Inductive Learning of Characteristic Concept Descriptions from Small Sets of Classified Examples

Werner Emde

GMD, FIT.KI
Schloß Birlinghoven
D-53757 St. Augustin
Germany
email: werner.emde@gmd.de

Abstract. This paper presents a novel idea to the problem of learning concept descriptions from examples. Whereas most existing approaches rely on a large number of classified examples, the approach presented in the paper is aimed at being applicable when only a few examples are classified as positive (and negative) instances of a concept. The approach tries to take advantage of the information which can be induced from descriptions of unclassified objects using a conceptual clustering algorithm. The system COLA is described and results of applying COLA in two real-world domains are presented.

1 Introduction

The learning task considered in "learning-from-examples" is to induce a description of a set of objects which are classified by a user or a system as instances (and non-instances) of a general meaningful class, i.e., a concept. The concept description is supposed to help to determine other instances (and non-instances) of the concept. Some examples of well known systems which are able to learn from examples are: AQ [Michalski 73], ID3 [Quinlan 83], FOIL [Quinlan 86], and RDT [Kietz/Wrobel 92].

Although previous research on learning-from-examples has lead to impressive results, the existing approaches are not able to explain an important aspect of human concept learning – namely the ability to learn new concepts from very few examples (see [Carey 85]). We are not interested here in constructing a cognitive model of human learning, but look for some kind of knowledge and/or generalization method which may help us to learn from few examples. However, the problem of learning from a small set of examples is also of great practical relevance. In some application domains, a classification of large number of examples is not available or the acquisistion of the data is too expensive. Also autonomous agents may sometimes be required to improve their behavior from very limited experience.

In the following, it is argued that the use of additional background knowledge (which is often available, but not used in other approaches) enables programs to

learn characteristic concept descriptions from small sets of positive and negative examples only. The approach described in this paper tries to take advantage of the information contained in descriptions of unclassified objects (as a kind of additional background knowledge) using a conceptual-clustering algorithm.

In the next section, the consequences of the number of examples available for concept learning on the quality of learning results is analyzed. In the third section of the paper, the general idea of the new approach based on the use of a conceptual-clustering algorithm is described. Section 4 describes an implementation of the new technique in the system COLA. Experimental results obtained by applying COLA to different real-world data sets are presented in section 5. We finish with a discussion of related work (section 6) and some final remarks.

2 Learning from few examples

In the literature two kinds of generalized descriptions are distinguished: *characteristic descriptions* and *discriminant descriptions* [Dietterich/Michalski 81]. A *characteristic description* of a class of objects describes the sufficient conditions for class membership and enables a system to identify all instances of the class and reject all instances of other (disjoint) classes. A *discriminant description* can be applied to discriminate between instances of one class from instances of a pre-defined set of other classes. As a discriminant description needs only to specify properties relevant in the context of a fixed set of other classes, these descriptions are usually more general.

These two kinds of descriptions can be related to two kinds of generalization algorithms: algorithms constructing *most-specific generalizations* and algorithms constructing *most-general generalizations* from a given set of positive and negative examples. A most-general generalization algorithm generalizes the descriptions of the positive examples to the most general description in the hypotheses space such that (only) the negative examples are excluded.[1] A most-specific generalization algorithm generalizes the descriptions of the positive examples to the most-specific description in the hypothesis space covering just all positive examples and excluding all negative examples. Most-general generalizations are built by programs like AQ [Michalski 73], GOLEM [Muggleton/Feng 90][2], RDT [Kietz/Wrobel 92], and FOIL [Quinlan 86]. Examples of programs, which construct most-specific generalizations are ARCH [Winston 75] and OGUST [Vrain 90]. A most-specific generalization algorithm tends to produce characteristic descriptions. A most-general generalization algorithm tends to produce discriminant descriptions.

Whether an intended learning goal can be achieved heavily depends on the number of available learning examples. If the number of examples is large, both kinds of generalizations are likely to deliver descriptions which are complete and

[1] It may also be the case that an algorithm searches for a borderline "in the middle" between positive and negative examples.

[2] GOLEM produces first a most specific generalization, which is reduced to a most general generalization in a later step.

correct not only on the training set, but on the future test sets as well. If only a small number of examples is available, a most-specific generalization is still likely to be correct, but will probably be incomplete. All objects predicted as instances of the goal concept are instances, but several instances will not be predicted. A most-general generalization from few examples on the other hand, is still likely to be complete, but will tend to be incorrect.

The consequence of this analysis is that many current learning programs are able to learn from relatively few examples, but the classification accuracy of the concept description is likely to be bad if it is used as a characteristic description. Current inductive learning programs cannot be improved simply by changing their generalization algorithm. What we can do is extend the learning input and improve the algorithms so that they can make use of it. An approach following this direction has already been developed with the system IOU [Mooney 93] which combines empirical and explanation-based learning. This system requires an overly-general domain theory as additional background knowledge in order to be able to learn more accurate concepts from fewer examples than other learning systems (section 6).

3 Conceptual-clustering–based generalization

For concept learning problems where only a relatively small number of positive (and negative) examples is known, but a large set of descriptions of unclassified objects is available, we propose a generalization approach called "conceptual-clustering–based-generalization" (CCG) incorporating a conceptual clustering step.

