IJDAR (2009) 12:21–32
DOI 10.1007/s10032-009-0079-7

ORIGINAL PAPER

A blackboard approach towards integrated Farsi OCR system

Hossein Khosravi · Ehsanollah Kabir

Received: 11 July 2008 / Revised: 20 November 2008 / Accepted: 5 January 2009 / Published online: 12 February 2009

Abstract An integrated OCR system for Farsi text is proposed. The system uses information from several knowledge sources (KSs) and manages them in a blackboard approach. Some KSs like classifiers are acquired a priori through an offline training process while others like statistical features are extracted online while recognizing. An arbiter controls the interactions between the solution blackboard and KSs. The system has been tested on 20 real-life scanned documents with ten popular Farsi fonts and a recognition rate of 97.05% in word level and 99.03% in character level has been achieved.

Keywords Farsi · Persian · OCR · Blackboard approach · Segmentation and recognition

1 Introduction

Nowadays with advancement in character recognition technology, OCR systems are well known for lots of peoples. There are several commercial OCR products available for popular languages in the world. Today anyone who buys a scanner expects to find an OCR software within the scanner package as well. At the same time users of some languages suffer from lack of such OCR product for their native language.

Farsi or Persian is the official language of more than 150 million peoples of the world. Like Arabic, Farsi is a right to left script, but there are some differences like number of alphabets, font styles, vocabulary and signs, which make Farsi OCR somehow different from Arabic. In last decades several researchers worked on Arabic OCR [1–3,8,14,22], and as a result today there exist some commercial OCR products for Arabic language. But in the field of Farsi language, lots of OCR papers are about isolated character/digit recognition [10,16,19,23] and there are only a few papers in the field of printed text recognition [4,6,17,18,24]. Here we review some important works reported on Farsi OCR.

It seems that the first paper about Farsi printed text recognition is [21]. In this paper, an OCR system for Farsi text with large font size like newspaper headlines is proposed. Amzi and Kabir [4] proposed a new segmentation algorithm based on the conditional labeling of the upper contour. They also proposed a technique to adjust the local base line for each subword. Menhaj and Adab [18] proposed a segmentation and recognition method for printed text recognition. They used Fourier descriptors as features and MLP as classifier. In [6], an OCR system based on subword shape recognition is proposed. In this approach each subword is treated as a shape and recognized as a whole without segmentation. They used characteristic loci as shape descriptor and K-means for clustering subwords of four fonts in three sizes. PCA has been applied to reduce feature length. They tested their work on two sets of images; one set of five good quality images generated in computer, printed and then scanned, and one set of four real documents. They achieved the recognition rates of 96 and 90% on first and second set respectively.

At this time there are only two OCR products that support Farsi language, Automatic Reader from Sakhr[1] and Readiris from Iris.[2] The accuracy of these products for Farsi language is very low, something between 60 and 70%, so that no one can rely on them to read his Farsi documents. This is due to

H. Khosravi (✉) · E. Kabir
Department of Electrical Engineering,
Tarbiat Modarres University, Tehran, Iran
e-mail: HosseinKhosravi@modares.ac.ir

[1] http://www.sakhr.com.

[2] http://www.Irislink.com.

the fact that both products are basically developed for other languages.

To have a Farsi OCR system with high accuracy we should consider all aspects of Farsi language. There are several knowledge sources (KSs) related to Farsi scripts, which can help us to develop more reliable and more accurate OCR system.

In this paper, we propose an integrated Farsi OCR system, which uses the information of several KSs in blackboard architecture. Some KSs like classifiers and vocabulary are stable and will not change during the recognition process, but some others like signs and dots information are transient and will change while recognizing a text. We used blackboard approach because it is simple to be modified and we are able to add new KSs without changing the code significantly.

The rest of the paper is organized as follow. In Sect. 2 Farsi script specifications from OCR point of view will be described. Section 3 describes the integrated OCR system which is the main part of our work. In Sect. 4 the process is demonstrated through an example. Section 5 includes experimental results and the conclusion follows in Sect. 6.

2 Farsi script specifications

Farsi, like Arabic, is a right to left script and has its own specifications. Here we describe the most important features of Farsi script from OCR point of view.

- Letters are written from right to left, but numerals are written from left to right. We should consider this attribute for regenerating recognized text.
- Some letters take different shapes according to their position in the word. For example letter Ein have an isolated form "ع", and three joint forms: beginning form "عـ", middle form "ـعـ" and final form "ـع". We divided all letters into two groups; isolated and joint form and trained separate classifiers for them (see Sects. 3–5 and 3–6).
- Each word is composed of one or more character(s) and/or subword(s). A subword is a combination of joint letters. For example the word Farsi "فارسی" is composed of one letter, "ر", and two subwords, "فا" and "سی", each of them having two letters. As described in Sects. 3–5 and 3–6 we recognize subwords through segmentation and recognition but isolated characters are recognized without segmentation.
- Some letters only differ in number or position of their dots, e.g. letters Je "ج", He "ح", Khe "خ" and Che "چ" have the same shapes, but different dots. Considering this attribute, we firstly recognize bodies then add dot information for each character.
- Some digits are very similar to some letters: digit 1 "1" and character Alef "ا", digit 0 "0" and dot ".", digit 5 "۵" and character Ha "ه". We resolve this ambiguity with context information through post processing.
- Each letter can take some signs like Tashdid "ـّ", Hat as in "آ", Sarkesh as in "گ" and other Arabic signs which are used for unfamiliar words like proper nouns, e.g. vowels "ـَـُـِ" and Tanvins "ـًـٌـٍ". As described in Sect. 3–6 we firstly remove all dots and signs and construct body (Fig. 5), then recognize signs using special engine and finally during segmentation and recognition of body, add signs info to the characters.
- In some fonts, digits 4, 5 and 6 and letters Kaf and Ye, have two shapes, Arabic and Farsi, which both of them may be used, e.g. "4" and "٤" for digit 4 and "ک" and "ك" for letter Kaf. We resolved this ambiguity using samples from both shapes while training the classifiers.

3 System architecture and knowledge sources

To have a reliable and efficient OCR system we should consider several KSs involving in Farsi script. We saw Farsi OCR as a problem which will be solved when the text is recognized satisfactorily. We had several KSs from line detector to character classifier that each of them provides useful information which can help us to solve the problem to some extent. A good platform for this type of problems is blackboard architecture. Blackboard is a shared memory where the problem is defined on it and every KS can share its information to find the solution. An arbiter controls the interaction between KSs and the blackboard.

In our OCR system we tried to include almost all possible KSs. Also some tools are developed where some of them are required to prepare KSs and others perform specific actions like removing dots from subword. The general diagram of our blackboard system is shown in Fig. 1. Here we describe KSs in detail.

3.1 KS 1: statistical features

Statistical features consist of five elements: *noise width*, *line height*, *base-line position*, *pen width* and *space width*.

