Abstract—Persian (Farsi) handwriting is inherently cursive and variable in style. Moreover, many Persian characters have similar body but different secondary strokes e.g. dots. As a result, Persian handwriting recognition is an extremely complex task. Many researchers have tackled the problem by different approaches; however, the research in this field is still in its infancy. In this paper, we propose a novel hierarchical system for recognition of Persian words. As a preliminary step, words are divided into subwords based on pen-up and pen-down information. Then, a three level fuzzy recognition method is applied to recognize each subword and each word is recognized by merging the recognition results of its subwords. In the highest level of subword recognition, subwords are divided into characters using a backtracking method with pruning. In the mid level, each character is recognized using a dynamic programming approach. In the lowest level, the segments of each character are classified. The result of each level is used in its above level. Fuzzy if-then rules and fuzzy inference is used in all three levels. Given the segments of each word, fuzzy feature vectors of the segments and the fuzzy-rule base for character recognition, the proposed method is hierarchically optimal, i.e. the system finds a word with maximum membership with respect to the hierarchy we used. Our preliminary experiments with the proposed system show satisfactory results.

Keywords: Persian handwriting, word recognition, fuzzy, dynamic programming, hierarchical.

I. INTRODUCTION

Handwriting recognition is an important field of research which involves the conversion of hand printed input to its equivalent text. This procedure changes the input from a meaningless vector or image to a set of characters which can be stored, searched, sorted, etc. easily by a computer.

Handwriting recognition can be divided into two categories: online and offline. In online handwriting recognition, the phases of writing and recognition happen simultaneously, while in offline recognition the writing phase happens prior to recognition. The input to an online system is a vector of points usually received via a light pen or mouse, so that the relative timing information of the points is available to the system. On the other hand, the input of an offline system is an image of the words or letters which is typically received by a scanner or camera.

The applications to such a system are numerous. Light pens can replace keyboard in today's small-size computers such as Personal Digital Assistants (P.D.A.s) and cell phones. Online handwriting recognition is also of benefit to inexpert users who prefer writing by hand to typing. Offline handwriting recognition is most useful in processing forms like postal addresses, cheques, etc. and also in digitizing old documents.

Persian (also called Farsi), Arabic and Urdu have very similar handwriting which is complex from the recognition point of view. The complexity in recognition of Persian script arises from the fact that the handwriting is fully cursive and very variable in style. In addition, each character has up to four shapes depending on its position within the word, and also different characters can have similar body but distinct secondary segments like dots. As an example, in figure 1, each row provides some samples of printed Persian characters (one or more) with same body and different secondary segments. The

This research is supported by Iran Telecommunication Research Center (ITRC).
equivalent characters are shown beside each. In the fourth row a single body (⦿) called “Dandaneh” (which means tooth and named so as a result of shape similarity) is shown to have six different secondary segments resulting in six different characters. A sequence of length three of the same body can lead to completely different character as seen in fifth row. The situation is much worse in handwriting of the characters below as each body and secondary segment can be written in many other ways.

As a consequence of the similarity between the handwriting of the above languages, a huge population of users in the world applies the handwriting. Any recognition method working efficiently on one can be useful for others as well. However, despite the large number of users, the research progress in Persian handwritten word recognition up to now is insignificant, especially in online category. The difficulty of problem forced many researchers to limit their work on isolated letter recognition as a preliminary step toward word recognition. Such approaches can be useful in cursive script recognition if an efficient segmentation method exists.

The main contribution of this paper is introducing a novel four-level method for online Persian word recognition which is proved to find a hierarchically optimal answer and its time complexity is not dependent on the lexicon size. This independence from lexicon size is due to the fact that only characters are modeled and recognized, not each individual word as in many other methods. The Persian characters are identified to our system by a set of fuzzy rules. A four level recognition-based segmentation cuts the input word to its building characters.

The remainder of this paper is organized as follows: In section 2, some related works in the field of Arabic handwritten word recognition is reviewed. In section 3, the structure of the system and the recognition method used is introduced. Finally, the experimental results are presented in section 5.

II. RELATED WORK

In this section we briefly review some works done on online recognition of Persian-family words. Offline methods or works on recognition of isolated characters are not discussed in this paper.

In 1982, Amin developed a system called IRAC II for recognition of online Arabic words [1]. This early system uses a structural method in which words are firstly segmented into characters and then recognized using an isolated character recognition approach. His system was tested on 400 Arabic words and obtained a recognition rate of 80%. In later versions, IRAC system also employed other techniques for Arabic handwriting recognition [2-3].