3.1 Conceptual Clustering

The learning task of a conceptual clustering algorithm can be described as follows. From a given set of descriptions of (unclassified) objects of a domain, a conceptual clustering algorithm is supposed to aggregate the objects into an organized set of meaningful classes and to find (intensional) descriptions of these classes. A class is regarded as meaningful if the organization of knowledge becomes more effective and/or more efficient, e.g., enables a system to infer missing information for partially described objects. Some conceptual clustering programs (e.g., CLUSTER [Michalski/Stepp 83] or COBWEB [Fisher 87]) construct a hierarchy of classes, other programs (e.g., UNIMEM [Lebowitz 87], KBG [Bisson 92b]) construct a directed acyclic graph of classes, which means that the classes are not necessarily disjoint.

An example of a directed-acyclic graph of classes constructed by the conceptual clustering system KBG is shown in figure 1 below. It is a sub-graph of a graph constructed from descriptions of unclassified diseased soybean plants (see section 5), i.e., without the information about the correct diagnoses. The leaves of the graph show the identifier numbers of the different plant disease cases. The nodes are the classes formed by KBG labelled with identifier numbers of

the classes. The soybean plant numbered 5 is an instance of class 39, class 38, and class 44. Such non-disjoint classes can be interpreted as capturing different views/similarities on a set of objects.

3.2 A Rough Description of CCG

Following this rough characterization of conceptual clustering, we can now describe the basic idea behind the CCG approach. We assume that the instances of the goal concept are similar to each other and that this similarity can be uncovered in the set of unclassified instances. Thus, we use the set of descriptions of unclassified instances to find a set of classes which capture different similarities among the instances. This is achieved using a conceptual clustering algorithm. The (small) set of classified examples is then used to identify one of the classes produced by the clustering algorithm as the class which captures the similarity of instances of the goal concept. The selection of the class is done after comparing the extensions of all classes with the positive and negative examples. The intensional description of the selected class (produced by the clustering algorithm) is finally proposed as a description of the goal concept. All other classes and class descriptions produced by the clustering system are not incorporated into the knowledge base of the overall system.

The conceptual clustering step can been seen as building a space of possible generalizations. The second step of selecting one class using the examples can been understood as search in the hierarchy (or graph) of generalizations.

The induced concept description covers the positive examples and those unclassified instances used in the clustering step, which are 'similar' to the positive examples. It excludes all negative examples and the unclassified instances, which are not 'similar' to the positive examples. Thus, the induced description of the goal concept is neither a most-specific generalization nor a most-general generalization of the sparse set of examples. The description covers more instances than the most-specific generalization, but less instances than the most-general generalization. In addition, the description produced by the CCG method tends to be intensionally more specific (e.g., the description contains more premises) than a description built by a most general generalization, because the class descriptions built by the clustering algorithm usually need to discriminate between a large number of classes. Therefore, we say that the CCG approach delivers a *characteristic* description of instances of the goal concept.

Before we discuss the details let us state the learning problem this paper deals with more clearly:

Given:

- B: background knowledge in the form of descriptions of objects in a domain,
- C: the name of a goal concept which is not used in the description of the objects in the background knowledge B, but would be helpful in order to complete the descriptions, and

- E: a (small) set of positive (and negative) examples of the instances of the concept C which are already partially described with the background knowledge B.

Find:

- H: an intensional characteristic concept description of the concept C, which covers all (or at least a large number of) the positive examples as well as similar objects described in B and excludes all (or at least most of) the negative examples in E as well as similar objects described in B.

3.3 Details of the CCG Approach

In the above rough description of the CCG approach, several important details are missing. For example, how should a learning system deal with the case that more than one class satisfies the requirement that all positive examples and no negative example is covered? Suppose case 15 (see figure 1) is described as

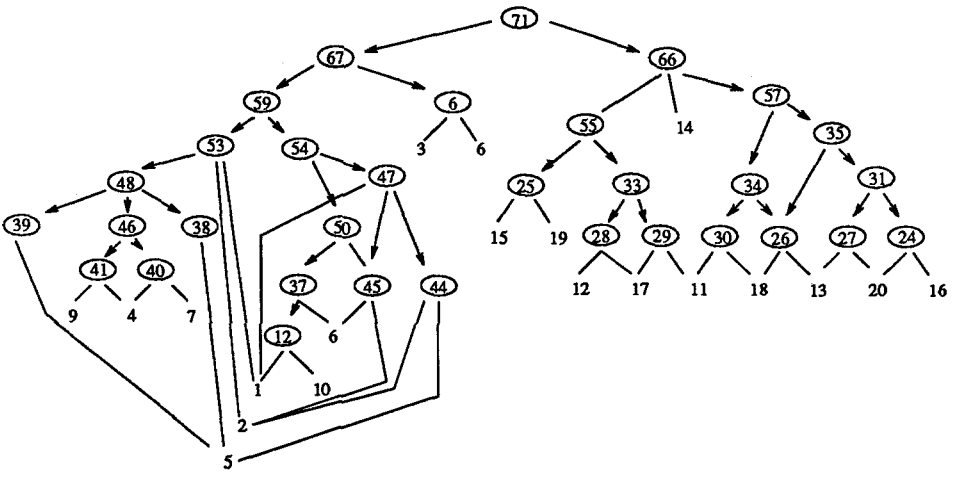

Fig. 1. Classes of diseased soybean plants

positive example and case 3 is described as negative example. The requirement is satisfied by class 25, class 55, and class 66. The first step to deal with this problem is to identify two extreme cases of possible generalizations: the generalization which delivers the *most specific CCG* and the generalization which delivers the *most general CCG*.

