A SMALLEST GENERALIZATION STEP STRATEGY

Claire NEDELLEC
email : cn@lri.lri.fr cn@isoft.uucp
L.R.I. Building 490
Université Paris-Sud, F-91405 Orsay France.

Abstract*

Over-generalization is a well-known problem in empirical learning. Incremental and prudent generalization is a means to avoid it. This is not always sufficient. The language in which the concepts are described may be incomplete, so that there is no conjunction to express a concept that is consistent with all the examples. This paper presents an interactive incremental learning method that generalizes in such a way that it is able to efficiently assist an user in locating the insufficiencies of the language and in correcting them whenever an over-generalization occurs. The generalization algorithm is based on a smallest generalization step strategy that determines processing order of the examples and the successive hypothesis to study.

1 INTRODUCTION

Empirically learning concept definitions requires the ability to generalize accurately. One approach to the problem of generalization consists in incrementally submitting examples to an oracle and asking him if they are representative of the target concept the system is looking for. The accepted examples are called positive, the rejected ones are called negative. The learner modifies its current hypothesis of the definition of the concept to learn according to the oracle's answers. At each learning step, the current hypothesis is consistent with all the tested examples.

As it is impossible to propose all the existing examples, it is possible that the learned concept verifies additional examples which have not been submitted to the oracle. If some of them are counter examples, the learned concept is said to be over generalized. The method described in this paper uses incremental and prudent generalization as a way to reduce this risk. Prudent generalization means that the successive hypothesis of the concept definition are always the most specific definitions that cover all the known positive examples, just as with timid generalization [Berwick R.C., 1986]. However, over generalization may nevertheless appear. Indeed, the description language may be insufficient, so that there is no way to express a concept definition that covers all the positive examples and no negative ones. The conception by an expert of a description language free from such insufficiencies [Shapiro, 83] is a critical aspect of the more general problem of knowledge acquisition .

At that time, the learner has no means of detecting the problem, since the learned concept is consistent with evrything the learner knows. The only solutions for the learner are either to have other strategic information about what is a good degree of generality or to ask the oracle to himself validate the learned concepts. In this paper we are interested in the latter approach which is safer in the case where the learned concepts must cover absolutely no negative example and all the known positive examples.

Then at the end of the learning phase, an expert is asked to evaluate the learned concept. If he detects an over generalization, the learned concept has to be specialized in such a way that it no longer covers any negative example . As it is the most specific description that covers all the known positive examples, it can not be specialized without positive examples being excluded. That means that it is necessary to modify the description language. The intervention of the expert is needed to integrate the missing vocabulary in the language and to modify the existent vocabulary. This would be a difficult task for an expert to locate and correct the causes of the over generalizations without further information.

We propose a method to tackle this problem whose generalization strategy gives it the capability to efficiently help an expert whenever an overgeneralization occurs. On one hand, it restricts the range of the search for the causes of the over generalization to a delimited sub-part of the vocabulary, and on the other hand, it guides the expert in the discovery of the missing vocabulary by presenting him positive and negative examples that are covered by the over generalized concept and that are *semantically close*.

2 PRUDENTLY GENERALIZING

* ISoft, Chemin du Moulon F-91490 Gif, France

This method is supported by implementation and experimentation in KAA, the learning module of APT. APT architecture is that of DISCIPLE, the multistrategy, integrated learning system [Tecuci, Kodratoff, 90]. KAA is able to incrementally learn the preconditions of application of problem-solving rules in a nonhomogeneous graph-structured domain theory by interactive learning and problem-solving sessions with an expert [Nédellec, 1990]. The concept definitions, that is to say the preconditons of the rules, and the examples are described with the same language which is restricted to the predicates of the domain theory. Then the improvement of the description language leads to the refinement and the completion of the domain theory.

KAA learns a general concept definition by using a specific example initially given by the expert. This example is expressed in the form of a conjunction of predicates completed by explanation that the system extracts from the domain theory. An example set is generated automatically by analogy with the completed example.
Initially, the search space of the concept definition is limited by a lower bound equal to the initial example and an upper bound equal to the same formula where all the predicates are replaced by the most general predicates in the corresponding hierarchies of the domain theory. Indeed, the predicate hierarchies determine completely the generality relation.

