A FUZZY LOGIC-BASED NEW METHOD FOR RECOGNIZING HANDWRITTEN PERSIAN NUMBERS

Navid Samimi Behbahan*

Amin Samimi Behbahab*

Milad Samimi Behbahan*

Abstract

Recognizing handwritten numbers can have many uses such as sorting mailers according to their P.O. Box number, automatic office books, sorting bank drafts, and so on. The present study is concerned with proposing a new method based on a blend of fuzzy logic, genetic algorithm, and simulated annealing, to process handwritten data sets. To recognize patterns of numbers with various sizes, lengths, and shapes, pre-processing will be an inevitably necessary phase. As a result, two stages of pre-processing, namely correctingslant characters and thinning them were used. The framing method was used to extract the specifications, whilegenetic algorithm was utilized to select effective features.The proposed method was tested on MODARES database, as a resource approved by researchers, while 96.67of the recognized cases were found to be correct.

Keywords: Genetic algorithm, simulated annealing algorithm, recognizing handwritten numbers, framing method, fuzzy logic

* Department of Computer Engineering, Omidiyeh Branch, Islamic Azad University, Omidiyeh, Iran

1. Introduction

Recognizing handwritten numbers and characters has been one of the interesting topics for those concerned. Some studies have also focused on recognizing handwritten Persian and Arabic numbers and characters. Darvish and collegues used shape-match algorithm for recognizing Persian handwritten numbers. In such a method, for each point on the sample of shape connector, there is a location distribution indicator determining the location of other pointsof the connector [7]. In another study, Parvin and colleagues proposed a method for enhancing the functioning of the recognition system [1]. Alizadeh and colleagues developed a genetics-based method for devising a set of neural networks through a weighted resorting selection method [2]. Shahabi and Rahmati, too, used Gabor filter banks which are designed for Persian handwriting and the visual system [3]. In another study carried out by Parvin and colleagues, paired dividers were used to reinforce this type of dividers. Through increased precision, the rate of error can be decreased in the specification space [4]. In a study, Biz transformational and categorical characteristics were used to recognize Persian handwritten characters [8]. Masrrori made use of dynamic time complexity to recognize digits [9].

The present study is founded on the *if-then*fuzzy rule with definite languagevalues dynamically separated. To categorize datasets, primarily fuzzy rules are produced randomly. Given the nature of the education paradigm and algorithm of genetics, the rules are constantly transformed. Rules with enhanced levels of recognition always make use of simulated annealing. The fuzzy-based method here includes 6 stages:

1. Develop primarily overlapping borders for various language values
2. Select effective features through the genetic algorithm
3. Produce fuzzy *if-then* rules randomly at the first phase
4. Recognize the class and the degree of rules' certainty
5. Create a new generation of rules through genetic algorithm
6. Select generations of rules likely to categorize more cases

In the following sections, the second phase as a pre-processing performed on the data, as well as framing method used to extract specifications will be explained. Next, section 4 will be concerned with the method for selecting effective features, while section 5 explain fuzzy logic the way it was practiced here. Section 6 will provide some insights into how fuzzy rules can be enhanced through synthesizing genetic algorithms with simulated annealing. Finally, the results will be discussed in section 7.

2. Pre-Processing Phase

Pre-processing involves three stages, namely equalizing digit sizes, slant correction, and thinning. Equalizing digit sizes can be accomplished by justifying all of the digits into 42 rows and 32 columns of pixels [10]. The next two stages are reviewed below.

