Pattern Recognition Vol. 14, Nos. 1-6, pp. 395-403, 1981.
Printed in Great Britain.

0031-3203/81/070395-09 $02.00/0
Pergamon Press Ltd.

AUTOMATIC RECOGNITION OF PRINTED FARSI TEXTS

B. PARHAMI and M. TARAGHI
Sharif University of Technology, Tehran, Iran

(*Received* 9 *January* 1980; *received for publication* 22 *December* 1980)

Abstract—The automatic recognition of printed Farsi (Persian) texts is complicated by several properties of the Farsi script: (a) connectivity of symbols, (b) similarity of groups of symbols, (c) highly variable widths, (d) subword overlap, and (e) line overlap. In this paper, a technique for the automatic recognition of printed Farsi texts is presented and its steps are discussed as follows: (1) digitization, (2) editing, (3) line separation, (4) subword separation, (5) symbol separation, (6) recognition, and (7) postprocessing. The most notable contributions of this work are in algorithms for steps (5) and (6) above. Practical application of the technique to Farsi newspaper headlines has been 100% successful. However, smaller type fonts, which could not be handled by the coarse digitization hardware used, will no doubt result in less than perfect recognition. The technique is also applicable with little or no modification to printed Arabic and Urdu texts which use the same alphabet as Farsi.

Character recognition Computer input Document input Farsi Feature selection
Optical character recognition Pattern recognition Persian Printed text recognition

BACKGROUND

Automatic recognition of printed or handwritten texts provides a convenient means of communication with computers. There have been numerous studies in this area for various languages.[1, 2] However, in the case of Farsi (as well as Arabic and Urdu) no such study is known to the authors. Recognition of handwritten Farsi texts appears to be beyond our present day capabilities in computing, since even human readers experience considerable difficulties in this respect. This point will become more clear as we subsequently describe the problems encountered in recognizing printed Farsi texts.

The difficulties which arise from unique properties of the Farsi script in the field of computing have been enumerated elsewhere.[3, 4, 5] The impeding properties pertaining to the automatic recognition of printed Farsi texts are as follows.

1. The possible connectivity of adjacent alphabetic symbols (similar to cursive Latin script).
2. The similarity of many groups of Farsi symbols, some differing only in number and/or places of small dots.
3. The relatively large number of Farsi alphabetic symbols, when taking into account the different context-dependent forms of each letter.
4. The varying geometric sizes of letters (particularly their widths), even in typewritten texts.

Figure 1 illustrates these and related problems.

In this paper, we will describe a technique for automatic recognition of printed Farsi texts. The method has been implemented and is in practical use at the Computer Systems Laboratory of Sharif University of Technology, Tehran. It consists of the following steps.

1. Digitization and preprocessing: (a) digitization of the printed Farsi text; (b) editing of the digitized text; (c) separation and juxtaposition of lines to form a single line of digitized text; (d) Identification and separation of subwords in text.
2. Separation of symbols: (a) Isolation of symbols (subsymbols) within each subword.
3. Recognition and post-processing: (a) Recognition of symbols and subsymbols; (b) Consolidation of subsymbols, consistency checks, and generation of final output.

Each of the parts 1, 2, and 3 will be described in one of the following sections of the paper.

Verification tests were conducted with newspaper headlines using the coarse digitization hardware available to us. The tests where 100% successful. However, less than perfect recognition would be expected for smaller type fonts due to the relatively higher effect of

Fig. 1. Difficulties in the automatic recognition of printed Farsi texts: (a) connectivity of symbols, (b) similarity of symbols, (c) variable-width symbols, (d) overlapping subwords, and (e) overlapping lines of text.

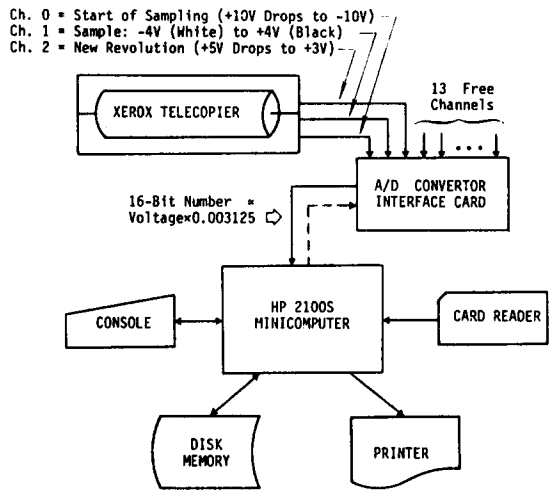

Fig. 2. Hardware configuration for the experimental text recognition system.

digitization noise and print imperfections, even if suitably fine digitization hardware is utilized.

DIGITIZATION AND PREPROCESSING

Text digitization in the experimental system is carried out by means of a Xerox Telecopier, connected to an HP 2100S minicomputer through an A/D convertor interface card[6] as shown in Fig. 2. The system's sampling resolution is seven samples cm^{-1} along the X (circumferential) axis and twelve samples cm^{-1} along the Y (longitudinal) axis. The finer vertical details of Farsi symbols, such as loops and dot separation, made it more desirable to place the text lines along the X axis. This was done in all the experiments.

The image obtained from the text is converted to a binary matrix (grid) by considering a threshold voltage of 3.5 V. A direct line-printer image of this matrix (with zeros represented by blanks) gives a realistic representation of the original text due to the unequal spacing of printed characters (ten per inch) and lines (six per inch). Thus, as shown in Fig. 3, the X axis appears in the direction of paper movement, enabling the representation of arbitrarily long texts. For storage efficiency, the image is stored by (x, y) pairs of its dark points.

