Contents lists available at SciVerse ScienceDirect

Image and Vision Computing

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

Offline text-independent writer identification using codebook and efficient code extraction methods☆

Golnaz Ghiasi, Reza Safabakhsh *

Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 4 May 2012
Received in revised form 27 January 2013
Accepted 24 March 2013

Keywords:
Handwritten documents
Writer identification
Farsi handwriting
English handwriting
Codebook

ABSTRACT

In this paper, an efficient method for text-independent writer identification using a codebook method is proposed. The method uses the occurrence histogram of the shapes in a codebook to create a feature vector for each specific manuscript. For cursive handwritings, a wide variety of different shapes exist in the connected components obtained from the handwriting. Small fragments of connected components are used to avoid complex patterns. Two efficient methods for extracting codes from contours are introduced. One method uses the actual pixel coordinates of contour fragments while the other one uses a linear piece-wise approximation using segment angles and lengths. To evaluate the methods, writer identification is conducted on two English and three Farsi handwriting databases. Both methods show promising performances with the performance of second method being better than the first one.

© 2013 Elsevier B.V. All rights reserved.

1. Introduction

Handwriting is a behavioral biometric which can be used for writer identification and verification. This biometric can be applied in different domains such as security [1,2], forensic and criminal justice systems [3] and historical document analysis [4,5]. In addition, it can be used for the enhancement of handwriting recognition systems [6].

Writer recognition includes writer identification and writer verification. In writer identification, the writer of query handwriting should be determined among a number of writers for each of whom sample handwriting is available. In writer verification, the goal is to determine whether the writer of two handwritten texts is the same person or not. Writer identification and verification can be performed offline, where only the images of handwritings are available; or can be done online, where dynamic features such as pen speed, pen pressure and height of the pen are also available further to the image of the handwriting. Finally, writer identification and verification can be divided into two categories of text-dependent and text-independent. In text-dependent methods, writers are required to write a specific fixed text to perform the identification or verification. In contrast, in text-independent methods, any text can be used for this purpose.

Fragmented parts of different people's handwritings are quite different. As a result, the occurrence histogram of different fragments appearing in handwriting can be employed as a feature vector for writer identification. To construct this histogram, a collection of fragments that usually appear in handwritings should be available. This collection is called the "codebook", and each of its members is called a "code". In cursive handwritings, connected components may be too long and may have a wide variety of shapes. But smaller fragments or codes are more desirable. This paper presents two novel methods for extracting simple fragments from connected components and using them to efficiently recognize human handwritings. The methods are offline and text-independent and are evaluated in English and Farsi writer recognition tasks, showing promising performances. The organization of the paper is as follows. In the next section, the related work is discussed. Section 3 describes the details of the proposed method, and Section 4 presents the experimental results. Finally, Section 5 summarizes the conclusions of the paper.

2. Related work

Various approaches for offline text-independent writer identification have been introduced in the previous studies. Some of the studies treat the writer identification task as a texture analysis problem. Said et al. [7] used Gabor filters and co-occurrence matrices for extracting features from handwritings and reported an accuracy of 96% on English handwritings of 40 people. Shahabinejad and Rahmati [8] proposed a method that extracts features from the outputs of Gabor filters based on the moments and a nonlinear transform. Their correct identification rate for Farsi handwritings of 40 people was 82.5%. Helli and Moghaddam [9] extracted features using the Gabor and Extended Gabor filters and used an LCS (longest common subsequence) based classifier to classify the features. Their correct writer identification rate was 95% on the PD100 database which contains Farsi handwriting of 100 people. In another work [10] they represented Gabor and Extended Gabor features using a Feature Relation Graph (FRG) for

☆ This paper has been recommended for acceptance by Cheng-Lin Liu.
* Corresponding author. Tel.: +98 21 64542728; fax: +98 21 66495521.
E-mail addresses: g.ghiasi@aut.ac.ir (G. Ghiasi), safa@aut.ac.ir (R. Safabakhsh).

0262-8856/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.imavis.2013.03.002

each person. This graph was constructed using a fuzzy method. In addition, a graph similarity algorithm was employed for identification. Their correct identification rate was about 100% when sufficient training data was used for 80 writers of PD100 database.

Schlapbach and Bunke [11,12] used a Hidden Markov Model (HMM) for writer identification. This method trains an individual recognizer for each writer using a text written by that writer. After training, each of the recognizers acts as an expert on the handwriting of exactly one writer. When a text of unknown writer is presented to the recognizers, each of them gives a score. The recognizer with the highest score determines the writer. A result of 97.03% correct identification rate was reported on a subset of the IAM database [13] containing 100 writers [11]. The same authors proposed the Gaussian Mixture Model (GMM) for writer identification [14]. This method also trained a model for each of the writers and every model became an expert on the handwriting of a particular writer. The resulting system was compared with a system based on Hidden Markov Models, and it was concluded that this system was simpler and faster for training. In addition, its performance was higher and their reported result was 98.4% on a part of the IAM database with 100 writers.

Features that attempt to describe visual characteristics of handwritings, such as height, width and the slant of different zones of handwritings can also be used for writer identification and verification. Marti et al. [15] extracted 12 visual features from handwritings and reported a performance of 90.7% on a part of IAM database containing English handwritings of 20 persons. Sadeghi and Moghaddam [16] used grapheme based and gradient features and fuzzy clustering method. Their accuracy was 90% on 50 writers from PD100 database.

Siddiqi and Vincent [17] developed a method which combined global and local features for writer identification. In the first step, global features of handwritings were exploited to classify handwritings. The Gabor filter was used to localize the orientated segments in handwritings and determine the handwriting's class according to the direction criteria. Then, in the second step, the writer of the query was identified by just searching in the specific class using local features. For this purpose, the patterns which occurred frequently in the handwriting of a

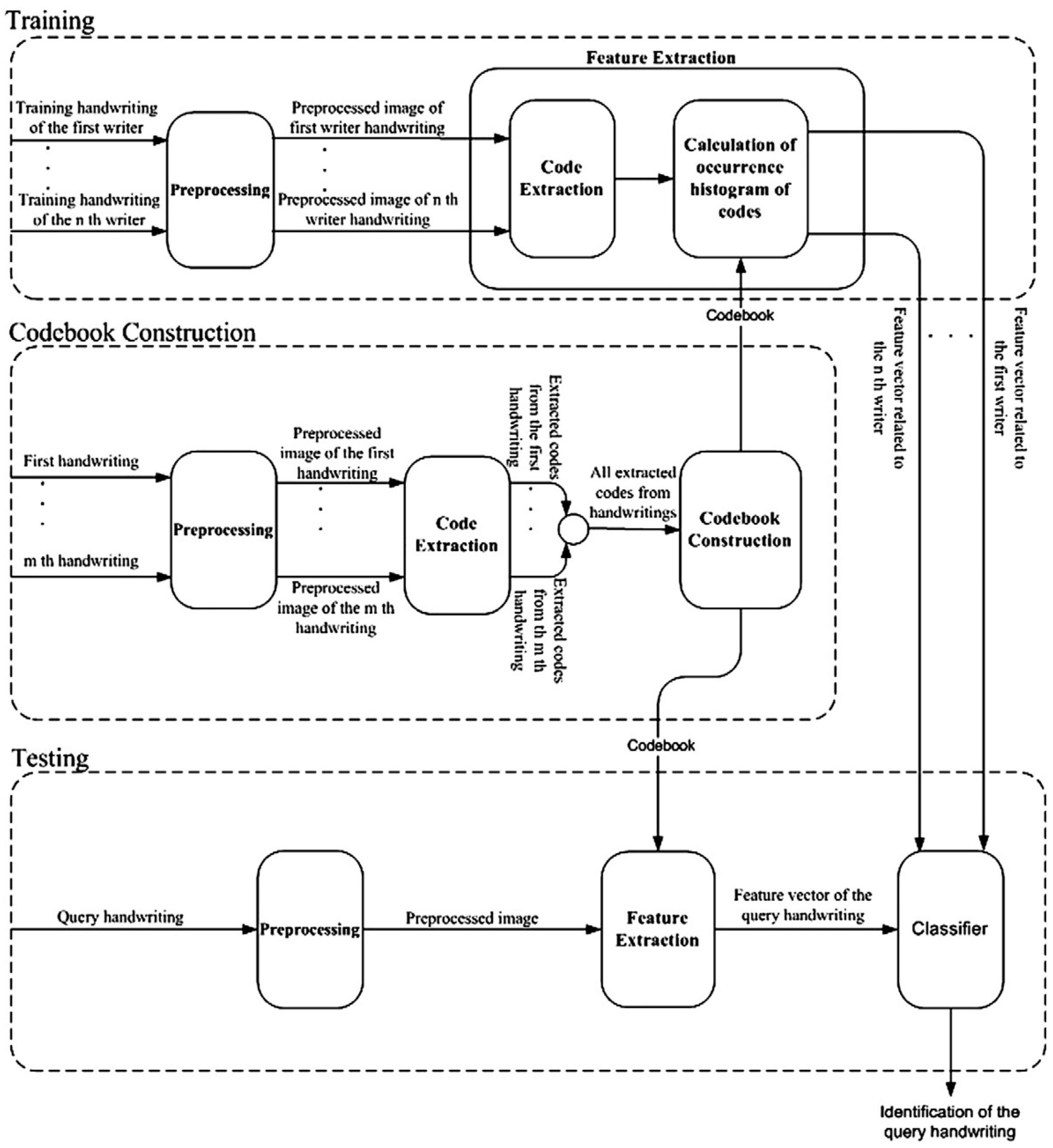
Training
Feature Extraction
Training handwriting of the first writer
Training handwriting of the n th writer
Preprocessing
Preprocessed image of first writer handwriting
Preprocessed image of n th writer handwriting
Code Extraction
Calculation of occurrence histogram of codes
Codebook
Feature vector related to the n th writer
Feature vector related to the first writer
Codebook Construction
First handwriting
m th handwriting
Preprocessing
Preprocessed image of the first handwriting
Preprocessed image of the m th handwriting
Code Extraction
Extracted codes from the first handwriting
Extracted codes from th m th handwriting
All extracted codes from handwritings
Codebook Construction
Codebook
Testing
Query handwriting
Preprocessing
Preprocessed image
Feature Extraction
Feature vector of the query handwriting
Classifier
Identification of the query handwriting

Fig. 1. Block diagram of the writer identification system.

