A Genetic-Programming-Based Approach for the Learning of Compact Fuzzy Rule-Based Classification Systems*

F.J. Berlanga[1], M.J. del Jesus[1], M.J. Gacto[2], and F. Herrera[2]

[1] University of Jaén, Dept. of Computer Science, E-23071 Jaén, Spain
{berlanga, mjjesus}@ujaen.es
[2] University of Granada, Dept. of Computer Science and Artificial Intelligence, E-18071 Granada, Spain
mjgacto@ugr.es, herrera@decsai.ugr.es

Abstract. In the design of an interpretable fuzzy rule-based classification system (FRBCS) the precision as much as the simplicity of the extracted knowledge must be considered as objectives. In any inductive learning algorithm, when we deal with problems with a large number of features, the exponential growth of the fuzzy rule search space makes the learning process more difficult. Moreover it leads to an FRBCS with a rule base with a high cardinality. In this paper, we propose a genetic-programming-based method for the learning of an FRBCS, where disjunctive normal form (DNF) rules compete and cooperate among themselves in order to obtain an understandable and compact set of fuzzy rules, which presents a good classification performance with high dimensionality problems. This proposal uses a token competition mechanism to maintain the diversity of the population. The good results obtained with several classification problems support our proposal.

1 Introduction

The Fuzzy Rule-Based Systems have been successfully applied to various fields such as control, modelling and classification. Traditionally, the main goal in the design of this kind of fuzzy systems has been the maximization of the precision, although the interpretability has also been taken into account in some recent studies [1]. This objective is more difficult to reach when the number of features for the problem increase due the exponential growth of the fuzzy rule search space. This growth makes the learning process more difficult and, in most cases, it leads to an FRBCS with a rule base with a high cardinality (with respect to the number of rules and the number of features included in each rule).

An analysis of the specialized literature indicates that exist two principal ways to tackle the problem of the high dimensionality:

1. Compacting and reducing the previously learned rule set in a postprocessing stage ([10], [13]), and

* Supported by the Spanish Ministry of Science and Technology under the Projects TIN-2005-08386-C05-03 and TIN-2005-08386-C05-01.

L. Rutkowski et al. (Eds.): ICAISC 2006, LNAI 4029, pp. 182–191, 2006.

2. Carrying out a feature selection process, that determines the most relevant variables before or during the inductive learning process of the FRBCS ([7], [16], [17]).

In this paper, we tackle the learning of FRBCSs with high interpretability by means of a genetic-programming (GP) based approach. The genetic programming [12] is a kind of evolutionary algorithm that uses variable-length trees to represent the different individuals in the population, instead of fixed-sized vectors (with binary, integer or real codification) as the Genetics Algorithms (GAs) do.

The FRBCSs learning has already been done previously in the specialized literature by means the GP. An initial paper in this topic is Fuzzy GP, developed by Geyer-Schulz [6], which combines a simple GA that operates on a context-free language with a context-free fuzzy rule language. Sánchez et al. propose an FRBCS learning process in [18] and [5] by combining GP operators with simulated annealing and GA respectively to establish the membership functions. Mendes et al. develop in [14] a co-evolutionary algorithm which includes a GP based algorithm for FRBCS learning and an evolutionary algorithm to evolve membership functions. Tsakonas et al. propose in [19] a GP-based algorithm with a Pittsburgh representation for the learning of FRBCSs.

In our proposal, the definition of a context-free grammar that allows the learning of DNF fuzzy rules, together with the use of a competition mechanism between rules which deletes irrelevant rules during the learning process, allow us to obtain of compact FRBCSs (with few rules and conditions per rule) with a high-generalization capability.

The paper is organized as follows. Our proposal is explained in Section 2. Section 3 presents the experimental study and the analysis carried out. Finally, the conclusions obtained are presented in Section 4.

