KNOWLEDGE INTEGRATION
AND
FORGETTING

Luís Torgo

LIACC - Laboratory of AI and Computer Science
University of Porto
Rua Campo Alegre, 823 - 2º
4100 Porto, Portugal

Miroslav Kubat

Computer Center
Technical University of Brno
Udolni 19
60200 Brno, Czechoslovakia

Abstract

In this paper a methodology of knowledge integration is presented together with some experimental results. The goal of this method is the integration into a single theory of the knowledge obtained by different 'learning agents' . We state that this methodology can also be seen as a way to forget useless parts of theories. We briefly describe how this could be done in two different learning scenarios :- single-agent and multi-agent learning environments.

Keywords : Concept learning, knowledge integration, forgetting, single-agent learning, multi-agent learning.

1 . INTRODUCTION

Generally speaking, the typical task of a concept learning algorithm is as follows. Given a set of concepts and a set of examples which are said to represent these concepts, try to obtain a set of concept recognition rules (a theory). Each example given to the learning algorithm is previously classified as being an example of a specific concept. So the

task of the learning algorithm is to obtain a general concept description for each of the concepts. A concept description is a set of concept recognition rules which can be included in some kind of expert system shell. These rules can then be used to classify new examples into one of the learned concepts.

When the examples are not available at the same time an incremental strategy is needed. The main motivation for that is efficiency as those systems need only to make small changes to the previously learned theory when a new example becomes available. These small changes need to be validated against the past empirical experience. For that reason incremental learning algorithms adopt a full-memory approach. With this approch all examples are retained in memory so that validation is possible. In the following sections we present some problems of this approach which lead to the need of forgetting during learning.

The next section is a brief introduction to concept learning both incremental and non-incremental. Following we discuss the notion of forgetting. Section 4 presents the idea of knowledge integration (KI) together with some experimental results. Finally we show how KI can be related to the notion of forgetting in two learning scenarios.

2. GENERAL OVERVIEW OF CONCEPT LEARNING

Typical concept learning algorithms, such as AQ [Michalski&Larson,1978] or ID3 [Quinlan,1983] perform learning from training sets of examples, producing concept descriptions in the form of production rules (AQ) or decision trees (ID3). Production rules have the form of *if-then* rules, where the *if*-part contains description of an object while the *then*-part contains classification of the object into one concept. In a decision tree the nodes represent attributes and the leaves the concepts. Each branch is a value of the attribute in the parent node. Notice that each path in a decision tree, from the root node to a leaf, can be viewed as a production rule.

Many algorithms for learning from examples have been published. Several commercial systems are already on the market. Among the criteria for the evaluation of these systems the most commonly used are accuracy, simplicity and robustness against noise in the given examples.

Usually these characteristics are measured by means of some *classification* task. Given a set of examples we divide it in a *training set* which is used for learning purposes and a *testing set* which is used to evaluate the learned theory. *Accuracy* is then calculated as the percentage of testing examples that are correctly classified by the learned theory. The algorithm should be *robust* in the sense that it should cope with the various forms of noise [Brazdil&Clark,1988], such as wrong attribute values, incorrect pre-classification of the examples given to the algorithm, etc. *Simplicity* can be expressed in terms of the number of rules (or tree branches) and the average length of the rules (or the average number of nodes of a path from the root to a leaf in the tree).

The first learning programs that appeared were based on algorithms that processed the whole set of training examples at the same time, producing a theory. When a new example appeared, the program had to be re-run on the whole training set (plus the new example, of course).

Later incremental algorithms of learning from examples were created. Programs such ID4 [Schlimmer&Fisher,1986], the first incremental version of ID3 were made, as well as

several evolutions (ID5, ID5R, IDL, etc.). Also some incremental versions of AQ appeared (AQ15, AQ16, etc.).

Apart from eliminating the need for re-runing the algorithm in the whole set these systems also provide at any moment of time a theory which explains the known examples. This present theory can be used for classification at that particular stage of the learning process.

Further, incremental systems do not require so many examples to develop a plausible theory. They require only those examples that lead to improvements. In this respect, higher efficiency is achieved (see [Markovitch&Scott,1989], but his idea was pointed out as early as in [Mitchell,1977]). Nevertheless given a set of examples, if we use it to learn both in a non-incremental system and in an incremental one, it should be expected that the performance of the non-incremental system is higher. This is obvious, as the non-incremental system makes all its decisions with a view of all the examples, while the incremental one does not (only in the end it is in the same position). Notice that this observation is valid only if the non-incremental and incremental algorithms are similar (let's say that one is an incremental version of the other but without any major strategy differences).

