Integrating Models of Knowledge and Machine Learning

Jean-Gabriel GANASCIA[2] Jérôme THOMAS[1-2] Philippe LAUBLET[1]

[1]ONERA, DMI, GIA
29, av. de la division Leclerc
BP 72
92322 Chatillon Cedex
e-mail:{laublet thomas}@onera.fr

[2]LAFORIA-CNRS
Université Pierre et Marie Curie
Tour 46-0, 4 place Jussieu
75252 Paris Cedex
e-mail:{ganascia thomas}@laforia.ibp.fr

Abstract: We propose a framework allowing a real integration of Machine Learning and Knowledge acquisition. This paper shows how the input of a Machine Learning system can be mapped to the model of expertise as it is used in KADS methodology. The notion of learning bias will play a central role. We shall see that parts of it can be identified to what KADS's people call the inference and the task models. Doing this conceptual mapping, we give a semantics to most of the inputs of Machine Learning programs in terms of knowledge acquisition models. The ENIGME system which implements this work will be presented

1. Introduction

In the classical view, the goals of machine learning are to develop computational theories of learning and to build learning machines. From this last point, machine learning can be seen as a process of transforming source data (observations, traces of computation, teachers' remarks...) into organized and usable knowledge. Among all the inputs of ML programs, some are easily identified, for instance, the notion of sources of data. But, there are many other inputs which are not so easily identified. They correspond to the notion of learning bias whose role has been emphasized by many authors in the past [Mitchell 82, Utgoff 86, Ganascia 88, Russel 90,...]. Without going into details, it appears that, since induction can be seen as a search process, and since the space of possible generalizations is huge, it is necessary to introduce many heuristics and constraints to guide the search.

For us, the process of knowledge acquisition can be seen as an interactive process with a machine learning program where the source data and the different parts of the bias must be progressively refined. In this process, the role of the expert will be to teach the learner when it makes a mistake and to give it a way to modify its behaviour. Hence, it is really useful for the apprentice to be able to explain, in an understandable way, the false reasoning to allow the expert to isolate one or more false steps in it. A first way to modify the behaviour of the system is to give a counter-example of the false rule to the learner. But this way of proceeding can be a little tedious especially if we don't have a great number of examples or if we have given too weak a bias. A better way to modify the behaviour of the learner is to modify its exploration constraints. Thus, these constraints and their use by the learning tool must make sense for the expert.

To fulfil these objectives, the solution proposed is to constrain the apprentice during the induction phase with kinds of knowledge that are issued from the KADS knowledge acquisition methodology. Such constraints have several points of interest. First, the learning bias will be knowledge which comes from the knowledge

acquisition framework, thus, its acquisition may be done by the tools developed in this area. Second, the same models, with the same semantics, will be used by all the knowledge acquisition actors, i.e., the learning tool, the knowledge engineer, the expert and the inference engine. We think that this point is really important because it allows the expert to foretell the effects of a constraint on the rule system produced.
In the course of this paper, we will use an example related to the first bid of the card game of bridge. Thus, as a preliminary step, let us explain the main features of this problem. The goal is, knowing the hand, to choose an opening. In this example, in order to simplify, we will only choose a suit and no level of opening. It is obviously a classification problem, but the expert uses several intermediate concepts (between the observables and the opening) to reach the solution. An expert resolution will roughly follow the reasoning: to value the hand, if the valuation is sufficient then pass else, the next step is to abstract the kind-of-distribution from the distribution and the strong-suit to see if we have a normal hand or an extraordinary one. Indeed, there are two different ways to solve the problem, if it's an extraordinary hand, we will follow a convention otherwise we will abstract the kind of hand from the observables, then we will compute the solution.
The next section will present the notion of learning bias as it is classically understood in the Machine Learning community. Then we shall provide some information concerning the CHARADE learning system and the notion of model of expertise in the KADS methodology. Finally, we shall present the role for learning of the inference structure and the task structure in sections four and five.

2. Learning bias

According to [Russel 90] one distinguishes three kinds of bias: the *domain bias*, the *restriction type bias* and the *preference type bias*.

