An Artificial Immune System for Fuzzy-Rule Induction in Data Mining

Roberto T. Alves[1], Myriam R. Delgado[1], Heitor S. Lopes[1], and Alex A. Freitas[2]

[1] CPGEI, CEFET-PR, Av. Sete de Setembro, 3165, CEP: 80230-901 Curitiba, Brasil
roberto_alves@msn.com, myriam@dainf.cefetpr.br, hslopes@cpgei.cefetpr.br
[2] Computing Laboratory, University of Kent, Canterbury, CT2 7NF, UK
A.A.Freitas@kent.ac.uk

Abstract. This work proposes a classification-rule discovery algorithm integrating artificial immune systems and fuzzy systems. The algorithm consists of two parts: a sequential covering procedure and a rule evolution procedure. Each antibody (candidate solution) corresponds to a classification rule. The classification of new examples (antigens) considers not only the fitness of a fuzzy rule based on the entire training set, but also the affinity between the rule and the new example. This affinity must be greater than a threshold in order for the fuzzy rule to be activated, and it is proposed an adaptive procedure for computing this threshold for each rule. This paper reports results for the proposed algorithm in several data sets. Results are analyzed with respect to both predictive accuracy and rule set simplicity, and are compared with C4.5rules, a very popular data mining algorithm.

1 Introduction

Data mining consists of extracting knowledge from real-world data sets. We stress that the goal of data mining is to discover knowledge that is not only accurate, but also comprehensible [1],[2], i.e. knowledge that can be easily interpreted by the user. Hence, the user can validate discovered knowledge and combine it with her/his background knowledge in order to make an intelligent decision, rather than blindly trusting the results of a "black box".

This work focuses on the classification task of data mining, where the goal is to discover a classification model (a rule set, in this work) that predicts the class of an example (a record) based on the values of predictor attributes for that example.

More precisely, this work proposes a new algorithm for inducing a set of fuzzy classification rules based on an artificial immune system (AIS), a relatively new computational intelligence paradigm [3]. The proposed algorithm discovers a set of rules of the form "IF (fuzzy conditions) THEN (class)", whose interpretation is: if an example's attribute values satisfy the fuzzy conditions then the example belongs to the class predicted by the rule. The fuzzy representation of the rule conditions not only gives the system more flexibility to cope with uncertainties typically found in real-world applications, but also improves the comprehensibility of the rules [4],[5].

The remainder of this paper is organized as follows. Section 2 presents a brief overview of AIS, fuzzy systems, and related work. Section 3 describes in detail the

proposed algorithm. Section 4 reports computational results. Finally, section 5 presents the conclusions and future research directions.

2 Artificial Immune Systems, Fuzzy Systems and Related Work

AIS consist of methods that are inspired by the biological immune system and designed to solve real-world problems [6]. This work focuses on one kind of AIS inspired by the clonal selection principle of the biological immune system. In essence, when an immune system detector (a lymphocyte) has a high affinity (a high degree of matching) with an antigen (invader microorganism), this recognition stimulates the proliferation and differentiation of cells that produce antibodies. This process, called clonal expansion (because new cells are produced by cloning and mutating existing cells), produces a large population of antibodies targeted for that antigen. This clonal expansion leads to the destruction or neutralization of the antigen and to the retention of some cells in the immunological "memory", so that the immune system can act more quickly the next time the same antigen is found in the body.

This process is a form of natural selection. The better a clone recognizes an antigen, the more it tends to proliferate. This process is also adaptive, because the clones undergo mutation. Since the reproduction rate is very high, the frequency of mutation is also very high. This mechanism is called somatic mutation or hypermutation. Jointly with the selective process, somatic mutation improves the clones' ability in recognizing the antigen (since the best mutations lead to a higher proliferation of the corresponding clones), producing clones with greater affinity for that particular antigen.

Fuzzy systems use symbols – called linguistic terms – that have a well-defined semantics and are represented by membership functions of fuzzy sets. This allows the numerical processing of those symbols or concepts. Fuzzy systems are very effective in expressing the natural ambiguity and subjectivity of human reasoning [4],[5].