The editing of digitized text consists of repeatedly replacing the value of each point by the value of the following logical expressions (see Fig. 3 for notation):

$$x \leftarrow x + bg(d + e) + de(b + g)$$

$$x \leftarrow x(g + bd + be + de + ah + cf).$$

The first replacement is intended to restore missing points (holes or notches) while the second one will delete noise and loosely connected edge points. The appearance of g in the second expression protects loose top edge points since deleting them would cause difficulties in recognizing serrated characters.

The next step is to transform the sample into a single line of text. This would have been straightforward were it nor for the line overlap feature depicted in Fig. 1. The algorithm we use is as follows. First, the number of dark points is found for each row of the grid. This number almost invariably peaks at or near the center row of each text line. Taking the top line first, we mark all dark points no lower than a certain distance below its 'peak' row (say 25% of the distance to next peak row). We then repeatedly mark the neighbouring dark points of each previously marked point. Finally, some character dots, which may still be unmarked after this phase, are assigned to the text line with closest peak.

Once the text is made into a single continuous line, the last preprocessing phase consisting of the elimination of subword overlaps is applied. This is done by means of an algorithm which also separates the dots from subwords.

Starting from the right end of the text, the columns are scanned. In a typical iteration, the ith column which may have several segments of dark points is considered. The longest of these segments is selected (since the dots usually correspond to smaller segments) and its points are marked. Then repeated marking of neighboring points is performed until a pass produces

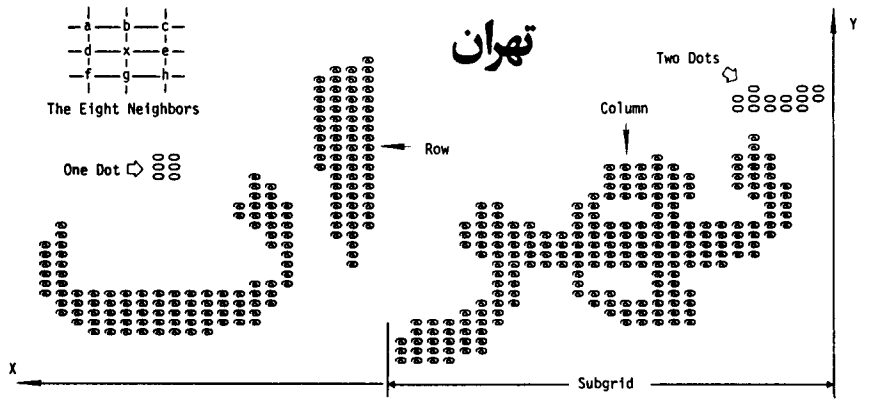

Fig. 3. A printed Farsi word and its digital representation after editing, elimination of subword overlaps and isolation of dots, shown along with grid notation.

no new marked points. Now, if the number of marked points is less than half the total number of dark points in the subgrid (the portion of grid between the right end and the last marked column), then we have marked the dot(s) of a character. The marked points are so designated and the process is repeated.

Otherwise, we have marked a subword (a portion of a word consisting of connected characters) excluding its dots, if any. We then start at the last column of the subgrid where any unmarked dark points belong to an overlapping subword. We trace the subword backwards by marking its points (in a different way). We have now separated the overlapping subwords. We start recording the subgrid columns on the final pre-edited file, with each record consisting of the y coordinate of dark points in a column (normally integers less than 100).

The points corresponding to dots are represented by $y - 100$, always a negative number. Finally, a blank column, represented by the single number 500 in the corresponding record, is added after the subword. Then, the recorded points are deleted from the grid and the process starts again from the first column containing non-recorded dark points.

The word shown in Fig. 3 has overlapping subwords whereas in its final digitized form, the overlap has been eliminated and a blank column inserted between the two subwords. Following the preprocessing phase, each subgrid (consisting of a number of consecutive non-blank columns between two blank columns) contains a subword whose symbols must be separated for recognition.

SEPARATION OF SYMBOLS

In order to understand the algorithm we use for separating the symbols in a subword it is necessary to examine some properties of the Farsi script. The Farsi font design is done using a rectangular-tip pen, having a length much greater than its width. As the designer moves the pen at certain angles to generate each symbol, lines of varying thicknesses appear.

In the design of connectable letters, the following rules are observed.

1. Each pair of letters is connected at a single point. Furthermore, all connection points in a line of text are horizontally aligned.

2. The longer side of the pen tip rectangle is perpendicular to the direction of hand movement at the connection point, thus producing a line with maximum thickness.

3. There is no symbol overlap; thus 'cutting' the text at a connection point by a vertical line will leave the symbols on both sides intact.

The first step in determining the connection points is to find the pen (script) thickness. Consider a column of digitized text. The dark points in this column appear in segments. Collect statistics of the length of these segments for the entire text. The segment length appearing most frequently (denoted by T) is a digitized approximation of the pen thickness. This is a statistical property of printed Farsi texts. The text does not have to be very long to obtain the correct result. In the great majority of cases, a single word is sufficient and a full line of text almost never produces an incorrect result.

To take the digitization error into account, we say that a segment has the pen thickness if its length t is in the range $(T - 1) \leq t \leq (T + 1)$. Experience has shown that for a type font to be recognizable, it must have $T \geq 4$. For smaller values of T, many symbols become indistinguishable, thus resulting in an unacceptable recognition error. Therefore, the minimum acceptable resolution for the digitization hardware is four samples per script thickness. This translates into about 100 samples per cm for type fonts in typewriters and most Farsi textbooks.

We can now define a potential connection column (PCC) as one having the following properties.

1. The unique segment of dark points in PCC has the pen thickness.

2. The first column to the right of PCC has a segment with pen thickness at the same vertical location (possibly shifted by $+1$ or -1) as PCC.