The most specific CCG of examples may be computed by climbing the graph from a leaf with a positive example up to the node in the graph covering all other positive examples, the least number of total instances, and no negative

examples. This type of generalization is especially appropiate if no negative examples are available to prevent over-generalizations. The most general CCG results from climbing the graph to the top-most class covering all other positive examples, the largest number of total instances, and no negative examples. In the last example, the definition of class 25 would be the most specific CCG and the description of class 66 would be the most general CCG.

It may be interesting to know that class 67 (see figure 1) covers all instances of the soybean disease Diaporthe Stem Canker; furthermore, class 66 covers all soybean plants suffering from Charcoal Rot. This clustering result means that conceptual-clustering-based generalization is able to deliver 100% correct concept descriptions of both soybean diseases from one example per disease only (if the positive examples of one disease is used as negative example for the other class). For example, if case 1 is given as positive example of Diaporthe Stem Canker and case 14 is given as positive example of Charcoal Rot, the most general CCGs of the examples would be the descriptions of class 67 and class 66. On the other hand, the most specific CCGs are class 17 and class 66. In this case, the generalization of case 1 is correct, but extremely incomplete.

The type of generalization a system should perform depends on the intended use of the description and the availability of negative examples. In this paper we assume that the user specifies which kind of generalization the system should perform and discuss in section 5 the classification accuracy of both kinds of generalizations in different domains.

If a clustering algorithm is used which delivers non-disjoint classes, there is still the problem that more than one class fulfills the above requirements. If it is acceptable that a learning system sometimes selects the wrong class, e.g., because the system contains a knowledge revision component able to repair faults in the knowledge base, the system might choose one of the competing class descriptions randomly. Otherwise, the selection should be made interactively with the user of the system.

Up to now we assumed that the conceptual clustering result contains a class covering all positive examples of the target concept. This need not always be the case. If the number of descriptions of unclassified objects is too small, if the described objects represent a non-representative subset of objects in the domain, if the data are noisy, or if the object descriptions are not well suited for the intended use of the descriptions, then a system may run into difficulties. Thus, it can not be assumed, that a class can be found which covers all positive and/or excludes all negative examples. In order to overcome this problem, we propose to use an evaluation function which computes a quality measure for the classes built by the clustering algorithm and enables the system to select one promising class description. The system COLA (see section 4) uses a evaluation function derived from a function to determine the accuracy of concept descriptions and computes a high quality measure for classes which cover a large number of positive examples and exclude a large number of negative examples. If two (or more) classes have the same quality, one is selected randomly.

However, the use of an evaluation functions is of no help if a target concept

can not be described by a conjunctive description. This poses a severe problem, because conceptual clustering systems typically generate only conjunctive class descriptions and the CCG approach described so far is only able to output concept descriptions built by the conceptual clustering algorithm. Although it has been argued that conjunctive descriptions are one of the most common descriptive forms used by humans [Michalski/Stepp 83], it is clear that disjunctive concepts can also be relevant. An obvious solution to this problem is to select a set of classes, which cover together all positive and exclude all negative examples, and to connect the corresponding class descriptions disjunctively.

In this section, the general idea of the CCG approach generalization has been described. The method is independent from a specific choice of a representation formalism and is also independent from a particular conceptual-clustering algorithm. The next section describes the details of an implemention of the method in a system called COLA.

4 COLA: The System

COLA is an inductive learning tool in the knowledge acquisition and machine learning system MOBAL [Morik et al. 93] and makes use of MOBAL's knowledge representation environment. This means that COLA uses an extended function-free Horn-clause representation (paraconsistent with negation) [Morik et al. 93, p. 27ff].

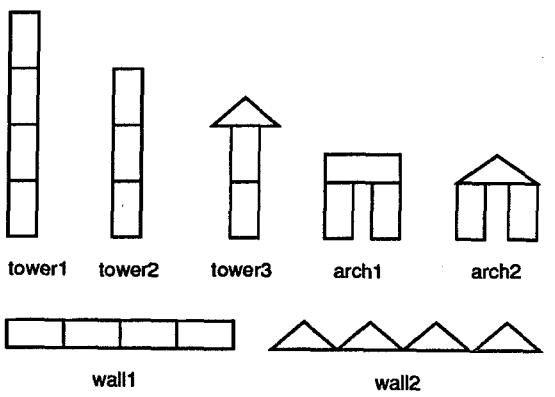

Fig. 2. Various block world objects

The attributes of objects and relations among objects in the domain are described by facts. A set of facts related to one building in a blocks world domain (see figure 2) is shown as an example in table 1.

```
block_world_object(arch2)
part_of(arch2_brick1,arch2)
part_of(arch2_brick2,arch2)
part_of(arch2_wedge,arch2)
brick(arch2_brick1)
brick(arch2_brick2)
wedge(arch2_wedge)
left_of(arch2_brick2,arch2_brick1)
not(touches(arch2_brick2,arch2_brick1))
supports(arch2_brick1,arch2_wedge)
supports(arch2_brick2,arch2_wedge)
on_table(arch2_brick1)
on_table(arch2_brick2)
larger_than(tower1,arch2)
```

