Learning maximal structure rules with pruning based on distances between fuzzy sets

Javier Albusac
Escuela Sup. de Informática
Univ. de Castilla-La Mancha
Paseo de la Universidad nº4
Ciudad Real, Spain
JavierAlonso.Albusac@uclm.es

J.J Castro-Schez
Escuela Sup. de Informática
Univ. de Castilla-La Mancha
Paseo de la Universidad nº4
Ciudad Real, Spain
JoseJesus.Castro@uclm.es

David Vallejo
Escuela Sup. de Informática
Univ. de Castilla-La Mancha
Paseo de la Universidad nº4
Ciudad Real, Spain
David.Vallejo@uclm.es

Abstract

The aim of this paper is to present a new fuzzy learning algorithm to generate IF-THEN rules, for classifying instances of one application domain. Really, this algorithm is a modification that improves the results offered by previously presented algorithm. In addition, the more common classification problems of the original algorithm are presented and a measure to determine the conflicts among generated rules is introduced.

Keywords: Inductive Learning, Fuzzy Logic, Expert System.

1 Introduction

One of the more important characteristics of an Expert System (ES) is simulating the expert behavior in a concrete domain. Normally, an ES is not designed for replacing completely to a human expert, whereas these systems are used by human experts to improve their productivity. This type of systems consists of a (i) knowledge base (KB), in which the domain knowledge is modeled; (ii) a set of facts related to the problem; and finally, (iii) an inference engine to model human reasoning expert. ES are often classified into rule-based, case-based, and Bayesian network based systems.

Concretely, a rule-based ES has a knowledge base with a set of rules used by the inference engine. Normally, the definition of these rules is made by a human expert. When the domain to define is complex and a high number of rules is needed, the KB building is a difficult and tedious task. In this case, learning algorithms may be a good way to improve productivity and efficiency.

In this paper, we present a modification of one inductive learning algorithm [1] based on the distance between fuzzy sets. This version solves some problems presented in the original algorithm, also improved in other works [2, 3]. The main advantages of this algorithm are the generation of high interpretable rules and low error percentages when classifying. This type of algorithms have been studied by other authors [8, 9, 10]. Herrera F. et al [8] presented a genetic learning process for learning fuzzy control rules from examples, developed in three stages: rule generation, combination, and adjustment. Song, Q. and Kasabov, NK [9] introduced a novel neural fuzzy inference method-NFI for transductive reasoning systems. Serrurier, M. et al [10] used fuzzy sets arid fuzzy implication connectives to increase the expressive power of the induced rules while keeping the readability of the rules. All these works make use of fuzzy logic and learn rules for different purposes.

The remainder of this paper is structured in the following way. In Section 2, we describe the aim of the fuzzy learning algorithm and its main steps. Besides, the main errors in the classification process are discussed. This section ends with the description of a measure to determine the conflicts among the rules learned by the algorithm. In Section 3, a modification of the learning algorithm based on distance among fuzzy rules is presented. In Section 4, we compare the results obtained with the original algorithm and the modified version. Finally, conclusions are presented in Section 5.

2 Fuzzy Learning Algorithm

In previous works, the inductive fuzzy algorithm [1] was successfully applied to the public Iris Data Base. The algorithm provided different results depending whether it worked with ambiguity or, on the contrary, ambiguity was dealt. In the last case, the results improved noticeably. Furthermore, these results were compared with the results obtained by the CART algorithm, being the results obtained by the fuzzy algorithm better than the results of the CART algorithm.

The aim of the algorithm is to obtain a set of fuzzy rules for classifying entities belonging to a domain. To achieve this aim, an expert must provide a set of variables $V = \{v_1, v_2, ..., v_n\}$ and their definition domain $DDV = \{DDV_1, DDV_2, ..., DDV_n\}$. Each DDV_i from DDV is defined as the set $DDV_i = \{L_{1i}, L_{2i}, ..., L_{mi}\}$ and it contains all possible values that the variable v_i (with L_{ji} being the linguistic label of the value j in the variable v_i defined by means of a trapezoidal function with parameters $(a_{ji}, b_{ji}, c_{ji}, d_{ji})$) can take. Moreover, there is a set of examples ε=$\{e_1, e_2, ..., e_n\}$, where each example has the structure $e_i = ((x_{i1}, x_{i2}, ..., x_{in}), o_i)$; being $x_{i1}, x_{i2}, ..., x_{in}$ the values of the input variables belonging to V and o_i is the class that the example belongs. V, DDV, and the set of examples is the input of the algorithm whereas o_i corresponds with the output. The form of the fuzzy rules generated is :

