Controlled Redundancy in Incremental Rule Learning

Luis Torgo
LIACC
R.Campo Alegre, 823 - 2o.
4100 PORTO
PORTUGAL
Telf. : (+351) 2 600 16 72 - Ext. 115
e-mail : ltorgo@ciup1.ncc.up.pt

Abstract. This paper introduces a new concept learning system. Its main features are presented and discussed. The controlled use of redundancy is one of the main characteristics of the program. Redundancy, in this system, is used to deal with several types of uncertainty existing in real domains. The problem of the use of redundancy is addressed, namely its influence on accuracy and comprehensibility. Extensive experiments were carried out on three real world domains. These experiments showed clearly the advantages of the use of redundancy.

1 Introduction

This paper presents the learning system YAILS capable of obtaining high accuracy in noisy domains. One of the novel features of the program is its controlled use of redundancy. Several authors ([5, 7, 2]) reported experiments that clearly show an increase in accuracy when multiple sources of knowledge are used. On the other hand, the existence of redundancy decreases the comprehensibility of learned theories. The controlled use of redundancy enables YAILS to better solve problems of uncertainty common in real world domains. Another important feature of the system is its mechanism of weighted flexible matching. This feature also contributes for the better handling of noisy domains. In terms of learning procedures the system uses a bi-directional search procedure opposed to the traditional bottom-up or top-down search common in other systems.

The next section gives a description of some of the main issues on the YAILS learning algorithm. The following section describes the classification strategies used by YAILS. Finally, section 4 describes several experiments carried out with YAILS that show the effect of redundancy on both accuracy and comprehensibility.

2 YAILS Learning Strategies

YAILS belongs to the attribute-based family of learning programs. It is an incremental rule learning system capable of dealing with numerical attributes and unknown information (unknown attribute values or missing attribute information).

YAILS search procedure is bi-directional including both specialisation and generalisation operators.

This section gives some details on the search mechanisms of YAILS as well as on the treatment of uncertainty.

2.1 Basic Search Procedures

YAILS algorithm involves two major steps. Given a new example to learn, the first step consists of modifying the current theory (possibly empty) in order to adapt it to the example. If it does not succeed it starts the second step which tries to invent a new rule that covers the example.

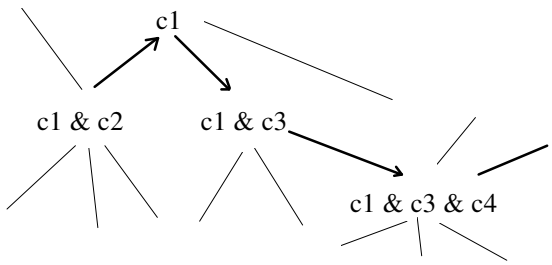

Fig. 1. A possible search path

Learning of this type of systems can be seen as a search over a space of all possible conjunctions within the language of the problem. In YAILS the search is guided by an evaluation function and employs two types of search transformations: specialisation and generalisation (fig.1). YAILS has two specialisation (and generalisation) operators. The first is adding (removing) one condition to a rule. The second is the restriction (enlargement) of a numerical interval within a rule.

YAILS uses exactly the same search mechanism when inventing a new rule while its goal is also to cover a particular example. For this purpose the specialisation operators are restricted in order to satisfy this goal. The restriction consists of adding only conditions present in the example being covered.

2.2 The Evaluation Function

The goal of the evaluation function is to assess the quality (Q) of some tentative rule. YAILS uses an evaluation function which relates two properties of a conjunction of conditions : consistency and completeness [8]. The value of quality is obtained by the following weighted sum of these two properties :

$$\text{Quality}(R) = [0.5 + \text{Wcons}(R)] \times \text{Cons}(R) + [0.5 - \text{Wcons}(R)] \times \text{Compl}(R) \qquad \textbf{(1)}$$

where

$$\text{Cons}(R) = \frac{\#\{\text{correctly covered exs.}\}}{\#\{\text{covered exs.}\}} \qquad \text{Compl}(R) = \frac{\#\{\text{correctly covered exs.}\}}{\#\{\text{exs. of same class as R}\}}$$

$$W_{cons}(R) = \frac{\text{Cons}(R)}{4}$$

This formula is a heuristic one, resulting from experiments and observations made with YAILS in real world domains. The formula weighs two properties according to the value of consistency (which is judged to be more important). Making the weights dependent on consistency is a way of introducing some flexibility into to the formula thus coping with different situations (such as rules covering rare cases or very general rules). Many other possible combinations of these and other properties are possible (see for instance [1,11]), and YAILS is in itself very easily changed in order to use another quality formula.