3. The column immediately to the left of PCC is not a PCC; i.e., it does not satisfy conditions 1 and 2.

Thus in Fig. 4(a), we find three PCCs, two of which are actual connection columns (ACCs) as shown in Fig. 4(b). We have not used the horizontal alignment property of connection points since we wish to recognize each subword, which may have a small number of connection points (typically three or less), independently of others. Note that PCC, as defined here, is

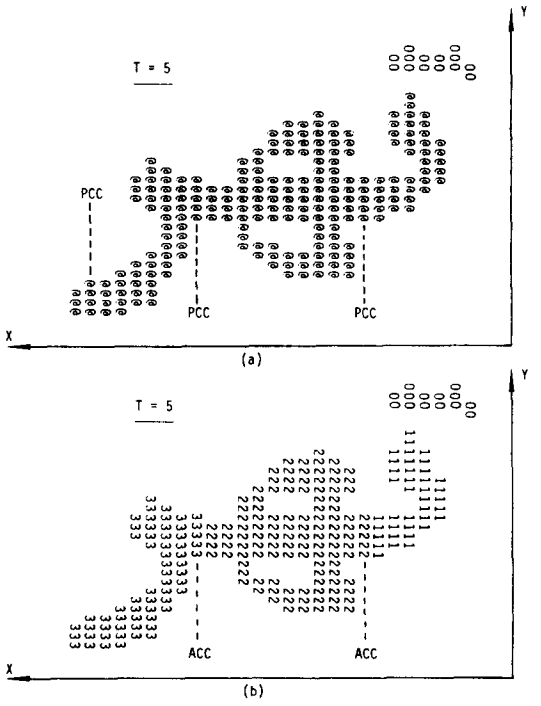

Fig. 4. Separation of symbols within a subword: (a) potential connection columns in the subword and (b) actual connection columns and separated symbols.

Table 1. Single Farsi symbols with PCCs.

GROUP	DESCRIPTION (ACTION)	SYMBOLS*
1	Solitary or terminal symbols with at most one PCC but with no ACC (not decomposed for recognition)	ص ض ر ش د ب ب ک گ گ ی ی ف ف ی ی و و ن ل ل ۰ ۲ ۴ ۶ ۰
2	Symbols with at most one PCC (may be decomposed before recognition)	ه ه ۴
3	Symbols with 2 or 3 PCC's (may be decomposed before recognition)	ـ س س ـ

* Symbol dots are not shown.

potentially the first column of a connected character.

Not every PCC is an ACC, as shown in Fig. 4. There are certain individual characters that have one to three PCCs within them (Table 1). We have divided these into three groups.

In Group 1, we have characters which end a subword and which may have a single PCC within them. A common property of symbols in this group is that their PCC is very close to the left-hand end of the symbol (and thus of the subword). Specifically, the distance between the PCC and the last column of the subword is less than T (frequently it is much less, as seen in Fig. 4 for the leftmost PCC). When this happens, the PCC is not an ACC unless it is followed by a terminal 'Aleph' character (symbol number 04 in Table 3). This can be easily tested, since in such a case segments with a length greater than $2T$ and with no point below the PCC segment will appear in columns to the left of the PCC.

In Group 2, we have all other characters with a single PCC within them. These will fail the test of Group 1. The easiest way to handle these is to consider their PCCs as ACCs and let them be decomposed into two subsymbols prior to recognition.

In Group 3, we have characters with two or three PCCs. The third or leftmost PCC, when present, can be eliminated by a test identical to that of Group 1. Again, we let these symbols be decomposed into three subsymbols along their remaining two PCCs prior to recognition.

RECOGNITION AND POST-PROCESSING

The recognition procedure to be described is based on certain geometric properties of Farsi symbols. To describe these properties, we first present some definitions (see Fig. 5).

C_{max} = the leftmost subgrid column containing dark points;
C_{min} = the rightmost subgrid column containing dark points;
R_{max} = the uppermost row containing dark points;
R_{min} = the lowermost row containing dark points;
$x_{ave} = (C_{max} + C_{min})/2$;
$y_{ave} = (R_{max} + R_{min})/2$.

We assume that the symbol dots, separated in the previous phase, have been removed and that their existence is taken into account only at the final stage of recognition.

We further define a *loop point* as one which cannot be reached from the point (1,1); i.e., no sequence of moves to neighboring points starting from the point (1,1) can lead to a loop point. Hence, a simple algorithm to find loop points is to start at point (1,1) and successively mark the eight neighbours of each previously marked point. The algorithm stops when a pass produces no new marked points.

Finally, we define a *concavity* in the positive X direction as a set of connected light grid points which can be reached from the point (1,1) but not if we disallow movement to the right. A concavity in the positive X direction is considered *major* if its points span at least $3T/4$ columns. It should be noted that here and in the subsequent discussion, lengths measured in terms of number of rows and number of columns correspond to the particular digitization hardware used. In other words, column distances are roughly 12/7 times greater than row distances.

It is easily seen that by disallowing movement to the right (i.e., only the upper, lower and the three left neighbors of a marked point are marked), an algorithm similar to the one used for detecting loop points is applicable here. Concavity in each of the other three directions is defined analogously (Fig. 5). In the case of concavity in the negative X or Y direction, the starting point should be the upper left corner of the subgrid. An algorithm can be designed to simultaneously obtain all four types of concavity and loop points by a more sophisticated marking scheme.

Note that the four quadrants in Fig. 5 are numbered in such a way that the two-bit binary numbers

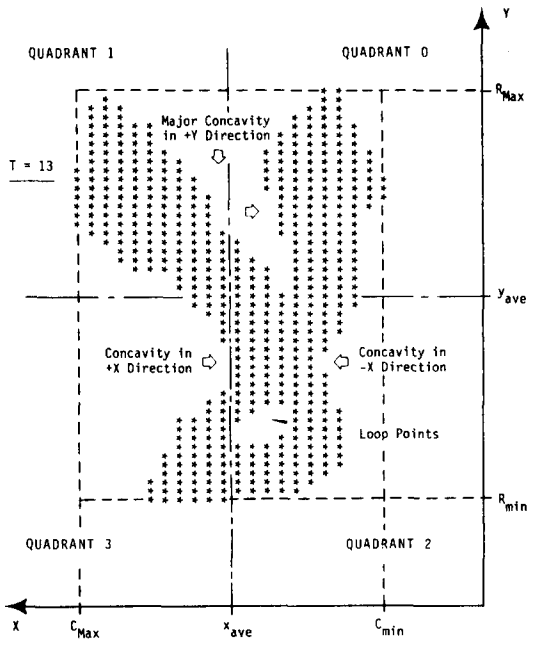

Fig. 5. Definitions of some terms and properties used in the recognition phase.