Table 1. Facts describing a block world object shown in figure 2

Before the CCG learning algorithm of COLA and the conceptual clustering program used in COLA is described, let us start with an example of a learning result achieved by COLA in the blocks world domain. As background knowledge, facts (like those shown in table 1) about all 'buildings' shown in figure 2 were added to the knowledge base. In addition, the following facts of an instance and a non-instance of the concept `is_an_arch` were added as positive and negative example to the knowledge base.

```
is_an_arch(arch2)
not(is_an_arch(tower1))
```

The description for the concept `is_an_arch` induced by COLA from this information is shown below. The description covers both arches and excludes all other buildings shown in figure 2. Note that only one positive and one negative example were necessary to build the description.

```
on_table(X4) & part_of(X1,X2) & ne(X4,X1) &
brick(X4) & supports(X4,X1) & supports(X3,X1) &
ne(X3,X4) & not(touches(X3,X4)) & on_table(X3)
                    -->is_an_arch(X2)
```

4.1 KBG-2

COLA uses the conceptual clustering program KBG-2 [Bisson 92b] to perform the conceptual clustering step. The advantage of KBG-2 compared to systems like UNIMEM or COBWEB lies in the fact that KBG's knowledge representation language is based on first-order logic (without negation and function symbols)

with some extensions, e.g., for numerical values. Based on this, COLA is able to learn from relational descriptions of objects and to deal with numerical data.

KBG requires as input a case-oriented representation, i.e., a set of object descriptions each in form of a conjunct of ground literals and, optionally, a set of rules as domain theory. The output of KBG consists of a graph of classes and a system of rules enables to predict the instances of the classes.

The conceptual clustering of KBG can be divided into three successive steps. In the first step, the description of the examples are saturated using a domain theory which can be specified optionally. COLA does not need to pass a domain theory to KBG, since (forward-inference) saturation with available rules is performed in MOBAL. In the second learning step, a set of generalization and clustering operators are applied iteratively in a bottom-up fashion guided by similarity measures in order to build a graph of generalizations. The last step aims at building a hierarchical system of rules from the generalization graph. In this step, KBG drops all premises in the class descriptions which are not neccessary to discriminate between the instances of different classes. For a detailed description of the learning step see [Bisson 92a, Bisson 92b]; the rule construction step is described in [Bisson 91].

4.2 Construction of the KBG input

In research on conceptual clustering it is usually assumed that the descriptions of the objects for clustering is given as a conjunction of attribute-value pairs. This assumption makes sense for a lot of applications, because such data is often available from existing databases. The assumption poses difficulties in knowledge based systems using a more powerful representation based on predicate logic, especially if the representation is fact-oriented. If, e.g., a domain is described by a set of facts and rules describing the attributes and relations between several kinds of objects, there is no naturally given input for a clustering algorithm. An obvious answer to this problem is: Collect all facts related to an object to be clustered and make a big conjunction. Unfortunately, this is not a very fruitful idea using a well elaborated knowledge base, because in the worst case it may turn out that each object is related somehow to all other object in the domain. So, each conjunction would contain all facts in the knowledge base.

The system COLA uses currently a language restriction to deal with this problem and to avoid a overly large search space. Using this language restriction COLA is able to learn 1-1 (non)determinate clauses (see [Muggleton/Feng 90]) with conclusions described by one-place predicates. As the restriction is not relevant for the experiments which are described in this paper, we will not go into details of the transformation process here.

4.3 Selection of a class description in COLA

All class descriptions formed by KBG are translated into rules of the MOBAL representation. This translation includes the transformation of the case-oriented

representation of the clustering program back into the fact-oriented representation. These resulting rules are applied to all facts stored in knowledge base in order to compute the extension of all classes formed by KBG.

In the next step, COLA searches for disjunctive combinations of the classes constructed by KBG. The maximum number of classes which are combined disjunctively can be specified by the user. As default COLA connects up to 4 class descriptions disjunctively. Only those classes covering at least one positive example and no negative example are taken into account in this step. From these classes COLA creates disjunctive class descriptions and adds them to the class descriptions formed by KBG. In order to avoid a too large search space and to promote the creation of meaningful classes COLA combines only classes whose extensions do not intersect. Furthermore, it is required that the number of covered positive examples is increased by the combination.

In the next step, the following evaluation function is applied to all class descriptions stored in the knowledge base.

ClassQuality(NumCovPos,NumCovNeg,NumPosE,NumNegE) :=
(NumCovPos + NumNegE − NumCovNeg) / (NumPosE + NumNegE)

where:
NumCovPos is the number of covered positive examples
NumCovNeg is the number of covered negative examples
NumPosE is the total number of positive examples
NumNegE is the total number of positive examples

Then, COLA selects the class with the best evaluation. If two or more classes are assigned the same quality, COLA selects the class covering the least number of instances (classified and unclassfied objects) if the system is searching for a most specific generalization. If the system is searching for the most general generalization, the system chooses the class covering more instances. If the same number of instances is covered, COLA selects the class whose class description contains the least number of disjuncts.