2.3 Unknown Information

The problem of unknown information is twofold. It raises problems during the learning phase and also in classification. The latter point is discussed in section 3.

YAILS system deals with two types of unknown information. The first arises when the value of some attribute is "unknown" and the second when the value is irrelevant. While the first case is interpreted as a kind of noise (thus presenting a problem) the second one is treated as a "don't care" situation (the human expert which has provided the examples to the system, may state that the attribute is irrelevant).

Before modifying the current theory to incorporate a new example, YAILS verifies whether the example is already covered. Both kinds of unknowns referred above may present some difficulties. These arise if one (or more) conditions of a rule tests an attribute for which the example has an "unknown" value. YAILS adopts a probabilistic strategy when dealing with this situation. A conditional probability is calculated as follows :.

$$P(A_i = V_i \mid A_j = V_j \wedge ... \wedge A_k = V_k) \qquad \textbf{(2)}$$

where $A_j = V_j \wedge ... \wedge A_k = V_k$ are the conditions satisfied by the example
and $A_i = V_i$ is the condition of the rule for which the example has an "unknown"
value.

Example :

Rule $\equiv$ colour = red $\wedge$ temperature > 37 $\wedge$ hair = dark
Example $\equiv$ colour = ? $\wedge$ temperature = 43 $\wedge$ hair = dark _ ....
In this example the calculated probability would be
P(colour = red | temperature > 37 $\wedge$ hair = dark)

This probability estimate is used to decide whether the example satisfies the rule. The decision requires a threshold that is user-definable.

The case when there is no information about some attribute is in fact stating that any rule with a condition with that attribute is not satisfied by the example.

3 YAILS Classification Strategies

YAILS uses mechanisms like the controlled use of redundancy and weighted flexible matching in order to achieve high accuracy but still keeping simple the used theory. The system is able to use different set ups of these mechanisms which contributes to the good flexibility of the program. The following sections explain these two strategies in more detail.

3.1 Redundancy

Most existing algorithms like for instance, AQ [10] and CN2 [4], use a covering strategy during learning. This means that the algorithm attempts to cover all known examples and that whenever some example has been covered it is removed. These systems would consider a rule useless if it covered examples that are already covered by other rules. AQ16 [15] uses a set of weights to remove this type of rules (considered *redundant* rules). This method is able to produce simpler theories than if the redundant rules were left in.

YAILS does not follow this method. Whenever the current theory does not cover a new example, new rules are created. The goal of this procedure is to find a "good" rule that covers the example. However the introduced rule may cover examples already covered by other rules. The only criterion used to consider a rule is its quality (see 2.2). Thank to this strategy, YAILS usually generates more rules than other systems.

The utility of such redundant rules can be questioned. The problem becomes even more relevant if we are concerned with comprehensibility. Nevertheless, there are several advantages on using these rules. YAILS is an incremental learning system and so what may seem a redundant rule may become useful in future. This implies that by not discarding some redundant rules the system can save learning time. In addition hand we can look at redundancy as a way of dealing with certain types of uncertainty that arise during classification. Suppose that we have a rule that cannot be used to classify an example because it tests attributes whose values are unknown in the example. If redundant rules are admitted, it is possible that one such rule can be found to classify the example. The advantages of redundancy are in efficiency and accuracy. The disadvantage is the number of rules (comprehensibility) of the theory.

YAILS uses a simple mechanism to control redundancy. Our goal is obtain the advantages of redundancy but at the same time minimise the number of rules used for classification. This mechanism consists on splitting the learned rules in two sets :- the *foreground* rules and the *background* rules. This split is guided by a user-definable parameter (minimal utility) which acts as a way of controlling redundancy. The utility of one rule is calculated as the ratio of the number of examples uniquely covered by the rule, divided by the total number of examples covered by the rule (this measure is basically the same as the u-weights used in [15]). Given a value of minimal utility YAILS performs the following iterative process :

Let the initial set of Learned Rules be the Candidate Foreground (CF)
REPEAT

```
    Calculate the utility of each rule in the CF
    IF the lowest utility rule in CF has utility less than the minimal utility THEN
        Remove it from CF and put it on the Background Set of Rules
UNTIL no rule was put on the Background
Foreground Set is the final CF
```

The higher the minimal utility threshold the less redundant is the theory in the foreground. The redundancy present in the foreground set of rules is called here *static redundancy*. YAILS uses only the foreground set of rules (FS) during classification. Only when it is not able to classify one example, it tries to find one rule in the background set (BS). If such rule is found the system transfers it from the BS to the FS so that in the end FS contains the rules used during the classification of the examples. This latter type of redundancy is called *dynamic redundancy*. The advantage of this strategy is to minimise the introduction of redundant rules.