Table 2. Features used in the recognition of printed Farsi symbols

Name	Description of feature
Q_M	Quadrant with maximum number of dark points (two-bit number)
Q_m	Quadrant with minimum number of dark points (two-bit number)
X_+	Major concavity in the positive X direction
X_-	Major concavity in the negative X direction
Y_+	Major concavity in the positive Y direction
Y_-	Major concavity in the negative Y direction
C_l	Connectivity to the left (next symbol)
C_r	Connectivity to the right (previous symbol)
L_u	Loop points in the upper quadrants (above y_{ave})
L_d	Loop points in the lower quadrants (below y_{ave})
C_2	Number of two-segment columns less than max($T/4$, 2), between max($T/4$, 2) and T, or greater than T (denoted by 00, 01, or 11, respectively)
R_2	Number of two-segment rows less than max($T/3$, 3) between max($T/3$, 3) and $3T/2$, or greater than $3T/2$ (denoted by 00, 01, or 11, respectively)
C_T	More than max($T/3$, 3) single-segment columns with script thickness
C_3	More than $T/2$ three-segment columns
C_M	Midpoint between the uppermost and the lowermost dark points of C_{max} in Quadrant 3
C_m	Midpoint between the uppermost and the lowermost dark points of C_{min} in Quadrant 2
R_W	Value of $C_{max} - C_{min}$ greater than T
R_M	Midpoint between the leftmost and the rightmost dark points of R_{max} in Quadrant 1
R_l	Maximum segment length in rows above y_{ave} greater than T
$R_\#$	Number of single-segment rows above y_{ave} greater than $3T/2$

corresponding to neighboring quadrants differ by one bit.

With these definitions, we now present the 20 geometric features used to classify Farsi symbols and subsymbols. Table 2 shows these features, which yield a 24-bit vector for a symbol to be recognized. Other than Q_M, Q_m, C_2, and R_2 which correspond to two-bit numbers, all features in Table 2 are represented by a single bit, with 1 denoting that the symbol being classified possesses that feature. The connectivity parameters (C_l and C_r) are obtained in the symbol separation phase.

The 24-bit vector thus obtained is compared against those given in Table 3 for Farsi symbols and subsymbols. The procedure is to first seek an exact match. If an exact match is not found, we then look for a best match in which the more reliable features (C_l, C_r, L_u, L_d, and R_W) are compared first.

The classification scheme of Table 3 assigns a given 24-bit vector to a unique symbol except in the case of the following six pairs: (05,53), (29,33), (29,35), (29,39), (32,52), and (20,52). In the first five cases the ambiguity is resolved when we consider the dots (discussed subsequently). For the symbol pair (20, 52), a simple test based on the maximum row and column segment lengths among the loop points is used. If the maximum length of row segments among loop points is more than the maximum length of column segments, then we have symbol number 20. Otherwise, the symbol being considered is number 52.

For the symbol dots, two factors are important; their location (above or below y_{ave}) and their number one, two or three). If the number of grid points covered by dots is denoted by D, then the following inequalities hold for most type fonts.

$$\begin{aligned} \text{One dot:} \quad & 0.1\ T^2 < D < 0.6\ T^2 \\ \text{two dots:} \quad & 0.6\ T^2 < D < 1.1\ T^2 \\ \text{three dots:} \quad & 1.1\ T^2 < D. \end{aligned}$$

A further test is provided by considering the smallest rectangle enclosing the dots. For two dots, the height of this rectangle is less than its width, while for one and three dots the converse is true (see Fig. 3, for example). This test allows for the identification of dots in the case of those type fonts slightly violating the above inequalities.

The number of dots in the case of ambiguous symbols discussed earlier is as follows.

$$\begin{aligned} &05: D(+ > 1)/D(-1)/D(-3) \\ &29: D(+1) \\ &32: D(+1)/D(+2) \end{aligned}$$

Table 3. Specification of features for classifying Farsi symbols*.