Membership functions determine to which degree a given object belongs to a fuzzy set. In a fuzzy system this degree of membership varies from 0 to 1. Membership functions can take different forms, varying from the simplest ones (triangular functions) to more complex functions (parameterized by the user).

According to [7], in a classification problem with n attributes, fuzzy rules can be written as: R_j : If x_1 is A_1^j and .... and x_n is A_n^j then Class C_j, $j = 1,..,N$, where $\mathbf{x} = (x_1,...,x_n)$ is an n-dimensional pattern vector, A_i^j (i=1,…,n) is the i-th attribute's linguistic value (e.g. small or large), C is the class predicted by the rule, and N is the number of fuzzy if-then rules. Hence, the antecedent ("IF part") of each fuzzy rule is specified by a combination of linguistic values.

We now briefly review related work. Probably the first AIS specifically designed for the classification task is AIRS [8]. In addition, it has been suggested that an AIS based on the clonal selection principle, called CLONALG, can be used for classification in the context of pattern recognition [6], although originally proposed for other tasks. However, unlike the AIS algorithm proposed in this paper, neither AIRS nor CLONALG discovers comprehensible IF-THEN rules. Hence, neither of those two algorithms addresses the data mining goal of discovering comprehensible, interpret-

able knowledge (see Introduction). Also, they do not discover fuzzy knowledge, unlike the algorithm proposed in this paper. An AIS for discovering IF-THEN rules is proposed in [9]. Unlike the algorithm proposed in this paper, that work is based on extending the negative selection algorithm with a genetic algorithm. We have avoided the use of the negative selection algorithm because this kind of AIS method has some conceptual problems in the context of the classification task, as discussed in [10]. Also, again that work does not discover fuzzy rules. A fuzzy AIS is proposed in [11]. However, that work addresses the task of clustering, which is very different from the task of classification addressed in this paper. To the best of our knowledge, the algorithm proposed in this paper is the first AIS for discovering fuzzy classification rules based on the clonal selection principle.

3 Description of the IFRAIS Algorithm

The proposed algorithm, called IFRAIS (Induction of Fuzzy Rules with an Artificial Immune System), discovers fuzzy classification rules. In essence, IFRAIS evolves a population of antibodies, where each antibody represents the antecedent (the "IF part") of a fuzzy classification rule. Each antigen represents an example (record, or case). The rule antecedent is formed by a conjunction of conditions (attribute-value pairs, e.g. "*Salary = low*"). Each attribute can be either continuous (real-valued, e.g. *Salary*) or categorical (nominal, e.g. *Gender*), as usual in data mining. Categorical attributes are inherently crisp, but continuous attributes are fuzzified by using a set of three linguistic terms (*low, medium, high*). In this work these linguistic terms are represented by triangular membership functions, for the sake of simplicity.

More precisely, an antibody is encoded by a string with n genes, where n is the number of attributes. Each gene i, i=1,…,n, consists of two elements: (a) a value V_i specifying the value (or linguistic term) of the i-th attribute in the i-th rule condition; and (b) a boolean flag B_i indicating whether or not the i-th condition occurs in the classification rule decoded from the antibody. Hence, although all antibodies have the same genotype length, different antibodies represent rules with different number of conditions in their antecedent – subject to the restriction that each decoded rule has at least one condition in its antecedent. This flexibility is essential in data mining, where the optimal number of conditions in each rule is unknown a priori.

The rule consequents (predicted classes) are not evolved by the AIS. Rather, all the antibodies of a given AIS run are associated with the same rule consequent, so that the algorithm is run multiple times to discover rules predicting different classes – as will be explained in more detail in subsection 3.1.

3.1 Discovering Rules from the Training Set

The IFRAIS algorithm (version 1.0) is described in the pseudocodes of Figure 1 – the Sequential Covering (SC) – and Figure 2 – the Rule Evolution (RE).