YAILS can use different types of classification strategies. The "normal" strategy includes both static and dynamic redundancy. Other possibility is to use only static redundancy disabling thus the use of the BS. Finally it is also possible to use all the rules learned disregarding the splitting referred above. This latter strategy corresponds to the maximum level of redundancy. Notice that for the two first strategies is always possible to state the level of static redundancy through the minimal utility parameter. Section 4 presents the results obtained with several datasets using different classification strategies showing the effect of redundancy on both accuracy and comprehensibility.

3.2 Weighted Flexible Matching

Systems like AQ16 [15] that strive to eliminate redundancy become more sensitive to uncertainty inherent in real world domains. A small number of rules means that few alternatives exist when classifying the examples. If some condition of those rules is not satisfied the rule can not be used and the system is unable to classify the example. To minimise this undesirable effect these systems use flexible matching. This mechanism consists basically of allowing rules to be used to classify examples even though some of their conditions are not satisfied. With this strategy the systems are capable of improving performance but keeping the theory simple. Nevertheless, flexible matching does not solve some types of problems. If we have very simple rules (one or two conditions) and an example with an unknown value, then flexible matching is not sufficiently reliable. Small rules are in fact quite frequent. When using for instance the "Lymphography" medical dataset the resulting theory can have on average 2 to 3 conditions per rule. Flexible matching may fail to help in these situations. That is the reason why YAILS uses both redundancy and flexible matching during classification.

To explain flexible matching in YAILS, we need to describe the notion of weights associated with all conditions in each rule. These are generated by YAILS in the learning phase. The aim of these values is to express the relative importance of a particular condition with respect to the conclusion of the rule. YAILS uses the

decrease of entropy originated by the addition of the condition as the measure of this weight:

$$\text{Weight}(c) = H(R\text{-}c) - H(R) \qquad \textbf{(3)}$$

where

c is a condition belonging to the conditional part of rule R,

R-c is the conjunction resulting from eliminating the condition c from the conditional part of R, and H(x) is the entropy of event x.

These values play an important role in flexible matching. Given an example to classify, YAILS calculates the value of its Matching Score (MS) for each rule. This value is 1 if the example completely satisfies all the conditions of the rule, and a value between 0 and 1 otherwise. In effect it is a ratio of the conditions matched by the example. These conditions are weigh using (3). On the other hand if the example has some unknown value, equation (2) is used as an approximation. The general formula to calculate MS values is the following :

$$\text{MS}(Ex,R) = \frac{\sum_{c_i \in R} [\, m_i \times \text{Weight}(c_i) \,]}{\sum_{c_i \in R} \text{Weight}(c_i)} \qquad \textbf{(4)}$$

where

$$m_i = \begin{cases} 0 & \text{if the example doesn't satisfy condition } c_i \\ 1 & \text{if the example satisfies condition } c_i \\ \text{Probability as in (2)} & \text{if the example has an unknown value on } c_i\text{'s attribute} \end{cases}$$

Just to better illustrate the idea observe the following example (between brackets the condition weights) :

Ex $\equiv a=a_3 \wedge b=37 \wedge c=? \wedge e=e_6 \wedge ....$

Rule $\equiv a=a_3(0.343) \wedge c=c_4(0.105) \wedge e=e_6(0.65) \wedge f>32(0.04) \Rightarrow X$

Supposing that $P(c=c_4 | a=a_3 \wedge e=e_6) = 0.654$

$$\text{MS}(Ex,Rule) = \frac{1 \times 0.343 + 0.654 \times 0.105 + 1 \times 0.65 + 0 \times 0.04}{0.343 + 0.105 + 0.65 + 0.04} \approx 0.9327$$

That is the matching score of the example relative to the rule is 93.27%.