2.1. Slant Correction

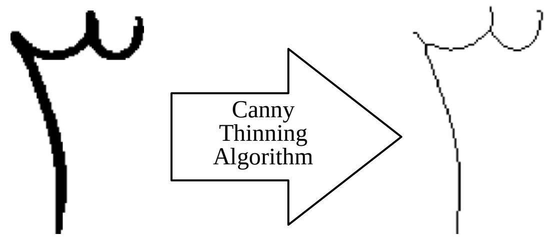
Handwritten letters, depending on individuals' style of writing, slant toward a particular direction. To solve this problem, each letter is divided into an upper and a lower part, and then the center in which most pixels occur should be determined in each of the halves. The slant line connecting the two parts will show the slant of the letter. This correction can be made through use of Conversion 1.In this conversion, *def*represents normal slant, x, y are the coordinates of the pixels before correction and help de-slant characters.y' and x' represent the same points after correction, s is slant of the connecting line between the two upper and lower parts of accumulated pixels [10].

$$X' = x - y \tan(s - def)$$
$$Y' = y \qquad (2)$$

2.2. Thinning

At this stage, the *canny* algorithm is used to narrow and find the skeletonized outline of each digit, without causing any blurring or breakage in the digit. This algorithm will result in a skeleton preserving the original shape of the digit and producing real (not fake) data. Figure 1 illustrates a processed sample. The output at this stage is just ainterconnected outline turned into the standard size 42*32.

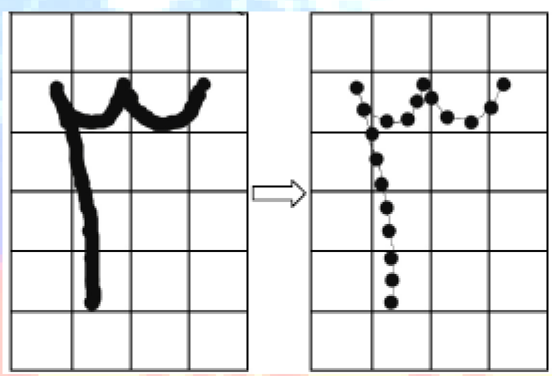
Figure 1: A thinned sample



3. Framing method

To extract features in the present study the framing method is used. The main advantage of this method is its resistance against minor changes, its facile implementation, and its high power of recognition. In this method, a binary picture sized 42*32 is converted into an array sized 6*4. As a result, the array will be composed of 24 fields sized 7*8. Figure 2 illustrates how number 3 is structured in the array. Each of the fields in the array is called a "frame".

Figure 2: A hypothetical array for extracting features



Each field provides 2 features (relation 2 and 3): first, the average of the distance between the on pixels in each frame from the start of the image (down left side, y_b), and second, the average of pixels angles from horizon (a_b). Then, there are 24 frames and 48 features which define each digit (Table 1). $n_b$represents the number of on pixels in the b box,d^b_ks is the distance of each the pixels from the starting point, and θ^b_k shows the angle of each of the pixels from the horizon [10, 11].

$$\gamma_b = \frac{1}{n_b}\sum_{k=1}^{n_b} d_k^b \,, \qquad b = 1,2,\dots,24 \tag{1}$$

$$\alpha_b = \frac{1}{n_b} \sum_{k=1}^{n_b} \theta_k^b \,, \qquad b = 1,2,\dots,24 \tag{2}$$

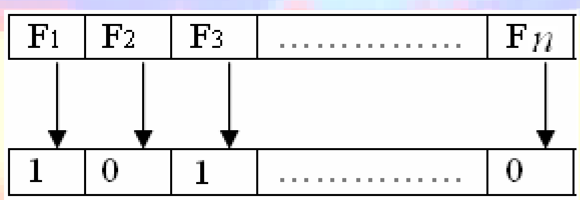
Table 1: Extracted features

:	:	:	:
:	:	:	:
BOX21	BOX22	BOX23	BOX24
$(\gamma_{21},\alpha_{21})$	$(\gamma_{22},\alpha_{22})$	$(\gamma_{23},\alpha_{23})$	$(\gamma_{24},\alpha_{24})$

4. Selecting Distinctive Features through the genetic Algorithm

As was explained above, each digit is composed of 48 features. In the proposed model, through the use of genetic algorithm, fitting features are selected from among all features to be categorized as samples. For these features, binary chromosomes with a length equal to the number of the features in the dataset were defined. If a gene equals 1, it is used, and if it equals 0, it is not used in the categorizations (Figure 3).