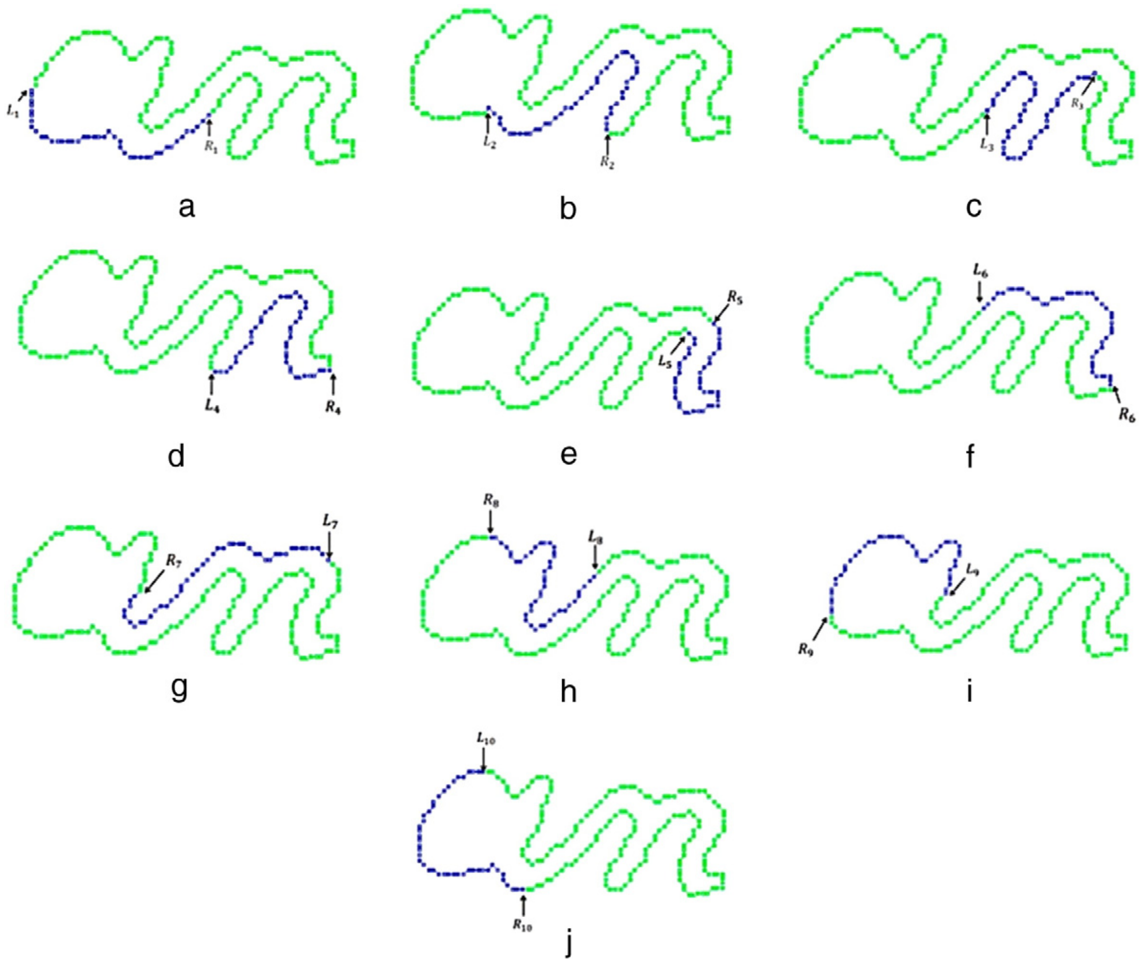
a
b
c
d
e
f
g
h
i
j
L_1 R_1 L_2 R_2 L_3 R_3 L_4 R_4 L_5 R_5 L_6 R_6 L_7 R_7 L_8 R_8 L_9 R_9 L_{10} R_{10}

Fig. 2. Curve fragment code extraction approach for the word *an* with the parameters $L_f = 44$ and $gap = 22$, (a) L_1 and R_1 are the left and the right of a segment with L_f points on the contour. This segment is considered as a code, (b), (c), (d), (e), (f), (g), (h), (i), (j) L_i and R_i are the left and the right of a segment on the contour and they are determined by shifting L_{i-1} and R_{i-1} as much as *gap* points. Fragments are considered as a code.

writer were determined by clustering a large number of small sub-images from that handwriting's image. Writer identification was performed by comparing patterns of different writers and the Bayesian classifier. The correct identification performance of the system on a part of the IAM database containing handwritings of 100 people was reported 92%. In another work by the latter authors [18], a set of features which extracts characteristics about contours at global and local levels was introduced. The system's identification performance on the IAM database with the handwritten texts of 650 writers was 86%. In addition, the same authors introduced a set of features that extracted orientation and curvature information from chain code sequence and polygon approximations of contours [19]. The performance of the method on the same database was 89%.

Bulacu et al. [20,21] proposed the use of contour-hinge computed by extracting directions of two contour fragments attached at a common end pixel. Their achieved performance using only this feature on the Firemaker lowercase database [22] was 81%. Brink et al. [23] showed that width patterns are valuable. They introduced Quill and

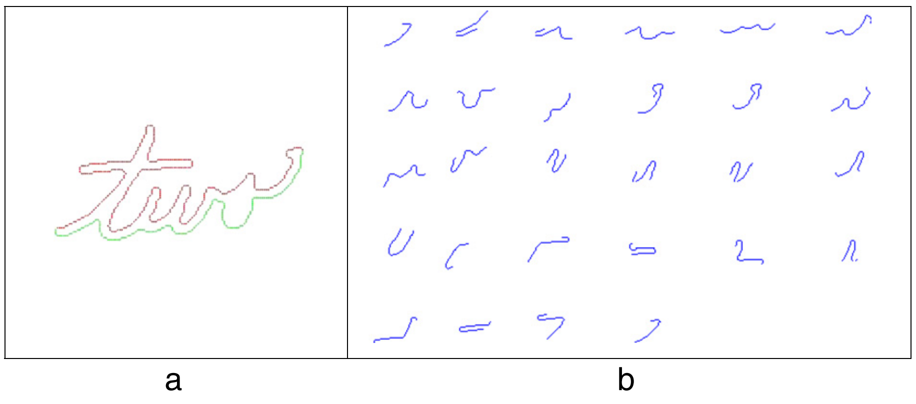

Fig. 3. Code extraction using curve fragment method: (a) contour of the word *two*, and (b) shapes of extracted fragments using curve fragment method when $L_f = 50$ and gap $= 20$.