Noise width is computed from projection analysis of text block as follows:

$$\mathrm{nw} = \frac{1}{n}\sum_{y}\{h[y]\,|h[y] < \mathrm{nw}_0\} \qquad \text{where}$$

$$h[y] = \sum_{x} f(x,y)$$

$$f(x,y) = \text{pixel value} \begin{cases} 1 & \text{for black pixels (foreground)} \\ 0 & \text{for white pixels (background)} \end{cases}$$

$$\mathrm{nw}_0 = \min\left[\frac{\max[h(y)]}{5}, \min[h(y)] + 15\right]$$

$$n = \#\{h[y]\,|h[y] < \mathrm{nw}_0\} \tag{1}$$

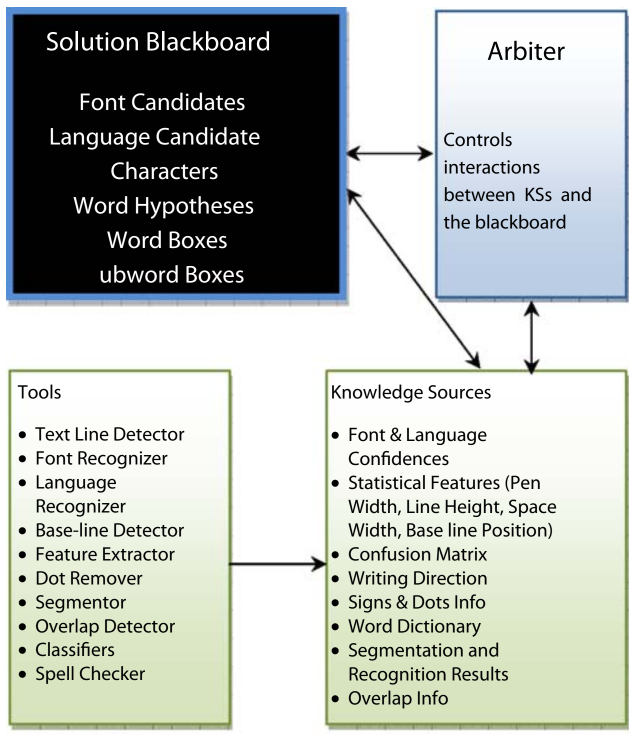


Fig. 1 Blackboard architecture for Farsi OCR

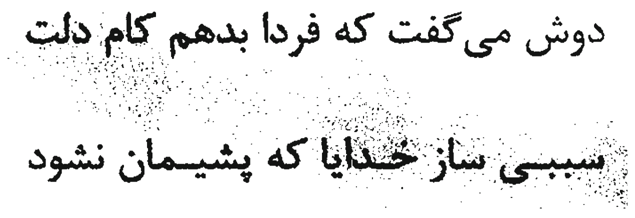

دوش می‌گفت که فردا بدهم کام دلت
سببی ساز خدایا که پشیمان نشود

Fig. 2 Sample noisy image with noise width of about ten

This value is used in line detector to avoid detection of noise pixels as lines (Fig. 2) and to prevent two successive lines to be merged incorrectly.

Line height is computed from average height of text lines detected by line detector as follows:

$$H = \frac{1}{n} \sum_{y} \{h[y]\,|h[y] > \mathrm{nw}_0\} \tag{2}$$

It helps the system to find merged or split lines and correct them.

Base line position is computed through horizontal projection of each line as follows (Fig. 3):

$$B = \arg\max_{y} \{h(y)\} \tag{3}$$

Base line is useful for segmentation module, because almost all segmentation points in Farsi script are located on base line. If a line is long, two base lines will be computed, one for the left part of the line and another for the right part.

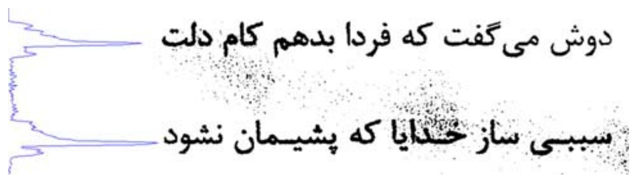


Fig. 3 Base line detection

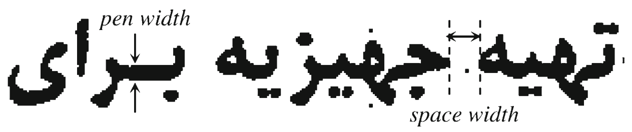


Fig. 4 Pen width and Space width for a typical line

This is to overcome small skew angles which are not removed in preprocessing.

Pen width is computed from mode of stroke width of the line (Fig. 4).

$$\mathrm{pw} = \mathrm{mod}(sw(x, y)) \qquad \text{where:}$$
$$\mathrm{sw}(x, y) = \begin{cases} \mathrm{cp}(x, y) & \text{if } \mathrm{cp}(x, y+1) = 0 \\ 0 & \text{otherwise} \end{cases}$$
$$\mathrm{cp}(x, y) = \mathrm{cp}(x, y-1) + f(x, y)$$
$$f(x, y) \text{ is the pixel value at (x,y), 0 for white and 1 for black.} \tag{4}$$

This is used to overcome image resolution dependencies; if *pen width* is too small, current text line is resampled to yield a typical pen width associated with a common font size scanned at 300 dpi. This is important for segmentation module where some dpi dependent thresholds are used.

The last parameter, *space width*, is the width between two successive words. It is computed as the mode of spaces in each line. If space between two successive words is less than this parameter, they are subwords of the same word otherwise they are two words (Fig. 4).

3.2 Classifiers and features

Before going into other KSs we prefer to have a brief description of classifiers and features used for training.

In our OCR system we have two classifiers for font and language recognition and three classifiers per each font for sign recognition, isolated character recognition and joint character recognition. All classifiers are boosted MLPs[3] which are trained based on AdaBoost.M2 algorithm [7]. MLP is selected because of its speed advantage over other classifiers like SVM, RBF or HMM.

To recognize font and language, we perform some preprocessing and extract special features which may be discussed in another paper [12], but for other classifiers we

[3] Multi Layer Perceptron.

implemented several statistical features and after some experiments considering speed and accuracy we selected a combination of them for each classifier, e.g. Characteristic loci (81 features), Size (3 features) and Gradient histogram (144 features) [11] is used for isolated character classifier.

3.3 KS2, KS3: font and language confidences

In our OCR system a font recognition module [12] recognizes the font type of individual lines. For each font, special character recognition engines have been trained; so every failure in font recognition may decrease overall performance. Since font recognition accuracy is not perfect we keep confidences of three most probable fonts as KS to be considered later. At first the system recognizes the text line assuming the most probable font, but if it is recognized with low confidence, its font will be changed to the next candidate and it will be recognized again. This process may be continued up to three times. The arbiter controls this process and decides through character confidences that a line is recognized well or not. Confidences are values between 0 and 100 which determine how good an action is performed, e.g. if a character is recognized with confidence of 99% the result can be approved without any doubt, but when its confidence is 40% it is not so satisfactory. Usually, confidences come from classifiers outputs and may be modified through the recognition process. For example if a character is recognized as Ghaf "ق" but its number of dots is different from two, its confidence will be decreased appropriately.

Another KS is language confidence. If a document has English characters between Farsi scripts, a language recognizer module will detect English words from Farsi words. Here we are not going to describe the structure of this module but like font recognition module, the result of language recognizer is not perfect, so we do not rely on it. At first we recognize the word with the proposed language from language recognizer; if it is recognized with acceptable confidence, the detected language will be approved, otherwise the word will be recognized with the engines of the other language and the result with the higher confidence will be accepted. It must be mentioned that English words are recognized using an existing OCR engine.

3.4 KS4 sign classifier

As described in Sect. 2, dots and signs have important role in Farsi letters, so that some letters only differ in number or position of dots, e.g. letters Je "ج", He "ح", Khe "خ" and Che "چ" have the same shapes, but different dot attributes. Considering this characteristic, we designed a sequential recognition strategy in which at first, character/subword body (Fig. 5) is recognized and then dots and signs info is considered to find final character.