2 The Genetic-Programming-Based Proposal

The first feature of our proposal is the kind of fuzzy rule used. Our method learns DNF fuzzy rules, which have the following form:

$$If\ X_1\ is\ \hat{A}_1\ and\ \ldots\ and\ X_n\ is\ \hat{A}_n\ then\ Y\ is\ C\ with\ CD$$

where each input variable X_i takes as a value a set of linguistic terms or labels $\hat{A}_i = \{A_{i1}\ or\ \ldots\ or\ A_{iLi}\}$ joined by a disjunctive operator, whilst the output variable (Y) has one of the class values. The definition of the fuzzy sets that specify the meaning of each linguistic term or label, is done by using expert knowledge, or in its absence, by using triangular fuzzy sets divided in a uniform way.

This rule also includes a certainty degree ($CD \in [0, 1]$), which represents the confidence of the classification in the class represented by the consequent. In our proposal, this certainty degree is obtained as the quotient Sj / S, where Sj is the sum of the matching degrees for the training examples belonging to class represented by the consequent which are covered by the antecedent of the rule, and S the sum of the matching degrees for all the training examples which are

covered by the antecedent of the rule, independently of the class they belong to. Is important to point out that, in our proposal, this kind of rule is generated according to the production rules of a context-free grammar. The definition of this grammar is completely explained in subsection 2.1.

One of the most important aspects in any evolutionary approach is its coding scheme. The different evolutionary methods follow two approaches in order to encode rules within a population of individuals [4]:

- The "Chromosome = Set of rules", also called the *Pittsburgh* approach, in which each individual represents a rule set.
- The "Chromosome = Rule" approach, in which each individual codifies a single rule, and the whole rule set is provided by combining several individuals in the population.

In turn, within the "Chromosome = Rule" approach, there are three generic proposals:

- The *Michigan* approach, in which each individual codifies a single rule. This kind of system is usually called a learning classifier system. It is rule-based, message-passing system that employs reinforcement learning and the GA to learn rules that guide its performance in a given environment [11].
- The *IRL (Iterative Rule Learning)* approach, in which each chromosome represents a rule, but the solution is the best individual obtained and the global solution is formed by the best individuals obtained when the algorithm is run multiple times. SLAVE [8] is a proposal that follows this approach.
- The *cooperative-competitive* approach, in which the complete population or a subset of it codifies the rule base. LOGENPRO [21] is an example with this kind of representation.

Our method follows the *cooperative-competitive* approach. However, this kind of representation makes necessary to introduce a mechanism to maintain the diversity of the population. In this proposal we use a token competition mechanism to promote the diversity in order to avoid that all the individuals in the population converge into the same area of the search space. This mechanism is described in subsection 2.1, together with the remaining components of the evolutionary learning process.

Finally, is important to point out that our proposal is made up by two different stages:

- The first stage consists in an evolutionary learning process that uses GP to obtain a compact fuzzy rule base with a good classification performance.
- The second one, consists in a postprocessing stage that eliminates redundant rules from the rule base in order to increase the interpretability.

In the following two subsections we describe these two stages.

2.1 Evolutionary Learning Process

The GP process starts with an initial population of rules that is randomly generated according to the grammar production rules.

In each iteration of the evolutionary process, parents are selected to generate offspring by the ranking selection scheme. All the individuals in the new population generate one descendant by means one of the genetic operators. Individuals in the population and offspring obtained by the application of the genetic operators, are joined to form a new population. The size of the resulting population is double the original. The individuals of this new population are ordered by their fitness score and the token competition is carried out. The token competition modifies the fitness of the individuals in order to maintain the diversity of the population. Once token competition is finished, the individuals in the population are ordered again by their modified fitness score, and the population size is set to its original one. This evolutionary process is repeated until a certain number of calls to the fitness function is reached.

Once the evolutionary process is concluded, the best evolved population is returned as the final rule set. The best population obtained in the evolutionary process is selected according to a certain measure of global fitness score.

In the following, the most important components of our method are described.