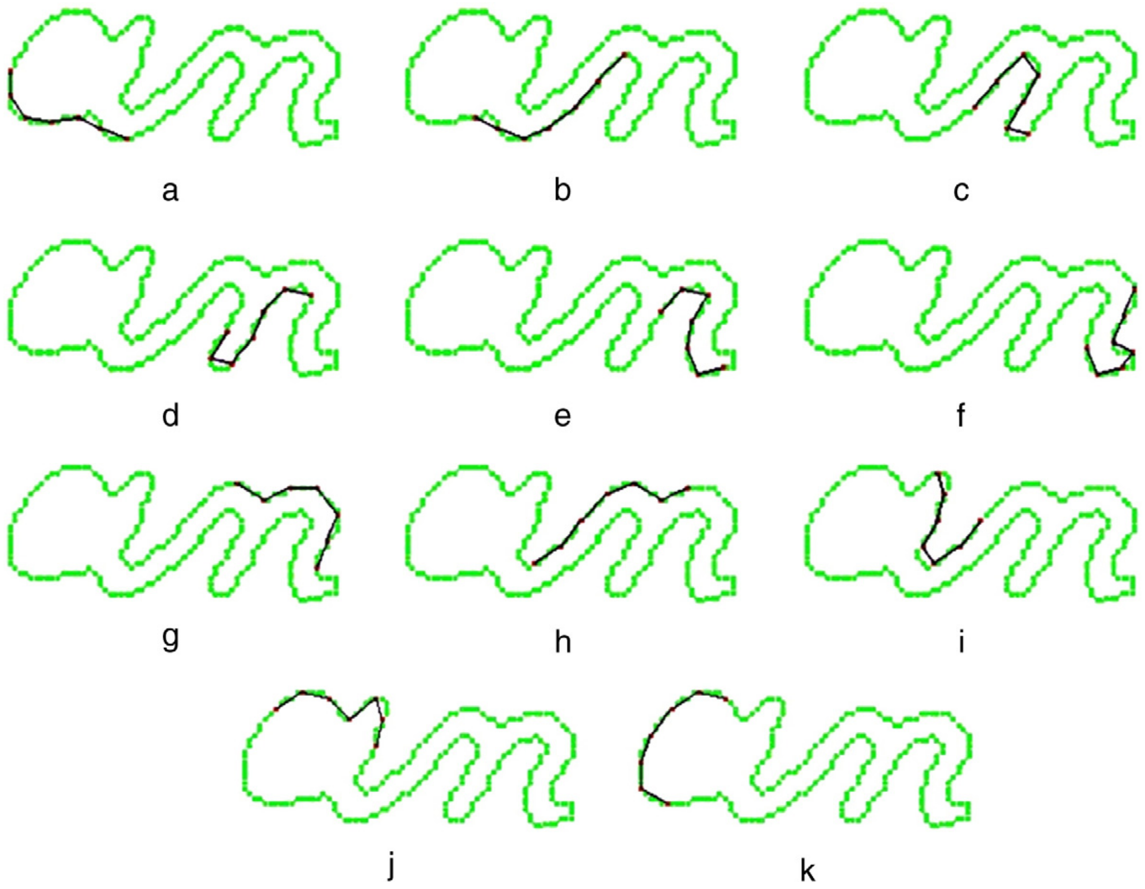

Fig. 4. Line fragment code extraction approach for the word *an* with the parameters $N_f = 6$, $L_f = 5$, and $gap = 20$: (a) lines that connect $Nf = 5$ marked points, which the fragment length between each of the two successive of them is $L_f = 5$ points, estimate the shape of a part of the contour. The angles and lengths of these lines make a code, (b), (c), (d), (e), (f), (g), (h), (i), (j), (k). Six points are shifted *gap* points each time and make new codes.

QuillHinge features which combine the ink width and ink directions. Their writer identification performance on Firemaker lowercase database was 86% and it was 97% on 650 writers of the IAM database.

Bensefia et al. [24,25] clustered the graphemes resulting from a handwriting segmentation method to produce a codebook called 'writer invariants' and identified writers by comparing the invariant patterns extracted from two sample texts. Their method's correct performance was 86% on 150 writers of the IAM database.

Instead of calculating invariant clusters for each writer, a common codebook for all the writers can be calculated. This codebook contains shapes which usually appear in people's handwritings. After having this codebook, handwriting feature vectors can be determined by computing the occurrence histogram of the members of the codebook in handwritings. In non-cursive handwritings, each connected component can be considered as a code for making a codebook [26]. But in cursive handwritings, connected components may contain several characters; so, they may be too long and have a wide variety of shapes. In such cases, shorter fragments are desirable. Schomaker and Bulacu [20,27–29] introduced a method in which the vertical local minima of the lower contour are found, and the nearest local

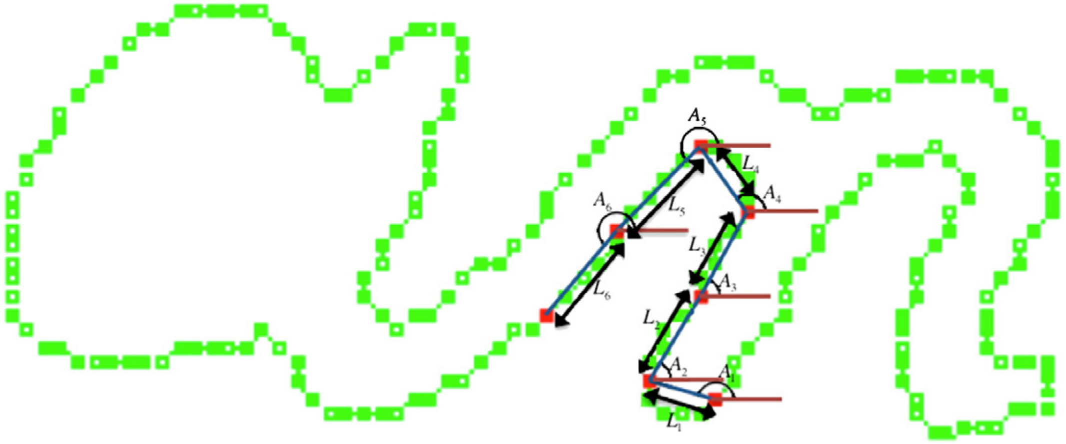

Fig. 5. Angles and lengths of the lines that connect the marked points of Fig. 4 part (c). Corresponding code of these points is determined by A_1, A_2, A_3, A_4, A_5, A_6, L_1, L_2, L_3, L_4, L_5, and L_6.

minimum in the upper contour for each of them is determined. If the distance between two corresponding points is in the order of the ink-line width, the connected component will be segmented at that point. Their reported performance was 75% using only this method on Firemaker lowercase database. In their method, connected components are preferably divided in such a way that the resulting fragments are meaningful. In fact, a similar segmentation method has been successfully used for cursive text recognition [30]. But extracted fragments for writer identification do not need to be meaningful, and only their shapes are important. Moreover, segmentation of a connected component in its local minima results in losing information about the shape of the local minima. In a previous work by the authors of this paper [31], the codebook method was used with an improved algorithm for the extraction of smaller parts. The results of the two methods on Farsi handwritings were compared and it was shown that the new method has a better performance. The performance of the method on Farsi handwritings of 180 people was 98.8%, when the size of the handwriting samples was one page. In another study [32], Siddiqi and Vincent extracted small parts of the handwritten text as the codes. These parts did not carry any semantic information and were obtained by the division of the handwriting's image into small windows [17]. Small windows were clustered by a hierarchical clustering algorithm. The performance of their method, on the 650 writers of the IAM database, was 84%. In addition, they combined the contour and codebook features and the performance of 91% was achieved on the same database. In this paper, two novel methods for code extraction are introduced and their performance in English and Farsi handwriting recognition is evaluated.

3. The proposed methods

The block diagram of a writer identification system is shown in Fig. 1. The system includes three main phases: codebook construction, training and testing. For codebook construction, a sufficient number of handwritings (m) are selected and after preprocessing their images, the code extraction method is applied to each one of them. Using all the codes extracted from these handwritings in a clustering method, a codebook of frequently appearing codes in handwritings is computed. In the training phase, a feature vector is extracted from every training handwriting sample. For this purpose, each image is preprocessed first and all its codes are extracted. Then, the most similar code to each extracted code is found in the codebook and the occurrence histogram of the codebook codes in that handwriting is computed. This histogram is normalized and used as a feature vector. In the testing phase, the query handwriting with an unknown writer is input to the system. The preprocessing and code extraction steps are performed and the feature vector is computed. Using this feature vector in the trained classifier, the writer of the query is identified.

In the preprocessing stage, first, the average filter with a window size of three is applied to the scanned image. Then, the image is binarized and its connected components are detected using 8-connectivity. In the last step, contours of connected components are extracted using the Moore's algorithm.

The rest of this section explains the important components of the system. The code extraction methods are introduced in Section 3.1. Details of codebook construction and making feature vectors are described in Section 3.2 and writer identification using feature vectors is explained in Section 3.3.

3.1. Code extraction

Two novel methods for code extraction from contours of connected components are introduced in this section. The first method utilizes the small curves obtained when moving on the contours of connected components and the second method approximates the shape of contours with a number of small line segments.

3.1.1. Curve fragment code extraction

This method utilizes curves from the contours of connected components for code extraction. For this purpose, contour fragments with a specific length or number of points (L_f) are extracted from contours. Each contour fragment with L_f points can be used as a code. The number of codes can be controlled by changing the distances between the starting points of fragments. These distances can be fixed to a certain value called the *gap* parameter. The code extraction approach with the curve fragment method for a sample contour is shown in Fig. 2. Fig. 3 shows the contour of the word ‘two’ and some extracted curve fragments using this method.

