

Boosting a Genetic Fuzzy Classifier

Frank Hoffmann
Royal Institute of Technology
Stockholm, Sweden
hoffmann@nada.kth.se

Abstract

This paper presents a new boosting algorithm for genetic learning of fuzzy classification rules. The method is based on the iterative rule learning approach to fuzzy rule base system design. The fuzzy rule base is built in an incremental fashion, in that the evolutionary algorithm extracts one fuzzy classifier rule at a time. The boosting mechanism reduces the weight of those training instances that are classified correctly by the new rule, such that the next iteration of the evolutionary algorithm focuses the search on those fuzzy rules that capture the currently uncovered or misclassified instances. The weight of a fuzzy rule reflects the relative strength the boosting algorithm assigns to the rule class when it aggregates the casted votes. The method is applied to the Wisconsin breast cancer diagnosis data set.

1 Introduction

Fuzzy systems are particularly suitable for modeling and classification problems as a human expert is able to analyze and comprehend the knowledge stored in form of linguistic variables and rules. Although fuzzy systems have been successfully applied in a large number of applications, they lack the ability to extract knowledge from a set of training data. Therefore, over the past years more research has been devoted to augment the approximate reasoning method of fuzzy systems with the learning capabilities of neural networks and evolutionary algorithms.

Over the past decade, there has been an increasing interest in evolutionary algorithms that adapt the knowledge base of a fuzzy system. These approaches are described by the general term genetic fuzzy rule based systems (GFRBS) [4]. The role of the evolutionary algorithm is to either tune the parameters of a fuzzy rule based system or to completely automate the fuzzy knowledge base design.

The majority of GFRBS is concerned with the design and optimization of fuzzy logic controllers [8], but recently numerous publications propose evolutionary op-

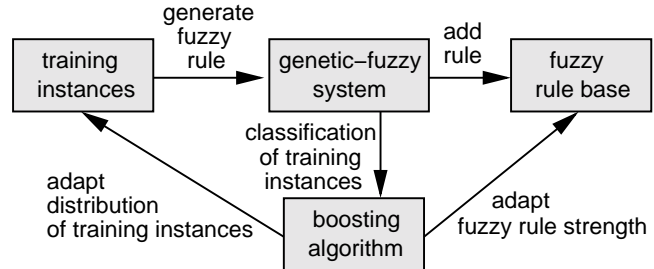


Figure 1. Architecture of boosted genetic fuzzy classifier.

timization of fuzzy rule based classification systems (FRBCS) or short genetic fuzzy classifier [3, 10, 13].

The evolutionary design method for FRBCS presented in this paper follows the iterative rule learning (IRL) approach [7]. The overall architecture of the proposed approach is depicted in figure 1. The rule base is built in an incremental fashion by repeatedly invoking a genetic fuzzy rule generation algorithm. The genetic fuzzy system identifies those fuzzy rules that best match and correctly classify the current distribution of training examples.

Boosting aggregates multiple hypotheses generated by the same learning algorithm invoked over different distributions of training data into a single composite classifier [5]. Boosting generates a classifier with a smaller error on the training data as it combines multiple hypotheses which individually have a larger error. Boosting requires *unstable* classifiers which learning algorithm is sensitive to changes in the training examples.

In our approach, the boosting mechanism adapts the distribution of training examples in a way, that the genetic rule generation algorithm focuses on the previously misclassified or uncovered instances. Thereby, the boosting scheme implicitly promotes cooperation among fuzzy rules. The class label of an unseen instance is obtained by weighting and aggregating the votes casted by the individual fuzzy classification rules [3, 9].

Previously, a boosting algorithm to induce fuzzy rules in

classification problems has been proposed in [11]. The boosting algorithm is able to deal with confidence rated classifications. The approach operates on a descriptive fuzzy system, in which the linguistic partition of the input space is defined in advance. The rule learner performs an exhaustive search over the possible combinations of input fuzzy sets in order to maximize the difference between the sum of membership functions of positive and negative examples that match the rule antecedent.