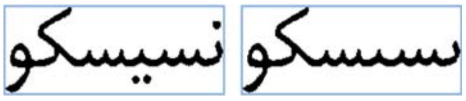


Fig. 5 Sample subword and its body

Table 1 Signs and dots classes

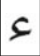
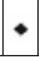
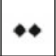
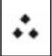
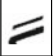
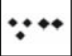
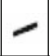
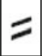
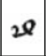
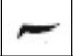
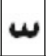
0	1	2	3	4	5	6	7	8	9	10	11
											invalid

Table 2 Isolated characters used in isolated character classifier

0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	۵	۶	7	8	9
10	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
ا آ أ إ	ب پ ت ث	ج چ ح خ	د ذ	ر ز ژ	س ش	ص ض	ط ظ	ع غ	ف
20	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**
ق	ک گ	ل	م	ن	و ؤ	ه ۀ ة	ى ي ئ	ء	.
30	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**
-	” ’ ،	)	(	!	؟	' "	“ ‘	»	«
40	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**
]	[	}	{	%	*	+	لا	ﷲ	SW

Table 1 shows all dots and signs used in Farsi scripts. Two extra classes are used, 5 and 11. Class 11 represents invalid signs or noise and Class 5 was included because sometimes Classes 2 and 3 may be merged due to noise or bad binarization. For each font, a classifier, called Sign Classifier, is trained for these symbols.

We divided all isolated characters according to their bodies into 50 classes (Table 2), some classes like 19 only include one character, but some others like ten include four characters.

3.5 KS5: Isolated character classifier

Inspecting Farsi letters, we found that some letters like "ب" are very probable to be over segmented, if we segment them like other subwords. On the other hand most of the letters take different shapes when joining to other letters to create a subword. Considering this, we designed Isolated Character Classifier which is responsible for isolated characters. This classifier takes the body of the input subwords (Fig. 5) and decides whether the input is a character or a subword. If it is a character, its code will be retrieved (Codes 0–48 in Table 2) and the final character will be found considering its dots; otherwise (Code 49) it will go through segmentation and recognition module (Sect. 3–6). Table 2 shows all classes of isolated characters. As it shows, two subwords لا and ﷲ

(classes 47 and 48) are treated as isolated characters because they have special shapes which cannot be segmented easily. Class 49 is an extra class developed for subwords; if the classifier activates this class, it means that the input is a subword not an isolated character, so it must go through segmentation and recognition module (Sect. 3–6).

It is important that sometimes this classifier may be uncertain, i.e. its output confidence for the winner class is low. In these cases the system does not rely on its result and sends the subword into segmentation module as well and decides later; if the overall confidence of the segmented subword is better than the confidence of the isolated character, it will be assumed a subword and the result of segmentation will be accepted otherwise the isolated character will be remained.

3.6 KS6: segmentation and recognition

3.6.1 Segmentation

An important part of our system is the segmentation module. Since a subword is a combination of joint letters, segmentation is required to break each subword to its composing components. There are several segmentation algorithms for Farsi/Arabic scripts, which can be divided into four groups: methods based on vertical projection analysis [1,20], contour-based methods [4,9], methods based on profile analysis [13] and recognition based segmentations [5].

Every failure in segmentation module may be unrecoverable on the other hand there is almost no segmentation algorithm which provides segmentation points perfectly. Segmentation and recognition is one of the best approaches which its segmentation errors can be recovered through recognition. In this method we can recover segmentation errors while recognizing each token of the segmented subword. It is important that a primary segmentation algorithm must be designed so that under-segmentation does not occur, but over segmentation usually will be recovered during recognition.

We designed a profile-based segmentation for the initial step (Fig. 6). This algorithm is more reliable from projection analysis methods and is faster than contour based methods. Segmentation process is as follows:

1. Crop subword body (Fig. 5) around the baseline position with height of 3 * penwidth (Fig. 7). Baseline position and pen width were provided previously by KS1. Segmentation points of Farsi subwords are always located on base line, so we crop subwords around baseline to prevent from invalid segmentation points.

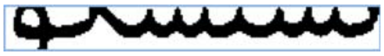


Fig. 6 Search area for segmentation points

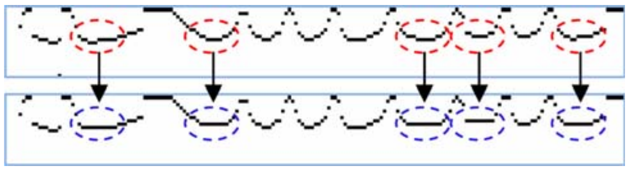


Fig. 7 *Top* modified top profile. *Bottom* Enhanced profile

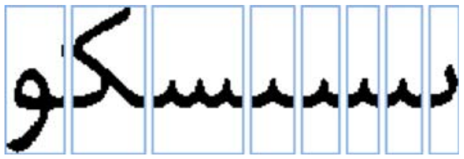


Fig. 8 Potential segmentation points

2. Find top-profile of the cropped subword. Usually this profile is computed from the first cross of white to black pixels viewing from top. But if we use this profile, some portion of base line will become invisible behind some characters like He "ـهـ" or Kaf "ـکـ". So we changed the profile to the last cross of white to black to solve this problem (Fig. 7).
3. Smooth the top profile for better decision of segmentation points; search for horizontal strokes which are broken in the middle so that a level with one-pixel deeper height is created in between. Change the middle level value to the side levels value to have a better profile for segmentation (Fig. 7).
4. Scan smoothed profile and assign middle point of horizontal strokes with length of three or more pixels as segmentation points (Fig. 8). These points are treated as potential segmentation points and sometimes, as in the case of Fig. 8, may be over segmented. This issue will be solved during recognition phase.

3.6.2 Recognition

Recognition is the second phase of the segmentation and recognition algorithm. Here we provided another classifier which is responsible for recognition of joint strokes. Joint character classifier is used to recognize subword tokens provided from segmentation process. This classifier includes 39 classes of joint characters in different positions and an extra class that includes all stroke combinations which do not compose a character (Table 3). This classifier helps us to recognize the subword and decide what segmentation point is valid.

Having segmentation points provided, we can simply recognize each stroke, store the results and move on. But in this way every failure in segmentation phase will be unrecoverable. The better way is to scan the segmented subword from right or left and to recognize strokes sequentially; if a

Table 3 Joint characters used in join character classifier

0	1	2	3	4	5	6	7	8	9
ا أ آ	بـ پـ يـ تـ ثـ	ـبـ ـپـ ـيـ ـتـ ـثـ	ـب ـپ ـت ـث	جـ چـ حـ خـ	ـجـ ـچـ ـحـ ـخـ	ج چ ح خ	ـد ـذ	ـر ـز ـژ	سـ شـ
10	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
ـسـ ـشـ	ـس ـش	صـ ضـ	ـصـ ـضـ	ـص ـض	ط ـط ظ ـظ	عـ غـ	ـعـ ـغـ	ـع ـغ	فـ قـ
20	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**
ـفـ ـقـ	ـف	ـق	کـ گـ	ـکـ ـگـ	ـک ـگ	لـ	ـلـ	ـل	مـ
30	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**
ـمـ	ـم	ـن	ـو	هـ	ـهـ	ـه	ي	ـلا	no char