IF ν_1 is ZD_1 and ν_2 is ZD_2 and ... and ν_n is ZD_n **THEN** choice = O_x,

where each ZD_i is a set of values disjunctively associated with the variable ν_i and they are taken from its definition domain DDV_i, verifying that $ZD_i \subseteq DDV_i$. Another way to write this rule is $((ZD_{1i}, ZD_{2i}, ..., ZD_{ni}),O_x)$.

The algorithm consists of two main steps. First, it is necessary *"to convert the example in particular rules"*. That is to say, each example of the training set is translated into one rule in which the value of each input variable is represented by means of a label, and the class is encoded. The second step is *"to construct the set of definitive or maximal rules"* (the set of fuzzy rules that identifies the system is generated), which is carried out by means of an amplification process. In Section 3 the algorithm steps are described with more detail.

The amplification (generalization) process allows us to generate definitive or maximal rules. However, these rules share a high number of regions, and therefore, a high number of conflicts. The conflicts between rules can difficult the task of classifying an input element in the correct class. For this reason, it is interesting to suggest possible improvements of the algorithm to reduce the number of conflicts between rules and, in consequence, the number of errors in the classification process (see Figure 1).

2.1 Possible errors in the classification process

In the original algorithm [1], there are several reasons which can cause errors in the classification process.

The first type of error happens when there is no rule that classifies one input example. It is an unusual fact because one of the main objectives of the algorithm is to build up general rules. This way, examples that have similar but not equal properties can be classified by the same rule.

The second type of error takes place when an example fires several rules of different classes (due to the generalization) and there are several maximum coincident degrees of convenience. In this case, the system is not able to classify the input example.

Example 2.1. Let the rules R_i*: if* v_0 *is* $\{A$ *or* $B\}$ *and* v_1 *is* $\{C\}$ *then* $CLASS_A$ and R_j*: if* v_0

is $\{B\}$ *and* v_1 *is* $\{B$ *or* $C\}$ *then* $CLASS_B$. If the input example (B, C) fires the rules R_i and R_j with the same degree of convenience, then the system is not able to classify the input example in one output class (see Figure 1).

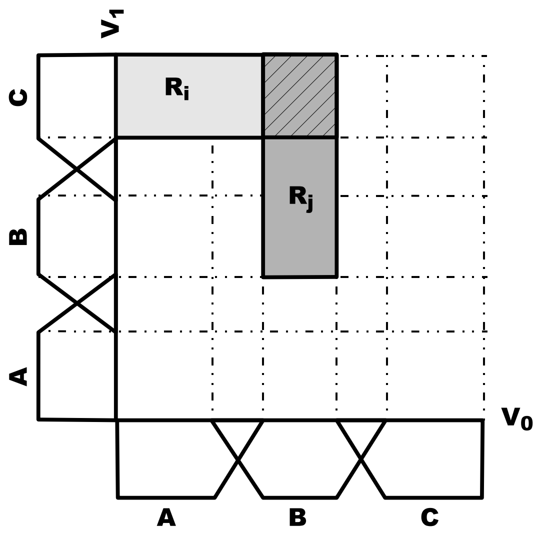

Figure 1: Conflicts between rules

Finally, the third type of error happens when the system classifies an input example in a wrong class. This situation occurs when one example fires a rule of another class and the degree of convenience is maximum.

Example 2.2. Let the rules R_i: *if* v_0 *is* $\{A$ *or* $B\}$ *and* v_1 *is* $\{C\}$ *then* $CLASS_A$ and R_j: *if* v_0 *is* $\{B\}$ *and* v_1 *is* $\{C\}$ *then* $CLASS_B$. If The input example ((B,C),$CLASS_A$) fires the rules R_i with a degree of convenience of 0.4 and R_j with 0.8, then the system classifies it on $CLASS_B$ being $CLASS_A$ the correct class.