The basic operation of both approaches is identical, namely to repeatedly generate a new fuzzy rule, determine its weight for the aggregation of votes and to reduce the weights of training examples that are correctly classified by the new rule. The major difference of our method is the genetic rule generation method, the approximate knowledge base, the computation of the rule weight and the criterion for updating the weight of training examples.

The paper is organized as follows. Section 2 introduces the evolutionary algorithm, coding scheme and fitness function employed by the genetic fuzzy system that generates the fuzzy classification rules. Section 3 describes the boosting algorithm which generates a new distribution of training examples for each run of the rule generation method and which aggregates the classifications of the individual fuzzy rules. Section 4 presents experimental results obtained by applying the proposed classification method to the Wisconsin breast cancer diagnosis data set in order to classify cell samples as either malignant or benign. Finally, section 5 draws some conclusions.

2 Genetic Fuzzy System

2.1 Evolution Strategy

Evolutionary algorithms constitute a universal optimization method that imitates the type of genetic adaptation that occurs in natural evolution [1]. Unlike specialized methods designed for particular types of optimization tasks, they require no particular knowledge about the problem structure other than the objective function itself. A population of candidate solutions evolves over time by means of genetic operators such as mutation, recombination and selection. The different types of evolutionary algorithms, namely genetic algorithms (GA), evolution strategies (ES), evolutionary programming (EP) and genetic programming (GP) all share the same principle mode of operation, but utilize different genetic representations and operators.

Evolution strategies are distinguished by self-adaptation of additional strategy parameters, which enables them

to adapt the evolutionary optimization process to the structure of the fitness landscape [1]. A pair of real-valued vectors $(\vec{x}, \vec{\sigma})$ forms the chromosome of an individual. The first part of the vector are the object variables (x_1, \dots, x_n) that represent a candidate solution in the n-dimensional space. The additional vector of strategy parameters $(\sigma_1, \dots, \sigma_n)$ defines the standard deviations of mutations applied to the object variables. A mutation replaces the parent $\vec{x}(t)$

$$\vec{x}(t+1) = \vec{x}(t) + \vec{N}(0, \vec{\sigma}) \quad (1)$$

with the offspring $\vec{x}(t+1)$ where $\vec{N}(0, \vec{\sigma})$ is a normally distributed random vector with zero mean and standard deviation $\vec{\sigma}$.

Rather than using a deterministic step-size adaptation, the evolution strategy itself controls its own mutability, as the strategy vector $\vec{\sigma}$ is subject to secondary mutation.

$$\vec{\sigma}(t+1) = \vec{\sigma}(t) e^{\tau' N(0,1) + \tau \vec{N}(0,1)} \quad (2)$$

The normally distributed random variable $N(0, 1)$ is sampled once for the entire chromosome, whereas the mutations $\vec{N}(0, 1)$ are uncorrelated for different x_i . The global factor $e^{\tau' N(0,1)}$ increases or decreases the overall rate of mutation, whereas $e^{\tau N_i(0,1)}$ adapts the individual step-size σ_i .

2.2 Fuzzy Rule Based Classification System

Fuzzy rule based systems have been widely applied to classification problems. A fuzzy approach is of particular value in domains for which it is important for a human expert to comprehend the classifier decision, for example medical diagnosis or safety critical applications. Each fuzzy rule covers a particular region of the attribute space described by the rule antecedent, for which it proposes the classification specified in the rule consequent. Assume a training set of K instances $T = \{(x^1, c^1), \dots, (x^K, c^K)\}$ where $x^k = \{x_1^k, \dots, x_N^k\}$ is an instance taken from some attribute space $\{X_1, \dots, X_N\}$, and $c^k \in \{C_1, \dots, C_M\}$ is the class label associated with x^k . We use upper indices k to denote the k-th training examples, and lower indices n to denote the n-th attribute x_n^k of a training example x^k . Fuzzy rules are of the form