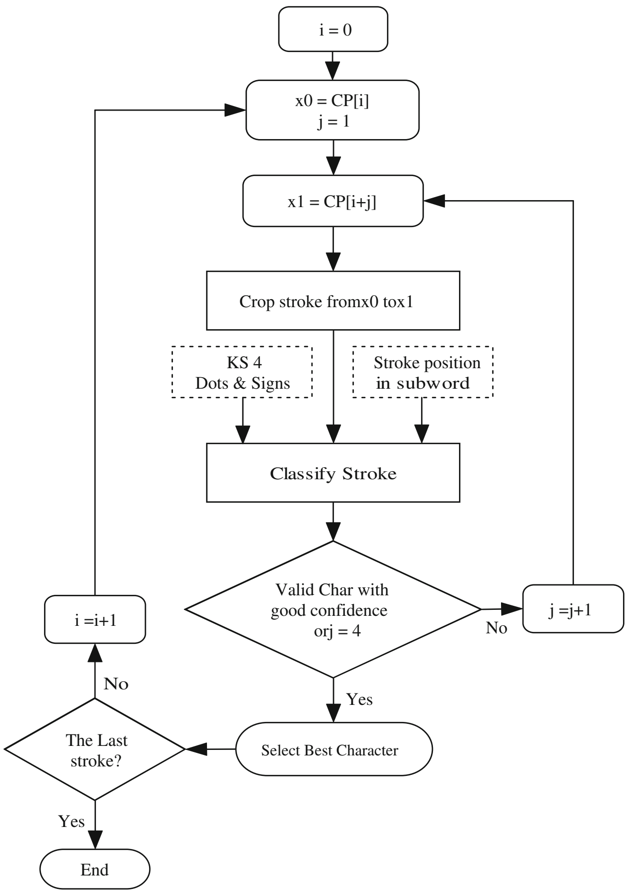


Fig. 9 Recognition algorithm while scanning from left

stroke was recognized as a valid character with good confidence, it will be stored; otherwise two successive strokes will be merged and recognized again. This process may be continued up to four times or while a valid character found. Although this method of scanning is better than recognizing all strokes separately, but if a non-character stroke

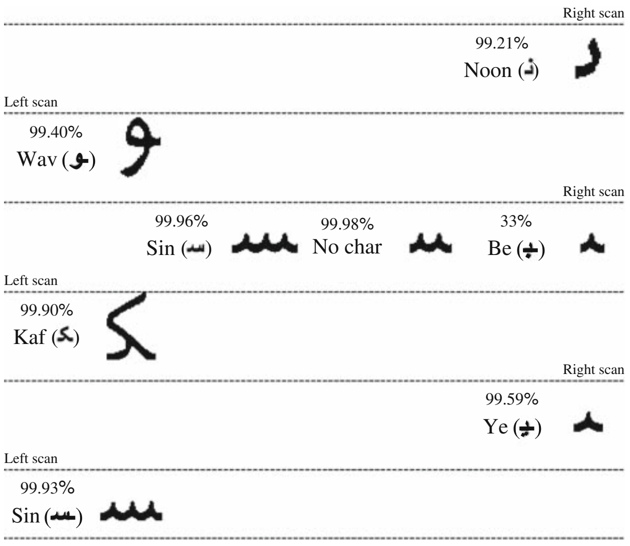


Fig. 10 Recognition of tokens for subword of Fig. 5

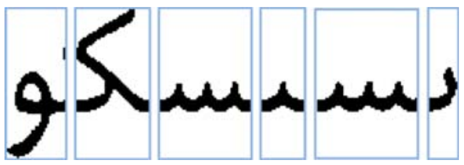
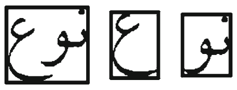


Fig. 11 Correct segmentation points of Fig. 4

Fig. 12 An example of vertical overlap



recognized as a character incorrectly, the error will be distributed from now on. Considering this problem, we chose to scan the segmented subword from both left and right simultaneously. In this way, we scan the subword from right until finding a valid character with good confidence, then switch to the left side and recognize one character from left, this process continues until all tokens scanned. Fig. 9 shows the flowchart of the recognition phase while scanning from left side. After this process we will have the correct segmentation points.

Figure 10 shows an example of segmentation and recognition algorithm applied to subword of Fig. 5. As this figure shows, except the character Sin, third row, which is recognized after three epochs, all other characters which are correctly segmented, are recognize in single epochs. Figure 11 shows the final segmentation points.

3.7 KS7: writing direction

As mentioned in Sect. 2, Farsi letters are written from right to left, but numeral strings are written from left to right. We must consider this rule during text regeneration. We should read