Having calculated this value for all rules YAILS disregards those whose MS is less than some threshold. The remaining set of rules are the candidates for the classification of the example. For those rules the system calculates the Opinion Value (OV) of each rule which is the product of the MS times the rule quality (Q) obtained during the learning phase. The classification of the example is the classification of the rule with highest OV. Note that if this latter set of rules is empty this means that there was no rule in FS able to classify the example. In that case the next step would be to apply the same procedure in the background set.

The mechanisms of redundancy and weighted flexible matching are interconnected in YAILS. The user can control this mechanisms through the minimal

utility parameter as well as the threshold referred to above. These two values enable YAILS to exhibit different behaviours. For instance, if you are interested in very simple theories then the minimal utility should be set near 1 and the flexible matching threshold to the lowest possible value but be careful not to deteriorate accuracy. On the other hand, if you are interested only in accuracy you could set the minimal utility to a value near 0 and raise the strictness of the flexible matching mechanism. Of course all these parameter settings are dependent on the type of domain. Section 4.1 shows some experiments with these parameters and their effect on accuracy and comprehensibility.

4 Experiments

Several experiments with YAILS were performed on real world domains. The three medical domains chosen were obtained from the Jozef Stefan Institute, Ljubljana. This choice enables comparisons with other systems as these datasets are very often used to test learning algorithms. On the other hand, the datasets offer different characteristics thus enabling the test to be more thorough. Table 1 shows the main characteristics of the datasets :

Table 1. Main characteristics of the datasets.

	Lymphography	*Breast Cancer*	*Primary Tumour*
Dimension	148 exs./18 attrs. 4 classes	288 exs./10attrs. 2 classes	339 exs./17attrs. 22 classes
Attributes	Symbolic	Symbolic + numeric	Symbolic
Noise	Low level	Noisy	Very noisy
Unknowns	No	Yes	Yes

The experiments carried out had the following structure: each time 70% of examples where randomly chosen for training and the remaining left for testing; all tests were repeated 10 times and averages calculated.

Table 2 presents a summary of the results obtained on the 3 datasets (standard deviations are between brackets).

Table 2. Results of the experiments.

	Lymphography	*Breast Cancer*	*Primary Tumour*
Accuracy	85% (5%)	80% (3%)	34% (6%)
No. of Used Rules	14 (2.7)	13.9 (5.6)	37.2 (2.8)
Aver. Conditions /Rule	1.86 (0.2)	1.94 (0.13)	1.96 (0.22)

The results are very good on two of the datasets and the theories are sufficiently simple (see table 3 for a comparison with other systems). This gives a clear indication of the advantages of redundancy. We should take into account that YAILS is an incremental system which means that all decisions are made sin a step-wise fashion and not with a general overview of all the data as in non-incremental systems. Because of this, a lower performance is generally accepted. This is not the case with YAILS (with exception to primary tumour) as we can see from the following table :

Table 3. Comparative results.

	Lymphography		*Breast Cancer*		*Primary Tumour*	
System	Accuracy	Complexity	Accuracy	Complexity	Accuracy	Complexity
YAILS	85%	14 cpxs.	80%	13.9 cpxs.	34%	37.2 cpxs.
Assistant	78%	21 leaves	77%	8 leaves	42%	27 leaves
AQ15	82%	4 cpxs.	68%	2 cpxs.	41%	42 cpxs.
CN2	82%	8 cpxs.	71%	4 cpxs.	37%	33 cpxs.

The results presented in table 3 do not establish any ranking of the systems as this requires that tests of significance are carried out. As no results concerning standard deviations are given in the papers of the other systems and the number of repetitions of the tests is also different, the table is merely informative. It should also be noted that AQ15 uses VL-1 descriptive language that includes internal disjunctions in each selector. This means that, for instance, the 4 complexes obtained with AQ15 are much more complex than 4 complexes in the language used by YAILS (which does not allow internal disjunction).

4.1 The Effect of Redundancy

The controlled use of redundancy is one of the novel features we have explored. Although redundancy affects positively accuracy it has a negative effect on comprehensibility. This section presents a set of experiments carried out in order to observe these effects of redundancy.

The experiments consisted on varying the level of "unknown" values in the examples given for classification. For instance a level of unknowns equal to 2 means that all the examples used in classification had 2 of their attributes with their values changed into "unknown". The choice of the 2 attributes was made at random for

each example. Having the examples changed in that way three classification strategies (with different levels of redundancy) were tried and the results observed. The results presented below are all averages of 10 runs.