Next, the training examples are selected successively in the example set. As in [Mitchell, 1978] the search space converges incrementally on the target concept : each rejected example leads to a specialization of the upper bound so that the resulting upper bound is the most general possible without covering the rejected example. Each accepted example leads to a generalisation of the lower bound so that the resulting lower bound is the most specific one that covers all the accepted examples. The generalization of the lower bound respects the subset principle [Berwick R.C., 1986], in that two example subsets S_i and S_{i+1} covered *by any two successive states* of the lower bound LB_i and LB_{i+1} are so that $S_{i+1} \supseteq S_i$. The transformation of LB_i in LB_{i+1} is called the generalization step. Each generalization step corresponds to the changes a new positive example causes to the current lower bound in the course of the incremental learning. At the end of the learning phase, if not over generalized, the learned concepts are added to the problem-solving rules set that the problem solving module of APT uses. Otherwise a search for the causes of the over generalization begins .

To restrict the range of the search, the learner guides the expert in the identification of the generalization step where the over generalization occurs, as it is described in [Nédellec 1990]. The negative examples that must be excluded from the concept definition are covered by this identified generalization step and by no previous one. This property follows from the subset principle. That means that the words of the vocabulary that has to be corrected to suppress the over generalization belong to the subpart of the language that is involved in the transformation of the previous step in the over generalized one.

A means to reduce this subpart is to limit as much as possible the difference between two successive generalization steps with the "smallest generalization step strategy", so that a generalization step consists in generalizing just one predicate of the current lower bound, on one degree, according to the generality relation of the domain theory. That is to say, replacing a predicate in the current lower bound by its "father" in the corresponding theory hierarchy. As a generalization step is performed only if the generalization is necessary to cover a new positive example, the smallest generalization step strategy determines the processing order of the examples. So, the examples are successively chosen in such a way that those that are declared positive by the expert lead to the smallest transformation of the current lower bound.

Thus, at each step, KAA chooses an example, whose predicates are all less general than the corresponding ones in the lower bound, except one and this particular predicate is less general than the most specific generalization (a father) of the corresponding predicate in the lower bound (figures 5 & 6). If this training example is accepted by the expert, it is called "near success" and the consequent generalization of the lower bound is one of the most elementary ones, since just one predicate of the lower bound has to be generalized on only one degree to cover this near success, as shown in the figure 1 (the black arrows indicate a direct generality relation between two predicates, the grey ones indicate a generality relation by transitivity between two predicates)

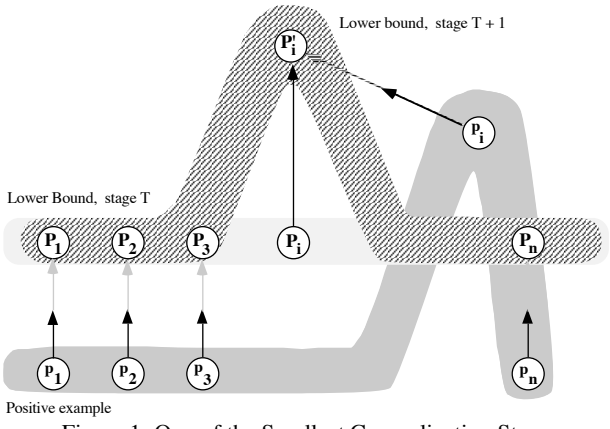

Figure 1: One of the Smallest Generalization Steps

If the example is refused by the expert, this near miss is excluded from the search space by the specialization of

the upper bound which is just the same as that described for the INBF algorithm [Smith, Rosenbloom, 90], which shows the efficiency of pruning the search space in that way. Since the guilty predicate is completely identified, (this is the only predicate that does not match the current lower bound) this allows to specialize the most the corresponding predicate in the upper bound, so that it becomes equal to the corresponding predicate in the lower bound, , without any risk of over specialization (figure 2).