$$R_i : \text{if } X_1 \text{ is } A_{1i} \text{ and } \dots \text{ } X_N \text{ is } A_{Ni} \text{ then } Y = c_i \quad (3)$$

in which X_n denotes the n-th input variable, A_{ni} the fuzzy set associated to X_n and $c_i \in \{C_1, \dots, C_M\}$ represents the class label of the rule. For a particular instance $x^k = \{x_1^k, \dots, x_N^k\}$, the rule activation

$$\mu_{R_i}(x^k) = \mu_{R_i}(\{x_1^k, \dots, x_N^k\}) = \min_{n=1}^N \mu_{A_{ni}}(x_n^k) \quad (4)$$

describes to what degree the rule matches the instance. For each possible classification C_m the degree of activation of fuzzy rules with a matching consequent $c_i = C_m$ is aggregated. The instance x^k is classified by the class label

$$C_{max}(x^k) = \operatorname{argmax}_{C_m} \sum_{R_i/c_i=C_m} \mu_{R_i}(x^k) \quad (5)$$

that accumulates the majority of rule contributions.

2.3 Coding of Fuzzy Rules

The role of the evolutionary algorithm is to discover fuzzy rules that cover a large number of positive examples and at the same time contain no or only a small number of negative examples. Our classifier employs an approximate fuzzy knowledge base, in which each rule employs its own definition of fuzzy sets, rather than being composed of linguistic labels that refer to a commonly defined set of membership functions [2]. The trapezoidal fuzzy sets A_{nj} are described by the four characteristic points a_n, b_n, c_n, d_n . The chromosome encodes the left most point a_n and the distances between successive points $\delta_n^1 = b_n - a_n, \delta_n^2 = c_n - b_n, \delta_n^3 = d_n - c_n$. This representation ensures that the order of points is maintained as long as the δ_n^i remain positive. The entire rule chromosome is formed by a real-valued vector that is the concatenation of the individual fuzzy set code segments $a_1, \delta_1^1, \delta_1^2, \delta_1^3, \dots, a_N, \delta_N^1, \delta_N^2, \delta_N^3$. As the number of rules required to cover the entire input space grows rapidly with the number of input dimensions the coding scheme provides for general rule antecedents that only refer to a subset of attributes. The chromosome contains an additional bit-string $S = \{s_1, \dots, s_N\}$ in which the bit s_n indicates whether the input clause “ X_n is A_{ni} ” occurs or is omitted in the rule antecedent. Adaptation of the bit-string S enables the evolutionary algorithm to generate fuzzy rules with those attributes that best discriminate among the different classes.

The initial population of rule chromosomes is not generated randomly, but the training instances are used as prototypes to initialize the membership function parameters. The initialization scheme randomly picks a pair of training instances (x^1, c^1) and (x^2, c^2) that share the same class label $c^1 = c^2$. The likelihood of a training instance x^k to serve as a prototype is proportional to the weight w^k assigned to it by the boosting algorithm described in the next section 3. The parameters of the trapezoidal fuzzy sets A_{ni} are selected such that the two training examples span the core of the fuzzy sets

$$\begin{aligned} b_n &= \min\{x_n^1, x_n^2\} \\ c_n &= \max\{x_n^1, x_n^2\} \end{aligned} \quad (6)$$

as depicted in figure 2. The left and right most points a_n, d_n of the fuzzy set A_{ni} are computed such that the

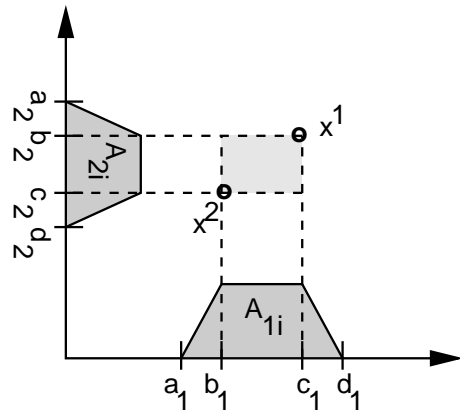


Figure 2. Initialization of trapezoidal fuzzy sets A_{ni} using a pair of training instances x^1, x^2 .