1. Grammar Definition: In our method, is necessary to define a grammar that allows the learning of DNF fuzzy rules and the absence of some input features. In Table 1, an example of the grammar for a classification problem with two features (X_1, X_2), three linguistic labels per feature (Low, Medium, High) and three classes (C1, C2, C3) is shown.

Table 1. Grammar example

$Start \longrightarrow [If],\ antec,\ [then],\ conseq,\ [.].$
$antec \longrightarrow descriptor1,\ [and],\ descriptor2.$
$descriptor1 \longrightarrow [any].$
$descriptor1 \longrightarrow [X_1\ is]\ label.$
$descriptor2 \longrightarrow [any].$
$descriptor2 \longrightarrow [X_2\ is]\ label.$
$label \longrightarrow \{member(?a,\ [L,\ M,\ H,\ L\ or\ M,\ L\ or\ H,$ $M\ or\ H,\ L\ or\ M\ or\ H])\},\ [?a].$
$conseq \longrightarrow [Class\ is]\ descriptorClass.$
$descriptorClass \longrightarrow \{member(?a,\ [C1,\ C2,\ C3])\},\ [?a].$

2. Genetic Operators: Offspring are generated by one of the next three genetic operators (these operators are selected in a probabilistic way):

1. Crossover: Produces one child from two parents. A part in the first parent is randomly selected and replaced by another part, randomly selected, in the second one, but under the constraint that the offspring produced must be valid according to the grammar.
2. Mutation: A part of the rule is selected and replaced by a randomly generated new part. Since the offspring have to be valid according to the grammar, a selected part can only mutate to another part with a compatible structure.

3. Dropping Condition: Due the probabilistic nature of GP, redundant constraints may be generated in the rule. Thus, it is necessary to generalize the rules, to represent the actual knowledge in a more concise form. Dropping condition selects randomly one descriptor in the antecedent part and then turns it into "any". The attribute in the descriptor is no longer considered in the rule, hence, the rule can be generalized.

3. Fitness Function: Our method uses a fitness function based on the estimation of two measures:

1. *Confidence*, which measures the accuracy of an individual, that is, the confidence of the consequent to be true if the antecedent is verified

$$confidence = \frac{tp}{(tp + fp)} * \frac{tn}{(fn + tn)} . \tag{1}$$

2. *Support*, which measures the coverage of the knowledge represented in the individual

$$support = \frac{tp}{(tp + fn)} * \frac{tn}{(fp + tn)} . \tag{2}$$

where tp and fp are the sums of the matching degrees for true and false positives, and tn and fn are the number of true and false negatives, respectively.

Both measures are combined to make up the fitness function in the following way

$$raw\_fitness = \begin{cases} support, & \text{if } support < min\_support \\ support * confidence, & \text{otherwise} \end{cases} . \tag{3}$$

If the support of the rule is below a user-defined minimum threshold, the confidence value should not be considered to avoid the waste of effort to evolve those individuals with a high confidence but low support.

4. Maintaining the Diversity of the Population: Token Competition [21] has been used as mechanism to maintain the diversity in the population in our approach. It assumes that each example in the training set can provide a resource called a token, for which all chromosomes in the population will compete to capture. If an individual (i.e. a rule) can match the example, it sets a flag to indicate that the token is seized. Other weaker individuals then cannot get the token.

The priority of receiving tokens is determined by the strength of the individuals. The individuals with a high fitness score will seize as many tokens as they can. The other ones will have their strength decreased because they cannot compete with the stronger ones. The fitness score of each individual is penalized based on the tokens it can seize. The penalized fitness is defined as:

$$Penalized\_fitness = raw\_fitness * \frac{count}{ideal} . \tag{4}$$

where *raw_fitness* is the fitness score obtained from the evaluation function, *count* is the number of tokens that the individual actually seized and *ideal* is the total

number of tokens that it can seize, which is equal to the number of examples that the individual matches.