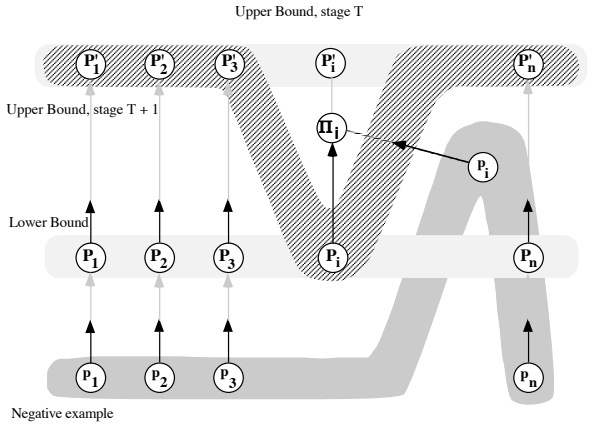

Figure 2: A Specialization Step

By this generalization strategy, the examples that are newly covered by a generalization step and not by the previous one are semantically close since their predicates have close common ancesters in the theory. Thus, when the generalization step responsible for the over generalization is identified by the expert, the comparison of positive and negative examples that are newly covered make sense for the expert. At that moment, he is able to find what is the missing vocabulary, which should allow to describe separetely negative and positive examples. By opposing close examples, it is significantly easier for the expert to complete the language than by just studying the successive generalization steps. Indeed, they are more abstract since they are more general than the examples.

3 EXAMPLE

KAA learns new rules by generalizing the condition part of rules such as the rule in figure 3. It describes how to build arches using three blocks.

IF *block*$_1$(x) & *block*$_2$ (y) & *block*$_3$(z)
& *supports*(x, z) & *supports* (y, z)

THEN BUILD arch ; solve the problem

; by solving the subproblems

ERECT x ERECT y LAY z

Figure 3: An Example of Rule in APT

The learning phase is divided into two different stages : the generalization stage and the revaluation stage.

3.1 GENERALIZATION

The search space of the target concepts is represented by an oriented generalization graph that is a boolean lattice [Ganascia, 1987]. Its root is the initial lower bound of the search space as defined above and the only leaf is the initial upper bound such as all its predicates are generalized the most in respect to the theory (figure 4). So they are equal to *something* in our example.

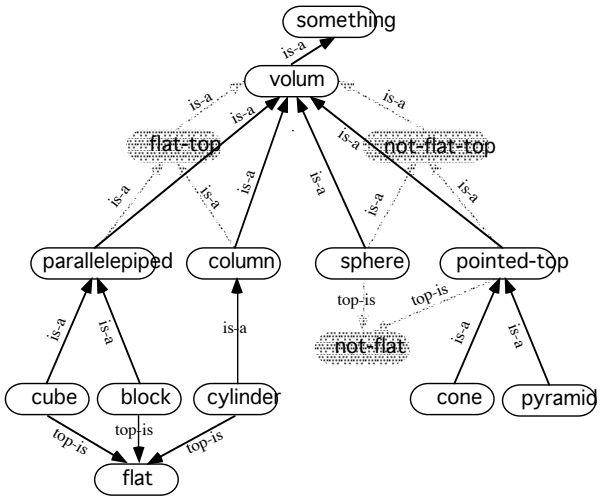

Figure 4: A Part of the Initial Domain Theory (in black)

The search space converges incrementally on the target concept. The algorithm ends when there are no more examples or the two bounds are equal.

3.2 EVALUATION

As the generalization algorithm ends, the expert evaluates with the help of the system the learned concepts represented by the disjunction of the most general nodes of the graph. If some of them are said to be over generalized, the revaluation phase starts.

In our example, at the end of the generalization phase, the concept associated with the most general accepted node of the graph is learned, it is noted C_n in the figure 5. Then, we suppose that the expert has accepted all the proposed

examples where the bases of the arch are *cubes* or *columns* and reject the examples where the base of the arch are not *volumes* (the top of the arch and the predicate *supports* remain unchanged).
It is submitted to the expert, who refuses it, as over-generalized. Indeed the expert knows intuitively it is impossible to build arches whose bases are any volume, but he is not yet able to complete the theory by himself so that this over generalization disappears.

C_n	*volume* (x) & *volume* (y) & *block* (z) & *supports* (x, z) & *supports* (y, z)
C_{n-1}	***parallelepiped*** (x) & *volume* (y) & *block* (z) & *supports* (x, z) & *supports* (y, z)
C_{n-2}	*volume* (x) & ***parallelepiped*** (y) & *block* (z) & *supports* (x, z) & *supports* (y, z)
C_{n-3} (z)	***parallelepiped*** (x) & ***parallelepiped*** (y) & *block* & *supports* (x, z) & *supports* (y, z)

Figure 5 : Four Related Concepts

When such an over-generalization is detected, it is necessary to specialize the current lower and upper bounds of the search space, so that they no longer cover the negative example but continue to cover all the known positive examples. Then, the system back tracks in the generalization graph until finding the most general nodes that are not over geneneralized.