2.2 Measure to determine conflicts among rules

The second and the third type of errors exposed in Section 2.1 have their causes in the conflicts among rules. Therefore, it would be interesting to design a measure for detecting possible collisions among rules. This measure may be used to see how the conflicts between rules increase or decrease when a new modification of the algorithm is applied. In this section, we propose a measure to determine the conflicts that may exist between rules.

Definition 2.1. Let us assume that R_i and R_j are two decision rules acquired from the learning algorithm:

R_i: If ν_1 is ZD_{1i} and ν_2 is ZD_{2i} and ... and ν_n is ZD_{ni} THEN the choice is o_x, R_j: IF ν_1 is ZD_{1j} and ν_2 is ZD_{2j} and ... and ν_n is ZD_{nj} then the choice is O_y,

with ZD_{ij} being the definition area of the variable ν_i in the Rule R_j, where $ZD_{ij} \subseteq DDV_i$. We define the conflict between two rules as the intersection existing between both rules:

$R_i \cap R_j = ZD_{1i} \cap ZD_{1j}$ and $ZD_{2i} \cap ZD_{2j}$ and ... and $ZD_{ni} \cap ZD_{nj}$.

$R_i \cap R_j \neq \phi$ iff $\forall v_x \in V$ verify that $ZD_{xi} \cap ZD_{xj} \neq \phi$.

The non-empty intersection between two rules R_i and R_j, both with different consequences ($O_x \neq O_y$), generates one conflictive classification zone (CCZ).

Definition 2.2. One conflictive classification zone between two fuzzy rules R_i and R_j, noted as $CCZ(R_i, R_j)$, is calculated by means of the following equation:

$$CCZ(R_i, R_j) = \bigcup_{x=1}^{n} ZD_{xi} \cap ZD_{xj} \qquad (1)$$

Example 2.3. Let be $V = \{v_1, v_2\}$ and $DDV = \{DDV_1, DDV_2\}$ with $DDV_1 = \{A, B, C, D\}$, $DDV_2 = \{E, F, G\}$ and $\delta = \{O_x, O_y\}$, if we have the following rules:

- R_i : IF v_1 is $\{A, B, C\}$ and v_2 is $\{E, F, G\}$ THEN choice $= O_x$.
- R_j : IF v_1 is $\{B\}$ and v_2 is $\{E, G\}$ THEN choice $= O_y$.

$CCZ(R_i, R_j) = \{ZD_{1i} \cap ZD_{1j}, ZD_{2i} \cap ZD_{2j}\} = \{\{A, B, C\} \cap \{B\}, \{E, F, G\} \cap \{E, G\}\} = \{\{B\}, \{E, G\}\}$.

Once the conflictive classification zone between two rules has been calculated, we need to determine its width. One wide zone means that there are more possibilities of conflicts among rules in the classification process. To measure the extension of a CCZ, we use a measure which calculates the area of a conflictive zone.

Definition 2.3. The area of a conflictive classification zone between two rules R_i and R_j is calculated as follows:

$$area(CCZ(R_i, R_j)) = \frac{\sum_{x=1}^{n} \frac{\sum_{\forall L_k \in ZD_{xi} \cap ZD_{xj}} area(L_k)}{\sum_{\forall L_k \in DDV_x} area(L_k)}}{n} \qquad (2)$$

where area(L_k) is the area of the trapezoid which defines the label L_k and n is the number of variables. Besides, it is verified that area($CCD(R_i, R_j)$) $\in [0, 1]$.

To study the quality of the learned rules by the inductive learning algorithm ($\Re$), we can observe the number of conflicts between rules and how wide they are (area).

Algorithm 1 total area of conflicts

```
tot = 0;
total_area = 0;
for every $R_i$, $R_j \in \Re$ do
  if $R_i \cap R_j \neq \phi$ then
    tot = tot + 1
    total_area = total_area + area ($CCZ(R_i, R_j)$)
  end if
end for
```

3 Algorithm modification based on distance between fuzzy sets

The original algorithm tries to amplify every initial rule by means of every label that belongs to a DDV. We propose a new modification of the algorithm based on the distance between fuzzy sets: only an amplification is possible whether the distance between the last label added to the rule and the new label studied does not exceed the allowed distance. Note that the value of the allowed distance value is empirically assigned.