The SC procedure starts by initializing the *DiscoveredRuleSet* to the empty set, and then it performs a loop over the classes to be predicted [2]. For each class, the algorithm initializes the *TrainSet* with the set of all examples in the training set and iteratively calls the RE procedure, passing as parameters the current *TrainSet* and the class

c to be predicted by all the candidate rules in the current run of that procedure. The RE procedure returns the best evolved rule, which is then stored in the variable *BestRule*. Next the algorithm adds the *BestRule* to the *DiscoveredRuleSet* and it removes from the current *TrainSet* the examples that have been correctly covered by the best-evolved rule. An example is correctly covered by a rule if and only if the example satisfies the rule antecedent and the example has the same class as predicted by the rule. In order to compute whether or not an example satisfies a rule antecedent we compute the affinity between the rule and the example, as follows.

```
Input: full training set;
Output: set of discovered rules;

DiscoveredRuleSet = ∅;
FOR EACH class c
  TrainSet = {set of all training examples};
  WHILE |TrainSet| > MaxUncovExamp
     BestRule = RULE-EVOLUTION(TrainSet, class c);
     DiscoveredRuleSet = DiscoveredRuleSet ∪ BestRule;
     TrainSet = TrainSet – {set of examples correctly covered by BestRule};
  END WHILE;
END FOR EACH class;
FOR EACH rule in DiscoveredRuleSet
   Recompute the fitness of the rule (antibody) using the full training set of examples;
END FOR;
```

Fig. 1. Sequential Covering (SC) procedure

```
Input: current TrainSet;
       the class c predicted by all the rules/antibodies in this run of this procedure;
Output: the best evolved rule;

Create initial population of antibodies at random;
Prune each rule antecedent in a stochastic way;
Compute fitness of each antibody;
FOR i = 1 to Number of Generations
   Perform tournament selection T times, getting T winners to be cloned;
   FOR EACH antibody to be cloned
     Produce C clones of the antibody, where C is proportional to fitness;
     FOR EACH just-produced clone
       Mutate clone with a rate inversely proportional to its fitness;
       Prune each clone in a stochastic way;
       Compute fitness of the clone;
     END FOR EACH clone;
   END FOR EACH antibody;
   Replace the T worst-fitness antibodies in the population by the T best-fitness clones;
END FOR i;
Return the rule whose antecedent consists of the antibody with the best fitness among all
antibodies produced in all generations, and whose consequent consists of class c;
```

Fig. 2. Rule Evolution (RE) procedure

First, for each condition in the rule decoded from an antibody, the algorithm computes the degree to which the original continuous value of the corresponding attribute

(in the database) belongs to the fuzzy set associated with the rule condition. These degrees of membership are denoted by $\mu_{A1}(x_1),...,\mu_{An}(x_n)$ where n is the number of conditions in the rule. The next step is to compute the degree to which the example satisfies the rule antecedent as a whole. This is computed by applying the standard aggregation operator *min* to the $\mu_{A1}(x_1),...,\mu_{An}(x_n)$ values. More precisely, the affinity between an antibody j and an antigen k is given by Equation (1):

$$Afin(k,j) \quad = f\left(\mu_{A_1}(x_1), \cdots, \mu_{A_n}(x_n)\right) = \mu_{A1}(x_1) \wedge \cdots \wedge \mu_{A_n}(x_n) \qquad \textbf{(1)}$$

An example satisfies a rule (i.e., a rule is activated for that example) if the degree of affinity between the rule and the example is greater than an activation threshold, i.e., if $Afin(k,j) > L_j$, where L_j denotes the activation threshold for the j-th rule.

This work proposes an adaptive procedure to automatically choose the best value of affinity threshold L_j for each rule j, out of a reasonably large range of values. More precisely, the algorithm considers m uniformly distributed values of L_j within the range [0.5, 0.7]. In this work $m = 20$, so that the algorithm considers all values of L_j in {0.50, 0.51, 0.52, …,0.69, 0.70}. For each of these values, the algorithm computes the fitness of the rule in the training set (as will be explained later) and chooses the L_j value that maximizes the fitness of the rule. It should be noted that this mechanism does not require recomputing the degree of matching between each example and the rule, which is the most computationally expensive part of fitness computation. It just requires recomputing the number of examples satisfying and not satisfying each rule, so that its processing time is not too long. In the above range, the lower bound 0.5 is a natural value, since a degree of affinity smaller than 0.5 would mean that the example does not satisfy the rule to a degree greater than the degree to which it satisfies the rule. Intuitively the rule should not be activated in this case. The upper bound of 0.7 and the value of $m = 20$ were empirically determined, and seem to cover a reasonably wide range of useful values for L_j.

