Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 155–159
ISBN 978-83-60810-14-9
ISSN 1896-7094

Accuracy Boosting Induction of Fuzzy Rules with Artificial Immune Systems

Adam Kalina
Value Based Advisors Sp. z o.o.
ul. Połabian 35, 52-339 Wrocław, Poland
Adam.Kalina@vba.pl

Edward Mężyk
and Olgierd Unold
Institute of Computer Engineering, Control and Robotics
Wroclaw University of Technology
Wyb. Wyspianskiego 27, 50-370 Wrocław, Poland
Olgierd.Unold@pwr.wroc.pl

***Abstract*—The paper introduces accuracy boosting extension to a novel induction of fuzzy rules from raw data using Artificial Immune System methods. Accuracy boosting relies on fuzzy partition learning. The modified algorithm was experimentally proved to be more accurate for all learning sets containing non-crisp attributes.**

I. Introduction

FUZZY-BASED data mining is a modern and very promising approach to mine data in an efficient and comprehensible way. Moreover, fuzzy logic [8] can improve a classification task by using fuzzy sets to define overlapping class definitions. This kind of data mining algorithms discovers a set of rules of the form "IF (fuzzy conditions) THEN (class)," whose interpretation is as follows: IF an example's attribute values satisfy the fuzzy conditions THEN the example belongs to the class predicted by the rule. The automated construction of fuzzy classification rules from data has been approached by different techniques like, e.g., neuro-fuzzy methods, genetic-algorithm based rule selection, and fuzzy clustering in combination with other methods such as fuzzy relations and genetic algorithm optimization (for references see [10]).

A quite novel approaches, among others, integrate Artificial Immune Systems (AISs) [3] and Fuzzy Systems to find not only accurate, but also linguistic interpretable fuzzy rules that predict the class of an example. The first AIS-based method for fuzzy rules mining was proposed in [2]. This approach, called IFRAIS (Induction of Fuzzy Rules with an Artificial Immune System), uses sequential covering and clonal selection to learn IF-THEN fuzzy rules. In [7] the speed of IFRAIS was improved significantly by buffering dicovered fuzzy rules in a clonal selection. One of the AIS-based algorithms for mining IF-THEN rules is based on extending the negative selection algorithm with a genetic algorithm [4]. Another one is mainly focused on the clonal selection and so-called a boosting mechanism to adapt the distribution of training instances in iterations [1]. A fuzzy AIS was proposed also in [6], however that work addresses not the task of classification, but the task of clustering.

This paper seeks to boost an accuracy of IFRAIS approach by exploring the use of fuzzy partitions learning.

II. IFRAIS

Data preparation for learning in IFRAIS consists of the following steps: (1) create a fuzzy variable for each attribute in data set; (2) create class list for actual data set; (3) and compute information gain for each attribute in data set.

Listing 1. Sequential covering algorithm

```
Input: full training set
Output: fuzzy rules set

rules set = 0
FOR EACH class value c in class values list DO
  values count = number of c in full training set
  training set = full training set
  WHILE values count > number of maximal uncovered
      examples AND
    values count > percent of maximal uncovered
      examples
    rule = CLONAL-SELECTION-ALGORITHM(training set,
      c)
    covered  = COVER-SET(training set, rule)
    training set = training set / covered with rule
      set
    values count = values count - size of covered
    ADD(rules set, rule)
  END WHILE
END FOR EACH
training set = full training set
FOR EACH rule R in rules set DO
  MAXIMIZE-FITNESS(R, training set)
  COMPUTE-FITNESS(R, training set)
END FOR EACH
RETURN rules set
```

IFRAIS uses a sequential covering as a main learning algorithm (see Listing 1). In the first step a set of rules is initialized as an empty set. Next, for each class to be predicted the algorithm initializes the training set with all training examples and iteratively calls clonal selection procedure with the parameters: the current training set and the class to be predicted. The clonal selection procedure returns a discovered rule and next the learning algorithm adds the rule to the rule set and removes from the current training set the examples that have been correctly covered by the evolved rule.

Clonal selection algorithm is used to induct rule with the best fitness from training set (see Listing 2). Basic elements of this method are antigens and antibodies which refers directly to biological immune systems. Antigen is an example from

data set and antibody is a fuzzy rule. Similarly to fuzzy rule structure, which consists of fuzzy conditions and class value, antibody comprises genes and informational gene. Number of genes in antibody is equal to number of attributes in data set. Each gene consists of a fuzzy rule and an activation flag that indicates whether fuzzy condition is active or inactive.