The selected class description is used to build the final concept description by renaming the conclusion predicate, e.g., a conclusion like instance_of_class13(X) is changed into `is_an_arch(X)`. This description is added to the knowledge base (while all class descriptions are deleted) and the concept description becomes available to the overall system as well as to following learning processes.

5 Experimental Results

COLA has been applied in several real-world domains in order to test if and under which circumstances the CCG approach improves upon existing methods. In the following we report only about experiments conducted in domains which can also be represented using an attribute-value representation. This has the advantage

that the results can also be compared with results achieved by learning programs not able to learn in relational domains (e.g., C4.5 and IBL).[3]

5.1 Test Domains

We present the results of experiments done in two well-known domains which represent extreme cases: the soybean domain with a highly regular data set and the primary tumor domain with a data set known to be relatively incomplete, i.e., not sufficient to induce high quality rules even if a large number of examples is supplied to a learning program. For example, C4.5 and IBL achieve an accuracy between 33%–38% using 275 examples if they are applied to learn 22 class descriptions [Aha et al. 91].

The soybean domain data set [Stepp 84] contains descriptions of 307 diseased soybean plants. Each plant is described by 35 attributes and suffers from one of 19 soybean plant diseases present in the data set. The learning task in this domain was to induce the description of four soybean diseases: Diaphorte Stem Canker (with 10 instances in the data set), Charcoal Rot (10 instances), Rhizoctonia Root Rot (10 instances), and Phytophthora Rot (40 instances). The 18 boolean attributes in the data set and the diagnoses are described in MOBAL using a one-place predicate. All other nominal attributes are represented with a two-place predicate. If the attribute value is missing in the original data set, a corresponding fact is not stored in the knowledge base of MOBAL.

This primary tumor domain data set[4] describes 339 tumor cases. Each case is described by 17 attributes and is assigned to one of 22 possible locations of primary tumor. For our experiments we selected the following four locations covering different numbers of instances as goal concepts: lung (84 instances), head/neck (20 instances), esophasus (9 instances), and breast (24 instances). The data set contains 14 boolean attributes and 3 nominal attributes which are described in the knowledge base using one-place and two-place predicates.

5.2 Learning Programs

In order to be able to make general statements about how the apprach implemented in COLA improves upon existing methods it was necessary to apply also other learning programs to the same learning data. We chose the learning program FOIL-6 [Quinlan 86] as a representative of systems which produce most general generalizations. As a representative of systems performing most specific generalizations we applied a program using the well-known least general generalization rule of Plotkin [71] to the positive examples (without considering the negative examples).

Note that FOIL-6 and the LGG program were no expected to produce learning results which show the best classification accuracy of all applicable learning

[3] The learning data stem from the UCI Machine Learning data repository.

[4] The data set was collected by M. Zwitter and M. Soklic at the Ljubljana University Medical Centre in Slovenia.

programs of their class in both test domains and under all test conditions considered in the experiments. Therefore, we were not so much interested in the absolute values of the classification accuracy of their learning results, but the general effect caused by the occurrence of instances of non-goal concepts in the test set.

5.3 Test Method

The learning programs were applied to induce descriptions of the four goal concepts in both domains. In each test run they were supplied with a small number N of randomly selected descriptions of instances of each goal concept including the correct diagnoses. All programs were supplied with the same (MOBAL-) representation of the data. FOIL-6 and COLA used the positive examples of one class as negative examples for the other goal concepts. So, COLA and FOIL-6 used N positive examples and N*3 negative examples to induce a concept description. As additional background, knowledge COLA was supplied with a varying number M of randomly selected descriptions of instances of goal (and sometimes non-goal) concepts in the domain without information about the correct diagnoses.

The accuracy of the learning result was tested using the set of descriptions of instances not given as examples. The concept descriptions induced by a program in one trial were tested one after another. The first concept description was applied to all test instances, the second was applied to all instances not classified as instances of the first concept and so forth. The classification accuracy results discussed below are averaged over 20 to 30 trials.

In a first set of experiments described in the next section we used the learning programs to induce descriptions of the goal concepts which were then applied to classify the remaining unseen instances of the goal concepts. In another set of experiments (section 5.5) we evaluated the classification accuracy of the induced concept descriptions applied to the descriptions of instances of non-goal concepts as well as the remaining unseen instances of the goal concepts.

5.4 Test Data Set Describes Instances of Goal Concepts

The kind of experiments described in this section is similar to the experiments described in other papers in the field of machine learning except that the number of examples supplied to the learning programs is much smaller than usually.

As described above, COLA can be parameterized to output the most general CCG or the most specific CCG. We let COLA build both kinds of generalizations from different numbers N (between 1 and 10) of examples per goal concept and varied the number of descriptions of unclassified instances (of the goal concepts) which were supplied as background knowledge. The goal concepts in the soybean data set cover altogether 70 instances. So, if M is set to 70, all descriptions of instances of all goal concepts are used in the conceptual clustering step. If M is set to 0, the conceptual clustering graph is built only from the N*4 example descriptions. The goal concepts in the primary tumor data set cover altogether

137 instances. Therefore, the maximum number of unclassified descriptions in the backgroud knowledge was 137.