Figure 3.Illustrating a chromosome and the way its features are selected



To establish the first population of the genetic algorithm, binary chromosomes are randomly selected. After each chromosome is tested, its fitness (the percentage of recognizing experimental data) will be then calculated. The way the calculation is accomplished is explained below.

The crossoveroperator, producing a random number between 1 and n (n is the number of the features in the dataset), and substituting the value 2 of the chromosome (from that point), produces new chromosomes. The mutation operator, too, is processed through producing a random number between 1 and n and changing the value of the gene from 1 to 0 or vice versa. Producing the next generation will be accomplished by randomly selecting70% of the previous generation and a 30% (random) crossover with the parents with more fitness. The mutation operator occurs with 20% probability, while the crossover operator with 80%.

Distinctive features can be found through implementing the genetic algorithm on some generations and selecting suitable chromosomes. The genetic algorithm processing particularly for this task can be very time-consuming. After each chromosome is produced, its fitness (ability to recognize) should be determined.

The degree of fitness is thus found: for each given chromosomes the system should be educated and finally calculate its degree of recognition, while the value reached at through the process will be the chromosomes fitness. The process should be performed for all of the chromosomes existing in each generation, which can be extremely time-consuming. Yet, after producing several generations, the number of the features will start to decrease and the magnitude of the problem will decline. The speed of categorizing in the testing stage will increase while making errors will remarkably decrease.

The genetic algorithm guarantees to select the most effective features and use the most optimal mode. Still, the algorithm is really slow and time-consuming.

5. Fuzzy Rule-Setting System

In this section, the way knowledge is illustrated in terms of fuzzy rules and categorizing the sets is reviewed. Then the fuzzy model used here, the setting of primary rules, determining classes and certainty degree for each rule will be presented. Following that, the procedures for assigning a class for a new case and the fuzzy argument used here will be investigated.

5.1. The Problem of Categorizing the Patterns

Categorizing the patterns is a problem with: *n* dimensions, *c* classes, and *m* educational patterns ($x_p = (x_{p1}, x_{p2} \ldots x_{pm})$, $p = 1,2, \ldots m$). Each educational pattern is normalized through a number between 1 and 0. That is, the pattern space for each feature is independently a number between 1 and 0. In the present study, *if-then* rules, which are based on categorizing systems, are used.

$$\text{Rule } R_j : \text{if } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{jn}$$
$$\text{Then class } C_j \text{ with } CF_j \; , \; j = 1,2,\ldots,N \quad (4)$$

R_j: j-rule of fuzzy *if-then*

$A_{j1\ldots}$ $A_{j1:}$ language values such as *large*, *small*, *and medium*, each of which having a range of numbers between 1 and 0 and having overlap.

$C_{j:}$ the class obtained for each pattern.

N: the total number of fuzzy rules.

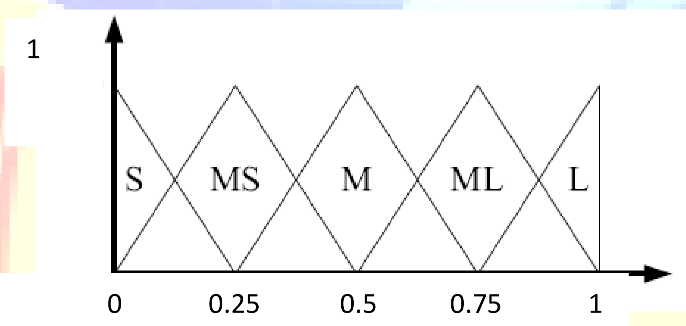
$CF_{j:}$ the degree of certainty of the R_j rule

A set as such is used as a primary fuzzy set belonging to triangular fuzzy sets [5, 6].

5.2. The Fuzzy Model Used in the Study

As Figure 3 shows, the fuzzy set used in the present study is a 5 element fuzzy set.

Figure 4: Fuzzy model have been used



To encode any of these areas, a number is assigned.