Listing 2. Clonal selection algorithm in IFRAIS (based on [2])

```
Input: training set, class value (c)
Output: fuzzy rule

CREATE randomly antibodies population with size s
    and class value c
FOR EACH antibody A in antibodies population
  PRUNE(A)
  COMPUTE–FITNESS(A, training set)
END FOR EACH
FOR i=1 TO number of generations n DO
  WHILE clones population size < s−1
    antibody to clone = TOURNAMENT–SELECTION(
        antibodies population)
    clones = CREATE x CLONES(antibody to clone)
    clones population = clones population + clones
  END WHILE
  FOR EACH clone K in clones population
    muteRatio = MUTATION–PROBABILITY(K)
    MUTATE(K, muteRatio)
    PRUNE(K)
    COMPUTE–FITNESS(K, training set)
  END FOR EACH
antibodies population = SUCCESSION(antibodies
    population, clones population)
END FOR
result = BEST–ANTIBODY(antibodies population)
RETURN result
```

In the first step the algorithm generates randomly antibodies population with informational gene equals to class value *c* passed in algorithm parameter. Next each antibody from generated population is pruned. Rule pruning has a twofold motivation: reducing the overfitting of the rules to the data and improving the simplicity (comprehensibility) of the rules [11]. Fitness of the rule is computed according to the formula

$$FITNESS(rule) = \frac{TP}{TP+FN} \cdot \frac{TN}{TN+FP} \qquad (1)$$

where TP is number of examples satisfying the rule and having the same class as predicted by the rule; FN is the number of examples that do not satisfy the rule but have the class predicted by the rule; TN is the number of examples that do not satisfy the rule and do not have the class predicted by the rule; and FP is the number of examples that satisfy the rule but do not have the class predicted by the rule. Since the rules are fuzzy, the computation of the TP, FN, TN and FP involves measuring the degree of affinity between the example and the rule. This is computed by applying the standard aggregation fuzzy operator *min*

$$AFFINITY(rule, example) = min_{i=1}^{condCount}(\mu_i(att_i)) \qquad (2)$$

where $\mu_i(att_i)$ denotes the degree to which the corresponding attribute value att_i of the example belongs to the fuzzy set accociated with the *i*th rule condition, and *condCount* is the number of the rule antecedent conditions. The degree of membership is not calculated for an inactive rule condition, and if the *i*th condition contains a negation operator, the membership function equals to $(1 - \mu_i(att_i))$ (complement). An example satisfies a rule if $AFFINITY(rule, example) > L$, where L is an activation threshold. Next, antibody to be cloned is selected by tournament selection from the antibodies population. For each antibody to be cloned the algorithm produces *x* clones. The value of *x* is proportional to the fitness of the antibody. Next, each of the clones undergoes a process of hypermutation, where the mutation rate is inversely proportional to the clone's fitness. 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 training set. In the last step the *T*-worst fitness antibodies in the current population are replaced by the *T* best-fitness clones out of all clones produced by the clonal selection procedure. Finally, the clonal selection procedure returns the best evolved rule, which will then be added to the set of discovered rules by the sequential covering. More details of the IFRAIS is to be found in [2].

III. Fuzzy partition inference

IFRAIS, as an Artificial Immune System evolves a population of antibodies representing the IF part of a fuzzy rule, whereas each antigen represents an example. As was stated, each rule antecedent consists of a conjunction of rule condition. In IFRAIS approach three and only three linguistic terms (low, medium, high) are associated with each continuous attribute. Each linguistic term is represented by the triangular membership functions (see Fig 1). It seems to be purposeful to infer a fuzzy partition for each continuous attribute over the data set instead of stiff, and the same for different attributes partitioning. There exist various methods to learn a fuzzy partions over a set of data [5]. We consider using a clonal selection algorithm to automatic infer partitions for each attribute, both crisp and continuous one. In such an approach a population of antibodies represent a set of partitions, and an antigen is a whole set of data.