The three different classification strategies tested are labelled in the graphs below as "Redund.+", "Redund." and "Redund.-", respectively. The first consists on using all the learned rules thus not making any splitting between the foreground and the background set of rules (c.f. section 3.1). The second is the normal classification strategy of YAILS, with some level of static redundancy and dynamic redundancy. The last corresponds to a minimal amount of static redundancy and no dynamic redundancy. The accuracy results are given in figure 2.

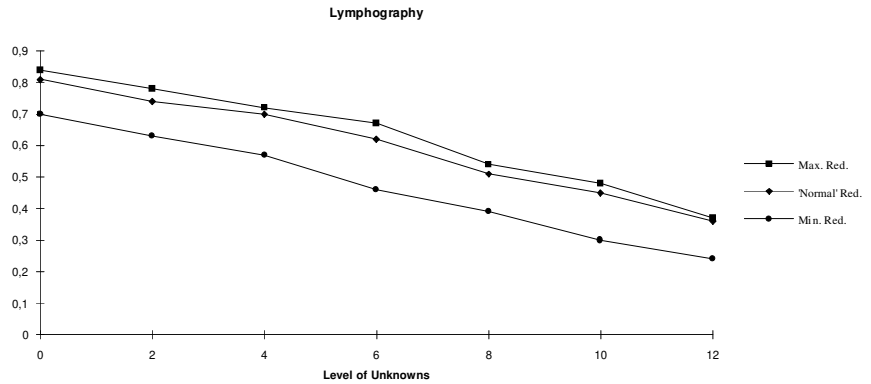

Fig. 2.a - Accuracy on the Lymphography dataset

Fig. 2.b - Accuracy on the Breast Cancer dataset.

Significant differences were observed. Redundancy certainly affects accuracy. With respect to the Breast Cancer dataset the advantage of redundancy is quite significant whenever the number of unknowns is increased. The advantages are not so marked on the Lymphography dataset.

Redundancy has a negative effect on simplicity. The decision of what tradeoff should be between accuracy and comprehensibility is made by the user. The cost can be high in terms of number of rules. For instance in the Lymphography experiment the "Redund.+" strategy used a theory consisting of 28, rules while the "Redund.-" used only 6 rules. The "Redund." strategy is in between, but as the level of unknowns grows, it approaches the level of "Redund.+" thanks to dynamic redundancy. The gap is not so wide in the Breast Cancer experiment but still, there is a significant difference.

In summary, these experiments show the advantage of redundancy in terms of accuracy. This gain is sometimes related to the amount of "unknown" values present in the examples, but not always. Redundancy can be a good method to fight the problem of "unknowns", but the success depends also on other characteristics of the dataset.

5 Relations to other work

YAILS differs from the AQ-family [10] programs in several aspects. AQ-type algorithms perform unidirectional search. In general they start with an empty complex and proceed by adding conditions. YAILS uses a bi-directional search. AQ-type programs use a covering search strategy. This means that they start with a set of uncovered examples and each time an example is covered by some rule the example is removed from the set. Their goal is to make this set empty. In YAILS this is not the case thus enabling the production of redundant rules.

The main differences stated between YAILS and AQ-type programs also hold in comparison to CN2 [4] with the addition that CN2 is non-incremental. In effect CN2 has a search strategy that is similar to AQ with the difference of using ID3-like information measures to find the attributes to use in specialisation.

STAGGER [14] system uses weights to characterise its concept descriptions. In STAGGER each condition has two weights attached to it. These weights are a kind of counters of correct and incorrect matching of the condition. In YAILS, the weights represent the decrease of entropy obtained by the addition of each condition. This means that the weights express the information content of the condition with respect to the conclusion of the rule. STAGGER also performs bi-directional search using three types of operators: specialisation, generalisation and inversion (negation). The main differences are that STAGGER learns only one concept (ex. rain / not rain) and uses only boolean attributes. YAILS differs from STAGGER in that it uses redundancy and flexible matching.

The work of Gams [6] on redundancy clearly showed the advantages of redundancy. In his work Gams used several knowledge bases which were used in parallel to obtain the classification of new instances. This type of redundancy demands a good conflict resolution strategy in order to take advantage of the

diversity of opinions. The same point could be raised in YAILS with respect to the combination of different rules. In [13] we present an experimental analysis of several different combination strategies.

The work by Brazdil and Torgo [2] is also related to this. It consisted of combining several knowledge bases obtained by different algorithms into one knowledge base. Significant increase in performance was observed showing the benefits of multiple sources of knowledge.