The following hypotheses were tested:

H1 The classification accuracy of the learning result increases with the number of examples N per goal concept. At some point, the number of examples is sufficient to select the best class descriptions in the conceptual clustering graph. If this point is reached, the classification accuracy can not be improved any more using more examples.

H2 The classification accuracy of the learning result increases with the number of descriptions of unclassified instances.

H3 The classification accuracy of a most general CCG is better than the accuracy of the most specific CCG. The most specific CCG covers only instances very similar to the examples. Therefore, a large number of instances will not be predicted as instance of one of the goal concepts. As all cases described in the test data sets are instances of the goal concept, a most specific CCG may be correct, but tends to be very incomplete. The most general CCG may be correct and tends to be complete.

H4 At least in the highly regular soybean data set, COLA will output concept descriptions with a better classification accuracy than FOIL-6 and the LGG program.

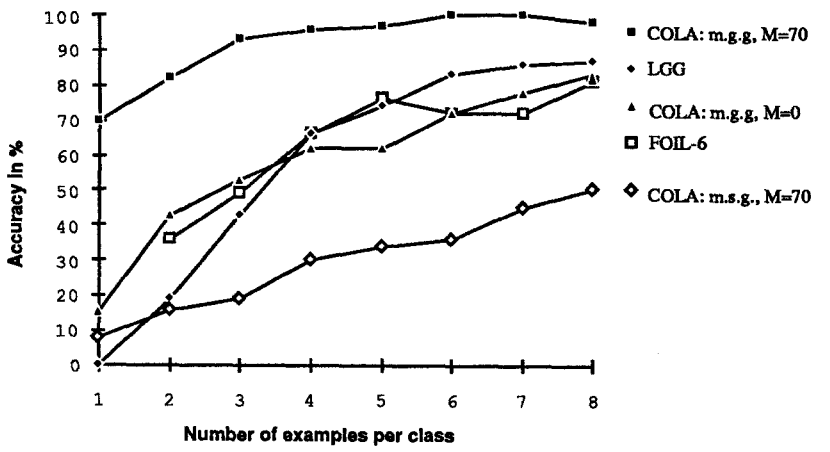

Fig. 3. Test Results in the Soybean domain without Instances of Non-Goal Concepts

The experimental results confirming these hypotheses to a large degree are shown in figure 3 and figure 4. Hypothesis H1 is only partially confirmed by the learning curves, because only two curves clearly show that the accuracy remains

Fig. 4. Test Results in the Primary Tumor Domain without Instances of Non-Goal Concepts

constant after a certain number of examples per class. Hypothesis H2 seems to be correct, but other experiments have shown that the classification accuracy of most specific CCGs can become better if less descriptions of unclassified instances are supplied as background knowledge. This can be explained by the fact that a larger number of descriptions of unclassified instances results in a conceptual clustering graph with more intermediate classes. Although this graph will contain a class covering all instances of a concept, the probability that COLA selects a more specific class increases. The hypotheses H3 and H4 are confirmed by the test results in both domains. As expected, COLA performs better in the soybean data set, but in the primary tumor data set FOIL-6 is the winner.

The conclusion which can be drawn from this first set of experiments is that existing methods should be preferred over the method described in this paper if it is known that the learning result will be applied only to classify instances of goal concepts, because usually it is unknown that a domain is highly regular (as is the case with the soybean domain, where COLA performs much better).

5.5 Test Data Set Describes Instances of Goal and Non-Goal Concepts

In the second set of experiments the concept descriptions built by the learning programs were applied to all descriptions of instances of the goal concepts not supplied as examples and all descriptions of instances of non-goal concepts.

Following hypotheses were tested:

H5 The classification accuracy of a most specific CCG is better than the accuracy of the most general CCG, because there are a large number of instances of non-goal concepts and a most general CCG can cover also instances of non-goal concepts.

H6 The classification accuracy of a most specific CCG is better than a description built by a most general generalization algorithm , because the latter will contain only descriptors necessary to discriminate between the instances of the goal concepts and, thus, tends to cover instances of non-goal concepts.

H7 The classification accuracy of a most specific CCG is better than a description built by a most specific generalization algorithm, because the most specific generalization will exclude a large number of instances of the goal concepts.