The WHILE loop is iteratively performed until the number of uncovered examples is smaller than a user-defined threshold *MaxUncovExamp*, so that this procedure discovers as many rules as necessary to cover the vast majority of the training examples. Finally, in the last step of the SC procedure we recompute the fitness of each rule in the *DiscoveredRuleSet*, by using the full training set.

The RE procedure starts by randomly creating an initial population of antibodies, where each antibody represents the antecedent of a fuzzy classification rule. For each rule, the system prunes the rule and computes the fitness of the antibody. Rule pruning has a twofold motivation: reducing the overfitting of rules to the data and improving the simplicity (comprehensibility) of the rules. The basic idea of this rule pruning procedure is that, the lower the predictive power of a condition, the more likely the condition will be removed from the rule. The predictive power of a condition is estimated by computing its information gain, a very popular heuristic measure of predictive power in data mining [2]. This rule pruning procedure was chosen because it has been shown to be both effective and very computationally efficient in [12]. After rule pruning, the algorithm computes the fitness of each antibody, and then it performs the outer FOR loop over a fixed number of generations.

This outer FOR loop starts by performing T tournament selection (with tournament size of 10) procedures, in order to select T winner antibodies that will be cloned in the next step. Tournament selection is well-known and often used in evolutionary algo-

rithms [13]. Once T antibodies have been selected, the algorithm performs its core step, which is inspired by the clonal selection principle (discussed in Section 2). This step consists of several sub-steps, as follows. First, for each of the T antibodies to be cloned the algorithm produces C clones. The value of C is proportional to the fitness of the antibody. The function used to implement this procedure is shown in Equation (2), where *fit(Ab)* denotes the fitness of a given antibody *Ab* and *MaxNumCl* denotes the maximum number of clones for an antibody. As indicated in the last part of the equation (the "otherwise" condition), the number of clones increases linearly with the antibody fitness when $0 < Fit(Ab) < 0.5$, and any antibody with a fitness greater than or equal to 0.5 will have *MaxNumCl* clones. We set *MaxNumCl* to just 10 to prevent the clone population from being very large, which would not only be inefficient but also possibly lead to overfitting of the rules to the data.

$$NumCl = \begin{cases} 1 & if\ \ fit(Ab) \le 0 \\ MaxNumCl & if\ \ fit(Ab) \ge 5 \\ round\_to\_integer\left(\dfrac{MaxNumCl \times fit(Ab)}{5}\right) & otherwise \end{cases} \quad \textbf{(2)}$$

Next, each of the just-produced clones undergoes a process of hypermutation. This process follows [6], where the mutation rate is inversely proportional to the clone's fitness (i.e., the fitness of its "parent" antibody). In words, the lower the fitness (the worse a clone is), the higher its mutation rate. More precisely, the mutation rate for a given clone *cl*, denoted *mut_rate(cl)*, is given by Equation (3):

$$mute\_rate\ \ (cl) = \alpha + ((\beta - \alpha) \times (1 - fit\ (cl))) \quad \textbf{(3)}$$

where α and β are the smallest and greatest possible mutation rates, respectively, and *fit(cl)* is the fitness of clone *cl*. The fitness of a clone is a number normalized between 0 and 1, as will be explained later, so that the above equation collapses to α when the clone has the maximum fitness of 1, and it collapses to β when the clone has the minimum fitness of 0. In our experiments we have set α and β to 20% and 50%, respectively – empirically-determined values. These numbers represent the probability that each gene (rule condition) will undergo mutation. Once a clone has undergone hypermutation, its corresponding rule antecedent is pruned by using the previously-explained rule pruning procedure. Finally, the fitness of the clone is recomputed, using the current *TrainSet*.

The next step consists of population updating. More precisely, the T-worst fitness antibodies in the current population (not including the clones created by the clonal selection procedure) are replaced by the T best-fitness clones out of all clones produced by the clonal selection procedure. This keeps the population size constant at the end of each generation. The parameter T was set to 10 in our experiments. The population size was set to 50 and the number of generations was set to 50. These values were empirically determined.

