

Fuzzy Classifier Design using Modified Genetic Algorithm

P.Ganesh Kumar¹

*Lecturer, Department of Information Technology,
Anna university Coimbatore, Coimbatore- 641 047, Tamilnadu, India.
E-mail:pganeshkumar_ms@yahoo.co.in*

D.Devaraj²

*Professor and Head, Department of Electrical and Electronics Engineering
Arulmigu Kalasalingam College of Engineering, Krishnankoil-626190, Tamil Nadu, India,*

Received: 13-12-2009

Accepted: 15-02-2010

Abstract

Development of fuzzy if- then rules and formation of membership functions are the important consideration in designing a fuzzy classifier system. This paper presents a Modified Genetic Algorithm (ModGA) approach to obtain the optimal rule set and the membership function for a fuzzy classifier. In the genetic population, the membership functions are represented using real numbers and the rule set is represented by the binary string. A modified form of cross over and mutation operators are proposed to deal with the mixed string. The proposed genetic operators help to improve the convergence speed and quality of the solution. The performance of the proposed approach is demonstrated through development of fuzzy classifier for Iris, Wine and Tcpdump data. From the simulation study it is found that the proposed Modified Genetic Algorithm produces a fuzzy classifier which has minimum number of rules and whose classification accuracy is better than the results reported in the literature.

Keywords: Fuzzy Classifier, If-then-Rules, Membership function, Genetic Algorithm.

1. Introduction

Fuzzy Logic has been successfully applied in solving classification problems where boundaries between classes are not well defined [1,2]. Typical fuzzy classifiers [27] consist of interpretable if-then rules with fuzzy antecedents and class labels in the consequent part. The antecedents (if-parts) of the rules partition the input space into a number of fuzzy regions by fuzzy sets, while the consequents (then-parts) describe the output of the classifier in these regions. An important issue in the design of fuzzy classifier system is the formation of fuzzy if-then rules and the membership functions. In general the rules and membership function are formed from the experience of the human experts. With an increasing

number of variables, the possible number of rules increases exponentially, which makes it difficult for experts to define a complete rule set for good system performance.

Data-driven approaches have been proposed for developing the fuzzy system from numerical data without domain experts [3-7]. But they are very weak in self learning and determining the required number of fuzzy if-then rules. The design of a fuzzy classifier system can be formulated as a search problem in high dimensional space where each point represents a rule set, membership function and the corresponding system behavior. Given some performance criteria, the performance of the system forms a hyper surface

in the space. Developing the optimal fuzzy system is equivalent to finding the optimal location of this hyper surface. This makes Evolutionary Algorithms such as genetic algorithm [8] a better candidate for fuzzy classifier design. Genetic Algorithms are search algorithms based on the mechanics of natural genetics. Genetic Algorithm based methods have been proposed for extracting fuzzy rules and tuning membership function for classification problems. These methods can be categorized into the following four types: Learning fuzzy rules with fixed fuzzy membership functions [9-12], Learning fuzzy membership functions with fixed fuzzy rules [13], Learning fuzzy rules and membership functions in stages (i.e., first evolving good fuzzy rule sets using fixed membership function, then tuning membership functions using the derived fuzzy rule sets [14, 15], Learning fuzzy rules and membership functions simultaneously [16,17].

In most of the above papers binary strings, are used to represent both the membership function and the rule set. This conventional binary-coded GA has Hamming cliff problems [18] which sometimes may cause difficulties in the case of coding continuous variables. Also, for discrete variables with total number of permissible choices not equal to 2^k (where k is an integer) it becomes difficult to use a fixed length binary coding to represent all permissible values. To overcome the above difficulties this paper proposes a Modified Genetic Algorithm (ModGA) approach in which a mixed form of representation is followed to encode the membership function and the rule set. In the proposed ModGA, floating point numbers are used to represent the membership function and binary strings are used to represent the rule set. For effective genetic operation, modified form of crossover and mutation operators which can deal with the mixed string are proposed. The performance of the proposed approach is evaluated through the Iris and Wine data set available in the UCI machine learning repository [19] and Tcpdump dataset available in the MIT Lincoln Labs [20].