support of A_{ni} is twice as wide as its core (see 2)

$$\begin{aligned} a_n &= b_n - \frac{|x_n^1 - x_n^2|}{2} \\ d_n &= c_n + \frac{|x_n^1 - x_n^2|}{2} \end{aligned} \quad (7)$$

The width of the core $|x_n^1 - x_n^2|$ also determines the initial values of the mutation rates $\sigma_{a_n, b_n, c_n, d_n} = \frac{|x_n^1 - x_n^2|}{5}$ in the evolution strategy (see equations 1,2).

2.4 Fitness Function

In [6], the authors propose a number of optimization criteria to evaluate the quality of a fuzzy classification rule, such as a high frequency value, a high covering degree over positive examples and the k-consistency property to penalize negative examples covered by a rule. The boosting algorithm described in section 3 changes the distribution of the training instances by weighting them according to their difficulty. Therefore, in each iteration of the boosting algorithm, the training examples are weighted by a factor w^k , that reflects the frequency of the instance x^k in the training set. The definition of the optimization criteria frequency value and rule consistency takes these weight factors into account. The fitness function considers two objectives, namely the number of training instances covered by the rule R_i compared to the number of training instances that have the rule class label c_i

$$f_1 = \frac{\sum_k |c^k = c_i| w^k \mu_{R_i}(x^k)}{\sum_k |c^k = c_i| w^k} \quad (8)$$

and the frequency of negative examples covered by a rule

$$f_2 = \frac{\sum_k |c^k \neq c_i| w^k \mu_{R_i}(x^k)}{\sum_k w^k \mu_{R_i}(x^k)} \quad (9)$$

Both objectives are aggregated into a scalar fitness value

$$f = \begin{cases} 0 & : f_2 > k_{max} \\ f_1 * (1 - \frac{f_2}{k_{max}}) & : f_2 \leq k_{max} \end{cases} \quad (10)$$

where the parameter k_{max} denotes the maximal rule inconsistency that is still tolerated [6]. Our experiments were performed with a value of $k_{max} = 0.2$, but the results do not depend critically on the exact value of the parameter.

3 Boosting a Genetic Fuzzy Classifier

3.1 Iterative Rule Learning Approach

Every GFRBS somehow has to address the issue of competition versus cooperation of fuzzy rules, namely the inherent conflict that in the selection step of the genetic algorithm fuzzy rules compete with each other, whereas in the fuzzy inference step the overall output is aggregated of the actions proposed by multiple, simultaneously active fuzzy rules. The iterative rule learning (IRL) approach, first proposed in [7], addresses the competition versus cooperation problem in that it divides the learning algorithm into two stages, a generation stage that obtains a preliminary set of rules, and a second post-processing stage which refines the obtained rules in order to improve the cooperation among them. The generation phase itself operates on two levels: an evolutionary algorithm to generate fuzzy rules that correctly classify subsets of the training set, and an iterative covering method that assures that each training instance is ultimately covered by some fuzzy rule [3]. The iterative covering method repeatedly invokes the evolutionary rule generation algorithm to incrementally add new rules to the rule base. Those training examples that are sufficiently covered by the current set of rules are removed from the training set, such that the evolutionary search concentrates on the remaining instances. One possible way to proceed with the iterative covering method in classification problems is to learn one particular class label at a time, as suggested in the SLAVE algorithm [6]. Once all the examples with that label are covered the fuzzy rule generation method is invoked to identify rule antecedents that capture instances with the next class label.

The major difference of the approach presented in this paper is that the boosting mechanism already considers the cooperation among fuzzy rules in the rule generation stage. Cooperation is promoted by assigning a larger weight to those instances which the current set of fuzzy rules either misclassifies or does not cover. Thereby the genetic fuzzy system is biased to generate particularly those fuzzy rules that complement the current set

of fuzzy rules and correct their deficiencies. In principle, the second rule post-processing stage becomes obsolete, although it can still be applied to further improve the performance of the final rule base.