Finally, it seems that incremental systems can cope with at least some of the problems posed by flexible concepts. These are concepts whose meaning varies with time [Michalski,1990].

3. FORGETTING

The notion of flexible concepts suggests that some kind of forgetting capability should be included in an incremental learning system which deals with this type of concepts in order to put away those aspects of flexible concepts that have become obsolete. In this respect, the mechanisms of forgetting within learning systems have been studied in [Kubat,1989]. Also it has been found that forgetting the irrelevant pieces of knowledge can improve the accuracy of knowledge bases modelling *static* concepts. This was pointed out in [Markovitch&Scott,1988] and some mechanisms for forgetting were suggested in [Markovitch&Scott,1989].

In our opinion, there are two explanations for the effectiveness of forgetting: (1) noise in training examples and (2) improper selection of training examples. The first point is normally solved using pruning techniques. These include pre-pruning and post-pruning (for example [Cestnik et al., 1987] apply them to trees), respectively if the pruning is done during learning or after it. These methods are based on statistical tests of significance of the hypotheses (rules). Those tests indicate portions of the learned theory that are untrustable and should not be considered.

In (2) improper selection of training examples can lead to the learning of useless rules. For an illustration of that, consider the set of examples from fig.1a. The examples are classified into two classes, C1 and C2. If we choose the examples marked by * as being the training examples, a typical learning algorithm (no matter whether being incremental or not) will produce rules similar to those in fig.1b.

If we apply these rules on the the testing set consisting of all five examples of fig. 1a, we find out that only three of these examples (1, 2 and 3) are correctly classified. Now, if we *forget* the rule 'u /\ s => C_1', we realize that the number of examples correctly

classified by the new set of (two) rules increased to four with only example 2 being classified incorrectly.

The set of examples:			The learned rules:
1	$u \wedge p \wedge b \Rightarrow C_2$	*	$u \wedge s \Rightarrow C_1$
2	$u \wedge s \wedge a \Rightarrow C_1$	*	$t \Rightarrow C_1$
3	$t \wedge p \wedge a \Rightarrow C_1$	*	$u \wedge p \Rightarrow C_2$
4	$u \wedge s \wedge c \Rightarrow C_2$		
5	$u \wedge s \wedge b \Rightarrow C_2$		
	(a)		(b)

Fig. 1 - Illustration of the meaning of forgetting

Now, an interesting problem arises: what part of the knowledge should be forgotten and under what circumstances? In the following sections, we present an approach based on the notion of knowledge integration, together with some experimental results. It is our belief that the performance gain achieved by this approach is obtained mainly by the fact that it solves the issue of reasonable forgetting of some useless rules, which makes it an alternative, or perhaps complement, to various pruning mechanisms.

4. KNOWLEDGE INTEGRATION

In this section we present a method of knowledge integration [Brazdil&Torgo,1990]. The main purpose of this method is as follows. Given a set of agents, each one involved in producing a theory, try to integrate the individually obtained theories into one integrated theory(IT) that *performs* better than any of the individual theories. Those agents can obtain their theories in no matter way, as long as they are expressed in the same agreed, integration language. Also the different theories should address the same problems, so that a performance gain can be obtained when *joining* the individual's expertise.

In the experiments described later, a system called INTEG.3 is used. In those experiments two different machine learning algorithms were used to create the individual agent's theories. Each theory is created using its own empirical evidence (examples). So the individual learning phase is done completely independently from the point of view of the agents. Then system INTEG.3 using the individual theories obtained from the agents builds an IT, which we verified that performed better than the initial individual theories.

During the process of integration the rules learned by all agents are evaluated and using this evaluation INTEG.3 decides which rules to include in the IT and which are to be forgotten. The process of evaluation is done using a set of examples (DI) which INTEG.3 uses to observe each agent's rule performance. The evaluation of the rules is done via quantitative and qualitative characterization. With this characterization which is described below system INTEG.3 obtains what we call the rules quality.

The Integration Algorithm

The integrated theory (IT) is constructed on the basis of a candidate set. Initially this set contains all the rules of the individual theories $T_1..T_n$. The objective is to select some rules from the candidate set and transfer them into IT so as to achieve good performance

(accuracy). The method relies on the qualitative and quantitative characterization of rules and includes the following steps :

(1) Order rules in the candidate set according to rule quality.
(2) Select the rule R with the best quality and include it in IT.
(3) Mark the cases covered by R.
(4) Recalculate the quality estimates of rules excluding the marked cases.
(5) Go back to (1).