In 1990, Al-Emami et al. suggested a method based on decision tree classification [4]. Their method consists of preprocessing and classification phases. In preprocessing, the input is segmented into building blocks of a nearly constant slope based on a threshold on the current length and angle of the segment. Next, the segmented parts are encoded to four direction codes representing each main direction. In the learning phase, each Arabic letter with all its variations is preprocessed with the mentioned method. Then the rules for each segment are stored in a decision tree. The recognition of a cursive Arabic word is completed in three steps. First of all, in a preprocessing step the input is segmented and encoded into direction codes. Next, the direction codes of the first letter of the word enter the decision tree to be recognized. As the number of codes of the first letter is not known in advance, a maximum of eight segments is considered for recognition. If these first eight codes did not match any character pattern, the last segment is discarded and the first seven segments are processed. This process continues until a match is found. If none is found and only one segment is remained, it will be regarded as an extra connection part and skipped. To evaluate the recognition accuracy, four words consisting of 10 Arabic letters are selected for training. Each word is written three times by ten Arabic writers generating a total of 120-word training data. The experimental results show that recognition rate for the training data is 100%. Test data is composed of 50 new words consisting of the letters appeared in training data, leading to 86% recognition rate. Another experiment shows that a 100% recognition rate is achieved if the writer is trained to adapt to the weaknesses of the recognition system.

In 2005, Razavi et al. [5] proposed a method for recognition of Persian Online subwords. In their method, secondary segments of the subword and their relative position with respect to the body are determined. According to this information, a set of characters with consistent properties is selected. If the set has only one member, that sole member is chosen as output. Otherwise, the subword is compared with a dictionary of consistent words. The subword within dictionary with minimum distance is returned as output. A total test set of 11092 subwords written by 124 writers and a 1000-word dictionary is used for experimental study. A recognition rate of 74.95% is reported for this data set which can be improved to 97.87 if the ten nearest subwords are considered as outputs. The recognition rate of this method is highly dependent on how accurate the secondary strokes are recognized.

Halavati et al. [6] in 2007 developed an elastic fuzzy pattern recognition approach to recognize online Persian words. In their method, the input and word patterns are described with fuzzy linguistic variables. Then the input is compared with each word pattern using a two-level approach. In the low level, an input segment is compared with a word segment using a look-up table which is filled manually. In the high level, a nondeterministic state machine is used to recognize a sequence of input segments with alphabet patterns. In this research, a set of 1250 Persian words written by 20 writers is used for
experiments. This paper has no solution for recognition of secondary segments. So these parts are eliminated from the test data. The recognition rate is 78% when no dictionary is used and 96% with a dictionary. In this method the time complexity and also the precision of recognition is dependent to the lexicon size i.e. the number of words that the system can recognize and the similarity between them.

III. SYSTEM STRUCTURE

In this section, we describe the structure of our word recognition system. The input to the whole system is an online Persian word, received by the input device (e.g. an optical pen). The system output is the characters inside the word.

First, in a preprocessing step, we prepare our input for recognition. Most of the preprocessing operations are from Soleymani et al. in recognition of isolated Persian characters [7]. An online Persian word is a sequence of points in the plain. Each point whose distance to the previous accepted input point is less than a threshold is discarded. Then, in a step called over-segmentation, we divide the input to a sequence of some units which are smaller than a character, called segments. Segments are geometrical units like curves, lines, closed loops, etc. The goal of over-segmentation is to provide a means of representing both the character patterns and the input words. Segments are selected to be smaller than a character because segmenting based on geometrical properties is much easier than segmenting the word into letters. In later stages, segments are combined to form a character, characters combined to form a subword, and subwords combined to form the output word. After preprocessing, the input is transformed to a sequence of segments (or their corresponding fuzzy feature vectors), which is ready to be used for recognition.

In the preliminary stage of recognition, words are divided into subwords based on pen-up and pen-down information. This step is a trivial process used in almost all online methods. The next step is to recognize the subwords. We propose a three level backtracking-dynamic-fuzzy recognition method for recognizing Persian subwords. The highest level uses backtracking to divide the subwords to their consisting characters in a way that it achieves the highest possible confidence in recognition. The mid level uses dynamic programming to match a subpart of input (i.e. a part of input that we expect to denote a character and determined by backtracking level) with alphabet characters in an optimal way. In the lower level, each input segment is compared with a character segment using a fuzzy inference approach. Figure 2 illustrates a general view of the system.