Coordinates of the fragment points are normalized to the origin (0,0) and the standard deviation of one, using the following relations [28]:

$$\vec{x} \leftarrow \left(\vec{x} - \mu_x\right)/\sigma_x \qquad (1)$$
$$\vec{y} \leftarrow \left(\vec{y} - \mu_y\right)/\sigma_y$$

where $\vec{x}$ and $\vec{y}$ are the collections of x and y coordinates of a contour fragment, μ_x and μ_y are averages, and σ_x and σ_y are the standard deviations of $\vec{x}$ and $\vec{y}$, respectively. Vectors containing coordinates of the normalized fragments are all of a length of $2L_f$ and are used as

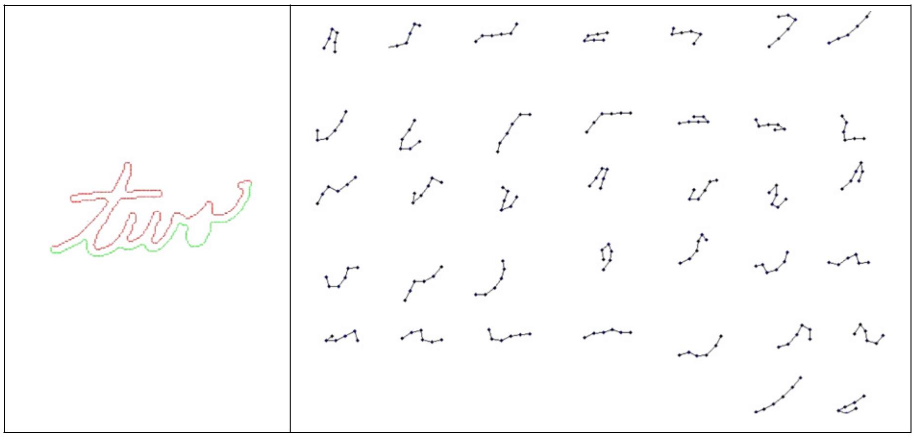

Fig. 6. Code extraction using line fragment method: (a) contour of the word *two*, and (b) shapes of extracted code using line fragment method when $N_f = 5$, $L_f = 7$ and gap $= 15$.

codes for training a codebook and extracting feature vectors from handwritings.

3.1.2. Line fragment code extraction

In the line fragment method, curves from the contours of connected components are estimated by the angles and lengths of successive segments. A code is defined with $N_f + 1$ points in which the length of curves between every two successive ones equals L_f. The segments connecting these points estimate the contour fragment with the length $N_f \times L_f$. The number of codes which can be extracted from a contour of connected components is at most equal to the number of points on that contour. The distances between the starting points of these sets of $N_f + 1$ points can be set to a fixed value and the number of extracted codes can be controlled by the value of this parameter. This distance parameter is called the *gap*. Other parameters of this method are the *number of successive fragments* (N_f) and *fragment length* (L_f). Each code includes the angles and lengths of segments connecting the N_f points. This code extraction method applied to the contour of the word *an* is shown in Fig. 4. Angles and lengths of the lines that connect the marked points of Fig. 4(c) are shown in Fig. 5. The corresponding code, defined by these points, is determined by the lengths L_1, L_2, L_3, L_4, L_5 and L_6; and angles A_1, A_2, A_3, A_4, A_5 and A_6. The length of the codes or the number of features in each code is equal to $2N_f$ in this method. The contour of a connected component and the corresponding shapes of some extracted codes are shown in Fig. 6.

3.2. Codebook construction and feature vector computation

Codes extracted from a sufficient number of handwritings can be used to train a clustering algorithm. After training, a specific number of fragments commonly appearing in people's handwritings are determined. Using the K-means, 1D Self-Organizing Map (SOM) and 2D SOM methods for clustering is investigated in [33]. It is shown that similar performances result for these clustering methods and the codebook method is robust to the used clustering technique. In this study, the standard 2D SOM is used for clustering.

After construction of the codebook, the feature vector for handwriting can be calculated by constructing an occurrence histogram whose bins correspond to the number of occurrence of codebook members in the handwriting. To construct this histogram, all codes of the handwriting are first extracted. Then, for each code, the most similar member of the codebook is selected using a Euclidean distance function, and the corresponding bin is increased by one. Finally, the histogram is normalized by the division of its members to the sum of its values. The resulting histogram is used as the feature vector. Therefore, the size of the feature vector is equal to the number of codes of the codebook.

Using the introduced methods, the number of extracted codes can be modified with changing the *gap* parameter. By decreasing this parameter, the number of extracted codes increases. This increase is especially useful when the available text is short.

3.3. Writer identification

The nearest neighbor rule is used as a classifier for doing writer identification using feature vectors. This classifier is simple and does not require any training. This is a good feature for writer identification applications with many classes and a small set of training data per class. In our case, the number of classes or writers of the large English dataset is as many as 900, and there are only two handwritings for each writer.

The nearest neighbor classifier is also an effective method and its good performance is reported in the literature [23,32]. This method may be slower than other classifiers in the test phase; but, it is fast in our case, where the length of the feature vector is equal to the size of the codebook (900) and the total number of handwritings is at most 1800 (for the large English dataset). Therefore, for finding the nearest neighbor of the test handwriting when the leave-one-out cross validation method is used, the distance of its corresponding feature vector should be compared to at most 1800 − 1 other feature vectors and the size of the feature vectors is 900.

The nearest neighbor classifier has been examined before the writer identification task with different distance functions such as X^2, Hamming, Euclid, Minkowski order 3 and Bhattacharya. The X^2 function has been reported to outperform other distance functions [20,32,33]. For this reason, the X^2 function is also used here for measuring the similarity between two handwritings.

The X^2 distance between the two feature vectors p_i and p_j is computed as:

$$X_{ij}^2 = \sum_{k=1}^{|p_i|} \frac{\left(p_{ki} - p_{kj}\right)^2}{p_{ki} + p_{kj}} \tag{2}$$

where $|p_i|$ shows the length of feature vector p_i and p_{ki} and p_{kj} are the k^{th} element of the feature vectors p_i and p_j, respectively. For a query containing an unknown writer text, the distances with all the known writer texts is calculated through Eq. (2). Then, the list of h writers whose handwritings have the h smallest distances to the query is determined. Here, h is the hit list size. If the writer of the query is among the writers in the hit list, identification is considered correct.

Table 1
Databases properties.

	Database name	Number of writers	Language	Length of text per sample	Explanation
1	Firemaker (lowercase)	250	Dutch	3 to 20 lines	Searches and matches performed on pages 1 and 4 of Firemaker database
3	IAM	650	English	3 to 13 lines	IAM database modified to contain 2 samples per writer
4	Large English	900	English	3 to 20 lines	Merger between Firemaker lowercase and IAM database
5	ImUnipen	215	English	–	A part of this database that contains 130 samples was used for English codebook construction.
6	Shahabinejad	40	Farsi	Half page (6 lines)	Arbitrary page of the database divided two equal parts and searches and matches performed on these two parts. This database was used for the construction of Farsi codebook, too.
7	AUT-FH short	180	Farsi	Half page (4 lines)	Searches and matches performed on the half page of page 1 and page 2 of the database
8	AUT-FH medium	180	Farsi	One page (8 lines)	Searches and matches performed on the page 1 and page 2 of the database
9	AUT-FH long	180	Farsi	One and half page (12 lines)	Searches and matches performed on the page 1 and first four lines of page 3, page 2 and second four lines of page 3
10	FHT	260	Farsi	One page (4 to 10 lines)	Searches and matches performed on 2 samples per writer from FHT database
11	Large Farsi	480	Farsi	4 to 10 lines	Merger between Shahabinejad, AUT-FH medium and FHT database

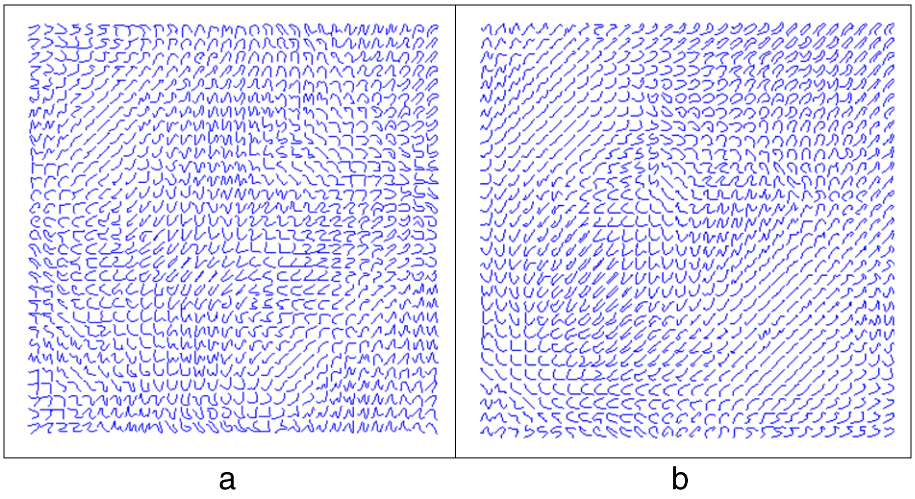

Fig. 7. Patterns of a 30 × 30 codebook when the curve fragment method is used for code extraction: (a) English codebook when $L_f = 50$ points, and (b) Farsi codebook when $L_f = 40$ points.

4. Experimental results

Writer identification is performed with both leave-one-out and 2-fold cross validation strategies. The leave-one-out strategy leads to a more difficult testing situation in which there are two distracters for each false writer; while in the 2-fold strategy, there is just one distracter for each false writer. In the rest of this section, whenever the cross validation strategy is not mentioned explicitly, the leave-one-out strategy is used. Similarly, whenever the hit list size is not specified, it is assumed to be equal to one.