The process of adding new rules to IT terminates when the accuracy of the 'best rule' in the candidate set falls below a certain threshold.

It can be seen that some kind of forgetting is performed via knowledge integration, because some learned rules are thrown away as a consequence of the evaluation process. Nevertheless, the performance gets better.

Rule Characterization

Qualitative characterization of a particular rule R consists of two lists. The first one mentions all the test cases (belonging to DI) that were correctly classified by the rule. The second list mentions all the examples incorrectly covered by the rule.

Quantitative characterization of some rule R is done using estimates of rule quality. Again these estimations are based on the tests made using DI. In INTEG.3 rule quality is calculated using the expression :

$$Q_R = Cons_R * e^{(Compl_{C,R} - 1)} \quad \textbf{(1)}$$

where $Cons_R$ represents an estimate of *consistency* of rule R and $Compl_{C,R}$ an estimate of *completeness*.

The notions of consistency and completeness are usual parameters of observing the performance of learning algorithms[Michalski,1983]. With consistency one tries to evaluate how well a rule classifies and with completeness we observe how well a rule covers the universe of examples of the concept to which the rule belongs.

When doing classification two type of errors can occur :- errors caused by *misclassification* sometimes referred to as errors of *comission* (Ec_R) and errors of *omission* (Eo_R) which arise whenever a rule fails to cover some case, that is when no classification is actually predicted.

The estimate of consistency of rule R is calculated using the formula :

$$Cons_R = \frac{C_R}{C_R + Ec_R} \quad \textbf{(2)}$$

where C_R represents the number of correctly classified cases, and Ec_R the number of misclassifications.

As we can see $Cons_R$ represents a ratio of correctly classified cases. The errors of omission (Eo_R) are not included in this expression. These play a role in $Compl_{C,R}$, the completeness of rule R with respect to concept C. This value is calculated as follows :

$$Compl_{C,R} = \frac{C_R}{C_R + Ec_R + Eo_R} \quad \textbf{(3)}$$

Notice that when estimating rule quality (1) we use the value of $Compl_{C,R}$ as a power of **e**. We wanted to differentiate the weight of rule consistency and rule completeness. By this method rule consistency is *affected* by rule completeness, in spite of being more important. In other terms, if we have two rules with equal consistency, the one which covers more cases (more complete) is preferred. With this solution good results were obtained (as it will be shown later). More details about this method and about comparisons with other methods of estimating rule quality can be found in [Brazdil&Torgo,1990b].

Experimental Results

In our experiments four different agents were involved. Two of them used an ID3-like algorithm (TreeL), and the other two an incremental rule learning algorithm (IncRuleL) [Torgo,1991]. Notice again that all agents use different examples. The purpose of our experiments was to compare the performance of the integrated theory with the performance of the individual theories obtained by each agent. The tests were performed on lymphography data obtained from JSI, Ljubljana. This data set contains 148 examples which are characterized by 18 attributes and there are 4 possible concepts to which each example can belong.

Each theory was generated by an inductive system (TreeL or IncRuleL) on the basis of a given number of examples which were selected from a given pool by a random process. The numbers of training examples used were 5, 10 ,15 ... 50. In order to exclude the possibility of fortuitous results the experiments related to N training examples were repeated 20 times and the mean value of these repetitions obtained.

Figure 2 presents two graphs showing the results obtained on those experiments. Figure 2a compares the performance of the IT with the mean performance of the four agents. On figure 2b we compare the number of rules (complexity) of the IT with the sum of the rules of all agents (giving an idea of how many rules were forgotten).