As a result of token competition, there exist individuals that cannot seize any token. These individuals are considered as irrelevant, and they can be eliminated from the population due to all of their examples are covered by other stronger individuals.

In order to increase the diversity into the population, new rules are also generated to cover those examples whose tokens have not been seized by any rule (if those examples exist).

5. Selecting the Best Population: At the end of each iteration in the evolutionary process, a process that keeps the best evolved population is carried out. This process checks if the current population is better than the others that have been evolved. One population A is considered better than other B if the global fitness score of A is greater than the global fitness score of B. The global fitness score is calculated adding four different measures

$$Global\_fitness = Percent \ + \ Nvar \ + Ncond \ + \ Nrules \, . \tag{5}$$

where *Percent* indicates the correct percentage on training, *Nvar* the average number of variables per individual (rule) in the population, *Ncond* the average number of labels per individual (rule) in the population and *Nrules* the number of rules in the population. These four measures are defined in the following way

$$Percent = W_1 \ * \ \%Tra \, . \tag{6}$$

$$Nvar = W_2 \ * \ (1.0 \ - \ \#V) \, . \tag{7}$$

$$Ncond = W_3 \ * \ (1.0 \ - \ \#C) \, . \tag{8}$$

$$Nrules = W_4 \ * \ (1.0 \ - \ \#R) \, . \tag{9}$$

where $\%Tra$ is the normalized value of the correct percentage on training, $\#V$ is the normalized value of the number of variables per rule, $\#C$ is the normalized value of the number of labels per rule, $\#R$ is the normalized value of the number of rules and W_i are some weights that allows give more importance to any of the four measures (in our experiments we have used the same value for all the weights, $W_i = 1$).

2.2 Rule Base Simplification

Once the evolutionary process has finished, a postprocessing stage is carried out for eliminating redundant rules. During the rule base learning process it may happen that the algorithm learns two rules, one included in the other. For example, in the following two rules

$$R1: \ If \ X_1 \ is \ Low \ then \ Class \ is \ C1 \ with \ \alpha_1$$

$$R2: \ If \ X_1 \ is \ Low \ or \ Medium \ then \ Class \ is \ C1 \ with \ \alpha_2$$

the second rule includes the first one, hence, it does not make sense to keep both of them in the rule set. In this case, it must be deleted the first rule because the examples that it covers are also covered by the second rule. Sometimes, it is also necessary to recalculate the certainty degree of the remaining rule. This process aims to increase the interpretability of the previously learned FRBCS, by deleting redundant rules.

3 Experimental Study

In order to analyse the behaviour of the proposed method, we use Pima, Wisconsin, and Wine databases from the UCI repository of machine learning Databases (*http://www.ics.uci.edu/mlearn/MLRepository.html*). Our method (from now on called FRBCS_GP) has been compared to other fuzzy rule learning techniques:

1. An extension of Wang & Mendel algorithm [20] for classification problems proposed by Chi et al. [2], that generates a fuzzy rule for each example in the training set and does not carry out any feature selection process.
2. A process for deriving fuzzy rules for high-dimensionality classification problems developed by Ravi et al. [16]. This approach uses a reduced set of features extracted from the original ones by the principal component analysis. After that, a fuzzy rule learning process is carried out following the method proposed in [9] which divides the pattern space in several fuzzy subspaces, learning a rule for each one. Finally, a modified threshold accepting algorithm [17] is used to build a compact rule subset with a high classification accuracy, from rule set obtained in the previous stage.
3. SLAVE, a GA-based method for the learning of DNF fuzzy rules proposed by Gonzalez et al. [8]. In [7], this method is extended by the inclusion of a feature selection process. This extension will be called 2SLAVE from now.
4. A GP-based FRBCS learning process designed by Tsakonas et al. [19] which uses a Pittsburgh approach to represent the solutions.
5. C4.5, a classification algorithm proposed by Quinlan [15] that constructs a decision tree, which can be later transformed into a crisp rule set.