The node C_n depends directly on the two nodes C_{n-1} and C_{n-2} (figure 5). The concepts represented by those two nodes are proposed to the expert who refuses both as being over generalized and points out the feature *volume* (that describes the two bases of the arch), as being too general. KAA back tracks by specializing *volume* and then proposes the concept of arch whose two bases are *parallelepipeds* (C_{n-3} in figure 5). The expert accepts it, since it is not over generalized. The problem is now that it is too specific, indeed it does not cover the positive examples that have allowed the generalization of the *parallelepipedic* base of the arch into *volume*. That is to say for instance, the example in figure 6.

cylinder (x) & *block* (y) & *block* (z) &
supports (x, z) & *supports* (y, z)

Figure 6 : A Positive Example of Arch

The most specific over-generalized nodes, C_{n-1} and C_{n-2}, are now identified, as well as the inconsistent example set which contains the negative examples that caused the over generalization, covered by C_{n-3} and not by C_{n-1} and C_{n-2}.

3.3 CORRECTING THE DOMAIN THEORY

After backtracking, KAA still has to identify the negative examples that belong to the identified inconsistent set, so that it will be able to help the expert in the correction of the domain theory. With this aim, it proposes to the expert examples of the inconsistent set whose bases are *volumes* but not *parallelepiped* such as *pyramid*, *sphere* or *column*.

pyramid (x) & *block* (y) & *block* (z) &
supports (x, z) & *supports* (y, z)

Figure 7: A Negative Example of Arch

When some of the negative examples are identified (figure 7), they are compared to positive examples that belong to the same set (figure 6) ; this comparison is significant for the expert, since the compared positive and negative examples are semantically close. Then, the expert is asked what differentiates *pyramid* or *sphere*, which appear in rejected examples, from *cylinder*, that appears in previously accepted examples.

The only solution to express the definition of concept of arch that covers all the positive examples and no negative ones, is to add new constraints to the over generalized definition and/or to complete the domain theory if the available vocabulary is insufficient.

Then KAA proposes various means as to group *pyramid*, *sphere* and *cone* in the same cluster and *column* and *parallelepiped* in another one and next, to choose the roots of these clusters among the existing predicates or to create new predicates (such as *flat-top* and *not-flat-top* in figure 4). KAA proposes also, to add a new relation to *pyramid*, *sphere* and *cone* (such as *top-is* related to the feature *not-flat*). Notice that this solution is better, the type of the top is just one aspect of the volums, moreover it avoids adding intermediate predicates that may cause complex treatments because of multiple inheritance. It proposes too to modify some of the existing relations to eliminate some ambiguities of the language, it is particularly usefull when the same predicate has two different meanings. Finally, KAA proposes to add constraints to the learned concepts to specialize them such as in figure 8.

volume (x) & *volume* (y) & *block* (z) & *supports* (x, z) &
supports (y, z)
& *top-is* (x, t) & *top-is* (y, t) & *flat* (t)

Figure 8: Constraining an Overgeneralized Concept

In this way, the negative examples are isolated from the positive ones and there now exists a way to express conjunctively the concept of building arches.

We have chosen this toy problem to illustrate the method, that has been also applied to real world problems.

4 RELATED WORKS

The algorithm of MARVIN, described in [Sammut and Banerji 86] presents some similarities with our work although it does not address the problem of identifying the causes of an over generalization but the problem of correcting them. The rewrite rules of the domain theory are used to generalize such as the premises are replaced in the current hypothesis by the rule head. The generalization strategy consists in executing, step by step, one of the elementary possible generalizations by applying just one rewrite rule. At each step, MARVIN submits to an expert an example that matches the current generalization but not the previous one. If the example is accepted, the generalization is carried out. If not, the system tries to specialize in the direction of the previous generalization. If no specialization is possible, the system backtracks and chooses another rewrite rule, among the possible ones. Our method differs on this last point. Indeed the domain theory representation in KAA is a semantic network and then the equivalent rewrite rules have just one premise unlike MARVIN rules. Although the representation by rewriting rules that can have more than one premise is more powerful, that means also that the smallest generalization step strategy in KAA leads to replace just one predicate by another one in the current hypothesis, when in MARVIN this strategy leads to replace by one predicate all the premises of the chosen rule. The consequences are various. First, in KAA no concept definition exists between two successive generalizations. Then, if an example is refused, KAA does not need to try to specialize *in the direction of* the previous generalization. Moreover, the examples that are newly covered by a step are less various, since two successive steps differ by only one predicate and not by several. That make the search for the cause of the over generalization easier since it is based on the comparison between these examples. Indeed, the less semantically different they are, the easier is it to compare them.