Finally, the RE procedure returns, to the caller SC procedure, the best evolved rule, which will then be added to the set of discovered rules by the caller procedure. The best evolved rule consists of the rule antecedent ("IF part" of the rule) represented by the antibody with the best fitness, across all antibodies produced in all generations,

and of the rule consequent ("THEN part" of the rule) containing the class *c*, which was the class associated with all the antibodies created by the RE procedure.

We now turn to the fitness function used by the RE procedure. The fitness of an antibody *Ab*, denoted by *fit(Ab),* is given by Equation (4):

$$fit(Ab) = \frac{TP}{TP + FN} \times \frac{TN}{TN + FP} \quad \textbf{(4)}$$

Where the variables TP, FN, TN and FP have the following meaning:

- TP = number of true positives, i.e. number of examples satisfying the rule and having the same class as predicted by the rule;
- FN = number of false negatives, i.e. number of examples that do not satisfy the rule but have the class predicted by the rule;
- TN = number of true negatives, i.e. number of examples that do not satisfy the rule and do not have the class predicted by the rule;
- FP = number of false positives, i.e. number of examples that satisfy the rule but do not have the class predicted by the rule.

This fitness function was proposed by [14] and has also been used by other evolutionary algorithms for discovering classification rules. However, in most projects using this function the discovered rules are crisp, whereas in our project the rules are fuzzy. Hence, in this project the computation of the TP, FN, TN and FP involves, for each example, measuring the degree of affinity (fuzzy matching) between the example and the rule. Note that the same affinity function (Equation (1)) and the same procedure for determining whether or not an example satisfies a rule are used in both the SC and the RE procedures.

3.2 Using the Discovered Rules to Classify Examples in the Test Set

The rules discovered from the training set are used to classify new examples in the test set (unseen during training) as follows. For each test example, the system identifies the rule(s) activated for that example. Recall that a rule *j* is activated for example *k* if the affinity between *j* and *k* is greater than the affinity threshold for rule *j*.

When classifying a test example, there are three possible cases. First, if all the rules activated for that example predict the same class, then the example is simply assigned to that class. Second, if there are two or more rules predicting different classes activated for that example, the system uses a conflict resolution strategy consisting of selecting the rule with the greatest value of the product of the affinity between the rule and the example (Equation (1)) by the fitness of the rule (Equation (4)), i.e., it chooses the class *C* given by Equation (5):

$$C = C_j = \max_j \left(Afin(k, j) \times fit(j) \right) \quad \textbf{(5)}$$

Third, if there is no rule activated for the example, the example is classified by the "default rule", which simply predicts the most frequent class in the training set [2].

4. Computational Results

The proposed algorithm was evaluated in six public domain data sets: BUPA, CRX, Wisconsin Cancer, Votes, Hepatities, Ljubljana Cancer. These data sets are available from the UCI repository (*http://www.ics.uci.edu/~mlearn/MLRepository.html*). The experiments used a well-known method for estimating predictive accuracy, namely 5-fold cross-validation [2].

Table 1 shows the number of continuous and categorical attributes for each data set (recall that only continuous attributes are fuzzified), as well as the average accuracy rate on the test set computed by the cross-validation procedure. The numbers after the "±" symbol are the standard deviations. Note that the Votes data set does not have any continuous attribute to be fuzzified. This data set was included in the experiments to evaluate IFRAIS' performance in the "degenerated" case of discovering crisp rules only. The other data sets have 6 or 9 continuous attributes that are fuzzified by IFRAIS. The accuracy rate is shown for IFRAIS and for C4.5Rules, a very popular data mining algorithm for discovering (crisp) classification rules [15]. For each data set, the highest accuracy rate between the two algorithms is shown in bold.