The databases used for conducting the experiments are introduced in Section 4.1. Then, in Section 4.2, the details of codebook construction are explained. The results are discussed in Section 4.3. Finally, the comparison of the results of this study with those of the existing studies is presented in Section 4.4.

4.1. Databases

Experiments are performed on three English and three Farsi databases. Sheets of handwritings in all databases are scanned at 300 dpi in 256 gray scales. English databases include Firemaker [22], IAM [13] and ImUnipen [34]. These English databases are the same as those used in [20], and have been prepared similarly. The Firemaker database contains handwritings of 250 Dutch persons each of whom has written 4 pages. Pages 1 and 4 of this database are used in our experiments. On page 1, subjects were asked to copy a text of five paragraphs; and on page 4, they were asked to describe the content of a given cartoon in their own words. The length of the written text is different for the different writers of page 4; it varies between 2 lines and a full page. The IAM database, after preparation [20], contains two handwritings from 650 individuals. The length of text varies in the samples from three lines up to a full page. The offline version [20] of ImUnipen [34], which is an online database, is used in this study for English codebook generation.

The first Farsi database [8] contains Farsi handwritings of 40 individuals in which every person has written a 12-line arbitrary text. We call it the Shahabinejad database. The second one is the FHT database [35]. This database contains forms with Farsi texts of variable contents. The forms are filled by 260 writers, each of whom has been asked to write 4 forms with different texts. Forms with the same text have been written by about 25 writers. We used a version of this database containing two samples for each writer. The number of lines in the samples is between 4 and 10. The third Farsi database (AUT-FH) is collected by the authors of this paper. It contains handwritings of 180 persons each of whom has written 3 pages of arbitrary 8 line Farsi texts. Three sizes of short, medium and large lengths of written texts are used in the experiments. These sizes are defined by their number of lines as: 4, 8 and 12 lines, respectively. Table 1 contains characteristics of the databases used in the tests.

Table 2
Constructed codebook properties.

Name	Method	Language	Parameters	Length of codes	Number of extracted codes	Training time (hours)
c_1	Curve fragment	English	L_f: 50, gap: 40	100	167526	93
c_2	Curve fragment	Farsi	L_f: 40, gap: 20	80	161755	70
c_3	Line fragment	English	N_f: 6, L_f: 11, gap: 20	12	320048	24
c_4	Line fragment	English	N_f: 6, L_f: 5, gap: 20	12	326288	24
c_5	Line fragment	Farsi	N_f: 6, L_f: 5, gap: 10	12	315809	17

4.2. Constructed codebooks

Siddiqi et al. investigated the impact of the number of handwritten samples used to generate codebook on the writer identification performance [32]. Reported results show that the number of samples does not cause a dramatic change in the results. In this paper, a number of 130 handwritings from ImUnipen database and 40 handwritten samples of the Shahabinejad database are used to extract codes for the training of a SOM neural network. During the SOM training, the learning rate was varied from 0.9, at the beginning, to almost 0, at the end. Furthermore, the neighborhood radius decreased from an initial value equal to the network size to the final value of 1. These parameters were decreased based on a power function. In [33], different codebook sizes from 5 × 5 to 50 × 50 are used for writer identification and it is shown that the performance becomes stable for a codebook size of 10 × 10 or larger. A similar experiment is conducted in [27] and different codebook sizes from 2 × 2 to 50 × 50 are tested. Their results show that writer identification performance for codebooks of size 20 × 20 or larger is almost the same. According to these results, the performance of writer identification is robust to

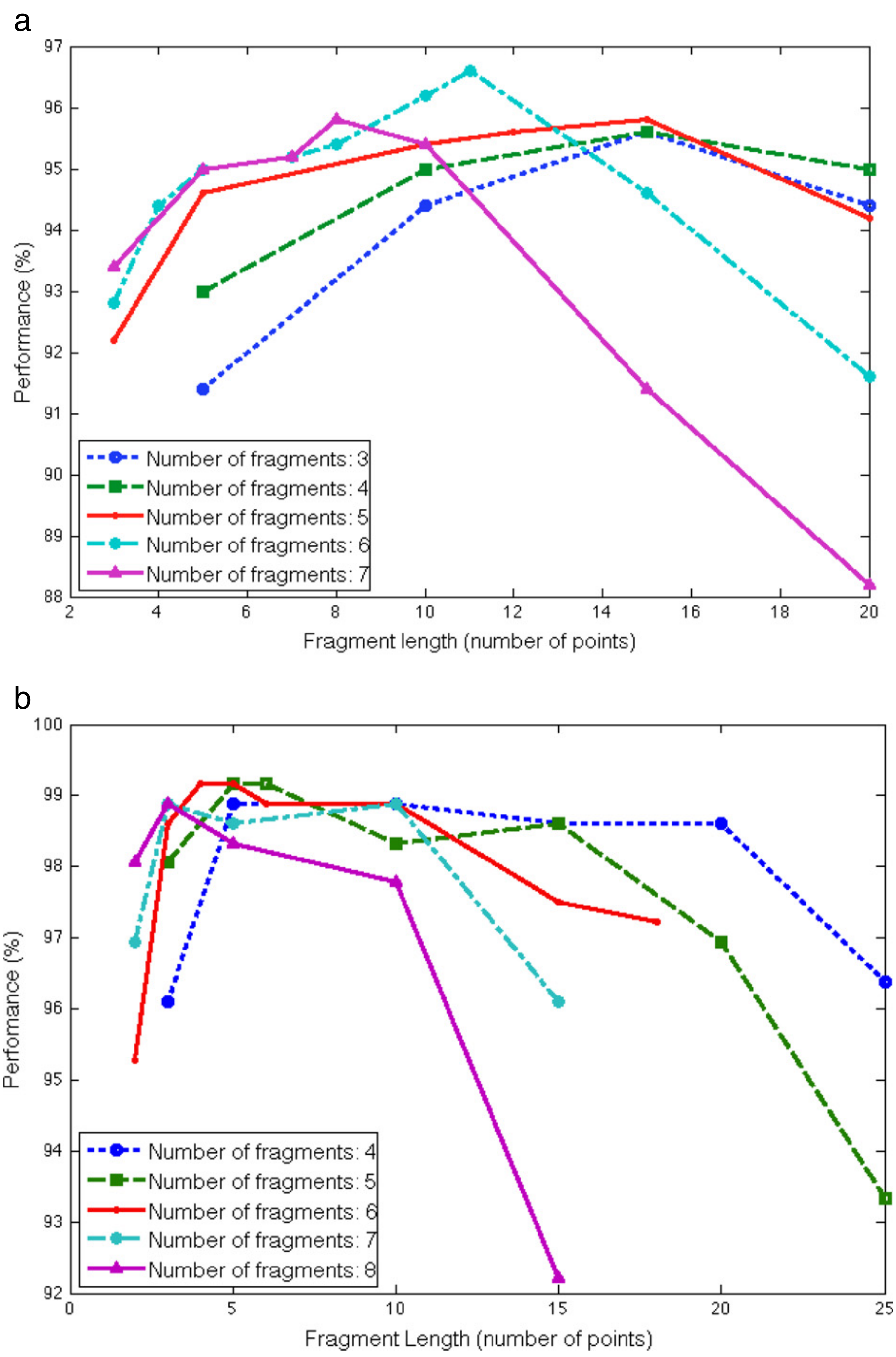

Fig. 8. Performance versus L_f for different N_f values: (a) on the IAM250 database, and (b) on the medium AUT-FH database.

the codebook size and thus the size of SOM is set to 30 × 30. The number of epochs used for training this network is 500.

Appropriate parameter values for the proposed methods are investigated using a validation set and a search on different values. The first 250 writers of the IAM database are chosen as a validation set for English handwritings. This set is called IAM250. Since in Firemaker database, the first handwritten page of all writers has the same content, it is considered as an exception and is not used for this task. Furthermore, the medium AUT-FH database is chosen as the validation set for Farsi handwritings.

The L_f parameter for the curve fragment method is empirically chosen equal to 50 for English handwritings and 40 for Farsi handwritings. Patterns of a trained codebook using the curve fragment method for both languages are shown in Fig. 7. Other properties of these codebooks are summarized in Table 2.