A) The domain bias contains the attributes, their syntax, hierarchies of descriptors, axioms expressing properties of attributes (as total order among the possible values, ...) and so on. It corresponds to a part of the *domain knowledge* in the KADS terminology.

B) So, the domain bias defines a structured hypothesis space which is the set of the candidate theories that the program could propose for any possible set of observation sentences. If the expert already has knowledge which allows to prune some parts of this space, it may be useful to provide the system with. According to [Russel 90], we will call such a knowledge the *restriction type bias*. This information is implicit in most of the learning systems. Making it explicit allows more flexibility in the use of an apprentice.

C) The third kind of bias is what [Russel 90] names the *preference type bias*. It provides the system with a preference ordering which allows the learner to choose between of the possible relations.

Our aim is to express all the learning bias as parts of the knowledge and to acquire it. Historically, designing the CHARADE system [Ganascia 87, 88], the aim was to provide some semantics for informations required as input to a learning system. Therefore, some of those inputs were related to the operationality of the induced knowledge, i.e., to the goal of the learned knowledge. We now pursue this work and for that we choose to provide some explicit links with the KADS methodology.

3. Background

The CHARADE system generates knowledge bases automatically from examples by detecting correlations in a training set of examples. It is based on the use of two distributive lattices, one for the learning set, i.e., the set of examples, and one for the description space. The properties of the lattices are used in the learning process to facilitate the detection of similarities and to limit the search cost. Concretely, a similarity corresponds to a correlation empirically observed in the learning set. If all the examples that have a conjunction of descriptors D1 in their description, also have the descriptor d2, it's possible to induce that D1 implies d2 in the learning set. The principle of induction used in symbolic learning consists in a generalization of these relations to the whole description space.

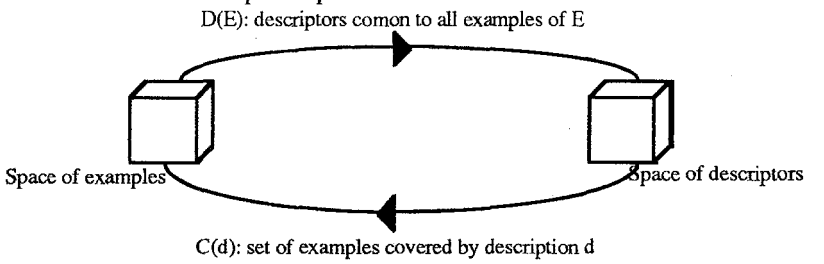

Fig. 1. The C and D functions between the two spaces

The main difficulty consists in the large number of potential regularities that are useless. It is important to note that the CHARADE system makes use of logical constraints to restrict the logical implications it generates, to useful ones.

For KADS [Wielinga 92], the model of expertise produced during the analysis phase allows the specification of the expertise that the future system has to use to perform the problem-solving tasks. It consists of four layers: the *domain layer*, the *inference layer*, the *task layer* and the *strategy layer*. In this paper, we are only interested in the first three.

* The domain layer describes the concepts, the relations and the different properties of the domain. For instance, in the bridge example, the domain layer will include all the concepts' descriptions (possible values, type,...) the relations between these concepts, semantic networks linking them,...
* The inference layer is the description of the possible inference steps that are considered as primitive ones at the abstraction level of expertise model. An inference step (or a fragment) consists of a *knowledge source* that describes the type of inference (e.g.: classify, abstract ...)., and of *metaclasses* as inputs and outputs of the knowledge source. A metaclass (or a role) describes the *role* of domain entities in the problem-solving process and it is linked to the domain concepts which fulfil this role. The inference structure network gives a visual representation of dependencies between the different inference steps (see figure 2 for the example).
* In the task layer, it is possible to define composite tasks recursively and the ordering of the tasks (simple or composite) is described in a textual form with selection and iteration operators. This task structure gives the way to go through the inference structure (see figure 3 for the example).

In the following sections of this paper, we suppose that it is possible to build this problem solving method by a process similar to those of [McDermott 88, Wielinga 92,...]. Then, we emphasize how this model can be useful not only to elicit new

domain knowledge as in the previously mentioned approaches, but also to learn this knowledge inductively from examples.