The modification of the algorithm based on the distance between fuzzy sets decreases the number of second and third type conflicts among rules. That is, the results are improved when most of errors come from a wrong choice of a class, or when there are several maximum values in the degree of convenience of the rules. However, first type errors (see Section 2.1) may increase because the rules are less general.

Definition 3.1. We define the distance between two fuzzy sets L_i and L_j as the existing area between them. Let L_{i+1} $(x_{i+1,1}, x_{i+1,2}, x_{i+1,3}, x_{i+1,4})$,..., L_{i+n} $(x_{i+n,1}, x_{i+n,2}, x_{i+n,3}, x_{i+n,4})$, be the fuzzy sets located between L_i and L_j. The distance between L_i and L_j is calculated as follow:

$$distance(L_i, L_j) = \frac{(x_{i+n,4} - x_{i+1,1}) + (x_{i+n,3} - x_{i+1,2})}{2} * h \qquad (3)$$

Being $i + n = j - 1$, and h the height of the trapezoids.

Next, the steps of the modified algorithm are described in detail:

Step 1: While there are examples in the training set do: To convert the example into an initial rule and include it into the set of initial rules (fuzzification process).

Step 2: To take an initial rule from the set of initial rules.

Step 3: To prove if it subsumes in some rule of the set of definitive rules. If it subsumes in some definitive rule then it is ignored, go to step 2.

Definition 3.2. The rule R_i: $((ZD_{1i}$, ZD_{2i}, ..., $ZD_{ni})$, $O_x)$ subsumes in the rule $R_k : ((ZD_{1k}, ZD_{2k}, ..., ZD_{nk}), O_y)$, if $ZD_{1i} \subseteq ZD_{1k}, ZD_{2i} \subseteq ZD_{2k}, ..., ZD_{ni} \subseteq ZD_{nk}$ and $O_x = O_y$.

Example 3.1. The initial rule (({A},{B}), O_x) subsumes in the final rule: if v_1 is $\{A, B, C\}$ and v_2 is $\{B, C\}$ then choice $= O_x$.

Step 4: For each variable in the initial rule:

Step 4.1: For each not considered label.

Step 4.1.1: To prove if it is possible to amplify the rule. If it is not possible, go to Step 4.1.

Step 4.1.2: To amplify the rule.

From every initial rule R_i: $((ZD_{1i}, ZD_{2i}, ..., ZD_{ni}), O_x)$ the amplification is made in each variable v_i, that is, each ZD_{ji}, $j = 1...n$ can be amplified with $\forall L_k \in DDV_i \;/\; L_k \notin ZD_{ji}$.

Definition 3.3. If we consider the rule R_i: $((ZD_{1i}, ZD_{2i}, ..., ZD_{ni}), O_x)$, the amplification from R_i to R_i': $((ZD_{1i'}, ZD_{2i'}, ..., ZD_{ni'}), O_x)$ is possible if the next two conditions are satisfied:

- There is no rule R_j: $((ZD_{1j}, ZD_{2j}, ..., ZD_{nj}), O_y)$ in the set of initial rules that verifies that $ZD_{1j} \subseteq ZD_{1i'}$, $ZD_{2j} \subseteq ZD_{2i'}$, ..., $ZD_{nj} \subseteq ZD_{ni'}$ and $y_p \neq y_q$.
- R_i can be amplified with a new label L_j, if the distance between the last label L_i added to the rule and L_j is lower than one value v_{id} defined by the expert (the expert must specify one distance value (v_{id}) for every variable). That is, distance$(L_i, L_j) \leq \; v_{id}$.

Step 5: To include the amplified rule into the set of definitive rules.

Step 6: If there are no considered rules in initial set of rules, go to step 2. Otherwise, END.