The values of parameters L_f and N_f for the line fragment method are also chosen empirically. The system performances for the different values of these parameters on the IAM250 and medium AUT-FH databases are shown in Fig. 8 parts (a) and (b), respectively. Each of the graphs in these figures presents the performance versus L_f for a specific N_f value. Besides, the value of parameter *gap* for extracting codebook training codes was 20 and it was set to 2 for extracting codes from testing and training handwriting samples. Since the product $L_f \times N_f$ should not be very large, for greater values of N_f, L_f should be smaller. This can be seen in these figures: for larger N_f, better performances are achieved for shorter fragment lengths. Based on these

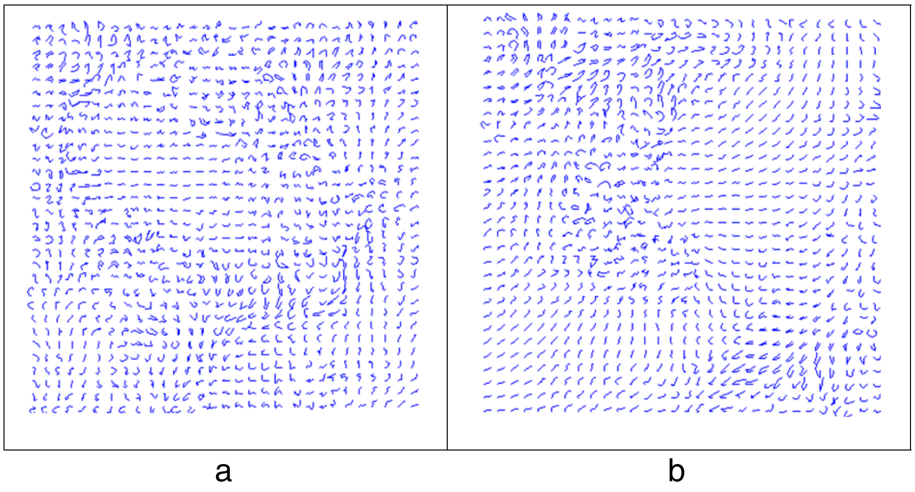

Fig. 9. Patterns of trained codebooks when line fragment method for code extraction is used: (a) trained on English handwritings with $N_f = 6$ and $L_f = 11$, and (b) trained on Farsi handwritings with $N_f = 6$ and $L_f = 5$.

diagrams, a number of fragments of 6 and a length of fragment of 11 are chosen for English handwritings. In addition, N_f and L_f for Farsi handwritings are set equal to 6 and 5, respectively. These differences between the optimal values for line fragment and curve fragment parameters show that the smaller parts of connected components have more varieties in Farsi handwritings, whereas the larger parts have more varieties in English handwritings. Patterns of the trained codebooks when the line fragment method is used for code extraction for both English and Farsi handwritings are shown in Fig. 9. The training characteristics of these codebooks are presented in Table 2. This table shows the lengths of training codes for each of the code extraction methods and the chosen parameters. The training times required for the codebooks are also shown in this table. All experiments are executed on a personal computer with a 3.0 GHz Intel Core2 Quad CPU, using one core only, and the Matlab language is used for coding. Although the number of codes extracted and used for training SOM by the line fragment method is more than the corresponding number by the other methods, the training time of SOM by these codes is lower. This is due to the shorter length of training codes of the line fragment method.

4.3. Identification results

Table 3 contains the achieved performances for English databases of Table 1 using the English codebooks of Table 2. In addition, the achieved performances for Farsi databases of Table 1 using Farsi codebooks of Table 2 are shown in Table 4. Moreover, Table 5 contains the results for the variants of AUT-FH databases and therefore, the effects of the size of the text on the performance. In these tables, the results are shown for both 2-fold and leave-on-out strategies and for the hit list sizes of 1 and 10. Besides, comparison between the average number of extracted codes from large English or large Farsi databases for different code extraction methods is presented in Table 6. This table also contains the average computation time of code extraction from the first 20 handwritings of IAM or medium AUT-FH databases, so that the speed of the methods can be compared.

The first rows of Tables 3, 4 and 5 present performances obtained by the c_1 and c_2 codebooks and curve fragment method for Farsi and English databases when the *gap* parameter is set equal to 2. The corresponding results of c_3 and c_5 codebooks for the line fragment code extraction method when the *gap* parameter is equal to 2 are presented in the second rows of Tables 3, 4 and 5. It can be seen that the performances for the line fragment method are higher than those for the curve fragment method. Actually, in the curve fragment method, the coordinates of fragments are used in the codes. This introduces the impact of details in the codes and these details vary in different samples of the codes. But, the line fragment method estimates the coordinates of points, and consequently eliminates some of these details.

The number of fragments extracted from handwritings, and as a result, the computation times of the proposed method, can be altered by changing the *gap* parameter. In fact, there is a trade-off between the performance and the computation time. For investigating the effect of increasing the *gap* parameter on the performance and computation time, results for the codebooks c_3 and c_5 with different *gap* parameters of 5 and 10 are also presented. It can be seen that both the computation times and performances for these values of *gap* parameter are decreased.

The codebook c_4 is a line fragment codebook and its only difference with c_3 is that its L_f parameter is 5 instead of 11. This codebook is constructed to show that the Firemaker database has an exceptional characteristic. The reported results for this codebook are better for the Firemaker database when the leave-one-out strategy is used and are worse in all other cases, even when the 2-fold strategy is used.

Table 3
Percent correct identification results for English databases.

Codebook	Database gap	Firemaker-lowercase 250 writers				IAM 650 writers				Large English 900 writers			
		2-Fold		Leave-one-out		2-Fold		Leave-one-out		2-Fold		Leave-one-out	
		Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10
c_1	2	93.4	97.4	86.4	96.0	93.4	97.2	92.6	96.9	92.7	97.1	89.8	96.1
c_3	2	94.7	98.0	89.2	98.6	94.8	98.0	93.7	97.7	95.0	98.1	91.8	97.4
c_4	2	94.2	97.6	91.8	98.6	93.3	97.6	92.5	97.1	93.4	97.3	91.5	96.6
c_3	5	95.2	99.2	84.6	98.0	93.0	98.1	91.3	97.5	93.0	98.0	88.4	96.8
c_3	10	91.6	99.2	76.0	93.0	87.0	97.3	84.6	95.2	86.4	96.4	79.8	93.4

Table 4
Percent correct identification results for Farsi databases.

Codebook	Database gap	AUT-FH (medium) 180 writers				FHT 260 writers				Large Farsi 480 writers			
		2-Fold		Leave-one-out		2-Fold		Leave-one-out		2-Fold		Leave-one-out	
		Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10
c_2	2	97.7	99.4	97.2	99.4	96.9	99.0	95.2	98.8	96.7	99.1	95.9	99.0
c_5	2	99.4	100	99.1	100	97.1	99.0	95.3	98.8	97.9	99.3	96.9	99.2
c_5	5	98.3	100	98.0	100	96.7	99.0	95.3	98.8	96.9	99.2	96.0	99.2
c_5	10	97.5	100	96.9	100	95.3	98.8	93.6	98.2	92.6	98.3	90.8	97.6

This happens because in the Firemaker database, one of the sets contains handwritings with the same fixed content; but the content of the other set is different. As a result, when the leave-one-strategy is used, in half of the cases, the correct matching handwriting should be found from a collection which contains many wrong manuscripts with the same content. Moreover, increasing L_f, and as a result, increasing the length of the contour fragment corresponding to the code, increases the dependency of the codebook members' histogram on the content of the handwritings. Therefore, by decreasing L_f from 11 to 5, the results improve for the mentioned case.

4.4. Comparison with other studies

In this section, a comparison between the proposed method and some other methods is presented. Since the line fragment method showed a better performance, this method will be used in the comparisons.

In the following, the line fragment method is compared to other code extraction methods and the performances are discussed. There are three existing code extraction methods for codebook-based writer identification. Schomaker et al. introduced a method that segments the contour of a connected component on its local minima [20,27]. Siddiqi et al. proposed a method that extracts codes by dividing handwriting into small windows [32]. The authors of this paper also proposed the *double fragment* code extraction method in [31]. This method extracts more codes compared to the Schomaker's method and the extracted codes do not have semantic meaning. For the purpose of comparison, a codebook is made based on this code extraction method with parameter $L_f = 30$. The results are presented in Table 7. The reported performances of the other two methods are also shown in this table. The fourth row of this table shows the performances of c_5 codebook and the line fragment code extraction method. It can be seen that the line fragment method outperforms all of the other methods. For instance, the resulted performance for the IAM database for both [20] and [32] is 80% and for double fragment method is 84.2%, which are lower than the achieved performance of 93.7% by the line fragment method.

Sometimes either there is no local minima on the lower and upper contours with the ink-line width distance, or the points with this property are far from each other. Thus, some of the fragments resulting from the Schomaker's method are complex. This fact is presented in Fig. 10, which shows the shapes of the codes resulting by this method for the word *book*. It can be seen that the second extracted code is complex and it is not proper for writer identification since it has some characters and its shape should be compared to the shapes of the codes which have exactly those characters in the same order. Furthermore, in the method introduced by Schomaker, connected components are divided, preferably, in such a way that the resulting fragments are meaningful. But, the extracted fragments for writer identification do not need to be meaningful; only their shapes are important. The numbers of codes obtained through other methods, which do not attempt to segment connected components in a meaningful way are higher, and the shapes of the fragments a person usually uses in his/her handwriting can be determined more accurately.

The length of the lower contour of the extracted fragments by the double fragment method is equal to a specific value and there will not be any large or complex extracted fragment. But, sometimes no code will contain some part of a connected components' contour and consequently, some information will be lost. This point is shown in Fig. 11. Codes extracted from the sample contour by the double fragment method contain information about a small part of the contour.