A. Representation

Each partition is represented by two crisp tables: a size table S and a range table R. S is one-dimensional table $S = \{s_1, s_2, \ldots, s_{setsCount}\}$, where $setsCount$ is the number of fuzzy sets the partition is consisted of (the number of ligustic terms). $setCount$ is drawn from the range $[3, 12]$ during inference. s_i is a size of a fuzzy set and is expressed in so-called "'division points"' pv. For a crisp attribute pv corresponds to one attribute value. For an attribute to be fuzzified pv is computed as follows

$$pv = \frac{maxValue - minValue}{apc} \qquad (3)$$

where $maxValue$ is a maximal attribute value, $minValue$ is a minimal attribute value, and $apc = setsCount \cdot pc$ defines

the number of pv-points allocated to the attribute. pc is an algorithm parameter set to 10, and pc is interpreted as a number of pv-points assigned to one fuzzy set of the partition. While fuzzy partition infering, the size of one fuzzy set s_i is drawn from a range $[1, apc - apc \cdot apcp]$, where $apcp$ is a percent of apc ($apcp$ is a program parameter set to 10%).

A range table R is two-dimensional table contains pairs $R = \{(r_{l_1}, r_{r_1}), (r_{l_2}, r_{r_2}), \ldots, (r_{l_{setsCount}}, r_{r_{setsCount}})\}$, where a pair (r_{l_i}, r_{r_i}) is a range expressed in pv for which triangle fuzzy set (linguistic term) will be created. r_{l_i} is a lower bound and r_{r_i} is upper bound of the ith range. For crisp attributes each cell of the R table contains induced list of attribute values. For example, partition of unfuzzy attribute FALLOUT, could be divided to 3 fuzzy sets $S = \{1, 1, 2\}$ with following value list $R = \{(snow), (rain), (hail, no fallout)\}$. The range for triangle fuzzy set is calculated on the basis of a size of fuzzy set s_i, pv, $minValue$, $maxValue$, $r_{r_{i-1}}$, and point overflow index oi. oi is calculated as a difference between a sum of all values in S table and apc. In the case when $oi > 0$, an overalap index ovi is randomly generated from the range $[1, oi]$ to prevent lack of attribute full covering by lingustic terms. If $oi > r_{r_{i-1}}$, then ovi is drawn from the range $[1, r_{r_{i-1}}]$.

To the first lower bound r_{l_0} is assigned $minValue$. The first upper bound r_{r_0} equals the size of a fuzzy set s_1 multiplied by pv value. Every next pair in the R table is calculated according to the following rules: if $oi > 0$ then draw ovi from $[1, oi]$ or from $[1, r_{r_{i-1}}]$ if $oi > r_{r_{i-1}}$. Otherwise ovi is equal to 0. The lower bound $r_{l_i} = ovi \cdot pv$, whereas the upper bound $r_{r_i} = r_{l_i} + s_i \cdot pv$. If $r_{r_i} > maxValue$ then r_{r_i} is truncated to $maxValue$. Figure 2 ilustrates the one of the fuzzy partion learning steps for an attribute TIME. The values from $S = \{15, 20, 5, 10, 25\}$ are represented by double side arrows. ovi values are represented by one side arrows. R table goes as follows $R = \{(0, 15), (5, 25), (23, 28), (25, 35), (25, 50)\}$.

B. Operations

Fuzzy variables ready to use by IFRAIS are created from the range table R. All partition modifications are made over the size table S, from which table R is derived according to the mentioned above rules. Fuzzy partitions are evaluated by using following operations:

strong splitting—removes randomly number χ of pv-points from randomly chosen cell s_i, and creates at randomly chosen position in S new cell including χ-points,

light splitting—removes only one pv-point form randomly chosen cell s_i, and creates at randomly chosen position in S new cell including one point,

strong joining—removes randomly number χ of pv-points from randomly chosen cell s_i, and adds χ-points to the randomly chosen cell,

light joining—removes only one pv-point from randomly chosen cell s_i, and adds one points to the randomly chosen cell.

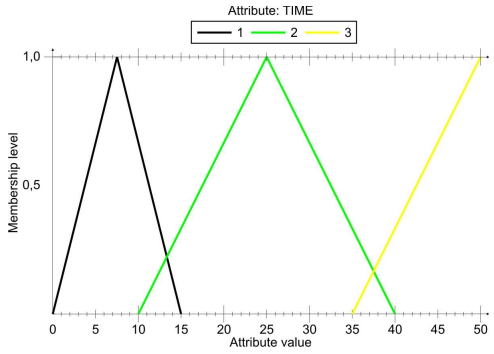

Fig. 1. An example of fuzzy partition for TIME attribute

C. Inferring

Fuzzy partition inferring is based on clonal selection algorithm (see Listing 3). In this algorithm it is worth underlining a clone mutation which undergoes for all clones in a population. While mutating, only the sizes of fuzzy sets (table S) for choosen attribute are generated according to the mentioned above rules, but without changing the number of liguistic terms (the size of table S). After a new partition generation, the operations (strong and light splitting, strong and light joining) are fired, each with the probability $muteRatio$

$$muteRatio = min + \frac{(max - min) \cdot (1 - f + No)}{2} \quad (4)$$

where min and max are minimal and maximal probability respectively (algorithm parameters), No is a number taken at random from $[0, 1]$, and f is a normalized clone fitness before mutation.