4. The Inference Structure

In ENIGME, the general idea is to learn only relations which follow the primitive inference steps of this structure. Indeed, as it has been asserted before, this structure gives the dependencies between the different concepts involved in a resolution process. Since the purpose of the produced rules is to accomplish such a resolution, the only useful relations are those which link concepts following these dependencies. For instance, once the inference structure (see figure 2) of the bridge example has been acquired, if ENIGME has to learn the relations of the select_conv knowledge source it will learn the relations between concepts involved in the roles: *valuation*, *syndrome1*, *solution classes* and *convention*.

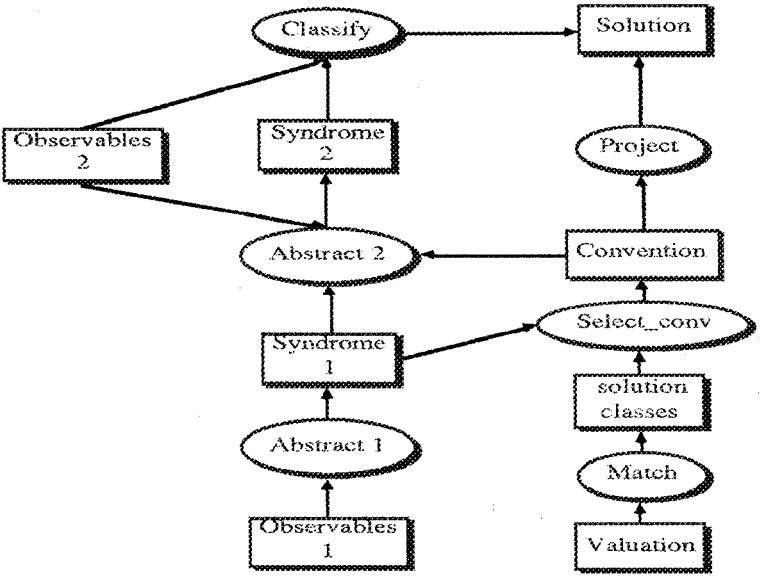

Fig. 2. The inference structure of the bridge example.

Thus, from the search point of view, the inference structure allows the learner to consider only the parts of the search space whose elements are conjunctions of concepts involved in the input roles of a fragment. Since the dimension of this space is exponential with the number of attributes, if all the concepts are not involved in the same fragment, the gain will be significant. In the example, the search space is reduced by a factor of roughly 10^9. Moreover, since it restricts the search space to consider only the rules following the reasoning given by this expert, it reduces the set of all the possible generalizations, thus the role of the preference bias will be strongly limited by this structure.

But with only this structure, the apprentice doesn't know the context of the fragment in the resolution process. This context is composed of the place of the fragment in the resolution process (i.e., what has be done, what will be done) and the control on this process. In the KADS model, this context is described in the task layer, and its use by ENIGME is the subject of the next part

5. The Task Structure

This structure is used to describe the chaining of the steps in the resolution process and the control on this chaining (see the figure 3 for the bridge example). The language used by the KADS formalism has one ground instruction: the fragment. A fragment matches one step of the problem solving method. For instance, the fragment *select_conv (syndrome1 valuation solution-classes) -> convention* (see figure 3) corresponds to the step of the resolution which decides if this resolution will use a convention. The other instructions describe the control over the resolution process. This control is mainly expressed through two features: the conditional instructions and the loops. ENIGME uses this control to discover which examples match which step of the resolution process.

```
Task {
  match (valuation) -> solution-classes
  If opening = yes[1]  Then {
     abstract1  (observables1) -> syndrome1
     select_conv (syndrome1  valuation  solution-classes) -> convention
     If  conventional-opening = club Or
          conventional-opening = no-trump then
               project (convention) -> solution
     Else {   abstract2 (syndrome1 observables2 convention) -> syndrome2
               classify (syndrome2 observables2) -> solution        }}
```

Fig. 3. The task structure of the bridge example.