S (small) = 1

M (medium small) = 2

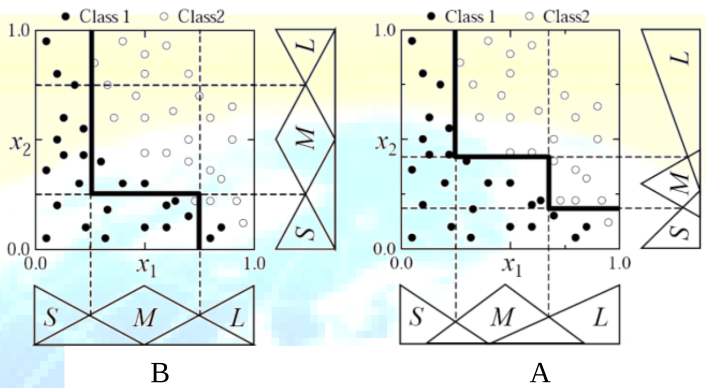
M (medium) = 3

ML (medium large) = 4

L (large) = 5

Of course, these are primary values used to delineate language values. Through the use of the genetic algorithm, attempt is made to continually produce new overlapping sets for these 5 areas, in such a way that an optimal mode can categorize with the maximum recognition rate. Figure 5 shows the degree of accuracy in categorizing for dynamic and static bordering.

Figure 5: In A the bordering of language values is dynamic while it is static in B



5.3. Developing Primary Fuzzy Rules

Each rule is encoded through a set of values between 1 and 5. At first, a set of 100 rules, each of which being a set as long as the number of features in datasets, are randomly produced and their degree of recognition is determined.

In many similar works, primary rules directly emerge from educational data. Although this method is appropriate as far as time complexity is concerned,it is highly effective for educational data because the rules have not been established through sound education. Thus, primary rules would be better to be produced randomly and constantly improved [5, 6].

Desired calculation and the degree of certainty of each fuzzy rule for optimal rules in a rule-governed fuzzy categorization system will be explained below. For instance, to determine the class (C_j) and the Degree of certainty (CF_j) belonging to Rule j, the following procedures should be followed:

Step 1.Calculating $\beta_h(x)$ for class h (h = 1,…, c)

$$\beta_h(R_j) = \sum_{x_{p\in Class h}} \mu_{j1}(x_{p1}) \times \ldots \times \mu_{jn}(x_{pn})$$

$$h = 1,2,\ldots,C \qquad (5)$$

Step 2.Finding the class ($\hat{h}$) with the maximum number of β_h (R_j)

$$\beta_{\hat{h}}(R_j) = \max\{\beta_1(R_j), \beta_2(R_j), ..., \beta_C(R_j)\} \quad (6)$$

If the number of the classes containing the maximum numbers exceeds one class, R_j rule cannot be assigned to C_j class. In a case like this, the class of the given rule will be null ($C_j = \phi$). If there is only one class with maximum numbers, then C_j = *Class h.*

Step 3. If there is only one class with maximum numbers for $\beta_h(R_j)$, the degree of certainty of Rule $j(CF_j)$ should be calculated through formula 7:

$$CF_j = \frac{\beta_{\hat{h}}(R_j) - \overline{\beta}}{\beta_h(R_j)}$$

$$\overline{\beta} = \frac{\sum_{h \neq \hat{h}} \beta_h(R_j)}{C-1} \quad (7)$$

5.4. Fuzzy Arguments

Through the rule-producing procedures mentioned above, *N* number of randomly produced fuzzy rules can be set. Then, the rule's relevant class and its degree of certainty can be determined for all of the *N* fuzzy *if-then* rules. The class of a new case (x) can be discovered through the following formulas [8, 9]:

Step 1.Calculating $\alpha_n(x)$ for class *h* (h = 1,…, c); as a result, we have:

$$\alpha_h(x) = \max\{\mu_j(x) \times CF_j \mid C_j = Class\, h,\ \mathrm{j} = 1,2,..., \mathrm{N}\}, h = 1,2,..., C \quad (8)$$

In which

$$\mu_j(x) = \mu_{j1}(x_1) \times ... \times \mu_{jn}(x_n) \quad (9)$$

Step 2.Finding a class for h_p^* with maximum number of $\alpha_n(x)$

$$\alpha_{h_p^*}(x) = \max(\alpha_1(x), ..., \alpha_C(x)) \quad (10)$$

If the number of the classes containing the maximum numbers exceeds one, the *x* cannot be categorized. Otherwise, the class h_p^* will be assigned to *x* [5].