IV. WORD RECOGNITION USING A HIERARCHICAL APPROACH

As stated in previous section, in the preliminary stage of recognition, words are divided into subwords based on pen-up and pen-down information. Each pen-up followed by a pen-down indicates the end of a subword. A pen-up with no pen-down indicates the end of the last subword. As an example, the three subwords of the Persian word “Farsi” are shown in figure 3.

Next, the Persian subwords are sent to the three levels recognition module. When the lower levels recognized all subwords, the final word is constructed by merging these results. Language specific knowledge, e.g. knowledge extract from a dictionary, can be embedded and used in this step in order to prune or improve the results. Currently, however, we do not use any language specific knowledge.
In the next stage, Persian handwritten subwords are recognized using a three-level approach. Each level is described in detail as follows.

A. Higher Level

In the highest level, we try to divide an input subword into its characters using backtracking. In this stage, it is assumed that the set of character boundaries is a subset of segment boundaries. So a backtracking algorithm selects the best subset of segment boundaries by systematically traversing all possible solutions in the search tree. Useful bounds are used to avoid visiting unpromising states.

Assume we cut an input subword in arbitrary positions. We name each piece a subpart. It is important to distinguish subparts from segments which we defined in section 1. Segments are well-defined geometric pieces smaller than letter, while subparts are created by arbitrary cuts and are consecutive subsequences of subword segments sequence (in other words, by putting together the segments of the subparts, we reach the segments sequence of the input subword). With this definition in mind, our goal in this level would be to adjust the subparts in a way that each contains exactly one character. The only constraint on our cutting positions is that none is allowed to lay inside a segment, which its necessity can be deduced from the assumption in the previous paragraph.

We fulfill this desire by systematically generating all possible cuttings, by backtracking on all segment boundaries. After each cutting generated the corresponding subparts (i.e. the subsequence of segments), the best way of matching each subpart with a character is found in the mid level using a dynamic-programming approach. In this way, the best possible characters for the current cutting are diagnosed. Each character is accompanied with the fuzzy membership of the subpart to the character class (this membership is returned by mid level). Then, the fuzzy membership of the input to a subword class is calculated by implying a T-norm (like minimum or multiplication) on the character memberships. In backtracking, we find a cutting with maximum total T-norm value.

This can be viewed as a fuzzy inference with rules that infer the subword from its character. A sample rule which describes the input based on its subparts is shown below:

**IF** `subpart1` = "و" **and** `subpart2` = "د" **and** `subpart3` = "پ" **and** `subpart4` = "سلام" **THEN** `input` = "سلام"

In the backtracking phase, we create the word with highest membership, assuming that the lower levels give us the optimal matching. Thus, this level is optimal by assuming the levels below are optimal (which will be shown below). Therefore, the three-level recognition module is hierarchically optimal.

B. Mid Level

In the mid-level, dynamic programming is used to find the best way to match the current subpart (i.e. a subsequence of segments of a subword, which we call it current sequence) with a character pattern. We do this for all character patterns. When this process is done for all characters, the character in which the subsequence has the highest membership is returned as the result of subpart recognition to the Higher Level. A character pattern has some sub-patterns and is defined by fuzzy rules such as:

**IF** `segment1` = "ذ" **and** `segment2` = "ص" **and** `segment3` = "ی" **THEN** `subpart` = "یمو"

It is clear that we must find all sub-patterns of a character pattern in the same order as they appeared in the rule. We want to assign exactly one segment to each sub-pattern such that the overall membership of this assignment is maximized. If the number of segments in current sequence is less than the number of sub-patterns, this goal is not achievable. If these two numbers are equal, there exists exactly one assignment, which is trivial. Otherwise, there exists more than one assignment, the best of which we are seeking for. It should be noted that since there may be exponential number of valid assignments in Mid Level and we have several calls of Mid Level in the Higher Level, checking all the assignments to find the best one in the Mid Level is infeasible due to inappropriate running time. We use dynamic programming to do this task efficiently.

The goal of dynamic programming is to match the current sequence of segments with the character sub-patterns. It is possible that some segments of the current sequence are left unmatched in order to handle the low quality of segmentation (i.e. discarding meaningless segments). So this process selects only the necessary segments of the subpart to match with all segments of the character pattern in an optimal way.