The first consequence of the use of such a structure is on the examples' description. Indeed, they have to represent an experience of the problem's resolution. Thus, their description must include all the concepts involved in all the roles of the inference structure their resolution goes through. Then if these examples come from a data-base, we can have to complete these examples with all the intermediate concepts (like the abstractions,...). On the other hand, if there are several ways to solve the problem, the examples can have different descriptions. For instance, in the bridge example, there are two ways to reach the solution. If the hand is normal, we will follow the left branch of the inference structure and, so, an example of a normal hand will be described with the concepts involved in the roles: *valuation, observables1&2, syndrome1&2, solution-classes, convention and solution.* On the other hand, if the hand is an extraordinary one, we will follow the right branch of the inference structure and, so, the example will be described with the concepts involved in the roles: *valuation, observables1, syndrome1, solution-classes, convention and solution*[2].
Another major consequence of the use of task structure is the possible reasoning done by the produced rules. Indeed, in most of the current machine learning tools, this reasoning is, implicitly, done with one inference step which associates directly the solution with the description of the case. In this tool, thanks to these structures, the produced rules belong to a complete rule system. For instance, with the bridge

[1] In this version of ENIGME, the tests in the instructions are expressed with domain concepts. It may be useful (and more in the KADS spirit) for this control to be expressed in more abstract terms.

[2] In this example, the second description language is included in the first one, but, it is not necessary.

example, a resolution process may have up to four inference steps. The expert is much more confident with this kind of rule system than with a single rule which associates, directely, the observables to the solution.

6. Conclusion

The aim of this paper is to provide some basis for a real integration of Machine Learning and Knowledge Acquisition which could not just be reduced to a juxtaposition. To achieve this integration, we have developed a parallel among the Knowledge Acquisition models and the inputs of Machine Learning tools. It helps to establish that some parts of the learning bias can use some parts of the structured model of expertise, as it is presented in the KADS methodology, efficiently. As a conclusion, we want to draw attention to three points.

In the first place, the positive result of this work is the achievement of an integrated system named ENIGME which strongly couples Knowledge Acquisition and Machine Learning. It was empirically tested on a non trivial problem which is presented in the paper. We shall pursue its development by inserting it in an existing Knowledge Acquisition environment in the VITAL project.

In the second place, we have to note that we strongly coupled Knowledge Acquisition and Machine Learning. This strong couple is based on the fact that most of the inputs of the learning algorithms could be interpreted both in terms of their role in Machine Learning and in terms of problem solving models. This result is very significant by itself.

Lastly, the positive results we obtained need to be tempered since we did not succeed in assimilating all the learning bias to components of problem solving models: the preference-type bias has no interpretation in terms of Knowledge Acquisition models. Up to now, it only has some sense in terms of Machine Learning algorithms, its main role being to order the generalization space. In the future, we shall examine deeply the exact role of the preference-type bias and its meaning.

We thank the whole machine learning and knowledge acquisition team at the LAFORIA, particularly Karine Causse and Bernard Le Roux. The research reported here was carried out in the course of the VITAL project P5365 partially funded by the ESPRIT II program of the Commission of the European Communities.

[Ganascia 88] Jean-Gabriel Ganascia, *Improvement and refinement of the learning bias semantic.*, ECAI 88

[Mitchell 82] Tom M. Mitchell, *Generalization as Search.*, Artificial Intelligence V18 N°2 March 1982

[McDemott 88] John McDermott, *Preliminary steps toward a taxonomy of problem solving methods.*, In Automating KA for expert systems ed. by S. Marcus

[Russel 90] Stuart J. Russel and Benjamin N. Grosof, *Declarative Bias: An Overview.*, in Change of representation and inductive bias. ed. by P. Benjamin

[Utgoff 86] Paul E. Utgoff, *Shift the bias for inductive concept learning*, Machine learning: An Artificial Intelligence approach V. 1

[Wielinga 92] B.J. Wielinga, A.Th. Schreiber and J.A. Breuker, *KADS: A Modelling Approach to Knowledge Engineering.*, Knowledge Acquisition volume 4, number 1, march 1992.