In this method, genetic algorithm is used only to produce new rules, which are always produced out of already existing rules.Genetic algorithm, the way it is used in this method, will be discussed below.

6. New Rules and Their Improvement

In this study, a synthesis of genetic algorithm and simulated annealing has been used to produce new rules and improve them. Through genetic algorithm, existing rules are transformed and new rules emerge, while simulated annealing can help avoid local extrema.

6.1. Genetic Algorithm

Genetic algorithms can be viewed as a random directed optimization method that gradually moves toward the optimal point. Comparing the specifications of genetic algorithm with other ones, one can introduce them as an algorithm without any information about the problem and without any limitations to the type of variables. Because of such characteristics, this algorithm is reliable and clearly efficient in finding the relative optimal. This method is capable of solving complicated optimization problems as opposed to classic methods which are either unreliable or uncertain in finding the optimal point.

6.1.1. The Structure of Genetic Algorithms

Genetic algorithms are generally composed of the following components:

- ✓ **Chromosomes**: in genetic algorithms, each chromosome represents a point in the search space and a possible solution for the problem at hand. The chromosomes (solutions) are composed of a certain number of genes (variables). To illustrate chromosomes, depending on the number genes (features), there are integer numbers ranging from 1 to 5.
- ✓ **Population**: a group of chromosomes form a population. As genetic operators affect the population, a new population emerges with the same number of chromosomes. The population of the proposed model is primarily 100.
- ✓ **Fitness function**: to solve every problem through genetic algorithm, first a fitness function should be proposed. For each chromosome, the function returns a non-negative number showing

the fitness or capability of the chromosome. The degree of certainty of each rule is considered a fitness function.

6.1.2. Genetic Operators

Genetic operators are as follows:

- ✓ Selection: selecting 2 good and 2 bad rules from among the existing rules according to their fitness
- ✓ Crossover: applying an even crossover operator to 2 goodrules to substitute selected rules.
- ✓ Mutation: applying an even crossover operator to 2 appropriate selectedrules and changing the values of these rules randomly to produce new rules.
- ✓ Substitution: substituting 2 new rules produced by crossover and mutation operators with 2 other rules selected because of their low fitness.

In the selection operator, form among 100 rules, 20 good and 20 bad rules are selected through random functions. Through the use of crossover and mutation operators, rules with high fitness are transformed and result in new rules.

6.1.2.1. Crossover

This operator randomly substitutes corresponding genes belonging to the 2 selected chromosomes (parents) from a number of points, producing 2 new rules, and replacing 2 rules with low fitness.

6.1.2.2. Mutation

The difference between this operator and crossover lies in the fact that instead of previous values, the selected genes of the 2 chromosomes (parents) are given a random value between 1 and 5, and the 2 new chromosomes replace 2 other ones with low fitness.

The major problem of the genetic algorithm in solving problems is falling in trap called *relative maximum cases*. This problem in some cases prevents the system from reaching optimal answers (absolute maximums) for problems. This problem is caused because the genetic algorithm is always tying to improve the problem-solving environment, although sometimes finding an optimal answer requires reaching worse answers which can help find an optimal answer (absolute maximums) after they are improved. Thus, in this method, the improving nature of genetic algorithms is not used, and instead the simulated annealed algorithmis used (to avoid being entangled by local maximums.)

6.2. Simulated Annealed Algorithm

This algorithm is an adaptation of the real-life heating and annealing. The processing simulated algorithm, which cools down a melted metal till it solidifies. The properties of solid structures depend on cold. If a liquid slowly turncold, it will turn into large crystals. On the contrary, if a liquid quickly turns cold, it will become a semi-crystal. Special movements because of the diffusion of heat in the mode space, appear as random up and downs. Yet, what has been used in the present simulated algorithm is as follows:

The rules recently produced by the genetic algorithm, in case they are more efficient than previous generations, will be definitely selected, or otherwise, will be less likely to be selected.Of course, the more the temperature, the more the probability of selecting bad rules, because in high temperature the movement of the particles is more random.