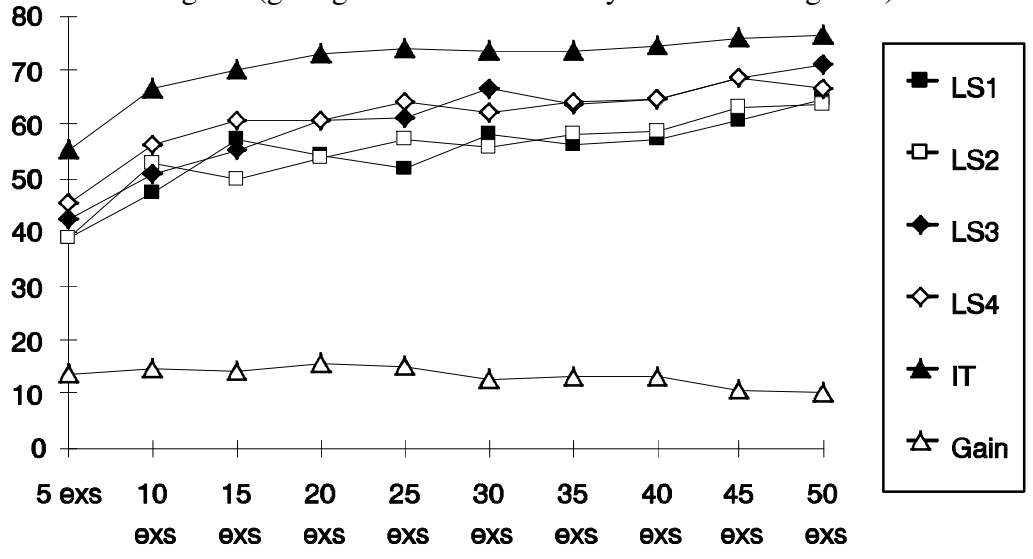

Fig. 2(a) - *Performance Comparison*

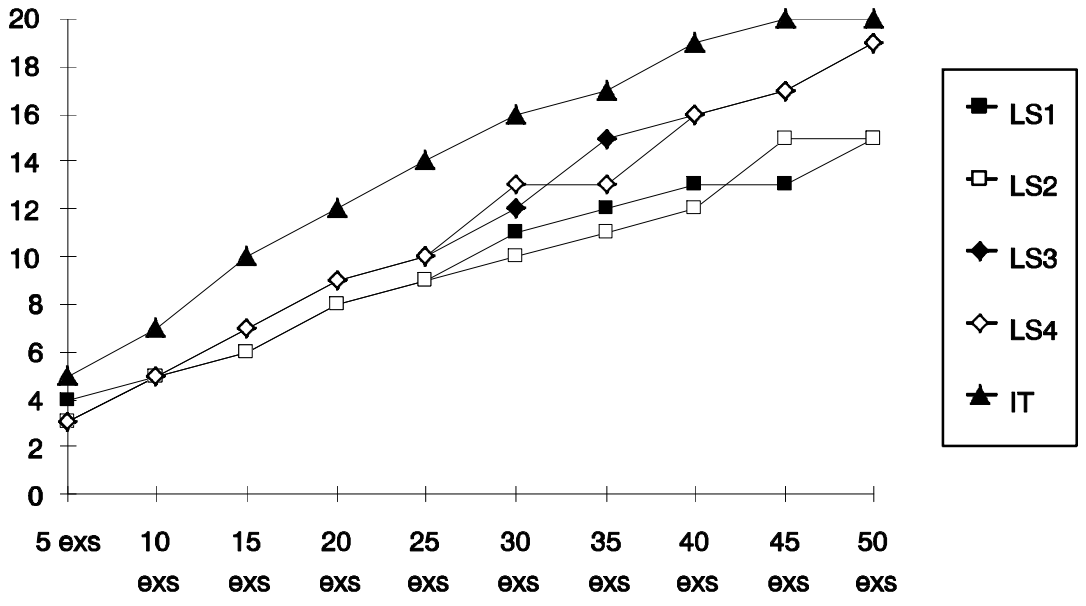

Fig. 2(b)- *Complexity Comparison.*

As it can be seen in spite of a big number of rules being forgotten (fig.2b) when compared with the sum of the number of rules of each agent, there is a raise in performance (fig.2a). So knowledge integration is a possible strategy for forgetting in a learning process.

5. KNOWLEDGE INTEGRATION AND FORGETTING

In this section we analyze the use of knowledge integration(KI) for the solution of the problem of forgetting in the course of learning. Namely KI can be used to forget some previously learned rules. In order to better describe our ideas about the application of the KI methodology we present two scenarios of machine learning processes :- a single-agent and a multi-agent environment.