4 Experimental Results

We have tested the algorithm with the *Wine Recognition Database* [4] located in *The UCI Repository of Machine Learning Databases* [5], and created and donated by Stefan Aeberhard. This database has been used by another authors for generating rules [6, 7]. The wine data consists of 178 samples (with 59 instances in class 1, 71 in class 2, and finally, 48 in class 3) with 13 continuous attributes from three classes. These attributes are the following: Alcohol, Malic acid, Ash, Alcalinity of ash, Magnesium, Total phenols, Flavanoids, Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines, and Proline. Table 1 resumes such attributes and their ranges.

Table 1: Range of the attributes.

Attribute	**Range**	
Alcohol	Min: 11.03	Max: 14.83
Malic acid	Min: 0.74	Max: 5.8
Ash	Min: 1.36	Max: 3.23
Alcalinity of ash	Min: 10.6	Max: 30
Magnesium	Min: 70	Max: 162
Total phenols	Min: 0.8	Max: 3.88
Flavanoids	Min: 0.34	Max: 5.08
Non. phenols	Min: 0.13	Max: 0.78
Proanthocyanins	Min: 0.29	Max: 3.58
Color intensity	Min: 1.14	Max: 13
Hue	Min: 0.48	Max: 8.21
OD280/OD315	Min: 0.65	Max: 4
Proline	Min: 278	Max: 1680

It is important to remark that the learning algorithm fuzzifies the input numeric examples. Besides, it groups the examples fuzzified in the same way in an only example. The original algorithm works fine with databases where there is a high number of examples fuzzified in the same way. On the contrary, the number of conflicts and errors in the classification process grows tremendously. In this case, the modified algorithm improves the result in a large extent. This is the main reason why this database has been selected to prove the new version of the algorithm.

Tables 2 shows the results obtained by the original algorithm, whereas Tables 4 and 5 show the results obtained with the new algorithm by using different distance values. These tables consist of five columns: number of test, percentage correct classification, number of conflicts among rules, total area of conflicts, and the number of rules learned by the algorithm.

When testing this database, we have used 178 instances, 80% for learning and 20% for proving the rules that have been learned. The distances used for every attribute are shown in Table 3. For every attribute, one uniform distribution with five labels has been made. The distances correspond with the area of one and two trapezoids, respectively. Note that the distribution of fuzzy sets does not need to be uniform, and the distance values correspond with the area of a fuzzy set. In fact, the domain expert should decide the number of sets and the best distance per attribute.

Table 2: Results of the original algorithm

Test	%	Conflicts	Area	Rules
1	83.33	104	89.39	24
2	69.44	80	69.05	23
3	83.33	76	64.37	22
4	77.77	82	70.08	24
5	83.33	65	55.07	22
6	83.33	36	30.48	19
7	94.44	97	82.33	27
8	77.77	137	117.11	29
9	88.88	126	107.52	32
10	75	89	76.51	22

Table 3: Distances used with five labels per attribute.

Attribute	1 trapez.	2 trapez.
Alcohol	0.63	1.26
Malic acid	0.84	1.68
Ash	0.15	0.3
Alcalinity of ash	3.23	6.46
Magnesium	15.33	30.66
Total phenols	0.51	1.02
Flavanoids	0.79	1.58
Non. phenols	0.11	0.22
Proanthocyanins	0.55	1.1
Color intensity	1.98	3.96
Hue	1.29	2.58
OD280/OD315	0.56	1.12
Proline	233.67	467.34

As shown in the results of the test, the new version of the algorithm normally generates a higher number of rules. This is because the modified version of the algorithm generates rules less general than the rules generated by the original algorithm. That is, in the original algorithm, there is a higher probability of subsuming an initial rule in some rule of the set of definite rules. In addition, we can observe that the new version of the algorithm reduces the total area of conflicts among rules, and therefore, it improves the percentage correct classification.