Listing 3. Fuzzy partition learning

```
Input: training data set, IFRAIS system parameters,
    algorithm parameters
Output: fuzzy partitions set

Randomly generate antibodies population of size s
FOR EACH antibody A in antibodies population
  COMPUTE–FITNESS(A, training data set)
END FOR EACH
FOR i=1 TO number of generations n DO
  SORT–DESCENDING(antibodies population)
  WHILE clones population size < maximal clones
      population size
    antibody to clone = TAKE–NEXT–BEST(antibodies
      population)
    clones = CREATE x CLONES(antibody to clone)
    clones population = clones population + clones
  END WHILE
  FOR EACH clone K in clones population
    MUTATE (K)
    COMPUTE–FITNESS(K, training data set)
  END FOR EACH
  Antibodies population = SUCCESION(antibodies
    population, clones population)
END FOR
result = BEST–ANTIBODY(antibodies population)
RETURN result
```

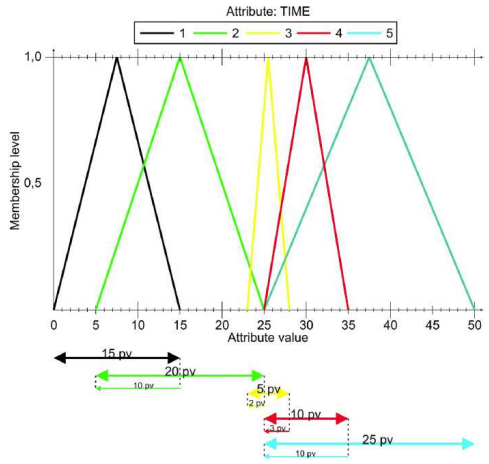

Fig. 2. Induced fuzzy partition for TIME attribute

TABLE I
DATA SETS AND NUMBER OF ROWS, ATTRIBUTES, CONTINUOUS ATTRIBUTES, AND CLASSES

Data set	#Rows	#Attrib.	#Cont.	#Class.
Bupa	345	6	6	2
Crx	653	15	6	2
Hepatitis	80	19	6	2
Ljubljana	277	9	9	2
Wisconsin	683	9	9	2
Votes	232	16	0	2

IV. EXPERIMENTAL RESULTS

In order to evaluate the performance of the speed boosting extensions, both IFRAIS and improved IFRAIS were applied to 6 public domain data sets available from the UCI repository (http://archive.ics.uci.edu/ml/datasets.html):

- Bupa (Liver+Disorders)
- Crx (Credit+Approval)
- Hepatitis (Hepatitis)
- Lubljana (Breast+Cancer)
- Votes (Congressional+Voting+Records)
- Winsconsin (Breast+Cancer+Wisconsin+(Original))

The experiments were conducted using a Distribution-Balanced Stratified Cross-Validation [12], which is a one of the version of well-known k-fold cross-validation, and improves the estimation quality by providing balanced intraclass distributions when partitioning a data set into multiple folds. Additionally, both IFRAIS method were compared to C4.5, well-known data mining algorithm for discovering classification rules [9] (results of C4.5 taken from [2]).

Table 1 shows the number of rows, attributes, continuous attributes, and classes for each data set. Note that only continuous attributes are fuzzified. The Votes data set does not have any continuous attribute to be fuzzified, whereas the other data sets have 6 or 9 continuous attributes that are fuzzified by IFRAIS. All experiments with IFRAIS were repeated 50-times using 5-fold cross-validation. Table 2 shows for each data set the average accuracy rate with standard deviations,