Using the line fragment and the Siddiqi's methods, small codes will certainly be extracted from all parts of connected components and they have the advantage of studying the frequent shapes which might even be parts of different characters.

The coordinates of the normalized contours or pixel values of sub-images which contain similar fragments are not exactly the same and have some differences. These differences are considered in the Schomaker's, the curve fragment, and the double fragment methods. But, the line fragment method, which estimates the shapes of fragments with some lines, eliminates some of these details and this simplification helps in grouping and finding similar shapes. Similarly, in Siddiqi's method the similarities between sub-images are not determined by comparing their pixel values and some features such as horizontal and vertical histograms and upper and lower profiles are used.

The lengths of codes in the line fragment method are much smaller than those in the previous methods. Codes of this method contain angles and lengths of N_f segments. For example, for $N_f = 6$ which was used for English handwritings, the code length is only 12. But, the codes of the Schomaker's method [27] contain pixel values of the sub-images and the length of the codes is 900. Similarly, the codes of Bulacu et al. [20] contain x and y coordinates of a part of a contour and the length of the codes with the used parameter is 200.

Table 5
Percent correct identification results for AUT-FH with different text sizes.

Codebook	Database gap	AUT-FH (short) 180 writers				AUT-FH (medium) 180 writers				AUT-FH (long) 180 writers			
		2-Fold		Leave-one-out		2-Fold		Leave-one-out		2-Fold		Leave-one-out	
		Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10
c_2	2	97.5	99.7	96.6	99.7	97.7	99.4	97.2	99.4	100	100	99.7	100
c_5	2	98.8	100	97.7	100	99.4	100	99.1	100	100	100	100	100
c_5	5	97.5	100	96.3	100	98.3	100	98.0	100	100	100	100	100
c_5	10	92.5	98.8	91.6	98.3	97.5	100	96.9	100	100	100	100	100

Table 6
Comparison between the average number of extracted codes and execution times for different code extraction methods.

Method	Gap	Language	Average number of extracted codes	Average execution time (sec)
Curve fragment	2	English	20993	79
Line fragment	2	English	20837	77
Line fragment	5	English	8345	31
Line fragment	10	English	4214	16
Curve fragment	2	Farsi	24766	95
Line fragment	2	Farsi	25094	82
Line fragment	5	Farsi	10147	35
Line fragment	10	Farsi	5166	19

Furthermore, the length of extracted feature vectors from sub-images in Siddiqi's method [32] is 54. Therefore, codebook construction and finding matching codes from codebook for the line fragment method will be faster.

The IAM database so far has been widely used in different handwriting recognition studies. Thus, it would be interesting to present a comparative overview of different methods for this database. Table 8 summarizes the writer identification performances of different studies on this database. These studies used different validation strategies including the holdout, k-fold and leave-one-out strategies. Since the validation strategy affects the performance, the validation strategy is also mentioned in the performance column. The best existing performance for the case of 650 writers is 97.1% [36] which is reported for the 4-fold cross validation strategy. Another recent good result is 97% [23]. Our method achieved a performance of 93.7% for the leave-one-out strategy, which is reasonable comparing to the state-of-the-art performances.

Comparison between the performance of the proposed method and other studies on the Firemaker lowercase database is represented in Table 9. Obviously, the presented method performed well in comparison to other methods and improved the best performance on the 250 writers from 86% to 91.8%. Besides, the reported performance in [20] on the large English database with 900 writers is 87% using the leave-one-out strategy. The proposed method has outperformed this performance and achieved an accuracy of 91.8%.

Comparison between writer identification methods on Farsi handwritings is shown in Table 10. There is not a public database used in different studies in Farsi. Therefore, the databases used in the methods reported in Table 10 are different and there is a column in this table which describes the sum of the text size that is available for training and testing samples. It can be seen that the number of writers used in the conducted experiments is more than all previous studies and the promising result of 96.9% is achieved for Farsi handwritings of 480 people. The best result in previous works was 100% accuracy [10] on the handwritings of 80 persons when handwritten samples with 35 lines were used for training and handwritten samples with 5 lines were used for testing. In that study, the total number of lines of training and testing samples was 40. Accuracy of 100% is also achieved by the line fragment method for the handwritings of 180 people when the total number of lines of training and testing samples is 24. The results of a previous study by the authors on the three variants of AUT-FH database are also presented in this table. The performances reported in this study outperformed those results. For instance, the performance on the short AUT-FH database is improved from 91.6% to 97.7%.

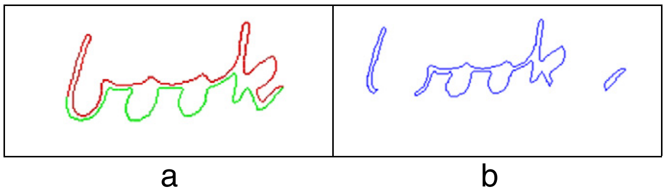
Fig. 10. Some of the extracted codes by Shomaker's method are complex: (a) contour of the word *book*, (b) extracted codes of (a) by Schomaker's method.

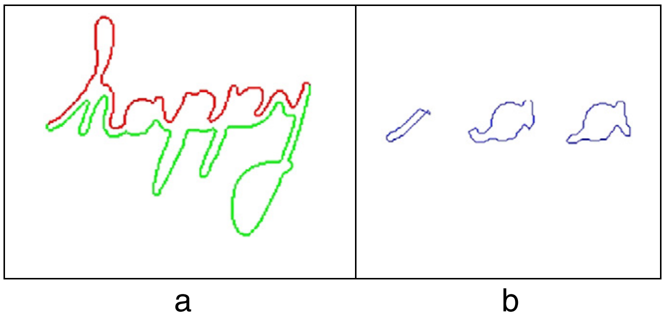
Fig. 11. Sometimes none of the extracted code by the double fragment method contains some part of a connected components' contour (a) contour of the word *happy*, (b) shapes of extracted codes by the double fragment method when $L_f = 40$ and $gap = 5$.

5. Conclusions

This paper studied the offline text-independent writer identification problem using the codebook approach. The curve fragment and line fragment code extraction methods were introduced and their performances were investigated. Results obtained through these methods were better than the performances of the existing code extraction techniques. These methods extract the code from small parts of connected components and they have the advantage of using the frequently appearing shapes which might be parts of different characters. Besides, these methods extract many codes from handwritings resulting in more robust feature vectors. The line fragment method showed the best performances. It approximates contour fragments using the segments'

Table 7
Comparison of different code extraction methods.

Database method	Firemaker-lowercase 250 writers				IAM 650 writers				Large English 900 writers			
	2-Fold		Leave-one-out		2-Fold		Leave-one-out		2-Fold		Leave-one-out	
	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10	Top 1	Top 10
Schomaker et al. [20]	–	–	75	92	–	–	80	94	–	–	76	92
Siddiqi et al. [32]	–	–	–	–	84	96	80	–	–	–	–	–
Double fragment [31]	90.8	97.6	80.6	94.4	86.3	96.0	84.2	94.4	85.8	95.9	81.5	93.8
Line fragment (c_5 codebook)	94.7	98.0	89.2	98.6	94.8	98.0	93.7	97.7	95.0	98.1	91.8	97.4

Table 8
Comparison of writer identification studies on the IAM database.

Study	Number of writers	Sum of training and testing samples	Performance (top 1)
Schlapbach and Bunke (2004) [12]	100	5	96.56% (4-fold)
Bensefia et al. (2005) [25]	150	2	86% (holdout)
Bensefia et al. (2005) [24]	150	2	86% (holdout)
Schlapbach and Bunke (2006) [14]	100	5	98.46% (4-fold)
Schlapbach and Bunke (2007) [11]	100	5	97.03% (4-fold)
Bulacu and Schomaker (2007) [20]	650	2	89% (leave-one-out)
Siddiqi and Vincent (2009) [18]	650	2	86% (leave-on-out)
Siddiqi and Vincent (2009) [19]	650	2	86.5% (leave-one-out) 89% (holdout)
Siddiqi and Vincent (2010) [32]	650	2	89% (leave-one-out) 91% (2-fold)
Kirli and Gulmezoglu (2011) [36]	650		97.1% (4-fold)
Brink and Smit (2011) [23]	657	2	97%
The proposed method	650	2	93.7% (leave-one-out) 94.8%(2-fold)

angles and lengths and eliminates some details. This simplification helps in grouping and finding similar shapes. In addition, the code length in this method is shorter than other methods, and therefore, its codebook construction is faster.

Two English databases, namely, the IAM and Firemaker and three Farsi databases were used for testing the methods. One of the Farsi databases had three varieties of short, medium and long texts. As a result, the effect of text size on the methods' performances was also investigated. The achieved performance of the proposed method on the IAM database was 93.7% when the leave-one-out strategy was used. In addition, the performance of 91.8% on the handwritings of 900 writers was achieved. For the Farsi language, the correct identification rates of 97.7%, 99.1% and 100% were reached for samples with 4, 8 and 12 lines, respectively. In addition, the performance of the method on a Farsi database with 480 writers was 96.9%.