The parameters of our algorithm are the following: It stops after 5000 evaluations, initial population size is 20, crossover probability is 0.5, mutation probability is 0.4, dropping condition probability is 0.1, and the minimum threshold for the support used in fitness function is 0.01. We have used 5 linguistic labels per variable in all the experiments. For each different database, we have used 10-fold cross-validation (except for the Sonar database, where only two partitions for training and test, with a distribution of 50%, have been used).

The results are showed in Table 2, where #R indicates the average rule number, #Var the average antecedent variables per rule, #Cond the average antecedent conditions number per rule and the %Test the correct percentage with test examples. The subscripts in %Test are related to the fuzzy reasoning method (FRM) [3] used, so 1 corresponds to the classical FRM (max-min) and 2 with the normalised sum respectively (except in the C4.5 algorithm in where FRM is

Table 2. Databases results

Pima

Method	**#R**	**#Var**	**#Cond**	**%Test$_1$**	**%Test$_2$**
WM	472.6	8	8	70.18	70.70
Ravi	354.9	5	5	70.06	70.06
2SLAVE	3.43	4.26	11.28	65.47	65.37
Tsakonas	17.4	1.41	1.74	52.8	51.6
C4.5	47.2	4.47	5.6	71.46	-
FRBCS_GP	10.27	1.14	2.29	71.37	72.73

Sonar

Method	**#R**	**#Var**	**#Cond**	**%Test$_1$**	**%Test$_2$**
WM	104	60	60	43.27	43.27
Ravi	277	6	6	73.08	75
2SLAVE	7.67	22.39	41.43	68.33	68.33
Tsakonas	18.33	1.65	1.73	49	50.67
C4.5	10	3.1	3.1	74	-
FRBCS_GP	11.33	6.5	14.62	75.33	75.33

Wisconsin

Method	**#R**	**#Var**	**#Cond**	**%Test$_1$**	**%Test$_2$**
WM	296.5	9	9	66.34	66.19
Ravi	344.3	5	5	93.85	93.85
2SLAVE	5.73	6.02	16.09	88.53	91.93
Tsakonas	22.7	1.15	1.19	72.2	56.9
C4.5	38.4	4.46	5.11	94.29	-
FRBCS_GP	11.6	1.67	3.81	93.87	94.5

Wine

Method	**#R**	**#Var**	**#Cond**	**%Test$_1$**	**%Test$_2$**
WM	159.4	13	13	78.56	79.74
Ravi	231.8	5	5	92.22	92.22
2SLAVE	5.67	6.9	16.86	89.87	89.87
Tsakonas	19.53	1.38	1.49	35.63	39.13
C4.5	7.1	2.15	2.28	90.51	-
FRBCS_GP	8.53	1.69	3.15	91.77	92.43

not used, so it has been decided to place the results in the first of the last two columns).

Analysing the results, we can point out the following considerations:

– Our method learns rule sets with a low number of variables and labels per rule. It also learns rule bases with a small number of rules. Therefore the resulting FRBCSs have a high interpretability level. Our results are comparable with the ones obtained by 2SLAVE.

– Analysing the performance of our approach, it presents a good performance in test for all the problems, obtaining the best results for all the different databases with the normalised sum FRM.

4 Conclusions

In this work, we have proposed a genetic-programming-based method to obtain FRBCSs with a high interpretability. The definition of a context-free grammar that allows the learning of DNF fuzzy rules and the absence of some input features, allows the obtaining of rules with fewer antecedent conditions. On the other hand, the use of token competition mechanism to increase the diversity into the population makes the rules compete among themselves giving out a smaller number of rules with a high-generalization capability.

The effectiveness of the method has been demonstrated over several classification problems and the results are promising. Therefore, we consider this approach to be an interesting alternative for the learning of interpretable FRBCSs.

As future work we will incorporate a proper multiobjective approach within the learning process.