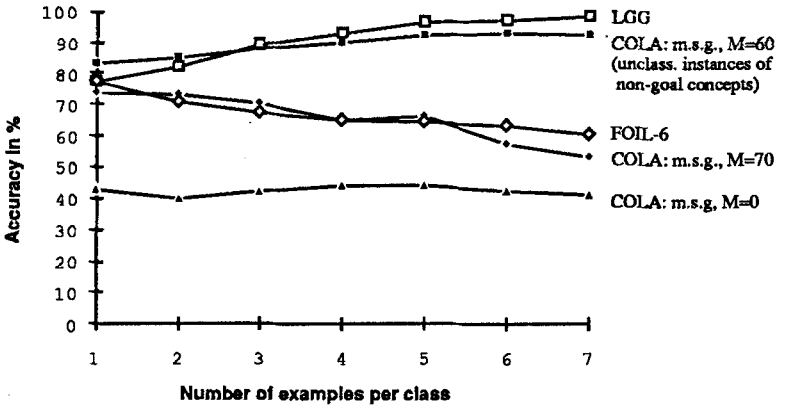

Fig. 5. Test Results in the Soybean domain with Instances of Non-Goal Concepts

Also these hypotheses were confirmed to a large degree (figure 5 and figure 6). Hypothesis 5 is confirmed by experiments in both domains. The classification accuracy of the descriptions built by the most specific CCG in the primary tumor domain (figure 6) is about 30% better than the classification accuracy of the descriptions produced by the most general CCG. A similar result could be observed in the soybean domain. In addition, the experiments have shown that the classification accuracy tends to be better, if the conceptual clustering algorithm is supplied not only with descriptions of (unclassified) instances of goal concepts but also with descriptions of instances of non-goal concepts. In addition, the classification accuracy of COLA decreases with an increasing number of examples, if only descriptions of non-goal concepts are available in the conceptual clustering step. This can be explained by the fact that KBG drops premises not necessary to discriminate between the instances of different classes.

As expected, the classification accuracy of descriptions produced by COLA is significantly better than the results produced by FOIL-6. In contrast to the experiments reported in the last section, the accuracy of the concept description built by FOIL-6 becomes not better with an increasing number of examples.

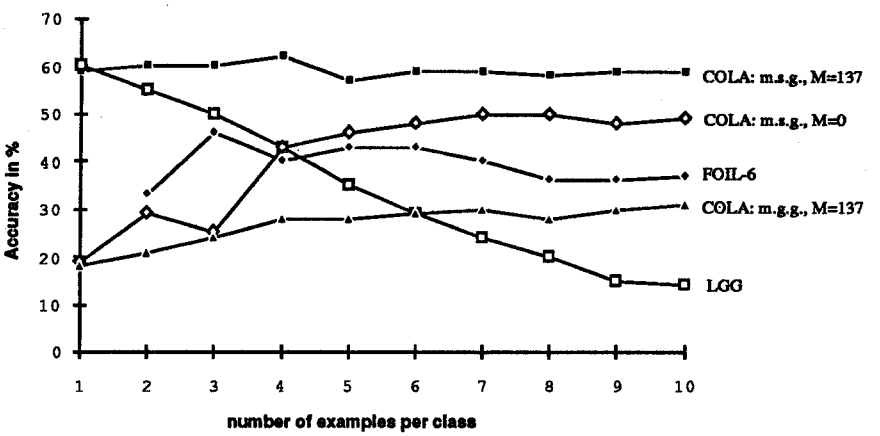

Fig. 6. Test Results in the Primary Tumor Domain with Instances of Non-Goal Concepts

Thus, hypothesis H6 is confirmed by the experiments.

The next hypothesis H7 was confirmed by the experiments in the primary tumor domain, but in the soybean domain, the LGG program performed better than COLA. As the most specific generalization excludes all instances of non-goal concepts the classification accuracy starts with 78% (1 example), all instances (of goal and non-goal concepts) were predicted as non-instances of the goal-concept. In addition, the LLG program is able to take advantage of the regularity in this domain. As the LGG program only constructs conjunctive concept description, the number of correct predicted instances increases with an increasing number of examples (without misclassification of non-goal concept instances). Note that with a decreasing number of descriptions of instances of non-goal concepts in test data set, the classification accuracy of the LLG program will decrease (while the accuracy of the FOIL-6 results will become better). In the primary tumor domain, the classification accuracy of descriptions built by the LGG program decreased with an increasing number of examples.

The conclusion which can be drawn from the second set of experiments is that the CCG approach should be prefered over existing methods, if it is known that the learning result will also be applied to descriptions of non-goal concepts. It is not reasonable to prefer a program which produces most specific generalizations, because usually it is not known that a domain is highly regular. Furthermore, it might be the case that a more sophisticated most specific generalization algorithm (e.g., able to built disjunctive concept descriptions) delivers concept description with a lower classification accuracy.

6 Related work

While there are no other approaches to learning-from-examples trying to use the information contained in unclassified examples, there are several other approaches to concept formation related to the work described in this paper. First of all, one has to mention the experiments made by Fisher using his conceptual clustering system COBWEB to learn from examples [Fisher 87, pp. 161ff]. In contrast to the approach described above, the information about concept membership was included as additional attribute-value pair in the cases presented to COBWEB. After incorporating every fifth instance, the remaining cases were classified and COBWEB predicted the value of the "goal attribute". In one experiment in the soybean domain, it turned out that a 100% correct diagnosis of cases, which were not used in the training phase, requires between 7 and 25 examples. Although COLA performs a little bit better, the result is still impressive.

Such a use of COBWEB is closely related to instance-based learning (IBL) [Aha et al. 91], although the underlying representation of concepts is very different. In both systems, prediction is performed by using the concept membership information of the most similar classified instance. Therefore, IBL and COBWEB can be expected to show a similar classification accuracy as FOIL-6.

An alternative to the use of descriptions of unclassified objects is described by Mooney [93]. His system IOU uses an overly general domain theory as additional background knowledge. The system requires that all features referenced in the domain theory are irrelevant for the induction of additional premises to specialize an over-general concept. This enables IOU to learn more accurate concepts from fewer examples than previous purely inductive systems. Although the assumption made by IOU is very strong, it seems promising to combine the use of information about unclassified objects with the use of an incomplete domain theory in order to learn concepts from small sets of examples.