Table 1. Characteristics of data sets and accuracy rate on the test set

Data Sets	Number of Attributes		IFRAIS	C4.5Rules
	Continous	Categorical		
Crx	6	9	86.29 ± 0.91	**90.22 ± 1.59**
Bupa	6	0	56.22 ± 2.44	**67.40 ± 1.60**
Hepatitis	6	13	**78.66 ± 1.70**	76.32 ± 2.79
Votes	0	16	**95.61 ± 0.86**	94.82 ± 0.82
Wisconsin	9	0	**95.75 ± 0.96**	95.32 ± 1.09
Ljubljana	9	0	**70.18 ± 3.97**	68.80 ± 4.45

In Table 1, IFRAIS obtained higher accuracy than C4.5Rules in four out of the six data sets, but the differences in accuracy rate are not significant – since the accuracy rate intervals (based on the standard deviations) overlap. C4.5Rules obtained a higher accuracy than IFRAIS in only two data sets (Crx and Bupa), and the difference was significant in both cases – since the accuracy rate intervals do not overlap. The reason for this seems to be that the rule sets discovered by IFRAIS in these two data sets were too simple, and so were underfitted to the data.

Table 2. Simplicity of the discovered rule set

Data Sets	# Rules		# Conditions	
	IFRAIS	C4.5 Rules	IFRAIS	C4.5 Rules
Crx	**7.2 ± 0.20**	15.6 ± 1.12	**19,4 ± 0,24**	63.0 ± 4.03
Bupa	**7.2 ± 0.37**	11.2 ± 1.65	**13.0 ± 1.30**	36.6 ± 6.16
Hepatitis	**4.8 ± 0.20**	5.0 ± 0.44	**10.0 ± 0.70**	10.4 ± 1.53
Votes	**3.4 ± 0.24**	5.6 ± 0.67	**4.0 ± 0.63**	14.4 ± 2.82
Wisconsin	**6.4 ± 0.40**	6.6 ± 0.50	**10.2 ± 1.01**	14.4 ± 1.28
Ljubljana	6.2 ± 3.37	**5.0 ± 0.54**	**11.2 ± 1.06**	15.2 ± 3.08

Table 2 shows the results of both IFRAIS and C4.5Rules with respect to the simplicity of the discovered rule set, measured by the average number of discovered rules and the average total number of conditions in all discovered rules. (Recall that the averages were computed over the five iterations of the cross-validation procedure.) With respect to this rule quality criterion, the results obtained by IFRAIS were much better than the results obtained by C4.5Rules in all data sets. (In the Ljubljana cancer data set, although C4.5Rules discovered a slight smaller number of rules, IFRAIS still discovered a significantly simpler data set, as shown by the total number of conditions.)

5 Conclusions and Future Research

This work proposed a new AIS, called IFRAIS, for discovering fuzzy classification rules. IFRAIS uses a sequential covering procedure that has two important differences with respect to the standard procedure used by conventional rule induction algorithms [2]. First, it stops the covering process when the number of uncovered examples has been reduced to a very low number (smaller than the *MaxUncovExamp* threshold), rather than stopping only when the number of uncovered examples has reached zero, as usual. The motivation for this modification was to avoid overfitting of a rule to just a very small number of examples, where there are not enough examples for a reliable induction of a new rule. In addition, this modification helped to produce simpler rule sets (as shown in Table 2), more easily interpreted by a human user, since it avoids the generation of one or more very specific rules covering very few examples. This modification led to improved results in our preliminary experiments and it was also successfully used in [16].

The second difference was the use of an affinity function and an affinity threshold to decide whether or not an example satisfies a rule. Of course, this modification is not necessary in conventional (crisp) rule induction algorithms, where the matching between an example and a rule is always binary. However, this is fundamental in our case, where the rules have fuzzy antecedents, producing degrees of matching between 0 and 1. We recognize that the value of the affinity threshold can potentially have a significant impact in the performance of the algorithm, and that it is difficult to determine the "optimal" value of this parameter without many experiments. Actually, there is no strong reason to believe that the "optimal" value of this parameter should be the same for all rules. Hence, we developed an adaptive mechanism for choosing the best affinity threshold for each rule – i.e., choosing the affinity threshold that maximizes the fitness of the rule, out of a wide range of possible affinity thresholds. This adaptive mechanism increases the autonomy of the system and relieves the user from the difficult task of adjusting this threshold, whose value has a significant impact in the predictive accuracy of the system.