3.2 Boosting Algorithm

This paper advocates to augment the iterative covering method by a fuzzy variant of the AdaBoost algorithm proposed in [5]. The idea of boosting is to repeatedly apply a weak learning algorithm on various distributions of the training data and to aggregate the individual classifiers into a single overall classifier. After each iteration the distribution of training instances is changed based on the error the current classifier exhibits on the training set. For learning algorithms that can deal with fractional training instances it is not necessary to sample the next set of training instances T from a distribution. Instead, the weights w^k specify the relative importance of the k -th training instance, which can be interpreted as if the training set would contain w^k identical copies of the training example (x^k, c^k) . The weights w^k of correctly classified instances (x^k, c^k) are reduced, such that future classifiers concentrate on the still incorrect examples.

Initially, all training examples are weighted uniformly with the weight $w^k = 1$. The boosting algorithm repeatedly invokes the genetic fuzzy rule generation method on the current distribution of training examples. Notice, that the fitness of a fuzzy rule in equations 8, 9 depends on the weights w^k .

The boosting algorithm computes the error $E(R_t)$ of the fuzzy rule R_t generated in iteration t . In our case each fuzzy classification rule constitutes an incomplete, weak classifier. Incomplete in the sense that it is able to classify those instances covered by its antecedent but makes no prediction about the other training examples. Therefore, the classification error $E(R_t)$ of a fuzzy rule R_t is weighted by the degree of matching $\mu_{R_t}(x^k)$ between the k -th training instance (x^k, c^k) and the rule antecedent as well as its weight w^k

$$E(R_t) = \frac{\sum_{k|c_k \neq C_i} w^k \mu_{R_t}(x^k)}{\sum_k w^k \mu_{R_t}(x^k)} \quad (11)$$

In other words, the goal of the fuzzy rule generation method is to find classification rules that perform well over the current distribution of training examples.

To compute the new set of weights $w^k(t+1)$ from the fuzzy rule R_t generated at iteration t , the weight $w^k(t)$ of an instance (x^k, c^k) is multiplied by some factor β^k if R_t classifies the example correctly and is left unchanged otherwise.

$$w^k(t+1) = \begin{cases} w^k(t) & \text{if } c_i = c_k \\ w^k(t) * \beta^k & \text{if } c_i \neq c_k \end{cases} \quad (12)$$

The factor

$$\beta^k = \left(\frac{E(R_t)}{1 - E(R_t)} \right)^{\mu_{R_t}(x^k)} \quad (13)$$

depends on the error E_{R_t} of the fuzzy rule and the degree of matching $\mu_{R_t}(x^k)$ between the fuzzy rule and the training example. Effectively, examples that are classified correctly and match the rule antecedent are down-weighted, and misclassified or uncovered examples keep their original weights. Thereby, the boosting algorithm increases the relative weight of those examples which are hard to learn for the genetic fuzzy system.

3.3 Fuzzy Classifier Aggregation

The remaining task is to aggregate the votes of the fuzzy rules R_t into a final classification. The vote of each rule R_t is weighted by the factor $\log(1/\beta_t)$, with

$$\beta_t = \frac{E(R_t)}{1 - E(R_t)} \quad (14)$$

so that rules with small classification error $E(R_t)$ obtain a larger weight. Our approach follows the voting aggregation scheme proposed in [3, 9], in that all the rules contribute to the overall decision, rather than to pursue a winner-takes-all approach in which the rule with the highest matching degree dictates the ultimate classification.