Assume the character has `n` sub-patterns and the current sequence has `m` segments. For all `1 ≤ i ≤ n` and `1 ≤ j ≤ m`, we define `best(i, j)` to be the...
maximum membership degree of the first \(j\) segments of current sequence with all the first \(i\) sub-patterns of the character pattern. It is clear that the best assignment is equal to \(best(n, m)\). Then we have

\[
best(i, j) = \begin{cases} 0 & i > j \\ \max\{best(i, j-1), best(i-1, j-1) \otimes \mu_{T_i}(j)\} & 1 \leq i \leq j \end{cases}
\]

where \(\mu_{T_i}\) shows the membership function of the \(i^{th}\) sub-pattern, \(T_j\) is the feature vector of the \(j^{th}\) segment of input subpart, and \(\otimes\) is any fuzzy T-norm.

If \(i > j\) then there are not enough segments in the input and so the matching value is zero. If \(i \leq j\) then two cases may happen. We find the best assignment in each case and choose the better one. The \(i^{th}\) sub-pattern may match with the \(j^{th}\) segment or may not. If it does not, we must match the first \(j-1\) segments of current sequence with all the first \(i\) sub-patterns of the character pattern in an optimal way, which is defined by \(best(i, j-1)\) (see the first term of maximization in the above formula). If it does, it remains to match the first \(j-1\) segments of current sequence with all the first \(i-1\) sub-patterns of the character pattern in an optimal way and combine the result with assignment of \(j^{th}\) segment to the \(i^{th}\) sub-pattern, which is the second term of maximization in the above formula. No other cases can happen, which shows the intuition behind the above formula.

C. Lower Level

In the lowest level, each segment of the subpart is compared with a sub-pattern of a character pattern using fuzzy inference which results in the value \(\mu_{T_i}(j)\), required by dynamic programming level. The fuzzy rules in this level are as below:

**IF direction=SouthWest and size=Small and type=NearLine Then segment=“\text{\textbullet}”**

In premise of the rule, the features are assigned linguistic values and in the conclusion the segment type is announced. An input segment can be classified by these rules using a fuzzy inference mechanism like Mamdani’s min-max inference. The result of classification which is the type of segment and the membership degree of it is passed to the level above (Mid level).

V. EXPERIMENTAL RESULTS

In this section, we present our preliminary experimental results of proposed fuzzy handwriting recognition system. Since this system has many parts, it is still under construction. Before reporting the final results, we are still working on our preprocessing module and completing our fuzzy rule base. Currently, we performed some experiments with words of limited character set. Our experiments show that:

- The system can correctly recognize the border of characters and subwords are correctly divided into characters by using the backtracking module (Higher Level).
- The system has enough flexibility to deal with meaningless segments by using dynamic programming module (Mid Level). We observed that the system tends to recognize characters with less number of sub-patterns to reach greater membership values. This tendency, sometimes, causes incorrect recognition. Thus, we added an additional constraint to force the system match at least an acceptable portion of a subpart with a character pattern. This additional constraint is almost solved the problem, but we are still seeking to use more advance methods (like introducing a penalty term in our dynamic programming formulation) for solving this problem.
- As it is evident, the system is greatly dependent to the quality of fuzzy rule base. When we used more accurate rules for a limited character set, our recognition quality greatly improved. This shows that to have an effective rule base for recognition of all Persian characters, we need a large amount of rules. Since constructing the rules by human experts are not an easy task, we are seeking to add an automatic rule generation module (like a neuro-fuzzy module) to our system.

CONCLUSION

We proposed building a novel hierarchical system for recognizing Persian words. In the proposed approach, the input words are recognized in four levels. In the preliminary level, words are divided into subwords. In the next level called the Higher Level, subwords are divided into characters using backtracking. Then, in the Mid Level, the characters are recognized using dynamic programming. In the lower level, the segments of each character are recognized. In all three levels of subword recognition, the same mechanism of fuzzy inference is used. Mamdani’s Min-Max mechanism of inference is our suggestion, while other fuzzy inference mechanisms are also possible to be used. The impact of using different fuzzy inference mechanisms can be a subject of an interesting future study. Another area of study is to develop effective bounds for our backtracking module to avoid it from extending impermissible states of the search space. Bounds on minimum and maximum character length can be the most trivial ones. Finally, it should be noted that our experimental results are still in preliminary stages of development, which will be completed in our future works.
ACKNOWLEDGMENT

The authors would like to thank Mr. Babak Behsaz for his valuable discussions and suggestions, and Ms. Mahdieh Soleymani Baghshah for her supports in initiation of this research.

REFERENCES