The performance of the line fragment method is promising. But, computation times of the feature vector and codebook construction in this method are large. These computation times may be decreased by using a hierarchical clustering method. This can be investigated in the future studies. In addition, the optimal values of the parameters of the introduced code extraction methods are different for different languages. Here, the optimal values have been determined empirically. However, it is desired to have a method for the automatic determination of the optimal values of the parameters. This can be considered as a problem for future work.

Table 9
Comparison between writer identification methods on the Firemaker lowercase database.

Study	Number of Writers	Performance (top 1)
Bulacu and Schomaker (2003) [21]	250	80% (leave-one-out)
Bulacu and Schomaker (2007) [20]	250	83% (leave-one-out)
Brink and Smit (2011) [23]	250	86% (leave-one-out)
The proposed method	250	91.8% (leave-one-out) 94.2% (2-fold)

Table 10
Comparison between writer identification methods on Farsi handwritings.

Study	Number of writers	Sum of training and testing sample size	Performance (top 1)
Shahabinejad and Rahmati (2007) [8]	40	3 pages (3 × 7 lines)	82.5% (holdout)
Helli and Moghaddam (2008) [9]	100	5 A5 pages (5 × 8 lines)	95% (holdout)
Sadeghi and Moghaddam (2009) [16]	50	5 A5 pages (5 × 8 lines)	96% (holdout)
Helli and Moghaddam (2010) [10]	80	5 A5 pages (5 × 8 lines)	100% (holdout)
Ghiasi and Safabakhsh (2010) [31]	180	1 page (8 lines)	91.6% (leave-one-out) 92.7% (2-fold)
Ghiasi and Safabakhsh (2010) [31]	180	2 pages (2 × 8 lines)	98.8% (leave-one-out) 99.4% (2-fold)
Ghiasi and Safabakhsh (2010) [31]	180	3 pages (3 × 8 lines)	99.7% (leave-one-out) 100% (2-fold)
The proposed method	180	1 page (8 lines)	97.7% (leave-one-out) 98.8% (2-fold)
The proposed method	180	2 pages (2 × 8 lines)	99.1% (leave-one-out) 99.4% (2-fold)
The proposed method	180	3 pages (3 × 8 lines)	100% (leave-one-out) 100% (2-fold)
The proposed method	480	1 page (4 to 10 lines)	96.9% (leave-one-out) 97.9% (2-fold)

References

[1] H. Feng, C. Wah, Private key generation from on-line handwritten signatures, Information Management and Computer Security, 2002, pp. 159–164.
[2] L. Ballard, D. Lopresti, F. Monrose, Evaluating the security of handwriting biometricsth, 10th International Workshop on Frontier Handwriting Recognition, 2006, pp. 461–466.
[3] M. Tapiador, J. Gómez, J.A. Sigüenza, Writer identification forensic system based on support vector machines with connected components, Innov. Appl. Artif. Intell. 3029 (2004) 625–632.
[4] A. Fornes, J. Llados, G. Sanchez, H. Bunke, Writer identification in old handwritten music scores, 8th IAPR Workshop on Document Analysis Systems, 2008, pp. 347–353.
[5] A. Fornés, J. Lladós, G. Sánchez, H. Bunke, On the use of textural features for writer identification in old handwritten music scores, 10th International Conference on Document Analysis and Recognition (ICDAR'10), 2009, pp. 996–1000.
[6] J. Sas, Handwriting recognition accuracy improvement by author identification, Int. Conf. on Artificial Intelligence and Soft Computing, 2006, pp. 682–691.
[7] H.E.S. Said, T.N. Tan, K.D. Baker, Personal identification based on handwriting, Pattern Recognit. 33 (2000) 149–160.
[8] F. Shahabinejad, M. Rahmati, A new method for writer identification and verification based on Farsi/Arabic handwritten texts, Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), 2007, pp. 829–833.
[9] B. Helli, M.E. Moghaddam, A text-Independent Persian writer identification system using LCS based classifier, IEEE Int'l Symposium on Signal Processing and Information Technology 2008 (ISSPIT'08), 2008, pp. 203–206.
[10] B. Helli, M.E. Moghaddam, A text-independent Persian writer identification based on feature relation graph (FRG), Pattern Recognit. 43 (2010) 2199–2209.
[11] A. Schlapbach, H. Bunke, A writer identification and verification system using HMM based recognizers, Pattern Anal. Appl. 10 (2007) 33–43.
[12] A. Schlapbach, H. Bunke, Using HMM based recognizers for writer identification and verification, 9th Int. Workshop on Frontiers in Handwriting Recognition, 2004, pp. 167–172.
[13] U.-V. Marti, H. Bunke, The IAM-database: an English sentence database for offline handwriting recognition, Int. J. Doc. Anal. Recognit. 5 (2002) 39–46.
[14] A. Schlapbach, H. Bunke, Off-line writer identification using gaussian mixture models, In Proc. the 18th International Conference on, Pattern Recognition, 2006, pp. 992–995.
[15] U. Marti, R. Messerli, H. Bunke, Writer identification using text line based features, Proc. of 6th Int Conference on Document Analysis and Recognition, 2001, pp. 765–768.
[16] S. Sadeghi, M.E. Moghaddam, Text-independent Persian writer identification using fuzzy clustering approach, International Conference on Information Management and, Engineering, 2009, pp. 728–731.
[17] I. Siddiqi, N. Vincent, Combining global and local features for writer identification, 11th Int Conference on Frontiers in Handwriting Recognition (ICFHR), 2008, pp. 19–21.

[18] I. Siddiqi, N. Vincent, A set of chain code based features for writer recognition, 10th International Conference on Document Analysis and Recognition, 2009, pp. 981–985.
[19] I. Siddiqi, N. Vincent, Combining contour based orientation and curvature features for writer recognition, Proceedings of the 13th International Conference on Computer Analysis of Images and Patterns, 2009, pp. 245–252.
[20] M. Bulacu, L. Schomaker, Text-independent writer identification and verification using textural and allographic features, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2007) 701–717.
[21] M. Bulacu, L. Schomaker, Writer style from oriented edge fragments, In Proc. of CAIP 2003, 2003, pp. 460–469.
[22] L. Schomaker, L. Vuurpijl, Forensic Writer Identification: a Benchmark Data Set and a Comparison of Two Systems, Available: http://www.academia.edu/452837/Forensic_Writer_Identification_A_Benchmark_Data_Set_and_a_Comparison_of_Two_Systems, 2000.
[23] A.A. Brink, J. Smit, M. Bulacu, L.R.B. Schomake, Writer identification using directional ink-trace width measurements, Pattern Recognit. 45 (2011) 162–171.
[24] A. Bensefia, T. Paquet, L. Heutte, Handwritten document analysis for automatic writer recognition, Electron. Lett. Comput. Vis. Image Anal. 5 (2005) 72–86.
[25] A. Bensefia, T. Paquet, L. Heutte, A writer identification and verification system, Pattern Recognit. Lett. 26 (2005) 2080–2092.
[26] L. Schomaker, M. Bulacu, Automatic writer identification using connected-component contours and edge-based features of uppercase Western script, IEEE Trans. Pattern Anal. Mach. Intell. 26 (2004) 787–798.
[27] L. Schomaker, K. Franke, M. Bulacu, Using codebooks of fragmented connected-component contours in forensic and historic writer identification, Pattern Recognit. Lett. 28 (2007) 719–727.
[28] L. Schomaker, M. Bulacu, K. Franke, Automatic writer identification using fragmented connected-component contours, Proc. of 9th IWFHR, 2004, pp. 185–190.
[29] M. Bulacu, L. Schomaker, A. Brink, Text-independent writer identification and verification on offline Arabic handwriting, Proc. of 9th Int. Conf. on Document Analysis and Recognition, 2007, pp. 769–773.
[30] A. El-Yacoubi, M. Gilloux, R. Sabourin, C.Y. Suen, An HMM-based approach for off-line unconstrained handwritten word modeling and recognition, IEEE Trans. Pattern Anal. Mach. Intell. 21 (1999) 752–760.
[31] G. Ghiasi, R. Safabakhsh, An efficient method for offline text independent writer identification, 2010 International Conference on Pattern Recognition, 2010, pp. 1245–1248.
[32] I. Siddiqi, N. Vincent, Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features, Pattern Recognit. 43 (2010) 3853–3865.
[33] M. Bulacu, L. Schomaker, A comparison of clustering methods for writer identification and verification, Eighth International Conference on Document Analysis and Recognition (ICDAR'05), 2005, pp. 1275–1279.
[34] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, S. Janet, UNIPEN project of on-line data exchange and recognizer benchmarks, Proc. Int. Conf. Pattern Recognition, 1994, pp. 29–33.
[35] M. Ziaratban, K. Faez, F. Bagheri, FHT: an unconstraint Farsi handwritten text database, 10th International Conference on Document Analysis and Recognition, 2009, pp. 4–8.
[36] O. Kirli, M.B. Gulmezoglu, Writer identification from handwriting text lines, 2011 International Symposium on Innovations in Intelligent Systems and Applications, 2011, pp. 133–137.