No	Q_M	Q_m	X_+	X_-	Y_+	Y_-	C_ℓ	C_r	L_u	L_d	C_2	R_2	C_T	C_3	C_M	C_m	R_W	R_M	R_ℓ	$R_\#$	Sym
01	0	1	0	0	0	0	0	0	0	0	01	00		0	0	0	1		1	1	ا
02		11	0		0	0	0	1	0	0	1	00		0	0	1	1		1	1	ا
03	0	1	0	0	0	0	0	0	0	0	00	00	0	0	0		0	0	0	1	ا
04		1	0	0	0	0	0	1	0	0	00	00	0	0		1	0		0	1	ا
05	1	01	0	0	1	0	0	0	0	0	0	01	1	0			1	0	0	0	ب
06	11	01	0	0	1	0	0	1	0	0	00	01	1	0			1		1	0	ب
07		01	0	0	0	0	1	0	0	0	01	00	1	0	1	0	1	0	0		ب
08	1	0	0	0	0	0	1	1	0	0	00	00	1	0	1	1	1	0		0	ب
09	1	10	1	1	0	0	0	0	0	0	11	00	0		1		1	1	1	1	ب
10		10	1	1	0		0	1	0	0	11	0	0		1	0	1	1	1		ب
11	AA	BB̄	1	0	0	0	1	0	0	0	1	00		0	1		1	1	1		پ
12		01	1		0		1	1	0	0	11	0		0	1	1	1	1	1		پ
13	1	01	1	0	0	0	0	0	0	0	11	0		0	1		1	1	1		پ
14	1	01	0	0	0		0	1	0	0	01	1	1	0	1	1	1	0	0	1	پ
15		0 01		0	0	0	0	0	0	0	01	00	0	0	1	0	1	0		1	پ
16		0 01		0	0	0	0	1	0	0	01	00	0	0	1	0	1	0		1	پ
17	AA	BB̄	0	0	1	0	0	0	1	0	1	11	1	0	1	0	1	0	1		پ
18	AA	BB̄	0	0	1	0	0	1	1	0	1	11	1	0	1		1	0	1		پ
19		01	0	0		0	1	0	1	1	11	0	1	0	1	0	1	0	1	0	پ
20		01	0	0		0	1	1	1	1	11	0	1	0	1	1	1	0	1	0	پ
21		01	0	0	0	0	1	0	1	1	11	01	0	0	1	0	1	0	1	0	ت
22		01	0	0	0	0	1	1	1	1	11	01	0	0	1	1	1	0	1	0	ت
23	1	00	0	0	1	0		0	0	1	11	01		0	1		1	1		1	ت
24	1	00	0	0	1	0		1	0	1	11	1		0	1	1	1	1		1	ت
25	AA	10	0	1	0	0	0	0	0	0	0	0	0	1	1	0	1	0	1	1	ت
26		10	0	1	0	0	0	1	0	0	11	0	0	1	1	0	1		1		ث
27		01	0	1	0	0	1	0	0	0		1 00		0	1		1	0	1		ث
28			0	0	0		1	1	0	0	01	01		0	1	1	1		1		ث
29		01	1	0	1	0	0	0		0	1	1	1	0			1	0	1		ث
30		01	0	0	1	0	0	1	1	0	0	0	1	0		1	1	0		0	ث
31	00	01	1	0	0	0	1	0		0	1	0		0	1		1	0	1		ج
32	1	0	0	0	0	0	1	1	1		01	01		0	1	1	1	0		0	ج
33	00	01	1	0	1	0	0	0	1	0	0	11	1	0	1		1	0	1		ج
34	00	01	1	0	1	0	0	1	1	0	0	11	1	0	1	0	1	0	1		ج
35		01		0		0	0	0	0	0	01	0	1	0	1	0	1	0	1	1	ج
36		01		1			0	1	0	0	11	01	1	0	1	1	1	0	1	1	چ
37		10		0	0	0	1	0	0	0	01	00	1	0	1	0	1	0	1	1	چ
38				1	0		1	1	0	0	11	0		0	1	1	1	0	1	1	چ
39		01				0	0	0		0	11	01	1		1	0	1	0	1		چ
40		01		1			0	1		0		1	1	1	1	1	1	0	1		چ
41		10			0	0	1	0		0	11	0			1	0	1	0	1		ح
42				1	0		1	1		0		0		1	1	1	1	0	1		ح
43	1	01	0	0	1	0	0	0	0	0	0	1	1	0	1		1	0	0	1	خ
44	1	01	0	0	1	0	0	1	0	0	0	1	1	0	1	1	1	0	0	1	خ
45		0 01	0	0	0	0	1	0	0	0	00	00	1	0	1		1	0	0	1	خ
46		0 01	0	0	0	0	1	1	0	0	00	0		1	0	1	1	1	0	0	1	د
47		1		0	1	0	0	0	0	1		1 11	0		0	0	1			0	ذ
48		AA	1	0	1	1	0	1	0	0		1 11	0	0		1	1			0	ذ
49	00	10	0	0	0	0	0	0	1	0	01	0		0	0	1	0	1	0	1	ر
50	0	1	0	0	0	0	0	1	1	0	01	0			0	1	0	1		1	ر
51	1	0	0	0	0	0	1	0	1	1		1 01	1	0	1	1	1	0		0	ز
52	1	0	0	0	0	0	1	1	1	1		1 01	1	0	1	1	1	0		0	ژ
53	1	01	0	0	1	0	0	0	0	0	01	1	1	0			1	0	0	0	ژ
54	1	01	0	0	1	0	0	1	0	0	01	1	1	0		0	1	0	0	0	ژ
55	00	1	1	0	0	0	0	0	1	0	01	0			1		1	0	1	1	ژ
56	00	1	1	0	0	0	0	1	1	0	01	0				1	0	1	0	1	ژ
57	11		0	0	0	0	0	0	1	1	11	01	0	0			1	1	1	0	ژ
58	0	1	0	0	0	0	0	1	1	1	01	01	0	0		1	1	0	1		ژ
59		0	0	0	0	0	1	0	1	1	11	01		0	1	1	1	1	1		ژ
60	0	1	0	0	0	0	1	1	1	1	00	1		1			1	0	1		ژ
61		01	1	1	1	0	0	0	0	0	01	1	1	1	1		1	0	1	1	س
62		01	1	1	1	0	0	1	0	0		1 1	1	1		0	1	0	1		ش
63			0	0	0	0	0	0	0	0	00	00		0			1			0	ش
64		11	0	0	0	0	0	0	0	0	00	00	0	0	0	1	1			1	ش
65	01	10	0	0		0	0		0	0	00	01		0	0	0	1		1		ش
66	01	10	0	1	1	0	0	0	0	0	01	01	0	0	0	0	1		1		ص
67	1	0		1	0	0	0	0	0	0	01	00	0	1		1	1	0	1	1	ص
68			0	0	0	0	0	0	1	1	11	11	0	0	1	1	1	1	1	1	ض
69		01	0	0	0	0	0	0	1	1	11	11	0	0	1	1	1		1	0	ض
70		10		1	0	0	0	0	0	0	11	00	0	0	1	0	1	0	1	1	ض
71		11	0	0	0	0	0	0	0	0	00	00	1	0	0	1	1	1	1	1	ط
72	0	1	0	0	1	0	0	0	0	0	00	11	0	0	0	0	1			0	ظ
73	1	0	0	0	0	1	0	0	0	0	00	11	0	0	1	1	1			1	ظ
74		11	0	0	0	0	0	0	1	0		1 01	0	0	0	1	1		1		ع
75	AA	BB̄	0	0	0	0	0	0	0	0	00	00	0	0	1	0	1	0		1	/
76			0	0	0	0	0	0	0	0	00	00	1	0			0		0	0	.
77	1	0	0	0	0	0	0	0	0	0	00	00	0	0	1	1	0		0	0	:
78	BB̄	AA	0		0	0	0	0		0	00	01	0	0	0					0	؟
79	0	1	0	0	0	0	0	0	0	0	00	00	0	0	0		0		0	0	!