2. Classification using Fuzzy IF-THEN Rules

Classification is a supervised learning technique that takes labeled data samples and generates a classifier model that classifies new data samples into different predefined groups or classes. The classification problem can be solved using fuzzy IF-

THEN rules. The general form of interpretable fuzzy if-then rules representing the input features and the output class of a fuzzy classifier is as below:

R_j : if x_{p1} is A_{ji} and ... and x_{pn} is A_{jn} then class C_j where A_{ji}, \dots, A_{jn} are antecedent fuzzy sets of the input variable x_{p1}, \dots, x_{pn} and C_j is one of the output class label. Collections of such rules are used as a knowledge base by the classifier upon which qualitative reasoning is performed to derive conclusion. The relation between input and output is expressed using a fuzzy relation constructed on the basis of fuzzy IF-THEN rules. A fuzzy relation is a fuzzy set defined on universal sets, which are Cartesian products. Mathematically, a fuzzy set A in the universe of discourse X is defined to be a set of ordered pairs,

$$A = \{(x, \mu_A(x)) \mid x \in X\} \quad (1)$$

where $\mu_A(x)$ is called the membership function of x in A . Triangular and trapezoidal membership function are the two most commonly used membership function.

Generally the rules and the membership functions used by the fuzzy logic for solving the classification problem are formed from the experience of the human experts. With an increasing number of variables, the possible number of rules for the system increases exponentially, which makes it difficult for experts to define a complete rule set for good system performance. Also the system performance can be improved by tuning the membership functions. In this paper a modified form of genetic algorithm is proposed to develop the fuzzy classifier.

3. Proposed Genetic Algorithm

Genetic Algorithm [8] is a generalized search and optimization technique inspired by the theory of biological evolution. GA maintains a population of individuals that represent candidate solutions to the problem. Each individual in the population is evaluated to give some measure of its fitness to the problem using the objective function. In each generation, a new population is formed by selecting the more fit individuals based on a particular selection strategy. The selection of individuals to produce successive generations' plays an important role in GA. Some members of the new population undergo genetic operations to form new solutions. The two commonly used operations are crossover and mutation. Crossover is a mixing operator that combines genetic material from selected parents. Mutation acts as a background operator and is used to

search the unexplored search space by randomly changing the values at one or more positions of the selected chromosome.

In a standard Simple Genetic Algorithm (SimGA), binary strings are used to represent the solution variables which may cause difficulties in the case of coding continuous variables due to Hamming Cliff problems and also, for discrete variables with total number of permissible choices not equal to 2^k (where k is an integer) it becomes difficult to use a fixed length binary coding to represent all permissible values. To overcome the above difficulties, the proposed Modified Genetic Algorithm has made the following modification to the standard Genetic Algorithm to improve its efficiency in designing the fuzzy classifier.

- Modified form of representation of chromosome has been followed for representing the rule set and the membership function in the genetic population (i.e., the membership functions are represented using real numbers and the rule set is represented by the binary string)
- With modified form of representation, the selection scheme remains the same but modifications are needed for cross over and mutation operator. Tournament selection [18] is used in this work. The details of the cross over and mutation operators used are given below:

3.1 Crossover operator

For Rule set which is represented in binary form two different types of cross over is applied. First two point crossover [8] operator is applied and then an advanced operator called Gene Cross Swap Operator (GCSO) [22] is applied. Gene Cross Swap Operator randomly selects two different chromosomes from the population and two genes, one from every selected chromosome, and swaps their values as shown in Fig.1

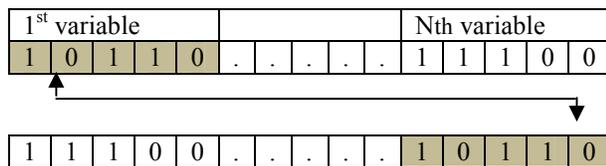


Fig.1. Gene Cross Swap Operator

While two point crossover exchanges information between high-fit chromosomes, the GCSO searches for alternative alleles, exploiting information stored even in low fit strings. For membership function

which is represented in real form, a simple arithmetic cross over is applied. Figure 2 illustrates the Arithmetic crossover for the one-dimensional case. In this figure p_i^1 and p_i^2 are the two parents and a_i, b_i are the lower and upper limit of the i^{th} variable.