5 Conclusions

In this paper, we have presented a new version of the algorithm for learning maximal structure rules from a training set. The original

Table 4: Results of the modified algorithm with five labels per attribute, and 1 trapezoid as distance

Test	%	Conflicts	Area	Rules
1	94.44	107	52.86	59
2	86.11	117	56.63	59
3	88.88	99	48.02	60
4	86.11	88	42.99	57
5	88.88	103	49.75	57
6	80.55	80	38.16	54
7	80.55	98	48.21	61
8	94.44	86	42.21	55
9	97.22	113	54.88	59
10	88.88	106	51.35	60

Table 5: Results of the modified algorithm with five labels per attribute, and 2 trapezoid as distance

Test	%	Conflicts	Area	Rules
1	83.33	56	36.19	27
2	86.11	74	48.19	33
3	83.33	70	45.79	31
4	83.33	73	47.15	32
5	88.88	88	56.77	32
6	80.55	56	36.49	29
7	80.55	98	48.21	32
8	91.66	71	40.06	32
9	91.66	89	57.91	33
10	77.77	60	38.54	30

algorithm works fine when the set used for learning consists of a number of instances that are often fuzzified in the same way. In this case, there is no a high number of conflicts among rules, and the learned rules represent to one wide set of instances. However, when the number of repetitions among the fuzzified instances is minimum, the number of conflicts among rules is incremented and the percentage correct classification is reduced.

The new version of the algorithm takes into account the distance between rules to amplify an initial rule. This version generates more rules, but it reduces the total area of conflicts and improves the percentage correct classification. If the distance used for one attribute is calculated as the sum of all trapezoids of this attribute, the new version works as the original algorithm. Therefore, in the worst case, the new version of the algorithm obtains the same results as the original.

Finally, we want to remark that the distance value for every attribute is actually chosen by one domain expert. For this reason, one of our lines of research consists of designing a new algorithm to decide the best distance value to optimize the results.

Acknowledgements

This work has been founded by the Regional Goverment of Castilla-La Mancha under Research Project e-PACTOS (ref. PAC-06-141), Sarasvati (ref. PBC06-0064) and the Spanish Ministry of Education and Science under the Research Project TIN2007-62568.

References

[1] Castro, JL. Castro-Schez, JJ and Zurita, J. Learning maximal structure rules in fuzzy logic for knowledge acquisition in expert systems. In *Fuzzy Sets and Systems, Elsevier*, volume 101, number 3, pages 331-342, 1999.

[2] Carmona, P. Castro, JL and Zurita, JM. Learning maximal structure fuzzy rules with exceptions. In *Fuzzy Sets and Systems, Elsevier*, volume 146, number 1, pages 63-77, 2004.

[3] Carmona, P. and Castro, JL. Using Ant Colony Optimization for Learning Maximal Structure Fuzzy Rules. In *The 14th IEEE International Conference on Fuzzy Systems*, pages 702-707, 2005.

[4] D. Goldberg. Wine recognition. Available via anonymous ftp from *ics.uci.edu* in directory */pub/machine-learning-databases/wine*, 1992.

[5] P.M. Murphy and D. W. Aha. *UCI Repository of Machine Learning Databases* [machine-readable data repository]. Maintained at Department of Information and Computer Science, The University of California at Irvine, 1992.

[6] Ishibuchi, H. and Murata, T. and Gen, M. Performance evaluation of fuzzy rule-based classification systems obtained by multi-objective genetic algorithms. In *Computers & Industrial Engineering, Elsevier*, volume 35, number 3-4, pages 575-578, 1998.

[7] Muhlbaier, M. and Topalis, A. and Polikar, R. Learn++. MT: A New Approach to Incremental Learning. In *Multiple Classifier Systems: 5th International Workshop, MCS 2004, Cagliari, Italy, June 9-11, 2004: Proceedings.*

[8] Herrera, F. and Lozano, M. and Verdegay, J.L. A learning process for fuzzy control rules using genetic algorithms. In *Fuzzy Sets and Systems, Elsevier*, volume 100, number 1-3, pages 143-158, 1998.

[9] Song, Q. and Kasabov, NK. NFI: a neuro-fuzzy inference method for transductive reasoning. In *Fuzzy Systems, IEEE Transactions on*, volume 13, number 6, pages 799-808, 2005.

[10] Serrurier, M. and Sudkamp, T. and Dubois, D. and Prade, H.Fuzzy Inductive Logic Programming: Learning Fuzzy Rules with their Implication. In *Fuzzy Systems, 2005. FUZZ'05. The 14th IEEE International Conference on*, pages 613-618, 2005.