* Symbol dots are not shown. Blanks correspond to don't-care conditions. AA denotes two identical and BB̄ two complementary bits.

Table 4. Description of Farsi symbols in terms of sub-symbols and dots for use in the post-processing phase.

DESCRIPTION	SYM	DESCRIPTION	SYM
01 / 03&D(+>1)	ا	13	د
02 / 04&D(+>1)	ا	14	د
03	ا	13&D(+1)	د
04	ا	14&D(+1)	د
05&D(-1)	ب	15	ر
06&D(-1)	ب	16	ر
07&D(-1)	ب	15&D(+1)	ز
08&D(-1)	ب	16&D(+1)	ز
05&D(-3)	ب	15&D(+3)	ژ
06&D(-3)	ب	16&D(+3)	ژ
07&D(-3)	ب	07*08*54	س
08&D(-3)	ب	08*08*54	س
05&D(+2)	ن	07*08*08	ش
06&D(+2)	ن	08*08*08	ش
07&D(+2)	ن	07*[08&D(+3)]*54	ص
08&D(+2)	ن	08*[08&D(+3)]*54	ص
05&D(+3)	ن	07*[08&D(+3)]*08	ض
06&D(+3)	ن	08*[08&D(+3)]*08	ض
07&D(+3)	ت	17 / 21*54	ط
08&D(+3)	ت	18 / 22*54	ط
09&D(-1)	ج	19 / 21*08	ظ
10&D(-1)	ج	20 / 22*08	ظ
11&D(-1)	چ	17&D(+1) / [21&D(+1)]*54	ع
12&D(-1)	چ	18&D(+1) / [22&D(+1)]*54	ع
09&D(-3)	ح	19&D(+1) / [21&D(+1)]*08	غ
10&D(-3)	ح	20&D(+1) / [22&D(+1)]*08	غ
11&D(-3)	خ	23	ف
12&D(-3)	خ	24	ف
09	چ	23&D(+1)	ق
10	چ	24&D(+1)	ق
11	ه	25	ع
12	ه	26	ع
09&D(+1)	ح	27	غ
10&D(+1)	ح	28	غ
11&D(+1)	خ	25&D(+1)	غ
12&D(+1)	خ	26&D(+1)	غ
27&D(+1)	ت	07&D(+1)	ذ
28&D(+1)	ت	08&D(+1)	ذ
29&D(+1)	ی	55	و
30&D(+1)	ی	56	و
31&D(+1)	ی	57	ه
32&D(+1)	ی	58	ه
33&D(+2)	ق	59	ه
34&D(+2)	ق	60	ه
31&D(+2)	ق	61	ی
32&D(+2)	ق	62	ی
35	ک	07&D(-2)	ی
36	ک	08&D(-2)	ی
37	ک	63	٠
38	ک	64	١
39 / 35&D(+>1)	ک	65	٢
40 / 36&D(+>1)	ک	07*65	٣
41 / 37&D(+>1)	ک	66 / 67	٤
42 / 38&D(+>1)	ک	68 / 69	٥
43	ل	70 / 71	٦
44	ل	72	٧
45	ل	73	٨
46	ل	74	٩
47	لا	75	/
48	لا	75&D(+1)&D(-1)	٪
49	م	76	.
50	م	76&D(+1) / 76&D(-1)	:
51	م	77	؛
52	م	77&D(-1)	؛
53&D(+1)	ن	78&D(-1)	؟
54&D(+1)	ن	79&D(-1)	!

& denotes combination
* denotes (right-to-left) concatenation
/ separates alternatives

D(±X) denotes the number of dots indicated by X above (+) or below (-) the center row
ij refers to Table III entries

33: $D(+2)$

35: $D(0)$

39: $D(0)/D(+>0)$

52: $D(0)$

53: $D(0)/D(+1)$.

Here, $D(+X)$ or $D(-X)$ denotes the number of dots indicated by X above or below y_{ave} and '/' separates

acceptable alternatives. In the case of symbol number 39, the 'dots' actually correspond to the small top stroke, which may appear to be disconnected from the main part of the symbol.

Once the symbols and subsymbols are recognized and the number and place of dots determined, this information must be combined to deduce the actual Farsi characters of the printed text under consideration. Since not all combinations of symbols, subsymbols, and dots make sense, several consistency checks are possible at this stage. Table 4 shows how each Farsi symbol is formed by a combination of subsymbols from Table 3 and dots.

Most special characters have not been included in Tables 3 and 4 since their features were not adequately tested by our implemented system using newspaper headlines as input. However, most such symbols can be easily incorporated into the system by adding appropriate entries in Tables 3 and 4. As an example, we have the following entry for the multiplication symbol, '×', in Table 3:

XX XX 1 1 1 1 0 0 0 0 11 11 0 0 X X 1 X X 0

Here, Xs denote 'don't-care' conditions. In a few cases, special tests, such as that introduced for the symbol pair (20, 52) may be needed for distinguishing such special symbols from those we have defined here (e.g., '−' from symbol number 63).