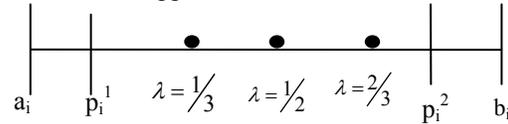


Fig.2. Schematic representation of Arithmetic cross over

Arithmetic Crossover randomly selects two chromosomes that represent parent p_i^1 and p_i^2 from a particular variable and produces two offspring as below:

$$C_i^1 = \lambda \cdot p_i^1 + (1 - \lambda) p_i^2 \tag{2}$$

$$C_i^2 = \lambda \cdot p_i^2 + (1 - \lambda) p_i^1 \tag{3}$$

where $\lambda \in [0,1]$ is a random number with uniform distribution.

3.2 Mutation operator

Mutation operator introduces new material into the population and thereby allows faster and more accurate convergence and prevents convergence to a local optimal value. For Rule set, bit-wise mutation is preferred which switches a few randomly chosen bits from 1 to 0 or from 0 to 1 with a small mutation probability (P_m). For Membership function “Uniform Mutation” operator is applied to the mixed variables with some modifications. First a variable is selected from an individual randomly. If the selected variable is a real number then it is set to a uniform random number between the variable’s lower and upper limit. On the other hand, if the selected variable is an integer then the randomly generated floating point number is truncated to the nearest integer

4. Genetic Algorithm Implementation

While applying Genetic Algorithms for any search or optimization problem, the following issues are to be addressed:

- Representation
- Fitness function formation

4.1 Representation

When designing a fuzzy system using Genetic Algorithms, the first important consideration is the representation strategy to be followed. A fuzzy system is specified only when the rule set and the membership function associated with each fuzzy set are specified. Binary strings are used to represent the rule set. In the rule set each variable is represented by a two bit substring. An additional two bit is used to represent the output class. In order to design a compact rule set only ‘MN_R’ rules are represented in the genetic population. Within those ‘MN_R’ rules, to select the optimal number of rules (ON_R), a rule selection bit is used.

The values of membership function are represented using floating point numbers. The range of each input variable is partitioned into three areas determined by fuzzy sets as shown in figure 3.

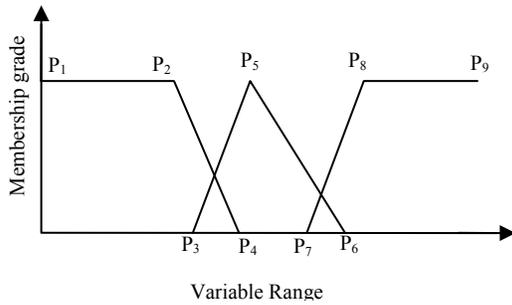
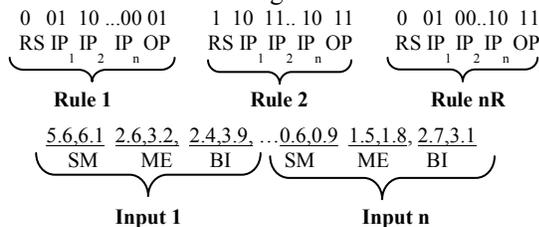


Fig. 3. Fuzzy Space

A total of nine membership points (P₁, P₂, P₃, P₄, P₅, P₆, P₇, P₈, P₉) are required for representing each input variable as a fuzzy set. In that nine points, first and last points (P₁ and P₉) are fixed which are the minimum and maximum of the input variable. The remaining seven membership points are evolved between the dynamic range such that P₂ has [P₁,P₉], P₃ has [P₂,P₉], P₄ has [P₂,P₃], P₅ has [P₄,P₉], P₆ has [P₅,P₉], P₇ has [P₅,P₆] and P₈ has [P₇,P₉] as limits. With the above representation a typical chromosome will look like the following:



This type of representation has a number of advantages over binary coding for membership function. The efficiency of the GA is increased as

there is no need to convert the input variables to the binary type.