Forgetting in a single-agent scenario

In the scenario of single-agent learning we propose adding KI to a learning algorithm in order to forget some useless rules that have been acquired before. This might seem contradictory as it was said that KI joined the knowledge of several agents into one single theory, and here we are talking about a single agent. Nevertheless, one can take advantage of the architecture and algorithm of KI and apply it to an existing learning algorithm. We illustrate this by the following figure :

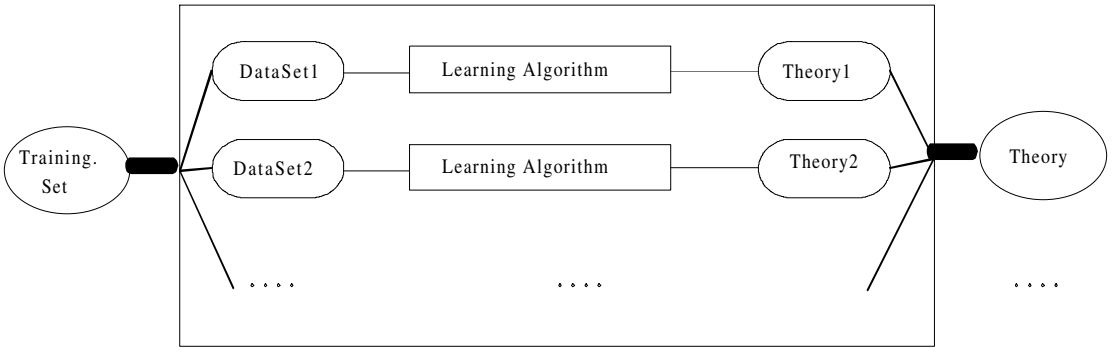

Fig. 3 - *Use of KI to build a new single-agent learning algorithm.*

This can be seen as a single-agent learning algorithm because we give it a set of examples and it produces a theory induced by this set. Inside of it, it is hidden another learning algorithm (which for this discussion is irrelevant) put together with the methodology of KI. This methodology provides a good way of dealing with the problem of forgetting. In this case, KI is used as a technique of pruning rules. This technique is different from others (for example [Cestnik et al., 1987]) as it prunes complete rules instead of parts of rules. For more details regarding the implementation of this strategy into a classical learning algorithm like ID3 see [Torgo,1991b].

Forgetting in a multi-agent environment

If we imagine a community of independent agents interacting with some reality we can use KI as a supervisor agent that incrementally monitors each agent's learning process, telling him what he should consider and what he should forget. The individual agents can ask the supervisor to arrange a meeting between them. This supervisor, using some kind of knowledge integration methodology can provide the exchange of information between the agents. During this exchange of information one can imagine several forms of communication such as :- adoption of other agent's rules; forgetting some personal rule; or the modification of a rule to accommodate some 'critics' of the other agents. Notice that this last aspect is not considered in the presented knowledge integration methodology although some solutions have been proposed (see [Sian,1991] for an approach to the solution of this problem). After this discussion phase, each agent can return to his individual learning task, but it is logical to expect that they adopt the IT as it was agreed that it performed better.

This adoption phase has some interesting aspects. First, it demands that each agent uses an incremental learning algorithm, as it needs to continue learning from the adopted IT. This seems logical in a scenario as the one presented above if we consider the limitations of non-incremental learning algorithms discussed in section 2. Finally the adoption phase has one major difficulty that arises from the functional aspects of incremental learning algorithms. Such algorithms require not only a theory, from which they can continue to learn, but also a set of examples that support this theory. Even if we don't adopt the full-memory approach, we still need some examples to allow us to

continue learning. If we don't have these examples the theory could be completely reformulated in the presence of a single new example, which is highly undesirable.

The difficulty of obtaining such a set of examples to support the IT arises from the fact that the rules contained in it possibly came from different sources(agents). The more logical solution to this problem is to ask to the agent responsible for the rule, the examples that support it. A problem of this solution is that if we put all the received examples together and present them to an incremental learning algorithm as a support for the IT, these examples force the algorithm to make some modifications to the theory. The problem is that examples used on learning of a specific agent's rule can induce modifications to other agent's rules. This leads us towards the problems of the above referred work of Sian.

Another possibility is that if we have a theory(IT) and we want a set of examples that support it, then we could use *deduction* to obtain such a set. In this case we could finish with a set of examples completely different from the examples used in learning the rules of IT, but this presents no problems. For this solution one has to decide how many examples to deduce so that the IT becomes robust to the arrival of new examples. The degree of robustness can be a function of the Q values obtained during the integration phase. Another important decision is which examples to deduce, because as we saw, we don't want any modifications to be induced by the obtained set of examples. Notice that if we adopted this solution, forgetting of examples would also be performed as we throw away all the examples used in the first individual learning phase and proceed with a single set of examples that should be the minimal set that guarantees that no modifications are induced to the IT.