CONCLUSION

We have shown the practicality of recognizing printed Farsi texts by means of a computer despite several impeding properties of the Farsi script, including the connectivity of symbols. Since connectivity is the rule rather than the exception in printed Farsi, the segmentation of text becomes more important here than it is for other languages.[7, 8] Segmentation of printed Farsi is in effect similar to the segmentation of cursive scripts in many other languages.[9] As a first study in this area,[10] the work reported here can be extended and refined in many different directions.

The experimental system has been designed to work with a variety of common type fonts. Thus Table 3 contains many 'don't-care' entries indicating either that the corresponding features may or may not exist in various type fonts or that the values involved in deciding the existence of that feature are too close for a definite conclusion, due to digitization error. In the second instance, the 'don't-care' entry actually carries some information, as opposed to the common case of 'don't-cares' corresponding to the state of no information. Thus if we consider only one particular type font, not only will we be able to fill in some of the blanks in Table 3, but we can also derive more information from those that remain.

One may look for 'better' or fewer features for the classification stage. A first step in this direction is to decide which of the twenty features presented in the previous section are most useful. To do this, we consider pairs of symbols from Table 3 (a total of 3081 pairs) and find the set of features which distinguish them. We then use the classical covering method of switching theory to obtain the following unique set of minimal features:

$$Q_m\ X_+\ Y_+\ C_l\ C_r\ L_u\ L_d\ C_2\ R_2\ C_T\ C_3\ C_m\ R_W\ R_M\ R_l\ R_\#.$$

The other four features have been retained in our system since they entail no additional storage requirement (space was available in partially used words) and cost very little in terms of execution speed. This second point is obvious for Q_M and C_M and is the result of the single algorithm used to obtain all four kinds of concavity simultaneously.

Thus, in a sense, the features we have used appear to be not far from optimal. However, the feature space being potentially infinite, better features are likely to be found in future as we accumulate experience with the present approach. Systematic selection methods from a large set of features (e.g., [11, 12]) have not been attempted in this study and constitute a possible area for future investigation.

There is a question of which new features to add in the classification phase in order to make the approach more reliable. Obvious choices are suggested by the ones already used, e.g. L_l, L_r, R_3 and R_m defined analogously to L_u, L_d, C_3 and C_m. However, there are many other alternatives. For example, considering nine or even sixteen sections in the subgrid, rather than just the four quadrants, will yield a better picture of the distribution of dark points in the subgrid. The ratio of the number of dark and light points in the rectangle defined by $R_{\min}$, $R_{\max}$, $C_{\min}$ and $C_{\max}$ is another possibility. Taking note of the overlaps which are eliminated in the subword separation phase is also useful, since only certain symbols can cause such overlaps. The number of concavities (not just major ones) in each direction is also a potentially helpful feature due to the curved nature of Farsi script.

However, as human readers can testify, certain pairs of symbols are intrinsically difficult to distinguish regardless of the features used. Some examples are: (03,64), (57,69), (63,76), (76,77). In such cases, the use of context[13,14] can be helpful. In general, by using contextual information, ambiguities can be resolved either completely (e.g. as in limited-vocabulary texts) or probabilistically by considering relative frequencies of various n-grams of Farsi symbols.

Rather than using T as an important decision parameter throughout the recognition process, one may attempt to eliminate its effect by reducing or 'thinning' the symbols to digital arcs[15, 16] by successively deleting external layers of dark points until no point can be removed without disconnecting the arc. Actually, a combination of the two methods may be found most suitable, with some of the features such as the four concavities, loop parameters and R_l extracted prior to the thinning process and other topological features[17] dealt with afterwards.

The insight gained from this study will be helpful in

the design of an OCR type font for Farsi. Here it is possible in most instances to add some features to one symbol from a troublesome pair in such a way as to make them distinguishable without seriously affecting human readability and aesthetic pleasantness of the resulting script. It is interesting to note that neither template matching nor comparison of projection profiles is a suitable technique for Farsi OCR due to extreme variations in Farsi symbol sizes (see Table 3).

Other extensions of this work are also possible. Studies in the human recognition of Farsi symbols, particularly the ones with minor differences[18] can identify useful features for character classification. Certain syntactic approaches appear to be suited to printed Farsi scripts.[19] We have assumed in our experimental system that the lines of text are always parallel to the X axis. In practice, a skew of a few degrees does not cause any problem. Larger skews can be easily detected and dealt with by a skew normalization method.[20] Finally, recognition of handwritten Farsi scripts does not appear to be practical except in highly constrained cases. Even in such cases, difficult segmentation and classification problems will be encountered.

SUMMARY

The automatic recognition of printed Farsi (Persian) texts is complicated by several properties of the Farsi script: (a) connectivity of symbols; (b) similarity of groups of symbols; (c) highly variable widths; (d) subword overlap; and (e) line overlap. In this paper, a technique for the automatic recognition of printed Farsi texts is presented and its steps are discussed as follows: (1) digitization, (2) editing, (3) line separation, (4) subword separation, (5) symbol separation, (6) recognition, and (7) post-processing. The most notable contribution of this work is in the algorithms for steps (5) and (6) above.

In order to understand the algorithms used, it is necessary to introduce a fundamental property of the Farsi script. Farsi font design is done by using a rectangular-tip pen having a length much greater than its width. As the designer moves the pen at certain angles to generate each symbol, lines with varying thicknesses appear. At the unique connection point of two adjacent symbols, the pen moves horizontally on the 'connection axis' and produces a line with (maximum) script thickness. The above property along with the fact that there is no symbol overlap at the connection point is used in the design of the symbol separation algorithm.

The recognition procedure is based on certain geometric properties such as relative width, existence of concavities and loops. In all, 20 geometric features are used for obtaining a 24 bit vector for each symbol to be recognized. The vector thus obtained is matched against templates for the Farsi symbols. In the rare cases when an exact match is not found, the algorithm looks for a best match in which the more reliable features are examined first. The implemented system is capable of storing new templates for each symbol, as they are encountered, in order to improve its performance by learning.