4.2 Fitness function

The next important consideration following the representation is the choice of fitness function. Evaluation of the individuals in the population is accomplished by calculating the objective function value for the problem using the parameter set. The result of the objective function calculation is used to calculate the fitness value of the individual. In the classification problem under consideration, there are two objectives; one is to maximize the correctly classified data and the other one is to minimize the number of rules. These two objectives are conflicting objectives. This is overcome by reformulating the first objective of maximizing the correctly classified data as minimizing the difference between total number of samples and the correctly classified data.

Given the total number of samples (‘S’) and the maximum number of rules (‘MN_R’), the task is to find out the difference between ‘S’ and the correctly classified data (‘Cc’) for the selected number of rules(‘SN_R’) of every GA run. During the GA run, the objective is to find out the minimum of the above said value. This is mathematically represented as,

$$\text{Min } f = (S - Cc) + (k \times SN_R) \tag{4}$$

where ‘k’ is a constant introduced to amplify ‘SN_R’ whose value is usually small. In this paper, the value of ‘k’ is taken as 5.

In general, GA searches for a solution with maximum fitness function value. Hence, the minimization objective function given by (4) is transformed to a fitness function to be maximized as

$$\text{Fitness} = \frac{K}{f} \tag{5}$$

where ‘K’ is another constant used to amplify (1/f), the value of which is usually small; so that the fitness value of the chromosome will be in a wider range. In this paper, the value of ‘K’ is taken as 1000.

5. Simulation Result

This section presents the details of the simulation carried out on three datasets to demonstrate the effectiveness of the proposed genetic algorithm based approach for fuzzy classifier design. The first two datasets IRIS and WINE are general benchmark datasets available in the UCI Machine Learning Repository [19] and the third data set TCPDUMP

data is available in the MIT Lincoln Labs [20]. The proposed GA-based approach is implemented in MATLAB and executed in a PC with Pentium IV processor with 2.40 GHz speed and 256 MB of RAM. Tests were carried out on all the three data sets both to examine the learning ability by including the entire samples and generalization ability by dividing the samples into training and testing data. For Comparison, two different genetic algorithm based approaches using binary representation are developed. The first one namely Simple Genetic Algorithm (SimGA) uses basic genetic operators and the other Improved Genetic Algorithm (ImpGA) uses advanced genetic operators in addition to the basic genetic operators. The results of the simulation are presented below:

Case (i): Iris data classification

Iris data set consists of 150 four dimensional vectors representing 50 plants each of three species *iris setosa*, *iris versicolor* and *iris virginica*. The four input features are sepal length, sepal width, petal length and petal width. All these input features are continuous variables. Each input variable is represented by three fuzzy sets and a total of 7 points are needed to represent each variable. The range of each membership function point is computed dynamically.

A maximum of seven rules are included in the genetic population, and each antecedent part is represented by two bit substring with 01 representing “low”, 10 representing “medium”, 11 representing “high” and 00 representing don’t care. The output class is also represented by two bit binary string with ‘01’ representing iris virginica, ‘10’ representing iris versicolor and ‘11’ representing iris setosa and ‘00’ representing don’t care. The substring corresponding to each rule three sections: rule selection bit, representation for the input variables and the representation for the output classes which when coded as binary string needs 11 bits and hence a total of 77 (11 × 7) bits are needed to represent all the rules in the genetic population.