Further research is needed to decide which of the presented alternatives is best in order to enable the agents to adopt the IT and proceed from it in their individual learning tasks.

6. CONCLUSIONS

A brief review of learning by examples methodologies was given. The main disadvantages of non-incremental algorithms were presented. Incremental learning and the problem of forgetting in learning was addressed.

We presented a methodology of knowledge integration that provides a means for the partial solution of the problem of forgetting in learning by examples. This method in spite of its forgetting capability also brings a improvement in performance as it was shown by extensive experimental results.

Knowledge integration also deals with problems such as multi-agent learning enabling a community of agents to interact, in order to improve the agent's view of the world. A possible architecture for such a multiple agent environment was presented and associated communication problems discussed.

Also a possible architecture for a learning by examples algorithm which took advantage of KI strategy was given. This possibility should be further developed as it might bring more insight to the relations and advantages of the use of KI methodology in learning.

Acknowledgments

The authors wish to express their gratitude to Pavel Brazdil for encouragement on this work as well for his work on the KI methodology.

REFERENCES

Brazdil, P., and Clark, P. : "Learning from Imperfect Data", in *Proceedings of International Workshop on Machine Learning, Meta Reasoning and Logics*, Sesimbra, Portugal, 1988.

Brazdil, P., and Torgo, L. : "Knowledge Acquisition via Knowledge Integration", in *Current Trends in Knowledge Acquisition* , B.Wielinga, et. al (eds), IOS Press, 1990.

Brazdil, P., and Torgo, L. (1990b) : "Knowledge Integration and Learning", working paper, LIACC, University of Porto, 1990.

Cestnik, B., Kononenko, I., Bratko, I. : "ASSISTANT 86: A Knowledge-Elicitation Tool for Sophisticated Users", in *Progress in Machine Learning*, I.Bratko and N.Lavrac (eds), Sigma Press, Wilmslow, 1987.

Kubat, M. : "Floating Approximation in Time-Varying Knowledge Bases", in *Pattern Recognition Letters (vol.10)*, 1989.

Markovitch, S., and Scott, P. D. : "The role of forgetting in Learning", in *Proceedings of the 5th International Workshop on Machine Learning*, Ann Arbor, U.S.A., 1988.

Markovitch, S., and Scott, P. D. : "Information Filters and their Implementation in the SYLLOG System", in *Proceeding of the 6th International Workshop on Machine Learning*, New York, 1989.

Michalski, R.S. : "A theory and methodology of inductive learning", in *Machine Learning - an Artifficial Intelligence Approach* , Michalski et. all (Eds), Tioga Publishing, Palo Alto, 1983.

Michalski, R.S. : "Learning Flexible Concepts : Fundamental Ideas and a Method Based on Two-Tiered Representation", *Machine Learning (vol III)*, R.Michalski, Y. Kodratoff (eds), Morgan Kaufmann Publ.Inc., 1990.

Michalski, R.S., and Larson, J.B. : "Selection of most representative training examples and incremental generation of VL1 hypotheses: The underlying methodology and description of programs ESEL and AQ11", Report 867, University of Illinois, 1978.

Mitchell, T. M. : "Version Spaces : A Candidate Elimination Approach to Rule Learning", in *Proceedings of the 5th International Joint Conference on AI*, Cambrige, Massachussets, 1977.

Quinlan, J.R. : "Learning efficient classification procedures and their application to chess end games", in *Machine Learning - an Artifficial Intelligence Approach* , Michalski et. all (Eds), Tioga Publishing, Palo Alto, 1983.

Quinlan, J.R. : "Induction of Decision Trees", *Machine Learning (3)* pag.81-106, Kluwer Academic Publishers, 1986.

Schlimmer, J. C., and Fisher, D. : " A case study of Incremental Concept Induction", in *Proceedings of the Fifth National Conference on Artificial Intelligence* , Morgan Kaufmann, 1986.

Sian, S. : "Extending Learning to Multiple Agents : Issues and a Model for Multi-Agent Machine Learning(MA-ML)", in *Proceedings of EWSL-91*, Porto, Portugal, 1991.

Torgo, L. : "Incremental Learning using IL1", working paper, LIACC, University of Porto, 1991.

Torgo, L. (1991b): "Knowledge Integration as a Learning Methodology", working paper, LIACC, University of Porto, 1991.