References

1. Casillas J., Cordón O., Herrera F., Magdalena L. (Eds.): Interpretability Issues in Fuzzy Modeling. Springer-Verlag. Series Studies in Fuzziness and Soft Computing, Vol. 128 (2003)
2. Chi Z., Wu J., Yan H.: Handwritten numeral recognition using self-organizing maps and fuzzy rules. Pattern Recognition 28:1 (1995) 59–66
3. Cordón O., del Jesus M.J., Herrera F.: A Proposal on Reasoning Methods in Fuzzy Rule-Based Classification Systems. International Journal of Approximate Reasoning 20 (1999) 21–45
4. Cordón O., Herrera F., Hoffmann F., Magdalena L.: Genetic Fuzzy Systems. Evolutionary tuning and learning of fuzzy knowledge bases. World Scientific (2001)
5. García S., González F., Sánchez L.: Evolving fuzzy based classifiers with GA-P: A grammatical approach. Lecture Notes in Computer Science, Vol. 1598. Genetic Programming: Second European Workshop, EuroGP'99 (1999) 203–210
6. Geyer-Schulz A.: Fuzzy rule-based expert systems and genetic machine learning. Heidel-berg: Physica-Verlag (1995)
7. González A., Pérez R.: Selection of relevant features in a fuzzy genetic learning algorithm. IEEE Transactions on Systems, Man and Cybernetics Part B 31:3 (2001) 417–425
8. González A. Pérez R.: SLAVE: A genetic learning system based on an iterative approach. IEEE Transactions on Fuzzy Systems 27 (1999) 176–191
9. Ishibuchi H., Nozaki K., Tanaka H.: Distributed representation of fuzzy rules and its application to pattern classification. Fuzzy Sets and Systems 52 (1992) 21–32
10. Ishibuchi H., Nozaki K., Yamamoto N., Tanaka N.: Selecting fuzzy if-then rules for classification problems using genetic algorithms. IEEE Trans. Fuzzy Systems 3:3 (1995) 260–270

11. Kovacs T.: Strength or Accuracy: Credit Assignment in Learning Classifier Systems. Springer-Verlag (2004).
12. Koza J.R.: Genetic programming on the programming of computers by means of natural selection. Cambridge MA, USA: The MIT Press (1992)
13. Krone A., Krause P., Slawinski T.: A new rule reduction method for finding interpretable and small rule bases in high dimensional search spaces. Proc. of the 9th IEEE International Conference on Fuzzy Systems vol. 2 (2000) 694–699
14. Mendes R.R.F., Voznika F. de B., Freitas A.A., Nievola J.C.: Discovering Fuzzy Classification Rules with Genetic Programming and Co-evolution. Principles of Data Mining and Knowledge Discovery: 5th European Conference (PKDD'01). Springer-Verlag. Lecture Notes in Computer Science, Vol. 2168 (2001) 314
15. Quinlan J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
16. Ravi V., Reddy P.J., Zimmermann H.J.: Pattern classification with principal component analysis and fuzzy rule bases. European Journal of Operational Research 126:3 (2000) 526–533
17. Ravi V., Zimmermann H.J.: Fuzzy rule based classification with FeatureSelector and modified threshold accepting. European Journal of Operational Research 123:1 (2000) 16–28
18. Sánchez L., Couso I., Corrales J.A.: Combining GP operators with SA search to evolve fuzzy rule based classifiers. Information Sciences 136:1–4 (2001) 175–191
19. Tsakonas A., Dounias G., Jantzen J., Axer H., Bjerregaard B., von Keyserlingk D.G.: Evolving rule-based systems in two medical domains using genetic programming. Artificial Intelligence in Medicine 32:3 (2004) 195–216
20. Wang L.X., Mendel J.M.: Generating fuzzy rules by learning from examples. IEEE Transactions on Systems, Man, and Cybernetics 22:6 (1992) 1414–1427
21. Wong M.L., Leung K.S.: Data Mining using grammar based genetic programming and applications. Kluwer Academics Publishers (2000)