The learning ability is examined by using all the 150 samples as training patterns and evolving the fuzzy classifier. A fuzzy system with three rules and 98.6% classification rate was evolved. Next, the generalization ability is examined by dividing the data set into training and testing data. Out of the 150

data 75 data (25 from each class) is used for training and 75 data (remaining 25 from each class) is used for testing. The training data set is used to find the optimal membership function and the rule set. The testing data set is used to evaluate the performance of the designed fuzzy classifier. During testing all the 75 data were classified correctly. The three rules used by the classifier are given below:

1. If petal length is medium then the flower is iris setosa.
2. If petal length is low and petal width is low then the flower is iris virginica.
3. If the sepal width is low and petal width is low then the flower is iris versicolor.

During the course of each generation the ranges of each membership function points are evolved and tuned simultaneously and above 25 generations the membership function points are begin to uniformly distributed over the range with halfway overlap between them. The optimal membership function obtained for the variable petal length is shown in figure 4.

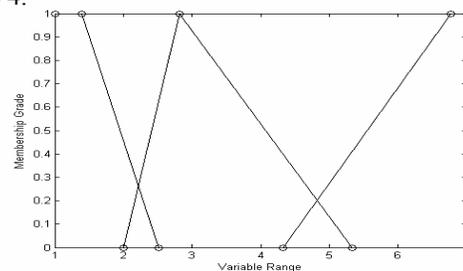


Fig. 4. Optimal MF for the variable petal Length

The above experiments were repeated using SimGA and ImpGA and their convergence behavior against ModGA are shown in figure 5

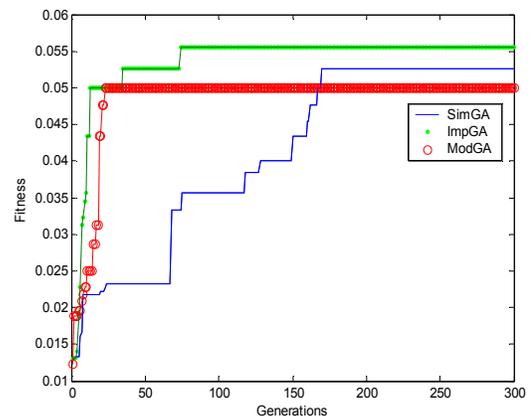


Fig. 5. GA Convergence

From the figure it is found that the proposed ModGA has faster convergence than the other GA models based on binary representation. The SimGA evolve gradually and it took around 175 generations to reach optimal solution. For the ImpGA case, there is a drastic improvement in the fitness value in the first 20 generations and then the value increases

gradually and finally the optimal value is obtained around 100 generations. The proposed ModGA has a steep increase in the fitness value in the first 25 generations and above convergence is very faster and optimal value is reached. Table 1 compares the performance of the proposed ModGA with the other two GA models that uses binary representation.

Table-1 Modified GA compared with binary coded GA for Iris Data

Performance Parameter	LEARNING ABILITY Full Data: 150			GENERALIZATION ABILITY Training: 75 , Testing: 75		
	Sim GA	Imp GA	Mod GA	Sim GA	Imp GA	Mod GA
Generations	300	240	200	300	75	50
Population Size	30	30	30	30	30	30
Cross Over Rate	0.8	0.8	0.9	0.8	0.8	0.9
Mutation Rate	0.05	0.05	0.1	0.05	0.05	0.1
Swap Probability Rate	***	0.01	0.02	***	0.02	0.02
Correctly Classified data	146	147	148	73	74	75
No. of Rules	3	3	3	3	3	3
Accuracy in percentage	97.3%	98%	98.6%	97.3%	98.6%	100%
CPU Time in Seconds	302	257	167.48	247.6	56.81	29.17

From table 1, it is found that for the two cases of the learning and generalization ability, the proposed modified GA outperforms the binary coded GA in all aspects. In table 2 a comparison is made between the proposed method and the various approaches proposed in the literature to design the fuzzy

classifier for the Iris data set. From this table it is inferred that the proposed Modified Genetic Algorithm has designed a Fuzzy Classifier which has a good classification rate and uses minimum number of rules. Also, the proposed approach took minimum number of generations compared to other approaches reported in the literature.