Generally, two sets of rules are stored:

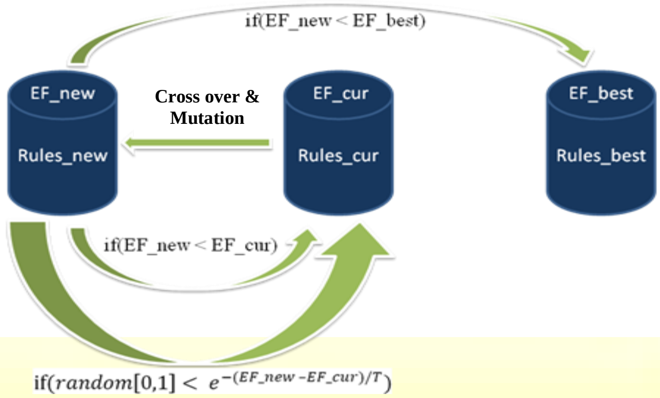
1. The best set of rules for which the highest degree of recognition of cases has been observed (S_{best}).
2. The set of rules that may be one of the best types of bad rules, which are used to avoid local maximums. Changes are always exerted on these rules ($S_{current}$).

Clearly, in high temperatures, the probability of selecting a set of bad rules for the current rules is stronger than low temperatures.

As illustrated in Figure 6, if the set of newly produced rules is better than the previous one, ir t should be put in the $S_{current}$ variable. Otherwise, again depending on the current temperature, it is still possible to store it as current rules of the system.

Figure 6: The synthesis of genetic algorithm and simulated annealing algorithm

EF_new: accuracy of new rules
EF_best: accuracy of current rules
EF_new: accuracy of best rules
T: temperature between 10 and 0.0001
EF_new: accuracy of new rules
EF_best: accuracy of current rules



In reality, through the genetic algorithm, we only try to produce new rules, while no rule selection takes place here. Accepting or rejecting new sets of rules can be accomplished through the simulated annealing algorithm (as depicted in Figure 6).

Genetic-Simulated Annealing
// S_{init} is the initial set of rules
// S_{best} is the best set of rules
// EF_{best} is Evaluation Fitness for best set of rules
// $EF_{current}$ is Evaluation Fitness for current set of rules
// $T_{\max}$ is initial temperature
// $T_{\min}$ is the final temperature
// α is the cooling rate
// β is a constant
// *Time* is the time spent for the annealing process so far
// *k* is the number of calls of metropolis at each temperature
Begin
    $T = T_{\max}$;
    $S_{current} = T_{init}$;
    $S_{best} = S_{current}$; // S_{best} is the best set of rules Soon so far
    $EF_{current} = NNCP\,(S_{current})$;
    $EF_{best} = NNCP\,(S_{best})$;
    Time = 0;
    Repeat
        For i = 1 to k
            Call Metropolis (
                $S_{current}, EF_{current}, S_{best}, EF_{best}, T$)
        Time = Time + k ;
    $T = \alpha \times T$;
Until ($T \geq T_{\min}$);
    Return (S_{best});
End. *//Genetic-Simulated Annealing*

Procedure Metropolis (
    $S_{current}, EF_{current}, S_{best}, EF_{best}, T$)
    // S_{new} is the new set of rules
Begin

$$S_{new} = \begin{cases} Selection(S_{current}); \\ Crossover\,(S_{current}); \\ Mutation(S_{current}); \end{cases}$$

    $EF_{new} = NNCP\,(S_{new})$;
    $\Delta EF = (EF_{new} - EF_{current})$;
    If ($\Delta EF < 0$) Then
        $S_{current} = S_{new}$;
        If $EF_{new} < EF_{best}$ Then
            $S_{best} = S_{new}$;
        End If
    Else If ($random[0,1] < e^{-\Delta EF / T}$) Then
        $S_{current} = S_{new}$;
    End If
End. *//Metropolis*

Figure 7: semi-code for proposed method

7. Conclusion

The present study was concerned with proposing a new method through merging rule-governed fuzzy logic (and language values with a dynamic range of changes), the geneticalgorithms, and simulated annealing algorithm, to recognize Persian handwritten numbers (Table 2 shows the values of related parameters).

