. Baets, Fast and accurate center of gravity defuzzification of fuzzy system outputs defined on trapezoidal fuzzy partitions, Fuzzy Sets and Systems 157 (2006) 904–918.

[21] R.C. Gonzalez, R.E. Woods, Digital Image Processing, 3rd ed., Prentice Hall, 2007.

[22] M. Blumenstein, Intelligent Techniques for Handwriting Recognition, School of Information Technology, Faculty of Engineering and Information Technology, Griffith University-Gold Coast Campus, 2000.

[23] X. Wang, X. Ding, C. Liu, Gabor filters-based feature extraction for character recognition, Pattern Recognition *38* (2005) 369–379.

Mehdi Haji is a postdoctoral fellow at department of computer science and software engineering, Concordia University, Montreal, Canada. He started his research in the filed of pattern recognition and document analysis during his Masters program which was on the recognition of handwritten Farsi words using continuous hidden Markov models. He completed his PhD under supervision of Drs. T. D. Bui and C. Y. Suen. The title of his doctoral thesis is Search and Classification of Unconstrained Handwritten Documents. Mehdi's research interests include document image analysis and understanding, statistical machine learning and soft computing. Mehdi has been awarded several prestigious awards during his doctoral studies at Concordia University including Dominic D'Alessandro Fellowship, Campaign for a New Millennium Student Contribution Graduate Scholarship, and Power Corporation of Canada Graduate Fellowship.

T. D. Bui is a Full Professor in the Department of Computer Science and Software Engineering, Concordia University, Montreal, Canada. Currently, he is an Associate Editor of Signal Processing (EUROSIP), the International Journal of Wavelets, Multi-resolution and Information Processing, and the Journal of Wavelets and Applications. Dr. Bui is co-author of the book Computer Transformation of Digital Images and Patterns published by World Scientific Publishing Co. 1989. He was a visiting professor at the Department of Mechanical Engineering, and the Lawrence Berkeley Lab. of the University of California at Berkeley in 1983–1984.

C. Y. Suen is the Director of CENPARMI at Concordia University, Montreal, Canada. Currently, he holds the distinguished Concordia Research Chair position in Artificial Intelligence and Pattern Recognition. He has guided/hosted 70 visiting scientists and professors, and has supervised 65 doctoral and master's graduates. He has served several professional societies as President, Vice-President, or Governor. He is also the Founder and Chair of several conference series, including ICDAR, ICFHR. Suen is a recipient of numerous prestigious awards, including: the IAPR ICDAR Award in 2005; and the recent ENCS Lifetime Awards in 2008.