Table-2 Performance Comparison with other approaches for Iris Data

Approach	Classification Accuracy	Rules	Generations
Fuzzy Rule Selection [9]	94.67%	13	1000
Evolutionary Fuzzy [17]	98%	4	1000
Multi Objective GA [23]	96.4%	3	690
Gaussian Fuzzy Classifier [24]	96.7%	3	300
Proposed Method (ModGA)	100%	3	50

Case (ii): Wine data classification

Next the performance of the proposed approach is tested on Wine data. The wine data contains the chemical analysis of 178 wines grown in the same region in Italy but derived from three cultivators which is designated as 3 output classes. The 13 attributes available for classification are: alcohol, malic acid, ash, alkalinity of ash, magnesium, total phenols, flavanoids, nonflavanoids phenols, proanthocyaninism color intensity, hue, OD280/OD315 of diluted wines and praline. Among the 13 input features 11 features are continuous and 2

are discrete. The same representation strategy that is followed in Iris data set is followed here also. A total of 91(13×7) points are needed to represent all the membership functions of all the input variables. A maximum of forty rules were generated initially. Each rule needs 29 bit binary string and hence a total of 1160(29×40) binary bits are used to represent the entire rule set. Simulations are carried out to test the learning ability by considering all the data and generalization ability by dividing the data into two sets. Among the 178 samples 118 data is used for training and 60 data is used for testing. The GA was run for a maximum of 800 generations and optimal

value is obtained at about five hundred generations. The proposed approach shows 96.67% classification rate with just thirty rules. Table 3 compares the

performance of the proposed Modified GA with binary coded GA for Wine data.

Table-3 Modified GA compared with binary coded GA for Wine Data

Performance Parameter	Full Data: 178			Training: 118 Testing: 60		
	Sim GA	Imp GA	Mod GA	Sim GA	Imp GA	Mod GA
Generations	1000	820	750	1000	820	500
Population Size	30	30	30	30	30	30
Cross Over Rate	0.8	0.9	0.9	0.8	0.9	0.9
Mutation Rate	0.05	0.1	0.1	0.05	0.1	0.1
Swap Probability Rate	***	0.02	0.02	***	0.02	0.02
Correctly Classified Class	161	166	170	55	56	58
No. of Rules	60	45	30	60	45	30
Accuracy in percentage	90.4	93.26	95.51	91.67	93.33	96.67
CPU Time in Seconds	8049	3683	2229	7453	2876	1821

Case (iii): Tcpdump data classification

Finally, the proposed approach was applied to design the classifier for the TCPdump data. The TCPdump data set is actually a Widows NT attack data set collected from the 2000 DARPA Intrusion Detection Specific Scenario Data sets. It consists of a total of 294 data with 259 normal cases and 35 abnormal cases with five input features (Source IP address, Destination IP address, Source Port Number, Destination Port Number and the Protocol used) and two output classes (Normal, Abnormal). All the input features are discrete in nature. In the genetic

population, a total of 35 (5×7) points are needed to represent all the membership functions of all the input variables and each rule is represented by 12 bits binary substring. A maximum of five rules were used during the GA run and hence the rule set contains 60(5×12) binary bits. In one hundred and eighty generations 98.98% classification rate is achieved for tcpdump data with two rules. Table 4 compares the performance of the proposed ModGA with binary coded GA that uses binary representation

Table-4 Modified GA compared with binary coded GA for TCPdump Data

Performance Parameter	Full Data: 294			Training: 196 Testing: 98		
	Sim GA	Imp GA	Mod GA	Sim GA	Imp GA	Mod GA
Generations	300	230	300	300	200	180
Population Size	30	30	30	30	30	30
Cross Over Rate	0.8	0.9	0.9	0.8	0.9	0.9
Mutation Rate	0.05	0.1	0.1	0.05	0.1	0.1
Swap Probability Rate	***	0.02	0.02	***	0.02	0.02
Correctly Classified Class	281	285	97	90	95	97
No. of Rules	2	2	2	2	2	2

From the simulation results carried out for Wine and Tcpdump data which is given in Table 3 and 4, it is found that the proposed Modified Genetic Algorithm evolved a fuzzy classifier with compact rule set and has high classification rate in minimum number of generation when compared with the binary coded GA. Also it is inferred that in designing a fuzzy classifier using genetic algorithm, it is good to

represent membership function using floating point numbers and rule set using binary strings.