Table 2: Assigning values to the parameters of the proposed model

Number of primary rule set	100
Primary temperature	10
Final temperature	0.00001
Temperature decrease coefficient	0.95
Number of stimulated cooling function repetitions in every temperature	15
Number of chromosomes in selecting effective features	20
Probability of crossover in the chromosomes of effective features	80
Probability of mutation in the chromosomes of effective features	20
Probability of crossover in new rule sets	70
Probability of mutation in new rule sets	30
Substitution percentage	20

1000 cases were extracted from the MODARES database (100 samples for each digit). The 10-fold cross-validation method was used in MATLAB. The results of applying this method for recognizing Persian digits compared to other methods indicate its high accuracy (96.67%).Because of a great number of features for each sample (48 features), effective features (30) were recognized through the genetic algorithm. One of the shortcomings of the proposed model is its time complexity in the education time, an aspect that can be ignored considering the appropriate recognition of the method.

References

[1] H.Parvin, H.Alizadeh, M.Moshki, B.Minaei-Bidgoli, N.Mozayani, "*Divide & Conquer Classification and Optimization by Genetic Algorithm*", Third 2008 International Conference on Convergence and Hybrid Information Technology.

[2] H.Alizadeh, H.Parvin, B.Minaei-Bidgoli, "*A New Approach to Improve the Vote-Based Classifier Selection*", 2008 Fourth International Conference on Networked Computing and Advanced Information Management.

[3] F.Shahabi, M.Rahmati, "*A New Method for Writer Identification of Handwritten Farsi Documents*", 2009 10th International Conference on Document Analysis and Recognition.

[4] H.Parvin, H.Alizadeh, B.Minaei-Bidgoli, M.Analoui, "*A Scalable Method for Improving the Performance of Classifiers in Multiclass Applications by Pairwise Classifiers and GA*", 2009 Fourth International Conference on Networked Computing and Advanced Information Management.

[5] Tomoharu nakashima and hisao ishibuchi: "*Using Boosting Techniques To Improve The Performance Of Fuzzy Classification Systems*" In "*Classification And Clustering For Knowledge Discovery*". Studies In Computational Intelligence. Vol 4. pp.146-157 . Springer, Netserlands. 2005.

[6] Ajith Abraham and ravi jain: "*Soft computing models for network intrusion detection system*". In "*Classification And Clustering For Knowledge Discovery*". Studies In Computational Intelligence. Vol 4. pp.190-207. Springer, Netserlands. 2005.

[7] A.Darvish, A.Kabir, H.Khosravi, "The application of shape matching in recognizing Persian handwritten digits", 11th Annual Conference on the Iranian Computer Assembly, Iran. (2005), pp. 285-296.

[8] R, Azmi, A, Kabir, "Recognizing handwritten digits".2nd Electrical Engineering Conference, Iran. (1994). Pp. 285-295

[9] K, Masroori, V, PourmohseniKhameneh, "Recognizing handwritten digits through DTW algorithm". 4th National Conference in Iranian Computer Assembly, Iran. 1997. pp. 1-7

[10]A, Kheirkhah, A,R, Rahmanian, "Optimizing recognition system of Persian handwritten letters based no a conscious selection of effective features of categorization according to genetic algorithm". 1st Congress of Fuzzy and intelligent Systems, Ferdowsi University, 2007

[11] H, Ebrahimnejad, G, Montazer, "Introducing the improved fuzzy method for recognizing Persian digit patterns through the entropy function",11thInternational Conference in Iranian Computer Assembly, Iran.2006