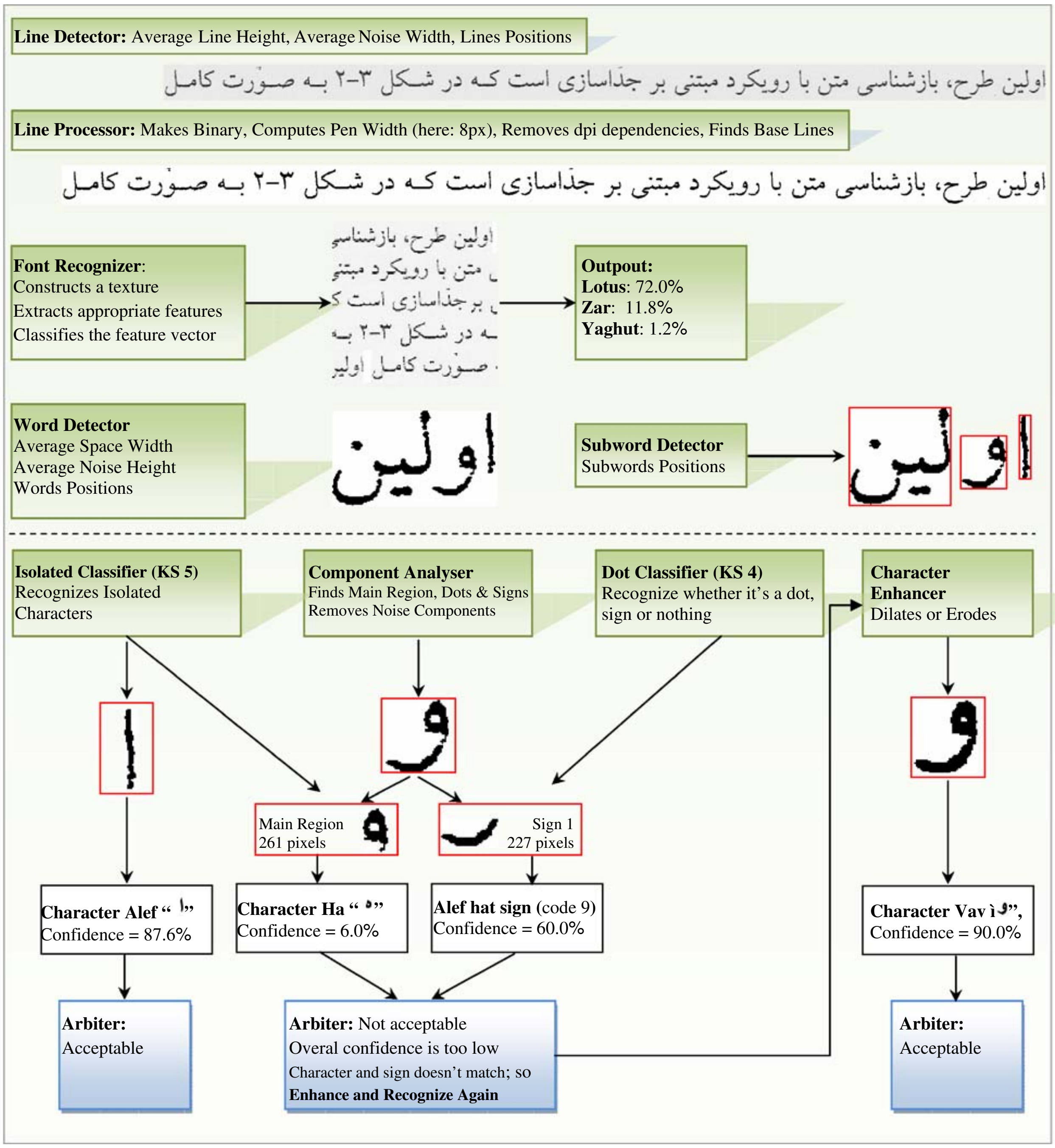

Line Detector: Average Line Height, Average Noise Width, Lines Positions
اولین طرح، بازشناسی متن با رویکرد مبتنی بر جداسازی است که در شکل ۳-۲ به صورت کامل
Line Processor: Makes Binary, Computes Pen Width (here: 8px), Removes dpi dependencies, Finds Base Lines
Font Recognizer: Constructs a texture Extracts appropriate features Classifies the feature vector
Outpout: Lotus: 72.0% Zar: 11.8% Yaghut: 1.2%
Word Detector Average Space Width Average Noise Height Words Positions
اولین
Subword Detector Subwords Positions
Isolated Classifier (KS 5) Recognizes Isolated Characters
Component Analyser Finds Main Region, Dots & Signs Removes Noise Components
Dot Classifier (KS 4) Recognize whether it's a dot, sign or nothing
Character Enhancer Dilates or Erodes
Main Region 261 pixels
Sign 1 227 pixels
Character Alef "ا" Confidence = 87.6%
Character Ha "ه" Confidence = 6.0%
Alef hat sign (code 9) Confidence = 60.0%
Character Vav "و", Confidence = 90.0%
Arbiter: Acceptable
Arbiter: Not acceptable Overal confidence is too low Character and sign doesn't match; so Enhance and Recognize Again
Arbiter: Acceptable

Fig. 13 An example showing KS interactions with the blackboard and arbiter decisions

subwords from right to left, but when encountering numeric subwords or English words, we must buffer their result until the next Farsi subword and then reverse their order to have the correct order in output text.

3.8 KS8, KS9: vocabulary and confusion matrix

Sometimes a word may be recognized with low confidence. If this word exists in the vocabulary it will not be changed because the low confidence may be due to noise, or low quality of the document. But if it does not exist in the vocabulary, it may be a segmentation failure, misclassification of characters or loosing dots or signs. In this case we will use the vocabulary to find the best word similar to the recognized one. If the cost of this replacement is low, for example requires substitution, deletion or addition of one character, it will be applied; otherwise it will be remained unchanged. The cost of this replacement is computed through a modified

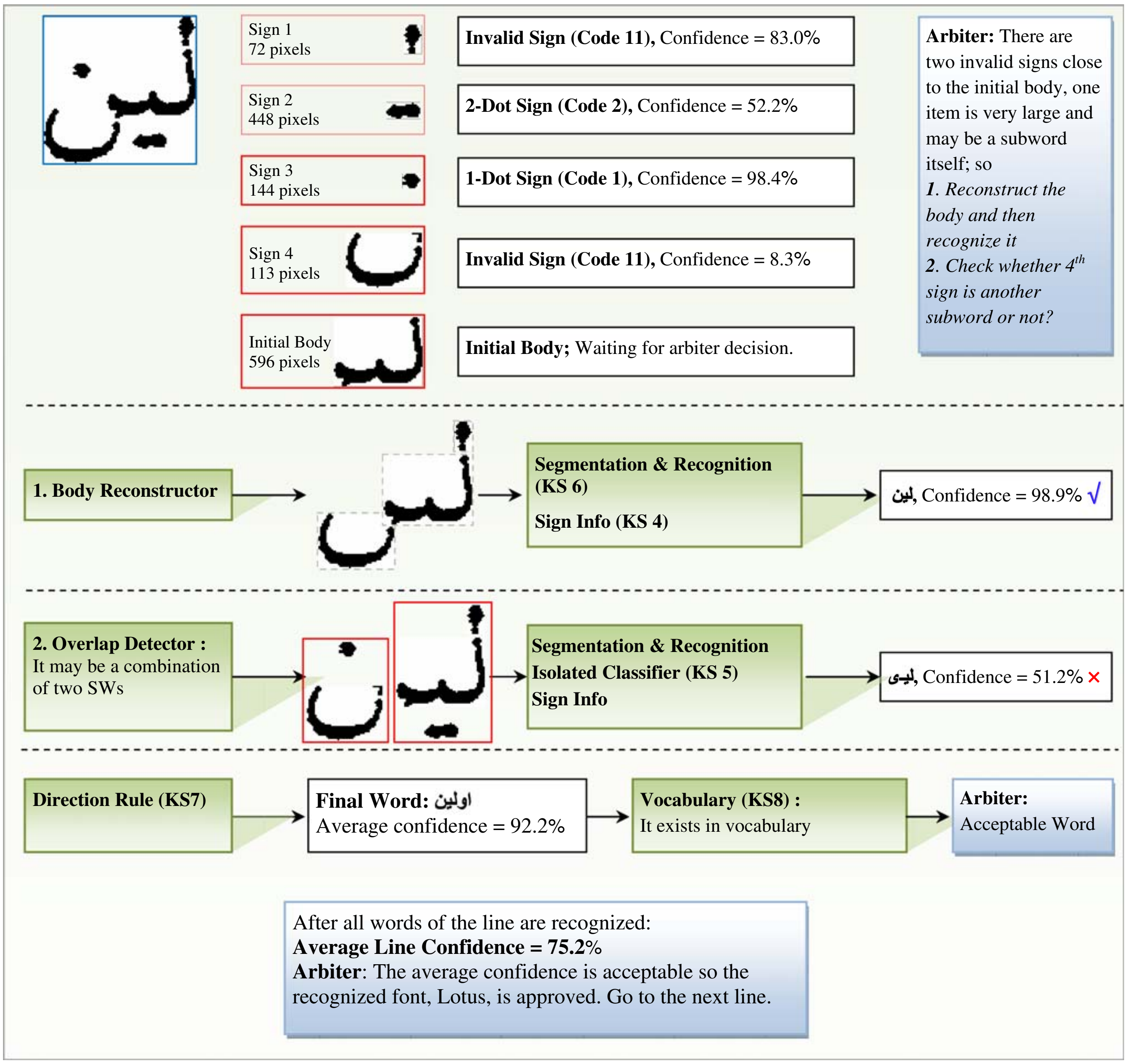

Sign 1
72 pixels
Sign 2
448 pixels
Sign 3
144 pixels
Sign 4
113 pixels
Initial Body
596 pixels
Invalid Sign (Code 11), Confidence = 83.0%
2-Dot Sign (Code 2), Confidence = 52.2%
1-Dot Sign (Code 1), Confidence = 98.4%
Invalid Sign (Code 11), Confidence = 8.3%
Initial Body; Waiting for arbiter decision.
Arbiter: There are two invalid signs close to the initial body, one item is very large and may be a subword itself; so
1. Reconstruct the body and then recognize it
2. Check whether 4th sign is another subword or not?
1. Body Reconstructor
Segmentation & Recognition (KS 6)
Sign Info (KS 4)
لین, Confidence = 98.9% √
2. Overlap Detector : It may be a combination of two SWs
Segmentation & Recognition
Isolated Classifier (KS 5)
Sign Info
لیـی, Confidence = 51.2% ×
Direction Rule (KS7)
Final Word: اولین
Average confidence = 92.2%
Vocabulary (KS8) : It exists in vocabulary
Arbiter: Acceptable Word
After all words of the line are recognized:
Average Line Confidence = 75.2%
Arbiter: The average confidence is acceptable so the recognized font, Lotus, is approved. Go to the next line.