It is clearly seen from the tables 1 to 4, that the proposed ModGA is slightly faster than the other two GA models. Since SimpGA and ImpGA uses only binary strings, it needs some extra time in decoding it to continuous values for membership function. Also from these tables, it is understood that the proposed

modified form of representation avoids the difficulties of coding continuous variables and allows all permissible values for membership function which results in better classification accuracy than the other two GA models.

Statistical evaluation of experimental results is considered an essential part of validation of new machine learning methods. For comparing the performance of the proposed algorithm with the existing algorithms on multiple data sets, Receiver Operating Characteristic (ROC) analysis [25] is performed. ROC graphs are a useful technique for organizing classifiers and visualizing their performance. An ROC curve is a two-dimensional depiction of classifier performance.

To compare the performance of the proposed ModGA based fuzzy classifier, the ROC performance is reduced to a single scalar value by calculating the Area Under ROC Curve (AUC) [26]. Since the AUC is a portion of the area of the unit square, its value will always be between 0 and 1.0. AUC of a classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. Table 5 gives the value of AUC for different GA-based approaches.

Table 5. Statistical Analysis of GA-based methods

Data Sets	SimGA	ImpGA	ModGA
Iris	0.583 (3)	0.589 (2)	0.595 (1)
Wine	0.955 (2)	0.920 (3)	0.962 (1)
Tcpdump	0.983 (3)	0.989 (2)	0.995 (1)
Average Rank	2.64	2.21	1.00

From table 5, it is clearly found that the proposed MGA based fuzzy classifier has higher probability than the other GA models. From these comparisons, it is found that the proposed ModGA has the ability to provide a richer measure of classification performance than the other algorithms in designing the fuzzy classifier in all the seven cases.

6. Conclusion

The bottleneck of the fuzzy logic based system for any application is the development of rule base and the formation of the membership function. This paper has proposed a modified genetic algorithm approach for the optimal design of the fuzzy system for the classification task. In the proposed approach, a modified form of representation is used to encode the rule base and the membership functions and is

evolved simultaneously. In addition to the basic genetic operators, advanced problem specific operator and modified form of genetic operators have been applied to improve the convergence and the quality of the solution. The effectiveness of the proposed approach for developing fuzzy classifier has been demonstrated through Iris, Wine and TCP Dump data set. For all the cases the proposed approach generated a compact fuzzy system with high classification rate when compared with the binary coded genetic algorithm.

References

1. Timothy Ross, Fuzzy Logic with Engineering Application. Tata McGraw Hill inc; 1995.
2. Devaraj D, Murthy TVSLN, Yegnanarayana B. A Fuzzy System model for plant condition Monitoring. In proceedings International AMSE Conference on Computer Modeling, Simulation and Communication, Jaipur, India; 1999. pp. 210-214.
3. Takagi T, Sugeno M. Fuzzy identification of systems and its application to modeling and control. IEEE Trans. Syst., Man, Cybern 1985;15:116-132.
4. Jang JSR. Fuzzy controller design without domain experts. in proc. 1st IEEE Int. Conf. Fuzzy Systems, San Diego; 1992. pp.289-296.
5. Wang LX, Mendel JM. Generating fuzzy rules by learning from examples. IEEE Trans. Syst., Man, Cybern. 1992; 22:1414-1427.
6. Sugeno M, Yasukawa T. A fuzzy-logic based approach to qualitative modeling. IEEE Trans. Fuzzy Syst. 1993; 1:7-31.
7. Abe S, Lan MS. A Method for fuzzy rule extraction directly from numerical data and its application to pattern classification. IEEE Trans. Fuzzy Syst. 1995; 3:18-28.
8. Goldberg DE. Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, MA: Addison-Wesley; 1989.
9. Ishibuchi H, Nozaki K, Yamamoto N, Tanaka H. Selecting fuzzy if-then rules for classification problems using genetic algorithms. IEEE Trans. Fuzzy Syst. 1995; 3:260-270.
10. Ishibuchi H, Nozaki K, Yamamoto N, Tanaka H. Adaptive Fuzzy Rule-Based Classification Systems. IEEE Trans. Fuzzy Syst. 1996; 4(3):238-250.
11. Ishibuchi H, Nakashima T, Murata T. Performance Evaluation of Fuzzy Classifier Systems for Multidimensional Pattern Classification Problems. IEEE Trans. on System,