Experience has shown that the minimum acceptable resolution for the digitization hardware is four samples per script thickness, since otherwise many symbols become indistinguishable. This translates into about 100 samples cm^{-1} for Farsi type fonts of typewriters and most textbooks. Practical application of the technique to Farsi newspaper headlines has been 100% successful. However, smaller type fonts, which could not be handled by the coarse digitization hardware used, will no doubt result in less than perfect recognition due to the higher effect of digitization noise.

The technique presented here is also applicable with little or no modification to printed Arabic and Urdu texts which use the same alphabet as Farsi. The insight gained from this study will be helpful in the design of an OCR type font for Farsi, Arabic and Urdu languages.

REFERENCES

1. M. D. Freedman, Optical character recognition, *IEEE Spectrum* **11**, 44–52 (1974).
2. W. Stallings, Approaches to Chinese character recognition, *Pattern Recognition* **8**, 87–98 (1976).
3. B. Parhami and F. Mavaddat, Computers and the Farsi language: a survey of problem areas, *Information Processing* '77 IFIP Congress, pp. 673–676, North-Holland, Amsterdam (1977).
4. B. Parhami, Impact of Farsi language on computing in Iran, *Mideast Comput.* **1**, 6–7, September (1978).
5. B. Parhami, On the use of Farsi and Arabic languages in computer-based information systems. Proc. Symp. Linguistic Implications of Computer-Based Information Systems, Session 3, paper 11, New Delhi, November (1978).
6. Hewlett-Packard Co., Hewlett-Packard Model 91000A Plug-in 20 KHz Analog-to-Digital Interface Subsystem. Cupertino, CA (1974).
7. R. N. Ascher, G. M. Kappelman, M. J. Miller, G. Nagy and G. L. Shelton, Jr., An interactive system for reading unformatted printed text, *IEEE Trans. Comput.* **C20**, 1527–1543 (1971).
8. R. L. Hoffman and J. W. McCullough, Segmentation methods for recognition of machine printed characters, *IBM J. Res. Dev.* **15**, 153–165 (1971).
9. A. K. Dutta, An experimental procedure for handwritten character recognition, *IEEE Trans. Comput.* **C23**, 536–545 (1974).
10. M. Taraghi, Automatic recognition of printed Farsi texts, M.S. thesis, Department of Mathematics and Computer Science, Sharif University of Technology, July (1978) (in (Farsi).
11. A. N. Mucciardi and E. E. Gose, A comparison of seven techniques for choosing subsets of pattern recognition properties, *IEEE Trans. Comput.* **C20**, 1023–1031 (1971).
12. D. E. Troxel, Feature selection for low error rate OCR, *Pattern Recognition* **8**, 73–76 (1976).
13. R. W. Ehrich and K. J. Koehler, Experiments in the contextual recognition of cursive script, *IEEE Trans. Comput.* **C-24**, 182–194 (1975).
14. W. Doster, Contextual postprocessing system for cooperation with a multiple-choice character recognition system, *IEEE Trans. Comput.* **C-26**, 1090–1101 (1977).
15. A. Rosenfeld, Arcs and curves in digital pictures, J. Ass. comput. Mach. **20**, 81–87 (1973).

16. B. Parhami, An introduction to the geometry of digital pictures, Proc. 9th National Mathematics Conf., Esfahan, Iran, March 1978 (to appear).
17. J. T. Tou and R. C. Gonzalez, Recognition of handwritten characters by topological feature extraction and multilevel categorization, *IEEE Trans. Comput.* **C21**, 776–785 (1972).
18. B. A. Blesser, T. T. Kuklinski and R. J. Shillman, Empirical tests for feature selection based on a psychological theory of character recognition, *Pattern Recognition* **8**, 77–85 (1976).
19. H. F. Feng and T. Pavlidis, Decomposition of polygons into simpler components: feature generation for handwritten character recognition, *IEEE Trans. Comput.* **C24**, 636–650 (1975).
20. W. C. Naylor, Some studies in the interactive design of character recognition systems, *IEEE Trans. Comput.* **C20**, 1075–1086 (1971).

About the Author—BEHROOZ PARHAMI was born in Tehran, Iran, in 1947. He received the B.S. degree in electrical engineering from University of Tehran in 1968, and M.S. and Ph.D. degrees in computer science from Oregon State University and University of California at Los Angeles in 1970 and 1973, respectively.

He is currently an associate professor of computer science in the Department of Mathematics and Computer Science, Sharif University of Technology, Tehran, where he served as the department vice-chairman from 1977 to 1978. Previously, between 1971 and 1974, he held the positions of research assistant, post-graduate research engineer and acting assistant professor with the Computer Science Department of the University of California, Los Angeles. He has also served as consultant to a number of private and governmental organizations since 1972.

His research interests are in the areas of computer architecture, fault-tolerant computing, software engineering, computer science education, and transfer of informatics technology, including language-dependent aspects of computer applications in Iran. He is the author or co-author of forty technical papers and three Farsi textbooks in computer science as well as numerous articles, technical reports, translations and critical reviews.

Dr. Parhami is a senior member of the IEEE and has been the chairman of the IEEE Iran Section since 1977. He is a member of the Association for Computing Machinery (ACM), the British Computer Society, and the Iranian Mathematical Society. He is also the founder of Informatics Society of Iran (ISI) and has served as its president and publications committee chairman since 1979.

About the Author—MAHMOOD TARAGHI was born in Tehran, Iran, in 1952. He received the B.S. degree in physics and the M.S. degree in computer science from Sharif University of Technology, Tehran, in 1975 and 1978, respectively.

He is currently a system designer with the Data Processing Center, National Iranian Radio and Television. He is a member of the Informatics Society of Iran.