6 Conclusions

A new incremental concept learning system was presented with novel characteristics such as the controlled use of redundancy, weighted flexible matching and a bi-directional search strategy.

YAILS uses redundancy to achieve higher accuracy. The system uses a simple mechanism to control the introduction of new rules. The experiments carried out revealed that accuracy can be increased this manner with a small cost in terms of number of rules.

The use of a bi-directional search mechanism was an important characteristic in order to make YAILS incremental. The heuristic quality formula used to guide this search gave good results.

The rules learned by YAILS are characterised by a set of weights associated with their conditions. The role of these weights is to characterise the importance of each condition.

Several experiments were carried out in order to quantify the gains in accuracy obtained as a result of redundancy. Different setups of parameters were tried showing that redundancy usually pays off. Further experiments are needed to clearly identify the causes for the observed gains. We think that the level of "unknown" values affects the results. Redundancy can help to solve this problem.

Future work could exploit redundancy in other types of learning methods. It is also important to extend the experiments to other datasets and compare YAILS to other systems. It should be investigated what are the causes for the relatively poor results obtained on the Primary Tumour dataset. It seems that the systems is not producing as many redundant rules as on the other datasets. This can be deduced from the number of rules per class in the different experiments. In the Lymphography dataset there are about 3.5 rules per class and in Breast Cancer 6.9, but in the Primary Tumour YAILS generates only 1.6 rules per class. This apparent lack of redundancy could be the cause of the problem on this dataset.

Acknowledgements

I would like to thank Pavel Brazdil for his comments on early drafts of the paper.

REFERENCES

1. Bergadano, F., Matwin,S., Michalski,R., Zhang,J. : "Measuring Quality of Concept Descriptions", in *EWSL88 - European Working Session on Learning*, Pitman, 1988.
2. Brazdil, P., Torgo, L. : "Knowledge Acquisition via Knowledge Integration", in *Current Trends in Knowledge Acquisition*, IOS Press, 1990.
3. Cestnik,B., Kononenko,I., Bratko,I.,: "ASSISTANT 86: A Knowledge-Elicitation Tool for Sophisticated Users", in *Proc. of the 2th European Working Session on Learning* , Bratko, I. and Lavrac, N. (eds.), Sigma Press, Wilmslow.
4. Clark, P., Niblett, T. : "Induction in noisy domains", in *Proc. of the 2th European Working Session on Learning* , Bratko,I. and Lavrac,N. (eds.), Sigma Press, Wilmslow, 1987.
5. Gams, M. : "New Measurements that Highlight the Importance of Redundant Knowledge", in *Proc. of the 4th European Working Session on Learning* , Morik,K. (ed.), Montpellier, Pitman-Morgan Kaufmann, 1989.
6. Gams,M. : "The Principle of Multiple Knowledge", Josef Stefan Institute, 1991.
7. Gams,M., Bohanec,M., Cestnik,B. : "A Schema for Using Multiple Knowledge", Josef Stefan Institute, 1991.
8. Michalski, R.S. : "A Theory and Methodology of Inductive Learning", in *Machine Learning - an artificial approach*, Michalski et. al (Eds), Tioga Publishing, Palo Alto, 1983.
9. Michalski, R.S. , Mozetic, I. , Hong, J., Lavrac, N. : "The multi-purpose incremental learning system AQ15 and its testing application to three medical domains", in *Proceedings of AAAI-86*, 1986.
10. Michalski, R.S. , Larson, J.B. : "Selection of most representative training examples and incremental generation of VL1 hypothesis: the underlying methodology and description of programs ESEL and AQ11", Report 867, University of Illinois, 1978.
11. Nunez,M.. : "Decision Tree Induction using Domain Knowledge", in *Current Trends in Knowledge Acquisition*, IOS Press, 1990.
12. Quinlan, J.R. : "Discovering rules by induction from large collections of examples", in *Expert Systems in the Micro-electronic Age*, Michie,D. (ed.), Edinburgh University Press, 1979.
13. Torgo, L. : "Rule Combination in Inductive Learning", in this volume.
14. Schlimmer,J., Granger,R. : "Incremental Learning from Noisy Data", in *Machine Learning* (1), pp.317-354, Kluwer Academic Publishers, 1986.
15. Zhang, J., Michalski, R.S. : "Rule Optimization via SG-TRUNC method", in *Proc. of the 4th European Working Session on Learning*, Morik, K. (ed.), Montpellier, 1989.