Fig. 13 continued

Levenshtein distance [15]. The vocabulary contains about 55,000 Farsi words with their occurrence probability.

Confusion matrix is a useful knowledge that shows how often two classes may be substituted with each other. This matrix can be extracted through trained samples, or built heuristically. From this knowledge we can decide to replace low confidence characters with suitable choices whenever required. For example, character Gaf "گ" usually is unrecognized as Kaf "ک", so we can substitute these two characters with very low cost.

Levenshtein distance is a cost function which shows how many operations, substitution/deletion/addition, is required to change the first string to the second one. We modified this cost function as follows:

1. The substitution cost is changed w.r.t. characters which are going to be substituted. If both characters are in the same position (beginning, middle or end) the cost will be reduced considering their substitution probability in the confusion matrix.
2. The final cost will be reduced according to the occurrence probability of the new word. This is useful when two substitutions produce the same cost.

Table 4 Experimental results for three strategies

Test strategy	Recognition rate in word level (%)	Recognition rate in char level (%)	Average document confidence (%)	Average execution time (s)
Proposed system	95.81	98.67	92.35	2.73
4 KSs removed	86.66	94.9	90.76	2.31
A KS improved	97.05	99.03	94.21	2.24

Table 5 Scanned document info and recognition results of the system

No.	Fonts included	Word count	Char count	ADC (%)	Word RR (%)	Char RR (%)
1	Titr, Naz	150	627	98.1	98.67	99.68
2	Compset, Yaghut	147	630	89.8	93.88	97.94
3	Lotus	263	1101	95.2	95.44	97.82
4	Nazanin, Yaghut	417	1555	94.3	97.12	99.1
5	Homa, Yaghut	116	484	92.9	100	100
6	Nazanin	63	253	81.9	95.24	98.81
7	Compset	315	1247	92.8	95.24	98.64
8	Lotus	85	451	92.5	95.29	98.67
9	Titr, Zar	162	608	95.8	99.38	99.67
10	Mitra	321	1297	84.2	95.02	98.69
11	Lotus, Mitra	253	1052	93.5	94.86	98.48
12	Karim, Titr	199	947	81.6	84.92	95.88
13	Lotus, Mitra, Naz	224	1156	91.6	91.52	97.92
14	Lotus	523	2054	94.5	96.94	98.88
15	Nazanin	305	1219	95.4	98.36	99.51
16	Nazanin, Traffic	267	1011	98.1	99.63	99.9
17	Nazanin, Zar	240	1049	92.3	95.42	98.67
18	Yaghut	173	898	91	93.06	97.1
19	Tahoma, Times	503	2280	95.5	96.22	99.04
20	Lotus, Mitra	265	1007	96	96.6	99.11
	Sum/Avg	4991	20926	92.35	95.81	98.67

Beside KSs described above, some tools were also developed to facilitate the recognition process; some of them like *Feature extractor, Boosted classifiers, Line detector* and *Spell checker* are required to produce some KSs and some others perform specific actions: *Overlap Detector* (Fig. 12), *Body Reconstructor*, *Dot Remover*, *Character Enhancer*, etc. In the next section we will demonstrate some of these tools through an example.

4 An example

Here we describe the recognition process through an example which shows how KSs interact with the blackboard to recognize a line of text. We tried to select a sample text line in which almost all KSs have good contribution to recognize it (Fig. 13). In this example, the recognition process of the first word of the line is described in detail and all required explanations are inserted on figure as comments. As this figure shows, each KS has its own role towards better recognition of the text line.

Here the arbiter controls all interactions and decides whether the input subword is recognized satisfactorily or not. For example in the case of character Vav “و” which is broken into two parts, at first Dot recognizer and Isolated character classifier, put their knowledge on the blackboard. Then arbiter finds that the final result is not satisfactory because the recognized character “ه” and the sign “Hat” will never join together. Also recognition confidence of the character is too low, 6%, so it proposes that the input subword should be dilated and recognized again. After dilation, which is performed by Character Enhancer tool, the sign is attached to the body and character Vav “و” with confidence of 90% is recognized. Now arbiter approves the result.

A similar process is happened while recognizing the next subword "لين", which its body is broken into three parts due to binarization.

5 Experimental results

To evaluate the proposed system we scanned 20 pages from 16 different books at 300 dpi, using a flatbed scanner (Appendix A). We designed three strategies for the recognition:

1. Recognizing documents with the proposed system.
2. Recognizing after removing some simple KSs from the system to find importance of each KS.
3. Recognizing after substituting KS2, font confidences, with the user provided knowledge about fonts which is more accurate.

The first strategy shows the performance of the system itself. Here the average recognition rate of 95.81% in word level and 98.67% in character level was achieved. The second strategy designed to show how much some simple KSs are important in this system; here we removed four KSs/Tools: Body Reconstructor, Overlap Detector, Character Enhancer and Fonts Confidences. As expected the average recognition rate was reduced from 95.81 to 86.66%. Font Confidences had the most effect on performance reduction. Body Reconstructor and Overlap Detector had almost the same effect on performance and finally Character Enhancer had the lowest effect on average recognition rate. The final strategy was designed to show how a low performance KS may decrease the overall recognition rate. In this stage we substituted the font confidences KS with user provided knowledge about fonts involved in each document. In other words we replaced the font recognizer module with a perfect font recognizer. In this way the average recognition rate increased to 97.05%. Table 4 shows the global results briefly and Table 5 shows documents information and their recognition rates using the proposed system.

We also put the average execution time for each strategy in Table 4. As it shows, average execution time per each document is fairly short. These times are computed on a PC with 2.4 GHz CPU and 2 GB of RAM. The required time in the first strategy is greater than others, because here a time for font recognition and taking care of its output confidences is added. The details of execution times are shown in Fig. 14.

According to Table 5, we find that test pages two and seven contain Compset font which is not supported in the proposed system. But since this font is too similar to Lotus, the overall recognition rate was not affected noticeably (Table 5, second and seventh row).