7 Final remarks

In this paper, a new approach for learning concepts has been described which tries to take advantage of the information contained in unclassified examples. It has been shown that the classification accuracy of concept description can significantly be improved by using this information as additional background knowledge.

It should be emphasize that classification accuracy should not be the only criterion to compare generalization methods. The concept descriptions produced by the CCG approach are more specific than descriptions produced by systems using a most general generalization algorithm, because such systems deliver concept descriptions using only descriptors (e.g., predicates or attributes) necessary to discriminate between the instances of the goal concepts. Therefore, the descriptions produced by a system based on the CCG approach may be more comprehensible.

The computational complexity of the CCG approach is mainly determined by the computational complexity of the conceptual clustering step. Although

KBG searches only for heuristic approximations of most specific generalizations, it was impossible to cluster the whole soybean or primary tumor data set (on a SUN Sparc-10). Therefore, it is necessary to improve our method by a clever strategy for filtering the data passed to the conceptual clustering system.

Experiments with relational data sets and different hypotheses language restrictions are in progress and will be described in another paper.

Acknowledgements: I would like to thank Jörg-Uwe Kietz, Katharina Morik, Edgar Sommer, and Stefan Wrobel for interesting discussions, comments, and suggestions related to the work described in this paper. I would also like to thank Gilles Bisson for his support regarding the use of KBG. Many thanks also to the anonymous referees of this paper (and previous versions of this paper) for their helpful criticism and suggestions. Parts of this work are supported by the European Community ESPRIT program under contract number 6020 "Inductive Logic Programming".

References

[Aha et al. 91] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-Based Learning Algorithms. *Machine Learning*, 6:37–66, 1991.

[Bisson 91] Gilles Bisson. Learning of rule systems in a first order representation. Rapport de Recherche 628, LRI, University de Paris-Sud, 1991.

[Bisson 92a] Gilles Bisson. Conceptual Clustering in a First Order Logic Representation. In *ECAI92P*, pp. 558–462, 1992.

[Bisson 92b] Gilles Bisson. Learning in FOL with a Similarity Measure. In *AAAI92*, pp. 82–87. AAAI Press, 1992.

[Carey 85] Susan Carey. *Conceptual change in childhood.* MIT Press, Boston, 1985.

[Dietterich/Michalski 81] Thomas G. Dietterich and Ryszard S. Michalski. Inductive Learning of Structural Descriptions. *Artificial Intelligence*, 16:257–294, 1981.

[Fisher 87] Douglas H. Fisher. Knowledge Acquisition Via Incremental Conceptual Clustering. *Machine Learning*, 2:139–172, 1987.

[Kietz/Wrobel 92] Jörg-Uwe Kietz and Stefan Wrobel. Controlling the Complexity of Learning through Syntactic and Task-Oriented Models. In S. Muggleton (ed.), *Inductive Logic Programming*, pp. 107–126. Academic Press, 1992.

[Lebowitz 87] Michael Lebowitz. Experiments with Incremental Concept Formation: UNIMEM. *Machine Learning*, 2:103–138, 1987.

[Michalski 73] R. S. Michalski. Discovering Classification Rules by the Variable-Valued Logic System VL1. In *3rd IJCAI-73, Stanford*, 1973.

[Michalski/Stepp 83] Ryszard S. Michalski and Robert E. Stepp. Learning from Observation: Conceptual Clustering. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell (eds.), *Machine Learning*, volume I, pp. 331–363. Tioga, Palo Alto, CA, 1983.

[Mooney 93] Raymond J. Mooney. Induction Over the Unexplained: Using Overly-General Domain Theories to Aid Concept Learning. *Machine Learning*, 10:79–110, 1993.

[Morik et al. 93] Katharina Morik, Stefan Wrobel, Jörg-Uwe Kietz, and Werner Emde. *Knowledge Acquisition and Machine Learning: Theory Methods and Applications.* Academic Press, London, New York, 1993. To appear.

[Muggleton/Feng 90] Stephen Muggleton and Cao Feng. Efficient induction of logic programs. In *Proceedings of the 1st Conference on Algorithmic Learning Theory*, 1990.

[Plotkin 71] Gordon D. Plotkin. A further note on inductive generalization. In B. Meltzer and D. Michie (eds.), *Machine Intelligence*, volume 6, chapter 8, pp. 101–124. American Elsevier, 1971.

[Quinlan 83] J. Ross Quinlan. Learning Efficient Classification Procedures and Their Application to Chess End Games. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell (eds.), *Machine Learning - An Artificial Intelligence Approach*, pp. 463–482. Tioga, Palo Alto, CA, 1983.

[Quinlan 86] J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986.

[Stepp 84] R.E. Stepp. *Conjunctive Conceptual Clustering: A Methology and Experimantation.* Report uiucdcs-r-84-1189, Univ. of Illinois, Urbana, 1984.

[Vrain 90] Christel Vrain. OGUST: A system that learns using domain properties expressed as theorems. In Yves Kodratoff and Ryszard Michalski (eds.), *Machine Learning - An Artificial Intelligence Approach*, volume III, chapter 13, pp. 360–382. Morgan Kaufmann, 1990.

[Winston 75] P. H. Winston. Learning structural descriptions from examples. In P.H. Winston (ed.), *The Psychology of Computer Vision.* McGraw-Hill, 1975.