Assigning a weight to each fuzzy rule R_t has the drawback, that the intuitive interpretation of the fuzzy rule base gets blurred, as the contribution of a rule to the overall classification not only depends on how well it matches the instance but also on its weight. Similar to equation 5 the votes are weighted by the matching degree $\mu_{R_t}(x^k)$ between the instance and the rule antecedent. For a given instance x^k , the boosting classifier outputs the class label C_{max} that maximizes the sum

$$C_{max} = \operatorname{argmax}_{C_m} \sum_t \log(1/\beta_t) \sum_{R_t/c_t=C_m} \mu_{R_t}(x^k) \quad (15)$$

4 Results

We conducted experiments on the Wisconsin breast cancer data set which contains medical data of 569 patients [12]. The instance attributes correspond to 10 features, such as texture, smoothness and radius, of cell nuclei obtained by fine needle aspiration of the breast lump. The features are automatically extracted by image analysis of the digital scan of the cells in the needle aspirate. The mean, standard error, and "worst" or largest

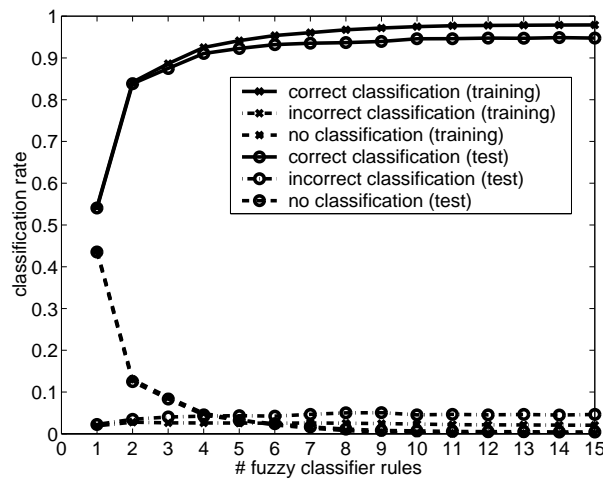


Figure 3. Average classification rate of the genetic fuzzy classifier over training (x) and test (o) set instances.

value of these features are computed for each image, resulting in 30 real-valued attributes. Each instance is classified as either malignant or benign according to the actual, known diagnosis of the patient. Out of the 569 instances, 312 are classified as benign, 212 as malignant.

The fuzzy classification rules were generated by means of an evolution strategy that adapted the parameterized fuzzy membership functions in the rule antecedents. The mutation operator was only applied to the parameters of active fuzzy sets. With a small probability $p = 0.025$ an inactive fuzzy clause could become active and vice versa. Recombination was applied on the rule code level to recombine entire fuzzy sets originating from different parents, but not on the level of fuzzy set parameters. The initial population was generated using the approach described in subsection 2.3. The selection scheme was a deterministic (μ, λ) -strategy, in which the $\mu = 10$ best individuals became parents to the new generation of $\lambda = 200$ offspring. The evolution strategy was run for 40 generations, after which the best overall generated fuzzy rule was added to the set of classifiers. In principle, the outer loop boosting algorithm can iterate until the combined classifier correctly predicts the class label of every training instance. However, in order to avoid over-fitting to the training data, the boosting algorithm was terminated after 15 fuzzy rules have been generated. In general, the best results on unseen data are obtained by selecting the combined classifier with those number of fuzzy rules that demonstrate the smallest error on a validation set not used during training. Figure 3 depicts the averaged classification rate of correctly, incorrectly and unclassified training and test instances as a function of the number of rules. In our experiments the average test set error stagnates after about 10 rules (see

# rules	training	best	avg.	worst
15	correct class.	98.3%	97.9%	97.6%
	incorrect class.	1.7%	2.0%	2.3%
	unclassified	0.0%	0.1%	0.1%
10	correct class.	97.7%	97.5%	97.2%
	incorrect class.	2.1%	2.3%	2.7%
	unclassified	0.1%	0.2%	0.3%
5	correct class.	94.5%	94.1%	93.8%
	incorrect class.	2.1%	2.6%	3.0%
	unclassified	2.8%	3.3%	4.0%
# rules	test	best	avg.	worst
15	correct class.	95.8%	94.8%	93.7%
	incorrect class.	3.7%	4.6%	5.7%
	unclassified	0.2%	0.4%	0.7%
10	correct class.	95.7%	94.6%	93.9%
	incorrect class.	3.5%	4.5%	5.3%
	unclassified	0.4%	0.7%	1.1%
5	correct class.	93.1%	92.3%	90.9%
	incorrect class.	3.5%	4.3%	4.9%
	unclassified	2.2%	3.2%	4.2%

Table 1. Classification accuracy of the genetic fuzzy classifier over training and test set instances for 5,10 and 15 rules.

figure 3, table 1), which is in accordance with a similar observation in [11].