IFRAIS was compared with C4.5Rules in 6 real-world data sets. IFRAIS obtained classification accuracies slightly better than C4.5Rules in 4 data sets, but classification accuracies significantly worse than C4.5Rules in 2 data sets. However, IFRAIS discovered rule sets were much simpler than the rule sets discovered by C4.5Rules in virtually all the 6 data sets. This is particularly important in the context of data mining, where knowledge comprehensibility is very important [1][2] – since discovered knowledge is supposed to be interpreted by the user, as discussed in the Introduction.

In addition, it should be recalled that C4.5Rules is the result of decades of research in decision tree and rule induction algorithms, whereas IFRAIS is still in its first version – and the whole area of AIS is still relatively new.

Some directions for future research, which might improve the predictive accuracy of IFRAIS, are as follows. First, the system could automatically determine the number of linguistic terms for each continuous attribute, rather than just using a fixed number as in the current version. Second, when there are two or more rules predicting different classes activated for a test example, the system could use all the activated rules to compute a predicted class, rather than just choosing the best rule as in the current version. Third, one could develop variations of the rule evolution procedure, which is based on the clonal selection principle, in order to investigate different trade-offs between exploration and exploitation in the search for rules.

References

1. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P.: From Data Mining to Knowledge Discovery: an Overview. In: Fayyad, U.M. et al (Eds.) Advances in Knowledge Discovery and Data Mining. AAAI/MIT (1996) 1-34
2. Witten, I. H., Frank, E.: Data Mining: Pratical Machine Learning Tools and Techniques with Java Implementation. Morgan Kaufmann, San Mateo (2000)
3. Dasgupta, D.: Artificial Immune Systems and Their Applications. Springer, Berlin (1999)
4. Zadeh, L.A.: Fuzzy Sets. Inform. Control. 9 (1965) 338-352
5. Pedrycz, W., Gomide, F.: An Introduction to Fuzzy Sets. Analysis and Design. MIT Press, Cambridge (1998)
6. Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computation Intelligence Approach. Springer-Verlag, Berlin (2002)
7. Ishibuchi, H., Nakashima, T.: Effect of Rule Weights in Fuzzy Rule-based Classification Systems. IEEE T. Fuzzy Syst. 9:4 (2001) 506-515
8. Watkins, A.B., Boggess, L.C.: A Resource Limited Artificial Immune Classifier. Proc. Congress on Evolutionary Computation (2002) 926-931
9. Gonzales, F.A., Dasgupta, D.: An Immunogenetic Technique to Detect Anomalies in Network Traffic. Proceedings of Genetic and Evolutionary Computation. Morgan Kaufmann, San Mateo (2002) 1081-1088
10. Freitas, A.A., Timmis, J.: Revisiting the Foundations of Artificial Immune Systems: a Problem-oriented Perspective. Proc. 2nd International Conference on Artificial Immune Systems. Lecture Notes in Computer Science, Vol. 2787. Springer-Verlag, Berlin (2003) 229-241
11. Nasaroui, O., Gonzales, F., Dasgupta, D.: The Fuzzy Artificial Immune System: motivations, Basic Concepts, and Application to Clustering and Web Profiling. Proceedings of IEEE International Conference on Fuzzy Systems (2002) 711-716
12. Carvalho, D.R., Freitas, A.A.: A genetic Algorithm with Dequential Niching for Discovering Small-disjunct Rules. Proceedings of Genetic and Evolutionary Computation, Morgan Kaufmann, San Mateo (2002) 1035-1042
13. Back, T., Fogel, D.B., and Michalewicz, T. (Eds.): Evolutionary Computation, Vol. 1. IoP Publishing, Oxford, UK (2000)
14. Lopes, H.S., Coutinho, M.S., Lima, W.C.: An Evolutionary Approach to Simulate Cognitive Feedback Learning in Medical Domain. In: Sanchez, E., Shibata, T., Zadeh, L.A. (eds.), Genetic Algorithms and Fuzzy Logic Systems. World Scientific, Singapore (1997) 193-207
15. Quinlan, J.R.: C4.5: Programs For Machine Learning. Morgan Kaufmann, San Mateo, 1993
16. Parpinelli, R.S., Lopes, H.S., Freitas, A.A.: Data Mining With an Ant Colony Optimization Algorithm. IEEE T. Evol. Comput. 6:4 (2002) 321-332