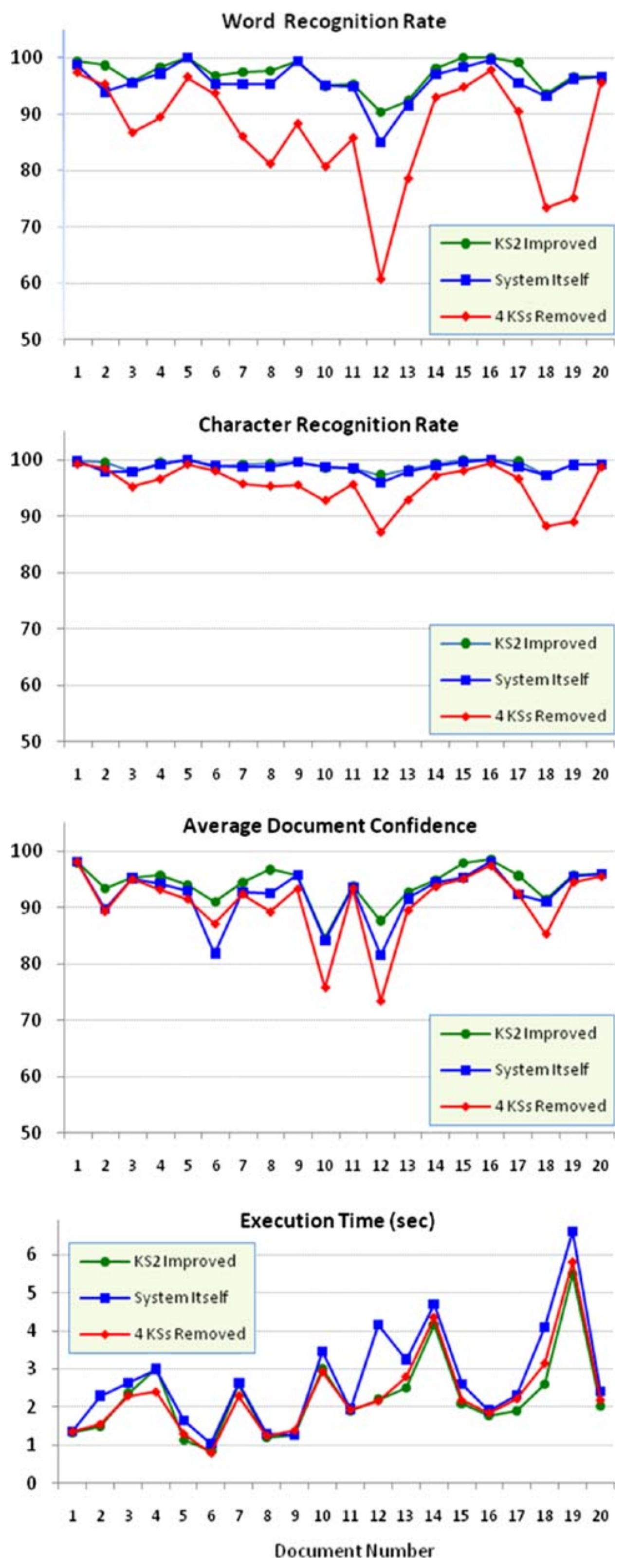


Fig. 14 Recognition rates in word level and character level and average document confidence for 20 test pages

In test page 12 which the main font is Karim, the recognition rate is the lowest within all pages, 84.92%, because this font is not supported and it is not similar to any other font. Other pages have been read with high accuracy.

These results show that the system performance on typical documents of familiar fonts with resolution of 300 dpi is quite well. However, for the case of low quality documents like Fax pages, old documents, camera images and low-resolution documents, the problem is still open. In this work, we did not consider these kinds of documents. In these cases there are several broken characters, nonlinear distortions and noise so that we need some other KSs to be designed to improve the system accuracy. For example a preprocessor module may be required to enhance the quality of the image.

One advantage of our system is that we can compute an average confidence for the whole document. We have several confidence measures from several KSs, Font Recognizer, Language Recognizer and Other Classifiers. From these confidences we can compute a confidence for each word and for the whole document as well. Here we computed a confidence measure called average document confidence as follows:

$$\mathrm{ADC} = \frac{1}{n}\sum_{i} C[i]$$

Where n is the number of detected words in the document and $C[i]$ is the confidence of word[i] which is computed through character confidences, signs confidences and word occurrence probability in the vocabulary.

This confidence shows how good a document is recognized. In Fig. 14 we show the recognition rates and average document confidences for all 20 test documents graphically. As this figure shows, the average document confidence almost goes with the recognition rates.

6 Conclusion

In this paper, we proposed an integrated Farsi OCR system based on blackboard architecture. In this system several KSs and tools interact with the blackboard and perform appropriate actions whenever required. An arbiter controls KSs actions and decides whether the problem is solved satisfactorily or not based on current blackboard state. The system can recognize ten popular Farsi fonts using a Font Recognizer module. It also detects English words in between Farsi sentences and recognizes them using an English OCR engine. The proposed system was tested on 20 real-life documents. The average recognition rates of 97.05% on word level and 99.03% on character level were achieved. Based on several confidences from some KSs, an average document confidence is computed which shows how good a document is recognized.

Appendix A: sample test pages

موفقیت چیست؟

دبی به دفتر کارش بازگشت. آشکار بود که روحیه‌ی او بسیار بهتر شده است. او امیدوار بود که کارایی تیم نیز بهتر خواهد شد. او بازهم تا حد امکان گوش می‌کرد و راه‌هایی برای خدمت به تیم جست‌وجو می‌کرد. او اغلب کارها را به اعضای تیم می‌سپرد تا خود وقت بیشتری برای فکر کردن به آینده داشته باشد. او به عوض تکیه کردن بر بخش منابع انسانی برای معرفی نیروی کار، خود به دنبال یافتن افراد بااستعداد بود. او وقت بیشتری را در جریان مصاحبه‌ها سرمایه‌گذاری می‌کرد و فعالیت هدفمندی برای به خدمت گرفتن قلب و مغز افراد انجام می‌داد.

گاهی سرش خیلی شلوغ می‌شد، اما در اعماق قلبش می‌دانست که در حال ایجاد زیربنایی برای چیزهای بزرگ‌تر است. همچنان که خود را برای جلسه‌ی آینده‌ی تیم آماده می‌کرد، واقعاً می‌خواست که تیمش را در بازسازی مداوم به کار گیرد. او ایمیلی ارسال کرد با این مضمون:

۹۳

Test Page # 1

مخفیه اسرار اتومبیل ۴۳

انگشت دست لمس کلید زمانی که این روغن هیچ گونه چسبندگی نداشته باشد بهترین زمان برای تعویض آن است .

- اگر روغن موتور شما خیلی سریع سیاه می شود نشانه این است که بنزین در محفظه احتراق کاملاً سوخته نمی شود و اصطلاحاً موتور دچار احتراق ناقص است برای این منظور می توانید جرقه را توسط دلکو تنظیم کنید اگر از خودروهای انژکتوری استفاده می کنید حتماً به نزدیکترین تعمیرگاه مجاز مراجعه کنید تا سیستم برق خودروی شما را با دستگاه های تنظیم Ecu کنترل کنند .
- اگر در روغن موتور شما اثری از آب دیده می شود ممکن است موتور شما در اثر خرابی واشر سر سیلندر آب سوز شده باشد برای این

Test Page #

گذشته

[illegible]

5

Test Page # 12

قانون اساسی جمهوری اسلامی ایران صفحه ۶۵

مراتب آن‌ها را که باید با رعایت اصول وحدت ملّی و تمامیّت ارضی و نظام جمهوری اسلامی و تابعیّت حکومت مرکزی باشد قانون معیّن می‌کند.

اصل یکصد و یکم

به منظور جلوگیری از تبعیض و جلب همکاری در تهیّه برنامه‌های عمرانی و رفاهی استان‌ها و نظارت بر اجرای هماهنگ آن‌ها، شورای عالی استان‌ها مرکّب از نمایندگان شوراهای استان‌ها تشکیل می‌شود.

نحوهٔ تشکیل و وظایف این شورا را قانون معیّن می‌کند.

اصل یکصد و دوم

شورای عالی استان‌ها حقّ دارد در حدود وظایف خود طرح‌هایی تهیّه و مستقیماً یا از طریق دولت به مجلس شورای اسلامی پیشنهاد کند. این طرح‌ها باید در مجلس مورد بررسی قرار گیرد.

اصل یکصد و سوم

استانداران، فرمانداران، بخشداران و سایر مقامات کشوری که از طرف دولت تعیین می‌شوند در حدود اختیارات شوراها ملزم به رعایت تصمیمات آن‌ها هستند.

اصل یکصد و چهارم

به منظور تأمین قسط اسلامی و همکاری در تهیّه برنامه‌ها و ایجاد هماهنگی در پیشرفت امور در واحدهای تولیدی، صنعتی و کشاورزی، شوراهایی مرکّب از نمایندگان کارگران و دهقانان و دیگر کارکنان و مدیران، و واحدهای آموزشی، اداری، خدماتی و مانند این‌ها شوراهایی مرکّب از نمایندگان اعضای این واحدها تشکیل می‌شود.