Table 1 shows the best, average and worst classification accuracy computed over 8 runs estimated by means of 10-fold cross-validation for 5, 10 and 15 rules. The average number of examples that could not be classified with 15 rules, as they were not covered by any fuzzy rule was less than 0.1% for the training data and about 0.4% for the test data. The number of correctly classified instances is comparable with the classification accuracy of about 97 – 98% on the training and 94 – 96% on the test data reported for a similar data set achieved by a genetic fuzzy system based on the Pittsburgh approach [13].

5. Conclusions

This paper presented a genetic fuzzy system for fuzzy classification rules based on the iterative rule learning approach. A novel boosting algorithm systematically reduces the weight of the correctly classified training examples, in order to focus the next iteration of the rule generation method on those training examples that are hard to learn. Cooperation among fuzzy rules is promoted as the genetic fuzzy system is biased to generate those fuzzy rules that correct the classification errors made by the current set of fuzzy rules. The proposed method was applied to the breast cancer data set and

was able to classify cell samples as malignant or benign with high accuracy.

Acknowledgments

This research presented in this paper has been sponsored by the Swedish Foundation for Strategic Research.

References

- [1] T. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, 1996.
- [2] O. Cordón and F. Herrera. A three-stage evolutionary process for learning descriptive and approximate fuzzy logic controller knowledge bases from examples. *International Journal of Approximate Reasoning*, 17(4):369–407, 1997.
- [3] O. Cordón, M. J. del Jesus, and F. Herrera. Genetic learning of fuzzy rule-based classification systems cooperating with fuzzy reasoning methods. *International Journal of Intelligent Systems*, 13(10-11):1025–1053, Nov. 1998.
- [4] O. Cordón, F. Herrera, F. Hoffmann, and L. Magdalena. *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases*. Advances in Fuzzy Systems. World Scientific, Singapore, 2001.
- [5] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proc. of the 13th Int. Conf. on Machine Learning ML-96*, 1996.
- [6] A. González and R. Pérez. Completeness and consistency conditions for learning fuzzy rules. *Fuzzy Sets and Systems*, 96:37–51, 1998.
- [7] A. González and F. Herrera. Multi-stage genetic fuzzy systems based on the iterative rule learning approach. *Mathware & Soft Computing*, 4:233–249, 1997.
- [8] F. Hoffmann. Evolutionary algorithms for fuzzy control system design. *Proceedings of the IEEE*, 2001. To appear in special issue on Industrial Innovation using Soft Computing.
- [9] H. Ishibuchi, T. Nakashima, and T. Morisawa. Voting in fuzzy rule-based systems for pattern classification problems. *Fuzzy Sets and Systems*, 103:223–238, 1999.
- [10] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka. Selecting fuzzy if-then rules for classification problems using genetic algorithms. *IEEE Transactions on Fuzzy Systems*, 3(3):260–270, 1995.
- [11] L. Junco and L. Sanchez. Using the adaboost algorithm to induce fuzzy rules in classification problems. In *Proc. Spanish Conference for Fuzzy Logic and Technologies (ESTYLF 2000)*, pages 297–301, Sevilla, Spain, September 2000.
- [12] W. S. O.L. Mangasarian and W. Wolberg. Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, 43(4):570–577, 1995.
- [13] C. Peña-Reyes and M. Sipper. A fuzzy-genetic approach to breast cancer diagnosis. *Artificial Intelligence in Medicine*, 17(2):155, 1999.