[Utgoff, 86] has addressed the problem of choosing and shifting the bias that determine how generalization are performed. The language in which concepts are described is one of the biases that Utgoff has presented. As the description language determines the limits of the hypothesis space, extending it to express more concepts means in effect to complete the language. Then, as we do, Utgoff studies how to change an incomplete language, (the syntax remaining the same) in the aim of expressing concepts that are consistent with example sets.
Utgoff's method named STABB is applied to LEX [Michell, 82]. It consists in incrementally building a consistent and disjunctive definition of the target concept such as the disjuncts are as few as possible. At the end of the generalization phase, if the concept definition remains expressed in a disjunctive form, a new symbol is automatically integrated in the language so that the concept may be expressed in a conjunctive form. KAA generalizes in a quite similar way but it does not automatically create new symbols. Expert abilities are required to modify the language, indeed, the needed modifications may be far more important than adding intermediate features in hierarchies as STABB do. So by interacting with an expert KAA is able do perform other changes such as those described above. More than completing a language, KAA refines it.

5 CONCLUSION

In this paper, we have presented a method to limit over generalization when empirically learning, by correcting the concept description language. It is based on prudent and incremental generalization and evaluation of the result with the interactive help of an expert of the domain. If necessary, the system guides him in locating the causes of the problem by comparing positive and negative examples which are semantically close examples of the same concept. Various means are next proposed to refine the subpart of the language that has been identified as incorrect.

Acknowledgements

The author would like to thank Yves Kodratoff for the support he has provided to this work, Celine Rouveirol and Gilles Bisson for their comments and suggestions.

This work is partially supported by CEC, through the ESPRIT-2 contrat MLT (n°2154).

References

BERWICK R.C., "Learning from positive-only examples : The subset principles and three cases studies", in *Machine Learning II : An Artificial Intelligence Approach*, pp 625-646, RR.S. Michalski J.G. Carbonell and T.M. Mitchell (Eds) Morgan Kaufmann, 1986.

GANASCIA, J.G, "*AGAPE et CHARADE : Deux techniques d'apprentissage symbolique appliquées la construction de bases de connaissances*.", Thèse d'Université, Université de Paris-Sud, 1987.

MITCHELL, T. M., "*Version Spaces : An Approach to Concept Learning*", Ph.D. diss. Standford University, 1978.

MITCHELL, T.M., "*Generalization as Search*", Artificial Intelligence 18, pp. 203-226, 1982.

NEDELLEC C. "*Smallest generalization step strategy to deal with over generalization*", Research Report 626, Université de Paris-Sud, 1990.

SAMMUT, C.A AND BANERJI, R.B., "Learning Concepts by asking Questions", in *Machine Learning II : An Artificial Intelligence Approach*, pp 167-192, R.S. Michalski J.G. Carbonell and T.M. Mitchell (Eds) Morgan Kaufmann, 1986.

SHAPIRO, E. Y. "*Algorithmic program debugging*", MIT press, Cambridge, MA, 1983.

SMITH B. D. AND ROSENBLOOM P. S., "Incremental Non-Backtracking Focusing: A Polynomially Bounded Generalization Algorithm for Version Space.", in *AAAI-90 : proceedings of the ninth conference*, pp 848-853.

TECUCI G. & KODRATOFF Y., "Apprenticeship Learning in Nonhomogeneous Domain Theories", in *Machine Learning III : An Artificial Intelligence Approach*, pp 514-552, Kodratoff Y. & Michalski R. (eds), Morgan Kaufmann, 1990.

UTGOFF P.E., "Shift of bias for inductive concept learning", in *Machine Learning II : An Artificial Intelligence Approach*, pp 107-148, RR.S. Michalski J.G. Carbonell and T.M. Mitchell (Eds) Morgan Kaufmann, 1986.