TABLE II
ACCURACY RATE ON THE TEST SET

Data set	C4.5	IFRAIS	Boosted IFRAIS
Bupa	67.40±1.60	58.38±0.78	72.34±0.99
Crx	90.22±1.59	86.03±0.26	87.37±0.08
Hepatitis	76.32±2.79	77.25±1.71	93.87±2.65
Ljubljana	68.80±4.45	69.55±1.09	74.70±0.79
Wisconsin	95.32±1.09	94.91±0.39	97.39±0.28
Votes	94.82±0.82	96.98±0.00	96.98±0.00

both for IFRAIS and IFRAIS extended by fuzzy partition learning (boosted IFRAIS), as well as for C4.5 method. As shown in Table 2 the boosted IFRAIS obtained better accuracy rates than standard IFRAIS for all data set but one (the Votes data set comprises only crisp attributes). For Bupa set the average gain is ca 14 %, and for Hepatitis even more (16.6 %)! The improved IFRAIS obtained higher accuracy than C4.5 in five out of the six data sets. C4.5 obtained a higher accuracy than IFRAIS in only one data set (Crx), but the differences in accuracy rate is not significant, since the accuracy rate intervals (based on the standard deviations) overlap.

V. CONCLUSION

The accuracy boosting extension was introduced to the IFRAIS algorithm—an AIS-based method for fuzzy rules mining. Boosting uses the fuzzy partion learning based on the clonal selection. The partition inferring improves significantly effectiveness of an algorithm. Although the proposed improvements increase the accuracy of the whole algorithm, it is worth noticing that the learning of fuzzy partition is additional time-consuming procedure.

It seems to be still possible to improve the Induction of Fuzzy Rules with Artificial Immune Systems, and not only considering the effectiveness of the induced fuzzy rules but also time of working. These two goals could be achieved mostly by modifying the fitness function to reinforce the fitness of high-accuracy rules, as in [1]. We also consider changing the triangular membership functions to various more sophisticated functions and manipulating all system parameters to obtain higher quality results. We have performed preliminary experiments, in which speed and accuracy boosting IFRAIS with modified fitness function and parameters is trained on well known data sets. The results are very promising.

ACKNOWLEDGMENT

The authors would like to thank Dr. Robert Alves for making available to them his source code of IFRAIS.

REFERENCES

[1] Alatas, B., Akin, E.: Mining Fuzzy Classification Rules Using an Artificial Immune System with Boosting. In: *Eder, J. et al. (eds.) ADBIS 2005*. LNCS, vol. 3631, pp. 283–293. Springer-Verlag Berlin Heidelberg (2005).

[2] Alves, R. T., et al.: An artificial immune system for fuzzy-rule induction in data mining. In: *Yao, X., et al (eds.) Parallel Problem Solving from Nature—PPSN VIII*. LNCS, vol. 3242, pp. 1011–1020. Springer Heidelberg (2004).
[3] Dasgupta, D.(ed.): Artificial Immune Systems and Their Applications. Spring-Verlag Berlin Heidelberg Germany (1999).
[4] Gonzales, F. A., Dasgupta, D.: An Immunogenetic Technique to Detect Anomalies in Network Traffic. In: *Proceedings of Genetic and Evolutionary Computation.* pp. 1081–1088. Morgan Kaufmann San Mateo (2002).
[5] Marsala C.: Fuzzy Partitioning Methods, Granular Computing: An Emerging Paradigm. Physica-Verlag GmbH Heidelberg Germany, pp. 163–186 (2001).
[6] Nasaroui, O., Gonzales, F., Dasgupta, D.: The Fuzzy Artificial Immune System: Motivations, Basic Concepts, and Application to Clustering and Web Profiling. In: *Proceedings of IEEE International Conference on Fuzzy Systems*, pp. 711–716 (2002).
[7] Mężyk E., Unold O.: Speed Boosting Induction of Fuzzy Rules with Artificial Immune Systems. In: *Mastorakis, E. M. et al. (eds) Proc. of the 12th WSEAS International Conference on SYSTEMS*, Heraklion Greece July 22-24, pp. 704–706 (2008).
[8] Pedrycz, W., Gomide, F.: An Introduction to Fuzzy Sets. Analysis and Design. MIT Press Cambridge (1998).
[9] Quinlan, J. R.: C4.5: Programs For Machine Learning. Morgan Kaufmann San Mateo (1993).
[10] Roubos J. A., Setnes M. Abonyi J.: Learning fuzzy classification rules from labeled data. Information Science 150, pp. 77–93 (2003).
[11] Witten, I. H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. 2nd edn. Morgan Kaufmann San Mateo (2005).
[12] Zeng X., Martinez T. R.: Distribution-Balanced Stratified Cross-Validation for Accuracy Estimations. *Journal of Experimental and Theoretical Artificial Intelligence.* Vol. 12, number 1, pp. 1–12. Taylor and Francis Ltd (2000).