- Man and Cybernetics-Part B: Cybernetics. 1999; 29(5):601-617.
12. Ishibuchi H, Nakashima T. Improving the Performance of Fuzzy Classifier Systems for Pattern Classification Problems with Continuous Attributes. IEEE Trans. on Industrial Electronics. 1999; 46(6):1057-1068.
 13. Marco Russo. Genetic Fuzzy Learning. IEEE Trans. Evolutionary Computation, 2000 4(3):259-273.
 14. Magne Setnes, Hans Roubos. GA-Fuzzy Modeling and Classification: Complexity and Performance. IEEE Trans. on Fuzzy Systems. 2008(5):509-522.
 15. Hans Roubos, Magne Setnes. Compact and Transparent Fuzzy Models and Classifiers Through Iterative Complexity Reduction. IEEE Trans. on Fuzzy Systems. 2001; 9(4): 516-524.
 16. Wang CH, Hong T, Tseng S. Integrating Fuzzy Knowledge by Genetic Algorithms IEEE Trans. Evolutionary Computation. 1998; 2(4):138-148.
 17. Yuhui Shi, Russell Eberhart, Yaobin Chen. Implementation of Evolutionary Fuzzy System. IEEE Transaction on Fuzzy System. 1999;7(2):109-119.
 18. Devaraj D, Yegnanarayana B. Genetic Algorithm-Based Optimal Power Flow for Security Enhancement. In IEE Proceedings on Generation, Transmission and Distribution. 2005; 152(6): 899 – 905.
 19. Newman DJ, Hettich S, Blake CL, Merz CJ (1998) UCI repository of machine learning databases, <http://www.ics.uci.edu/~mllearn/MLRepository.html>, University of California, Irvine.
 20. Zissman M. (2000) DARPA intrusion detection scenario specific data sets. http://www.ll.mit.edu/IST/ideval/data/data_index.html, Lincoln Laboratory, Massachusetts Institute of Technology, Lexington.
 21. Jang JSR, Sun CT, Mijutani E. Neuro-Fuzzy and Soft Computing, Prentice Hall Englewood Cliffs. New Jersey; 1997.
 22. Durairaj S, Devaraj D, Kannan PS. Voltage stability constrained reactive power planning using Improved Genetic Algorithm. International Journal of Water and Energy. 2006; 1: 56-64.
 23. Ishibuchi H, Yamamoto T, Fuzzy Rule Selection by Multi-Objective Genetic Local Search Algorithms and Rule Evaluation Measures in Data Mining. Fuzzy Sets and Systems 2004; 141(1): 59-88.
 24. Heesoo Hwang, Identification of a Gaussian Fuzzy Classifier. International Journal of Control, Automation, and Systems 2004.
 25. Fawcett T, (2004) ROC graphs: notes and practical considerations for researchers Kluwer Academic Publishers: 1-38
 26. Bradley AP (1997) The use of area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition. 30(7):1145-1159.
 27. Enrico Zio, Piero Baraldi, Irina Crenguta Popescu, (2008) From Fuzzy Clustering to a Fuzzy Rule Based Classification Model. International Journal on Computational Intelligence System, 1(1), 60-76.