چگونگی تشکیل این شوراها و حدود وظایف و اختیارات آن‌ها را قانون معیّن می‌کند.

Test Page # 14

گلچینی از فاطمه فاطمه است

از علی خواسته بود تا او را شب دفن کنند، گورش را کسی نشناسد. و علی چنین کرد.

امّا کسی نمی‌داند که چگونه؟ و هنوز نمی‌داند کجا؟ در خانه‌اش؟ یا در بقیع؟ معلوم نیست. آنچه معلوم است، رنج علی است. امشب بر گور فاطمه. مدینه در دهان شب فرو رفته است. مسلمانان همه خفته‌اند. سکوت مرموز شب گوش به گفتگوی آرام علی دارد. و علی که سخت تنها مانده است، هم در شهر و هم در خانه، بی‌پیغمبر، بی‌فاطمه، همچون کوهی از درد، بر سر خاک فاطمه نشسته است.

ساعت‌ها است.

شب ـ خاموش و غمگین ـ زمزمهٔ درد او را گوش می‌دهد. بقیع آرام و خوشبخت و مدینهٔ بی وفا و بدبخت، سکوت کرده‌اند. قبرهای بیدار و خانه‌های خفته می‌شنوند.

نسیم نیمه شب کلماتی را که به سختی از جان علی بر می‌آید از سر گور فاطمه به خانهٔ خاموش پیغمبر می‌برد: ـ «بر تو، از من و از دخترت، که در جوارت فرود آمد و به شتاب به تو پیوست، سلام ای رسول خدا».

ـ «از سرگذشت عزیز تو ـ ای رسول خدا ـ شکیبایی من کاست و چالاکی من به ضعف گرایید. امّا، در پی سهمگینی فراق تو و سختی مصیبت تو، مرا اکنون جای شکیب هست».

هم اکنون دخترت ترا خبر خواهد کرد که قوم تو بر ستمکاری در حق او همداستان شدند. به اصرار از او همه چیز را بپرس و سرگذشت را از او خبر گیر. اینها همه شد، با اینکه از عهد تو دیری نگذشته است و یاد تو از خاطر نرفته است.

بر هر دوی شما سلام، سلام وداع کننده‌ای که نه خشمگین است نه ملول.

Test Page # 16

3rd Conference on Information and Knowledge Technology
Ferdowsi Univ. of Mashad, Faculty of Engineering, Nov. 27-29 2007

عنوان نشست: پردازش تصویر ۲	کد نشست: C1-3
زمان نشست: ۱۴ الی ۱۵:۴۰	زمان ارائه: ۱۴:۴۰

یافتن زاویهٔ چرخش سند مبتنی بر زوایای خطوط متن با استفاده از عملگر مورفولوژی

حسین خسروی1، احسان الله کبیر2

[illegible]

چکیده

[illegible]

[illegible]

Test Page # 17

References

1. Abdelazim, H.Y., Hashish, M.A.: Arabic reading machine. In: Proceedings of the 10th National Computer Conference, Jeddah, pp. 733–744 (1988)
2. Al-Shoshan, A.I.: Arabic OCR based on image invariants. In: Proceedings of the International Conference on Geometric Modeling and Imaging—New Trends, pp. 150–154 (2006)
3. Amin, A.: Off-line Arabic character recognition: the state of the art. Pattern Recognit. **31**(5), 517–530 (1998)
4. Azmi, R., Kabir, E.: A new segmentation technique for omnifont Farsi text. Pattern Recognit. Lett. **22**, 97–104 (2001)
5. Cheung, A., Bennamoun, M., Bergmann, N.W.: An Arabic optical character recognition system using recognition-based segmentation. Pattern Recognit. **34**, 215–233 (2001)
6. Ebrahimi, A., Kabir, E.: A pictorial dictionary for printed Farsi subwords. Pattern Recognit. Lett. **29**(5), 656–663 (2008)
7. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: International Conference on Machine Learning, Bari, Italy, pp. 148–156 (1996)
8. Gouda, A.M., Rashwan, M.A.: Segmentation of connected Arabic characters using hidden Markov models. IEEE International Conference on Computational Intelligence for Measurement Systems and Applications, USA pp. 115–119 (2004)
9. Houle, G., Shridhar, M.: Handwritten word recognition with OCR-based segmenter. In: Proceedigns of the Workshop on Document Image Analysis, pp. 51–58 (1997)
10. Khosravi, H., Kabir, E.: Introducing a very large dataset of handwritten Farsi digits and a study on their varieties. Pattern Recognit. Lett. **28**(10), 1133–1141 (2007)
11. Khosravi, H., Kabir, E.: Introducing two fast and efficient features for Farsi digit recognition (in Farsi). Machine Vision and Image Processing, Mashhad, pp. 1126–1131 (2007)
12. Khosravi H., Kabir, E.: Farsi font recognition based on Sobel-Roberts features. Pattern Recognit. Lett. (Under Review) (2008)
13. Kimura, F., Shridhar, M., Chen, Z.: Improvements of a Lexicon directed algorithm for recognition of unconstrained handwritten words. In: Proceedings of 2nd ICDAR Conference, pp. 18–22 (1993)
14. Kurdy, B., AlSabbagh, M.: Omnifont Arabic optical character recognition system. In: Proceedings of International Conference on Information and Communication Technologies: From Theory to Applications, pp. 469–470 (2004)
15. Levenshtein, V.: Binary codes capable of correcting deletions, insertions, and reversals. Sov. Phys. Doklady **10**(8),707–710 (1966)
16. Mansoory, S., Hassibi, H., Rajabi, F.: A heuristic Persian handwritten digit recognition with neural network. In: The 6th Iranian Conference on Electrical Engineering, pp. 131–135 (1998)
17. Mehran, R., Pirsiavash, H., Razzaziy, F.: A front-end OCR for omni-font Persian/Arabic cursive printed documents. Digital Imaging Computing: Techniques and Applications, pp. 385–392 (2005)
18. Menhaj, M.B., Adab, M.: Simultaneous segmentation and recognition of Farsi/Latin printed texts with MLP. In: International Joint Conference on Neural Networks, pp. 1534–1539 (2002)
19. Nabavi, S.H., Ebrahimpour, R., Kabir, E.: Recognition of handwritten Farsi digits using classifier combination. In: Third Conference on Machine Vision, Image Processing and Applications, Tehran, pp. 116–119 (2005)
20. Nashida, H., Mori, S.: An Algebraic approach to automatic construction of structured models. Pattern Anal. Mach. Intell. **15**(12), 1298–1311 (1993)
21. Parhami, B., Taraghi, M.: Automatic recognition of printed Farsi texts. Pattern Recognit. Lett. **14**, 395–403 (1981)
22. Sarfraz, M., Nawaz, S.N., Al-Khuraidly, A.: Offline Arabic text recognition system. In: Proceedings of International Conference on Geometric Modeling and Graphics, pp. 30–35 (2003)
23. Soltanzadeh, H., Rahmati, M.: Recognition of Persian handwritten digits using image profiles of multiple orientations. Pattern Recognit. Lett. **25**(14), 1569–1576 (2004)
24. Yazdi, S.A.B., A'rabi, B.N.: Printed Farsi text recognition with simultaneous use of HMM. In: Dynamic Programming and SVM (in Farsi), Machine Vision and Image Processing, Mashhad (2007)